Autonomous Mobile Systems for Long-Term Operations in Spatio-Temporal Environments

Cedric Pradalier

Autonomous Mobile Systems for Long-Term Operations in Spatio-Temporal Environments

Synthèse des travaux de recherche réalisés
en vue de l'obtention de
l'Habilitation à Diriger des Recherches
de l'Institut National Polytechnique de Toulouse

Cédric PRADALIER
GeorgiaTech Lorraine – UMI 2958 GT-CNRS
2, rue Marconi
57070 METZ, France
cedric.pradalier@georgiatech-metz.fr

Soutenue le Vendredi 6 Juin 2015 au LAAS/CNRS

Membres du jury:

Francois Chaumette, INRIA, Rapporteur
Tim Barfoot, University of Toronto, Rapporteur
Henrik Christensen, Georgia Institute of Technology, Rapporteur
Olivier Simonin, INSA Lyon, Examinateur
Roland Lenain, IRSTEA, Examinateur
Florent Lamiraux, LAAS/CNRS, Examinateur
Simon Lacroix, LAAS/CNRS, Examinateur

Contents

Chapter 1

Introduction

This document reports on research conducted between 2001 and 2015 in the field of autonomous mobile robotics, specifically in what became known "field robotics": a focus of robotics on outdoor, little-structured environments close to industrial applications. Chapter 2 describes a number of research projects, starting with activities initiated during my doctorate research at INRIA between 2001 and 2004, followed by a post-doctoral fellowship at CSIRO, in Canberra and Brisbane, Australia between 2004 and 2007. From 2007 to 2012, my role as Deputy-Director of the Autonomous Systems Lab at ETH Zürich, Switzerland, gave me the opportunity to supervise a number of projects ranging from space robotics and mechatronic design to European projects on indoor navigation or autonomous driving. On the other hand, chapter 3 will describe my reseach plan stemming from this experience and preliminary results from projects started in my current position as Associate Professor at GeorgiaTech Lorraine, the French campus of the Georgia Institute of Technology, also known as GeorgiaTech, located in Atlanta, USA.

This research has been published in a number of journal publications which will support our synthesis. A selection of these publications is attached at the end of this report and will be described in more details in chapter 2:

1. Ming Liu, Cédric Pradalier, and Roland Siegwart. Visual homing from scale with an uncalibrated omnidirectional camera. *IEEE Transaction on Robotics*, (99):1–13, 2013 – Appendix A

2. Marie-Ève Garneau, Thomas Posch, Gregory Hitz, François Pomerleau, Cédric Pradalier, Roland Siegwart, and Jakob Pernthaler. Short-term displacement of planktothrix rubescens (cyanobacteria) in a pre-alpine lake observed using an autonomous sampling platform. *Limnography and Oceanography*, 58(5), 2013 – Appendix B

3. Elena Stumm, Andreas Breitenmoser, Francois Pomerleau, Cedric Pradalier, and Roland Siegwart. Tensor voting based navigation for robotic inspection of 3d surfaces using lidar point clouds. *The International Journal of Robotics Research*, 31(11), 2012 – Appendix C

4. Ambroise Krebs, Cédric Pradalier, and Roland Siegwart. Adaptive rover behavior based on online empirical evaluation: Rover–terrain interaction and near-to-far learning. *Journal of Field Robotics*, 27(2):158–180, 2010a – Appendix D

5. Cédric Pradalier, Ashley Tews, and Jonathan Roberts. Vision-based operations of a large industrial vehicle: Autonomous hot metal carrier. *Journal of Field Robotics*, 25 (4-5):243–267, 2008 – Appendix E

6. Cédric Pradalier and Kane Usher. Robust trajectory tracking for a reversing tractor trailer. *Journal of Field Robotics*, 25(6-7):378–399, 2008 – Appendix F

7. Cédric Pradalier, Jorge Hermosillo, Carla Koike, Christophe Braillon, Pierre Bessière, and Christian Laugier. The cycab: a car-like robot navigating autonomously and safely among pedestrians. *Robotics and Autonomous Systems*, 50(1):51–67, 2005 – Appendix G

Out of this body of work, three main themes can be separated: the development of navigation systems for various robotic systems in a deployment context [Liu et al., 2013, Stumm et al., 2012, Krebs et al., 2010a, Pradalier et al., 2008, Pradalier and Usher, 2008, Pradalier et al., 2005], the development of solutions to challenging perception problems in natural environments, mostly using vision [Liu et al., 2013, Krebs et al., 2010a, Pradalier et al., 2008, 2005] and the design of autonomous mobile systems for the monitoring of natural and industrial environments [Garneau et al., 2013, Hitz et al., 2012, Negre et al., 2008]. Chapter 2 will present more details about these themes and link them to demonstrate how they converge to a common research question: how to design efficient and robust autonomous mobile systems for natural and unmodified environments, while taking into account the constraints resulting from the need to deploy and evaluate these systems in the field.

From mobility and monitoring in natural unstructured system, our research is now evolving towards long-term observation of large-scale environments subjected to changes on different time scale. The environments we are considering here can be described on a multi-kilometer scale but may require centimeters of precision in their representation, at least locally (depending on the task). In these environments, change is expected to happen at scales varying from seconds to years depending on the natural processes at play. Chapter 3 will provide some insight on our current research and long-term perspectives.

Chapter 2

Autonomous mobile systems for natural and unmodified environments

2.1 Localization, navigation and control for mobile robotic systems

Designing the software architecture for mobile robotic systems involves, most of the times, the dedication of a significant share of the work to the problems of localization, navigation and multiple layers of control. This section will succinctly describe the systems we implemented on some of the robotic systems we encountered in the last 15 years, in France, Switzerland and Australia

2.1.1 CyCab: Navigation among pedestrians

Figure 2.1: The CyCab: an autonomous car-like vehicle for urban mobility

The CyCab (figure 2.1) is a small autonomous car-like vehicle. Its specificity is its ability to steer both front and rear axle, giving it an excellent maneuverability well suited for urban

environments. The work presented in Pradalier et al. [2005], Coué et al. [2006], Pradalier and Bessière [2008] was conducted at INRIA Rhone-Alpes (Grenoble) between 2001 and 2004.

Autonomous navigation with the CyCab required to first solve the problems of localization and mapping for which we presented an original approach [Pradalier and Sekhavat, 2002] based on the identification of invariant features in the measurement process. From then on, with the collaboration of J. Hermosillo, we could integrate a global planner accounting for the dual-steering capability of the vehicle and an obstacle avoidance system built on a probabilistic framework. The fully integrated systems was presented in Pradalier et al. [2005] as well as in a number of public demonstrations.

2.1.2 The Hot Metal Carrier

Figure 2.2: The Hot Metal Carrier, a modified forklift for the transport of liquid aluminium

Our work on the Hot Metal Carrier (HMC, see figure 2.2) took place at the CSIRO[1] in Brisbane, Australia, between 2004 and 2007. Its summary is presented in Pradalier et al. [2008]. For this system, the challenges were to develop a localization and navigation system, the control layers, and most importantly vision-based load handling. The latter is the main contribution in Pradalier et al. [2008]. Based on markers on the load and a vision-based state estimator, the 6-meter long vehicle could insert its hook into the lifting ring of the crucible with better than 5cm of precision and an excellent repeatability. Because of its industrial nature, this project put a particular focus on robustness and a significant effort was dedicated to the evaluation of the system in real conditions.

2.1.3 Reversing a tractor-trailer system

In parallel with the developments on the HMC at the CSIRO, we experimented with the control of a tractor-trailer system on a reversing trajectory. Initially designed as an experi-

[1] Commonwealth Science and Industrial Research Organisation

Figure 2.3: The CSIRO tractor-trailer system

ment to learn the control principles applicable to chained systems [Tilbury et al., 1995], the implementation of the controller on a real system (see figure 2.3) lead to interesting practical contributions reported in Pradalier and Usher [2008].

The first item of interest in this case was that state-of-the-art control systems for a tractor-trailer lead to overly complex model when using a real system and a chained-form decomposition. In our particular case, the required model had to consider one tractor with 4 trailers and a 10-dimensional state. The resulting control proved to be unable to deal with the non-linearities inherent to a real system: actuator lag, dead zone, actuator limits,... Finally, when using a real localization system, the estimated vehicle state is subjected to non-Gaussian noise and abrupt steps, both of which are hard to deal with the linear control law from the chained-form model. Our solution, based on nested PID controllers proved to be resilient to all the challenges above and compared favorably with human drivers on a 400m test course.

2.1.4 Magnebike

The Magnebike is a robotic system developed at the Autonomous Systems Lab at ETH Zürich, Switzerland [Tâche et al., 2009]. Its purpose is the inspection of metallic pipes such as the steam chest on the output of a coal power plant. The ASL designed the system and proposed a localization system based on the alignment of 3D laser point clouds. Our contribution, presented in Stumm et al. [2012], is focusing on the navigation system: we propose a path planner suitable for planning on non-planar manifolds (2D manifolds embedded in a 3D space) and the associated trajectory following control system.

The challenge in this planning system is that the interior of pipes forms a complext 2D manifold which cannot be projected to a planar system due to branching pipes. Furthermore, in order for the system to be self-sufficient, it should not rely on CAD-map of the pipe network (such maps are often out-of-date anyway) but should be able to create its own map out of the 3D point clouds. This generate an additional challenge because the aligned point clouds do not define a nice mesh of the surface but rather a haze of 3D measurements around the

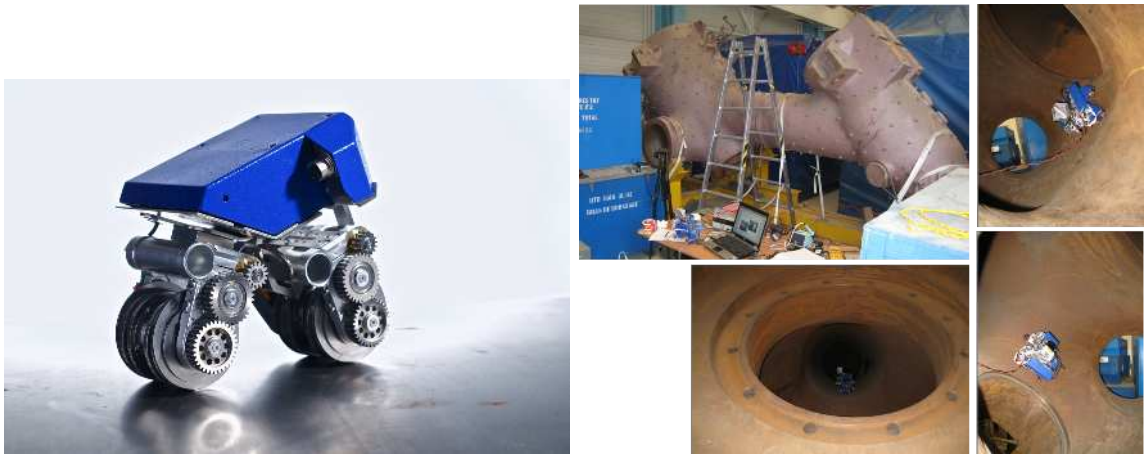


Figure 2.4: Magnebike: a robotic system to inspect metallic pipes

real surface. Rather than finding complex solutions to build an approximating mesh out of the 3D point cloud, we decided to build a planner directly on the point cloud itself. This was made possible by using tensor voting [Medioni et al., 2000] to estimate the local saliency of the pipe surface. Finally, the Magnebike is able to drive with any orientation with respect to gravity due to magnetic wheels but to maintain adherence when passing edges (e.g. entering a side tube), the planner needs to enforce arriving on such edges with a 90 degree incidence angle. With help of tensor voting, we can use a single process to not only detect the pipe surfaces but also identify edges and their saliency. Integrating this additional input led to a complete system able to plan in complex environment using only data collected on-board.

2.1.5 Rovers: adaptive planning, wheel control, odometry

Between 2007 and 2012, the Autonomous Systems Lab participated to various projects funded by the European Space Agency to design prototypes of planetary rovers and build their control systems. Planetary rovers are designed with a very specific set of challenges. First of all, up to now very little decision autonomy has been expected, so most of the research and development focused on control systems. Most of the difficulty stems from the fact that these vehicles typically have a passive suspension system, 4 to 6 wheels with individual high-torque electrical motors as well as 4 to 6 steering motors allowing steering all the wheels independently on some of the systems we designed. The challenges we addressed at the Autonomous Systems Lab included torque control of individual wheels to maximize traction in slippery conditions [Krebs et al., 2010b], using suspension deformation to reconstruct the traversed terrain shape [Strupler et al., 2014], steering control optimization to minimize kinematic errors on 6-wheeled rovers [Schwesinger et al., 2012], 3D odometry, ... Looking forward to a future where rovers will be trusted with more decision autonomy, we also built a probabilistic framework to learn the relation between remote terrain appearance and proprioceptive perceptions (in this case vibrations) and to include these predictive models in the trajectory planner to let the rover decide to avoid terrain patches where appearance hints at bad proprioceptive properties. This approach, evaluated on the CRAB rover (figure 2.5, top left) in outdoor conditions on earth (no planetary exploration mission was available for this test), has been presented in Krebs et al. [2010a].

Figure 2.5: Planetary rover prototypes developed at the Autonomous Systems Lab

2.1.6 Synthesis

All the systems presented above (which represent a selection of the systems we worked on) share a common research question: how can we build integrated navigation systems for mobile robotic systems working in real environments with real sensors? Are there elements that can be reused from one system to the next, are there aspects that must be kept specific? Between 2001 and 2014, the robotic community has matured as a whole, focusing also on these questions and proposing solutions to mutualize all the common parts valid for a wide range of systems. This also highlight a very important aspect of robotics and in particular field robotics: deployment on real robots requires a significant engineering effort, to keep the robots working, to keep their software up-to-date, to develop all the software infrastructures to interact with the sensing and command hardware. These efforts are difficult to publish but they are critical when one wants to properly evaluate solutions for robotic systems. Without a strong engineering layer, robotic systems tend to be evaluated using anecdotal experiments without statistical significance. In line with the common principle behind all our experiments, the robotic community is showing more and more awareness of this issue and promoting well designed field experiments and reference data-sets. As will be discussed in chapter 3, the next step in this direction is naturally the deployment of robotic systems with a high level of decision autonomy for long duration. This will raise new scientific problems, some of them specific to robotics, and will lead to more robust, adaptive and autonomous systems.

2.2 Challenging perception systems

The papers we discussed in the previous section were focusing on navigation and control with the aim of creating autonomous integrated systems. However, when designing complete robotic systems, perception tends to be one of the most challenging sub-system. As a result, a significant part of our work was dedicated to the design of perception systems specialized to a given task.

2.2.1 Self-similar landmarks

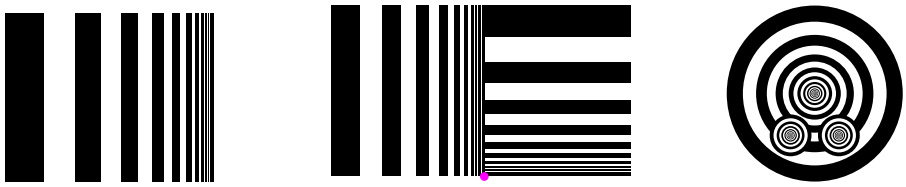


Figure 2.6: Self-similar landmarks: original from Briggs et al. [2000], orthogonal setup for load handling, circular setup for underwater docking

Figure 2.7: Experiment on self-similar landmark robustness, from left to right: original detection, non-rigid deformation, occlusions, perspective and loss of contrast.

Self-similar landmarks (seen in figure 2.6) are barcode-like patterns designed so that if a pixel at a distance r from the reference point is black, a pixel at a distance $p \cdot r$ is black as well and one at $\sqrt{p} \cdot r$ is white. This property led to a marker that can be detected independently of scale with a simple integral on a line of pixels. The original markers were presented in Briggs et al. [2000] and we adapted them in two different ways as seen in figure 2.6. On the Hot Metal Carrier, an orthogonal combination of markers was attached to the handle of the crucible because of the geometry of the support and the fact that the tracking problem could be reduced to two dimensions. For the autonomous underwater vehicle (AUV) Starbug in the context of a docking task, a large circular landmark was associated to two smaller ones. The large landmark could be detected at a large enough distance to guide the submarine to the vicinity of the landmark. At this point, the smaller landmarks became visible and let the system estimate the full pose of the vehicle with respect to the landmark frame (using an accelerometer to solve potential ambiguities). The robustness of the self-similar landmark detection can be observed in figure 2.7. The work on the Starbug AUV was published in Negre et al. [2008]. The use of the self-similar landmark is at the core of the successful operations of the Hot Metal Carrier and was published in Pradalier et al. [2008].

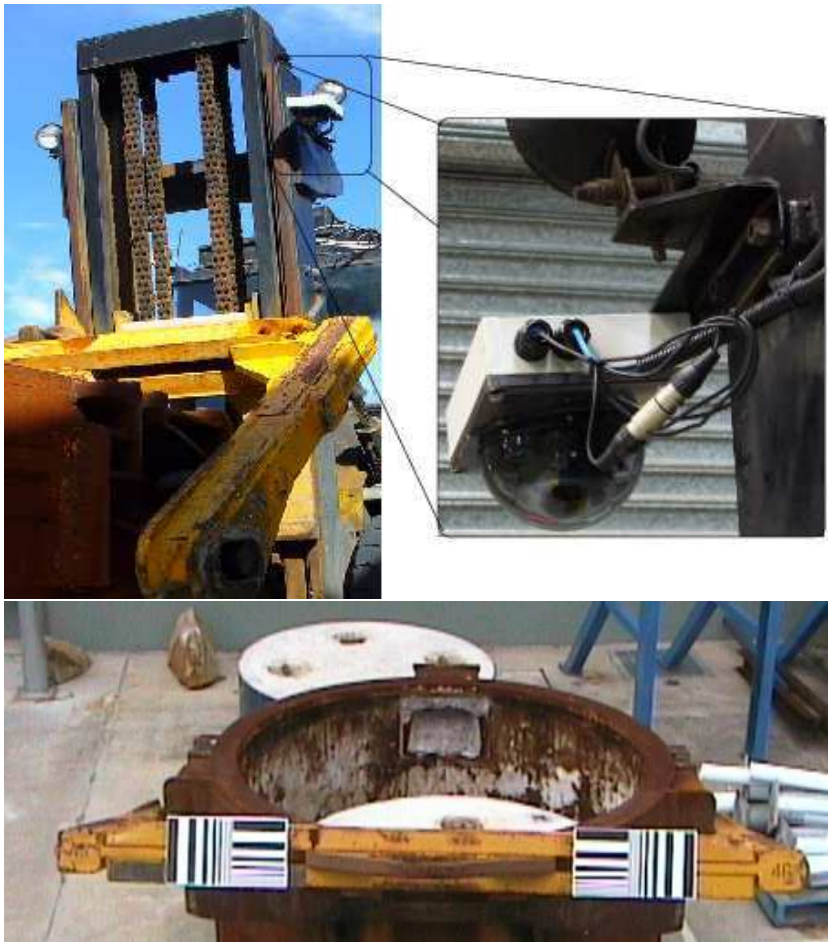


Figure 2.8: Vision setup on the hot-metal carrier. Top camera placement on top of the HMC mast, bottom camera view.

2.2.2 Vison-based manipulation

Vision-based load pickup on the Hot Metal Carrier was another of those problems where the challenges stemmed out from the specific task and the deployment constraints (see figure 2.8 and Pradalier et al. [2008]). Because of the industrial nature of these systems carrying molten aluminium, the perception system could not be placed safely at the bottom of the pickup mast where the crucible detection would have been easiest. Also, a successful pickup requires to detect the position and orientation of the crucible in real-time from a distance of 5 to 10 meters, in order for the 6 meter long HMC to align its hook with the 15cm ring on the crucible. This detection system had to be robust to different lighting conditions and able to reliably identify non detection. All these challenges led to a vision-based solution on a pan-tilt-zoom platform with artificial markers on the crucible handle (see figure 2.8). A Bayesian state estimator (particle filter) as well as the logic to cross-validate detections let us develop a very reliable and repeatable system. The evaluation shown in figure 2.9, conducted over 5 hours of continuous operations, display the trajectories of the HMC hook in 15 different pickups. At the time of the publication of these results in Pradalier et al. [2008], we did not observe anymore failures: the systems was always either able to successfully pick the load up

Figure 2.9: Performance of the vision based load pickup

or to identify that the pickup was not possible and re-initiate the pickup sequence or request assistance as appropriate.

2.2.3 Vison-based homing using scale from SIFT features

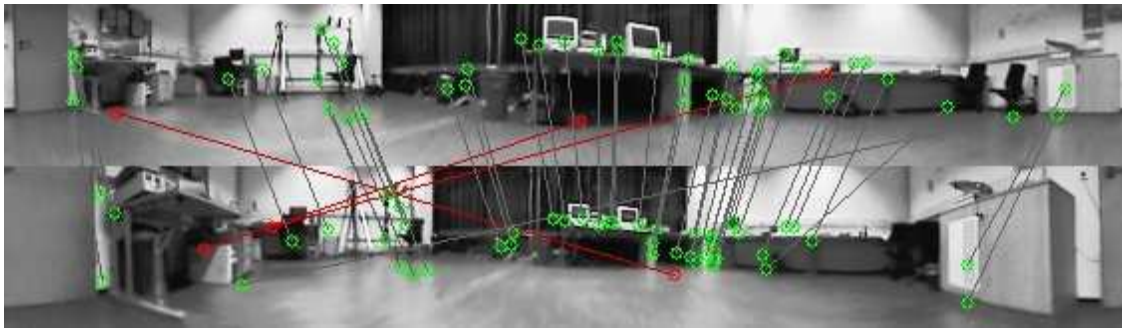


Figure 2.10: Matching process of SIFT features in omnidirectional images.

The work presented in Liu et al. [2013] started in the European project Robots@Home where the intent was to develop algorithms for navigation and in particular global localization for a companion robot moving in an apartment. The cost constraints for such a system prohibited the use of laser scanners and led to the choice of omnidirectional vision as the main sensor. By identifying that SIFT scale [Lowe, 1999] could be used as a proxy for feature distances in a visual-servoing [Espiau et al., 1992] framework, we could create a topological navigation system. In the resulting map, locations were described by their omnidirectional appearance and the associated SIFT features. Relations between places were just recorded from odometry, based on the assumption that an apartment can actually be described by a

road map of limited complexity.

The study in Liu et al. [2013] was particularly important because it proposed a detailed evaluation of the suitability of SIFT scale as a features for visual servoing. We demonstrated stability in the Lyapunov sense and made a detailed experimental validation showing the sensitivity of the approach to parameters such as matching errors, feature noise, scale uncertainty, convergence area. One of the image of the validation database can be seen in 2.10. The theoretical and detailed practical evaluation is fundamental in our approach to robotics: we strive to propose solutions to real problems with solid theoretical grounding and validate them experimentally in relevant situations showing the limits and sensitivity of the approach.

2.2.4 Sonar-based navigation

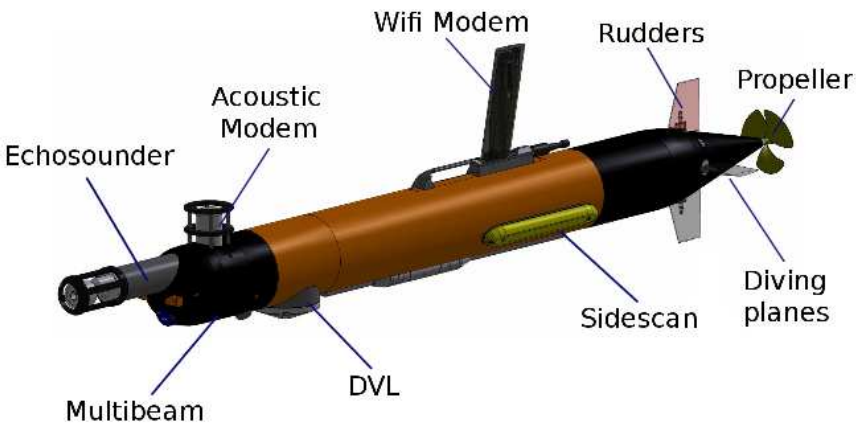


Figure 2.11: The Noptilus Autonomous Underwater Vehicle (AUV)

Following up on the sensory-motor trajectories tried on the CyCab [Pradalier and Bessière, 2008], we are currently participating to the European project Noptilus where our task is to realize sensory-motor trajectory following underwater on Autonomous Underwater Vehicles (AUVs) – see figure 2.11 – using a variety of sonar sensors: single-beam echo sounders, Doppler velocity loggers (DVL), multi-beam sounders and side-scan sonars. Single-beam and DVL have already been used for tasks such as global localization with respect to a known bathymetric map in Bayesian framework as shown in figure 2.12. However, for sensory-motor replay, a sensor with a wider field of view such as multi-beam sounders and side-scan sonars will be required. These sensors are particularly interesting from a research perspective since there have been very little effort on integrating them in a control loop onboard AUVs even though most modern system will actually embed them in their standard sensor suites. Side-scan sonar are particularly interesting because a significant modelling work is required to exploit the wealth of information they report, i.e. sonar energy received as a function of time instead of simpler ranges reported by other sensors.

At this stage, limited results have been published on this project [Michalec and Pradalier, 2014] but more publications are expected in 2015. However, this project is also representative of an aspect of our research line consisting in exploring the use of informative sensors in field applications, using real data, real robots and real environment to solve problems relevant to practical applications.

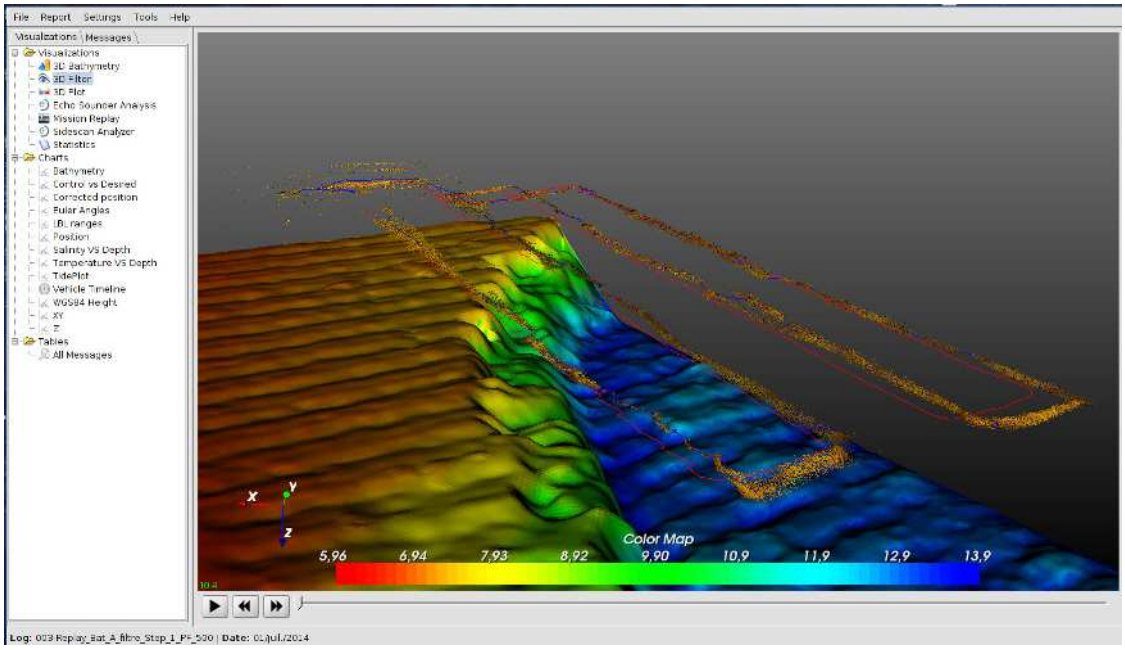


Figure 2.12: Terrain-based localization using a multi-beam sounder and a particle filter on the Noptilus AUV. The bronze cloud around the trajectory is the accumulated particle position over the whole trajectory. The 3D elevation map was realized by processing multibeam echosounder data acquired on a survey mission.

2.3 Systems for environment observation

Out of all the robotic systems we've discussed so far, a common target application can be seen emerging: the monitoring of complex environments, natural or industrial. On the one hand, inspection is one of the few target application for civilian robotics where the business model makes sense with the current state of the art. On the other hand, environment science is currently awakening to the potential of automation and robotics for monitoring and sampling applications. In the following sections, we will discuss several research projects we conducted in this context of environment observation. Most of these projects have a significant mechatronic design component and have been published as such. However, we will exclude the planetary rover we already discussed in section 2.1.5: these systems are ultimately designed for environment observation tasks, but our work was just too far ahead of the target application to consider more than the basic control challenges.

2.3.1 Industrial inspection

Between 2007 and 2012, at ETH Zürich, we developed a number of systems focused on industrial inspection. I will only mention here the one where I could make a technical contribution instead of just taking care of the project management. As mentioned earlier, the Magnebike [Stumm et al., 2012] is the archetype of the inspection robot system. It is designed specifically for the inspection of metallic pipes in power plants and its design cost is negligible in comparison with the loss of income resulting from a day off the grid of such a power plant. Our main contribution on this system was the development of the navigation system as well

as a significant contribution to the low-level embedded system. In a similar vein, at the be-

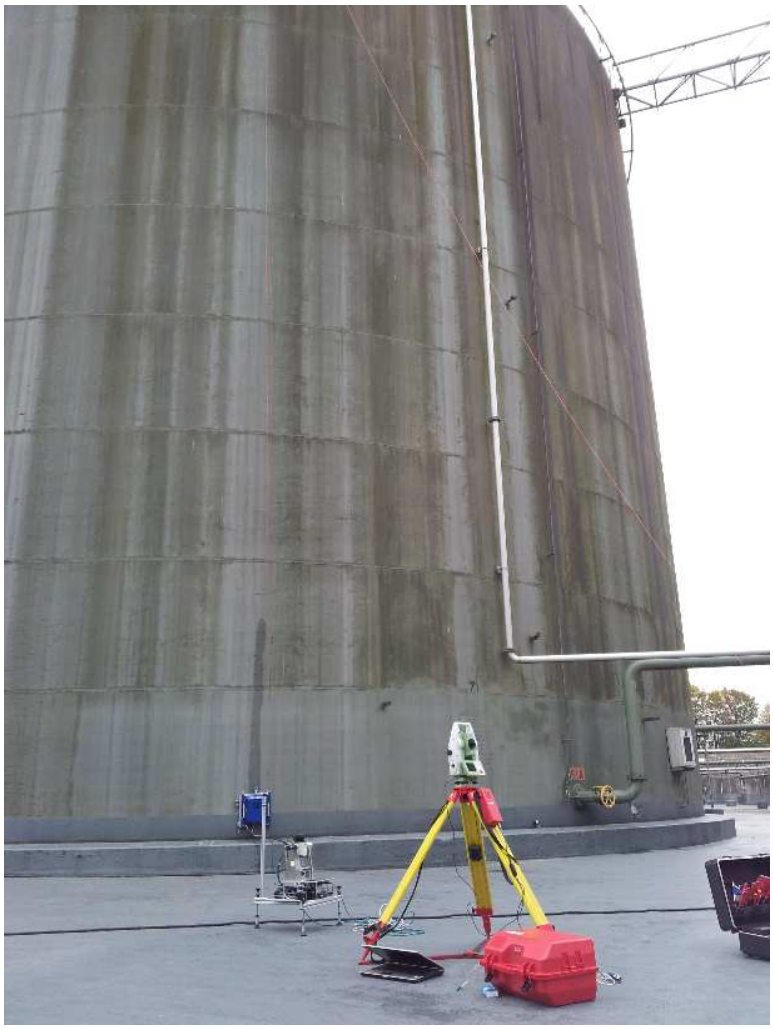


Figure 2.13: Example of storage tank targeted by the KTI project on robotic inspection

ginning of 2012, the Autonomous Systems Lab and the company Alstom Inspection Robotics started a project funded by Swiss Chamber of Industry (KTI) under my supervision on the automation of the inspection of storage tanks, and in particular oil and gas tanks. This is one application where the European regulation is becoming more and more stringent – every side panel of the tank needs to be inspected on 5 points using non-destructive method based mostly on Eddy currents and ultra-sounds. The current state of the art is for a person to abseil from the top of tanks similar to the ones in figure 2.13 while carrying the inspection device. While this is rather a technology transfer project than a scientific contribution, this project is interesting because it has the potential to significantly change industry standards and the safety associated with storage tanks.

2.3.2 Robots for natural environment monitoring

Natural environment autonomous monitoring raises a lot of interesting research challenges for robotics. The first one stems from the design of robotic systems, from the mechanical and electrical design to the software architecture. The second and third challenges are mobility and perception in unstructured and natural environment. Without salient lines and corners, a lot of the approaches presented in the literature have limited success in a forest or on a lake shore. Working 5 years at the Autonomous Systems Lab, within the mechanical

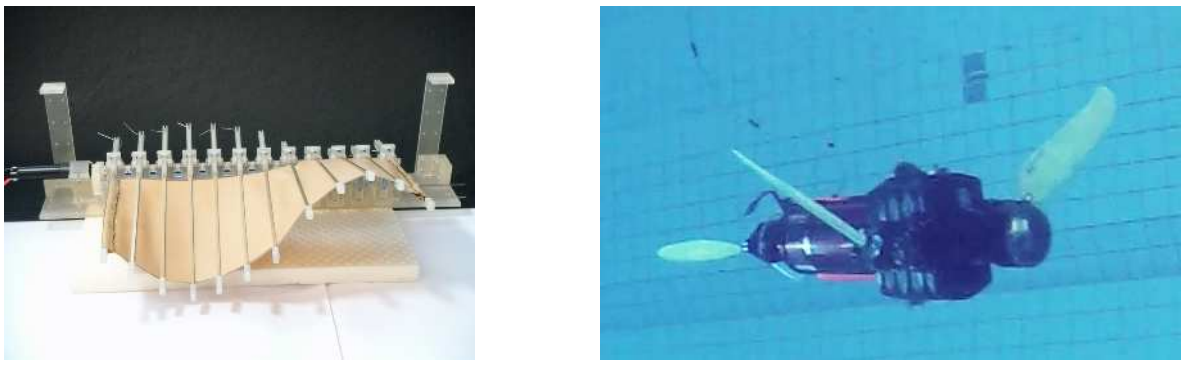


Figure 2.14: Bio-inspired mechatronic design projects: a membrane-based underwater locomotion system inspire by a cuttlefish (left) and underwater system with a kinematic comparable with a sea-turtle (right).

Figure 2.15: Autonomous boat designed at the Autonomous Systems Lab: Avalon (left) and Lizhbeth (right) used in the Limnobotics project (www.limnobotics.ch).

engineering department of ETH Zürich, provided a strong incentive to address the challenges of mechatronic design for autonomous mobility in natural environment. Section 2.1.5 (on planetary rovers) shows some examples of systems designed for mobility in rock-and-sand environments. In parallel with these works for the European Space Agency, we could conduct mechatronic design projects up to the prototype stages. Some of them such as a concept study on membrane-based locomotion underwater (similar to cuttlefish)[Peter et al., 2010]

Figure 2.16: Autonomous vehicle specifically designed for environment monitoring: the Starbug AUV from CSIRO (left) and the Kingfisher ASV used at GeorgiaTech Lorraine (right).

or a longer study on the design of an underwater system with a mobility similar to a sea-turtle [Siegenthaler et al., 2013], were mostly focusing on the mechatronic challenges and did not reach the autonomy stage under my watch. Both projects can be seen in figure 2.14.

A number of projects at ASL focused on the design of autonomous boats, two of which involved me technically (see figure 2.15). The first one consisted in the design of the autonomous sail boat Avalon [Erckens et al., 2010]: a system designed from the beginning for long-term autonomy both in terms of energy and decision. This boat was designed and completed during the course of year with a couple of further projects related to software autonomy. Experiments in autonomy over 70 nautical miles were conducted around Toulon, France, only interrupted by a mechanical failure. A second project started with the realization of an electric-powered catamaran for the monitoring of algal blooms in Lake Zürich. This work was later funded by the Swiss National Fund (1 PhD and 1 Post-Doc) in the project Limnobotics, which I prepared and co-supervised. It demonstrated the non-heterogeneous nature of algal bloom distributions and their dynamics, which was published by Garneau et al. [2013]. The mechatronic design focused on the stability and deployability of the boat on fresh-water lake, while allowing to reliably monitor the first 20 meters of the water column. In this project, the system was built from the beginning for autonomous navigation, which allowed precise and repeatable transects in Lake Zürich and Lake Cadagno, a small lake at an altitude of 2000m in the Swiss Alps.

Finally, I also had the opportunity to contribute to the autonomy layer of systems designed from the outset for environment monitoring. Specifically, at CSIRO in Brisbane, Australia, in collaboration with M. Dunbabbin, we developed a number of experiments on the Starbug AUV (see figure 2.16). The contribution there was on one hand, the implementation of a task scheduling system (also used on the Hot Metal Carrier, section 2.1.2) and on the other hand, the contribution of a robust vision-based docking approach presented in Negre et al. [2008]. These applications were deployed in the field happened after I left CSIRO but these tools inspired the ones we developed for the Limnobotics project [Hitz et al., 2012, Garneau et al., 2013]. The continuity of this experience also permitted the fast development of the autonomy layer for the Kingfisher (see figure 2.16), which is at the core of our current projects towards long-term natural environment monitoring. Preliminary results on this platform have been presented in Griffith et al. [2014].

Chapter 3

Perspectives: Towards observation and operations in spatio-temporal environments

The previous chapters provided an overview of my research path up to the beginning of 2015. The next sections are now looking towards the future and I will discuss the directions my projects are currently following as well as the continuity with the last fifteen years of research I was involved in.

The core component of my research interests is the autonomous observation of environments in which change happens at a variety of scales, from seconds to years, some changes being the result of abrupt events, others resulting from smooth variations of parameters. Some changes will also be recurrent while others will happen only once in the observation period. At another level, changes can be spatially localized or happen everywhere in the environment, simultaneously or not. I qualify these environments as spatio-temporal environments: to be understood they must be explored through time and space.

Monitoring spatio-temporal environments will require the development of spatio-temporal models describing the structure of the environment itself and the dynamics of phenomena occurring therein. Using an autonomous platform will let us perform a precise and repeatable monitoring, sometimes with less supervision requirements. In a later stage, we also want to close the loop and take advantage of autonomy to not only detect changes but also inspect them in more details. This raises further interesting questions in fields such as autonomous sampling or information-aware planning. Integrating such decision layers on-board, within the autonomy framework of a robotic machine will also be a significant engineering feat out of which practical contributions are expected.

3.1 Applications

Environment Sciences: As a field roboticist, my research objectives are driven first by a set of applications which exhibit the problems I am interested in. The problem of long-term environment monitoring applies naturally to a large range of studies in environmental sciences, such as limnology, planetary exploration, oceanography, ethology, etc. To identify these application fields in environment sciences, I am currently involved in the Zone Atelier Moselle, a confederation of research laboratories interested in environment sciences around the

Moselle watershed: from agrology and geology to bio-chemistry and toxicology. Research in these fields relies on a lot of data sources but little use have been made so far of automation. This is partially due to the cost of autonomous systems and the scarcity of embeddable sensors suitable for these problems. Current projects to rehabilitate fallen industrial sites in the Lorraine area could also support research on long-term monitoring to document and evaluate the remediation actions.

Agriculture: Agriculture is another field in which autonomous long-term monitoring and operations would make a lot of sense. A field where a crop is grown is naturally changing over time, from ploughing and seeding to harvest. Most of these changes are what is expected out of the natural growth dynamic of the plant of interest, but some changes can be the results of disease, lack of input (water, fertilizers) or other damages. Regular and precise monitoring, as well as remediation actions, will be the topic of the Flourish project, a project funded by the European Commission in which the UMI 2958 GeorgiaTech-CNRS will take part from March 2015 to September 2018. Half of our contribution to the project will be focused on precise remediation, but a contribution to the perception and planning layers is also to be expected.

Inspection: As mentioned above, inspection is also a natural application field for long-term monitoring. Such monitoring can take a variety of form, from natural environment to human-made infra-structures (bridges, canals) and industrial sites. Developing these projects will require the establishment of industrial collaborations with end-users or equipment manufacturers. Participation to challenges such as the Argos challenge [arg] will also be considered when appropriate[1]. The objective of the Argos challenge was the development of an autonomous system for the monitoring and inspection of off-shore infrastructures. The development of this direction of research will also be supported by existing cross-disciplinary research within the UMI 2958 GeorgiaTech-CNRS and GeorgiaTech Lorraine which also hosts specialists in acoustics and non-destructive testing, material science and simulation of defects propagation in anisotropic materials.

The themes listed above do not intend to be exhaustive but it definitely covers a significant part of the practical questions which are going to inspire our research in the upcoming years.

3.2 Challenges and Research Directions

Monitoring spatio-temporal environments will raise challenges at a number of level. First, the core of the required research will focus on environment perception challenges in natural or uncontrolled environments. This includes the development of spatio-temporal models, matching and data association in changing environments, detection and classification of changes. Secondly, combined with this understanding of spatio-temporal environments, robotic systems have to take advantage of their mobility to take autonomous decisions. This will lead to challenges in the field of planning, control and mobility, such as information-aware planning, multi-resolution observations or remediation in the context of a agricultural robot. Finally, a given observation and inspection task will most likely require the development of a specialized platform with proper mobility and a suitable sensor payload.

[1]Participation to the Argos challenge was denied due to legal and IP issues

Although not a research objective in itself, the engineering challenges related to the development of such platforms is essential to the ability to observe real environments and often provide interesting collaboration opportunities with industrial partners.

3.2.1 Environment Perception

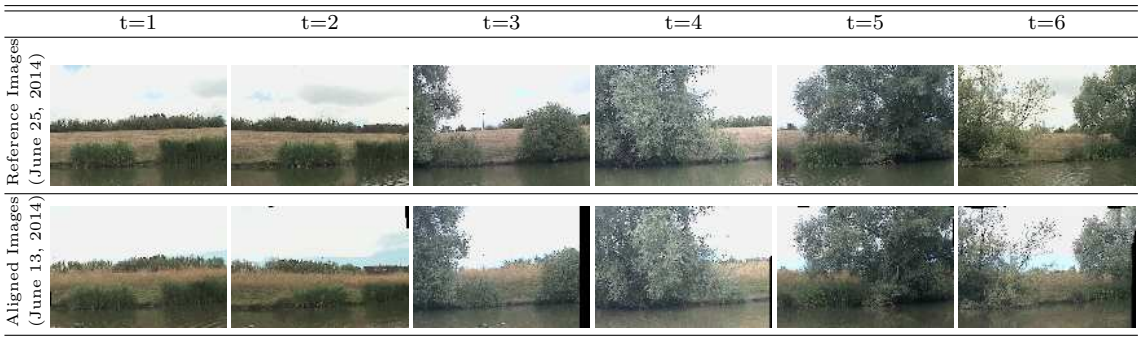


Figure 3.1: The aligned and the reference images for six consecutive places along lake Symphonie's shore where the June 25, 2014 survey was compared to the June 13, 2014 survey. The mowed grass is one of the most salient, non-natural changes between them, which appears in all six.

Monitoring an environment over space and time generates a tremendous quantity of data, in particular when using digital imagery. As an example (depicted in figure 3.1), we are currently conducting a weekly monitoring of a small lake next to the UMI 2958 GT-CNRS in Metz, France, and even with the low resolution images used, this represents between 4 and 8 GB of compressed data per week to describe close to 1 km of shore line. Figure 3.1 depicts two aligned image sequences from these data-set, captured within a twelve-day interval. The first challenge to be addressed is to build three dimensional spatio-temporal maps of the environment being monitored. This will include some form of 3D representation extended with an appropriate representation of the observed dynamic of the observed changes. At this stage, it is unclear what an optimal representation is, but an avenue to explore could be an extension of n-dimensional Haar-wavelets describing both the 3D volume, and maybe its appearance, over time.

From a computer science perspective, the development of data-structures allowing an efficient representation of spatio-temporal environments, while still being usable in real-time applications and sufficiently easy to update is a significant challenge that will have to be addressed at the same time as the choice of the representation.

One of the specific challenges resulting from the observation of a natural environment can be seen in figure 3.2: natural environment are subjected to significant changes in lighting conditions over time and this in turns makes them appear significantly different. This has also been reported in Churchill and Newman [2013]. This is a challenge because monitoring an environment over time requires associating a view acquired at a first moment with a view acquired later, even though most of the environment appearance has changed in between. We must hence separate the structural changes in the environment with the changes due to variation in illumination. Our first experiments on this topic show that it will require rethinking data-association and matching techniques for natural environments. For instance, the ubiquitous SIFT features [Lowe, 1999] does not perform well in scenes consisting mostly

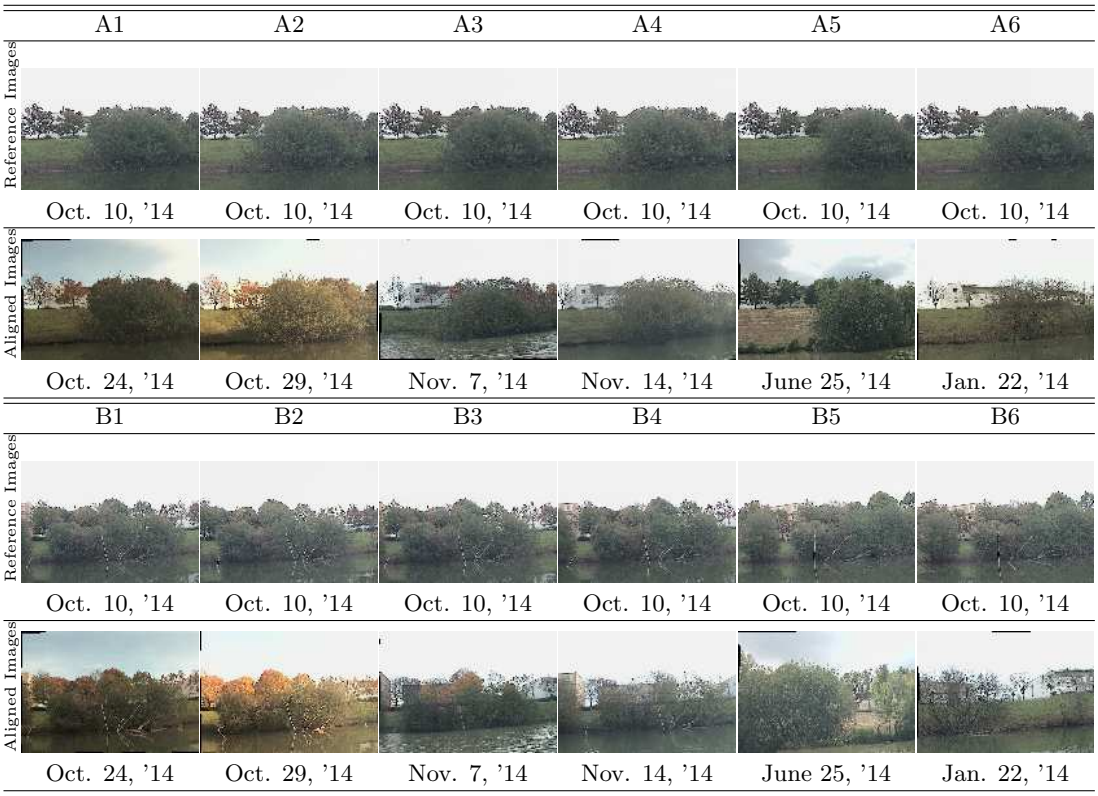


Figure 3.2: Surveys captured at various times are aligned to the survey from Oct. 10, 2014 at two different sections of lake Symphonie's shore, denoted by A1-A6 and B1-B6. Each warped image is shown with the reference image from the Oct 10 survey that it tried to align with.

of natural features such as shrubberies, trees and grass. This is probably due to the low distinctiveness of objects, appearances and key-points in these environments. Note, that in our preliminary experiments, other features were not performing significantly better in our test data-sets.

Another important challenge to address will be the detection and classification of changes. As mentioned earlier, we are interested in environments where change happens at the scale of seconds to years, with abrupt changes and smooth ones. This will require developing new approaches able to detect changes in large spatio-temporal data-sets and hopefully classify them, at least with respect to their temporal properties. Ultimately, it would be particularly interesting to distinguish expected changes from unexpected ones. In effect, an expected change such as leaves falling down in fall would just be an expected observation of a spatio-temporal representation encoding this particular dynamic in a known environment. This event which would be a change in a map representing a static world would actually be a non-event in a spatio-temporal map.

Finally, monitoring large scale environments over long duration will probably require conducting the observation with an heterogeneous set of sensors and in particular with different resolutions. These monitoring runs may also not be conducted synchronously. This will raise further challenges linked with the management and fusion of heterogeneous data from multiple spatio-temporal resolutions. To address these type of issues approaches using a continuous time assumption as is currently popular in the SLAM community [Furgale et al., 2012, Oth

et al., 2013] could be explored. A continuous representations of the spatio-temporal map as a n-dimensional continuous function with local support could be updated with data at any spatio-temporal resolution by just sampling it at the space-time point of the observation.

3.2.2 Planning, Control and Mobility

Monitoring spatio-temporal environments can be done using hand-held sensors, or sensors mounted on manually driven vehicles. However, using autonomous systems offers certain advantages which naturally come with interesting research challenges. The main advantage of autonomous systems for such a monitoring task is the precision and repeatability of the observation patterns, as noted in Garneau et al. [2013], combined with the potentially low signature and reduced human involvement. The first challenge we consider is the optimization of observation trajectories for the surveillance and monitoring of natural or unmodified spatio-temporal environments. Because these environments change over time, they raise a new definition of the coverage problem[2]: the only real coverage solution is to observe the entire environment simultaneously and continuously. This covers both the spatial and temporal dimension of the place of interest. Obviously, such a solution is not applicable to autonomous systems with a limited perception footprint that cannot be everywhere at the same time. New approaches will have to be developed to find locally optimal observation solutions that take advantage of the part of the spatio-temporal model already known. Note that until enough observation has been made to start such a model, the coverage problem is necessarily reduced to its spatial dimension.

In a similar vein, using a spatio-temporal model of the environment could allow a focused multi-resolution observation strategies: the autonomous system could gather long-range observations of the environment and compare them with its predictions and then only close-up to collect more imagery from places where unexpected changes have been detected. This type of planning is similar to information-aware planning [Hitz et al., 2014]. It could be further extended if we consider that the autonomous vehicles are typically limited in autonomy by their power source. In this condition we could envision planning multi-resolution monitoring paths under the constraint of fixed/maximum observation duration.

Predicting and recognizing environment change, especially its appearance, could also be expanded towards predicting environment properties following the lead of Krebs et al. [2010a]. In a riverine environment, this could allow use to predict changes in water eddies which in turn could affect the feasibility or optimality of paths. In a ground environment, change of appearance of the ground could be linked (possibly from experience) to the changes in expected mobility conditions: hard ground turning to mud, wet grass turning slippery, icy road surfaces, etc... Similarly, combining inspiration by the work of Churchill and Newman [2013] with our work on sensory-motor trajectories [Pradalier et al., 2005], it would be interesting to learn sensory-motor trajectories in a spatio-temporal environment and use the spatio-temporal model of this environment to make such trajectories resilient to slow evolutions of environment properties.

Finally, in an agricultural setting, the Flourish project [flo] depicted in 3.3, starting in March 2015 will address the remediation aspect of spatio-temporal environment monitoring. In a field, the seeded crop hopefully grows continuously while weeds may grow asynchronously. By monitoring the changes and identifying growing weeds, an autonomous farming robot

[2] the choice of a trajectory that completely observe a given environment

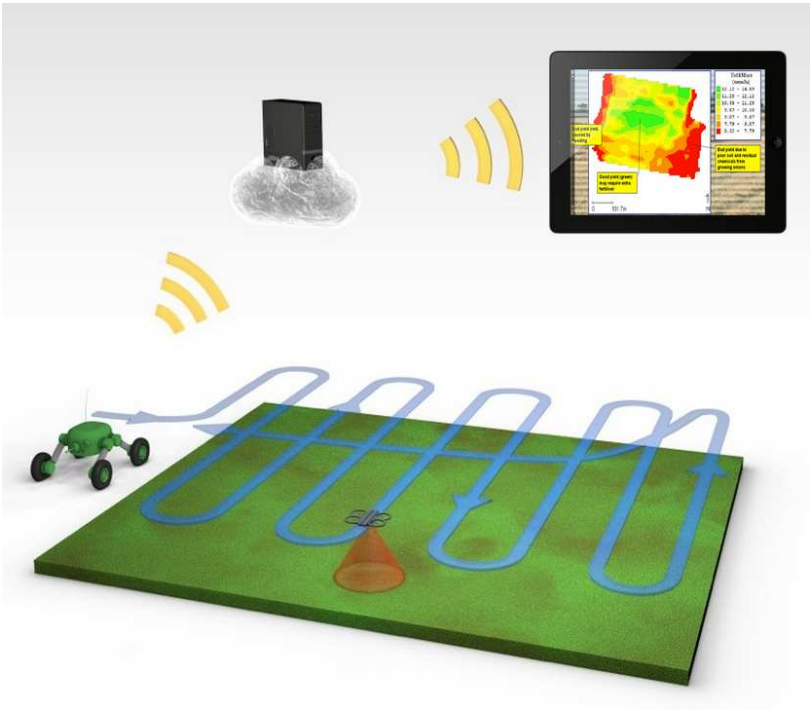


Figure 3.3: Concept drawing of the heterogeneous system planned for the European project Flourish (courtesy of the Flourish consortium): a ground robot and a small aerial vehicle work together to monitor and maintain a field.

could intervene and extract or destroy weeds. This will require precise localization of changes and precise control to remove the identified weeds potentially in harsh field conditions while meeting strong efficiency requirements.

3.3 Initial Results

This section presents first results obtained in the direction of this research plan and submitted to international conferences at the time of this writing. This work has been conducted mostly by S. Griffith, PhD student at the Georgia Institute of Technology under my supervision since September 2013.

So far, we have been developing an autonomous monitoring framework to assist a human trying to detect changes in lakeshore environments. This is a large spatial and temporal scale study, involving over 50 surveys of a lakeshore collected over a year and a half with an autonomous surface vehicle. In spite of the large variation of appearance across surveys, our framework provides a human with aligned images and a way to quickly detect changes between them. First visual SLAM [Košecka, 2013, Beall and Dellaert, 2014, He et al., 2006, Agarwal et al., 2011] is used to find a coarse alignment of images between surveys, and second, SIFT Flow [Liu et al., 2011] is applied to achieve dense correspondence. We present aligned images to a human using the flickering paradigm, which a human can quickly use to perform change detection. Results show that our method can align images in the midst of variation in appearance of the sky, the water, changes in objects on a lakeshore, and the seasonal changes

of plants. Initial results from our approach can be seen in figure 3.1 and 3.2

The problem of finding dense correspondences between images from two different surveys of a natural environment is similar to related work on synchronizing image sequences. Evangelidis and Bauckhage [2013] synchronize two image sequences by maximizing the number of matching feature points between image pairs, and then refining the estimate based on image similarity. A database is made for the first sequence, which is queried with images from a new sequence to get a visually close frame. The result is refined using a 2D homography to find the best corresponding image. Based on their algorithm's performance to major changes in illumination between video sequences, they recommend a different approach if sequences have lots of variation between them. In a different approach to synchronization based on time–warping, Wang et al. [2014] give a human the power to explore the space of possible synchronizations through an interactive system.

In this report, image sequences from consecutive surveys are aligned to assist a human with identifying changes. The aspect of our work regarding assisting humans perhaps makes it most similar to that of Williams et al. [2012], in which sea-bottom imagery from an autonomous underwater vehicle is annually captured over four years. They perform visual SLAM and dense 3D reconstruction from stereo images to create a data exploration tool, which a human uses to visually compare separate datasets. In contrast, in our work a human compares directly aligned images, which exploits human skill at detecting changes between flickering images of the same scene. Rather than try to match image patches or sparse features, we align whole images and achieve dense correspondence on the order of a pixel–level alignment between images of the same scene captured week(s) apart.

3.3.1 Experimental Setup

Robot

We used Clearpath's Kingfisher autonomous surface vessel (ASV) for our experiments (depicted in 2.16). It is about 1 meter long and 2/3 meters wide, with two pontoons, a water-tight compartment to house electronics, and an area on top for sensors and the battery. It is propelled by a water jet in each of its pontoons, which can turn it by differential steering. It can reach a top speed of about 2 m/s, but we mostly operated it at lower speeds to maximize battery life, which is about an hour with our current payload.

Our Kingfisher is outfitted with several sensors befitting an autonomous surface vehicle. A prominent 704x480 color pan-tilt camera stands on top, capturing images at 10 frames per second. Beneath it sits a single scan line laser-range finder with a field of view of about 270 degrees. It is pointed just above the surface of the water and provides a distance estimate for everything less than 20m away. The watertight compartment houses a GPS, a compass, and an IMU.

Environment

The ASV was deployed on Lake Symphony, Metz, France, a lake about 400 meters long and 200 meters wide with an 80-meter-wide island in the middle. The nature of the lakeshore varied, with shrubs, trees, boulders, grass, sand, buildings, birds, and people in the immediate surroundings. People mostly kept to the walking trail and a bike path a few meters from the shore, and fishermen occasionally sat along the shore.

Behavior

We used a simple set of behaviors to autonomously steer the robot around the perimeter of the lake and the island. As the boat moves at a constant velocity of about 0.4m/s, a local planner chooses among a set of state lattice motion primitives to keep the boat 10m away from the lakeshore on its starboard side. Passing through a predefined waypoint along the perimeter marks the transition to surveying the island. At this point the pan-tilt camera is rotated from pointing starboard–side to pointing port–side, and the planner begins optimizing the boat's proximity to land on its port side. With this configuration, the robot is capable of performing an entire survey autonomously; however, we occasionally took control using a remote control in order to avoid fishing lines, debris, to swap batteries, or to fix unexpected control failures (due to software errors, a sharp turn–around point, or the shore line being behind the first row of trees in a flood).

We deployed the robot up to once per week over a period of 1.5 years, which began August 18, 2013. This report analyzes data from eight different surveys, which cut across 11 months of variation. Each survey was performed in the daytime on a weekday in sunny or cloudy weather, at various times of the day. They usually consisted of one complete run around the entire lakeshore, including the island. In some cases a survey had to be cut short, resulting in a partial run of the lakeshore.

Experimental Variation Several factors over the lifetime of the experiment added variation to the survey data, including changes with the boat, the maturity of the software, our knowledge of the system, etc., The boat's first pump-jets lasted approximately 25 hours, which meant that we had to replace them after around 30 surveys. Because the pontoons, the pump-jets, and the batteries were redesigned, the Kingfisher's configuration changed slightly after we swapped them.

3.3.2 Methodology

Enabling interactive natural scene monitoring between surveys consists of a two–step coarse-to-fine alignment process, in which first the rough location where each image was captured is identified, followed by second, a pixel-wise alignment is computed for images of the same scene from two different surveys. The first step consists of visual SLAM. The second step involves image registration.

Data Collection

A single survey represents a collection of image sequences, measurements of the camera pose, and other useful information about the robot's movement. During a survey, k, the robot acquires the tuple $\mathcal{A}^k=\{\mathcal{T}_i^k, \mathcal{I}_i^k, \hat{C}_i^k, \omega_i^k\}_{i=1}^{|\mathcal{A}^k|}$ every tenth of a second, where $\mathcal{T}$ is the current time, $\mathcal{I}$ is the image from the pan-tilt camera, $\hat{C} \in \mathrm{SE}(3)$ is the estimated camera pose, and ω is the boat's angular velocity as measured from its IMU. The estimated camera pose is derived from the boat's GPS position, the measured heading from the compass, and the pan and tilt positions of the camera. Each survey is down–sampled by a factor of five to reduce data redundancy and speed up computation time.

Visual SLAM

Finding nearby images in two long surveys is possible using raw measurements of the camera pose, but because these measurements are prone to noise that could lead to trying to align images of two different scenes, we use a visual slam framework to improve our estimates of the camera positions.

Visual Feature Extraction We used generic feature tracking for Visual SLAM, which is based on detecting 300 Harris corner features and then tracking them using the pyramidal Lucas–Kanade Optical Flow algorithm (from OpenCV) as the boat moves. To ensure a uniform distribution of visual features in a scene and to reduce the search time for optical flow, images are divided into a 12x20 grid and new features are detected and tracked only if they are located in an empty cell. Lucas–Kanade optical flow tracks each feature by searching in a neighborhood of cells for the match that optimizes a brightness constraint and a smoothness assumption. Features are removed from tracking if no suitable match is found, if the displacement is too different from the boat's movement, if too many features are clustered together, or if its positions violate epipolar constraints. The result of this process is a set of visual feature tracks $F = f_1, ..., f_z$, where $f_j = m^j_{t_{first}}, ..., m^j_{t_{last}}$ and each $m^j_t \in \mathbb{N}^2$ defines a 2D pixel coordinate observed at time t.

Optimization We apply a graph–based SLAM approach for optimizing the camera poses and the visual feature locations. A factor graph is used to represent the set of measurements of the camera poses and the landmark positions, and the different constraints between them. The GTSAM bundle adjustment framework is applied to the factor graph to reduce the error in the initial estimates of the positions [Dellaert, 2012].

A factor graph is constructed with the measured camera poses, $\hat{C}^k = p_1, ..., p_n$, where $n = |\mathcal{A}^k|/5$, and the visual feature tracks, $F = f_1, ..., f_z$, the robot acquires as it moves through its environment. At each time step, t, a node, x_t, is created to represent the camera pose with p_t being a position prior. If any new visual features are tracked, a new node, l_j, is created for each one. The node for the camera pose is connected by an edge to the node of each observed visual feature, which represents the visual constraint that the 3D position of the visual feature matches its observed 2D pixel location, m^j_t. An edge also connects the camera node to the one for the previous time step, which constrains the two positions by the kinematic motion model of the robot.

Usually, the robot kinematic constraint between two camera poses is directly observed using odometry measurements. However, because our ASV is propelled by jet thrusters, rather than wheels, typical odometry readings are lacking. As a result, the factor graph is a bit more complicated in order to adequately constrain the degree of movement between any two camera poses. Because two consecutive poses are related by a relatively constant velocity and a relatively constant angular velocity of the boat, which is directly observable, an additional node, v_t, is created at each time step to represent the velocity of the robot.

Whereas nodes in the factor graph represent the variables to be optimized, factors in the factor graph describe the constraints on each variable. For each direct measurement of a variable (e.g., the compass directly measures the boat heading) a factor is used to limit the value of the variable to within the acceptable variance of that measurement (e.g., the standard deviation of the compass measurement is estimated as 10 degrees due to possible distortions of the magnetic field caused by peak currents in the motors). A factor, u_t, is used

between consecutive pose nodes and the previous velocity node to describe the new position as a function of the previous one, the previous velocity, and some deviation. Consecutive velocity nodes are related by a factor, r_t, which forces the velocity to change slowly between each time step. A prior constraint, q_t, is defined on the value of the velocity. Finally, an observed pixel location, m_t^j, of the visual feature, l_j, is defined as a factor which constrains the 3D position of the visual feature.

Because several noisy sensor readings are combined to describe the value of each variable, any estimate of a variable's true value is going to differ from its observations. The level of expected divergence is manually specified through the expected standard deviation associated with each constraint. Accordingly, given an estimate of the values of all the variables in the factor graph, a heuristic estimate of the total error is computed. The goal of optimization is to find values for the variables that minimize this error.

We optimize the estimated positions according to the constraints in the factor graph using the iSAM2 bundle adjustment framework [Kaess et al., 2012]. iSAM2 makes incremental bundle adjustment computationally feasible by converting the factor graph into a Bayes tree. A Bayes tree data structure hierarchically organizes variables, which reduces the size of the optimization problem by isolating the variables with large amounts of error. The Levenberg–Marquardt algorithm is used to perform the optimization step, which incrementally refines initial estimates of the variables over several iterations.

Because optimizing a factor graph of a large scale environment can require a significant amount of computation time, we also make use of smart factors [Carlone et al., 2014]. Smart factors employ the Schur complement to split a large optimization problem into smaller, equivalent sub-problems. Smart factors also detect degenerate instances, i.e., cases when a landmark feature is only observed once or is observed through a degenerate motion. In our work, smart factors were applied to isolate the optimization for the 3D position of each visual feature, which depend on many robot poses over long feature tracks.

For each survey $\mathcal{A}^k$, the result of optimization is optimized camera positions, $x_t^k|_{t=1}^n$, and the visual feature locations $l_e^k|_{e=1}^z$. The optimized camera positions are used to identify two images of the same scene for image registration. The visual features are used to compute the reprojection error, which serves as an indicator how closely the optimized position estimates the true position. Using the values for, $l_e^k|_{e=1}^z$, and the initial measurements, $f_j = m_{t_{first}}^j, ..., m_{t_{last}}^j$, the reprojection error was determined to average around 4-6 pixels in the surveys.

3.3.3 Results

Figures 3.4 and 3.5 give an overview of the performance of our natural image alignment framework. In figure 3.4, images from two datasets taken within two weeks at the end of June 2014 are selected. Below each image, is the aligned image generated by SiftFlow with the change manually highlighted in the red ellipse. Because we can precisely align the images before displaying them to the user, detecting changes is made significantly easier. In fact, most of the changes reported here were detected when reviewing the performance of our approach, without prior knowledge of their occurrence. In figure 3.5, we illustrate the significant robustness to appearance change exhibited by our approach. This is mostly due to the robustness of SiftFlow in itself but also to the prior knowledge of the scene structure that we can recover from the Visual SLAM reconstruction of the environment.

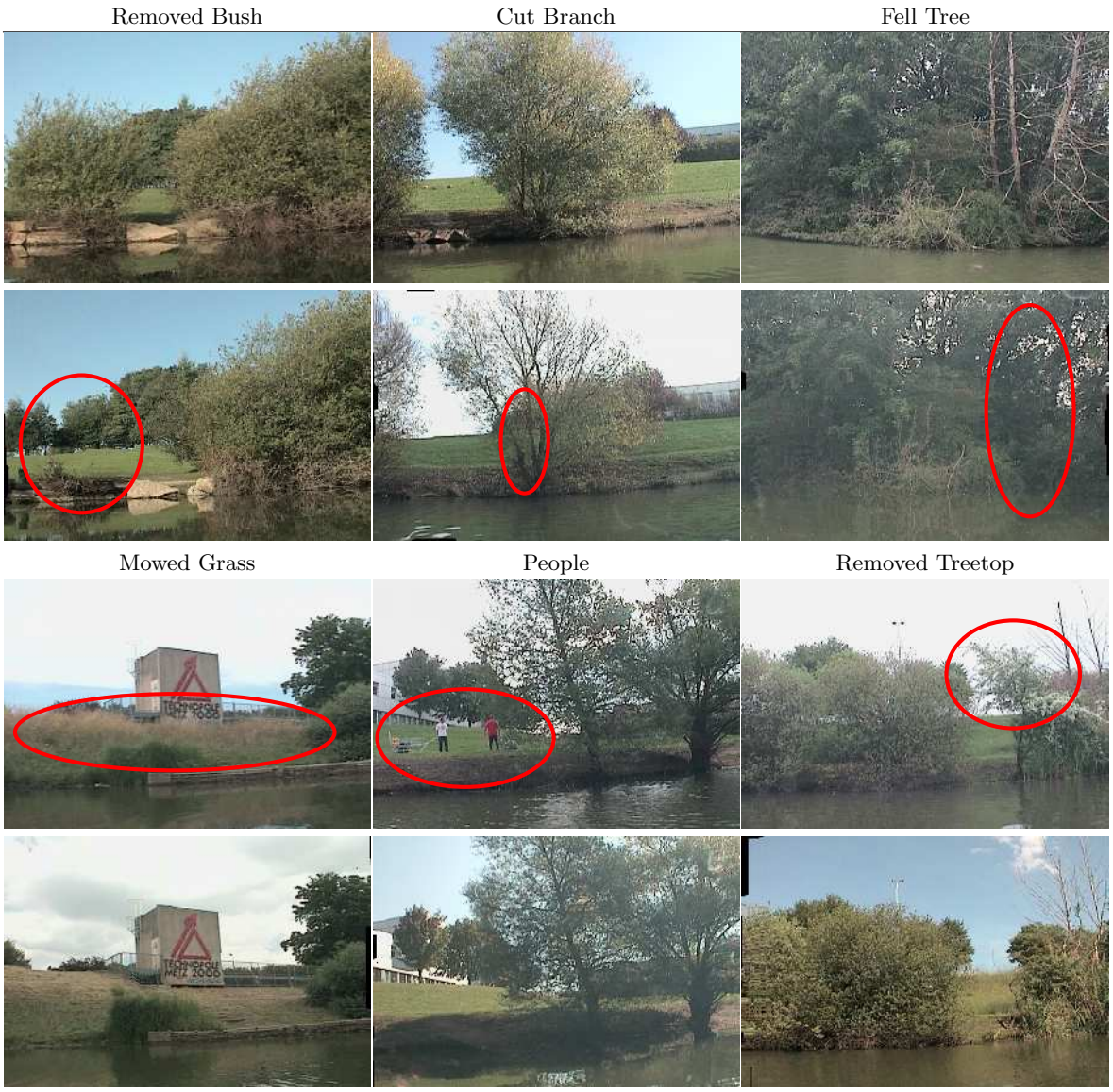


Figure 3.4: Examples of automatically aligned images with manually detected scene changes. Red ellipses highlight the place where changes could be detected.

Figure 3.5: Robustness of the combination of Visual SLAM and Sift Flow to significant variations of scene appearance.

3.3.4 Selection a of minimum view set

A full survey of our 1km-long lake shore corresponds to approximately 60'000 images. It is unreasonable and inefficient to expect a person to visually analyze such a large amount of data. On the other hand, our robot is moving quite slowly along the lake shore, in order to maximize overlap and reduce motion blur. Hence there is actually a minimal subset of images that observe the entire lake shore. The goal of this section is to describe how such a minimal subset can be found.

To find a minimal set of boat poses covering the entire lake shore, we will build our solution on the well-known "Set Cover Problem" (SCP) [Chvatal, 1979]. Adapted to our case, the SCP can be expressed as follows. First, let $\mathcal{S}$ be the set of positions that need to be observed on the lake shore. Each image i in our data set, observes a set $\mathcal{I}_i$ of these shore points. From now on, we will assume that $\mathcal{S}$ is the union of all the point sets $\mathcal{I}_i$. In reallity this union might be larger than $\mathcal{S}$. In this case, we will only consider the part of the $\mathcal{I}_i$'s which intersect $\mathcal{S}$. There is no interest in considering the union smaller than $\mathcal{S}$ since it would mean that $\mathcal{S}$ contains points of the shore that have not been observed and thus can be ignored in our context. The SCP search for a minimal set of indices J such that

$$\mathcal{S} = \bigcup_{j \in J} \mathcal{I}_j \tag{3.1}$$

The Set Cover Problem is known to be NP-Hard (or NP-Complete). It can be approached using linear programming or a simple greedy approach which gives sufficient performance for our application.

One of the challenges we face when using the SCP to solve our shore coverage problem is that we first need to construct a set of shore points to be observed. To improve further the quality of the selected viewpoints, we can add additional constraints to the search.

Constructing the shore points set

This section assumes that the output of the visual-slam has given us a fairly accurate estimate of the pose at which image has been captured. Because the robot is controlled to move at a constant distance d –10m in our case– from the shore, we simply consider that for each pose, every point at distance $d \pm \epsilon$ in the camera frustrum stemming from the pose is a shore point. In practice we take $\epsilon = 1m$. This is implemented by rasterizing the shore map into a pixel map and drawing a thick 10m arc centered on every pose with an angle consistent with the camera intrinsic parameters. At the same time, for every shore pixel on the map, we can also record the set of viewpoints at which it was seen. The set of shore points to observe in two specific datasets is shown in figure 3.6.

Viewpoint constraints

For practical reasons, we need to consider additional constraints when selecting a pose as part of the minimal covering set. The first obvious condition is that the camera must be in a valid configuration, i.e. not pointing at the sky or transitioning between one side and the other. The second constraint is that we want images with a minimal motion blur. Given the low deployment speed of our robot, motion blur only occurs when fast rotations are needed. We filter these images by rejecting any pose with too high a rotation rate, as measured by the on-board gyroscopes.

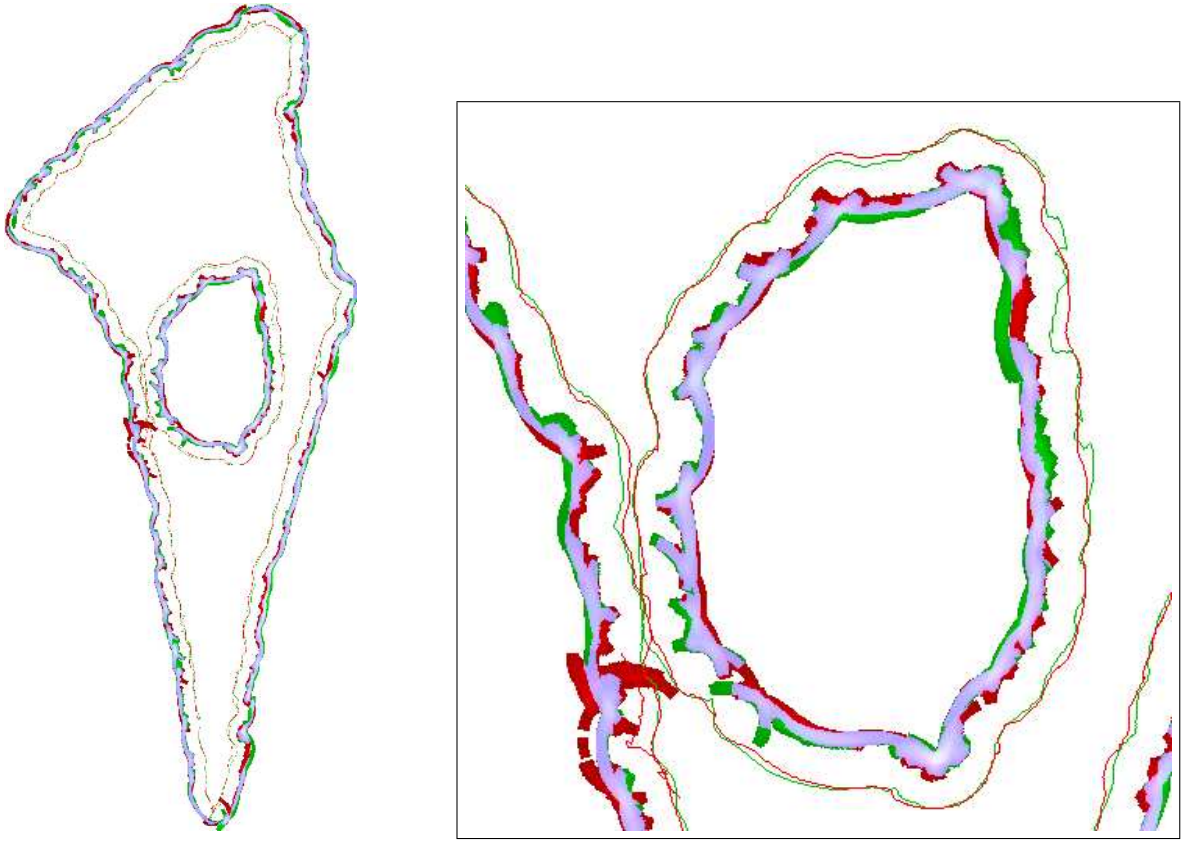


Figure 3.6: Shore points seen from the recorded path of in two datasets. The shore points seen from the red path are displayed in red, the one in green are seen from the green path, and those in magenta/pink are seen from both paths. The right image is a zoomed-in version of the left one.

Finally, the purpose of this selection is to compare one data-set with one acquired at an earlier or later time. Hence, we must ensure that there is actually corresponding viewpoint in the other data-set. A pair of pose is considered useable if the boat positions are close enough, and the intersection of the camera axis with the shore at a specified distance d are close enough. The second constraint could be expressed on the bearing angles but a distance between points lets us keep comparable values.

After removing the poses which don't meet our constraints, we may not be covering the full lake shore anymore. We believe that this is still the best result we can achieve with our data and that including the remaining poses would just require more analysis without significant benefits.

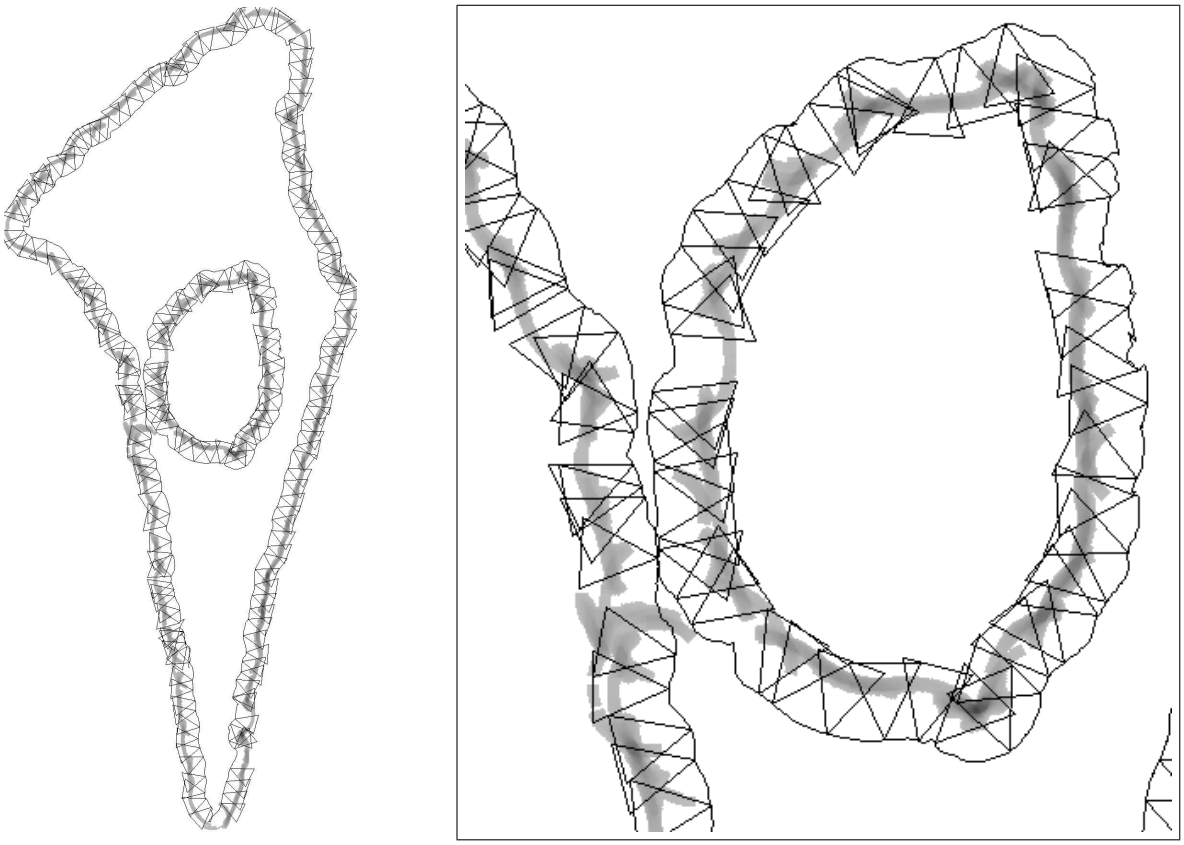


Figure 3.7: Set of viewpoints required to cover the red dataset from figure 3.6 while accounting for visibility in the green dataset. Black triangles correspond to the visibility frustrum of the selected images. The right image is a zoomed-in version of the left one.

Solving the Set Coverage Problem

Under these assumptions, solving the Set Coverage Problem with the greedy algorithm works as follows:

```
Let L be the list of selected viewpoints, initially empty;
while there are shore points to observe do
    Select the valid shore point P which is the least observed;
    Let V be a viewpoint such that
            V observes P and;
            V observes the largest number of unobserved shore points;
    Remove P and all shore point observed in V from the list of points to observe;
    Append V to L;
end
return L
```

Algorithm 1: Algorithm solving the shore coverage problem based on a heuristic for the Set Cover Problem.

On our dataset, the greedy algorithm (see 1) runs in less than 30s and provides results as illustrated in figure 3.7. Out of the 60'000 images, in average 200 are sufficient provide a complete coverage of the high quality observations of the shore. This focus on quality is

at the cost of not considering some images of the shore, either because they don't have a sufficiently close high-quality match in the other dataset or because the rotation speed is too high and causes motion blur.

3.4 Conclusion

This chapter presented our research approach to the monitoring of spatio-temporal environments and in particular natural and unmodified environments where changes occur in a wide range of frequencies. The coming years will see us explore various approaches to modelling, capturing and updating representations of natural environments such as lake and river shores, agricultural fields, crops, forest and parks. Building on spatio-temporal models, we expect to contribute to the state of the art with approaches that take advantage of an autonomous robotic platforms to act in a spatio-temporal environment. Actions could just focused on data acquisition and monitoring with the challenges of information aware planning or extend up to remediation as will be experimented within the Flourish project.

As mentioned in section 3.3, we are currently focused on long-term environment monitoring in natural and changing environments. On the shore of our testing site, we have so far collected more than 1.5 million images of more than 50 hours of autonomous operation over more than 18 months. This unique dataset will provide us with extremely valuable tools for the development of spatio-temporal environment models. As of this writing, preliminary results focusing on building 3D models of the environment and precisely aligning images taken at different time are showing very promising performance.

Bibliography

The argos challenge. http://www.argos-challenge.com/en/content/challenge. Accessed: 2015-01-18.

Robotics projects resulting from H2020 – LEIT ICT 23 Call 1. http://ec.europa.eu/information_society/newsroom/cf/dae/document.cfm?action=display&doc_id=8396. Accessed: 2015-02-12.

Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building Rome in a Day. *Communications of the ACM*, 54(10): 105–112, 2011.

Chris Beall and Frank Dellaert. Appearance-based localization across seasons in a Metric Map. In *6th Workshop on Planning, Perception and Navigation for Intelligent Vehicles (PPNIV)*, Chicago, USA, September 2014.

Amy J Briggs, Daniel Scharstein, Darius Braziunas, Cristian Dima, and Peter Wall. Mobile robot navigation using self-similar landmarks. In *ICRA*, pages 1428–1434, 2000.

Luca Carlone, Zsolt Kira, Chris Beall, Vadim Indelman, and Frank Dellaert. Eliminating Conditionally Independent Sets in Factor Graphs: A Unifying Perspective based on Smart Factors. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

Winston Churchill and Paul Newman. Experience-based navigation for long-term localisation. *The International Journal of Robotics Research*, 32(14):1645–1661, 2013.

Vasek Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of operations research*, 4(3):233–235, 1979.

Christophe Coué, Cédric Pradalier, Christian Laugier, Thierry Fraichard, and Pierre Bessière. Bayesian occupancy filtering for multitarget tracking: an automotive application. *The International Journal of Robotics Research*, 25(1):19–30, 2006.

Frank Dellaert. Factor Graphs and GTSAM: A Hands-on Introduction. Technical Report GT-RIM-CP&R-2012-002, GT RIM, Sept 2012. URL https://research.cc.gatech.edu/borg/sites/edu.borg/files/downloads/gtsam.pdf.

Hendrik Erckens, G-A Busser, Cedric Pradalier, and Roland Y Siegwart. Avalon. *Robotics & Automation Magazine, IEEE*, 17(1):45–54, 2010.

Bernard Espiau, François Chaumette, and Patrick Rives. A new approach to visual servoing in robotics. *Robotics and Automation, IEEE Transactions on*, 8(3):313–326, 1992.

Georgios D Evangelidis and Christian Bauckhage. Efficient subframe video alignment using short descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35 (10):2371–2386, 2013.

Paul Furgale, Timothy D Barfoot, and Gabe Sibley. Continuous-time batch estimation using temporal basis functions. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2088–2095. IEEE, 2012.

Marie-Ève Garneau, Thomas Posch, Gregory Hitz, François Pomerleau, Cédric Pradalier, Roland Siegwart, and Jakob Pernthaler. Short-term displacement of planktothrix rubescens (cyanobacteria) in a pre-alpine lake observed using an autonomous sampling platform. *Limnography and Oceanography*, 58(5), 2013.

Shane Griffith, Paul Drews, and Cédric Pradalier. Towards autonomous lakeshore monitoring. 2014.

Xuming He, Richard S Zemel, and Volodymyr Mnih. Topological map learning from outdoor image sequences. *Journal of Field Robotics*, 23(11-12):1091–1104, 2006.

Gregory Hitz, François Pomerleau, M-E Garneau, Cédric Pradalier, Thomas Posch, Jakob Pernthaler, and Roland Y Siegwart. Autonomous inland water monitoring: Design and application of a surface vessel. *Robotics & Automation Magazine, IEEE*, 19(1):62–72, 2012.

Gregory Hitz, Alkis Gotovos, Franccois Pomerleau, Marie-Eve Garneau, Cedric Pradalier, Andreas Krause, and Roland Y Siegwart. Fully autonomous focused exploration for robotic environmental monitoring. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2658–2664. IEEE, 2014.

Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John J Leonard, and Frank Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *The International Journal of Robotics Research*, 31(2):216–235, 2012.

Jana Košecka. Detecting changes in images of street scenes. In *Computer Vision–ACCV 2012*, volume 7727 of *Lecture Notes in Computer Science*, pages 590–601. Springer, 2013.

Ambroise Krebs, Cédric Pradalier, and Roland Siegwart. Adaptive rover behavior based on online empirical evaluation: Rover–terrain interaction and near-to-far learning. *Journal of Field Robotics*, 27(2):158–180, 2010a.

Ambroise Krebs, Fabian Risch, Thomas Thueer, Jérôme Maye, Cédric Pradalier, and Roland Siegwart. Rover control based on an optimal torque distribution-application to 6 motorized wheels passive rover. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 4372–4377. IEEE, 2010b.

Ce Liu, Jenny Yuen, and Antonio Torralba. Sift flow: Dense correspondence across scenes and its applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):978–994, 2011.

Ming Liu, Cédric Pradalier, and Roland Siegwart. Visual homing from scale with an uncalibrated omnidirectional camera. *IEEE Transaction on Robotics*, (99):1–13, 2013.

David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

Gérard Medioni, Chi-Keung Tang, and Mi-Suen Lee. Tensor voting: Theory and applications. *Proceedings of RFIA, Paris, France*, 3, 2000.

Romain Michalec and Cédric Pradalier. Sidescan sonar aided inertial drift compensation in autonomous underwater vehicles. In *OCEANS'14 MTS/IEEE, St John's, Canada*, September 2014.

Amaury Negre, Cedric Pradalier, and Matthew Dunbabin. Robust vision-based underwater homing using self-similar landmarks. *Journal of Field Robotics*, 25(6-7):360–377, 2008.

Luc Oth, Paul Furgale, Laurent Kneip, and Roland Siegwart. Rolling shutter camera calibration. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1360–1367. IEEE, 2013.

Benjamin Peter, Roman Ratnaweera, Wolfgang Fischer, Cédric Pradalier, and Roland Yves Siegwart. Design and evaluation of a fin-based underwater propulsion system. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3751–3756. IEEE, 2010.

Cédric Pradalier and Pierre Bessière. The cycab: Bayesian navigation on sensory–motor trajectories. In *Probabilistic Reasoning and Decision Making in Sensory-Motor Systems*, pages 51–75. Springer Berlin Heidelberg, 2008.

Cédric Pradalier and Sepanta Sekhavat. Concurrent matching, localization and map building using invariant features. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 1, pages 514–520. IEEE, 2002.

Cédric Pradalier and Kane Usher. Robust trajectory tracking for a reversing tractor trailer. *Journal of Field Robotics*, 25(6-7):378–399, 2008.

Cédric Pradalier, Jorge Hermosillo, Carla Koike, Christophe Braillon, Pierre Bessière, and Christian Laugier. The cycab: a car-like robot navigating autonomously and safely among pedestrians. *Robotics and Autonomous Systems*, 50(1):51–67, 2005.

Cédric Pradalier, Ashley Tews, and Jonathan Roberts. Vision-based operations of a large industrial vehicle: Autonomous hot metal carrier. *Journal of Field Robotics*, 25(4-5):243–267, 2008.

Ulrich Schwesinger, Cedric Pradalier, and Roland Siegwart. A novel approach for steering wheel synchronization with velocity/acceleration limits and mechanical constraints. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5360–5366. IEEE, 2012.

Cédric Siegenthaler, Cédric Pradalier, Fabian Gunther, Gregory Hitz, and Roland Siegwart. System integration and fin trajectory design for a robotic sea-turtle. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 3790–3795. IEEE, 2013.

Pascal Strupler, Cédric Pradalier, and Roland Siegwart. Terrain mapping and control optimization for a 6-wheel rover with passive suspension. In *Field and Service Robotics*, pages 297–310. Springer, 2014.

Elena Stumm, Andreas Breitenmoser, Francois Pomerleau, Cedric Pradalier, and Roland Siegwart. Tensor voting based navigation for robotic inspection of 3d surfaces using lidar point clouds. *The International Journal of Robotics Research*, 31(11), 2012.

Fabien Tâche, Wolfgang Fischer, Gilles Caprari, Roland Siegwart, Roland Moser, and Francesco Mondada. Magnebike: A magnetic wheeled robot with high mobility for inspecting complex-shaped structures. *Journal of Field Robotics*, 26(5):453–476, 2009.

D. Tilbury, O. Sørdalen, L. Bushnell, and S. Sastry. A multisteering trailer system: conversion into chained form using dynamic feedback. *IEEE Trans. on Robotics and Automation*, 1995.

Oliver Wang, Christopher Schroers, Henning Zimmer, Markus Gross, and Alexander Sorkine-Hornung. VideoSnapping: Interactive Synchronization of Multiple Videos. *ACM Trans. Graph.*, 33(4):77:1–77:10, July 2014. ISSN 0730-0301. doi: 10.1145/2601097.2601208. URL http://doi.acm.org.prx.library.gatech.edu/10.1145/2601097.2601208.

Stefan B Williams, Oscar R Pizarro, Michael V Jakuba, Craig R Johnson, Neville S Barrett, Russell C Babcock, Gary A Kendrick, Peter D Steinberg, Andrew J Heyward, Peter J Doherty, et al. Monitoring of benthic reference sites: Using an autonomous underwater vehicle. *Robotics & Automation Magazine, IEEE*, 19(1):73–84, 2012.

Appendix A

Liu et al. [2013]

Ming Liu, Cédric Pradalier, and Roland Siegwart. Visual homing from scale with an uncalibrated omnidirectional camera. *IEEE Transaction on Robotics*, (99):1–13, 2013

Visual Homing From Scale With an Uncalibrated Omnidirectional Camera

Ming Liu, *Student Member, IEEE*, Cédric Pradalier, *Member, IEEE*, and Roland Siegwart, *Fellow, IEEE*

***Abstract*—Visual homing enables a mobile robot to move to a reference position using only visual information. The approaches that we present in this paper utilize matched image key points (e.g., scale-invariant feature transform) that are extracted from an omnidirectional camera as inputs. First, we propose three visual homing methods that are based on feature scale, bearing, and the combination of both, under an image-based visual servoing framework. Second, considering computational cost, we propose a simplified homing method which takes an advantage of the scale information of key-point features to compute control commands. The observability and controllability of the algorithm are proved. An outlier rejection algorithm is also introduced and evaluated. The results of all these methods are compared both in simulations and experiments. We report the performance of all related methods on a series of commonly cited indoor datasets, showing the advantages of the proposed method. Furthermore, they are tested on a compact dataset of omnidirectional panoramic images, which is captured under dynamic conditions with ground truth for future research and comparison.**

***Index Terms*—Omnidirectional camera, topological visual navigation, visual homing, visual servoing.**

I. Introduction

VISUAL homing is defined as navigation by vision of a robot from an arbitrary starting position to a previously specified reference home position [2]. It is considered to be one of the basic abilities of a mobile robot, as well as one of the most important components of visual topological navigation [3], [4]. On one hand, visual homing it is a lightweight method for robot navigation. It utilizes only one pair of images taken at reference and current poses to navigate a mobile robot, regardless the path and motion before reaching the current state. On the other hand, it can be easily extended to an integrated navigation system, by sequentially setting the target nodes on a graph as references. Compared to methods that are based on metric maps [5]–[7], the visual topological navigation framework has the following advantages.

Manuscript received January 28, 2013; accepted June 26, 2013. Date of publication August 1, 2013; date of current version December 2, 2013. This paper was recommended for publication by Associate Editor E. Marchand and Editor G. Oriolo upon evaluation of the reviewers' comments. This paper was presented in part at the 2012 IEEE International Conference on Robotics and Automation. This work was supported in part by the EU project Robots@home under Grant IST-6-045350 and in part by EU FP7 project NIFTi under Contract 247870.

M. Liu is with Autonomous Systems Lab, ETH Zürich, Zürich 8092, Switzerland and also with The Hong Kong University of Science and Technology, Hong Kong (e-mail: ming.liu@mavt.ethz.ch).

C. Pradalier and R. Siegwart are with Autonomous Systems Lab, ETH Zürich, Zürich 8092, Switzerland (e-mail: cedric.pradalier@georgiatech-metz.fr; rsiegwart@ethz.ch).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2013.2272251

1) *Sparse representation of environment:* Usually, the topological map that is used in visual topological navigation is created incrementally, considering the feature changes. A typical representation of the environment is a collection of visual features at certain poses. The computational and memory cost is relatively low.
2) *Independent of precise maps:* As a result, visual homing is less sensitive to error accumulation, which commonly occurs in metric approaches.
3) *Lightweight planning:* The path planning on metric maps can be computationally very expensive; conversely, the planning of visual topological navigation is based on graph structures, which is easy to compute.

The primary challenge of the visual homing problem is the estimation of the *homing vector*. It defines the direction along which the robot can reach the home position. Our methods solve this problem by taking inspiration from the generic framework of visual servoing [8], [9]. Visual servoing has been widely applied in the area of motion control, such as the control of robotic arms for industrial robots. In this paper, we propose a generic visual servoing framework for the visual homing problem. This is implemented as three different homing methods using different properties of features. When deploying such an approach on real robots, special care must be taken with the computational complexity of the algorithm. In general, visual servoing approaches require the computation of the pseudoinverse of a matrix whose size is proportional to the number of sampled features n. In order to ameliorate this, we propose an approach that is inspired from the visual servoing approach, but with a cost linear in the number of features. We show in Section VI that the resulting control is stable, and we compare its performance with other related methods. The originality of this paper is that we take advantage of the scale information of the scale-invariant feature transform (SIFT) features to compute our control law. We show in Section IV that the scale of the features is sufficient to build such a control law.

For practical implementations of visual homing methods, the robotic platforms necessarily have to manage the maneuvers in the presence of dynamic objects, i.e., objects whose position may change from the reference to the current image. If not accounted for, features on these objects will be integrated in the control output, and may lead to unstable or unsafe behavior of the robot. A fast outlier rejection approach is discussed in this paper. It is designed to handle such problems and to improve the overall system performance. The parametrization and validation of the approach will be presented in simulation and experiment.

The major contributions of this paper are as follows.

1) A comparative study of three novel visual homing approaches that are based on bearing and scale information

under classical image-based visual servoing (IBVS) framework.
2) Observability analysis on the system states regarding navigation problem for an omnidirectional camera.
3) A robust and fast homing method that is based on scale information of key-point descriptors, with the proof of stability.
4) Evaluation of the aforementioned methods, as well as an outlier rejection method, via simulations and experiments.
5) A concise dataset for future study of visual control problems using omnidirectional cameras.

In the following, we first give an overview of related work in Section II. Section III reviews the visual homing problem under visual servoing framework. Sections IV and V discuss the fundamental aspects and observability of visual servoing-based homing. Sections VI describes the algorithm and control strategy of a simplified homing algorithm. The control loop is introduced in Section VII, followed by the simulation and experimental results. Conclusions and our future visions are provided at the end of the paper.

II. Related Work

Visual homing is often implemented using bearing-only methods. An initial work was presented by Cartwright and Collett [10] as the "snapshot" model. Franz *et al*. [11] continued this direction by analyzing the error and convergence properties. In our previous work [12], we gave a proof of the convergence of a simplified bearing-only method, based on the Lyapunov stability theory. In this paper, we formulate the bearing-only problem [12] as part of the classical IBVS framework. Meanwhile, two other homing methods under this framework are discussed as well.

Our work is stimulated by the work of Corke [13], where the author used the average landmark vector (ALV) [14] principle to implement a visual servoing task. The ALV method converts the homing problem to a vector operation process, by summing up the bearing vectors to a number of key points at the reference and the current positions. The difference between these two sums is then used to compute the homing vector. However, the following limitations should be considered. First, outliers in landmark detection will greatly affect the accuracy of the homing vector. Second, the outputting homing vector of the ALV algorithm highly depends on the estimation of the global orientation. Third, the homing vector is directly calculated from summarized bearing information of landmarks in current robot pose frame, while the contribution of each landmark is transparent to the final output. Finally, the algorithm is heuristic at intermediate poses. This means that the calculation of accurate homing direction is not feasible, and the mathematical convergence of the algorithm is not guaranteed. In comparison, our approach takes advantage of the scale information attached to the key points to calculate the homing vector without distance estimation, the usage of scale information guarantees the convergence and observability of the system states. The proofs are conducted in this paper.

According to Goedeme *et al*. [15], knowing the structure of an environment and in particular the landmark positions is not necessary for visual homing. This information can be recovered by estimating the ratio of the distances to the matching key points by triangulation using an extended Kalman Filter. Using feature scales, we can avoid this estimation phase and use the scale variation as a proxy for distance errors.

Lim and Barnes [16] presented a homing algorithm that was based on the following principle. They divided the 2-D plane into four regions and estimated the current robot position by measuring the bearings to known landmarks. Compared with their approach, we prove the convergence of our method using the Lyapunov theory. It guarantees the stability of the controller.

A new trend for solving visual homing problems [17] was proposed by Aranda *et al*. and improves the performances by using 1-D trifocal tensors from the omnidirectional camera. Compared with the works using 1-D-trifocal tensors [18], [19], which rely on three-view geometry, our method infers the homing vector directly from the current appearance, and the result is less reliant on feature association. Besides, our approach does not require solving nonlinear equations constructed from the tensors, where algebra error is embedded in the SVD process and impossible to eliminate. Since the reconstruction of the environmental structure is not needed, our servoing-based method requires less computational power.

Furthermore, the authors in [20] used a sliding-mode control law to exploit the epipolar geometry; the authors in [21], directly calculated the homographies from raw images; and Cherubini and Chaumette [22] proposed a redundant framework for visual homing problem, which in particular allows online obstacle avoidance. The comparison with these works is not considered here, since the basic strategies and premises are significantly different.

Some related early works used SIFT as main features for visual homing [23], [24]. They considered the epipolar geometries as well as the orientation and scale of SIFT features for monocular cameras, following a framework similar to [8]. Among these, the work by Vardy and Oppacher [25] is the closest to our simplified approach using scale information. Their work developed a scale invariant local image descriptor for visual homing, based on the optical flow of unwrapped panoramic image from an omnidirectional camera. It was continued by Churchill *et al*. [26], which presents results of real-time homing experiment using the scale difference field in panoramic images, computed from SIFT matches. In comparison to their work, we stress the following two main differences. First, we utilize the error that is caused by the variation of scales, by embedding the scale measures inside the visual servoing framework. Second, we give a mathematical proof of the convergence of the controller, and show the observability of feature states and robot headings. We refer to their method as HSVS, namely heuristic scale-space visual servoing, in the remainder of the paper.

III. Problem Definition

A. Formulation

The visual homing problem can be defined as shown in Fig. 1, where $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n$ are n key points, which are extracted by SIFT, SURF [27], or other methods providing the scale

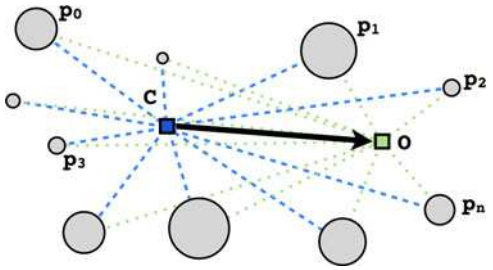


Fig. 1. Abstracted problem of homing. Key points $\boldsymbol{p}_1$ to $\boldsymbol{p}_n$ can be observed at the current position C and at the *Home* position O. The variant sizes indicate the differences of key points in scale.

information of key points. It is assumed that all the listed key points can be seen from the current position C and the home position O. The objective is to guide the robot from C to O only by knowing the observed scale s_i and bearing angle β_i that are associated with key point $\boldsymbol{p}_i$. Negre *et al*. [28] showed that the intrinsic scale can be used to aid the measurement of the time to collision. Hoffmann *et al*. [24] showed a direct relation between the scale and the distance to the feature point. However, for different setup and different environment, the absolute distances to the features cannot be mapped directly. The scale information that we extract from the key points comes from the interpolation of the difference-of-Gaussian pyramid levels, which is commonly used as SIFT key-point detector. We believe that the variation of the scale of a key point can be seen, in first approximation, as a proxy for the variation of its distance. Furthermore, by introducing the scale information, we also prove that the distances to key points are observable given the control speed, as discussed in Section V.

B. Principles of Visual Servoing

In this paper, we assume that the robot can be controlled using a velocity vector, including directions and the absolute values of the speed. This neglects the nonholonomic properties of most robotic platforms. It is acceptable for the simple differential robots used in our experiments, but more work would be needed to adapt this control framework to a more complex system such as a car or a space rover.

When we consider the homing problem as a control problem in the appearance space, it can be summarized as an IBVS problem. In this context, the objective is to drive an error $\boldsymbol{e}$ between the observed and desired features to zero. In a classical visual servoing approach, the error would be the difference in feature coordinates (in pixels). According to the fundamentals of visual servoing, this error can be minimized, if the error dynamics can be linked to the control input v using an *interaction matrix* $\boldsymbol{L}_e$ and the following relation [29]:

$$\dot{\boldsymbol{e}} = \boldsymbol{L_e}\, v. \tag{1}$$

Once the visual error can be properly represented, direct control commands can be calculated in order to minimize the error $\boldsymbol{e}$. Generally, the control commands can be represented in the form

$$v = -\lambda\, \boldsymbol{L}_{\boldsymbol{e}}^{+}\, \boldsymbol{e} \tag{2}$$

where $\boldsymbol{L}_{\boldsymbol{e}}^{+}$ is the pseudoinverse of the *interaction matrix* $\boldsymbol{L_e}$. This controller is designed to have an exponential convergence rate of the error, if the stability of the system is ensured. According to the stability analysis of IBVS in [29], the interaction matrix and its pseudoinverse need to be full rank, in order to guarantee local asymptotic stability.

IV. Image-Based Visual Servoing

In this paper, we adopt the visual servoing framework to build a robot controller, by using the scale and bearing information of the key points, instead of their coordinates. We assume that the key points are extracted from an omnidirectional camera, and we can easily convert image coordinates to bearing angles. We also assume that we are able to control the target robot by velocity commands, and the robot configuration can be summarized by its position (x, y) and its heading θ.

A. Definitions

The error of the system is comprised of two components: the scale error and the bearing angle error. Therefore, the vector of the error can be written as

$$\boldsymbol{e} = (\boldsymbol{s} - \boldsymbol{s}^*, \boldsymbol{\beta} - \boldsymbol{\beta}^*)^T \tag{3}$$

where $\boldsymbol{s} = (s_1, \ldots, s_n)$ is the vector of observed scale of the key points, and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_n)$ is the vector of their bearing angles. The "*" superscript denotes the reference variables.

Before computing the derivative of the error, we need to derive the relation between the scale of a feature s_i and the distance to the corresponding entity l_i. The focal length of the camera is denoted by f and S denotes the physical size of the region defined by the corresponding key-point patch. Using simple triangulation and the camera pin-hole model, we have

$$s_i = \frac{S\, f}{l_i}, \text{ and } s_i^* = \frac{S\, f}{l_i^*} \tag{4}$$

which leads to

$$s_i = s_i^* \frac{l_i^*}{l_i}. \tag{5}$$

Assuming that the physical key point i is located at the 2-D coordinates (x_i, y_i) in the same frame as the robot, we can make explicit the relation between l_i, β_i, and the robot position

$$l_i = \sqrt{(x_i - x)^2 + (y_i - y)^2} \tag{6}$$

$$\beta_i = \text{atan2}(y_i - y, x_i - x) - \theta. \tag{7}$$

B. Error Derivative

To derive the error dynamics $\dot{\boldsymbol{e}}$, we compute independently the scale and bearing derivatives, by considering them as a function of the robot pose

$$\frac{d}{dt}\left[s_i(x, y, \theta) - s_i^*\right] = \frac{d\, s_i}{dx}\frac{dx}{dt} + \frac{d\, s_i}{dy}\frac{dy}{dt} + \frac{d\, s_i}{d\theta}\frac{d\theta}{dt}. \tag{8}$$

Using (5) and (6), we have

$$\frac{d}{dt}\left[s_i(x,y,\theta)-s_i^*\right] = s_i^* l_i^* \left[v_x \frac{d}{dx}\frac{1}{l_i} + v_y \frac{d}{dy}\frac{1}{l_i}\right] = -\frac{s_i^* l_i^*}{l_i^2}\left[v_x \cos\beta_i^g + v_y \sin\beta_i^g\right] \quad (9)$$

with $v_x = \frac{dx}{dt}$, and $v_y = \frac{dy}{dt}$.

Similarly, the bearing error can be derived as

$$\frac{d}{dt}\left[\beta_i^g(x,y,\theta)-\beta_i^*\right] = \frac{d\,\beta_i^g}{dx}\frac{dx}{dt} + \frac{d\,\beta_i^g}{dy}\frac{dy}{dt} + \frac{d\,\beta_i^g}{d\theta}\frac{d\theta}{dt}. \quad (10)$$

By plugging in the (7), this leads to

$$\frac{d}{dt}\left[\beta_i^g(x,y,\theta)-\beta_i^*\right] = -\frac{y_i-y}{l_i^2}v_x - \frac{x_i-x}{l_i^2}v_y - \omega = -\frac{1}{l_i}\left[v_x \sin\beta_i^g + v_y \cos\beta_i^g\right] - \omega \quad (11)$$

with $\omega = \frac{d\theta}{dt}$.

Similarly, the derivative of the distance to a key point i is

$$\frac{d}{dt}\left[l_i(x,y,\theta)-l_i^*\right] = -(v_x \cos\beta_i^g + v_y \sin\beta_i^g). \quad (12)$$

Note that β_i^g is the bearing angle of feature i in the global frame instead of robot local frame. In order to transform them to the robot local frame, the heading difference needs to be considered, such that the bearing observation by the robot is

$$\beta_i = \beta_i^g - (\theta - \theta^*). \quad (13)$$

In practice, we can consider $\theta^* = 0$, meaning that we take the reference robot pose as globally zero degree heading.

Combining (9) to (11), we can write the error dynamics as follows:

$$\dot{\boldsymbol{e}} \doteq \frac{d}{dt}\boldsymbol{e} = \boldsymbol{L_e} v \quad (14)$$

$$\frac{d}{dt}\begin{pmatrix} s_1 - s_1^* \\ \vdots \\ s_n - s_n^* \\ \beta_1 - \beta_1^* \\ \vdots \\ \beta_n - \beta_n^* \end{pmatrix} = \begin{pmatrix} -\frac{s_1^* l_1^*}{l_1^2}\cos\beta_1^g & -\frac{s_1^* l_1^*}{l_1^2}\sin\beta_1^g & 0 \\ \vdots & \vdots & \vdots \\ -\frac{s_n^* l_n^*}{l_n^2}\cos\beta_n^g & -\frac{s_n^* l_n^*}{l_n^2}\sin\beta_n^g & 0 \\ -\frac{1}{l_1}\sin\beta_1^g & -\frac{1}{l_1}\cos\beta_1^g & -1 \\ \vdots & \vdots & \vdots \\ -\frac{1}{l_n}\sin\beta_n^g & -\frac{1}{l_n}\cos\beta_n^g & -1 \end{pmatrix} \times \begin{pmatrix} v_x \\ v_y \\ \omega \end{pmatrix}.$$

As mentioned earlier, the interaction matrix in (14) is required for the implementation of a visual servoing controller. One remaining problem is that neither the distance l_i nor l_i^* can be quantified easily using a single camera. Based on [30] and the analysis on the visual errors in [9], we assume these values can be approximated by constants due to the low sensitivity of the controller to these parameters.

A direct way to reduce the complexity is that either the upper part or the lower part of (14) is sufficient to form a visual servoing interaction matrix. As it is trivial to rotate the robot on spot once the translational error has been corrected, a two-stage controller can be designed. First, deal with the translation error, then correct the heading. We consider mainly the first stage, since it is the key issue for homing. In practice, this means that we can either implement a scale-only visual servoing or a bearing-only visual servoing.

The interaction matrix for scale-only visual servoing is shown as follows:

$$\frac{d}{dt}\begin{pmatrix} s_1 - s_1^* \\ \vdots \\ s_n - s_n^* \end{pmatrix} = \begin{pmatrix} \alpha_1 s_1^* \cos\beta_1^g & \alpha_1 s_1^* \sin\beta_1^g \\ \vdots & \vdots \\ \alpha_n s_n^* \cos\beta_n^g & \alpha_n s_n^* \sin\beta_n^g \end{pmatrix} \begin{pmatrix} v_x \\ v_y \end{pmatrix}. \quad (15)$$

The controller that is based on (15) is denoted as scale-only visual servoing (SOVS) in the remainder of this paper. A similar method, using the lower part of (14), is called a bearing-only approach (BOVS), whose error dynamics can be derived as

$$\frac{d}{dt}\begin{pmatrix} \beta_1 - \beta_1^* \\ \vdots \\ \beta_n - \beta_n^* \end{pmatrix} = \begin{pmatrix} \gamma_1 \sin\beta_1^g & \gamma_1 \cos\beta_1^g \\ \vdots & \vdots \\ \gamma_n \sin\beta_n^g & \gamma_n \cos\beta_n^g \end{pmatrix} \begin{pmatrix} v_x \\ v_y \end{pmatrix}. \quad (16)$$

According to the generic properties of the controller that are stated in Section III-B, the local asymptotic stability is maintained if each interaction matrix and its pseudoinverse are full ranked. This can be ensured by using reasonable big number of matched features in real applications.

Recalling (13), the estimation of the heading θ is crucial for the calculation of the interaction matrices. This implies that a robot may need to have absolute heading references such as a magnetic compass or a reliable visual compass for better accuracy. Regarding the dataset that we use in Section IX, where the robot is well aligned, this problem is trivial. However, this matter needs to be considered in real applications.

V. Observability of the Scale-Based Visual Control System

A. Definitions

Given the raw sensor measurements, the observability analysis of system states is important before we step on the designation of other advanced controllers. The observability analysis also gives hints for what results can be expected from the system configuration. The configuration of the discussed IBVS system is described via the following system states:

$$\mathbf{x} = \begin{cases} x : x\text{-coordinate of the robot position} \\ y : y\text{-coordinate of the robot position} \\ \theta : \text{heading of the robot} \\ \boldsymbol{s} : \text{vector of observed scales} \\ \boldsymbol{\beta} : \text{vector of observed bearings} \\ \boldsymbol{l} : \text{distance to features.} \end{cases} \quad (17)$$

In order to get the full state for a motion constrained in 2-D plane, we require at least three positively matched key points, considering the number of geometrical constraints. Without the

loss of generality, we can use the following subset of the full state, where the minimum required key points are denoted by subscripts 1, 2, and 3

$$\underline{\mathbf{x}} = (x\, y\, \theta\, s_1\, s_2\, s_3\, \beta_1\, \beta_2\, \beta_3\, l_1\, l_2\, l_3)^T. \tag{18}$$

Following the definitions in (9), (11), and (12), using "*" to denote the reference variables, the state derivatives of the system state is presented as (19), shown at the bottom of the page.

The observation function, which is also the zero-order Lie derivative, includes the scale and bearing information

$$L^0 h = h(x) = (\, s_1\, s_2\, s_3\, \beta_1\, \beta_2\, \beta_3\,)^T. \tag{20}$$

The further Lie derivatives of the observation function over the control functions are shown in (21)–(23). The control functions are comprised of the three columns of (19), denoted by f_1, f_2, f_3, respectively. For the purpose of conciseness, we use C_i to denote $\cos(\beta_i + \theta - \theta^*)$, and S_i for $\sin(-\beta_i - \theta + \theta^*)$

$$\nabla L^0 h = \nabla h(x) = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}. \tag{21}$$

B. Observation Matrix

We stop the computation by (23), because it has illustrated that the observation matrix shown in (24) is already row full ranked using (21)–(23), but not column full ranked. We notice that continuing to calculate further derivatives would not help it become column full rank, since the entries are independent of x or y, (22)–(24), all shown at the bottom of the next page.

Using all the measurable variables and the known control commands, the analysis of the null space of M reveals the inherent relations among each state. The null space of the full observation matrix M is given as

$$\text{nullspace}(M) = \{[1, 0, 0, \mathbf{0}, \mathbf{0}, \mathbf{0}], [0, 1, 0, \mathbf{0}, \mathbf{0}, \mathbf{0}]\}. \tag{25}$$

This shows that the position of the robot (x, y) in the global frame is not observable due to nonzero entries in the nullspace. This is fine, since they are not of interest without knowing the global reference frame. More importantly, it shows that the other states θ, $\boldsymbol{s}$, $\boldsymbol{\beta}$, and particularly the distances to the feature points $\boldsymbol{l}$ are observable, as long as the control command to the robot is known.

Though not explicitly calculated by the homing algorithm, these observable constraints imply that it is able to determine the translation and rotation to a predefined position, since three known distances to static feature points are adequate to well determine such a transform for the motion in a 2-D plane.

VI. Fast Visual Homing

From the observability analysis, we could see that the scale information and bearing angles from local frame and especially their changes are sufficient information for pose control. Derived from this, we describe a scale-based visual homing approach that does not require the computation of the pseudoinverse of an interaction matrix in this section. Above all, this approach is independent of the global heading estimation. Since the global heading is usually approximated by visual compass and with nonneglectable error [31], this method avoids such potentially extra errors for real applications. We also provide the convergence proof for the resulting control law.

$$\dot{\underline{\mathbf{x}}} = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ \sin\theta & -\cos\theta & 0 \\ 0 & 0 & 1 \\ -\dfrac{s_1^* l_1^* \cos(\beta_1 + (\theta - \theta^*))}{{l_1}^2} & -\dfrac{s_1^* l_1^* \sin(\beta_1 + (\theta - \theta^*))}{{l_1}^2} & 0 \\ -\dfrac{s_2^* l_2^* \cos(\beta_2 + (\theta - \theta^*))}{{l_2}^2} & -\dfrac{s_2^* l_2^* \sin(\beta_2 + (\theta - \theta^*))}{{l_2}^2} & 0 \\ -\dfrac{s_3^* l_3^* \cos(\beta_3 + (\theta - \theta^*))}{{l_3}^2} & -\dfrac{s_3^* l_3^* \sin(\beta_3 + (\theta - \theta^*))}{{l_3}^2} & 0 \\ -\dfrac{\sin(\beta_1 + (\theta - \theta^*))}{l_1} & -\dfrac{\cos(\beta_1 + (\theta - \theta^*))}{l_1} & -1 \\ -\dfrac{\sin(\beta_2 + (\theta - \theta^*))}{l_2} & -\dfrac{\cos(\beta_2 + (\theta - \theta^*))}{l_2} & -1 \\ -\dfrac{\sin(\beta_3 + (\theta - \theta^*))}{l_3} & -\dfrac{\cos(\beta_3 + (\theta - \theta^*))}{l_3} & -1 \\ -\cos(\beta_1 + (\theta - \theta^*)) & -\sin(\beta_1 + (\theta - \theta^*)) & 0 \\ -\cos(\beta_2 + (\theta - \theta^*)) & -\sin(\beta_2 + (\theta - \theta^*)) & 0 \\ -\cos(\beta_3 + (\theta - \theta^*)) & -\sin(\beta_3 + (\theta - \theta^*)) & 0 \end{pmatrix} \cdot \begin{pmatrix} v_x \\ v_y \\ \omega \end{pmatrix}. \tag{19}$$

A. Scale-Based Control for a 1-D Robot

Recalling (9)

$$\frac{d}{dt}\left(s_i - s_i^*\right) = -\frac{s_i^* l_i^*}{l_i^2}\left[v_x \cos\beta_i + v_y \sin\beta_i\right].$$

For the sake of argument, let us first consider an 1-D robot, which is only able to move along the direction toward key point i. Because the right side of the aforementioned equation can be seen as the projection of the robot velocity in the direction toward the key point, denoting $e_i = s_i - s_i^*$, and $v_i = v_x \cos\beta_i + v_y \sin\beta_i$, we have

$$\frac{d}{dt}e_i = -\frac{s_i^* l_i^*}{l_i^2} v_i. \tag{26}$$

Following the designation strategy of visual servoing, we would like to ensure an exponential decoupled decrease of the error [29]. The following trivial control would achieve this goal (λ is a positive constant)

$$v_i = \lambda_i e_i. \tag{27}$$

B. Generic Scale-Based Control

We abuse the intuition that the individual controllers for the 1-D case may be combined to calculate the required velocity for the 2-D case as follows:

$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} = \sum_{i=1}^{n} \lambda_i \left(s_i - s_i^*\right) \begin{pmatrix} \cos\beta_i \\ \sin\beta_i \end{pmatrix}. \tag{28}$$

However, even if the convergence was obvious in the 1-D case, there is no guarantee that this sum of control contributions would lead to a stable controller. In order to show the convergence, we resort to the Lyapunov theory. We define the following

$$L_{f_1}^1 h = \frac{\partial h(x)}{\partial x}\cdot f_1 = \nabla L_{f_1}^0 h \cdot f_1 = \left(-\frac{s_1^* l_1^* C_1}{{l_1}^2}, \quad -\frac{s_2^* l_2^* C_2}{{l_2}^2}, \quad -\frac{s_3^* l_3^* C_3}{{l_3}^2}, \quad \frac{S_1}{l_1}, \quad \frac{S_2}{l_2}, \quad \frac{S_3}{l_3}\right)^T$$

$$\nabla L_{f_1}^1 h = \begin{pmatrix} 0 & 0 & -\frac{s_1^* l_1^* S_1}{{l_1}^2} & 0 & 0 & 0 & -\frac{s_1^* l_1^* 1 S_1}{{l_1}^2} & 0 & 0 & 2\frac{s_1^* l_1^* C_1}{{l_1}^3} & 0 & 0 \\ 0 & 0 & -\frac{s_2^* l_2^* S_2}{{l_2}^2} & 0 & 0 & 0 & 0 & -\frac{s_2^* l_2^* S_2}{{l_2}^2} & 0 & 0 & 2\frac{s_2^* l_2^* C_2}{{l_2}^3} & 0 \\ 0 & 0 & -\frac{s_3^* l_3^* S_3}{{l_3}^2} & 0 & 0 & 0 & 0 & 0 & -\frac{s_3^* l_3^* S_3}{{l_3}^2} & 0 & 0 & 2\frac{s_3^* l_3^* C_3}{{l_3}^3} \\ 0 & 0 & -\frac{C_1}{l_1} & 0 & 0 & 0 & -\frac{C_1}{l_1} & 0 & 0 & -\frac{S_1}{{l_1}^2} & 0 & 0 \\ 0 & 0 & -\frac{C_2}{l_2} & 0 & 0 & 0 & 0 & -\frac{C_2}{l_2} & 0 & 0 & -\frac{S_2}{{l_2}^2} & 0 \\ 0 & 0 & -\frac{C_3}{l_3} & 0 & 0 & 0 & 0 & 0 & -\frac{C_3}{l_3} & 0 & 0 & -\frac{S_3}{{l_3}^2} \end{pmatrix}. \tag{22}$$

$$L_{f_2}^1 h = \frac{\partial h(x)}{\partial x}\cdot f_2 = \nabla L_{f_2}^0 h \cdot f_2 = \left(-\frac{s_1^* l_1^* S_1}{{l_1}^2}, \quad -\frac{s_2^* l_2^* S_2}{{l_2}^2}, \quad -\frac{s_3^* l_3^* S_3}{{l_3}^2}, \quad -\frac{C_1}{l_1}, \quad -\frac{C_2}{l_2}, \quad -\frac{C_3}{l_3}\right)^T$$

$$\nabla L_{f_2}^1 h = \begin{pmatrix} 0 & 0 & -\frac{s_1^* l_1^* C_1}{{l_1}^2} & 0 & 0 & 0 & -\frac{s_1^* l_1^* 1 C_1}{{l_1}^2} & 0 & 0 & 2\frac{s_1^* l_1^* S_1}{{l_1}^3} & 0 & 0 \\ 0 & 0 & -\frac{s_2^* l_2^* C_2}{{l_2}^2} & 0 & 0 & 0 & 0 & -\frac{s_2^* l_2^* C_2}{{l_2}^2} & 0 & 0 & 2\frac{s_2^* l_2^* S_2}{{l_2}^3} & 0 \\ 0 & 0 & -\frac{s_3^* l_3^* C_3}{{l_3}^2} & 0 & 0 & 0 & 0 & 0 & -\frac{s_3^* l_3^* C_3}{{l_3}^2} & 0 & 0 & 2\frac{s_3^* l_3^* S_3}{{l_3}^3} \\ 0 & 0 & -\frac{S_1}{l_1} & 0 & 0 & 0 & -\frac{S_1}{l_1} & 0 & 0 & -\frac{C_1}{{l_1}^2} & 0 & 0 \\ 0 & 0 & -\frac{S_2}{l_2} & 0 & 0 & 0 & 0 & -\frac{S_2}{l_2} & 0 & 0 & -\frac{C_2}{{l_2}^2} & 0 \\ 0 & 0 & -\frac{S_3}{l_3} & 0 & 0 & 0 & 0 & 0 & -\frac{S_3}{l_3} & 0 & 0 & -\frac{C_3}{{l_3}^2} \end{pmatrix}. \tag{23}$$

$$M = \begin{pmatrix} \nabla L^0 h \\ \nabla L_{f_1}^1 h \\ \nabla L_{f_2}^1 h \end{pmatrix} \tag{24}$$

nonnegative energy function (Lyapunov candidate function):

$$E = \frac{1}{2}\sum_{i=1}^{n}\left(\frac{s_i - s_i^*}{s_i^*}\right)^2. \tag{29}$$

In this autonomous system with n-dimensional states $\boldsymbol{s}$, the only equilibrium is where $\boldsymbol{s} = \boldsymbol{s}^*$ in the feature space; and physically it is the reference home position. According to the Lyapunov theory, we need to show that

$$\begin{cases} \dfrac{d}{dt}E(t) = 0, & \text{only when all } s_i = s_i^* \\ \dfrac{d}{dt}E(t) < 0, & \text{otherwise.} \end{cases} \tag{30}$$

Based on the calculation in (9), the derivative of the energy function is

$$\begin{aligned} \frac{d\,E}{dt} &= \sum_{i=1}^{n}\frac{s_i - s_i^*}{s_i^*}\frac{d\,s_i}{dt} \\ &= -\sum_{i=1}^{n}\frac{s_i - s_i^*}{s_i^*}\frac{s_i^* l_i^*}{l_i^2}\left[v_x\cos\beta_i + v_y\sin\beta_i\right] \\ &= -\left[v_x\sum_{i=1}^{n}\frac{s_i^* l_i^*}{l_i^2}\frac{s_i - s_i^*}{s_i^*}\cos\beta_i + v_y\sum_{i=1}^{n}\frac{s_i^* l_i^*}{l_i^2}\frac{s_i - s_i^*}{s_i^*}\sin\beta_i\right]. \end{aligned} \tag{31}$$

Denoting

$$\lambda_i = \frac{l_i^*}{l_i^2} \tag{32}$$

and combining with (28), (31) is simplified as

$$\frac{d\,E}{dt} = -\left[v_x^2 + v_y^2\right] = -\sum\lambda_i^2(s_i - s_i^*)^2 \le 0. \quad \square \tag{33}$$

Since the distances l_i and l_i^* are nonnegative, (33) shows that the control law of (28) is stable, and converges to $s_i = s_i^*$ (i.e., the reference home position). However, l_i and l_i^* are not directly measurable in practice, though observable. Following the error analysis in [9], we approximate the λ_i's by constants, since they do not affect the convergence. Further validation via simulation is given in Section VIII-B1).

VII. Integration and Control Loop

In this section, we will discuss how the control laws can be instantiated on a real system.

A. Control Loop

The control loop is depicted in Fig. 2. We first focus on the visual processing pipeline on the right hand side of the dotted line. It starts with capturing a new panoramic image. During each running cycle, the omnidirectional image that is acquired from the camera is unwrapped first, using a simple polar-coordinate transformation. Note that this implies a minor assumption on the calibration of the camera, namely that the image center is known, but it is not necessary to know a full model of the catadioptric shape.

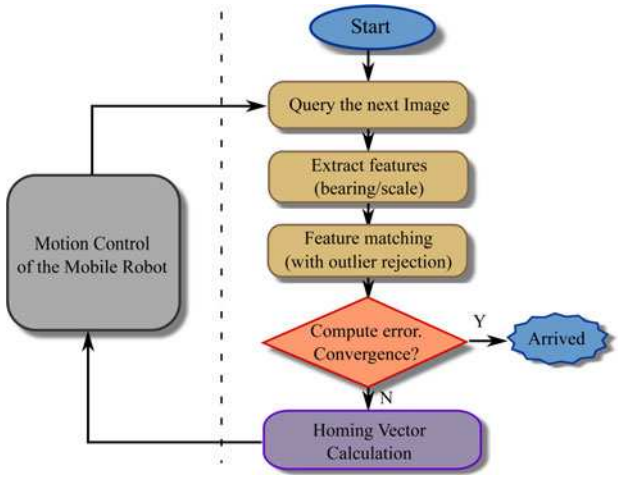


Fig. 2. Control loop of multipoint homing.

SIFT features are then extracted from the unwrapped images and used to compare the current image with the one acquired at the home position. Although there is certainly a distortion in the SIFT features due to mirror nonlinearity, this approach is also used in related works and in particular in [25].

From the matched key points, the visual servoing control law is applied, and a homing vector is computed. This vector is then transmitted to the robot motion controller. The control vector is then implemented while accounting for the nonholonomic constraints of the target robot. The control loop is executed with each newly captured image until convergence.

B. Outlier Rejection

When deploying visual homing algorithms in a real environment, it is likely that the scene captured at the home position will change. For instance, the original image may contain furnitures or objects that are moved over time, e.g., moving people. In general, in an omnidirectional image, these objects only cover a small portion of the image. In this section, we will describe how to remove the falsely matched features, which may be caused by these dynamic objects. Note that if a feature is extracted from the objects, which disappear from the home image, This usually does not raise a significant issue. This is because they usually cannot get matched to anything in the current frame.

The outlier rejection method is the key to enhancing the reliability of our control method against dynamic objects. The underlying assumption is that the main transformation between the current image and the home image is simply a pure rotation. This is true as long as the features are far enough from the camera, or distance between the current position, and the home position is small in comparison to the average distance to the features. In practice, we take every matched feature in the current image, and use the bearing difference with its home correspondence to vote for a potential rotation angle. Once every feature has cast its vote, the resulting histogram contains a main peak corresponding to the main rotation. Other peaks correspond to a set of features that moved coherently between

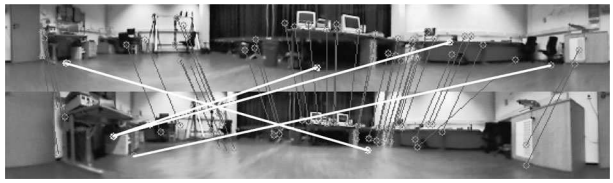


Fig. 3. Result of our outlier rejection algorithm. The gray thin linkages mark the inliers of the matching, and the white linkages mark the outliers.

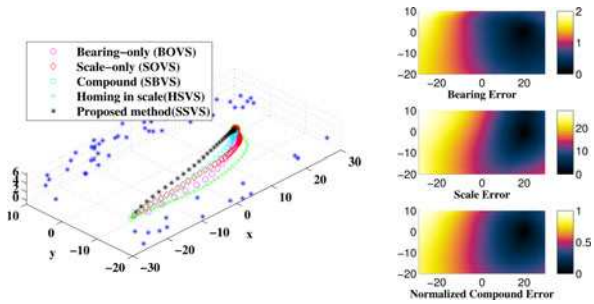


Fig. 4. (Left) Simulated trajectories by different visual homing methods. (Right) Error evaluation on the simulated environment referring to (20,0).

the two images, i.e., dynamic objects. The background of the histogram refers to outliers or points that violate the hypothesis of a pure rotational error, i.e., points too close to the camera. The assumption may appear a rough approximation of the general case, but used with a lenient threshold, it provides a reliable filter for gross outliers, which would have the worse effect on the control law. In Section VIII-C, we provide an evaluation of the effect of such an assumption. As a result, an instance of outlier rejection using our approach is shown as Fig. 3. The white linkages mark the detected outliers from the raw matching result.

VIII. Simulation Results

In this section, we present a number of simulation results highlighting some properties, and basic principles of the homing approaches. The first evaluation is to compare the convergence behavior of different control methods. To this end, it is important to have a common metric to compare all the image-based homing, and characterize the influence of various assumptions and parameters. In particular, we will illustrate the effect of not knowing the distance to the features and the influence of the pure-rotation assumption for outlier rejection.

A. Performance Comparison Using Image-Based Visual Homing

1) Comparison of Convergence Behavior: In previous sections, we have discussed four approaches and one related method by [26]:

BOVS: Bearing-only visual servoing, which only takes the bearing angles to the features to compute the control law [see (16)].

SOVS: Scale-only visual servoing, which mainly utilizes the scale information, and the bearing plays a role in the interaction matrix as well [see (15)].

SBVS: Scale and bearing visual servoing uses the full interaction matrix, which integrates the bearing and scale error [see (14)].

HSVS: As a comparison, we also implemented the algorithm in [26], referred to in this paper as HSVS. The implementation follows the summarized equivalent equation:

$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} = \sum_i \text{sign}(s_i - s_i^*) \begin{pmatrix} \cos\beta_i \\ \sin\beta_i \end{pmatrix}.$$

SSVS: The simplified scale-based visual servoing, which also uses scale information but does not require the pseudoinverse computation [see (28)].

Fig. 4 depicts the simulated environment, where visual features (blue stars) are assumed to be located on walls surrounding a ground robot, each with a specified objective scale. The green star marks the start position of the robot, while the red-filled circle is the target home position. A simulated robot uses the observed features to implement the control laws that were presented earlier in the paper, as well as HSVS from [26]. The scale information is computed using a simple pin-hole model, as described in (4). Fig. 4 depicts the incrementally generated trajectories of the five methods.

We tune the controller gain for all the controllers such that the number of iterations required by each method are similar. The four visual servoing trajectories shows that all the control laws guide the robot to the home position in a reasonable manner. However, the proposed SSVS method leads to the straightest trajectory, whereas the HSVS method makes the biggest detour.

The error field of the simulated environment is depicted on the right of Fig. 4, by taking (20,0) as the reference home position. Although the error field of the bearing error appears *flatter* than the scale error field, we could observe that combining bearing and scale error definitely help to shape the error field into a round-shaped potential well.

Fig. 5 provides a graphical view of the normalized error convergence using the various approaches. Note that the first row of the graph represents the bearing error in radian, whereas the other rows represent the error in the scale or combined space. In each graph, the abscissa refers to the number of iterations. As expected, the methods proposed in this study (BOVS, SOVS, SBVS, and SSVS) all show overall exponential convergence, fitting the justification of (2) indicated by [29]. Since the exponential convergence of HSVS is not mathematically guaranteed, the behavior of the error convergence does not reflect such characteristic.

B. Influence of the Assumptions for SSVS

For the visual servoing approaches, the convergence proof relies on the knowledge of the distance to the features. In practice, this assumption cannot be easily fulfilled with a single camera. For the standard visual servoing, it has been shown that

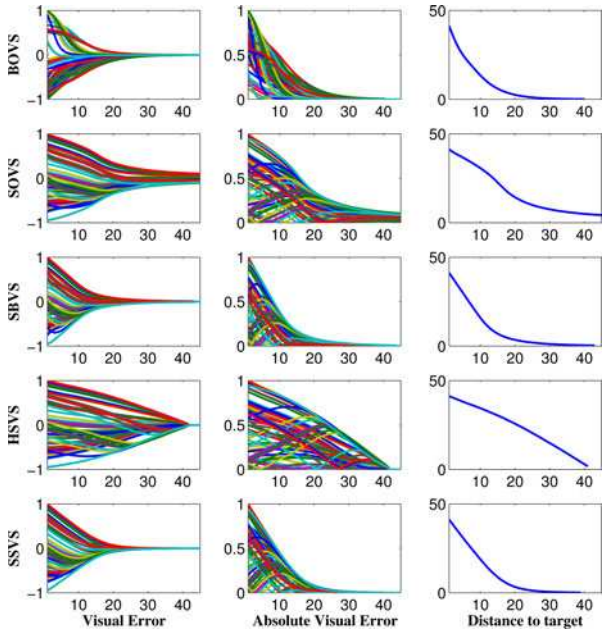


Fig. 5. Vsual feature errors and the error distance to the home position over time.

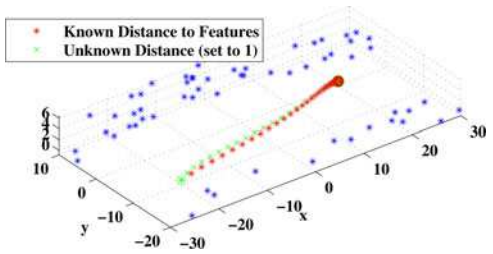


Fig. 6. Comparison between the assumptions with known and unknown distances to features.

assuming that all the features are at unit distance from the camera leads to efficient control strategies [29].

Because we abuse this assumption for SSVS, it is necessary to show its influence on the trajectories. Fig. 6 shows that there is indeed a small difference, but the performance loss, in terms of additional traveled distance, is minor. As for control parameters, different distance assumptions lead to different gains. For the test in Fig. 6, we have tuned the gain so that the number of iterations until convergence is similar in both cases. Taking the same starting and ending positions, by changing the environment features and varying the assumed distance constant, we do the simulation 100 times. We obtain an average traveled distance of 42.1 when the distances are set to unity against 41.3 when knowing the distances. The average absolute curvature are also similar by 0.036 against 0.030. This example shows that similar to the standard visual servoing, the scale-based approach is not sensitive to the known distance assumption, and that distances can be safely approximated to be constant (e.g. 1.0) for real implementations.

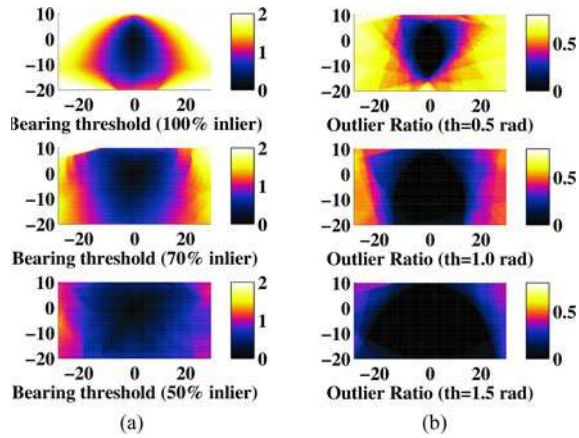


Fig. 7. Justification of the assumption for outlier rejection. (a) Bearing threshold (in rad) for a fixed inlier ratio. (b) Outlier ratio by fixing bearing threshold (in rad).

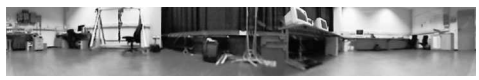


Fig. 8. Instance at (8,4) from the A1originalH database of [32].

C. Parametrization of the Outlier Rejection

A successful homing process does not depend on a huge numbers of matched features. Therefore, a relatively low inlier ratio, e.g., 50%, is usually more than sufficient to control the robot, given reasonable large numbers of matches. As a reminder, we select inliers if they globally correspond to a pure rotation of the robot, thus ignoring the translational error with respect to the home position. This is done by first finding the best rotation by a voting scheme, and then selecting as inlier the features that correspond to a best rotation approximation, within a given angular threshold.

In order to justify the assumption and evaluate within what region would such an assumption be true, we carry out the test as follows. Considering the simulated environment depicted in Fig. 4, we take the position at coordinate (0,0) as the reference, then evaluate all the other positions (with a resolution of 0.1m) by measuring the numbers of features that do not fit the assumption, namely the outlier ratio.

The analysis result is shown in Fig. 7. The color in Fig. 7(a) implies the minimum required value of the bearing threshold (in radian) for different inlier ratio. This defines the area of the workspace where the transformation between the observed features and the reference can be seen as a pure rotation within a given threshold and for a desired inlier ration (100%, 70%, and 50%). Intuitively, a lower demanded inlier ratio will relax the needs on a precise threshold. Fig. 7(b) depicts an alternative visualization of the results from Fig. 7(a). It shows the ratio of outliers by fixing the rotation threshold. The darker color indicates lower outlier ratio, i.e., potentially better matching results. Taking the lowest figure in Fig. 7(b) as an instance, the dark area implies that by allowing a rotation threshold 1.5 rad, potentially sufficient numbers of matches can be retrieved from

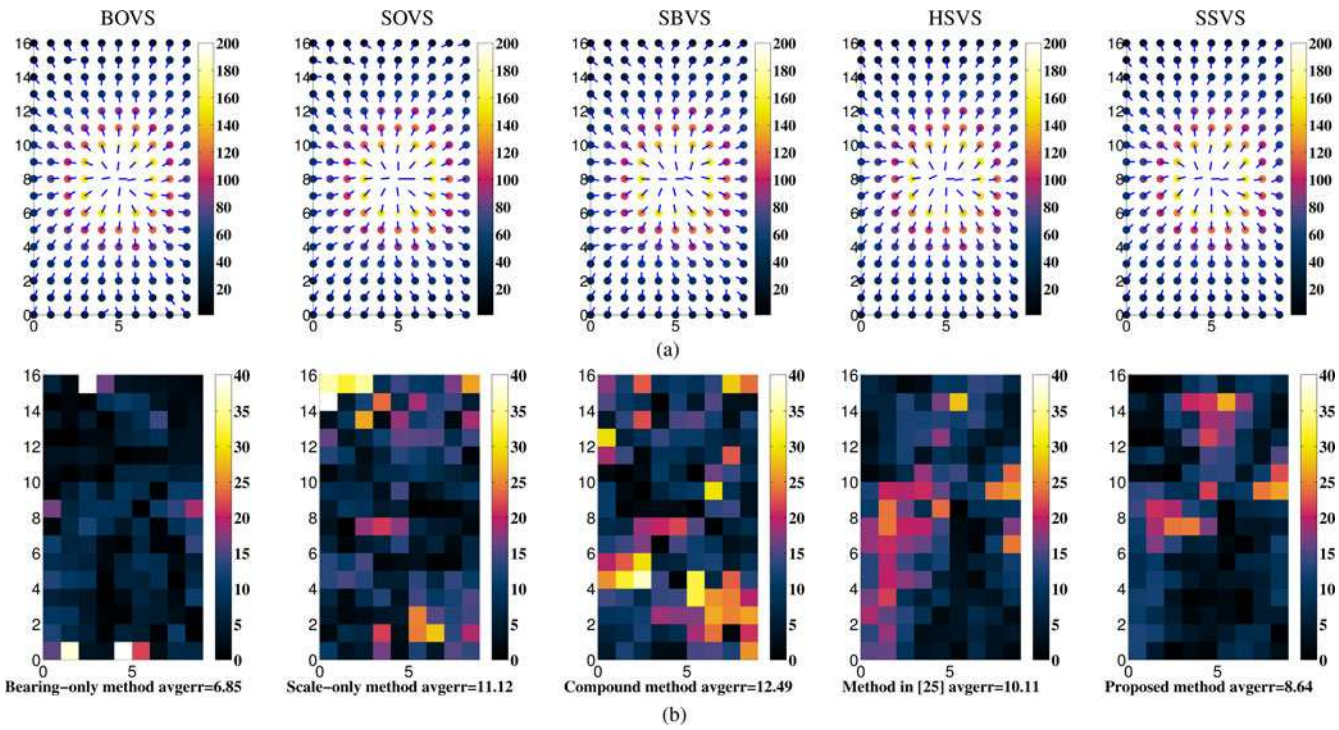


Fig. 9. Homing vectors and error analysis referring to (5.8). The color-map in the first row indicates the number of matched key points; in the second row, the color-map indicates the average angular error in degree. (a) Homing Vectors. (b) AAE analysis (in degree).

a large area around the home position, though it may lead to a high false negative ratio.

For real applications, the rotation threshold needs to be tuned according to the characteristic of the feature distribution in the test environment. Empirically, we choose a threshold of 1.0 rad, resulting in good outlier rejection performance, while still keeping an inlier ratio of more than 80% in the tests on the dataset introduced in Sections IX and X.

IX. Results on an Indoor DataSet

In order to compare all related methods under the same conditions, we test on a widely cited dataset provided by Vardy [32]. The datasets are a collection of panoramic images (some with unwrapped version) obtained from typical indoor environments, plus the calibration information and pose information. Some of the datasets includes dynamic objects in the scene, such as a mobile chair, etc. All the raw images have a resolution of 640×480 (unwrapped image: 561×81) and the actual intervals between the two nearest nodes are equal constants, which is typically 30 cm. An instance from the database is shown in Fig. 8. By taking position (5,8) of A1originalH dataset as the reference home, the homing vectors calculated from other images in the same dataset using different methods are shown as Fig. 9(a). The color of the filled circles indicates the differences in number of matched features. It is interesting to see that SSVS exhibits clean behavior pointing toward the home position, even when the matching ratio is low.

According to the comparison done in [26], the total average angular error (TAAE) can be an important statistic result when evaluating the homing ability. The overall average angular error (AAE) can be obtained as follows:

$$\mathrm{AAE}(\mathbf{ss}) = \frac{1}{mn}\sum_{x=1}^{m}\sum_{y=1}^{n}\mathrm{AE}(\mathbf{ss}, \mathbf{cv}_{xy})$$

where AE is the absolute angular error between the computed homing vector and the ground truth. The subscript $\mathbf{ss}$ and $\mathbf{cv}$ stands for saved scene and current view, respectively. For the entire image database $\mathbf{db}$, the TAAE computes the overall average of $\mathrm{AAE}(\mathbf{ss})$ as follows:

$$\mathrm{TAAE}(\mathbf{db}) = \frac{1}{mn}\sum_{x=1}^{m}\sum_{y=1}^{n}\mathrm{AAE}(\mathbf{ss}_{xy}).$$

The AAE for each position of A1originalH is illustrated in Fig. 9(b).

Test results on the full Vardy datasets are illustrated in Table I, where we show extended statistics of the angular error, such as maximums, minimums, and standard derivations. Darker background on specific numbers mark the best performance of the row. The "*" after the name of datasets indicate that those datasets are unwrapped by detection and operation on the largest circle of the panoramic images. In terms of precision, SSVS and BOVS show the best performance in general. However, we must notice that BOVS requires a good enough estimation of the global heading, which is in general vague using visual compass methods [31]. Moreover, since SSVS and HSVS hold the lowest computational complexity, the advantage of SSVS is revealed.

TABLE I
ERROR ANALYSIS FOR ALL THE ALGORITHMS (IN DEGREE)

Database	BOVS				SOVS				SBVS				HSVS				SSVS			
	TAAE	Max	Min	StdVar	TAAE	Max	Min	StdVar	TAAE	Max	Min	StdVar	TAAE	Max	Min	StdVar	TAAE	Max	Min	StdVar
A1originalH	15.16	50.07	6.52	6.50	14.99	31.90	9.32	3.97	16.04	42.21	10.98	3.43	15.79	30.68	8.97	4.67	12.71	24.98	8.06	3.41
Arboreal*	13.95	35.20	7.06	3.87	20.50	31.63	12.70	4.40	21.26	34.76	12.70	4.13	14.41	24.61	9.63	2.92	12.81	20.44	7.53	2.71
Hall1*	12.97	29.55	6.95	4.67	23.18	35.25	14.28	4.71	24.29	36.97	14.34	5.69	11.73	21.46	8.33	2.03	11.15	19.12	7.55	2.05
Hall2*	21.04	42.03	14.02	4.55	34.28	45.29	25.55	3.91	32.69	41.99	24.83	3.83	17.78	26.01	10.63	3.03	15.24	24.70	9.46	3.08
KitchenH	19.33	40.29	12.17	5.28	26.31	40.60	17.68	5.04	23.63	36.90	14.38	4.58	21.48	38.99	13.34	5.29	20.54	36.94	13.20	4.44
Roeben1H	27.68	48.50	15.17	5.89	25.86	42.11	14.51	5.68	24.74	40.89	13.00	5.71	21.69	42.38	11.01	6.28	19.50	38.56	8.15	6.52
CHall1H	12.21	28.09	6.39	3.96	22.34	37.47	14.20	4.57	22.50	35.85	15.13	4.05	12.26	20.10	8.23	1.94	11.82	18.94	8.35	1.92
CHall2H	18.45	47.63	10.69	5.70	34.16	55.60	24.62	5.43	31.50	49.77	20.83	5.09	19.89	33.63	12.91	3.91	16.63	27.63	9.91	3.05
Chair*	17.66	29.10	8.77	4.15	24.71	41.04	13.08	5.36	23.32	38.03	12.45	4.20	19.28	36.75	11.49	4.28	18.28	35.55	9.53	4.39

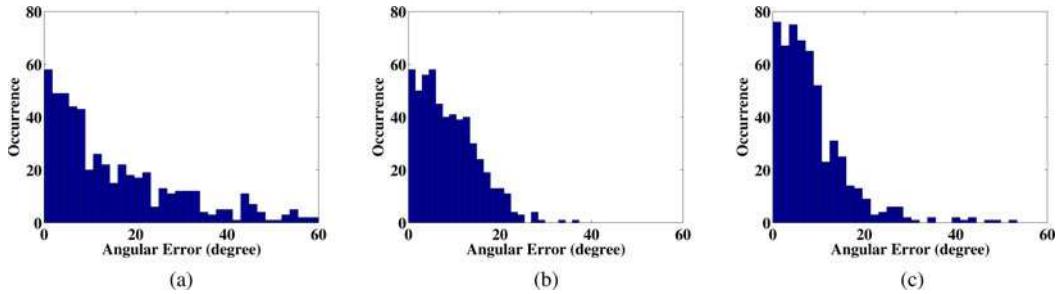


Fig. 10. Histogram of angular errors for the selected algorithms under the test condition with moving people. (a) BOVS. (b) HSVS. (c) SSVS.

Fig. 11. Sample image of the dataset with arbitrarily moving objects.

X. RESULTS OF THE REAL-TIME HOMING TEST

A. Homing Vectors to One Reference

In order to show the performances in a more dynamic environment, a dataset taken together with vicon motion capture system is used for further evaluation. In this test, four people are continuously moving in the scene of which the robot is taking the data. A sample image is shown in Fig. 11.

The calculated homing vectors taking (0,0) position as reference home is depicted in Fig. 12, using SSVS. We could see that the robot trajectory is arbitrary. Thanks to the motion tracking system, the 6 DoF ground truth is simultaneously recorded. It can be noticed that even with relatively low numbers of matched feature points, the robot can show reasonable homing directions using SSVS.

As comparison, only the outperforming algorithms from previous tests—BOVS, SSVS, and HSVS are carried out on this dataset due to the limited space. Concerning BOVS requires global heading estimation, we carry out the evaluation for BOVS using the ground truth heading information. In practice, visual compass is required to provide such information, leading to worse performance. For related works of visual compass, see [31]. The histograms of the angular errors for the three methods are shown in Fig. 10. It indicates that the SSVS has the best precision and BOVS performs the worst even with ground truth heading information. This result shows SSVS is better fit for the robot navigation task in dynamic environments for real-time applications.

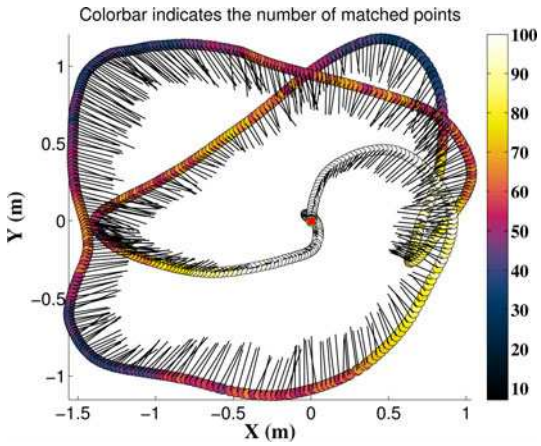


Fig. 12. Results of SSVS with moving people in the scene.

B. Experiment of Outlier Rejection

The outlier rejection method is the key to ensuring the correctness of generated homing vectors in dynamic environments. We also compare the cases with and without the proposed outlier rejection method. The statistics of the generated angular errors show the effect of the proposed outlier rejection method, as depicted in Fig. 13. It shows that the implementations with the proposed outlier rejection have lower error mean and smaller derivation. Regarding the simple assumption, the additional computation that is required for outlier rejection is minor. Therefore, such an algorithm is generally feasible for all

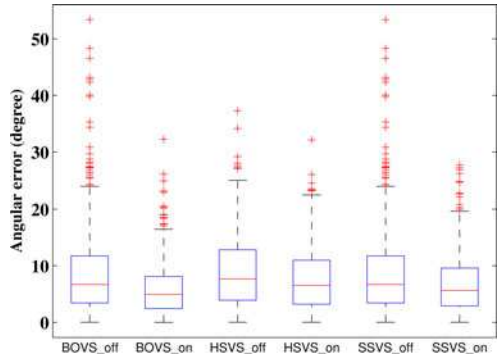


Fig. 13. Error comparison in the cases with/without outlier rejection. The labels marked with _on indicate the boxplot with outlier rejection and _off indicates the results calculated from the raw matching result.

similar applications using key-point features that are obtained from omnidirectional cameras.

XI. Conclusion and Future Work

In this paper, we have presented a visual homing framework using visual servoing, which is based on the scale and bearing measurements of popular visual features such as SIFT. After showing how these measurements could be used in the standard visual servoing framework, we proposed a simplified controller with a complexity linear with the number of observed features. We have demonstrated the usability of scale-based visual servoing, and we have shown that our simplified approach is stable and offers better performance than other methods based on the results on standard datasets and experiments in an environment with dynamic objects. We also demonstrated the necessary observability and stability analysis.

Overall, the proposed approach shows certain robustness to dynamic changes in the environment. Similar to any other visual-based homing solution, our approach is sensitive to strong illumination changes, etc., which may completely destroy the appearance. The extended results for a navigation framework using the proposed algorithms are provided in our recent report [33]. Concerning future work, we are considering applying it to outdoor navigation on terrain manifold and extending it to 3-D space for unmanned aerial vehicles.

Appendix
Public Dataset

A compact dataset has been captured during the experiment of this study. It is comprised of raw and unwrapped panoramic images captured by an omnidirectional camera. The raw images have a resolution 640×480, and the unwrapped version has a resolution of 1014×172. Moreover, the dataset provides pose ground truth with sub-millimeter precision, supported by a vicon motion capture system. A summary is shown in Table II. Please refer to http://www.asl.ethz.ch/people/lium/personal for more information.

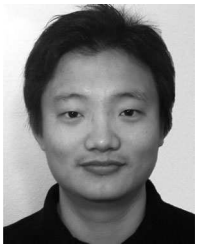
TABLE II
Available Public Dataset Related to This Study

Data-set	#Images	Comments	Potential Purpose
CleanRun	876	No dynamic objects	Function test
FourPeople	653	Four moving people	Robustness test
StillCamera	379	Four moving people; the camera keeps position.	Outlier rejection etc.

References

[1] M. Liu, C. Pradalier, F. Pomerleau, and R. Siegwart, "Scale-only visual homing from an omnidirectional camera," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 3944–3949.
[2] R. Basri, E. Rivlin, and I. Shimshoni, "Visual homing: Surfing on the epipoles," *Int. J. Comput. Vis.*, vol. 33, no. 2, pp. 117–137, 1999.
[3] M. Franz and H. Mallot, "Biomimetic robot navigation," *Robot. Auton. Syst.*, vol. 30, no. 1–2, pp. 133–154, 2000
[4] A. Angeli and J.-A. Meyer, "Incremental vision-based topological SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2008, pp. 22–26.
[5] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proc. Nat. Conf. Artif. Intell.*, 2002, pp. 593–598.
[6] S. Thrun. (1998). Learning metric-topological maps for indoor mobile robot navigation. *Artif. Intell.* [Online]. *99(1)*, pp. 21–71. Available: http://www.sciencedirect.com/science/article/B6TYF-3T0XP9D-2/2/c696e28377beaa9998c9a067cb030cbb
[7] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.
[8] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," *IEEE Trans. Robot. Autom.*, vol. 8, no. 3, pp. 313–326, Jun. 1992.
[9] F. Chaumette and E. Malis, "2 1/2 D visual servoing: A possible solution to improve image-based and position-based visual servoings," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2002, vol. 1, pp. 630–635.
[10] B. Cartwright and T. Collett, "Landmark maps for honeybees," *Biol. Cybern.*, vol. 57, no. 1, pp. 85–93, 1987.
[11] M. Franz, B. Scholkopf, H. Mallot, and H. Bulthoff, "Where did I take that snapshot? Scene-based homing by image matching," *Biol. Cybern.*, vol. 79, no. 3, pp. 191–202, 1998.
[12] M. Liu, C. Pradalier, Q. Chen, and R. Siegwart, "A bearing-only 2D/3D-homing method under a visual servoing framework," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2010, pp. 4062–4067.
[13] P. Corke, "Mobile robot navigation as a planar visual servoing problem," in *Robotics Research: The Tenth International Symposium.* New York, NY, USA: Springer-Verlag, 2003, vol. 6, pp. 361–372.
[14] D. Lambrinos, R. Möller, T. Labhart, R. Pfeifer, and R. Wehner, "A mobile robot employing insect strategies for navigation," *Robot. Auton. Syst.*, vol. 30, no. 1, pp. 39–64, 2000.
[15] T. Goedeme, T. Tuytelaars, L. Van Gool, D. Vanhooydonck, E. Demeester, and M. Nuttin, "Is structure needed for omnidirectional visual homing?" in *Proc. IEEE Int. Symp. Comput. Intell. Robot. Autom.*, 2005, pp. 303–308.
[16] J. Lim and N. Barnes, "Robust visual homing with landmark angles," in *Proceedings of Robotics: Science and Systems.* New York, NY, USA: MIT Press, 2009.
[17] M. Aranda, G. Lopez-Nicolas, and C. Sagues, "Omnidirectional visual homing using the 1D trifocal tensor," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2010, pp. 2444–2450.
[18] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. vol. 2, Cambridge, U.K.: Cambridge Univ. Press, 2000, ch. 15.
[19] H. Becerra, G. López-Nicolas, and C. Sagüés, "Omnidirectional visual control of mobile robots based on the 1d trifocal tensor," *Robot. Auton. Syst.*, vol. 58, no. 6, pp. 796–808, 2010.
[20] H. Becerra, G. Lopez-Nicolas, and C. Sagüés, "A sliding-mode-control law for mobile robots based on epipolar visual servoing from three views," *IEEE Trans. Robot.*, vol. 27, no. 1, pp. 175–183, Feb. 2011.
[21] G. López-Nicolis, J. Guerrero, and C. Sagüés, "Multiple homographies with omnidirectional vision for robot homing," *Robot. Auton. Syst.*, vol. 58, no. 6, pp. 773–783, 2010.
[22] A. Cherubini and F. Chaumette, "A redundancy-based approach for obstacle avoidance in mobile robot navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 5700–5705

[23] T. Nierobisch, J. Krettek, U. Khan, and F. Hoffmann, "Optimal large view visual servoing with sets of sift features," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 2092–2097.
[24] F. Hoffmann, T. Nierobisch, T. Seyffarth, and G. Rudolph, "Visual servoing with moments of sift features," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, 2006, vol. 5, pp. 4262–4267.
[25] A. Vardy and F. Oppacher, "A scale invariant local image descriptor for visual homing," in *Biomimetic Neural Learning for Intelligent Robots* (Lecture Notes Computer Science). vol. 3575, New York, NY, USA: Springer, 2005, p. 362
[26] D. Churchill and A. Vardy, "Homing in scale space," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2008, pp. 1307–1312.
[27] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," in *Proc. Eur. Conf. on Comput. Vis.*, 2006, vol. 3951, pp. 404–417.
[28] A. Negre, C. Braillon, J. Crowley, and C. Laugier, "Real-time time-to-collision from variation of intrinsic scale," *Algorithm. Found. Robot. VI*, p. 75, 2005
[29] F. Chaumette and S. Hutchinson, "Visual servo control, Part I: Basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, pp. 82–90, Dec. 2006.
[30] E. Malis, F. Chaumette, and S. Boudet, "2 1/2 D visual servoing," *IEEE Trans. Robot. Autom.*, vol. 15, no. 2, pp. 238–250, Apr. 1999.
[31] A. Vardy, "A simple visual compass with learned pixel weights," in *Proc. IEEE Can. Conf. Electr. Comput. Eng.*, 2008, pp. 001-581–001-586.
[32] A. Vardy. (2012). "Panoramic image database," [Online]. Available: http://www.ti.uni-bielefeld.de/html/research/avardy/index.html
[33] M. Liu, C. Pradalier, F. Pomerleau, and R. Siegwart, "The role of homing in visual topological navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 567–572.



Ming Liu (S'12) received the B.A. degree in automation from Tongji University, Shanghai, China, in 2005. During his Master's study at Tongji University, he was a Visiting Scholar with Erlangen-Nünberg University and Fraunhofer Institute IISB, Germany, for one year. As a result of his performance and under a special grant, in 2007, he was admitted as a Ph.D. student at Tongji University without attaining the master's degree. Since 2009, he has been working toward the Ph.D. degree with the Department of Mechanical Engineering, ETH Zürich, Zurich, Switzerland.

His current research interests include autonomous mapping, visual navigation, environment modeling, etc.

Mr. Liu received the Best Student Paper Award from the IEEE International Conference on Multisensor Fusion and Information Integration in 2012.

Cédric Pradalier (M'07) received the Ph.D. degree in 2004 from the National Polytechnic Institute of Grenoble, Grenoble, France, on the topic of autonomous navigation of a small urban mobility system. He received the Ingénieur degree from the National Engineering School for Computer Science and Applied Math, Grenoble.

Since November 2007, he has been the Deputy Director with the Autonomous Systems Lab, ETH Zürich, Zurich, Switzerland. He is also the Technical Coordinator of the V-Charge Project (IP, 2010-2014) and is involved in the development of innovative robotic platforms such as autonomous boats for environment monitoring or prototype space rovers funded by the European Space Agency. He is a founding member of the ETH start-up Skybotix, within which he is responsible for software development and integration. From 2004 to 2007, he was a Research Scientist with The Commonwealth Scientific and Industrial Research Organisation, Highett, Vic., Australia. He was then involved in the development of software for autonomous large industrial robots and an autonomous underwater vehicle for the monitoring of the Great Barrier Reef, Australia.

Roland Siegwart (M'90–SM'03–F'08) received the Diploma degree in mechanical engineering and the Ph.D. degree in mechantronics from the Eidgenössische Technische Hochschule (ETH) Zurich, Zurich, Switzerland, in 1983 and 1989, respectively.

From 1989 to 1990, he was a Postdoctoral Fellow with Stanford University, Stanford, CA, USA. He was then a part time R&D Director with MECOS Traxler AG, Winterthur, Switzerland. He was also a Part-Time Lecturer and the Deputy Head of the Institute of Robotics, ETH Zurich, where, since July 2006, he has been a Full Professor of autonomous systems. In 1996, he was an Associate Professor, and later, a Full Professor for autonomous microsystems and robots with the Ecole Polytechnique Federale de Lausanne (EPFL), Lausanne, Switzerland, where he was Co-Initiator and the Founding Chairman of the Space Center EPFL and the Vice Dean of the School of Engineering. In 2005, he held visiting positions with the National Aeronautics and Space Administration (NASA) Ames, Moffett Field, CA, USA, and Stanford University, Stanford, CA. He leads a research group of around 35 people working in the field of robotics, mechantroics, and product design. He is a Coordinator of two large European projects, plays an active role in various rover designs of the European Space Agency, and is a Co-Founder of several spin-off companies.

Dr. Siegwart is a member of the Swiss Academy of Engineering Sciences and a Board Member of the European Network of Robotics. From 2004 to 2005, he was the Vice President for Technical Activities and a Distinguished Lecturer from 2006 to 2007. He was an AdCom Member (2007–2009) of the IEEE Robotics and Automation Society.

Appendix B

Garneau et al. [2013]

Marie-Ève Garneau, Thomas Posch, Gregory Hitz, François Pomerleau, Cédric Pradalier, Roland Siegwart, and Jakob Pernthaler. Short-term displacement of planktothrix rubescens (cyanobacteria) in a pre-alpine lake observed using an autonomous sampling platform. *Limnography and Oceanography*, 58(5), 2013

Limnol. Oceanogr., 58(5), 2013, 1892–1906

doi:10.4319/lo.2013.58.5.1892

Short-term displacement of *Planktothrix rubescens* (cyanobacteria) in a pre-alpine lake observed using an autonomous sampling platform

Marie-Ève Garneau,[1,*] Thomas Posch,[1] Gregory Hitz,[2] François Pomerleau,[2] Cédric Pradalier,[2] Roland Siegwart,[2] and Jakob Pernthaler [1]

[1] Limnological Station Kilchberg, Institute of Plant Biology, University of Zurich, Zurich, Switzerland
[2] Autonomous Systems Lab, Swiss Federal Institute of Technology (ETH) Zurich, Zurich, Switzerland

Abstract

Short-term changes in temporal and spatial distributions of the toxic cyanobacterium *Planktothrix rubescens* in Lake Zurich were investigated using high-resolution data sets acquired with an autonomous surface vessel (ASV). Data profiles were recorded with a multi-parameter probe while the ASV navigated along 1.5 km toward a waypoint located on the other side of the lake by using a global positioning system. We deployed the ASV seven times on five consecutive days during the stratification period (July 2011) to generate cross-sectional views of temperature, light, oxygen, and phycoerythrin and chlorophyll fluorescence from surface to 18 m. The data were also used to compute daily photosynthetic rates. Data showed a daily reshaping of the *P. rubescens* distribution in the metalimnion on both horizontal and vertical axes, from patches to a shore-to-shore spreading. A thermocline depression observed after 16 h of sustained winds forced the accumulation of *P. rubescens* on the downwind shore. The compression of the metalimnion and its downward shift by 6 m within 24 h suggested the modulation of a longitudinal seiche following the wind event. This passive transport of the metalimnetic *P. rubescens* population resulted in a 90% light reduction, and a decrease of the averaged daily photosynthetic rate from +21 mmol O_2 m^{-2} d^{-1} to −10 mmol O_2 m^{-2} d^{-1}. Negative photosynthetic rates were computed on 2 d out of 5 d, meaning that the transport of *P. rubescens* by seiches significantly affected the balance between oxygen production and utilization in Lake Zurich, especially because it is the dominant primary producer.

In the pre-alpine Lake Zurich, Switzerland, the species *Planktothrix rubescens* dominates the biomass of primary producers (Micheletti et al. 1998; Bossard et al. 2001). The ecological success of *P. rubescens* lies in its adaptation to the seasonal stratification that takes place in the clear mesotrophic waters of Lake Zurich (Walsby and Schanz 2002). Not only does *P. rubescens* thrive in the metalimnion during late summer and early autumn, but the cyanobacterium also outcompetes other phytoplankton species under the low-irradiance conditions that prevail in that layer by maximizing the absorption of green light with phycoerythrin pigments (Davis and Walsby 2002; Oberhaus et al. 2007). *P. rubescens* can also regulate its buoyancy with gas vesicles to maintain and adjust its vertical position according to the irradiance, preferably at depths above its photosynthetic compensation point where it has the biggest competitive advantage (Walsby et al. 2001). The prevalence of *P. rubescens* in Lake Zurich, a drinking water reservoir for 1.5 million people, is a serious issue because it produces cyclic hepatotoxins called microcystins, including the very acute form [D-Asp^3,(*E*)-Dhb^7]microcystin-RR (Blom et al. 2001). Microcystins represent major threats to domestic animals as well as considerable hazards for human health through water consumption and bathing (Codd et al. 2005). Even though programs for water quality restoration by nutrient control have been undertaken since the 1950s in Lake Zurich (Bossard et al. 2001), *P. rubescens* populations have shown no reduction, but rather an increase during the past decades (Posch et al. 2012).

The spatial distribution of cyanobacteria varies vertically as well as horizontally in shallow (Moreno-Ostos et al. 2009; Pobel et al. 2011) and in deep lakes (Cuypers et al. 2010; Salcher et al. 2011). In the epilimnion, spatial variations may occur on short time scales (i.e., from hours to days) because most of these changes arise from meteorological conditions. For example, during quiet days of minimal mixis, cyanobacteria can form surface scums (Huisman and Hulot 2005), and whenever winds generate surface currents, these scums may accumulate locally (Pobel et al. 2011). In the metalimnion, physical changes may also arise from weather-forced events, such as when steady winds blow over a stratified lake and pile up surface waters at the lee end shore (Boegman 2009). This water buildup pushes down the thermocline on the leeward shore and creates a compensatory upward movement of the isotherms at the windward shore (Horn et al. 1986). Once winds stop, the thermal layers of the lake slide over each other to redistribute a new equilibrium generating an oscillation, called an internal seiche, which produces vertical movements of water at depth, but produces very little motion at the surface (Boegman 2009). These short-term, wind-induced events generate vertical displacements of metalimnetic populations of cyanobacteria, causing episodic changes in the irradiance that reaches the cells, which in turn modify photosynthetic rates with cascading effects on the ecosystem of small and large lakes (Cuypers et al. 2010; Pannard et al. 2011).

Short-term variations in the vertical distribution of *P. rubescens* due to displacement by seiches have been scarcely

* Corresponding author: me.garneau@gmail.com

Fig. 1. The autonomous sampling vessel, which is a catamaran 2.5 m long and 1.8 m wide called *Lizhbeth,* during a sampling mission on Lake Zurich. The winch system is used to lower a light sensor and a multi-parameter probe, which is inserted in the plane-shaped fiberglass casing. The casing has elongated perforations to allow the water to reach the sensors. The front tip of the casing is a 4.2 kg steel weight.

documented (Thomas and Märki 1949; Thomas 1950; Cuypers et al. 2010; Carraro et al. 2012), and there is no information on the variations in light conditions that are experienced by *P. rubescens* in situ. Comprehensive calculations indicated that vertical shifts significantly affect the net photosynthetic production balance because this metalimnetic population is located close to its photosynthetic compensation point (Walsby et al. 2001). Nonetheless, the ecological effects of such punctual events are poorly described because the rapid and occasional nature of their advent hinders their documentation using traditional sampling methodologies. More studies demonstrate the effectiveness of high-frequency data sampling by fixed fluorometer probes to monitor harmful cyanobacterial blooms (Leboulanger et al. 2002; Brient et al. 2008; McQuaid et al. 2011) but broader spatial information is still limited (Porter et al. 2009).

A promising avenue to address harmful cyanobacteria ecology from both a spatial and a temporal angle is the application of autonomous mobile robots which, when equipped with multiple sensors, can accurately and rapidly measure limnological parameters at biologically relevant spatiotemporal scales. The field of autonomous mobile robotics has gained more interest in autonomous surface vessels (ASV), and an increasing number of platforms are designed for lake-ecosystem studies (Caron et al. 2008; Dunbabin et al. 2009; Ferri et al. 2011). So far, none of these mobile deployments have specifically aimed for the monitoring of harmful cyanobacteria. Our newly designed ASV provides a sampling method for high temporal and spatial resolutions, both horizontally and vertically, and is thereby applicable to the variability of lake hydrology and of microbiological processes. The vessel holds a custom-designed winch employed to lower a commercial multi-parameter probe (Fig. 1). It can be rapidly deployed in Lake Zurich, thereby allowing for the monitoring of short-term variations in *P. rubescens* distribution.

Our objective was to document the role of episodic lake-water movements on the light-based positioning of *P. rubescens* in the water column and the consequences for its growth. We hypothesized that physical processes substantially affect the net photosynthetic rate of the *P. rubescens* population. We deployed our ASV to measure the underwater irradiance and to locate depths where *P. rubescens* stratified, and we then compared the measured values with the theoretical depth where the daily insolation would support its neutral buoyancy.

Methods

Study site and sample collection—Lake Zurich (Fig. 2) is a large and deep mesotrophic lake ($\sim$ 10–20 μg L^{-1} total phosphorus) located at 406 m above sea level on the Swiss Plateau, at the northern edge of the Alps (Bossard et al. 2001). The lake has a surface area of 68 km^2, a maximal depth of 136 m (Bossard et al. 2001), and 30 km long thalweg path (i.e., the deepest continuous line in the lake channel; Horn et al. 1986). The lake basin is mainly exposed to northeasterly and southwesterly winds, which are stormy in winter and spring (MeteoSwiss; http://www.meteoswiss.admin.ch/files/kd/normwerte/norm8110/windrosen/fr/SMA.pdf). Summer and autumn are comparatively calm, but various local winds circulate from the surrounding hills toward the lake at night, and from the lake back to the hills during daytime (Hantke et al. 1979). Summertime is also characterized by short episodes of strong west winds and thunderstorms that usually occur in the late afternoon (Hantke et al. 1979). The wind stress at the surface of the lake is non-uniform and quite variable because of changes in the orientation in the thalweg direction and the surrounding topography, resulting in several longitudinal internal seiche modes in Lake Zurich (Horn et al. 1986).

Meteorological parameters were obtained from the website of the Zurich Water Police (Zurich Wasserschutzpolizei, http://www.tecson-data.ch/zurich/seepozh/mythenquai.html). Continuous records of global solar irradiance (Kipp and Zonen CM3 pyranometer in W m^{-2}), wind speed (m s^{-1}), and wind direction ($^\circ$) were measured at 10 min intervals from 10 July, which is the day prior sampling, to 17 July. The weather station is located at 4.1 km north of the sampling transect (Fig. 2) and meteorological data are presented in Fig. 3.

Robotic approaches and water sampling—The ASV collected vertical data profiles using a YSI-6600 multi-parameter probe (YSI Incorporated) equipped with an array of sensors while it navigated autonomously along a linear path following pre-defined global positioning system (GPS) waypoints. The precise GPS-based navigation ensured that the positions of the measurements are reproducible from one sampling run to the other, which permitted data comparisons between all runs. The conception and manufacturing of the platform, the software design as well as the robotic aspects are fully described in Hitz et al. (2012). The navigation speed for sampling was set for maximal data coverage of the studied area while

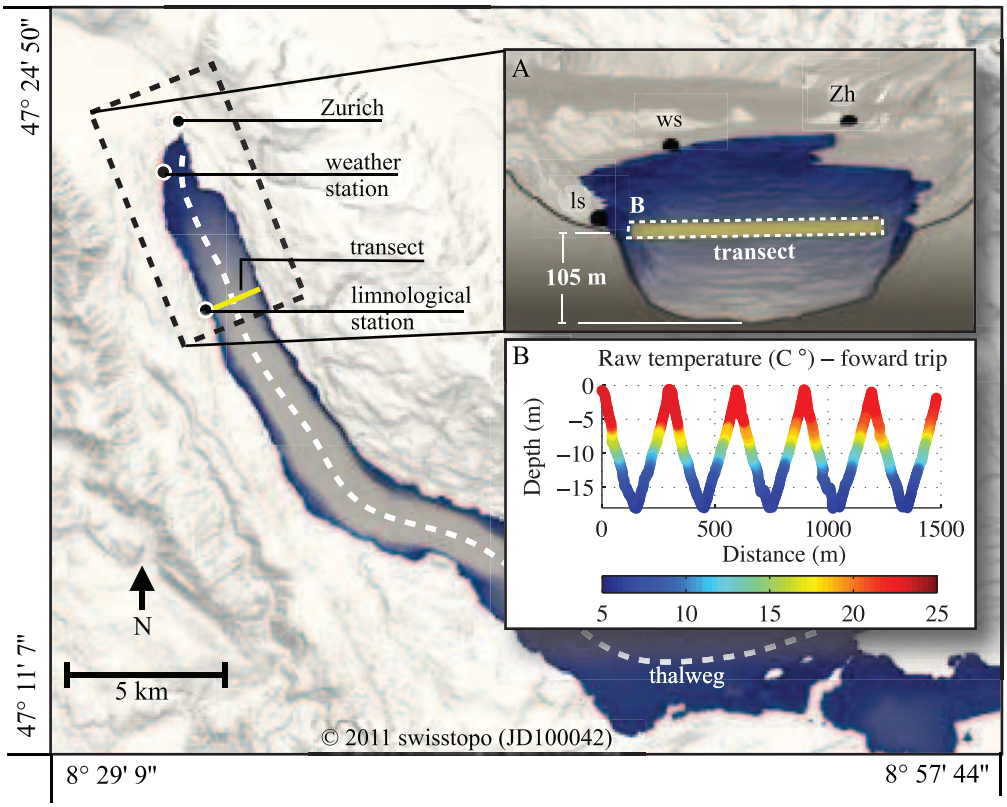


Fig. 2. Map of Lake Zurich, Switzerland, and its surrounding topography (Source: *Bundesamt für Landestopografie swisstopo* [Art. 30 GeoIV]: 5704 000 000). Also shown are the location of the sampling transect, the weather station, and the thalweg path. (A) Schematic illustration of the 1.5 km transect in Lake Zurich, where the probe is lowered and raised between 2 and 18 m. Zh stands for Zurich, ws for weather station, and ls for limnological station. (B) Plot of the raw data recorded by the temperature sensor during the forward trip toward the east shore that illustrates the zigzag path of the probe.

considering hardware-related constraints, such as the sensor measurement frequencies and the capacity of the two batteries. A sailing speed of 0.7 m s^{-1} and a vertical deployment speed of 0.1 m s^{-1} for the probe, which measures at a rate of 0.5 Hz, was the most advantageous trade-off for a round-trip sampling from east to west along the 1.5 km transect (Fig. 2A). The ASV sailed the transect in 40 min while alternately lowering and raising the probe, from the surface layer down to the upper part of the hypolimnion (2–18 m), to capture the entire metalimnion layer where *P. rubescens* accumulates during summer and autumn. The combination of the horizontal and vertical motions of the probe results in a zigzag sampling path (Fig. 2B). For a given sampling run, the raw measurements of the two inverse zigzag paths of the probe (i.e., one on the forward trip and another on the return trip) were combined to enhance the spatial coverage and to increase the accuracy of the interpolation method (details below). A complete field survey took 80 min, which is $< 10\%$ of the shortest seiche period (17 h) in Lake Zurich (Horn et al. 1986).

The sampling campaign took place from 12 to 16 July 2011 along a straight-line transect starting on the west shore (47.320119°N, 8.554894°E), and ending on the east shore (47.326092°N, 8.572475°E; Fig. 2). The ASV was deployed daily, and two times on 12 and 14 July to evaluate the variations that may occur within a day. Sampling took place ~ 2 h before and after noon, which was when the sun was at its zenith and when the irradiance was maximal. The bell-shaped curves of irradiance indicated that the light at 10:00 h and 15:00 h was usually similar (Fig. 3B). Environmental data were collected every 2 s using sensors for depth (m), temperature (°C), dissolved oxygen concentration (mg L^{-1}) and saturation level (%), chlorophyll fluorescence (relative fluorescence unit, RFU), which is a proxy for the total phytoplankton biomass, and finally for phycoerythrin fluorescence (also in RFU), which is a proxy for *P. rubescens* biomass. Additionally, a spherical quantum sensor (LI-COR) was installed on the probe to measure the in situ photosynthetically active radiation (PAR; μmol m^{-2} s^{-1}). A second sensor placed on the boat measured the PAR above the surface for reference. All sensors were calibrated according to the manufacturer's guidelines. Delays in the response of the sensors on the multi-parameter probe were detected because we were deploying them at a faster speed than recommended by YSI Incorporated. The time delay for a measurement was 15 s with the sensor for dissolved oxygen, 14 s for the phycoerythrin sensor, and 2 s for the temperature sensor. The delays in the sensor responses were easily adjusted back to the correct position of the reading, but it was not possible for the dissolved oxygen 6562 Rapid Pulse™

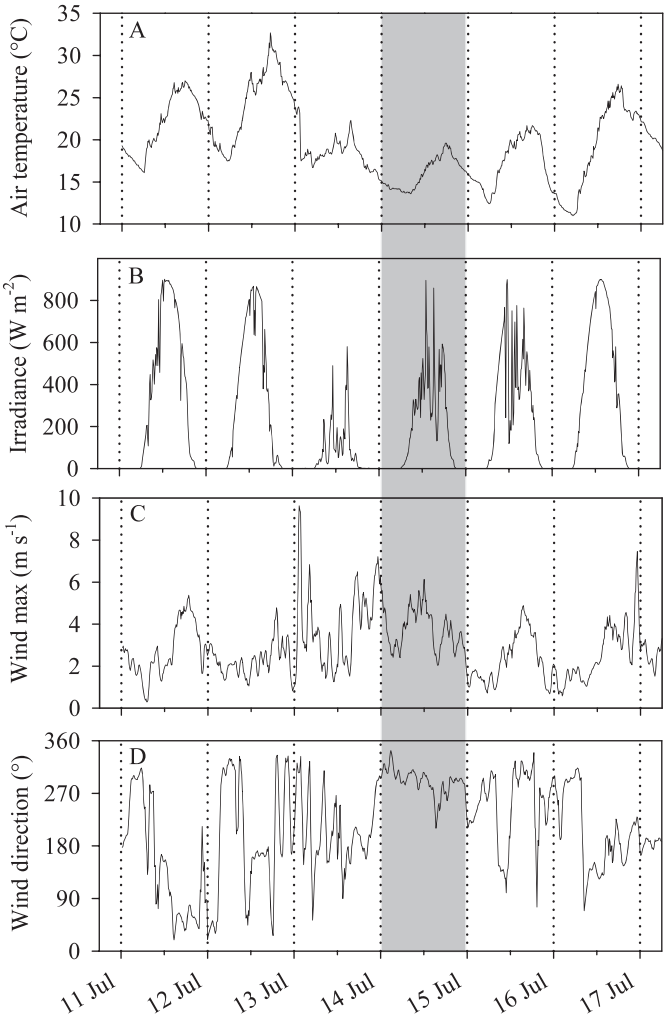


Fig. 3. Meteorological data measured from July 11, 00:00 h to July 17, 23:50 h. Presented are running means of the data recorded every 10 min for (A) air temperature, (B) solar irradiance, (C) wind speed maxima, and (D) wind direction (0° = 360° = north, 90° = east, 180° = south, 270° = west). The grey-shaded area represents the period when sustained winds blew from the northwest for 16 h.

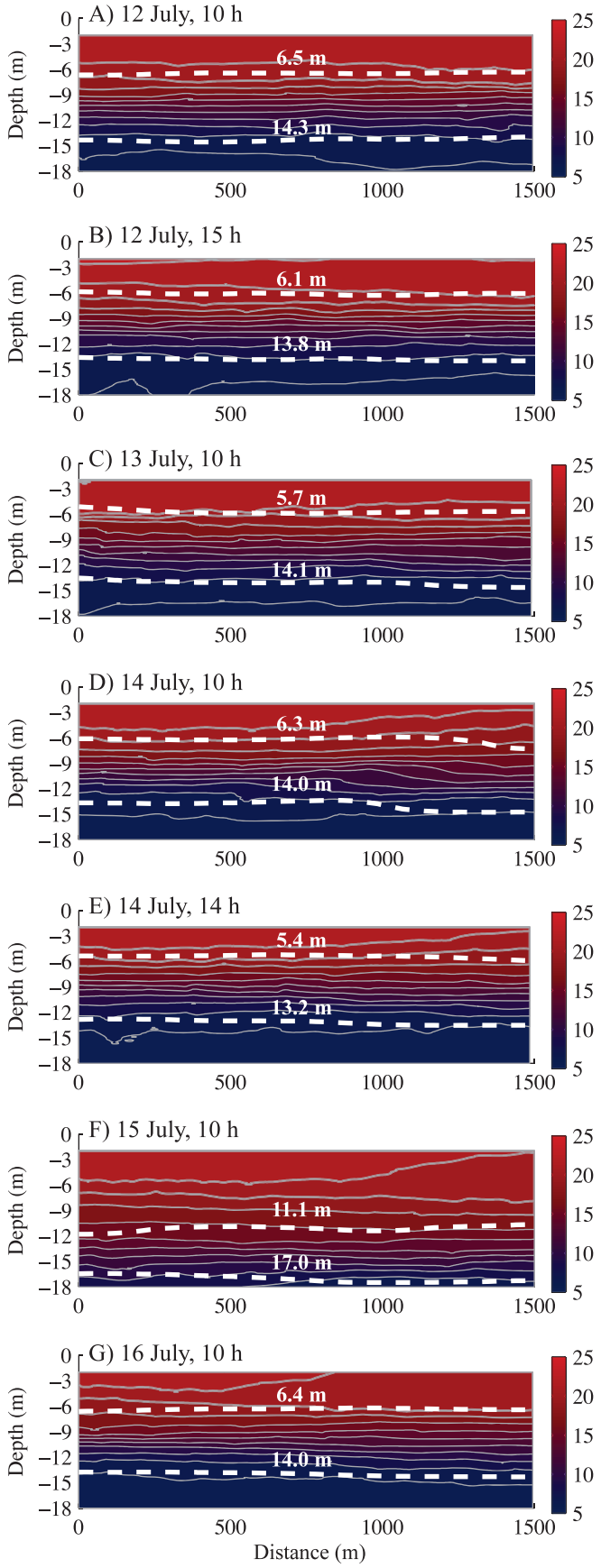
Sensor (an electrochemical sensor) when the probe was moving upward. As a result, the values were inaccurate, and this half of the data set was excluded. A likely explanation for inaccurate oxygen values is that the sensor may respond poorly when passing from a lower oxygen zone to higher oxygen zone. Electrochemical oxygen sensors are influenced by oxygen concentration, as well as changes in temperature, salinity, pressure, and stirring (Tengberg et al. 2003). Thus, the number of data collected (n) from 2 m to 18 m with each sensor amounted to 13,904 (or $>$ 1900 for each sampling run); whereas, the data collected with the dissolved oxygen sensor (n_{DO}) was 6851.

On the first day of the sampling campaign and 1 week after, on 19 July, vertical algal pigment profiles were recorded using a TS-16-12 fluoroprobe (bbe Moldaenke GmbH) that was deployed from a small craft above the deepest point of the lake (47.307011°N, 8.577167°E). The fluoroprobe was calibrated to differentiate between green algae, diatoms, cryptophytes, and *P. rubescens*.

Quantification of Planktothrix *by quantitative PCR for sensor calibration*—The RFU values given by the phycoerythrin fluorescence sensor were considered good estimators of *P. rubescens* biovolume because its phycoerythrin content was shown to be relatively constant throughout seasons (Ernst et al. 2009) and under various light conditions (Skulberg and Skulberg 1985). The sensor was thus calibrated by associating the RFU reading at a given depth to the actual *P. rubescens* abundance, which was determined by quantitative PCR (qPCR) analysis on water samples taken at that same depth. We collected 47 samples at various depths from 0 m to 20 m from 24 March to 03 November 2011. For each depth, two 400 mL volumes were filtered through glass fiber filter membranes. The filters were processed as crude lysates, as in Garneau et al. (2011), and filtered through a 0.22 μm, low-binding cellulose acetate syringe-tip filters to efficiently remove cellular debris and filter remnants. The *P. rubescens* filaments were quantified following the qPCR TaqMan assay on the 16S ribosomal deoxyribonucleic acid (rDNA) gene developed by Ostermaier and Kurmayer (2009). The qPCR approach was calibrated using a four-point 1 : 10 serial dilutions of DNA extracts of a *P. rubescens* culture (strain A7 isolated from Lake Zurich by A. E. Walsby), for which the initial biovolume was 375 mm^3 L^{-1} ($\pm$ 126 mm^3 L^{-1}, $n = 7$). This concentration was determined by microscopic examination of formaldehyde-fixed filaments collected onto black polycarbonate filters (0.22 μm pore size). The autofluorescent filaments under green light were quantified by automated image analysis (Zeder et al. 2010). Counts were translated into biovolumes (Van den Wyngaert et al. 2011).

Calibration curves were loaded for each qPCR run of unknown samples to directly translate threshold cycle (C_t) values into biovolumes. The calibration curve spanned over concentrations from 3.75×10^0 mm^3 L^{-1} to 3.75×10^{-3} mm^3 L^{-1}, which corresponded to 2.34×10^{-5} mm^3 to 2.34×10^{-8} mm^3 of filaments per qPCR reaction. Six calibration curves were run and combined to give a master calibration curve described as $y = -3.24x + 9.36$, $r^2 = 0.96$. The qPCR efficiency (E) was calculated as $E = 100 \times 10^{(-1/m)} - 1$ (where m is the slope of the master calibration curve) and equaled 103%, which is within the acceptable range of 90–110% for assay validation (Invitrogen 2008). The qPCR method accuracy was confirmed by comparing qPCR results with microscopic slides counted in triplicate for 11 lake samples. The standard deviation on biovolumes determined by microscopy was, on average, 30%. For a given sample, the standard deviation on the triplicate qPCR runs was, on average, 16%, and was never above 30%. Correlation analysis showed that the two quantification methods gave very similar abundance values ($m = 0.82$, $r^2 = 0.93$). The linear regression used to convert RFU values into qPCR-based biovolume (BioV in mm^3 L^{-1}) was BioV = 0.806 [RFU] $-$ 0.899 ($r^2 = 0.90$, $n = 47$, $p \leq 0.05$).

Data interpolation—Sensor data were post-processed using a two-dimensional interpolation method to create a



cross-sectional representation. These cross sections were shaped using an anisotropic distance kernel because there was a large difference between the scale of the horizontal axis (1.5 km) and the vertical axis (18 m). Interpolations were calculated over a depth range of 2–18 m using MATLAB. This interpolation procedure was evaluated, and results indicated that the error of the interpolation is contained within the temporal variability observed in the field (Hitz et al. 2012). For a given environmental parameter, each interpolated value is considered representative of a volume that has a length of 30 m, a width of 30 m, and a height of 0.6 m. Using the interpolated temperature profiles, the thermocline stratification index (TSI in °C m^{-1}) was computed as follows:

$$TSI = \Delta T / \Delta h \quad (1)$$

where ΔT is the difference in water temperature and Δh is the depth interval (Yu et al. 2010). The stratification increases with TSI values. The metalimnion boundaries displayed on Figs. 4 and 5 were determined using a gradient value of > 1°C m^{-1}.

Calculations of daily insolation, neutral buoyancy depth, and photosynthetic compensation point—Two different irradiance sensors were used in this study: (1) a quantum LI-COR sensor, which measured the in situ PAR (or visible range, 400–700 nm) in μmol m^{-2} s^{-1}; and (2) a radiometric Kipp and Zonen CM3 pyranometer, which measured the global solar spectrum (305–2800 nm) in W m^{-2} at the surface of the lake. The conversion of these radiometric units into photon units was estimated by the LI-COR manufacturer to be roughly 4.6 μmol m^{-2} s^{-1} $\approx$ 1 W m^{-2}, which is equivalent to 1 J m^{-2} s^{-1} (Biggs 2000). Because about 46% of the total solar energy is emitted in the visible range (Walsby 2001; Blumthaler 2012), a value of 1 J m^{-2} s^{-1} on the CM3 pyranometer should be equivalent to 2.116 μmol m^{-2} s^{-1} on the LI-COR sensor (i.e., 4.6 μmol m^{-2} s^{-1} $\times$ 0.46). Thus, we used the theoretical ratio LI-COR : CM3 of 2.116 μmol J^{-1} (i.e., 2.116 μmol m^{-2} s^{-1} : 1 J m^{-2} s^{-1}) to convert the pyranometer values into PAR equivalent values.

Each 10 min record of the global solar irradiance was first multiplied by 600 s (i.e., 10 min = 10 $\times$ 60 s) to transform values in J m^{-2}. The resulting values were afterward multiplied by 2.116 μmol J^{-1}. The daily irradiance at the water surface Q_0 (in mol m^{-2}) was then calculated from the sum of the 10 min records over a period of 24 h. The daily insolation Q_z at depth z was calculated as follows:

←

Fig. 4. Cross-sectional views of temperature (in °C) along the transects across Lake Zurich between 12 and 16 July 2011, indicated by letters A to G. The dashed white lines indicate the upper and lower boundaries of the metalimnion, and depths indicated on each line are mean values computed over the full transect. The thin grey lines are the temperature isolines.

$$Q_z = Q_0 \frac{I_z}{I_0} \quad (2)$$

where I_z is the irradiance at depth z and I_0 is the irradiance measured just below the water surface, both in μmol m^{-2} s^{-1}.

The neutral buoyancy depth z_N is defined as the depth where 50% of the *P. rubescens* filaments naturally float, which corresponds to the depth where the daily photon insolation Q_N is equal to 0.28 mol m^{-2} (Walsby et al. 2004), and is calculated as follows:

$$z = z_I + \Delta z \frac{\ln\left(\frac{Q_I}{q}\right) - \ln\left(\frac{Q}{q}\right)}{\ln\left(\frac{Q_I}{q}\right) - \ln\left(\frac{Q_{I+1}}{q}\right)} \quad (3)$$

where $z = z_N$ and $Q = Q_N$; Δz is the sampling depth interval; q is a constant equal to 1 mol m^{-2}; z_I and z_{I+1} are, respectively, sampling depths immediately above and below the depth where the insolation equals Q_N; and Q_I and Q_{I+1} are the daily insolation at these depths.

The growth compensation point E_g was defined as the irradiance in the light phase that supports no growth over a cycle of 12 h of light and 12 h of darkness (Davis and Walsby 2002). A E_g value of 0.8245 μmol m^{-2} s^{-1} was determined for a *P. rubescens* culture growing at 15°C during 12 h of light (Davis and Walsby 2002). To obtain the daily photon insolation Q_C that corresponds to the compensation point for *P. rubescens* growth, the E_g value was multiplied by 43,200 s (equivalent to 12 h), which gives a resulting Q_C value of 0.0356 mol m^{-2}. The compensation depth z_C is calculated using Eq. 3 where $z = z_C$ and $Q = Q_C$. Interpolated data were used to calculate z_N and z_C for each sampling run. The resulting data were depicted in Fig. 5.

Calculation of the daily integrals of photosynthesis—Daily integrals of photosynthesis were computed using our vertical profile measurements of *P. rubescens* biovolumes, temperature, and irradiance in Lake Zurich, and following the calculation procedures of Walsby (1997, 2001). The relationship between photosynthesis P and irradiance I is described as follows:

$$P = P_m \left[1 - \exp\left(\frac{-\alpha I}{P_m}\right)\right] + R + \beta I \quad (4)$$

where P is the photosynthetic rate, P_m is the maximum photosynthetic rate, and R is the rate at zero irradiance (all in μmol cm^{-3} h^{-1}); α is the initial slope of the P/I curve at low irradiance and β is the P/I curve slope at high irradiance (both in μmol cm^{-3} h^{-1}[μmol m^{-2} s^{-1}]$^{-1}$); and I is the photon irradiance in μmol m^{-2} s^{-1}.

The photosynthetic coefficients α, β, P_m, and R were determined experimentally by Walsby et al. (2001), where R was the standardized respiration R_s calculated as $-0.11 \times P_m$ (set 1 in Table 1). Among the photosynthetic coefficients available in the literature, the coefficients from Walsby et al. (2001) were the most relevant to our data set for three reasons: (1) the coefficients were measured for a natural population of *P. rubescens* in Lake Zurich; (2) the samples were incubated at 11 m in Lake Zurich, which reflects the in situ light and temperature T' of 15.5°C; and (3) values are means of four P/I curves measured at different time points to account for daily variations in the coefficients. The daily photosynthetic rates computed with the parameter set 1 were considered default values. They were compared to rates computed using two other sets of parameters, also determined for *P. rubescens* from Lake Zurich (Table 1). The coefficient set 2 was calculated from cultures growing at 20°C under controlled light conditions (Walsby 2001); whereas, the set 3 was calculated from resuspended filaments collected in the lake and incubated at 12°C (Micheletti et al. 1998).

The photosynthetic rates P_{zt} (in μmol m^{-3} h^{-1}) for a given depth z at a given time t were computed from the potential rate of photosynthesis that is predicted by the photosynthetic coefficients and the in situ irradiance measurements. Irradiance I_{tz} at a given depth z and time t is given as follows:

$$I_{tz} = I_{0t}\left[\frac{I_z}{I_0}\right] \quad (5)$$

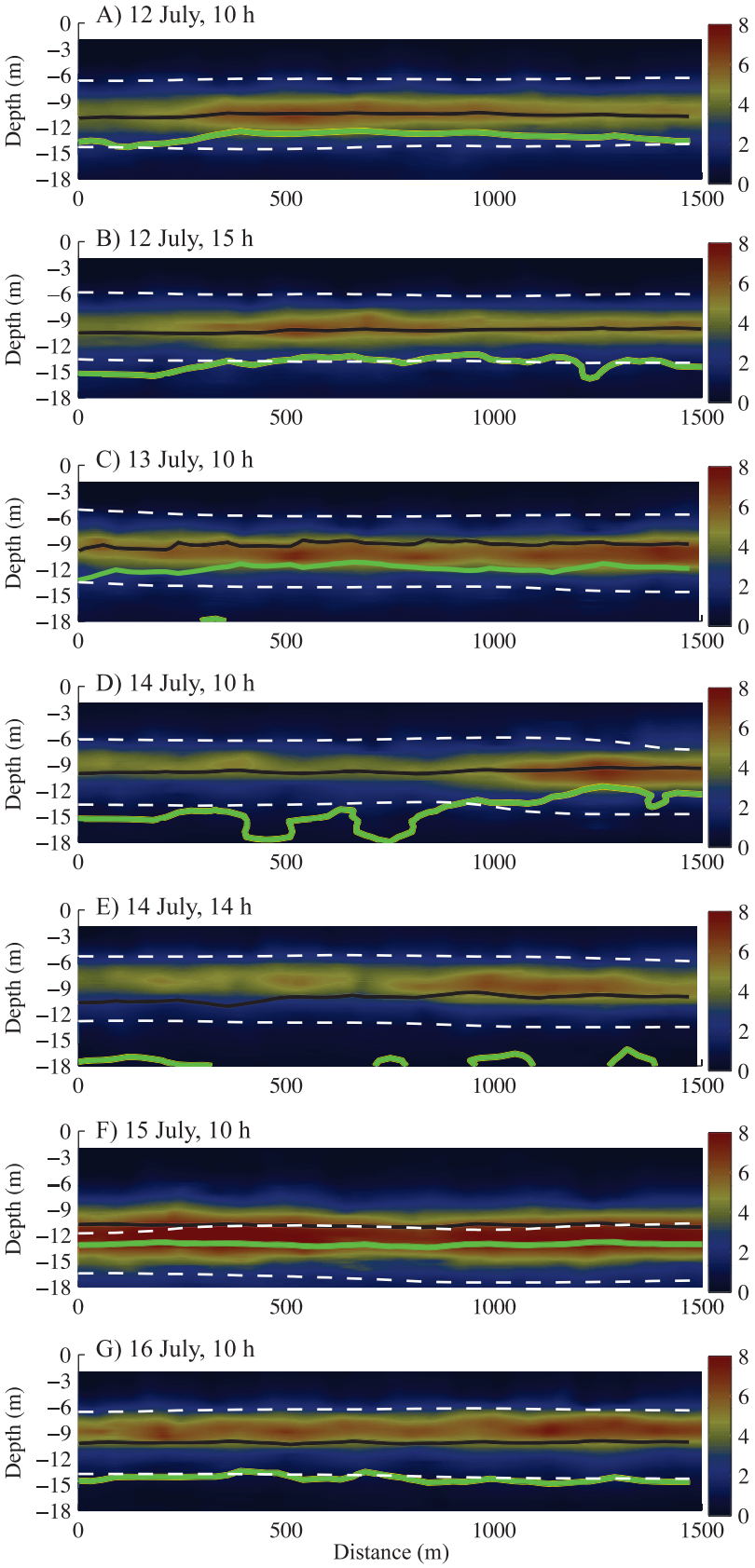
where I_{0t} is the irradiance below the surface at time t. The parameter I_{tz} replaced I in Eq. 4, which is re-arranged as follows:

$$P_{zt} = B_z\left[P_m\left(1 - \exp\left(\frac{-\alpha I_{tz}}{P_m}\right)\right) + R + (\beta I_{tz})\right] T_{10}^{\left(\frac{(T_z - T')}{10}\right)} \quad (6)$$

where B_z is *P. rubescens* biovolume at depth z (in cm^3 m^{-3}). Equation 6 also incorporates a correction term for temperature, where $T_{10} = 2$ and is the relative increase in P over 10°C; T' is the incubation temperature for the P/I curves (given in Table 1), and T_z is the temperature measured at depth z.

We used the interpolated data to compute P_{zt}. The interpolation gave a value for a volume that has a length of 30 m, a width of 30 m, and a height of 0.6 m. In other words, the 1.5 km (or 1500 m) transect is composed of 50 interpolated volumes on the horizontal axis (distance) and 27 interpolated volumes on the vertical axis (depths from 2 m to 18 m). This sequence of interpolated areas can be seen as a series of consecutive columns aligned from the west shore to the east shore of the lake, over a distance range of 0 m to 1500 m. Thus, for a given column, we calculated a P_{zt} value for each depth z at each time point t. Then, double integrals $\sum\sum(P)$ over a day (24 h) and throughout depths (from 2 m to 18 m) were calculated with the trapezoidal rule. Daily integral photosynthetic rates are expressed in mmol O_2 m^{-2} d^{-1} and are estimators of the photosynthesis by *P. rubescens* during a day in a given surface area of Lake Zurich, from the surface down to the upper part of the hypolimnion. The transects also have a horizontal dimension (i.e., the distance); therefore, a mean value was calculated for each transect.

Statistical analysis—Statistical analyses on sensor data were performed using MATLAB. The data were not following a normal distribution; therefore, we used non-parametric tests. Medians and interquartile ranges ($Q1$ and



	West shore	Center	East shore
Prub	3.2	**3.9****	3.2
Temp	12.8	13.1	12.7
I	**7.1***	**4.9***	6.4
DO%	102	108	105
n (n_{DO})	328 (199)	403 (213)	322 (111)
Prub	3.3	**3.7****	3.1
Temp	11.8	12.7	13.3
I	**6.9***	6.5	**6.4***
DO%	99	103	105
n (n_{DO})	379 (214)	377 (198)	318 (109)
Prub	**3.3***	3.6	**3.8***
Temp	13.7	**13.2***	**13.6***
I	2.4	**1.9***	**3.4***
DO%	104	105	107
n (n_{DO})	381 (209)	378 (197)	326 (121)
Prub	2.6	2.9	**3.9****
Temp	12.7	12.0	13.7
I	2.9	2.8	**2.2****
DO%	**90***	100	**105***
n (n_{DO})	304 (185)	367 (196)	296 (107)
Prub	**2.8****	3.1	3.4
Temp	12.2	**11.6***	**13.0***
I	**6.3***	**3.4***	5.0
DO%	89	95	**100****
n (n_{DO})	305 (185)	375 (192)	312 (116)
Prub	**5.4***	**4.7***	5.4
Temp	12.6	12.4	12.9
I	0.5	**0.5***	**0.5***
DO%	100	95	95
n (n_{DO})	275 (161)	312 (155)	264 (107)
Prub	**2.9****	**3.3****	**3.8****
Temp	**10.6***	12.6	**13.3***
I	4.8	6.5	7.0
DO%	**81****	93	98
n (n_{DO})	340 (213)	367 (201)	298 (98)

Table 1. Photosynthetic coefficient sets for *Planktothrix rubescens* of Lake Zurich. Set 1 was used to compute the default value of daily integral photosynthetic rates; the two other sets are for comparison. Coefficients were measured from experimental determination of *P/I* curves (i.e., the relationship between photosynthesis *P* and irradiance *I*) of *P. rubescens* that was incubated in situ (set 1) or in controlled conditions using cultures (set 2) or resuspended filaments in lake water (set 3). *See* Methods section for details on each photosynthetic coefficient.

Coefficient and unit	Set 1*	Set 2†	Set 3‡
α (μmol cm^{-3} h^{-1} (μmol m^{-2} s^{-1}) $^{-1}$)	26.7	14.9	88.2
β (μmol cm^{-3} h^{-1} (μmol m^{-2} s^{-1}) $^{-1}$)	0.522	−0.0797	−0.269
P_m (μmol cm^{-3} h^{-1})	676	486	1472.4
R (μmol cm^{-3} h^{-1})	−74	−47.3	−100.3
T' (°C)	15.5	20.0	12.9
T_{10}	2	2	2

* Walsby et al. 2001.
† Walsby 2001.
‡ Micheletti et al. 1998.

*Q*3) were used to summarize each sampling run as well as each lake layer. The relationship between two variables was evaluated based on Spearman's rank correlation coefficient (r_S) at the level of significance $p < 0.05$. Kruskal–Wallis analyses on rank-ordered data (KW) were used to compare data between different groups. The null hypothesis that mean ranks of samples were the same for all the groups was rejected when $p < 0.05$. First, the three layers of the lake were compared: (1) epilimnion, (2) metalimnion, and (3) hypolimnion. Data were grouped by layer according to the metalimnion depths determined using Eq. 1. Values from 2 m to the upper edge of the metalimnion were allocated to the epilimnion; whereas, values from the lower edge of the metalimnion down to 18 m were assigned to the hypolimnion. Analyses were performed for each sampling run separately and for all the data together. Second, KW tests were also performed for each sampling run to determine whether the data recorded in the metalimnion in three different areas of the lake come from the same population. The three areas were (1) west shore or windward shore (0–500 m), (2) central lake area (500–1000 m), and (3) east shore or leeward shore (1000–1500 m). Finally, KW tests were used to examine temporal variations by comparing the seven sampling runs (Figs. 4A–G, 5A–G). Dunn's post-hoc tests were used to determine which layers, areas, or runs differed from each other. It is worth noting that unlike the one-way analysis of variance, the KW analysis does not test the null hypothesis that the populations have equal means or medians. This implies that two groups having identical medians or means could be significantly different if their mean ranks are different (McDonald 2009).

Results

Meteorological and physico-chemical parameters—The weather during the week of sampling was fluctuating, and when clouds appeared on 13 July, the air temperature and the irradiance decreased accordingly (Fig. 3A,B). On 14 July, steady winds of moderate intensity accompanied by gusts of up to 7 m s^{-1} blew from the northwest for more than 16 continuous hours (Fig. 3C,D). On 15 July, the winds subsided, the temperature gradually rose, and the next day was sunny (Fig. 3). Temperature profiles showed that Lake Zurich was thermally stratified throughout the sampling campaign (Fig. 4). Statistical analysis confirmed that the epi-, meta-, and hypolimnion layers were significantly different in terms of temperature, dissolved oxygen saturation level, and in situ irradiance, whether considering all the data or each sampling run separately (KW $p < 0.05$; Table 2). All data taken together indicated that between 12 and 16 July, the median temperature in the metalimnion was 12.8°C and the median oxygen concentration was 11.4 mg L^{-1}, equivalent to a median of 99% of saturation. The light intensity in the metalimnion amounted to 3.5 μmol m^{-2} s^{-1} (Table 2). Comparatively, the median light intensity was much higher (median 207 μmol m^{-2} s^{-1}) in the epilimnion, and much lower (median 0.60 μmol m^{-2} s^{-1}) in the upper part of the hypolimnion (Table 2).

Biovolumes of P. rubescens—Vertical profiles of pigments recorded with the fluoroprobe indicated that *P. rubescens* was the dominant primary producer in Lake Zurich during the sampling period (Fig. 6). *P. rubescens*

←

Fig. 5. Cross-sectional views of *P. rubescens* biovolume (in cm^3 m^{-3}) in Lake Zurich between 12 and 16 July 2011. The dashed white lines indicate the upper and lower boundaries of the metalimnion. The black line indicates the depth of the neutral buoyancy (z_N), while the green line indicates the depth of the compensation point for growth (z_C). The right panel presents corresponding median values in the metalimnion for *Planktothrix* biovolumes (Prub), temperature (Temp), underwater irradiance (*I*), and dissolved oxygen saturation (DO%) calculated from the sensor data in each of the three transect areas for a given sampling run. The areas of the metalimnion that were significantly different (Kruskal–Wallis analysis [KW], $p < 0.05$) for a given variable are in bold. A double asterisk indicates an area that differs from all other areas; whereas, a single asterisk indicates two areas that are significantly different (KW $p < 0.05$). *n* indicates the size of the data sets for Temp, *I,* and Prub, and n_{DO} is the size of the DO data set.

Table 2. Median values and interquartile ranges ($Q1$ and $Q3$, in italics) for temperature (Temp), underwater irradiance (I), *Planktothrix rubescens* biovolumes (Prub), and dissolved oxygen (DO) in the epilimnion, the metalimnion, and the upper hypolimnion from 12 to 16 July 2011. Values were calculated from sensor data that were recorded over the entire lake transect. All the layers were significantly different (Kruskal–Wallis analysis, $p < 0.05$) than the other layers for all variables, except for two variables. n indicates the size of the data sets for Temp, I, and Prub; and n_{DO} indicates the size of the DO data set.

Layer	n	Temp (°C)	I (μmol m^{-2} s^{-1})	Prub (cm^3 m^{-3})	n_{DO}	DO (mg L^{-1})	DO (%)
Epilimnion	3230	*20.7*–21.5–*21.9*	*90*–207–*410*	*0.23*–0.63–*1.4**	1530	*10.8*–11.2–*11.5**	*118*–121–*125*
Metalimnion	7018	*9.55*–12.8–*16.5*	*1.2*–3.5–*18*	*2.2*–3.4–*4.9*	3487	*10.1*–11.4–*11.9**	*81*–99–*112*
Upper hypolimnion	3656	*6.11*–6.40–*6.84*	*0.30*–0.60–*0.80*	*0.55*–0.79–*1.1**	1834	*7.6*–7.9–*8.3*	*61*–63–*66*

* Indicates the two layers that *are not* significantly different from each other ($p > 0.05$) for Prub and for DO.

accounted on average for 57% of the total chlorophyll *a* concentration, whereas the second most abundant group, the diatoms, accounted for 38%. No cryptophytes, which also contain phycoerythrin pigments, were detected (Fig. 6). *P. rubescens* filaments were well-stratified in the metalimnion, where highest biovolumes were found (Table 2). The median biovolumes in the epi- and hypolimnion were similarly low, and were significantly different from the biovolumes of the metalimnion (KW $p < 0.05$; Table 2). When considering the *P. rubescens* biovolume data of all runs and all layers together, the only strong correlation (i.e., $r_S > 0.60$) found was with chlorophyll fluorescence ($r_S = 0.75$, $p < 0.05$), concordant with the observation that this organism was the dominant autotroph in the lake at the time of sampling. There was also an association between oxygen concentration and *P. rubescens* ($r_S = 0.48$, $p < 0.05$), showing the importance of *P. rubescens* as primary producer. This is in keeping with the recent work of Salcher et al. (2011), who demonstrated that *P. rubescens* was the dominant primary producer at the scale of the lake. The association with oxygen increased when considering the total chlorophyll fluorescence (i.e., all the primary producers in the top 18 m [$r_S = 0.84$, $p < 0.05$]).

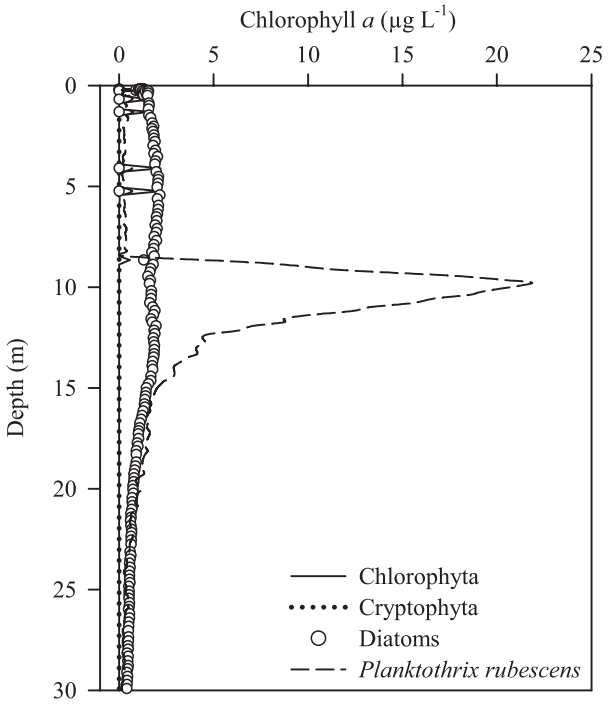


Fig. 6. Depth profiles of four phytoplankton groups in the upper water column of Lake Zurich on 19 July 2011. The concentration of each algal group is expressed in μg chlorophyll *a* L^{-1}. No cryptophytes were detected.

When the three layers were analyzed separately, more correlations were found. The epilimnion data subset indicated that *P. rubescens* was more abundant as we go deeper in the epilimnion (*P. rubescens* vs. depth, $r_S = 0.76$) where underwater irradiance (I), temperature (temp), and dissolved oxygen saturation level (DOsat) were lower (*P. rubescens* vs. *I*, $r_S = -0.64$; vs. temp, $r_S = -0.77$; vs. DOsat, $r_S = -0.59$; $p < 0.05$). The metalimnion subset showed a stronger association between *P. rubescens* and the dissolved oxygen concentration ($r_S = 0.53$, $p < 0.05$) than when all data were considered.

Spatial and temporal variations in the metalimnion—Most filaments were confined into a distinctive *Planktothrix* layer (Fig. 5) that moved according to the oscillations of the metalimnion (Fig. 4). The metalimnion width and location changed within a day but, on average, the upper boundary was located at 6.8 ± 2.0 m while the lower boundary was at 14.4 ± 1.2 m. Cross-sectional views also revealed horizontal heterogeneity in isotherm depths, metalimnion widths (Fig. 4), and *P. rubescens* biovolumes (Fig. 5). The metalimnion tended to be slightly larger on the east shore than on the west shore, and this was particularly apparent on 15 July (Fig. 4F). Two transversal views from the first sampling day depicted a particularly dense patch of *P. rubescens* in the central portion of the lake (Fig. 5A,B). Biovolumes and underwater irradiance in the metalimnion were significantly different there than on both shores (KW $p < 0.05$; Fig. 5A,B). On the next day, 13 July, *P. rubescens* was more abundant on the east than on the west shore, as suggested by KW test showing a significant ($p < 0.05$) difference between these two areas; whereas underwater light and temperature in the central part were different from the west shore (KW $p > 0.05$; Fig. 5C).

The windy conditions that lasted 16 h during the night of 13 to 14 July (Fig. 3D) seemed to have induced the thermocline depression that was visible on the morning of 14 July (Fig. 5D). The core of the *P. rubescens* population was then located on the shore opposite to the prevailing winds (i.e., the east shore) where metalimnetic biovolumes were significantly different compared with elsewhere in the lake (KW $p < 0.05$; Fig. 5D). The over-saturating oxygen concentrations and the low underwater irradiance in the metalimnion also showed differences between the three

Table 3. Daily irradiance (Q_0), irradiance measured just below the water surface (I_0), and range of underwater irradiance (I_{meta}) in the metalimnion (median, $Q1$ and $Q3$ [in italics]). Double integrals of daily photosynthesis were computed using the parameter set 1 (DP-1), set 2 (DP-2), and set 3 (DP-3) and averaged over the transect. Also presented are the medians of sensor data measured from 2 m to 18 m for *Planktothrix* biovolumes (Prub) and dissolved oxygen saturation level (DO). Letters A to G refer to the seven sampling runs, as used in Figs. 4 and 5. Sampling runs that were significantly different (Kruskal–Wallis, $p < 0.05$) from all other sampling runs for a given variable are in bold and indicated by an asterisk (*).

Date, time	Q_0 (mol m^{-2})	I_0 (μmol m^{-2} s^{-1})	I_{meta} (μmol m^{-2} s^{-1})	Prub (cm^3 m^{-3})	DO (%)	DP-1 (mmol O_2 m^{-2} d^{-1})	DP-2 (mmol O_2 m^{-2} d^{-1})	DP-3 (mmol O_2 m^{-2} d^{-1})
A) 12 Jul, 10 h	50.0	490	*1.3*–5.9–*37*	1.8	99	23.2	8.38	141
B) 12 Jul, 15 h	50.0	440	*2.4*–6.6–*34*	1.8	95	34.0	13.1	170
C) 13 Jul, 10 h	11.5	180	*0.70*–2.4–*12*	1.6	**96***	−20.8	−11.6	39.5
D) 14 Jul, 10 h	32.2	240	*1.0*–2.6–*13*	1.4	90	12.2	3.56	105
E) 14 Jul, 14 h	32.2	340	*1.8*–4.7–*23*	1.3	**78***	29.9	11.5	156
F) 15 Jul, 10 h	41.6	490	***0.30*–0.50–*1.8****	**2.8***	**112***	−10.3	−8.29	119
G) 16 Jul, 10 h	58.1	490	*2.3*–5.8–*37*	1.3	81	76.2	32.0	284

areas of the lake (KW $p < 0.05$; Fig. 5D). In the afternoon, the biovolume on the west shore was significantly different compared with the two other areas, and temperature was higher on the east than in the center (KW $p < 0.05$; Fig. 5E). Also observed in that afternoon was the shallowest depth of the metalimnion layer (Fig. 4E). The concentrations of *P. rubescens* in the metalimnion on 14 July were slightly lower (Fig. 5D,E) when compared with all sampling runs (Table 3).

A striking change was observed on 15 July, when the metalimnion was at its deepest location (Fig. 4F) and was thinnest (average of 5.9 m). It harbored a dense and elongated *Planktothrix* layer aligned on the 12 m depth isoline that was significantly different in the central area than on the west shore (KW $p < 0.05$), but not different from the east shore (Fig. 5F). The quartiles for *P. rubescens* biomass were as follows: west shore $Q1 = 3.4$, $Q3 = 7.5$; center $Q1 = 2.6$, $Q3 = 7.2$; and east shore $Q1 = 3.1$, $Q3 = 7.5$. Similarly, even though the medians for irradiance were equal, the center differed from the east shore (KW $p < 0.05$; Fig. 5F). Quartiles were as follows: west shore $Q1 = 0.30$, $Q3 = 1.5$; center $Q1 = 0.30$, $Q3 = 2.5$; east shore $Q1 = 0.20$, $Q3 = 1.4$. *P. rubescens* concentrations were maximal on 15 July; whereas, underwater light was the lowest, and accordingly, both parameters significantly differed from all other sampling runs (KW $p < 0.05$; Table 3). On 16 July, the core of the *Planktothrix* layer was shifted 3 m upward (Fig. 5G) and its biovolume was reduced when compared with the previous day (Table 3). Temperature, dissolved oxygen saturation level, and *P. rubescens* biovolume were low and all different on the west shore (KW $p < 0.05$; Fig. 5G).

Neutral buoyancy and compensation depths, and daily photosynthesis—During the sampling period, the shallowest mean depth for neutral buoyancy was 9.0 m on 13 July, and the deepest was 10.9 m on 15 July (Fig. 5). The bulk of *P. rubescens* biomass was found aligned with the neutral buoyancy depth on three occasions: twice on the sunny, warm, and calm day of 12 July, and once more on the relatively windy and cloudy morning of 14 July (Fig. 5A,B,D). On two instances, most of *P. rubescens* filaments were located above the neutral buoyancy depth (Fig. 5E,G). The compensation point was usually positioned close to the lower border of the metalimnion. Its mean value fluctuated between 12.2 m and below 18.0 m (Fig. 5). Most of the *P. rubescens* biomass was found above the compensation depth, except on 15 July, where that point was located at a mean depth of 13.1 m (Fig. 5F). The deepest compensation depth (> 18 m) was found on 14 July in the afternoon (Fig. 5E).

The daily irradiance Q_0 varied significantly during the sampling period, with values oscillating between 11.5 mol m^{-2} on 13 July to a maximum of 58.1 mol m^{-2} on the 16 July (Table 3). Daily photosynthesis (DP) calculated for each sampling run indicated that the amount of oxygen produced by *P. rubescens* varied from 12.2 mmol O_2 m^{-2} d^{-1} to 76.2 mmol O_2 m^{-2} d^{-1} (DP-1 in Table 3). Negative daily photosynthesis rates of −20.8 mmol O_2 m^{-2} d^{-1} and −10.3 mmol O_2 m^{-2} d^{-1} were computed on 13 and 15 July, respectively, indicating net oxygen consumption by *P. rubescens*. On both days, most of the *P. rubescens* biomass was located below the neutral buoyancy depth (Fig. 5C,F), even though these 2 d were different in terms of light field in the metalimnion. Solar radiation and underwater irradiance were minimal on 13 July; whereas, these values were among the highest on 15 July (Fig. 3B; Table 3), but a large portion of the biomass was located much deeper (Fig. 5F). On 15 July, the percentages of oxygen saturation and the biovolumes of *P. rubescens* were also significantly different from all the other days (KW $p < 0.05$; Table 3), and a notable proportion of *P. rubescens* biomass was found below the compensation depth (Fig. 5F). Interestingly, on 14 and 16 July, two dates after which a negative photosynthesis value was obtained, we observed oxygen saturation percentages that were lower than 100% or, in other words, a relative oxygen deficit (Table 3). The deficit in dissolved oxygen on 14 July in the afternoon (median 78%) was significantly different from all other sampling runs (KW $p < 0.05$; Table 3).

Two daily photosynthesis values were calculated for 12 and 14 July. For both days, the values calculated using the afternoon data set were 1.5–2.5 times higher than when calculated using the morning data set (Table 3). The rates seemed to follow the available light in the metalimnion,

rather than the light below the water surface at the time of the sampling (Table 3). Nonetheless, the two daily photosynthetic rates computed for a given day were of the same order of magnitude and showed the same trend for a net production. The parameter set used to compute the daily photosynthetic rates had a large influence on the resulting values (Table 3). The parameters determined on a culture grown at 20°C (set 2) yielded rates that were 30–80% lower than when using parameters determined in situ at 15°C (set 1; Table 3). Nonetheless, both parameter sets gave similar trends in the computed photosynthesis (e.g., negative rates were computed for the same days). This was not the case when using the parameter set 3, which gave much larger estimations and no negative values (Table 3).

Discussion

High-resolution depiction of P. rubescens *distribution*—The sampling at high temporal resolution highlights the rapid remodeling of the *Planktothrix* distribution under wind-forcing, on both horizontal and vertical axes. The *Planktothrix* layer was moved downward along with the oscillating metalimnion by about 6 m in < 24 h, which is much larger than the mean filament sinking velocity of 0.4–0.52 m d^{-1} (Walsby et al. 2001; Walsby and Holland 2006). The seiche-induced vertical displacement reduced the light quantity in the metalimnion by $\sim 90\%$ with effects on daily photosynthesis calculations that returned negative values, denoting oxygen consumption by *P. rubescens*. These results showed that seiche movements can sometimes move a metalimnetic population close to, or even below, its compensation depth and, thus, have consequences on the balance between oxygen production and utilization. This is especially significant in an ecosystem such as Lake Zurich, where the phytoplankton biomass is dominated by the metalimnetic cyanobacteria *P. rubescens*.

The brief but intensive sampling campaign carried out in Lake Zurich gave fine temporal observations of the *Planktothrix* layer movements from a cross-section perspective. The existence of seiche movements was inferred from the isotherm displacements at our fixed transect location, but our sampling was not designed to show the complexity of physical processes at the basin scale. Such detailed work would have required time-series data from current meters and thermistor chains fixed on several moorings (Horn et al. 1986). Our study rather intended to illustrate the effects of known physical phenomena on the ecology of *P. rubescens* in Lake Zurich. The results are among the first to depict a short-term (3 d) displacement of the *Planktothrix* layer caused by a storm event.

Robotics combined with sensing technologies had made possible the observation of seiche motions by the acquisition of high-resolution data in both space and time. These new and promising approaches for collecting continuous measurements will surely contribute to our understanding of ecosystems by generating large amounts of data at relevant but yet unexplored scales. The organization, integration, and analysis of such huge data sets are challenging, and thus new software tools intelligible to ecologists need to be developed for a better management and visualization of these high-frequency data. Technological shortcomings may also arise with the response time of the sensors when profiling at fast speed, which were particularly critical with the dissolved oxygen sensor. Oxygen optodes may be an interesting solution because they are highly precise and accurate optical sensors, and are more stable than electrochemical sensors (Tengberg et al. 2003), but so far, the response time is still too slow for fast profiling speeds. The development of better, faster sensors is necessary for rapid profiling applications, especially because a greater utilization of sensor technologies for field biology studies is foreseen (Porter et al. 2009).

Patchiness in P. rubescens *spatial distribution*—The transverse sections of Lake Zurich confirmed that *P. rubescens* stretches from the western to the eastern side of the basin, displaying a patchy pattern similar to the one observed on a longitudinal north–south transect (Salcher et al. 2011). Our results are in line with other observations of the patchiness in *P. rubescens* horizontal distribution in pre- and alpine lakes (Cuypers et al. 2010; Sotton et al. 2011). Large patches (~ 500 m long) of dense *Planktothrix* biomass recurrently formed in the metalimnion during our sampling, with apparent effects on the local physico-chemistry. In most areas where thick aggregations were present, less light was measured, dissolved oxygen was found in over saturating concentrations, and higher temperatures were recorded (Fig. 5). Local increases in dissolved oxygen are probably driven by high photosynthetic rates of an abundant cyanobacterial biomass. Compact buildup of cyanobacterial cells may also have contributed to slightly warm the area due to the light absorption by their photosynthetic pigments (Agustí and Philips 1992). In the same vein, the accumulation of *Planktothrix* filaments over a growth season was recognized as the major modulator of light attenuation in Lake Zurich (Walsby et al. 2001). Our results showed that this could also be applied to shorter time scales, because condensed biomass induced a localized steepening of the light gradient leading to self-shading. Self-shading may have an effect on toxin synthesis (Kardinaal et al. 2007). Results of the latter authors' competition experiments with *Microcystis* cultures indicated that non-microcystin-producing strains compete better for light than microcystin-producing ones. Similarly, dense, shaded patches of biomass may favor non-microcystin-producing *P. rubescens* filaments, which, even though they are more vulnerable to grazers (Kurmayer and Jüttner 1999), can hide from grazers in the toxic accretion.

At a larger scale, underwater light, and thus photosynthetic production, can be modified by cloud cover, which has an effect on the buoyancy response of the *Planktothrix* filaments. Theoretically, the neutral buoyancy depth is located where the light averaged over a day can support the photosynthetic production of dense components (i.e., carbohydrates) in moderate amounts that would still permit 50% of filaments to naturally float (Walsby et al. 2004). Fluctuations in cloud cover were shown to affect the neutral buoyancy depth on a daily basis (Walsby et al. 2004), and we also observed this within a single day. When

the sun was visible in the afternoon of 14 July, more light was measured in the water column than in the morning (Table 3), and thus the depth of z_N was pushed downward by $\sim$ 1 m on the west shore (Fig. 5D,E). Interestingly, the z_N depth stayed roughly at the same place on the east shore because of light attenuation by the denser *Planktothrix* patch. The median depth of this *Planktothrix* patch had slightly moved upward by $\sim$ 1 m during that 4 h interval. This motion indicated a relative increase of gas vesicle volume compared with the amount of photosynthesized carbohydrates, which consumption can be inferred by a concomitant decrease in oxygen saturation percentages. Still, this displacement was greater than the estimated average speed of *Planktothrix* filaments ($\sim$ 0.5 m d^{-1}), meaning that they were entrained upward along with the metalimnion. Filaments can only maintain their depth if the water column is completely stable, and their buoyancy response is delayed relative to changes in irradiance (Walsby and Schanz 2002). Hence, the *Planktothrix* layer was aligned on the theoretical neutral buoyancy depth only on some occasions of calm weather. This limitation in buoyancy capacities was already described for motile phytoplankton species living in the epilimnion, such as the harmful dinoflagellate *Ceratium*. *Ceratium* cells cannot migrate to optimal light conditions during windy days, but can form a distinct layer during calm periods, even if they last only a couple of hours (Alexander and Imberger 2009). As with *P. rubescens,* the *Ceratium* population is located at a depth of favored light conditions only for a short time each day. Whether *P. rubescens* formed a dense layer that stretched from one shore to the other or rather was localized in a patch, the filaments were moved daily below or above the calculated neutral buoyancy depth during our sampling campaign, presumably by the action of seiches.

Vertical displacements of P. rubescens *and effects on daily photosynthesis*—The vertical heterogeneity in the *Planktothrix* distribution provided further evidences for the up-and-down movements generated by wind-induced seiches. This phenomenon was recently documented for Lake du Bourget, a large and deep alpine lake, where the depth of the *Planktothrix* maximum was displaced by up to 10 m at a given sampling location within a day (Cuypers et al. 2010). In Lake Zurich, seiche motions were reported to move the thermocline and the *Planktothrix* layer by up to 2 m over 36 h (Walsby et al. 2001). We noted larger vertical movements than in this previous study, but the 6 m downward shift of the metalimnion in < 24 h is similar to observations in Lake du Bourget.

The thermocline depression and biomass accumulation observed on the east shore of Lake Zurich on 14 July were likely caused by sustained winds that blew from the northwest for 16 consecutive hours the night before. These winds could also have contributed to a modulation of the dominant longitudinal seiche in the lake, which has a period of 44 h (Horn et al. 1986), to form a seiche with two vertical motions and one horizontal motion (V2H1 seiche). Just after wind relaxation, seiches usually display one vertical and one horizontal motion; whereas, V2H1 seiches are generated as the oscillation progresses in the lake (Boegman 2009). The second vertical mode of a V2H1 seiche creates oscillations in the epilimnion and the hypolimnion that are in opposite modes, which produce alternating movements of compression and stretching of the metalimnion (Münnich et al. 1992). On two occasions, our results depicted patches in the middle of the lake that had stretched into a large band that spread shore to shore within a day. No such remodeling of the *Planktothrix* layer was previously documented in Lake Zurich, even if seiche motions were detected (Walsby et al. 2001). In a small and elongated Swiss alpine lake (5 km long and 1 km wide), a V2H1 seiche was reported to provoke a simultaneous maximal thickness of the metalimnion at one end of the lake and minimal thickness at the other end (Münnich et al. 1992). Our transversal illustrations may be snapshots of the compression and extension of the metalimnion by a V2H1 seiche that travels longitudinally, entraining *P. rubescens* filaments from north to south in the lake basin. This may explain the sudden increase in biovolume noted on 15 July (Fig. 5F). Large changes in *P. rubescens* biovolumes along the north–south longitudinal axis of Lake du Bourget were also very likely generated by two-vertical-mode seiches (Cuypers et al. 2010). Furthermore, the prevailing longitudinal mode of 44 h ($\sim$ 1.8 d) in Lake Zurich fits with our time-series of isotherms depicted in Fig. 4, showing the metalimnion at nearly the same place on the mornings of 12, 14, and 16 July. The cross-sections of *P. rubescens* biomass are two-dimensional, and thus cannot reflect the variations across the entire lake basin. Even though our data set is limited to a fixed transversal line, the sampling scheme allowed for the observation of the temporal variations in *P. rubescens* distribution that were likely occurring at the scale of the lake basin.

Vertical movements due to seiches drastically affect the depth of the metalimnion and, thus, the irradiance that can reach the *Planktothrix* layer. As pointed out previously (Walsby 2001), little alterations in the layer depth lead to substantial changes in daily photosynthesis because the *Planktothrix* population was often located at its compensation depth during our sampling. Consequently, the balance between net positive and negative daily photosynthesis had fluctuated from one day to the other. The largest variation to occur between two consecutive days was an eight-fold increase in daily photosynthetic rates from -10 mmol O_2 m^{-2} d^{-1} to 76 mmol O_2 m^{-2} d^{-1}. Seiche displacement of the *Planktothrix* layer was previously shown to induce a two-fold change in light regime, which had contributed to increase the daily photosynthesis from 9 mmol O_2 m^{-2} d^{-1} to 53 mmol O_2 m^{-2} d^{-1} (Walsby et al. 2001). In the small Lake Bromont (0.45 km^2), changes in light experienced by the metalimnetic cyanobacterial population caused by a seiche induced comparatively smaller variations (up to $\pm$ 25%) in photosynthetic productivity (Pannard et al. 2011).

Irradiance within the metalimnion reached 37 μmol m^{-2} s^{-1} on the sunniest days, which falls in the light range of 20–200 μmol m^{-2} s^{-1} that would support maximal growth rates of *P. rubescens* (Walsby and Schanz 2002). Most median light values in the metalimnion (3–7 μmol m^{-2} s^{-1}) were above the lower limit of 2 μmol m^{-2} s^{-1} for net growth

by *P. rubescens* (Bright and Walsby 2000), but not on 13 and 15 July. Negative photosynthetic rates were obtained on these days during which part of the *P. rubescens* biomass was located close to, and even below, the compensation depth for growth. At such low irradiances, *P. rubescens* is capable of heterotrophic growth through the assimilation of amino acids (Zotina et al. 2003; Walsby and Jüttner 2006) and glucose (Salcher et al. 2013), which may have reduced oxygen stocks. Furthermore, on both days, the majority of the *Planktothrix* population was located below the neutral buoyancy depth. It is quite conceivable that the filaments would tend to float upward by increasing the gas-vesicle volume ratio through the respiration of dense carbohydrates (Walsby et al. 2004). This may also explain the relative oxygen deficit observed on each day following these episodes of heterotrophy. On 16 July, the upward movement of the *Planktothrix* layer well above the neutral buoyancy and compensation depths resulted in a return to net daily photosynthesis.

The primary production by *P. rubescens* in Lake Zurich from July to November 1995 was found to equal zero only on 4 d, when most of the population was located below 15 m (Micheletti et al. 1998). These daily photosynthesis calculations were based on the parameter set 3 that included a higher P_m value than reported in other works (*see* Table 1), mainly because it was determined in incubation chambers containing other photosynthetic organisms (Bright and Walsby 2000). Likewise, when using the parameters of Micheletti et al. (1998), our daily photosynthesis values were nine times higher and no negative values were obtained (Table 3). Hence, it is likely that the daily photosynthesis values calculated for the 1995 growth season are overestimated compared with subsequent studies. This implies that much more negative photosynthetic rates may have occurred during that 4 month period. This is further confirmed by Walsby (2001), who recalculated the daily photosynthetic rates using a more conservative P_m value and concluded that positive production occurred most of the time, except on six occasions, during the 4 months of their weekly sampling campaign. Our high-resolution sampling indicated drastic changes in daily photosynthetic rates in Lake Zurich, from positive primary production on a given day, to negative values on the next one.

The cross-sectional lake views obtained from our robotic platform illustrated the considerable effect of seiche-induced movements on photosynthetic rates by a metalimnetic cyanobacterial species. The daily photosynthetic rates varied by two-fold for a given day, depending on the collection time of the data that were used for the calculations. The lake system was also shown to alternate between an autotrophic and heterotrophic state at daily time scales, but the frequency of heterotrophic episodes as well as their occurrence at the basin scale are still unresolved. These observations stressed the influence of the sampling time, location, and area on photosynthetic rate calculations. Clearly, the dynamic and patchy nature of cyanobacterial distribution complicates the design of an efficient sampling strategy that would give a representative monitoring. A recent review on marine harmful-bloom surveillance recommended to gather several data types (point data, transects, synoptic) from both water samples and sensors, and to collect them using various systems (i.e., ships, fixed buoys, autonomous underwater vehicles, and satellites) to improve predictive models (Stumpf et al. 2010). Similarly, lake studies should include classical sampling methods as well as real-time, high-frequency data acquisition by fixed and moving autonomous robotic platforms. A step further would be the application of adaptive sampling algorithms for environmental monitoring, but this remains a rather unexplored aspect of robotics (Dunbabin and Marques 2012). The combination of traditional techniques and new technologies would eventually allow for a finer documentation of cyanobacteria abundances in time and space, which is essential to identify promoting factors, improve prediction models, and facilitate decision-making in water management.

Acknowledgments

We are grateful to all of our colleagues at the Limnological Station in Kilchberg, particularly E. Loher for his essential contribution to sampling efforts, and M. Salcher for thoughtful comments. We extend our thanks to P. Countway and two anonymous referees for their conscientious reviews and valuable comments. We also thank the Swiss National Science Foundation for supporting this work (project CR22I2-130023).

References

Agustí, S., and E. J. Philips. 1992. Light absorption by cyanobacteria: Implications of the colonial growth form. Limnol. Oceanogr. **37:** 434–441, doi:10.4319/lo.1992.37.2.0434

Alexander, R., and J. Imberger. 2009. Spatial distribution of motile phytoplankton in a stratified reservoir: The physical controls on patch formation. J. Plankton Res. **31:** 101–118, doi:10.1093/plankt/fbn101

Biggs, W. W. 2000. Principles of radiation measurements, Li-COR®. Lincoln (NE): Li-COR Biosciences [accessed 2013 April 29]. Available from: http://www.licor.com/env/pdf/light/Rad_Meas.pdf

Blom, J., J. Robinson, and F. Jüttner. 2001. High grazer toxicity of [D-Asp^3,(*E*)-Dhb^7]microcystin-RR of *Planktothrix rubescens* as compared to different microcystins. Toxicon **39:** 1923–1932, doi:10.1016/S0041-0101(01)00178-7

Blumthaler, M. 2012. Solar radiation of the High Alps, p. 11–20. *In* C. Lütz [ed.], Plants in alpine regions. Springer Vienna.

Boegman, L. 2009. Currents in stratified water bodies 2: Internal waves, p. 539–558. *In* G. E. Likens [ed.], Encyclopedia of inland waters. Academic Press.

Bossard, P., and others. 2001. Limnological description of the Lakes Zürich, Lucerne, and Cadagno. Aquat. Sci. **63:** 225–249, doi:10.1007/PL00001353

Brient, L., and others. 2008. A phycocyanin probe as a tool for monitoring cyanobacteria in freshwater bodies. J. Environ. Monit. **10:** 248–255, doi:10.1039/b714238b

Bright, D. I., and A. E. Walsby. 2000. The daily integral of growth by *Planktothrix rubescens* calculated from growth rate in culture and irradiance in Lake Zürich. New Phytol. **146:** 301–316, doi:10.1046/j.1469-8137.2000.00640.x

Caron, D. A., and others. 2008. Macro- to fine-scale spatial and temporal distributions and dynamics of phytoplankton and their environmental driving forces in a small montane lake in southern California, USA. Limnol. Oceanogr. **53:** 2333–2349, doi:10.4319/lo.2008.53.5_part_2.2333

Carraro, E., and others. 2012. Coupling high-resolution measurements to a three-dimensional lake model to assess the spatial and temporal dynamics of the cyanobacterium *Planktothrix rubescens* in a medium-sized lake. Hydrobiologia **698:** 77–95, doi:10.1007/s10750-012-1096-y

Codd, G. A., L. F. Morrison, and J. S. Metcalf. 2005. Cyanobacterial toxins: Risk management for health protection. Toxicol. Appl. Pharm. **203:** 264–272, doi:10.1016/j.taap.2004.02.016

Cuypers, Y., B. Vinçon-Leite, A. Groleau, B. Tassin, and J. Humbert. 2010. Impact of internal waves on the spatial distribution of *Planktothrix rubescens* (cyanobacteria) in an alpine lake. ISME J. **18:** 1–10.

Davis, P. A., and A. E. Walsby. 2002. Comparison of measured growth rates with those calculated from rates of photosynthesis in *Planktothrix* spp. isolated from Blelham Tarn, English Lake District. New Phytol. **156:** 225–239, doi:10.1046/j.1469-8137.2002.00495.x

Dunbabin, M., A. Grinham, and J. Udy. 2009. An autonomous surface vehicle for water quality monitoring. Australasian Conference on Robotics and Automation 2009, Sydney, Australia. [accessed 2012 February 18]. Available from: http://www.araa.asn.au/acra/acra2009/papers/pap155s1.pdf

———, and L. Marques. 2012. Robots for environmental monitoring: Significant advancements and applications. IEEE Robot. Autom. Mag. **19:** 24–39, doi:10.1109/MRA.2011.2181683

Ernst, B., S. J. Hoeger, E. O'Brien, and D. R. Dietrich. 2009. Abundance and toxicity of *Planktothrix rubescens* in the prealpine Lake Ammersee, Germany. Harmful Algae **8:** 329–342, doi:10.1016/j.hal.2008.07.006

Ferri, G., A. Manzi, F. Fornai, B. Mazzolai, C. Laschi, F. Ciuchi, and P. Dario. 2011. Design, fabrication and first sea trials of a small-sized autonomous catamaran for heavy metals monitoring in coastal waters. IEEE Int. Conf. Robot **2011:** 2406–2411.

Garneau, M.-È., A. Schnetzer, P. D. Countway, A. C. Jones, E. L. Seubert, and D. A. Caron. 2011. Examination of the seasonal dynamics of the toxic dinoflagellate *Alexandrium catenella* at Redondo Beach, California, by quantitative PCR. Appl. Environ. Microbiol. **77:** 7669–7680, doi:10.1128/AEM.06174-11

Hantke, R., and others. 1979. Der Zürichsee und seine Nachbarseen. [Lake Zurich and its neighboring lakes]. Fribourg. Office du Livre S.A.

Hitz, G., F. Pomerleau, M.-È. Garneau, C. Pradalier, T. Posch, J. Pernthaler, and R. Siegwart. 2012. Design and application of a surface vessel for autonomous inland water monitoring. IEEE Robot. Autom. Mag. **19:** 62–72, doi:10.1109/MRA.2011.2181771

Horn, W., C. H. Mortimer, and D. J. Schwab. 1986. Wind-induced internal seiches in Lake Zurich observed and modeled. Limnol. Oceanogr. **31:** 1232–1254, doi:10.4319/lo.1986.31.6.1232

Huisman, J., and F. Hulot. 2005. Population dynamics of harmful cyanobacteria: Factors affecting species composition, p. 143–176. *In* J. Huisman, H. C. P. Matthijs, and P. M. Visser [eds.], Harmful cyanobacteria. Springer.

Invitrogen 2008. Real-time PCR: From theory to practice. Invitrogen.

Kardinaal, W. E. A., L. Tonk, I. Janse, S. Hol, P. Slot, J. Huisman, and P. M. Visser. 2007. Competition for light between toxic and nontoxic strains of the harmful cyanobacterium *Microcystis*. Appl. Environ. Microbiol. **73:** 2939–2946, doi:10.1128/AEM.02892-06

Kurmayer, R., and F. Jüttner. 1999. Strategies for the coexistence of zooplankton with the toxic cyanobacterium *Planktothrix rubescens* in Lake Zurich. J. Plankton Res. **21:** 659–683, doi:10.1093/plankt/21.4.659

Leboulanger, C., U. Dorigo, S. Jacquet, B. Le Berre, G. Paolini, and J. F. Humbert. 2002. Application of a submersible spectrofluorometer for rapid monitoring of freshwater cyanobacterial blooms: A case study. Aquat. Microb. Ecol. **30:** 83–89, doi:10.3354/ame030083

McDonald, J. H. 2009. Kruskal–Wallis test and Mann–Whitney *U* test, p. 165–172, Handbook of biological statistics. Sparky House.

McQuaid, N., A. Zamyadi, M. Prevost, D. F. Bird, and S. Dorner. 2011. Use of in vivo phycocyanin fluorescence to monitor potential microcystin-producing cyanobacterial biovolume in a drinking water source. J. Environ. Monit. **13:** 455–463, doi:10.1039/c0em00163e

Micheletti, S., F. Schanz, and A. E. Walsby. 1998. The daily integral of photosynthesis by *Planktothrix rubescens* during summer stratification and autumnal mixing in Lake Zürich. New Phytol. **138:** 233–249, doi:10.1046/j.1469-8137.1998.00196.x

Moreno-Ostos, E., L. C. Pizarro, A. Basanta, and D. G. George. 2009. Spatial heterogeneity of cyanobacteria and diatoms in a thermally stratified canyon-shaped reservoir. Int. Rev. Hydrobiol. **94:** 245–257, doi:10.1002/iroh.200811123

Münnich, M., A. Wüest, and D. M. Imboden. 1992. Observation of the second vertical mode of the internal seiche in an alpine lake. Limnol. Oceanogr. **32:** 1705–1719, doi:10.4319/lo.1992.37.8.1705

Oberhaus, L., J. F. Briand, C. Leboulanger, S. Jacquet, and J. F. Humbert. 2007. Comparative effects of the quality and quantity of light and temperature on the growth of *Planktothrix agardhii* and *P. rubescens*. J. Phycol. **43:** 1191–1199, doi:10.1111/j.1529-8817.2007.00414.x

Ostermaier, V., and R. Kurmayer. 2009. Distribution and abundance of nontoxic mutants of cyanobacteria in lakes of the Alps. Microb. Ecol. **58:** 1–11, doi:10.1007/s00248-009-9484-1

Pannard, A., B. E. Beisne, D. F. Bird, J. Braun, D. Planas, and M. Bormans. 2011. Recurrent internal waves in a small lake: Potential ecological consequences for metalimnetic phytoplankton populations. Limnol. Oceanogr. **1:** 91–109.

Pobel, D., J. Robin, and J.-F. Humbert. 2011. Influence of sampling strategies on the monitoring of cyanobacteria in shallow lakes: Lessons from a case study in France. Water Res. **45:** 1005–1014, doi:10.1016/j.watres.2010.10.011

Porter, J. H., E. Nagy, T. K. Kratz, P. Hanson, S. L. Collins, and P. Arzberger. 2009. New eyes on the world: Advanced sensors for ecology. Bioscience **59:** 385–397, doi:10.1525/bio.2009.59.5.6

Posch, T., O. Köster, M. M. Salcher, and J. Pernthaler. 2012. Harmful filamentous cyanobacteria favoured by reduced water turnover with lake warming. Nat. Clim. Change **2:** 809–813, doi:10.1038/nclimate1581

Salcher, M. M., J. Pernthaler, N. Frater, and T. Posch. 2011. Vertical and longitudinal distribution patterns of different bacterioplankton populations in a canyon-shaped, deep prealpine lake. Limnol. Oceanogr. **56:** 2027–2039, doi:10.4319/lo.2011.56.6.2027

———, T. Posch, and J. Pernthaler. 2013. In situ substrate preferences of abundant bacterioplankton populations in a prealpine freshwater lake. ISME J. **7:** 896–907, doi:10.1038/ismej.2012.162

Skulberg, O. M., and R. Skulberg. 1985. Plantic species of *Oscillatoria* (Cyanophyceae) from Norway. Characterization and classification. Arch. Hydrobiol. Beih. **71:** 157–174.

Sotton, B., O. Anneville, S. Cadel-Six, I. Domaizon, S. Krys, and J. Guillard. 2011. Spatial match between *Planktothrix rubescens* and whitefish in a mesotrophic peri-alpine lake: Evidence of toxins accumulation. Harmful Algae **10:** 749–758, doi:10.1016/j.hal.2011.06.006

Stumpf, R. P., V. Fleming-Lehtinen, and E. Graneli. 2010. Integration of data for nowcasting of harmful algal blooms. Venice-Lido: OceanObs-09 Proceedings. [accessed 2012 November 28]. Available at: https://abstracts.congrex.com/scripts/jmevent/abstracts/FCXNL-09A02b-1860409-1-ppstumpf.pdf

Tengberg, A., and others. 2003. Optodes to measure oxygen in the aquatic environment. Sea Technol. **44:** 1–6.

Thomas, E. A. 1950. Auffallige biologische Folgen von Sprungschichtneigungen im Zürichsee. Schweiz. Z. Hydrol. **12:** 1–24. [Conspicuous biological effects of thermocline depressions in Lake Zurich.]

———, and E. Märki. 1949. Der heutige Zustand des Zürichsees. Mitt. Int. Ver. Theor. Angew. Limnol. **10:** 476–488. [The current status of Lake Zurich.]

Van den Wyngaert, S., M. M. Salcher, J. Pernthaler, M. Zeder, and T. Posch. 2011. Quantitative dominance of seasonally persistent filamentous cyanobacteria (*Planktothrix rubescens*) in the microbial assemblages of a temperate lake. Limnol. Oceanogr. **56:** 97–109, doi:10.4319/lo.2011.56.1.0097

Walsby, A. E. 1997. Numerical integration of phytoplankton through depth and time in a water column. New Phytol. **136:** 189–209, doi:10.1046/j.1469-8137.1997.00736.x

———. 2001. Determining the photosynthetic productivity of a stratified phytoplankton population. Aquat. Sci. **63:** 18–43.

———, Z. Dubinsky, J. C. Kromkamp, C. Lehmann, and F. Schanz. 2001. The effects of diel changes in photosynthetic coefficients and depth of *Planktothrix rubescens* on the daily integral of photosynthesis in Lake Zürich. Aquat. Sci. **63:** 326–349, doi:10.1007/PL00001358

———, and D. P. Holland. 2006. Sinking velocities of phytoplankton measured on a stable density gradient by laser scanning. J. R. Soc. Interface **3:** 429–439, doi:10.1098/rsif.2005.0106

———, and F. Jüttner. 2006. The uptake of amino acids by the cyanobacterium *Planktothrix rubescens* is stimulated by light at low irradiances. FEMS Microbiol. Ecol. **58:** 14–22, doi:10.1111/j.1574-6941.2006.00143.x

———, G. Ng, C. Dunn, and P. A. Davis. 2004. Comparison of the depth where *Planktothrix rubescens* stratifies and the depth where the daily insolation supports its neutral buoyancy. New Phytol. **162:** 133–145, doi:10.1111/j.1469-8137.2004.01020.x

———, and F. Schanz. 2002. Light-dependent growth rate determines changes in the population of *Planktothrix rubescens* over the annual cycle in Lake Zürich, Switzerland. New Phytol. **154:** 671–687, doi:10.1046/j.1469-8137.2002.00401.x

Yu, H., H. Tsuno, T. Hidaka, and C. Jiao. 2010. Chemical and thermal stratification in lakes. Limnology **11:** 251–257, doi:10.1007/s10201-010-0310-8

Zeder, M., S. Van den Wyngaert, O. Köster, K. M. Felder, and J. Pernthaler. 2010. Automated quantification and sizing of unbranched filamentous cyanobacteria by model-based object-oriented image analysis. Appl. Environ. Microbiol. **76:** 1615–1622, doi:10.1128/AEM.02232-09

Zotina, T., O. Köster, and F. Jüttner. 2003. Photoheterotrophy and light-dependent uptake of organic and organic nitrogenous compounds by *Planktothrix rubescens* under low irradiance. Freshw. Biol. **48:** 1859–1872, doi:10.1046/j.1365-2427.2003.01134.x

Associate editor: Dariusz Stramski

Received: 03 February 2013
Accepted: 04 July 2013
Amended: 08 July 2013

Appendix C

Stumm et al. [2012]

Elena Stumm, Andreas Breitenmoser, Francois Pomerleau, Cedric Pradalier, and Roland Siegwart. Tensor voting based navigation for robotic inspection of 3d surfaces using lidar point clouds. *The International Journal of Robotics Research*, 31(11), 2012

Tensor Voting Based Navigation for Robotic Inspection of 3D Surfaces Using Lidar Point Clouds

Elena S. Stumm, Andreas Breitenmoser, François Pomerleau, Cedric Pradalier and Roland Siegwart

The International Journal of Robotics Research published online 1 October 2012
DOI: 10.1177/0278364912461537

The International Journal of
Robotics Research
0(0) 1–24

DOI: 10.1177/0278364912461537
ijr.sagepub.com

Tensor-voting-based navigation for robotic inspection of 3D surfaces using lidar point clouds

E Stumm[1,2], A Breitenmoser[1], F Pomerleau[1], C Pradalier[1] and R Siegwart[1]

Abstract

This paper describes a solution to robot navigation on curved 3D surfaces. The navigation system is composed of three successive subparts: a perception and representation, a path planning, and a control subsystem. The environment structure is modeled from noisy lidar point clouds using a tool known as tensor voting. Tensor voting propagates structural information from points within a point cloud in order to estimate the saliency and orientation of surfaces or curves found in the environment. A specialized graph-based planner establishes connectivities between robot states iteratively, while considering robot kinematics as well as structural constraints inferred by tensor voting. The resulting sparse graph structure eliminates the need to generate an explicit surface mesh, yet allows for efficient planning of paths along the surface, while remaining feasible and safe for the robot to traverse. The control scheme eventually transforms the path from 3D space into 2D space by projecting movements into local surface planes, allowing for 2D trajectory tracking. All three subparts of our navigation system are evaluated on simulated as well as real data. The methods are further implemented on the MagneBike climbing robot, and validated in several physical experiments related to the scenario of industrial inspection for power plants.

Keywords

robotic inspection, 3D perception, environment modeling, path planning, autonomous climbing

1. Introduction

Inspection and maintenance of industrial facilities is becoming increasingly important, especially since many of the existing installations are reaching the ends of their life expectancy. In the case of coal and nuclear power plants, periodic inspections of the piping system are crucial for detecting defects in the structure early on, and thus prolonging the continuation of a power plant's safe operation. However, the inspection task is often a difficult one, which is to this day conducted by human inspectors and involves environments with limited accessibility. In addition, disassembly prior to inspection is commonly required, further adding to the complexity and risk of damage during the inspection procedure. Autonomous inspection robots offer an opportunity for in-situ inspections, improvements in inspection efficiency and safety, and a reduced overall outage time of the inspected power plant sections.

MagneBike is a compact mobile robot designed for inspecting the inside of complex, tube-like industrial structures, specifically steam chests of power plant facilities. Inspecting the inner surfaces of a steam chest for structural defects requires a mobile robot to carry a sensor, such as a camera or a wall-penetrating ultrasonic probe, to any desired point on the surface. This inspection task, combined with the complexity of the 3D environment, presents a number of challenges to a robot's locomotion, localization and navigation systems. The MagneBike robot has been designed to cope with some of these challenges. Two magnetic wheels are arranged in a bicycle configuration with a freely rotating fork and supporting wheels for stabilization and lifting, enabling MagneBike to climb on ferromagnetic surfaces with respect to any orientation, and traverse highly curved and even step-like surfaces (Tâche et al., 2009). MagneBike is equipped with a rotating lidar for taking 3D laser scans of its environment, which has been successfully demonstrated to enable global localization via iterative closest point (ICP) methods by Tâche et al. (2011). The basic configuration of the MagneBike robot can be seen

[1]Autonomous Systems Lab (ASL), ETH Zurich, Switzerland
[2]Robotics and Interactions (RIS), LAAS-CNRS Toulouse, France

Corresponding author:
Elena S Stumm, Robotics and Interactions (RIS), LAAS-CNRS, 7 av. du Colonel Roche, 31077 Toulouse, France.
Email: elena.stumm@laas.fr

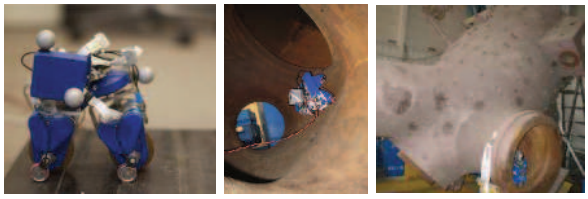


Fig. 1. The MagneBike robot and its expected environment. Left: Close-up of the MagneBike robot. Middle: MagneBike inside a steam chest at a junction facing a hole-like obstacle. Right: Outside view of the steam chest, with MagneBike climbing along the tube perimeter.

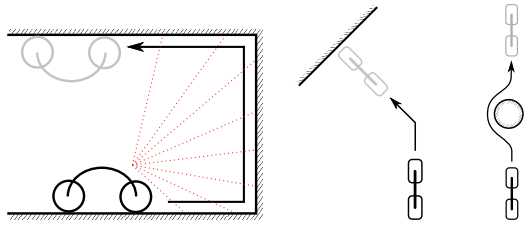


Fig. 2. Perception and navigation objectives. Left: MagneBike plans its motion constrained to a 2D manifold embedded in 3D space, based on laser range data; when confronted with obstacles, MagneBike must detect and decide to either attack the obstacle (middle), or avoid the obstacle (right).

in Figure 1 on the left, and Figure 1 in the middle and on the right shows the robot with the mounted rotating lidar deployed in a typical example of a steam chest environment.

Up until now, MagneBike has always been fully guided via remote control by a human operator. This work therefore focuses on navigation and the further automation of the MagneBike robot. In order to bridge the gap between complete manual control and autonomy, MagneBike needs to be capable of autonomous perception and navigation on a local scale. We assume that a human operator or a higher level planner on the robot will then provide the global guidelines for exploration trajectories or inspection missions. In general, a plan of the industrial structures, such as a map or 3D model is not available. As a result, the navigation requires the creation of a local model of the environment, obstacle and feature detection, as well as a method for safe traversal through the robot's surroundings. The generation of short paths that are feasible and avoid high-risk maneuvers must be enforced, as reliability is a high priority in inspection applications and recovery from failure situations may be extremely difficult or impossible in the narrow enclosures the MagneBike robot operates in. The solution of the presented inspection task is non-trivial due to the lack of knowledge of the environment and the unique complexity, which includes moving on surfaces in 3D space and passing of various types of obstacles.

This paper proposes a navigation system for the novel application domain of robots navigating on curved 3D surfaces, and makes the following contributions:

(1) A method that originates from computer vision, so-called tensor voting, is applied to robotics as a valuable tool to model arbitrarily complex environments from point clouds provided by laser range finders; the representation is point-based and allows for the application of subsequent processing steps (e.g. path planning) without the cost of building representations such as polygonal surface meshes.

(2) A path planning method for planning based on noisy point clouds is introduced; the planner is graph-based and establishes connectivities between points with regard to surface normals and robot constraints incrementally.

(3) The navigation system is evaluated, implemented on the MagneBike robot and tested in the challenging environment of 3D tube-like structures. To the best of the authors' knowledge, this is the first system demonstrating autonomous navigation constrained to complex 3D surfaces independently of gravity.

The remainder of this section provides additional information concerning the underlying assumptions about the environment, as well as the perceptive sensors of MagneBike. Section 2 describes related work in environment representation or mapping, and path planning. Section 3 introduces and further motivates the proposed navigation system. Next, Sections 4, 5 and 6 explain the theoretical basis behind our perception, path planning and control strategies. Sections 7 and 8 include evaluations based on simulated and real data, and physical experiments with MagneBike navigating a test environment. Finally, Section 9 concludes the paper and gives an outlook to future work.

1.1. Environmental characteristics

The environmental characteristics define many requirements on the navigation system. In this paper, the inspection scenario and the steam chest will serve as an instance of a 3D environment, which incorporates many characteristic features such as narrow corridors, tunnels, obstacles to avoid, and uneven terrain to negotiate, which are present in many of today's mobile robotic applications. Since the MagneBike robot is expected to operate in arbitrarily oriented, interconnected tube-like chambers of steam chests (as shown in Figure 1 in the middle and on the right, and in Figure 27), it must be able to drive along circumferential paths, through tight openings in tubes, over convex and concave step transitions (for instance, wherever there is a change in tube diameter), and over small gaps in the structure. A detailed discussion on the mobility requirements of the MagneBike robot can be found in the work by Tâche et al. (2009).

An important capability to note is that step-like obstacles can be traversed if approached in a perpendicular direction, due to MagneBike's ability to use stabilizing wheels as lifters. However, if not approached perpendicularly, proper

adhesion between the magnetic surfaces can be lost, causing MagneBike to detach from the surface. Through experiments documented by Tâche et al. (2009), the maximum angle of attack for convex steps was found to be 14°, while the maximum angle of attack for concave edges is 23°. Similar behaviors might also be observed in different contexts, such as the negotiation of curbs in an urban scenario or slopes by an outdoor all-terrain vehicle. Such mobility requirements dictate that the proposed navigation system must be able to identify various environmental features, and if presented with any obstacles, either avoid or overcome them safely. Figure 2 in the middle and on the right shows two such examples of possible scenarios, one where MagneBike attacks a step head-on and one where MagneBike must circumvent a hole.

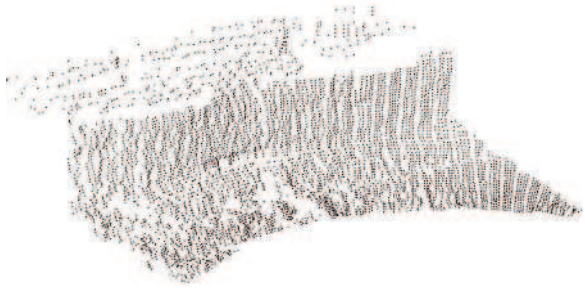


Fig. 3. Example of a lidar point cloud of a small metal step, which is heavily affected by noise and density variations. The point cloud contains 6,000 points, and was obtained from a Hokuyo URG-04LX laser range finder.

1.2. Sensing the environment

Figure 2, on the left, illustrates the overall perception and navigation principle for moving on surfaces in 3D space. Following the same reasoning as Tâche et al. (2011), a laser range finder has been selected for perception, which allows for localization and modeling of the environment. This choice is motivated by the inability to use other standard robotic sensors such as GPS and vision in such heavily confined, dark, and lightly textured metallic structures such as the steam chest.

The MagneBike is equipped with a 3D lidar sensor, i.e. a rotating Hokuyo laser range finder, which builds a 3D point cloud representation of the surrounding structures as it rotates around the robot's vertical axis. In this work, we make use of both the Hokuyo URG-04LX and the Hokuyo UTM-30LX laser range finders; a characterization of these sensors is presented in the work by Tâche et al. (2011), Kneip et al. (2009), Wong et al. (2011) and Pomerleau et al (2012).

Lidars, in addition to cameras, are one of the most widely used sensing devices in current mobile robot systems. The lidar point clouds can often be noisy, anisotropic, and have huge density variations. In particular, specular reflections occurring at the metallic surfaces add to the deformation of the obtained point clouds. A severe demonstration of these effects can be seen in Figure 3. Some additional alignment errors are expected if instead of one single point cloud, several compounded point clouds are used (see Figure 27 for an example of a point cloud composed of multiple scans). Furthermore, for the presented work on inspection we assume that, due to the nature of the environment, no dynamic changes occur and thus the true environmental features remain static throughout time. This assumption must be relaxed when applying the navigation system to more dynamic environments; extensions of the representation and planning subsystems along this direction are however outside the scope of this paper and subject to future research.

In combination with the lidar sensor, odometry estimations from the MagneBike robot can be used to perform six-degree-of-freedom (6-DoF) localization in the global reference frame. This process and experimental results have been presented in previous work on MagneBike by Tâche et al. (2011). In this paper, we primarily focus on the environment modeling, path planning and control, and therefore make the simplifying assumption that the approximate 6-DoF pose of the robot is readily available as input to the robot's navigation system. In the experiments of Section 8 we use a Vicon motion capture system to obtain the robot pose, replacing MagneBike's native localization capabilities.

2. Related work

Autonomous navigation in 3D environments using lidar data presents a challenging problem for environment perception and modeling, path planning, and control. This section explores related approaches used in previous work to address some of these problems.

2.1. Environment perception and modeling

The proposed navigation system enables robots to move on 3D surfaces. The MagneBike robot in particular, is capable of driving on the walls of complex and interconnected enclosures. Representations of such environments must be able to model the geometric structure of closed and cyclical surfaces. Methods that create 2.5D representations such as elevation maps, which store terrain heights in the cells of a 2D grid and are commonly used in robotics to model rough terrain environments for navigation (Hebert et al., 1989; Singh and Kelly, 1996; Thrun et al., 2004b), are therefore not applicable. Extensions to multi-level surface maps by Triebel et al. (2006) introduce several surface classes per cell and thus allow for modeling vertical structures and environments with multiple overlapping levels (e.g. bridges, underpasses). However, this solution is still intended for ground-based robots that encounter only a few of such overlapping levels; it is not tailored to robots which are expected to traverse vertical surfaces and ceilings, and thus move in full 3D space. Environments can

alternatively be modeled by 3D evidence or voxel grids. More recent approaches use tree-based representations such as octrees (see Wurm et al., 2010, and references therein), which offer increased compactness and flexibility regarding multi-resolution and dynamic expansion of maps of unknown environments. Aside from grid-based representations, polygonal surface meshes are able to represent the surfaces of complex structures more accurately. However, 3D surface reconstruction methods undergo a permanent trade-off between computation cost and accuracy. Accurate but computationally demanding approaches from computer graphics (see Kazhdan et al., 2006, and references therein), are confronted with simpler and faster but less accurate methods for robotic applications (Marton et al., 2009; Gingras et al., 2010). Another option is to work directly on the point clouds (Pauly et al., 2002; Smith et al., 2010). This is beneficial when dealing with noise and topological distortions, as it makes the merging of data sets easier and avoids the cost of one entire processing step by bypassing explicit surface reconstruction. Intelligent sampling and inference methods further simplify the point clouds, which helps to reduce memory requirements. More compact representations of 3D structures can for example be obtained by fitting basic geometric structures, such as planes and spheres, to the input data but require the environment to be well composed from these basic structures (Thrun et al., 2004a; Schnabel et al., 2007). Vandapel et al. (2004) used statistical methods to classify different types of structures in lidar data (e.g. clutter, lines, planes) for navigation in vegetated terrain. King (2008) presented a method that uses a process known as tensor voting to infer structure from laser range data. This method has the benefits of being able to segment and classify different types of structures, estimate the structures' orientations, as well as interpolate missing parts of these structures. We therefore return to tensor voting in Section 4, where we present our approach to environment representation, which makes use of tensor voting.

2.2. Path planning and control

Path planning algorithms are highly dependent on the types of applications, robot platforms, and environments they are meant for. In general, search-based, sampling-based, as well as combinatorial planning rely on a graph structure that connects through different states. Graph search algorithms such as Dijkstra's algorithm, A* or D* algorithms (Hart et al., 1968; Koenig and Likhachev, 2002) are commonly used for path generation. Both the Theta* and the Field D* algorithms and their extensions to three dimensions (Carsten et al., 2006; Ferguson and Stentz, 2006; Nash et al., 2010), generate paths that are no longer constrained to graph edges in 2D maps and 3D volumes. Kimmel and Sethian (1998) and Surazhsky et al. (2005) plan geodesic paths on surfaces embedded in 3D space based on fast marching and window propagation methods, by employing the surface mesh as graph structure.

In our approach, we also use a graph structure for connecting the robot states. Although feasible paths are constrained to the surfaces in 3D space, in contrast to above methods, we build on a point-based representation of the environment and establish graph connectivity in a local and incremental fashion. This allows us to expand states wherever required 'on the fly' by taking the constraints from the surface orientation and robot pose into account. The presented navigation system extracts feature maps from the point clouds and decomposes the environment around the robot into a 3D grid structure. The graph connectivity across neighboring grid cells is then established by locally examining a set of possible movement vectors, which combines elements from graph search and sampling-based planning. This approach has some parallels with motion primitives and lattice-based graphs, which incorporate mobility constraints directly into the search space (Pivtoraiko and Kelly, 2005).

2.3. Related applications and environments

The task of inspecting narrow tube-like structures with the MagneBike robot presents a rather unique application. Often MagneBike's level of integration and mobility has not been reached by other climbing robots. An overview of related climbing robots is given by Tâche et al. (2009). However, related applications and environments with similar demands on the navigation system can be found in rovers for rough terrain navigation, mining robots, and robots for different inspection tasks: even though these robots generally do not climb walls of enclosed structures, they move along surfaces embedded in 3D space.

For planning trajectories over rough terrain, Wettergreen et al. (2010) and Singh and Kelly (1996) generate kinematically feasible trajectories by forward simulation on mesh and elevation map models. For global planning, an A* search is applied. The approach by Howard and Kelly (2007) builds on a detailed vehicle model and an optimization-based trajectory generation framework, which accounts for various constraints, such as constraints on the rover's heading at specific positions for instrument placement. A cost functional is optimized in order to avoid obstacles or minimize the risk of traversing difficult terrain with high slopes or curvatures. Note that these properties resemble the intended behaviors of our navigation system for obstacle avoidance, obstacle negotiation and navigation on curved surfaces.

Thrun et al. (2004b) applied a wheeled robot equipped with a tiltable laser range finder to the exploration of underground mines. The robot was mostly operated by remote control, and performs simultaneous localization and mapping based on 2D maps. 3D range scans are used for navigation; the environment's traversability is represented in a 2.5D map.

Tavakoli et al. (2011) deployed a pole-climbing robot and multiple ground robots for automated inspection of

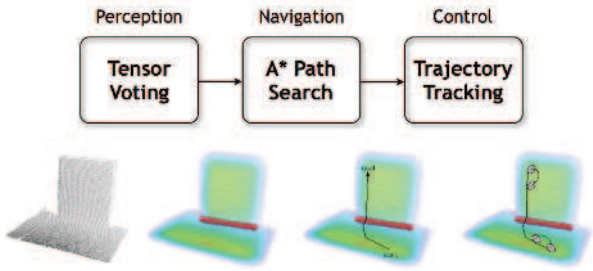


Fig. 4. Framework of the problem solving approach: perception and environment representation, path planning and control, illustrated for a floor to wall transition.

the outer surface of industrial piping systems. The focus is on the localization of the climbing robot and the mapping of the structure to be climbed through the team of ground robots. Owing to the uniformity and circular geometry of the pipe structures, however, mapping reduces to a 2D problem and the climbing robot's motion following the pipe is rather a linear one. More interestingly, the work of Englot and Hover (2011) deals with in-water ship hull inspection by an autonomous underwater vehicle. The ship hull is modeled as surface mesh from sonar range data, and inspection paths encompassing the reconstructed model are generated by sampling-based methods. While the robot achieves complete sensor coverage, it moves in proximity but not in direct contact with the ship hull surfaces. The robot's motion resides in open, full 3D space and, different from our inspection scenario, is not directly constrained by the surface geometry.

3. Navigation solution

In order to attain autonomous movement on a local scale, several fundamental components are required. First, a method is required for inferring the structure of the environment, i.e. to sense and create a model of the environment using sparse, noisy input data. Obstacles such as steps or holes must be identified and included in the description. Second, a method is needed to generate a feasible and safe trajectory towards the goal state. Finally, a control method must steer the robot along the provided trajectory, using feedback from a robot localization method. Figure 4 illustrates the framework of the complete navigation system.

The perception and representation task will be accomplished using tensor voting, which is able to estimate the orientation and dimensionality of the underlying structures given by the lidar point clouds. Most applications of tensor voting in three dimensions, including the work by King (2008), follow the process with a marching cubes algorithm in order to create a triangular surface mesh. We argue that such a full surface reconstruction is not necessary for obtaining a valid representation for path planning, as the raw tensor voting results, which are organized in a 3D grid structure, can be used almost directly as environmental maps.

This 3D grid structure lends itself well to graph-based planning algorithms. Therefore, the classic A* algorithm, which provides optimal low-cost paths for given heuristics, has been chosen for the task of trajectory generation in this work. Note that other graph-based planners, such as D* (Koenig and Likhachev, 2002), which further enable efficient replanning in situations with changing environment or imperfect sensor information, could be used instead. However, as explained in Section 1.2, this lies outside of the scope of this paper. We establish the graph connectivity and cost functions in such a way that the mobility requirements as discussed in Section 1.1 are satisfied, resulting in the generation of feasible 6-DoF paths in 3D space.

These paths are then transformed into 2D, planar trajectories by projecting movements into local surface planes, in a sense unwrapping the path from the 3D surface. Once this is accomplished, any appropriate trajectory tracking algorithm can be applied in two dimensions, using nonlinear feedback of the robot's pose.

4. Tensor voting for environmental representation

This section gives a brief introduction on tensor voting, followed by a description of how it is utilized in our navigation system. A more general overview of tensor voting principles can be found in the work of Medioni and Mordohai (2004) and Mordohai and Medioni (2005). In addition, King (2008) provides valuable insight into the application of tensor voting to 3D environment modeling.

4.1. Basics of tensor voting

Tensor voting is a generic solution for perceptual organization problems, and is based on a set of Gestalt principles, which are believed to be used by the human visual system (Medioni and Mordohai, 2004). Tensor voting is particularly useful because it can be applied to a wide range of applications and data types. Moreover, it can be performed in any number of dimensions to segment input tokens, such as points in a point cloud, into distinct structures. For example, in two dimensions, input can be grouped into curve-like structures (which have an associated direction) and regions or junctions (which have no associated direction). Similarly, in three dimensions, input can be grouped into surfaces (which have two associated directions), curves (with one preferred direction), and directionless points or regions (characteristic of noise or volumes). Note that curve elements can be found at the intersection of two surfaces, which will prove extremely useful for classifying step-like obstacles in the robot's environment.

Raw point clouds generated from lidar systems provide sparse, unoriented input tokens, whereas for our application, dense, oriented data is desired for navigation and control. Tensor voting is able to densify sparse input data. This densification is achieved through voting performed by each input token at every location in a predefined grid structure, essentially providing information at any desired resolution.

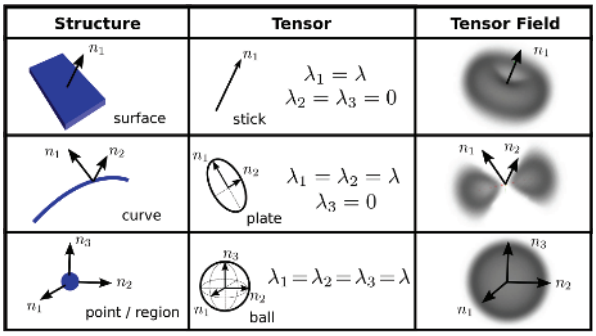


Fig. 5. Visualization of tensor voting in three dimensions. The fundamental types of structures are shown in the left-most column. The second column shows the form of the representative 3D tensors, with their eigenvalues λ_i. The last column shows the form of the voting fields cast by each fundamental structure. The vectors representing the structures' normal spaces are illustrated in each column as well. (Adapted from Medioni and Mordohai (2004).)

In addition, the orientation of any directional feature is estimated by tensor voting simultaneously.

As mentioned, in three dimensions, there are the three fundamental types of structures: surfaces, curves, and unoriented points. Information needed for voting and segmentation into these fundamental structures is encoded as 3D, second-order, symmetric tensors. This can be conceptualized as a 3×3 matrix, or alternatively as an ellipsoid where the eigenvectors of the tensors represent the axes of the ellipsoid, and the eigenvalues represent the size of the ellipsoid in each corresponding direction. There are therefore also three fundamental types of tensors, 'stick', 'plate' and 'ball' tensors, signifying the three aforementioned structures, as shown in Figure 5. These tensors are aligned with the normal space of the corresponding surface. Therefore, surface elements are represented by a 1D stick tensor, aligned with the surface normal. Curves or edges, on the other hand, are given by 2D plate tensors, where the tangent direction is given by the third eigenvector, which has a corresponding null eigenvalue. Tokens with no associated direction are encoded as unoriented, 3D symmetric ball tensors. In general, the magnitudes of the eigenvalues designate the associated saliency, and any arbitrary tensor can be decomposed into its ball, plate and stick components in order to evaluate the dominance of each type of structure at that location. In three dimensions, decomposition is achieved as follows:

$$T = \lambda_1 \hat{e}_1 \hat{e}_1^{\mathrm{T}} + \lambda_2 \hat{e}_2 \hat{e}_2^{\mathrm{T}} + \lambda_3 \hat{e}_3 \hat{e}_3^{\mathrm{T}} \quad (1a)$$

$$= (\lambda_1 - \lambda_2)\hat{e}_1 \hat{e}_1^{\mathrm{T}} + \quad (\text{stick component})$$
$$+ (\lambda_2 - \lambda_3)(\hat{e}_1 \hat{e}_1^{\mathrm{T}} + \hat{e}_2 \hat{e}_2^{\mathrm{T}}) \quad (\text{plate component})$$
$$+ \lambda_3(\hat{e}_1 \hat{e}_1^{\mathrm{T}} + \hat{e}_2 \hat{e}_2^{\mathrm{T}} + \hat{e}_3 \hat{e}_3^{\mathrm{T}}) \quad (\text{ball component}) \quad (1b)$$

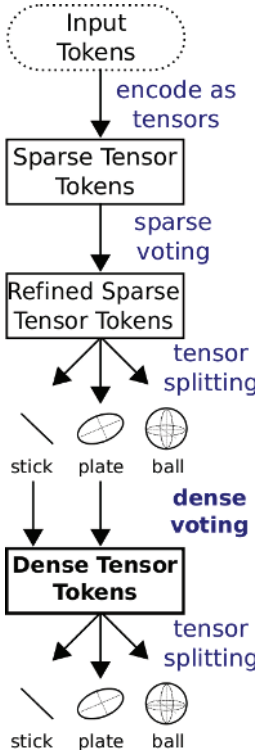


Fig. 6. Basic tensor voting framework for unoriented input tokens. (Adapted from Medioni et al. (2000).)

with λ_i representing the eigenvalues in order of decreasing magnitude, and $\hat{e}_i$ representing the corresponding eigenvectors (Medioni and Mordohai, 2004). Note that throughout this paper, the '$\hat{\cdot}$' symbol represents unit vectors.

Through voting, each component of the tensor (ball, plate, stick) is able to propagate its information according to specific voting fields. The idea is for votes to convey their apparent structure as if they were smoothly continued to the location where a vote takes place. The strength of the vote is based on the likely prevalence of the hypothetical structure at that location. Votes are cast by every input token (e.g. points in a point cloud), and collected using simple tensor addition. Ball tensors have no preferred orientation, and therefore propagate uniformly in all directions, with a ball-shaped tensor field. Plate tensors, which represent curve elements, have a roughly stick-shaped tensor field, as the structure is likely to continue only along the tangent directions. Stick tensors, on the other hand, vote with a plate-shaped tensor field in order to continue the surface-like structure. An illustration of the various tensor field forms can be seen in Figure 5. The scale of these voting fields, denoted by the voting scale parameter σ, can be chosen based on the approximate feature size. Further details regarding the specifics of the voting fields are discussed next in Section 4.2.

The general framework of the tensor voting algorithm, as proposed by Medioni et al. (2000), is shown in Figure 6.

To start, the sparse input tokens are transformed into tensors, which encode any known saliency or directionality. If no preliminary structural information is known, the input tokens are simply encoded as unit ball tensors. Once the sparse tensors are initialized, they are refined through a round of sparse voting. During the sparse vote, all input tokens cast votes only at the locations of all other input tokens. After sparse voting, the decomposed components of the sparse tensors are then used to densify the information through another round of voting, where votes are cast by all input tokens throughout the entire space (represented by a 3D grid). Note that ball components do not vote in this round, as our analysis is focused on surfaces and edges, rather than regions. Once the final votes have been collected and summed, the components of the tensors can be used to analyze the underlying structures.

4.2. Tensor voting applied to lidar point clouds

When originally described by Medioni et al. (2000), tensor voting was intended for computer vision applications, where everything is usually in two dimensions, dense, and uniformly distributed. The framework could therefore benefit from a few modifications when used in different types of applications such as ours. Here we discuss a few changes to the algorithm which were used in our implementation. Similar alterations are outlined by Mordohai and Medioni (2005) and King (2008), which were instrumental resources for this section. First, an analytic formulation of the voting fields is described, followed by a different proposal for the vote decay functions, and then a method of token reduction is presented.

4.2.1. Calculation of voting fields A vote cast by a tensor is meant to impart its contained structure, as if the structure was continued smoothly towards the location of the vote. For a stick vote, the assumption is made that the smoothest continuation of a surface is described by an arc of constant curvature which runs tangent to the surface at the voter's location, or in other words, an osculating sphere. Therefore a vote cast by a stick voter is also a stick, with the direction that is normal to the osculating sphere at that location (depicted in Figure 7). In the original implementations of tensor voting (as discussed by Medioni and Mordohai (2004)), the plate and ball fields were created by integrating the stick field as it is rotated about one or two of its axes, respectively. These integrals unfortunately have no closed-form solution, and so the integration is typically done numerically, by rotating the stick tensor by fixed steps and adding the results from each position. The resulting field is then normalized such that its energy is equivalent to that from the stick field. The results are stored in a look-up table at the desired resolution, and then during the voting procedure, values are obtained by interpolating between fields in the look-up table.

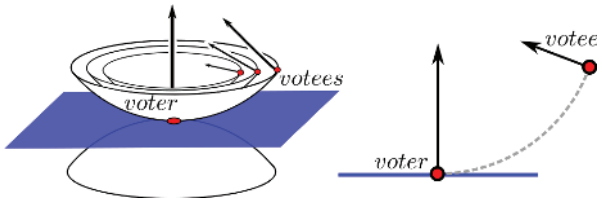


Fig. 7. Formulation of the stick vote in three dimensions. The osculating spheres represent where the plate-shaped tensor field spreads from the voter. The vectors at the votees (location where the votes are cast) show the directions of the votes, which are normal to the osculating spheres.

Calculation of votes using these pre-computed look-up tables can be computationally costly and inaccurate, especially when the number of dimensions is increased. Therefore a new analytic formulation for the voting fields is presented by Mordohai and Medioni (2005) and then further simplified by King (2008). Mordohai and Medioni (2005) outlined a comparison to the original implementation of tensor voting, and have shown improvements in the three-dimensional case. We therefore adopt this analytic approach in our implementation.

4.2.2. Vote decay function In general, the magnitude of a vote is modulated based on how likely the continuation of the hypothetical structure is. This can be evaluated based on two of the Gestalt principles, which are proximity and smooth continuation (Medioni and Mordohai, 2004). Therefore, the vote weighting could be represented as a function of distance for proximity and curvature for smooth continuation. Typically, saliency decays exponentially with distance squared. As a result, two tokens placed at almost the exact same location will produce extremely strong votes for each other, despite supplying very little new information regarding the local structure. This can easily lead to amplification of small-scale noise, as well as introducing large effects from density variations. In addition, the commonly used curvature decay function was designed for computer vision applications, and therefore exhibits strange behavior when distances are not defined in pixels. For example, the shape of the decay function changes drastically as the voting scale parameter σ is varied, and is completely invalid for σ values less than one.

In our framework, we used a new weighting function for votes derived by King (2008), which solves the aforementioned issues. The intuition used in the derivation of the new decay function is that tokens should have the strongest votes at a distance of σ, as this is the expected feature scale. Any votes much closer or much farther than σ will receive a lower weighting. In addition, the curvature decay function and the distance decay function are decoupled, eliminating unwanted side effects caused by the choice of σ. This new way of choosing the decay function has also been shown to provide smoother and more accurate results by King (2008).

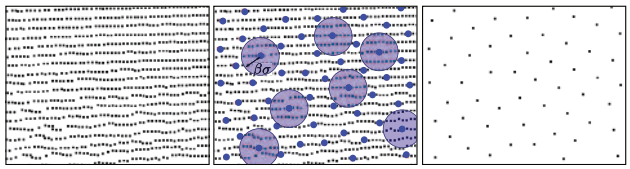


Fig. 8. Example of token reduction on real lidar data. The left frame shows the anisotropic point cloud, the middle frame shows the selected tokens and a few examples of the subsampling radius, while the last frame shows the reduced point cloud.

4.2.3. Token reduction Typically, lidar point clouds can contain a large number of points, with sizeable density variations. These density variations can skew the saliency results from the tensor voting in favor of densely sampled regions. Although there should be some preference for detecting structures at denser regions, the emphasis on these areas is usually too strong, preventing any structure inference at the less dense regions. The input tokens could therefore be subsampled in order to unify the density and reduce the computation time. However, it is important not to throw out a lot of pertinent information, compromising the feature extraction. In this section, one method of token reduction is proposed, and then later evaluated in Section 7.1.1.

Under the assumption that relevant features (and the strongest votes) occur on the scale of σ, subsampling should be related to σ as well. Once tokens become too sparse, and the inter-token distance goes above σ, the saliencies resulting from the tensor voting should reflect a low confidence in these areas. In contrast, in any dense regions where the inter-token distance drops below σ, the resulting saliencies should be considerably higher, but less dependent on the local density. The way this is achieved is by iteratively selecting tokens based on highest saliency after sparse voting, and then removing any nearby tokens within a given radius (taken as a fraction of the voting scale, $\beta \cdot \sigma$). As a result, in high-density regions, the remaining tokens are spaced such that the inter-token distance is roughly equal to this radius, whereas in low-density regions, the inter-token distance remains the same as before. This approach has been adapted and simplified from a multi-scale tensor voting method proposed by King (2008). An illustration of this process performed on a lidar point cloud can be seen in Figure 8.

4.3. Structure inference and modeling

After the tensor voting algorithm has been applied to the lidar point cloud, the acquired results are used for scene interpretation and path planning. The final tensors resulting from the second round of voting are densely distributed at every cell throughout a 3D grid of the chosen resolution. These tensors can be decomposed again using Equation (1b), which produces dense structural information for each underlying dimension. The final decomposed tensors are exploited and utilized for creating a set of environmental feature maps, which can be navigated by the path planning algorithm described in Section 5.

During tensor decomposition, each structure type has an associated saliency and directionality. For our application, the environment is expected to consist of only surfaces and edges, therefore limiting our analysis to the final stick and plate tensors. The surface map stores the surface saliencies along with the surface normals, which are given by the largest magnitude eigenvalue and eigenvector of the tensor.

Edge information, on the other hand, is first thresholded, and only classified as an edge if the plate saliency is high enough. Any of these cells will be treated as a form of obstacle, and so the saliency value is no longer important. A binary edge map is constructed, which distinguishes each grid cell to be either an edge or not. The edge tangent, given by the lowest magnitude eigenvalue and eigenvector, is relevant for edge traversals of the robot, and is thus mapped as well.

For safety reasons, edge obstacles are inflated by a radius, which can be chosen to suit the specifications of the robot. This is done by simply marking all of the neighboring grid cells of an edge in the binary edge map positive as well. The corresponding tangents are given by averaging the tangents from the nearby grid cells that contain the original edge. Figure 9 shows an example of these environmental feature maps, produced from a lidar point cloud of an L-shape structure.

5. Path planning on 3D surfaces

In addition to perception and environment representation, the unique environment in which MagneBike operates provides a number of challenges to path planning. Navigating on 3D surfaces eliminates the possible use of many standard path planning or trajectory generation algorithms, which assume surfaces to be modeled through 2D maps or 2.5D elevation maps. More complex surfaces in 3D space are traditionally represented by a surface mesh. Accurate mesh generation, however, is particularly complicated if point clouds are noisy or accumulated from matching several smaller point clouds. The use of tensor voting to build environmental feature maps allows us to forgo complex mesh construction while still being able to plan feasible and cost-optimized paths along the surface by using a specialized graph-based planning algorithm.

5.1. Problem definition

The task of the path planner in the navigation system is to generate 6-DoF paths between two given poses on a local scale, connecting the current robot pose or start pose, respectively, to a nearby goal pose on the robot's surrounding surface via a kinematically feasible and preferably optimal trajectory. Feasibility implies that the resultant paths always remain constrained to the perceived surface,

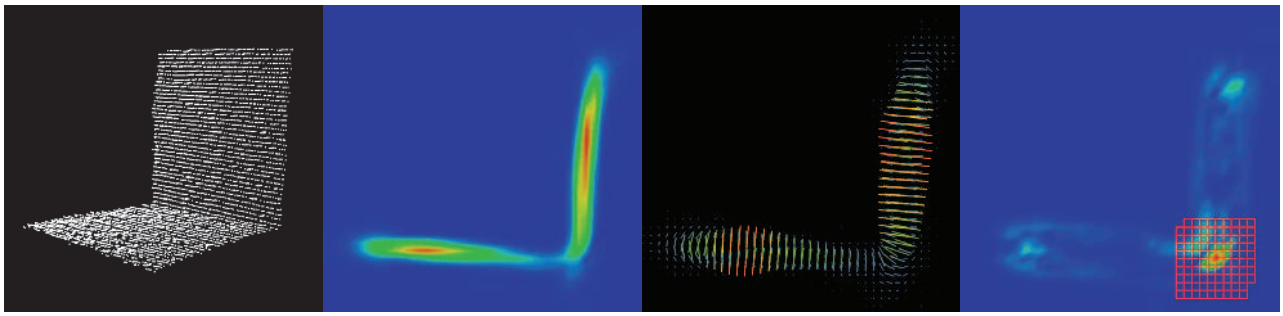


Fig. 9. Extraction of environmental feature maps: surface and edge maps are extracted from the tensor voting results for modeling the environment. The input tokens are shown in the first image, followed by a cross section of the surface saliency map, then a cross section of the surface normal map (scaled by saliency) and finally a cross section of the edge saliency with an overlay of the binary edge map.

i.e. unknown areas or uncertain areas of low saliency are avoided, and limits on the path curvatures are met, i.e. the paths are sufficiently smooth. Optimality aims at minimum travel time and increased safety in minimizing the path length and overall risk of a path. For the MagneBike robot, this means avoiding unknown areas, as well as reducing the attack angle when traversing high curvature regions. Sharp bends, including the negotiation of step transitions are only safe if the curvature lies in the robot's x–z (sagittal) plane, thus maintaining proper contact between the wheels and the surface. These criteria must be satisfied by the path planning algorithm in order to achieve practical results.

The main functionality of the proposed navigation system is to act as a local planner on an intermediate level in the system architecture. This works under the assumption that there is a human operator or some higher-level planner giving general guidelines on where the robot needs to go. Goal poses requested from a higher-level layer can be selected according to an arbitrary policy. In the context of robotic inspection, typical policies set out goal poses for exploration and coverage of the structure. As Section 8.2 will show, the navigation system can furthermore act as global planner and is capable of planning longer, more complex paths throughout large and noisy point clouds.

The dense tensor voting process creates a 3D grid-based environment model, and path planning will utilize the same grid structure in our case, linking a series of cells through the grid to the goal. The linking defines a graph $G(V, E)$, with nodes V representing the grid cells and edges E representing possible transitions. The best path towards the goal pose can then be found by using graph search algorithms, such as A* in our case, which have been applied in many mobile robotic applications, and have been proven capable of providing admissible, minimal cost paths in an optimal way (Hart et al., 1968). The graph-based path planner inherits desirable properties of optimality and completeness guarantees from the underlying graph search algorithm. Reasonable goal poses lie on the surface and are reachable by the robot. In the event of the goal pose falling outside the navigable subset of the registered point cloud, the planner reports failure of finding a feasible path.

In a more general formulation, our path planning could be applied to a more abstracted set of locations, as opposed to grid cells. The grid results from the densification step of tensor voting and provides a dense, regular structure. The way we establish connections bears some similarities to deterministic sampling on the grid; future extensions of the graph-based planner might increase the algorithm's adaptivity and scalability by introducing multi-resolution grids or iterative grid refinement, or trade off optimality against sparsity by exchanging the graph search by sampling-based planning (LaValle, 2006).

Definition of the graph structure $G(V, E)$ is not straightforward. Edge transition costs as well as heuristic functions must be chosen in a way that will satisfy all previously described constraints. In the next two subsections, further details behind the graph construction are discussed.

5.2. Graph connectivity

Each grid cell has associated structural information, and now the question arises of how this information can be used to define kinematically feasible movements between such cells (as illustrated in Figure 10), and locally connect the cells to a graph structure. The fact that robot motion is constrained to a 2D manifold reduces the number of cells to be searched, and only the cells in close proximity of the estimated surface need to be considered. Further, robot mobility is directly linked to the local surface properties, therefore the path planning algorithm must be cognizant of both the robot's and the structure's orientations. To achieve this, node state must also include orientation information. Thus, each cell in the 3D grid (defined by 3D coordinates in the world frame $\mathcal{F}_w$) is actually represented by a set of nodes $V_i \subset V$, which corresponds to a set of discretized heading vectors $\hat{h}_w$ (each of them defined in the global frame $\mathcal{F}_w$). The node states are illustrated in Figure 11 (only 6 heading vectors are shown here, as opposed to the 98 in our actual implementation).

Using this, node connectivity is established 'on the fly' by restricting transitions to those neighboring nodes (i.e. cell locations and orientations) deemed physically reachable (see Figure 12 for further illustration of the concept). This allows for the direct inclusion of some kinematic constraints, as well as allowing for the complete goal pose to be specified, while limiting the search space to tangential

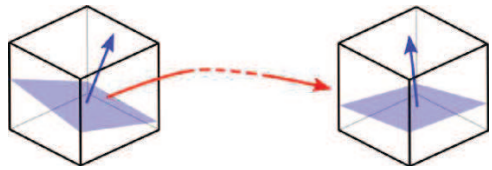


Fig. 10. Establishing movement from one cell to another is aided by the structural information given by the tensor voting framework, such as the local surface normal estimates shown here.

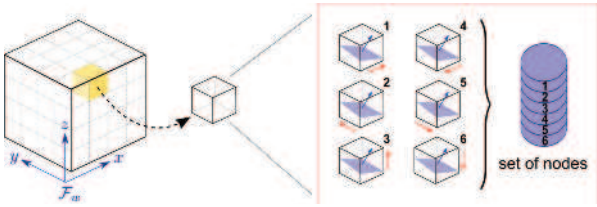


Fig. 11. Node state contains the 3D location (i.e. the cell location in our case), as well as the potential robot heading vector (here discretized into six directions, and shown by vectors drawn next to the cells). Therefore, each location has a set of associated nodes, one for each (discretized) orientation.

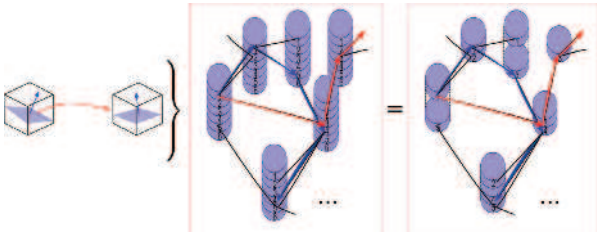


Fig. 12. Single nodes from each set of nodes are connected in a sparse graph, due to given constraints in the robot's and surface's orientations, which determine connectivity. According to these constraints, paths may go through the same physical 3D location (i.e. the same set of nodes) but with a different heading, and thus remain unconnected (i.e. different nodes).

movements along the estimated surface. Note the parallel to planning in state lattices, where cells of regular grids are linked by motion primitives (Pivtoraiko and Kelly, 2005).

The connections between nodes are established as each node is taken from the A* OPEN list. The connectivity for each node is dependent on the environmental feature maps provided by tensor voting (see Section 4.3), along with the robot orientation in the world frame $\mathcal{F}_w$. There are essentially two different scenarios, either the current node belongs to an edge structure, or it does not. The case is determined using the binary edge map, and specifies a predetermined set of subsequent movement vectors in the robot's local body frame $\mathcal{F}_b$. Figure 13 illustrates this concept for both scenarios. The transformation between the body frame $\mathcal{F}_b$ and world frame $\mathcal{F}_w$ is deduced using the robot's heading vector, along with the local surface properties. Once this transformation has been determined, the appropriate set of movements is transformed into $\mathcal{F}_w$, and each movement is then discretized by snapping it to the closest grid cell, providing the location coordinates of the possible child nodes. The movement vectors to these candidate child locations are a subset of all of the possible heading coordinates (illustrated by Figure 11), and so the full node state of the resulting candidates are determined accordingly. Next the surface saliencies are compared with a threshold Σ, and only nodes with strong surface values are permitted. Finally, the turn angle is calculated by projecting the movements into the local surface plane, and the graph connectivity is only established for turns smaller than a threshold T. Then the cost of this connection is evaluated to determine whether the node will be added to the OPEN list. The complete algorithm is outlined in Algorithm 1.

Algorithm 1 Establishing graph connectivity

($\hat{i}^b_w$, $\hat{j}^b_w$, and $\hat{k}^b_w$ denote the axes of the robot's body frame $\mathcal{F}_b$ represented in the coordinates of the world frame $\mathcal{F}_w$.)

if no edge **then**
  $\hat{k}^b_w \leftarrow$ current surface normal $\hat{n}_w$
  $\hat{i}^b_w \leftarrow$ current heading $\hat{h}_w$
  $\hat{j}^b_w \leftarrow \hat{k}^b_w \times \hat{i}^b_w$
  list of moves $\vec{m}_b \leftarrow$ regular moves (Figure 13)
else if edge **then**
  $\hat{k}^b_w \leftarrow$ current surface normal $\hat{n}_w$
  $\hat{j}^b_w \leftarrow$ current edge tangent $\hat{t}_w$
  $\hat{i}^b_w \leftarrow \hat{j}^b_w \times \hat{k}^b_w$
  list of moves $\vec{m}_b \leftarrow$ edge moves (Figure 13)
end if
rotation matrix $R = [\ \hat{i}^b_w \quad \hat{j}^b_w \quad \hat{k}^b_w\ ]$
for all possible moves $\vec{m}_b$ **do**
  Transform movement into wold frame, $\vec{m}_w = R\vec{m}_b$
  $\vec{m}'_w \leftarrow \vec{m}_w$ 'snapped' to the closest grid cell
  possible child position $\leftarrow$ current position $+ \vec{m}'_w$
  if surface saliency $> \Sigma$ **and** turn angle $< T$ **then**
    Establish edge connectivity
    Expand node by evaluating the cost of transition $\vec{m}'_w$ (add the child node to the OPEN list)
  end if
end for

This choice of graph connectivity ensures perpendicular traversal of edges in the environment, limited turn angles and staying within the close neighborhood of the best surface estimate. Once a valid connection between two nodes is established, costs based on the risk of traversing a connection are assigned to the graph edge in order to meet the remaining requirements of the path planning.

5.3. Cost functions and heuristics

After a node has been removed from the OPEN list, and its children have been found using the process described above, then the edge cost of traveling to each child node

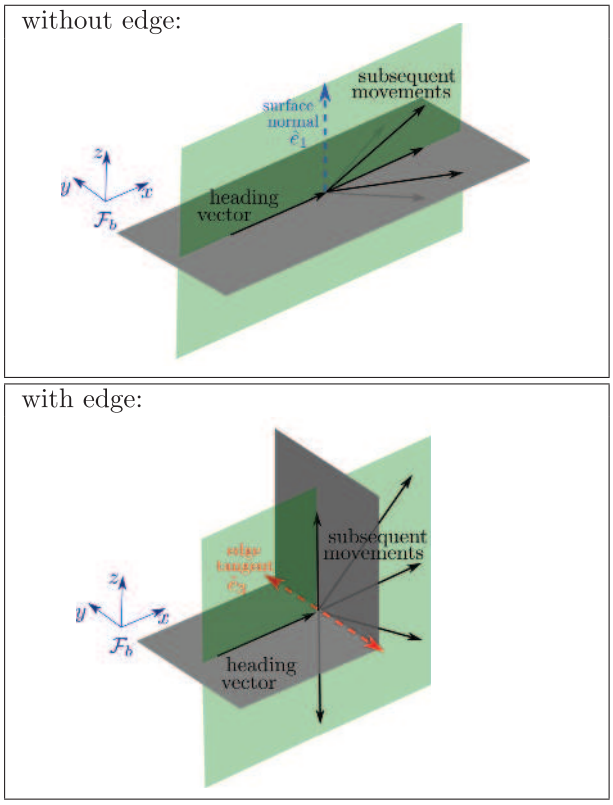


Fig. 13. Set of possible movement vectors $\vec{m}_b$, based on the robot's local frame $\mathcal{F}_b$ for the two possible cases: (1) regular moves, where the current node is not nearby an edge; (2) edge moves, where the current node is near an edge of the environment.

needs to be evaluated to determine whether this child node should eventually be added to the OPEN list. For the MagneBike robot, there are essentially four different contributions to the traversal costs between two nodes:

- distance (step size);
- surface saliency;
- turn angle;
- relative surface curvature.

The overall cost per edge is given in Equation (2), where each individual cost is weighted appropriately:

$$\begin{aligned} total\_cost = {} & k_0 \cdot (dist\_cost) + k_1 \cdot (surf\_cost) \cdot (dist\_cost) \\ & + k_2 \cdot \frac{turn\_cost}{dist\_cost} + k_3 \cdot (curvature\_cost). \end{aligned} \tag{2}$$

The details of each contribution to the cost function are described in the following subsections, with the exception of the distance cost, which is simply given by the Euclidean distance between a node and its child node, approximating the edge length as a straight line. The A* heuristic function is similarly given by the Euclidean distance from the child to the goal node.

5.3.1. Surface saliency In practice, when using lidar point clouds, the surface saliencies vary a lot and, typically, any values above a relatively low threshold are very close to the actual surface. Very high saliency regions usually result from very dense sampling in these areas. The token subsampling strategy described in Section 4.2.3 helps to reduce this saliency variation, but does not eliminate it completely. Higher saliency regions should have some preference, however not so much that the path meanders unnecessarily in order to traverse densely sampled areas. Therefore, the relative surface saliency is raised to an exponent (tuned empirically to $n = 4$ in our experiments), penalizing low saliency regions much more than high saliency regions:

$$surf\_cost = \left(\frac{max\_saliency - local\_saliency}{max\_saliency} \right)^n, \quad n \geq 1. \tag{3}$$

Note that in the final cost calculations, the surface cost is also multiplied by the step size, since longer steps over low confidence areas should have a higher penalty than shorter steps.

5.3.2. Turn angle Turn angles are calculated by projecting the movement vector $\vec{m}_w$ and heading vector $\hat{h}_w$ into a local surface plane, $\vec{m}_{w_{avg}} = \vec{m}_w - \hat{n}_{w_{avg}}(\vec{m}_w \cdot \hat{n}_{w_{avg}})$ and $\vec{h}_{w_{avg}} = \hat{h}_w - \hat{n}_{w_{avg}}(\hat{h}_w \cdot \hat{n}_{w_{avg}})$. The local surface plane is given by the normal vector $\hat{n}_{w_{avg}}$, which results from averaging the parent and child surface normals. The cost function is then obtained by taking the absolute value of the angle between the two projected vectors:

$$\begin{aligned} \Delta\theta &= \cos^{-1}\left\{ \frac{\vec{m}_{w_{avg}}}{|\vec{m}_{w_{avg}}|} \cdot \frac{\vec{h}_{w_{avg}}}{|\vec{h}_{w_{avg}}|} \right\}, \\ turn\_cost &= |\Delta\theta|^2. \end{aligned} \tag{4}$$

Similar to the surface cost calculation, the resulting angle is squared, such that two smaller turns are preferred over one larger one and, in addition, the turn cost is divided by the step size in order to prefer more gradual turns.

5.3.3. Relative surface curvature The MagneBike robot is capable of traversing curved surfaces, however, areas of high curvature can be problematic if approached at the wrong angle. Therefore, the angle between the plane of curvature and the robot's x–z plane should be minimized in areas of high curvature (see Figure 14). The curvature cost is therefore the product of the magnitude of change in surface normal and the approach angle:

$$\begin{aligned} & curvature\_cost \\ & = |change\_in\_surf\_normal| \cdot |approach\_angle| \\ & = \cos^{-1}(\hat{n}_{parent} \cdot \hat{n}_{child}) \cdot \cos^{-1}\left(\frac{\hat{n}_{parent} \times \hat{n}_{child}}{|\hat{n}_{parent} \times \hat{n}_{child}|} \cdot \hat{j}_w^b \right) \end{aligned} \tag{5}$$

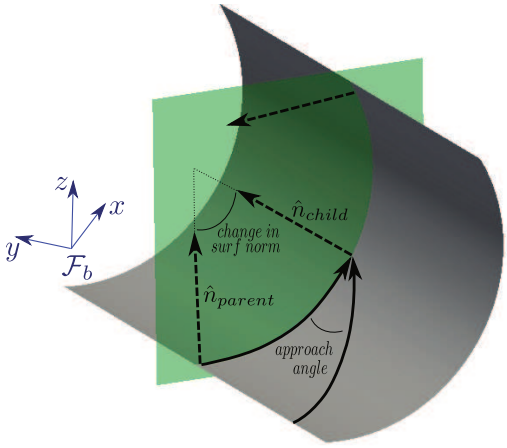


Fig. 14. Traveling at large angles with respect to the curvature plane is penalized with higher costs.

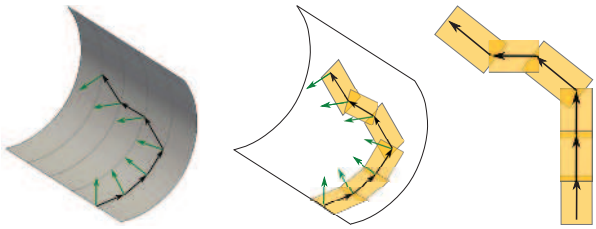


Fig. 15. Transforming a 6-DoF path on a curved surface into two dimensions, by projecting each step into its local surface plane.

where $\hat{n}_{parent}$ and $\hat{n}_{child}$ are the surface normals located at the parent and child nodes respectively, and $\hat{f}_w^b$ is the y-axis of the robot's local body frame, represented in the coordinates of the world frame. The change in surface normal should intuitively be inversely weighted by the step size, however the overall curvature cost should be directly weighted by the step size, therefore canceling out any dependence on step size.

6. Control scheme

Once a feasible 6-DoF path through 3D space has been generated, a control strategy is required in order to steer the robot from the start to the goal pose. Despite having suitable surface paths, the path following or trajectory tracking task is far from simple in such a constrained environment. Section 6.1 describes a method which transforms the 6-DoF way points (given by a sequence of connected nodes) into a 2D frame. Hence, the robot can be modeled in two dimensions, which allows for the use of simpler 2D control schemes. Section 6.2 models the robot as a front-wheel-drive bicycle in regard to the MagneBike platform, and nonlinear feedback is used in Section 6.3 to track the computed trajectory according to the chosen model.

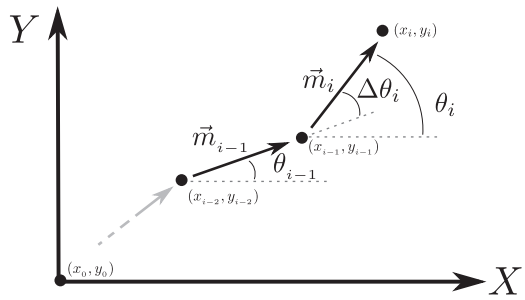


Fig. 16. Iterative computation of 2D way points based on the projected 3D movements and corresponding turn angles.

6.1. Path dimensionality reduction

Although the motion is in 3D space, the robot is always constrained to travel on a 2D manifold. Therefore, similar to the idea by Furgale and Barfoot (2010), the path generated by the A* algorithm can be transformed into a lower-dimensional space before a control strategy is applied. This is done by iteratively projecting each movement into its local surface plane, and aligning the 2D movements based on the relative yaw angles between them, essentially unwrapping the path from the surface. An illustration of this transformation can be seen in Figure 15.

Local surface planes are found by averaging the surface normals of the corresponding parent and child nodes. The movement vectors can then be projected onto the local surface, and the relative turn angle $\Delta\theta$ between the movement vectors can be calculated (see Section 5.3.2, Equation (4)). The 2D coordinates of each way point are then iteratively given by the coordinates of the previous way point plus the projected movement $\vec{m}_i$, rotated by $\Delta\theta_i$ with respect to the previous movement vector. This formulation is shown in Figure 16 and Equation (6):

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} x_{i-1} \\ y_{i-1} \end{bmatrix} + |\vec{m}_i| \begin{bmatrix} \cos\theta_i \\ \sin\theta_i \end{bmatrix}, \qquad \theta_i = \theta_{i-1} + \Delta\theta_i, \tag{6}$$

where x_0 and y_0 are initialized at the origin of the 2D control frame, and θ_0 is initialized to zero.

6.2. Modeling the MagneBike robot

Now that the desired path has been mapped into 2D space, the motion model can be simplified to two dimensions as well. The MagneBike robot is approximated by a simple front-wheel-drive bicycle, with control inputs given by the speed $v_s(t)$ and steering angle $\phi(t)$ of the front wheel. This model is shown in Figure 17 and Equation (7), where l denotes the distance between the two wheels:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v_s(t)\cos\phi(t)\cos\theta(t) \\ v_s(t)\cos\phi(t)\sin\theta(t) \\ v_s(t)\sin\phi(t)/l \end{bmatrix}. \tag{7}$$

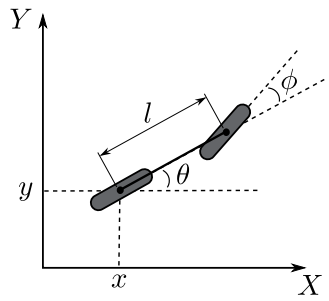


Fig. 17. Simplified front-wheel-drive model of the MagneBike robot.

Equation (7) can be rewritten in a simpler form

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta(t) & 0 \\ \sin\theta(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix}, \tag{8a}$$

with

$$\begin{aligned} \phi(t) &= \tan^{-1}(\omega(t)\, l / v(t)), \\ v_s(t) &= v(t) / \cos\phi(t), \end{aligned} \tag{8b}$$

where $v(t)$, the speed of a reference point on the rear wheel, and $\omega(t)$, the angular speed around the rear wheel, are now the control inputs of the system. The original control inputs, $v_s(t)$ and $\phi(t)$, can be calculated directly from $v(t)$ and $\omega(t)$. Note that this new expression for the system equations is analogous to that of a differential-drive robot, which has been well treated for control strategies in the literature. One possible method of control using state feedback to track a parametrized trajectory is outlined in the following section.

6.3. Trajectory tracking

In order to control the robot along the given 2D path, a set of nonlinear feedback controls for trajectory tracking can be used. This is done for MagneBike by using the kinematic model described in the previous section, and then following a virtual reference bike with known state along the path, as suggested by Samson and Ait-Abderrahim (1991). To establish the state of the reference bike for all time, a B-spline is fitted through the 2D way points, creating a continuous, parametrized path, from which the state of the virtual reference bike can be inferred. For simplicity, the reference bike will be assumed to follow the spline at a chosen constant velocity v_r. Once the state of the reference bike has been determined, a nonlinear feedback law can be used to stabilize the position and orientation error of the MagneBike robot, relative to the reference bike, to zero. Position and orientation feedback are assumed to be provided by a localization system, such as that described by Tâche et al. (2011). The state vector of the system is given by

$$X = \begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_\theta \end{bmatrix}, \tag{9}$$

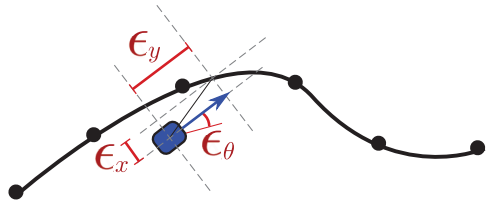


Fig. 18. Tracking a virtual reference bike through nonlinear feedback of the pose error.

where ϵ_x and ϵ_y represent the relative position of the reference bike in MagneBike's local frame $\mathcal{F}_b$, and $\epsilon_\theta = \theta - \theta_r$, where θ and θ_r are the respective orientations of the bike and reference bike in the control frame (see Figure 18). The feedback control laws suggested by Samson and Ait-Abderrahim (1991) are

$$\begin{aligned} \omega &= \omega_r - \tfrac{k_3}{k_2}\epsilon_\theta + \tfrac{k_6}{k_2}\epsilon_x \\ &\quad - \tfrac{k_1}{k_2}\left(\delta_x\, \omega_r \tfrac{\cos\epsilon_\theta - 1}{\epsilon_\theta} - v_r \tfrac{\sin\epsilon_\theta}{\epsilon_\theta}\right)(\epsilon_y + \delta_x\, \epsilon_\theta), \\ v &= v_r + k_3 k_5 \epsilon_x \\ &\quad + \left[2k_3k_4 + \left(v_r \tfrac{\cos\epsilon_\theta - 1}{\epsilon_\theta} + \delta_x \omega_r \tfrac{\sin\epsilon_\theta}{\epsilon_\theta}\right) + k_6\right]\epsilon_\theta \\ &\quad + \left[(1 - k_1)\,\epsilon_y - k_1 \delta_x\, \epsilon_\theta\right]\omega, \end{aligned} \tag{10}$$

where v_r and ω_r are the velocities of the reference bike, and δ_x is the control point offset given by the distance along the local x-axis from the rear wheel. In addition, the control gains can be tuned empirically under the following conditions: k_1 and k_2 are positive real numbers, k_3, k_4 and k_5 are positive scalars assumed to be constant, $0 \leq {k_4}^2 < k_5$, and k_6 is any real scalar. The state vector X is guaranteed to converge to zero, so long as v_r and ω_r are differentiable for all $t \geq 0$, these derivatives remain bounded, and the reference bike does not stop moving.

Since a 2D control strategy is used, any 3D pose feedback needs to be transformed into the 2D control plane before it can be applied. This can be done by finding the nearest way points in the 6-DoF path in 3D space, and then using their corresponding surface normals to project the position and orientation vectors into the local surface plane to find the approximate offsets $[\tilde{\epsilon}_x, \tilde{\epsilon}_y, \tilde{\epsilon}_\theta]$.

7. Experimental evaluation

Validation of the proposed navigation system was done through a series of experiments, which are explained and discussed in this and the next section. Section 8 evaluates the navigation system based on the specific example of the MagneBike robot and the steam chest inspection scenario, and in this process relies on point clouds from real lidar sensors. However, the experiments of this section are kept more general. The evaluation does not depend on MagneBike, and is done based on simulated, computer-generated point clouds. The goal is to study particular behaviors of the overall navigation system. Insight into

the influence of various parameters, and how they can be assigned, is provided. Results show that tensor voting can be sufficiently robust to density and noise variations, and that the path planning algorithm is capable of providing feasible and low-cost paths over curved surfaces.

7.1. Evaluation of tensor voting

Lidar point clouds suffer from noise as well as irregular and anisotropic density distributions. These characteristics have a negative influence on the tensor voting outcome, and therefore need to be investigated to ensure that meaningful results can still be obtained using the lidar point clouds.

The scale at which the tensor voting is performed, determines what types of features are detected. As a result, the proper choice of σ is crucial. If σ is too large, structures will be over-smoothed, and small steps or holes might be overlooked. Alternatively, if σ is too small, then noise in the point cloud (such as ripples) could be detected as features. The scale parameter σ is one of the few parameters to adjust in the tensor voting subpart of the navigation system, but it must be chosen in accordance to the robot model and the robot's environment. Therefore, we identified $\sigma = 6$ cm for MagneBike, which will be described in Section 8.1, and use this value in all of the experiments within this and the next section. Although different values for σ could have been chosen, the conclusions gained from the evaluations in this section are independent of σ.

7.1.1. Robustness against variation in point density First, the variation of density was investigated. The investigation was carried out on two different shapes, a cylindrical tube structure and an L-shaped structure, which provide examples of the basic structural forms. The point clouds were artificially generated by randomly distributing points over the surface. We use the deviation in estimated surface orientations as a measure for the robustness of tensor voting. The effects of density variation can be seen in Figure 19, where the error in surface orientation is plotted versus the average distance between points (expressed as the relative average distance $\bar{d} = d/\sigma$). The error values were determined by calculating the angle between the estimated and actual orientation vectors at 6,000 sample locations on the structures. When evaluating orientation vectors, deviations in surface normals were checked where the stick saliency was dominant (i.e. not near an edge), and deviations in edge tangents where the plate saliency was dominant (i.e. at edges). It can be concluded from these plots that the error remains well behaved, even when point density drops so low that the shape is hardly recognizable. Note that because each sample density is plotted in terms of the distance between points, the actual size of the point cloud decreases by the square of these values. Given our experience with the robot configuration and targeted environments, we observed that typical values for the density of real lidar point clouds generally fall within the first two (highest density) samples in the plots, with an inter-point distance d of less than 2 cm (or 0.33σ). Areas with much higher inter-point distances will result in low saliency values and will thus be handled implicitly as unsafe regions by the path planner.

In addition, the effects of the token subsampling parameter β were analyzed. This was done by creating point clouds with varying density, again using the tube and L-shape structures. These point clouds can be seen in Figure 20, along with graphical results of the error values, and computation time, as β is increased. These plots show that computation time of dense voting can be drastically decreased, without compromising the outcome of the tensor voting. During all of the experiments in this paper, β was chosen to be 0.5, due to the fact that higher values of β show little further impact on time. As discussed in Section 4.2.3, this method of token reduction also unifies the point density, resulting in more uniform saliency distributions, and therefore a better representation of the actual surface.

7.1.2. Robustness against variation in noise Noise variation was then evaluated using the same techniques as for the density experiments. The tests were conducted on the same tube and L-shaped structures, this time as noise levels were increased. The results are plotted in Figure 21, showing the error in surface orientation, in relation to the standard deviation of the Gaussian noise distribution. The images above the graphs illustrate the extent of noise variation for the tests. In real applications, this noise depends on the sensor used and the type of reflecting surfaces. Characterization of the URG-04LX and the UTM-30LX laser scanners over different metallic surfaces shows that the standard deviation on depth measurement is 0.028 m and 0.018 m, respectively (Pomerleau et al., 2012). One can see that with this range of noise, the orientation errors in the structure prediction from the tensor voting remain below approximately 2.5°.

7.2. Evaluation of path planning

Now that the validity of the tensor voting procedure has been established, the results can be used to evaluate the path planning algorithm. We demonstrate that any generated trajectories are able to satisfy the mobility requirements specified in Section 1.1, namely the ability to climb on complex surfaces, overcome step obstacles in a perpendicular direction, and avoid unnegotiable obstacles. Note that all of the paths presented in this section are the result of processing a single point cloud, with the start and goal poses as input. No intermediate way points are provided by the user.

7.2.1. Impact of cost functions To start, the impact of the different cost functions is illustrated. This is done in the left-hand side of Figure 22, where one can see several paths on a tube, which has a hole-type obstacle on the top part of the surface. In this scenario, a trajectory must be generated from one side of the hole to the other. The cost function of Equation (2) with non-zero weights is used for all paths in this figure; each path emphasizes the effect of the different cost function weights. One path represents a search with

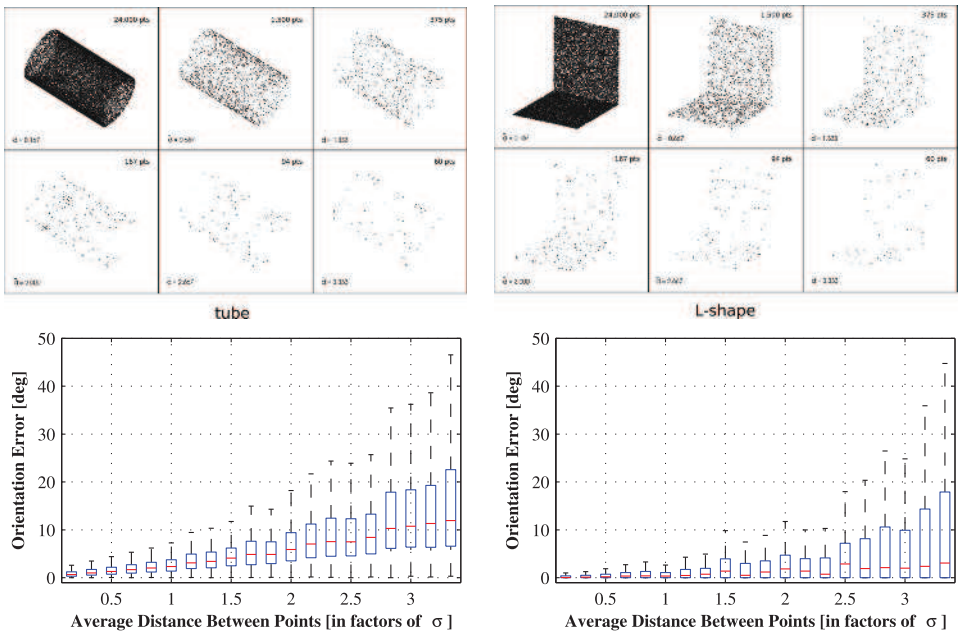


Fig. 19. Error in estimated orientations, predicted from tensor voting performed on a tube structure (left) and an L-shaped structure (right) as density is varied. The results are shown in terms of average relative distance, $\bar{d} = d/\sigma$. The density is inversely proportional to the square of this value. A few samples of the point clouds that were used are shown above.

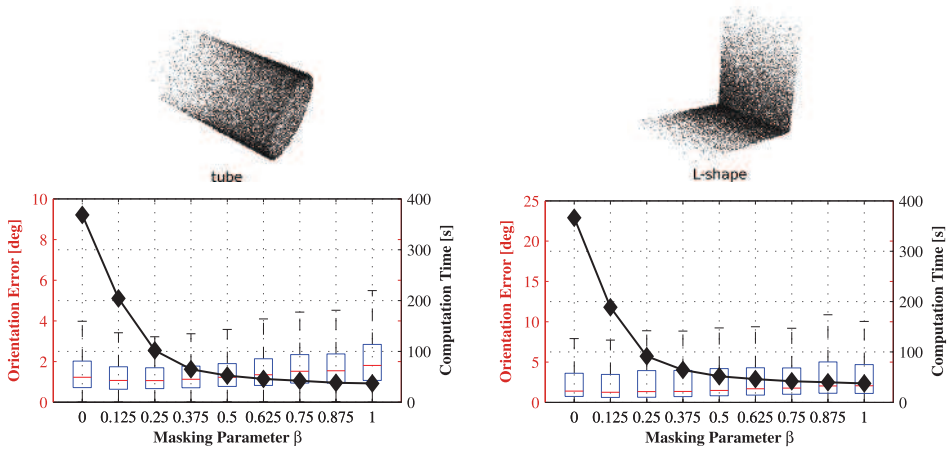


Fig. 20. Error in estimated orientations, predicted from tensor voting performed on a tube structure (left) and an L-shaped structure (right), as well as computation time, as the masking parameter β is varied. Both point clouds contain 24,000 points.

high turn cost, therefore traveling through unsafe areas near the hole. Another path demonstrates the significance of the relative curvature cost; the path always prefers to travel in the plane of curvature, or in a direction where there is no curvature. A third path shows a preference for remaining on highly salient areas of the surface, avoiding the low saliency obstacle. The weighting of these cost functions can be tuned to suit the design of the robot, nature of the environment, and the safety requirements. An example of this is shown in a path which balances the effects of each weight in order to get a safe, yet efficient path.

7.2.2. Perpendicular edge traversal Next, the ability of safe edge traversal is displayed in the right-hand side of Figure 22. The analysis is performed on an L-shaped structure, and the image shows the detected edge feature, as well as several paths traversing the edge perpendicularly. Each path was obtained using a different inflation radius for the binary edge map. The three paths correspond to increasing radii of 0 cm, 10 cm, and 20 cm. This shows that depending on the safety requirements and the size of the robot, edges can be approached from a safe distance in order to ensure successful step traversal.

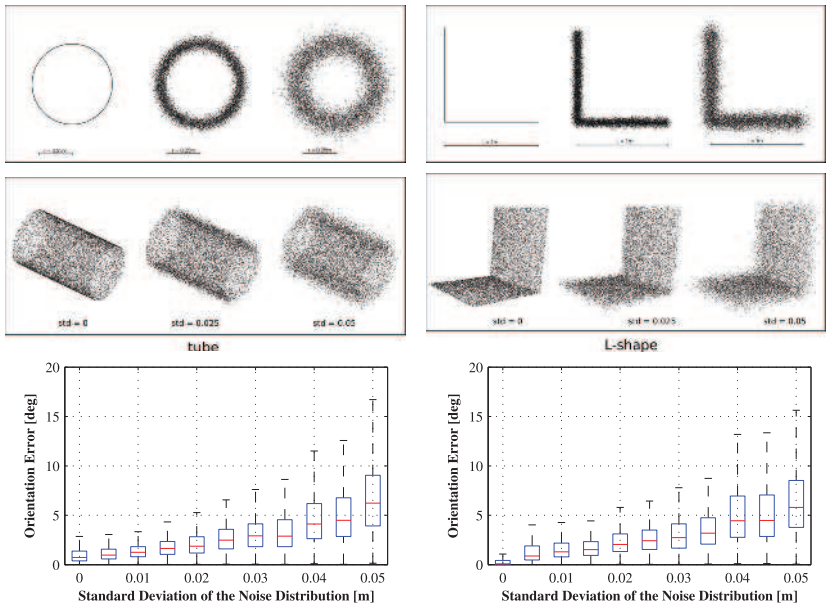


Fig. 21. Error in estimated orientations, predicted from tensor voting performed on a tube structure (left) and an L-shaped structure (right) as noise is varied. The results are shown in relation to the standard deviation of the Gaussian noise distribution, which was used to generate the point clouds. A few samples of the point clouds are shown above. All point clouds contain 12,000 points.

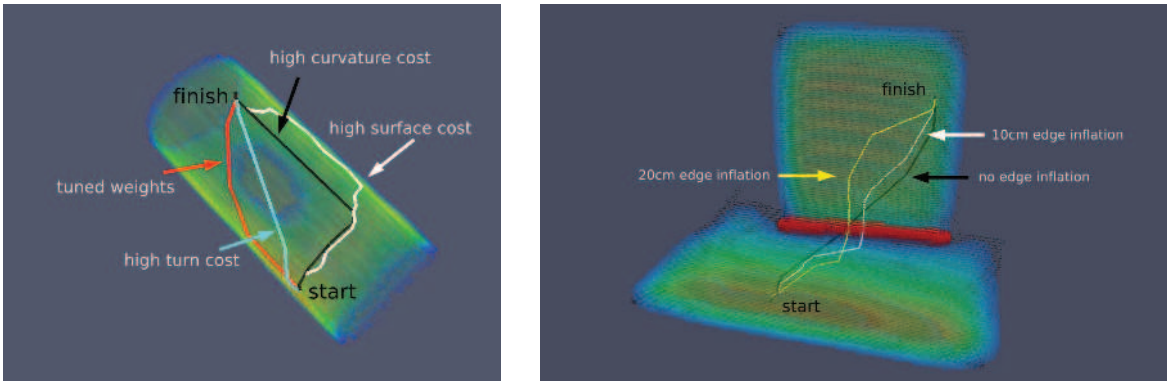


Fig. 22. Left: Illustration of paths resulting from different cost weights. The object pictured is a tube structure with a hole in it, with surface shading representing relative surface saliency. The four paths represent different settings of the cost function weights. One path emphasizes curvature cost, one turn cost, one surfaceness cost and one shows a practical balance between the different weights. Right: Demonstration of paths resulting from varying the edge inflation radius. The paths show 20 cm, 10 cm and 0 cm inflations. Again, surface shading represents the relative surface saliency. The detected edge of the L-shape structure is also highlighted.

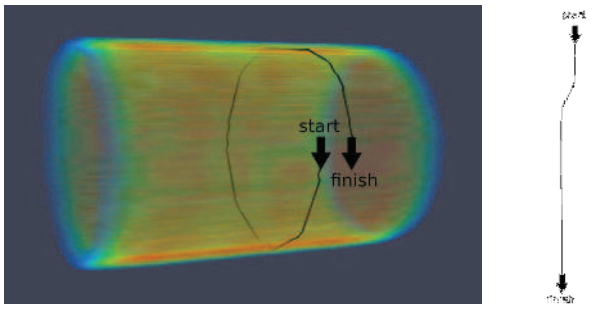


Fig. 23. An example of an almost circular path, along with the corresponding unwrapped 2D path. The robot is traveling in a loop on the inside of the tube.

7.2.3. Planning more complex paths Figure 23 provides an example of traveling on a non-planar surface with the ability to follow an almost circular path. In this scenario and for given cost weights, looping around the tube represents a safer path, in comparison to turning around on the highly curved surface. The equivalent 2D path is also shown, demonstrating the result of the unwrapping procedure described in Section 6.1.

Another example on a more complex surface highlights the full abilities of the navigation system. This can be seen in Figure 24, which shows the input point cloud, the feature detection results, as well as the 3D and 2D trajectories. In this example, the robot must travel perpendicularly over the

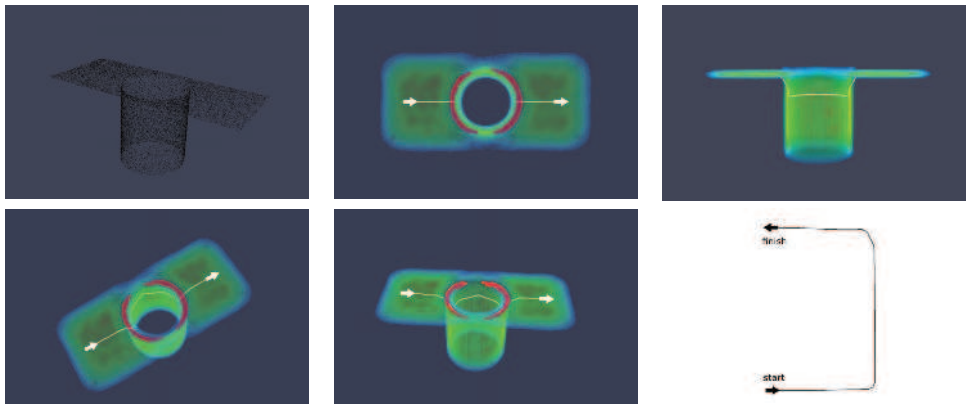


Fig. 24. The above results exemplify the ability of the algorithm to provide paths on a highly complex surface. The input point cloud and 2D path are also shown. Here the robot must travel down and around in the tube section in order to cross to the other side of the plate. Note that the edges, which are highlighted, are crossed perpendicularly when entering and exiting the tube section, according to the initially specified requirements on attack angles for step climbing.

detected edge, around in the tube (mostly traveling in the plane of curvature), and again perpendicularly over the next edge, in order to cross to the other side of the plate safely.

8. Deployment on the MagneBike climbing robot

The presented navigation system has been fully integrated with the MagneBike robot. Once all of the components were implemented, thorough testing of the entire system was conducted. Implementation details and system evaluation are discussed throughout this section. Section 8.1 describes the overall setup. Section 8.2 presents the planning of global paths in a real steam chest environment. Section 8.3 demonstrates how MagneBike plans local paths to navigate in a smaller steel tube. The test results from both sections confirm the navigation system's capability of environment representation, 3D path planning, and control, while avoiding or overcoming obstacles as necessary. Section 8.4 concludes with a discussion of remaining issues relevant for the end-to-end use case, such as path replanning, exploration and sensing in occluded environments.

8.1. MagneBike platform

Successful implementation requires full integration of the three subsystems for perception/representation, path planning and control with the MagneBike robot. The MagneBike software is implemented in C++ using a Robot Operating System (ROS) framework (for a ROS overview, see Quigley et al. (2009)). The implementation of our navigation solution is also realized in ROS/C++, allowing for easy and modular integration with other software components. Modularity is important, so that subsystems can be easily added or removed, or also used in an iterative process. An outline of our implementation can be seen in Figure 25, showing the process of taking a scan of the environment, performing tensor voting, planning a path and

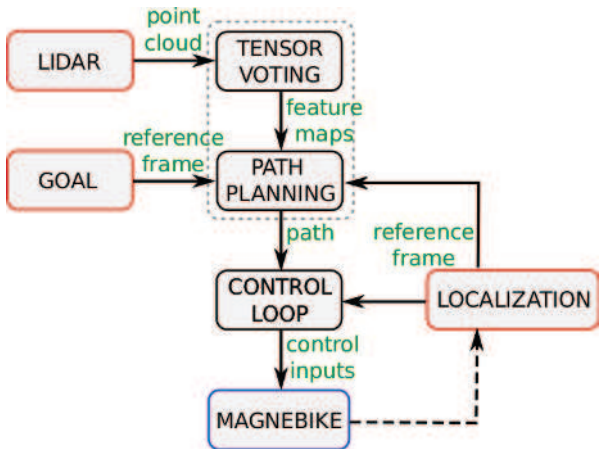


Fig. 25. Graphical representation of the architecture of the navigation system under ROS.

traveling along a computed trajectory towards the desired goal. Only low-level control processes were performed on-board MagneBike, the rest of the computations were performed in real time, but off-board. The tensor voting and path planning components of our current implementation are co-dependent, since the path planning algorithms specifically rely on the tensor voting results. Therefore, these two components were implemented within the same ROS node, reflected by the gray dashed-line box in Figure 25. The architecture allows for iterating the entire navigation process based on a set of way points. The system has been set up to continuously wait for new inputs and recompute required information when necessary. If a goal lies outside the currently available point cloud, failure to reach the goal is reported, which permits the system to take new actions.

The MagneBike robot is currently equipped with a rotating Hokuyo URG-04LX laser range finder for retrieving point clouds of its surroundings. However, work on future

MagneBike prototypes equipped with a Hokuyo UTM-30LX laser range finder is underway. From our experience, the UTM-30LX provides significantly better results in both range and accuracy, especially in reflective environments. An example of this can be seen in Figure 30, where scans of a steel test environment are compared. As a result of the poor performance of the Hokuyo URG-04LX in metallic environment, most of the experiments have been conducted using a rotating Hokuyo UTM-30LX, operated independently from MagneBike. Despite this, useable results have still been achieved with the Hokuyo URG-04LX, as shown in Figure 32 and then demonstrated in Section 8.3.2. For the sake of clarity, we will specify for each experiment in this section which of the Hokuyo laser range finders has been used.

In addition to the laser point cloud, robot pose estimates are required for the control algorithm. The laser point cloud of the real steam chest in Section 8.2 was collected in the course of a field experiment, where the robot was navigated through the environment by remote control; here the robot localizes itself using the method presented by Tâche et al. (2011). Section 8.3 in contrast, uses a Vicon motion capture system rather than MagneBike's own localization system in order to focus the problem to the previously unresolved tasks of environment representation and path planning. Therefore, reflective markers have been rigidly attached to the MagneBike robot to produce reliable 3D position and orientation information, and can be seen in the included image and video content.

8.1.1. Identification of the scale parameter Before experiments can be conducted, the scale parameter σ of the tensor voting subsystem must be determined. Intuition would say that relevant features with respect to locomotion will likely occur at roughly the same scale as the robot's wheel size. Therefore, a good starting guess for the scale parameter σ is the wheel diameter of the MagneBike robot ($\sigma = 6$ cm). In order to experimentally determine σ, tensor voting was performed on a small step feature of roughly the same height as the wheel diameter (and, therefore, about the scale where features become relevant to MagneBike). The plate or edge values for various σ values were analyzed. Figure 26 shows that smaller scale voting at 2 cm causes noise to trigger high edge saliencies, and large-scale voting at 10 cm causes the step to be blurred over a large distance. We can see that our intuition was correct, and a good choice for σ is in the expected range, so within this work a σ value of 6 cm will be assumed for all tests. Note that the determination of σ is inherently more challenging for smaller robots, such as MagneBike, because the relative scale of noise compared with feature size is quite large.

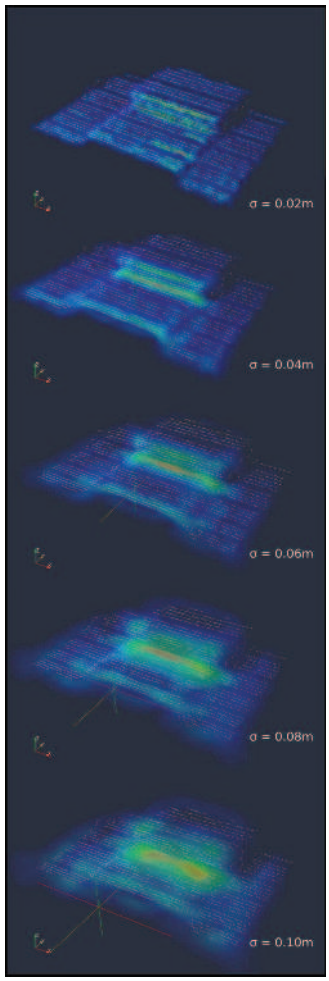


Fig. 26. Results of plate saliency for various σ values, performed on a typical lidar point cloud. This lidar scan containing 3,200 points was taken of a small metal step (see also Figure 3), approximately 6 cm high (using a Hokuyo UTM-30LX laser range finder). Surface shading represents the relative plate saliencies. The cell size used was 1 cm^3. White dots show the original point cloud used for tensor voting.

8.2. Planning paths through a steam chest

To evaluate the robustness of the path generation against real sensor noise, we use the point cloud created in the work of Tâche et al. (2011). The global map was realized by operating the MagneBike robot by remote control in a real steam chest, which was removed for maintenance purposes. The pipe is over 4 m long and has seven entry points (see Figure 27 on the top). For the global map, 59 scans were recorded using the rotating URG-04LX laser scanner mounted on MagneBike, and registered along a 5.8 m long path. Figure 27 in the center shows a cut view of the resulting global point cloud. The shading of the points follows the scan acquisition numbers, while the white line represents the trajectory of the laser and the spheres every pose where a scan was taken. This map highlights several challenges that a real inspection procedure may create. The exit points have a lower density than the core of the steam chest. The long middle section of the steam chest reveals the noise which is created by deformed scans and registration

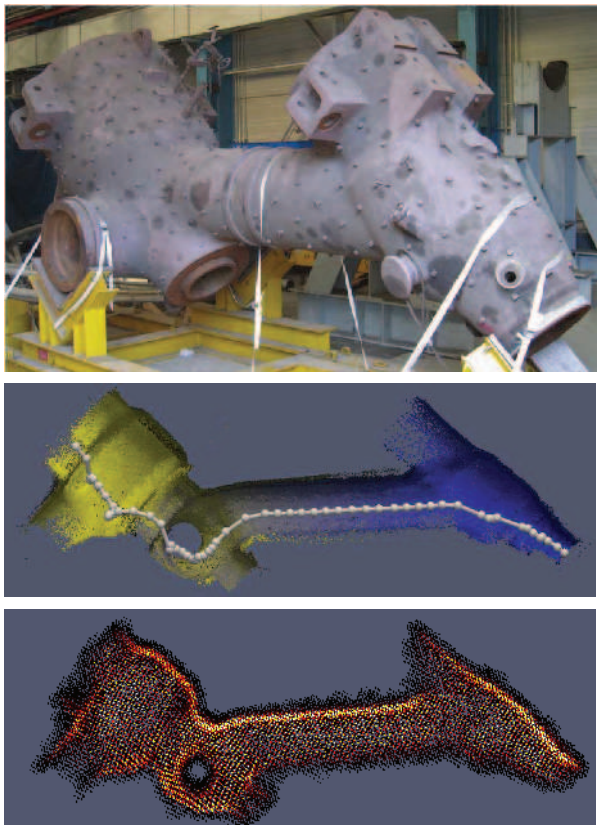


Fig. 27. Real steam chest environment. Top: Example of a steam chest, which was used as the test site. Center: Global map resulting from registered point clouds. The shading of the points follows the scan acquisition numbers and the white line describes the trajectory of the laser scanner. Bottom: Saliency map after dense voting has been performed. Grid cells with very low saliency were removed for better visualization.

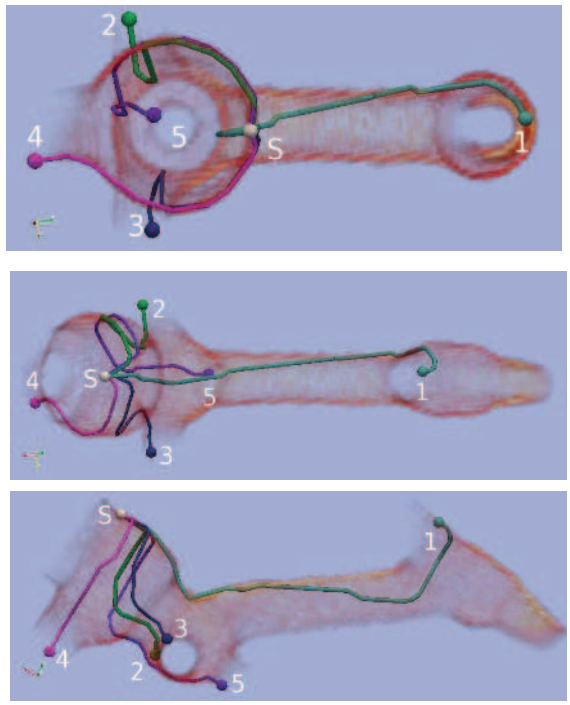


Fig. 28. 6-DoF paths through the steam chest. Top and side views of the five generated paths. The starting position of the robot is at the large opening of the steam chest, the goal poses are placed at five of the exit points. The global map is faded for better visualization of the planned paths.

errors. Finally, the lower part of the middle section has a very low point density due to the scanner position and robot self-occlusion during the point cloud acquisition process.

Figure 27 on the bottom presents a cut view of the same representation after dense tensor voting reconstruction. The lighter shades represent high saliency surface regions. Note that the regions of low density are still represented but obtain a much lower saliency. The observation holds for the exit points of the steam chest, as well as for the lower part of the middle section, which is less salient when compared with its upper part. Finally, the noise present in the middle section is minimized, as we can observe from the fact that the main saliency defines the shape of the pipe, even in the middle section, properly.

For the path planning, we use five of the exit points as final goals with the large opening of the steam chest as starting point. Figure 28 presents the resulting paths through the steam chest. The starting point is marked with an 'S', and the five goal poses with their respective path number. The paths are properly contained within the surface, even in presence of noise and variable density. Path 1 follows the high saliency zone of the middle section on half of its length before leaving with a smooth turn to finish in the opposite orientation. Paths 2 and 3 present almost symmetric trajectories, even though the goal positions are not set to the same height. This supports that the planning is repeatable under steady environment conditions.

Figure 29 shows the allocation of computation time for each of the paths shown in Figure 28. These figures justify the applicability of the system. The entire computation time is within about 30 s, the same order of magnitude as the time it takes for MagneBike to record a single 3D point cloud (about 50 s), and therefore can be considered a very usable result. This particular point cloud contains 32,000 points, which explains why the sparse tensor voting accounts for the majority of the computation time. After sparse voting, the number of tokens is reduced to 8,000 using the reduction methods explained in Section 4.2.3. This drastically reduces the computation time of the dense voting, despite having dense grid information at a resolution of up to 880,000 voxels with 3.5 cm side lengths. Further savings in computation time can be achieved by using an approximate range-limited nearest-neighbor search in the sparse voting.

This experiment shows positive results in view of planning complex paths in three dimensions based on a full size inspection environment. In order to enhance the observations gained from this section, and further validate the

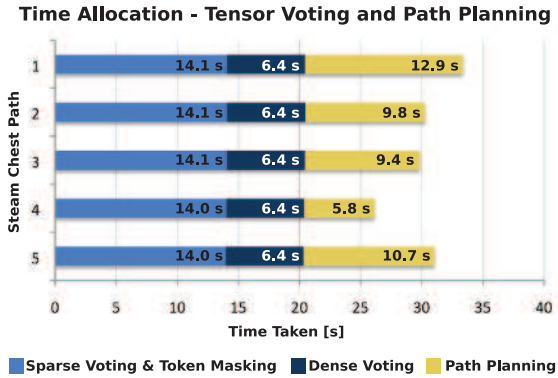


Fig. 29. Breakdown of computation times corresponding to the paths displayed in Figure 28.

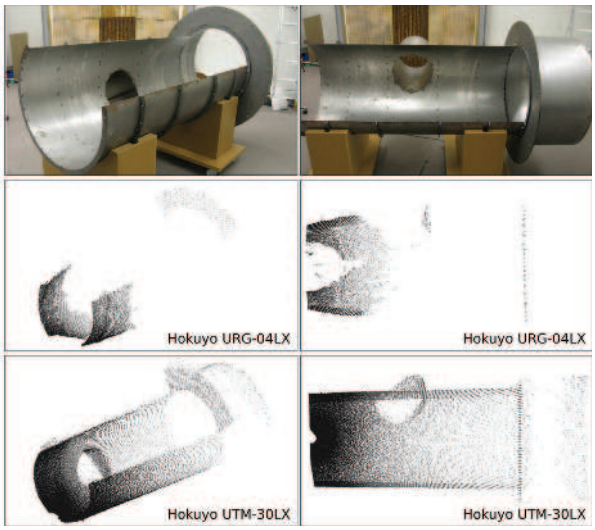


Fig. 30. Comparison of two Hokuyo laser range finders, URG-04LX and UTM-30LX, using scans of the test environment. The scans were taken from the bottom of the left end of the tube, with the lasers rotating around a vertical axis. The UTM-30LX shows large improvements in both range and accuracy for this metallic surface.

feasibility of planned paths, we realized a smaller scale experiment in a steel tube setup.

8.3. Navigation inside a steel tube

This section outlines a number of experiments, which illustrate the capabilities of the navigation system when applied to MagneBike. Three representative tests have been chosen to demonstrate the ability of full 3D navigation, obstacle avoidance, and perpendicular edge traversal. The documentation of these experiments is additionally supported with video footage given in Extensions 1, 2, and 3.

8.3.1. Test setup For realistic evaluation of the proposed navigation solution, a test setup was built to mimic the expected steam chest environment. The test setup consists of a steel tube composed of three parts: one main cylinder of 0.8 m diameter and 1.9 m length, an expanded section of 1 m diameter and 0.5 m length, and a small 0.4 m diameter and 0.6 m long cylinder branching off from the main section. In order to track the paths of the robot with the Vicon motion capture system, the top portion of the main tube can be taken off, allowing cameras to see inside. Images of this test environment can be seen in Figure 30, with the top removed. A series of experiments conducted using this test environment will now be described in the following subsections.

8.3.2. Experiment 1: Circling around the tube The first experiment requires MagneBike to travel on a circumferential path around the tube, completing just over half a loop. This is a task that outlines the success at planning and following true 6-DoF paths in 3D space, in contrast to standard 2D projection approaches, which would fail in this scenario. Figure 31 shows the successful results using a point cloud given by the rotating Hokuyo UTM-30LX, while Figure 32 shows an analogous test using a point cloud given by the rotating Hokuyo URG-04LX mounted on the MagneBike robot. One can see that despite the low quality of the point cloud shown in Figure 32, practical results were still achieved, and MagneBike reached its goal pose safely. The trials plotted in Figure 31 can also be seen in a video of the test in Extension 1.

8.3.3. Experiment 2: Avoiding a hole obstacle In the second experiment the robot is set up to navigate around a hole obstacle, similarly to Figure 22. If the hole was not there, then a simpler, direct path could be chosen which runs diagonally through the tube. This test was done using a point cloud provided from the rotating Hokuyo UTM-30LX sensor. As a result, MagneBike is again successful in reaching the goal, as shown in Figure 33, and in a video of the test in Extension 2.

8.3.4. Experiment 3: Traversing an edge obstacle The final experiment requires MagneBike to maneuver from the main tube section into the small section branching off, demonstrating its ability to follow more complex trajectories with additional constraints. To be successful, it must detect the edge at the junction of the two tubes and cross it perpendicularly. See Figure 34 for the presentation of the results, and Extension 3 for a video of the experiment.

8.4. Discussion of end-to-end use case

In this and previous publications (Tâche et al., 2009, 2011), we have developed and tested all of the subsystems that are required for the successful deployment of MagneBike in the use case of robotic inspection of a steam chest environment (as shown in Figure 27). In the following, we discuss some of the remaining issues we expect to find along the

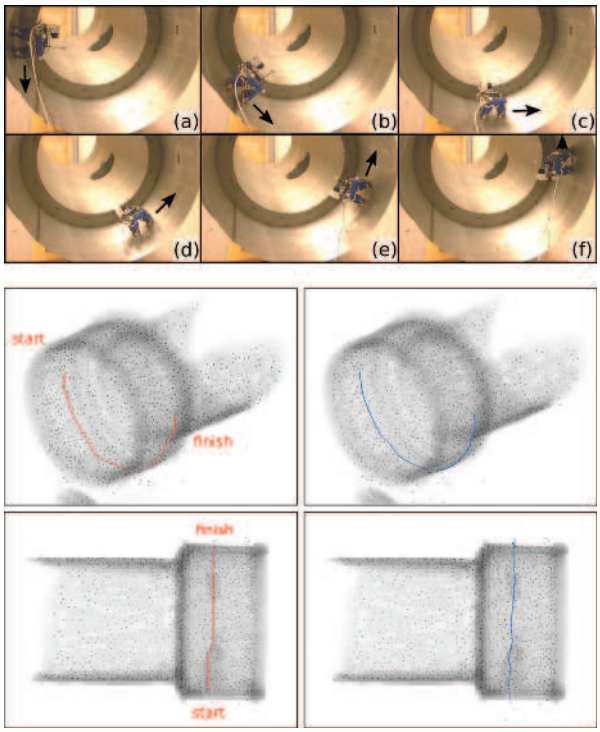


Fig. 31. Experiment 1(a): Tracking a circumferential path (using a Hokuyo UTM-30LX point cloud containing 15,000 points). An image sequence of the experiment, followed by plots of the trajectories below. The planned target path is shown on the left, and the actual path as executed by the robot is plotted on the right.

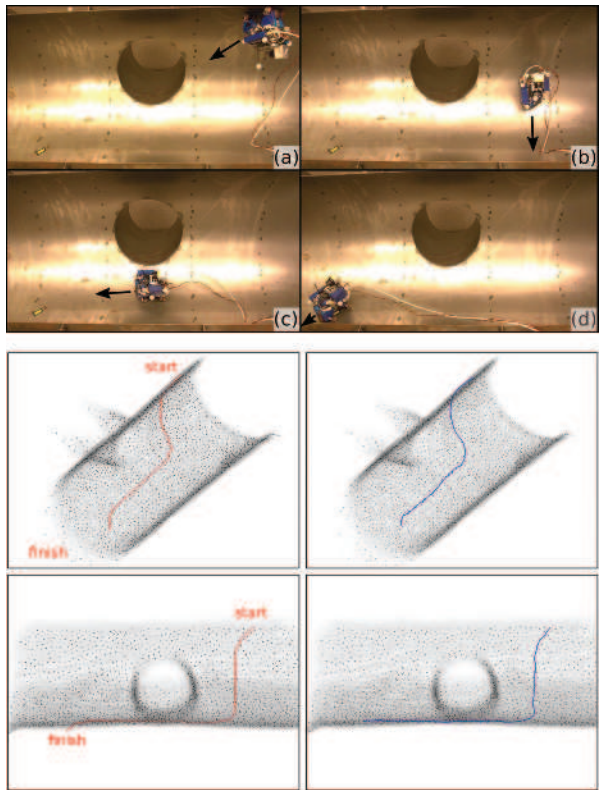


Fig. 33. Experiment 2: Avoiding a hole obstacle (using a Hokuyo UTM-30LX point cloud containing 40,000 points). An image sequence of the experiment, followed by plots of the trajectories below. The planned target path is shown on the left, and the actual path as executed by the robot is plotted on the right.

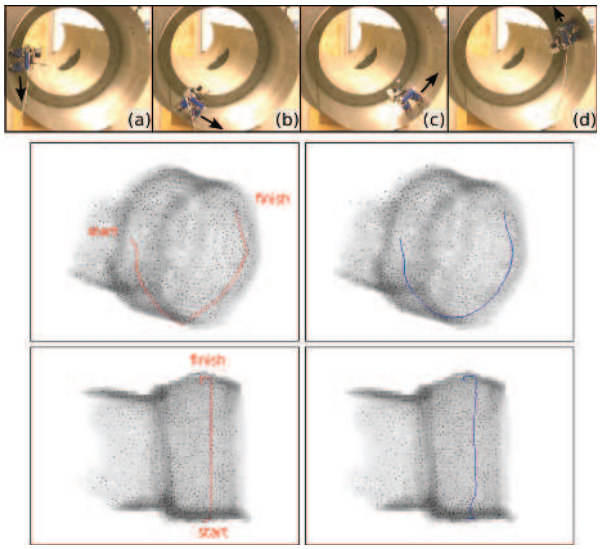


Fig. 32. Experiment 1(b): Tracking a circumferential path (using a Hokuyo URG-04LX point cloud containing 22,000 points). An image sequence of the experiment, followed by plots of the trajectories below. The planned target path is shown on the left, and the actual path as executed by the robot is plotted on the right. Note the strong deformations when compared with the point clouds of Figure 31.

route towards the final demonstration of locomotion, localization, environment modeling and path planning, working all together on the MagneBike robot.

The point cloud acquisition, localization, mapping and planning must be performed in an iterative way. The robot will take a scan, move a short distance, take another scan, and finally use odometry estimates and scan matching techniques, such as ICP, to build a consistent 3D point cloud. Occlusions and limited perception range will not influence the navigation system directly. The system accepts a goal only if it is positioned in the immediate surrounding of the surface. Given a feasible goal as input, the system will compute and return a path to the goal position, as well as the set of corresponding controls.

In static environments such as the steam chest, an efficient computation of paths is much more important than path replanning. Given a static environment, acquiring new scans at the same locations improves the environment models slightly but not substantially, and replanned paths will remain close to previously planned paths. In such an environment, it is sufficient that path computation updates the paths only upon change in the goal points. If parts of the environment - hidden by a junction in the steam chest for example - are not known a priori, the environment needs

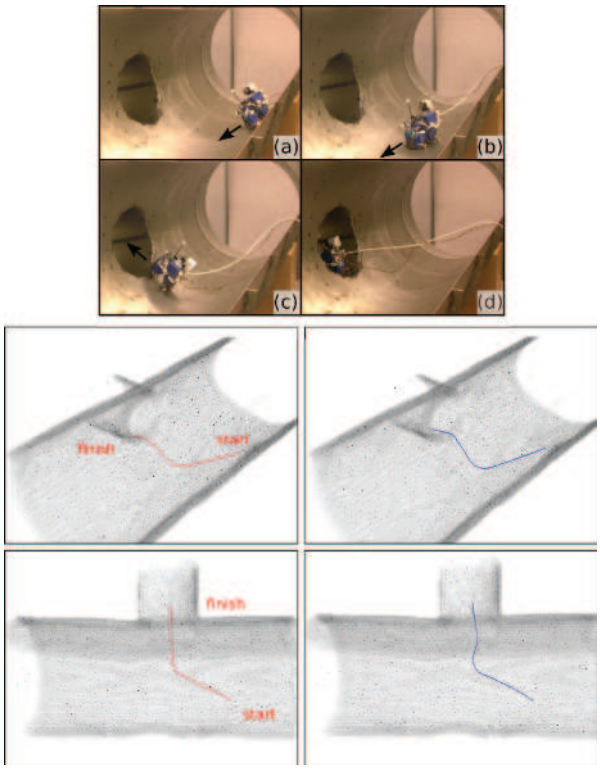


Fig. 34. Experiment 3: Traversing an edge obstacle (using a Hokuyo UTM-30LX point cloud containing 20,000 points). An image sequence of the experiment, followed by plots of the trajectories below. The planned target path is shown on the left, and the actual path as executed by the robot is plotted on the right.

to be explored. An appropriate exploration strategy of a higher level planner will select the new goal points for the navigation system. Alternatively, the goal points can be set manually by a human operator. As the robot cannot plan paths to occluded and thus unknown locations, only a path up to the occlusion point is computed in any case. New laser scans are taken as soon as the occlusion points are reached. Once the map is updated, new goal points and paths to be planned will be part of the newly registered point cloud, and thus paths must be computed completely anew rather than being replanned.

9. Conclusion

This paper addresses the challenging problem of navigating uneven surfaces in 3D space. A navigation solution is proposed, which comprises different subsystems for perception and environment representation, through to path planning and control. First, a model of the environment is obtained from a lidar point cloud using tensor voting. Surfaces and edges are identified, and their orientation is estimated at the sparse points of the point cloud, as well as at all locations in a dense 3D grid. The tensor voting process provides a number of environmental feature maps, which are directly used by a graph-based path planning algorithm. A specialized A* planner represents the cells of the maps as a set of oriented nodes, and establishes graph connections between the nodes iteratively, by taking the full 3D position and orientation of the robot, as well as the structural information from the feature maps into account. The obtained graph structure limits the search space and enables the efficient generation of safe and feasible 6-DoF paths along the curved surfaces. Once a path over the surface is obtained, the equivalent 2D path is found by projecting the movements into local surface planes, and a 2D control strategy is applied to guide the robot along the trajectory using nonlinear feedback laws. The navigation system achieves smooth motions, avoidance of obstacles in the structures, as well as obstacle negotiation by perpendicular edge traversal.

The proposed system has been successfully integrated on the MagneBike robot. Paths planned in a real steam chest environment and tests with MagneBike driving on the inner casing of a steel tube in an inspection task have demonstrated the potential of tensor voting, path planning and control for navigating a robot over complex 3D surfaces in the presence of poorly structured point clouds and different types of obstacles. The last remaining step will be to combine MagneBike's locomotion, localization, environment modeling and path planning to work all together concurrently, in order to take the robot to final field testing in its operational context.

In future work, we furthermore will look at generalizations of our navigation system for the planning on point-based representations, or closer linkage of modeling and planning techniques. The flexibility of the approach should further be investigated through implementation on other platforms, such as rough terrain ground-based robots. Moreover, policies at higher level can make use of the navigation system for robotic exploration and coverage. Another interesting direction is on the improvement of MagneBike's lidar system, and on building several of the MagneBike robots for the collaborative inspection on 3D surfaces.

Funding

This research project was supported by ALSTOM and by the EU FP7 IP projects Natural Human–Robot Cooperation in Dynamic Environments (ICT-247870). F Pomerleau was supported by a fellowship from the Fonds québécois de recherche sur la nature et les technologies (FQRNT).

Acknowledgments

The authors would like to thank Wolfgang Fischer for his help with the design of the test setup, Dario Fenner and Markus Bühler for great technical support and Roland Moser for providing his advice with respect to industrial inspection of power plants.

References

Carsten J, Ferguson D and Stentz A (2006) 3D field D*: Improved path planning and replanning in three dimensions. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3381–3386.

Englot B and Hover F (2011) Planning complex inspection tasks using redundant roadmaps. In *Proceedings of the 15th International Symposium of Robotics Research (ISRR)*.

Ferguson D and Stentz A (2006) Using interpolation to improve path planning: The field D* algorithm. *Journal of Field Robotics* 23: 79–101.

Furgale PT and Barfoot TD (2010) Visual path following on a manifold in unstructured three-dimensional terrain. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 534–539.

Gingras D, Lamarche T, Bedwani J-L and Dupuis E (2010) Rough terrain reconstruction for rover motion planning. In *Proceedings of the Canadian Conference on Computer and Robot Vision (CRV)*, pp. 191–198.

Hart PE, Nilsson NJ and Raphael B (1968) A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics* 4: 100–107.

Hebert M, Caillas C, Krotkov E, Kweon I and Kanade T (1989) Terrain mapping for a roving planetary explorer. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 997–1002.

Howard TM and Kelly A (2007) Optimal rough terrain trajectory generation for wheeled mobile robots. *The International Journal of Robotics Research* 26: 141–166.

Kazhdan M, Bolitho M and Hoppe H (2006) Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry processing*, pp. 61–70.

Kimmel R and Sethian JA (1998) Computing geodesic paths on manifolds. *Proceedings of the National Academy of Sciences of the USA* 95: 8431–8435.

King BJ (2008) *Range Data Analysis by Free-space Modeling and Tensor Voting*. Ph.D. thesis, Rensselaer Polytechnic Institute.

Kneip L, Tâche F, Caprari G and Siegwart R (2009) Characterization of the compact Hokuyo URG-04LX 2D laser range scanner. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1447–1454.

Koenig S and Likhachev M (2002) Improved fast replanning for robot navigation in unknown terrain. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 968–975.

LaValle SM (2006) *Planning Algorithms*. Cambridge: Cambridge University Press.

Marton ZC, Rusu RB and Beetz M (2009) On fast surface reconstruction methods for large and noisy datasets. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3218–3223.

Medioni G, Lee M-S and Tang C-K (2000) *A Computational Framework for Segmentation and Grouping*. Amsterdam: Elsevier Science B.V.

Medioni G and Mordohai P (2004) The tensor voting framework. In Medioni G and Kang SB (eds), *Emerging Topics in Computer Vision*. Englewood Cliffs, NJ: Prentice-Hall, pp. 191–253.

Mordohai P and Medioni G (2005) *Dimensionality Estimation and Manifold Learning using Tensor Voting*. Technical report, Institute for Robotics and Intelligent Systems, University of Southern California.

Nash A, Koenig S and Tovey CA (2010) Lazy theta*: Any-angle path planning and path length analysis in 3D. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Pauly M, Gross M and Kobbelt LP (2002) Efficient simplification of point-sampled surfaces. In *Proceedings of the Conference on Visualization (VIS)*, pp. 163–170.

Pivtoraiko M and Kelly A (2005) Efficient constrained path planning via search in state lattices. In *Proceedings of the 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space*.

Pomerleau F, Breitenmoser A, Liu M, Colas F and Siegwart R (2012) Noise characterization of depth sensors for surface inspections. In *Proceedings of the International Conference on Applied Robotics for Power Industry (CARPI)*.

Quigley M, Conley K, Gerkey BP, et al. (2009) ROS: an open-source robot operating system. In *ICRA Workshop on Open Source Software*.

Samson C and Ait-Abderrahim K (1991) Feedback control of a nonholonomic wheeled cart in cartesian space. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1136–1141.

Schnabel R, Wahl R and Klein R (2007) Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum* 26: 214–226.

Singh S and Kelly A (1996) Robot planning in the space of feasible actions: Two examples. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3309–3316.

Smith M, Posner I and Newman P (2010) Efficient non-parametric surface representations using active sampling for push broom laser data. In *Proceedings of Robotics: Science and Systems*.

Surazhsky V, Surazhsky T, Kirsanov D, Gortler SJ and Hoppe H (2005) Fast exact and approximate geodesics on meshes. In *ACM SIGGRAPH*, pp. 553–560.

Tâche F, Fischer W, Caprari G, Siegwart R, Moser R and Mondada F (2009) Magnebike: A magnetic wheeled robot with high mobility for inspecting complex-shaped structures. *Journal of Field Robotics* 26: 453–476.

Tâche F, Pomerleau F, Caprari G, Siegwart R, Bosse M and Moser R (2011) Three-dimensional localization for the MagneBike inspection robot. *Journal of Field Robotics* 28: 180–203.

Tavakoli M, Faria R, Marques L and de Almeida AT (2011) Autonomous mapping for inspection of 3D structures. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4277–4283.

Thrun S, Martin C, Liu Y, et al. (2004a) A real-time expectation-maximization algorithm for acquiring multiplanar maps of indoor environments with mobile robots. *IEEE Transactions on Robotics and Automation* 20: 433–443.

Thrun S, Thayer S, Whittaker W, et al. (2004b) Autonomous exploration and mapping of abandoned mines. *IEEE Robotics and Automation Magazine* 11(4): 79–91.

Triebel R, Pfaff P and Burgard W (2006) Multi-level surface maps for outdoor terrain mapping and loop closing. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2276–2282.

Vandapel N, Huber DF, Kapuria A and Hebert M (2004) Natural terrain classification using 3-D ladar data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5117–5122.

Wettergreen D, Moreland SJ, Skonieczny K, Jonak D, Kohanbash D and Teza J (2010) Design and field experimentation of a

prototype lunar prospector. *The International Journal of Robotics Research* 29: 1550–1564.

Wong U, Morris A, Lea C, et al. (2011) Comparative evaluation of range sensing technologies for underground void modeling. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3816–3823.

Wurm KM, Hornung A, Bennewitz M, Stachniss C and Burgard W (2010) OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In *Proceedings of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation*.

Appendix: Index to Multimedia Extensions

The multimedia extension page is found at http://www.ijrr.org

Table of Multimedia Extensions

Extension	Type	Description
1	Video	Experiment from Section 8.3.2 of the MagneBike navigating around the circumference of a tube.
2	Video	Experiment from Section 8.3.3 of the MagneBike navigating around a hole-type obstacle.
3	Video	Experiment from Section 8.3.4 of the MagneBike navigating over an edge-type obstacle.

Appendix D

Krebs et al. [2010a]

Ambroise Krebs, Cédric Pradalier, and Roland Siegwart. Adaptive rover behavior based on online empirical evaluation: Rover–terrain interaction and near-to-far learning. *Journal of Field Robotics*, 27(2):158–180, 2010a

Adaptive Rover Behavior Based on Online Empirical Evaluation: Rover–Terrain Interaction and Near-to-Far Learning

Ambroise Krebs, Cédric Pradalier, and Roland Siegwart
Autonomous Systems Laboratory, Swiss Federal Institute of Technology Zürich (ETHZ), 8092 Zürich, Switzerland
e-mail: ambroise.krebs@mavt.ethz.ch, cedric.pradalier@mavt.ethz.ch, roland.siegwart@mavt.ethz.ch

Received 4 January 2009; accepted 2 November 2009

Owing to the fundamental nature of all-terrain exploration, autonomous rovers are confronted with unknown environments. This is especially apparent regarding soil interactions, as the nature of the soil is typically unknown. This work aims at establishing a framework from which the rover can learn from its interaction with the terrains encountered and shows the importance of such a method. We introduce a set of rover–terrain interaction (RTI) and remote data metrics that are expressed in different subspaces. In practice, the information characterizing the terrains, obtained from remote sensors (e.g., a camera) and local sensors (e.g., an inertial measurement unit) is used to characterize the respective remote data and RTI model. In each subspace, which can be described as a feature space encompassing either a remote data measurement or an RTI, similar features are grouped to form classes, and the probability distribution function over the features is learned for each one of those classes. Subsequently, data acquired on the same terrain are used to associate the corresponding models in each subspace and to build an inference model. Based on the remote sensor data measured, the RTI model is predicted using the inference model. This process corresponds to a near-to-far approach and provides the most probable RTI metrics of the terrain lying ahead of the rover. The predicted RTI metrics are then used to plan an optimal path with respect to the RTI model and therefore influence the rover trajectory. The CRAB rover is used in this work for the implementation and testing of the approach, which we call *rover–terrain interactions learned from experiments* (RTILE). This article presents RTILE, describes its implementation, and concludes with results from field tests that validate the approach.

1. INTRODUCTION

The field of mobile robotics has attracted a lot of attention in recent years, due to very famous missions such as planetary exploration with the Mars Exploration Rover (MER), the Mars Science Lab (MSL), and ExoMars and competitions such as the DARPA Grand Challenge and AUVSI. Although robotic platforms have become very popular, they are only a means to perform a more valuable task and not really an achievement themselves. Therefore, and especially in the context of all-terrain robotic platforms, robotic rovers have to be designed as best as possible with respect to their task (*what* has to be done), their context (*where* it has to be done), and their mission constraints (*how* it has to be done). For example, in the context of planetary exploration, the robot is required to provide a safe and reliable locomotion with optimal performance, while consuming as little energy as possible, in order to accomplish scientific analysis in the area of interest.

In the end, the performance of the robot is influenced by two factors:

- First, the performance depends on the robot's physical and mechanical properties, corresponding to its structure, suspension mechanism, actuators, and sensors.
- Second, the performance is related to the control of the robotic platform, in a very generic sense. This includes research in fields such as control, obstacle avoidance, path planning, pose estimation, and so forth.

As mentioned in the latter point, the robot's performance is related to the interaction of the robot with its surroundings and its capability to sense and represent the environment. A natural environment, which is usually the operating place for all-terrain rovers, involves a great diversity in terrain, soil and obstacle types, shapes, and appearances. This diversity is difficult to model and hence implies additional uncertainty that the rover must cope with.

1.1. Objectives

On 26 April 2005 (SOL 446[1]), one of NASA's two Martian rovers, Opportunity, almost got stuck in a sand dune in Meridiani Planum. It took 5 weeks for the scientists to cautiously extract the rover from this delicate situation and allow the mission to continue its course. This example shows the importance of the uncertainties that automatically come with an exploration mission. Although MER rovers were extremely well designed, and even if their interactions with expected terrains were cautiously characterized and tested in many situations, such unexpected events can still occur. In the end, the terrain on which the rover has to operate is, at least partially, unknown and is extremely difficult to characterize beforehand. This effect is even more apparent in the case of applications in rough terrain. Hence we are interested in designing a rover with the capability to learn from its experience while it operates in a mission. Then the accumulated knowledge could be used to improve the rover's behavior. Thus the idea is to make the best out of an operating platform by giving it the capability to link remote and local data. Remote data describe the environment at a distance and are provided by sensors such as cameras, LIDARs, and so forth. Local data express the rover–terrain interaction (RTI) model, which refers to a metric characterizing some aspect of the rover behavior. The RTI model can be related to sensors such as an inertial measurement unit (IMU), actuator encoders, and other proprioceptive sensors. Learning the correspondence between local data (information that is near) and remote data (information that is far) allows anticipating the RTI characteristics ahead of robot position. This information can be used to influence the rover behavior by changing its path. The data association can be referred to as *near to far* and is a concept at the very center of our approach. In fact, this data association allows, in our case, the inference of local data based on remote data, which corresponds to generating a RTI model of the terrain lying ahead of the rover.

1.2. State of the Art

In all-terrain robotics, one of the main tasks of a rover is to successfully move from a point A (starting position) to a point B (goal). The first aspect is to ensure that the rover uses only traversable areas to reach the goal. This criterion is fundamental, but it usually does not limit the possible paths to a single trajectory. The different possible paths can be more or less efficient according to the RTI model. The research of the present article proposes an approach allowing the RTI model, and not the traversability, to be assessed based on remote data. Based on this knowledge, the path with the best RTI can be used. This subsection describes the main research trends related to this work as trying to identify the terrain around the robot is not a new preoccupation in the field. They are presented along three topic lines. We begin by presenting the research using *near-to-far* approaches enabling the traversability prediction. Then we describe the approaches allowing recognizing or identifying the terrain on which the rover is driving, making use of an RTI model. Finally, the approaches enabling the RTI predictions are described. The last part relates to the present work and its specificities with respect to other works described in this state of the art.

Several research projects are trying to assess whether the terrain ahead is traversable before negotiating it. These remote-based methods use different combinations or types of remote sensors—including radars, vision cameras, and laser range finders—e.g., Lalonde, Vandapel, Huber, and Hebert (2006), Manduchi, Castano, Talukder, and Matthies (2005), Poppinga, Birk, and Pathak (2008), and Vandapel, Huber, Kapuria, and Herbert (2004). The assessed traversability is then used to plan the robot's route toward the goal and to ensure its safety. Since 2005, the DARPA-funded LAGR[2] project (Mulligan & Grudic, 2006a, 2006b) has contributed enormously to the development and integration of these approaches. It offered a common robotic platform to develop and test off-road algorithms and compare the results. Trials were regularly organized, offering challenging environment in which the robot had to find its way to reach the goal and learn from its experiments. Thus, the different teams involved in the project, such as Bajracharya, Howard, Matthies, Tang, and Turmon (2009), Huang, Ollis, Happold, and Stancil (2009), Kim, Sun, Min Oh, Rehg, and Bobick (2006), Konolige et al. (2009), Otte, Richardson, Mulligan, and Grudic (2009), Procopio, Mulligan, and Grudic (2009), and Sermanet et al. (2009), showed solutions to enable a robot to learn from its experiment and handle unexpected events. Nevertheless, this research is focused on giving an answer to whether the terrain is traversable. This is expected as it is in line with the goals of the LAGR project, which offers the rover several trials to reach a goal, enabling the rover to learn from its previous experiences. Furthermore, Hadsell et al. (2009) and Happold, Ollis, and Johnson (2006) use an image-based classification to enhance the traversability estimation of the terrain. Thus, from *near to far*, the approach becomes *further to near*, extending the perception horizon of the rover. In Wellington and Stentz (2004), the true ground height in vegetation is computed by making use of a *near-to-far* approach. Again this information allows the autonomous vehicle to define whether the terrain is traversable.

In parallel, a line of research proposes to classify terrains based on the rover–terrain characteristics (Ojeda, Borenstein, Witus, & Karlsen, 2006), such as the vibrations induced in the robot (Iagnemma, Kang, Shibly, &

[1]http://marsrovers.nasa.gov/mission/status_opportunityAll_2005.html#sol446

[2]LAGR stands for Learning Applied to Ground Robots.

Dubowski, 2004). Vibration is mainly measured in terms of the linear acceleration of the wheel bars or the robot body (Brooks & Iagnemma, 2005, 2007; Weiss, Froehlich, & Zell, 2006). See Weiss, Fechner, Stark, and Zell (2007) for a comparison of different methods for classification of three and six types of terrains using a frequency-based representation. In relation to the vibration characteristics, the effect of the robot's velocity on the terrain classification is studied in Dupont, Moore, Collins, and Coyle (2008), and a velocity-independent classification method is proposed and tested on a very limited number of terrains. Similarly, Ward and Iagnemma (2009) propose a speed-independent solution for vehicles with a higher dynamics, classifying terrain using a single suspension-mounted accelerometer. Another famous characteristic used to express the rover–terrain behaviors is slippage. Angelova, Matthies, Helmick, and Petrona (2007) propose to learn the slippage model of the rover with respect to the terrain type and geometrical characteristics. Ishigami, Miwa, Keiji, and Kazuya (2006) also propose a slippage model in relation to the terrain slope. In Weiss and Zell (2008) it is argued that an autonomous system, in addition to learning from training data, should be able to detect and classify new terrains. The authors propose a Gaussian mixture model for detection and classification of novel terrains. Halatci, Brooks, and Iagnemma (2008) propose a classification method making use of both visual and vibration data, comparing several classification methods. It shows great results in classifying terrains encountered, but fundamentally the process is a supervised one. Apart from Angelova et al. (2007) and Brooks and Iagnemma (2007), there is no *near-to-far* approach used to anticipate the RTI. In Angelova et al. (2007), the terrain appearance is divided into well-known classes providing a prior set of trained classifiers. The number and type of the terrain is then fixed, and the slippage model is then learned for each one of those. The end-to-end approach including this work is presented in Helmick, Angelova, and Matthies (2009). In the case of Brooks and Iagnemma (2007), it is the other way around as the visual characteristics of the terrain are learned online while the rover–terrain characteristic classifiers were trained beforehand. One way or another, those approaches rely on trained classifiers, which implies a given and fixed number of classes.

1.3. Contributions

The research of the present article proposes an approach allowing the RTI model, and not the traversability (as in the LAGR-related projects), to be assessed based on remote data. Thus, based on a *near-to-far* approach, the RTIs can be estimated and abstracted into classes. We also argue that it seems more appropriate for a robot to classify terrains based on its needs and according to how they affect its behavior and not necessarily based on human-defined classes. In other words, the presented approach categorizes the terrain in a way that is suitable for its path planning.

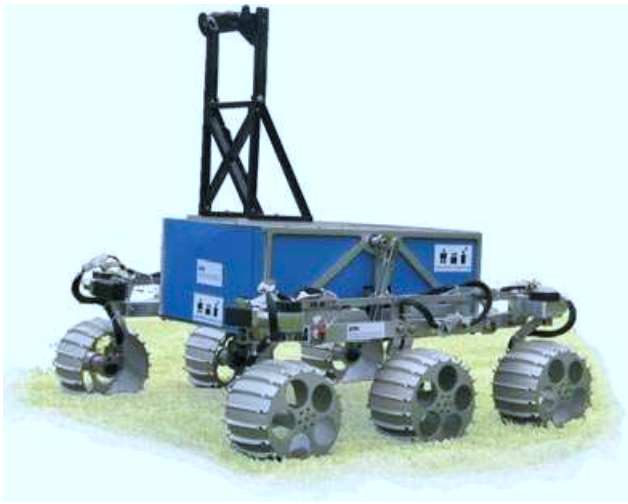


Figure 1. CRAB rover used in the context of this research.

Finally, another essential point is that this classification is not fixed to a rigid number of classes but rather results from the rover's experience. The terrain representation is actually learned and can evolve online with respect to the situation encountered by the rover. Therefore, the approach is very flexible and is capable of incorporating new types of terrain or RTI models while traversing the terrain. Also note that the present work does not take the traversability into account because the focus lies instead on the RTI model learning and prediction. The overall approach is named *RTILE*, which stands for rover–terrain interactions learned from experiments.

1.4. Content

The objective of our work is to implement an online terrain classification and prediction algorithm. Its goal is to modify the rover's behavior according to a given metric. This metric could be, for example, the slippage minimization in the context of the MER rover. The next section focuses on giving an overview of the approach, including a theoretical description of all the components needed to achieve it. Then, the implementation of this approach in the specific context of the CRAB rover platform is described in Section 3. The CRAB is a rover with six motorized wheels (Figure 1). We present the results of three described experiments in Section 4. The first one shows the learning capabilities of the approach, and the other two expose the overall effects of the approach, namely, the rover's behavior being influenced by its experience. The last part of the article is dedicated to the conclusion and future work envisioned.

2. APPROACH OVERVIEW

This section describes our approach to tackle the RTI problem. The focus is placed on the theoretical and generic

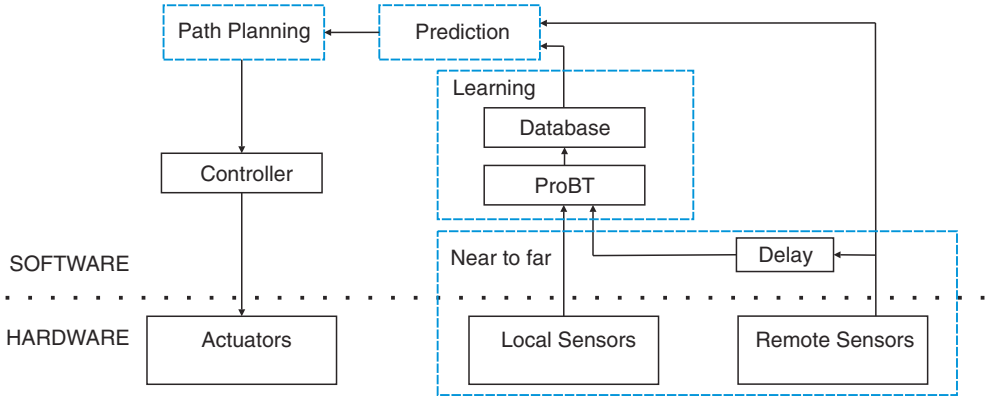


Figure 2. Schematic summarizing the RTILE approach.

concepts of RTILE. Figure 2 shows a simplified description of the approach and highlights its most important elements. First the *near-to-far* part is realized using a grid-based approach, allowing for local and remote data associations. A Bayesian model (Bessière, Laugier, & Siegwart, 2008) is used to handle the uncertainties, and its various probability distributions are assessed in the *learning* part. The *prediction* makes use of the knowledge acquired to estimate the RTI ahead of the rover. Finally the *path planning* uses the predicted RTI to influence the rover trajectory.

In this section, we go further into the details of the different aspects summarized above. To this end, the association of characteristics representing the terrain ahead of the rover and characteristics describing the rover's behavior, named *near to far*, are described. Then the *probabilistic model* is presented, leading to the *inference* subsection presenting how the prediction of the local features F_l is performed. The next two subsections, *Class Association* and *Class Definition and Novelty Detection*, focus on the learning aspects of the method. The subsection *subspaces* describes the local and remote data representations. The last subsection refers to the *path planning* of the rover trajectory. First, we describe the meaning of *rover behavior*.

2.1. Rover Behavior

The rover behavior refers to its manner of behaving or acting within its environment. It can be influenced in many different ways as it is dependent on various levels of control. In many research projects, the elements acted upon are the control type or the control parameters; however, in this study, we are interested in the path used by the rover. Accordingly, the traversability, in other words where the robot is able to drive or not, is essential to the robot. The focus is on a complementary aspect, namely to differentiate the areas traversable with respect to their trafficability—ability of a rover to traverse different soil types—from their terrainability—ability to negotiate terrain irregularities (Apostolopoulos, 2001). Specific metrics are used here to learn and characterize the rover's performance. For example, in the case of the rover Opportunity, as well as other rough-terrain robots, a possibility would be to use a metric referring to the slippage. However, our approach is not limited to slippage. The metrics used in the context of this research are presented in the next section.

An example is the robot Roomba,[3] a very famous domestic robot aimed at cleaning home floors. The robot behaves the same regardless of the type of floor (parquet, carpet, etc.) of the diverse areas it has to clean. It would be nice to make this robot infer the initially unknown, diverse types of floors from the noise the rover makes while cleaning them. Thus the Roomba could optimize its cleaning pattern according to a metric driving the rover behavior, which in this case could be the amount of noise generated. For example, the robot could prefer to clean the area where it makes less noise at night.

2.2. Near to Far

The capability of a rover to learn from its environment is linked to its ability to acquire data from the environment, which is directly linked to the rover sensors. Two different types of sensors are needed here, namely local and remote. First, we require the ability to characterize aspects of the RTIs with the local sensors. Then, in order to influence the rover's behavior, i.e., its path, we need to obtain information on the terrain ahead of the vehicle with the remote sensors. The remote sensors observe the terrain ahead of the vehicle and extract some terrain characterizations. Later, when driving over the same terrain patch, the behavioral sensor records another terrain characterization. It is important to link all these data, to generate a bridge from near to far. The remote characteristics (or features) are expressed as F_r, and F_l corresponds to the local ones. The goal then is to

[3]http://www.irobot.com

form samples s_i associating the corresponding remote and local features, samples that can be then processed and used for learning:

$$s_i = (F_l, F_r). \tag{1}$$

It is important to note that in Eq. (1) the remote and local features correspond to the same spatial area in order to have consistent s_i. Thus, F_l and F_r are not acquired at the same time, and the use of F_r is delayed to form the samples.

2.3. Probabilistic Model

Based on the samples reflecting the rover experience, a predictive model can be built. This model is used to associate predictive terrain observations and behavioral terrain characterizations (roughness, slippage, softness, etc.). Therefore, the joint probability distribution and its decomposition include F_r and F_l:

$$P(F_r, F_l) = P(F_r)P(F_l \mid F_r).$$

Such a model, although theoretically correct, leads to complexity and dimensionality issues. In fact, the computed featured are continuous objects in multiple dimensions, and the corresponding distributions are difficult to implement and compute. For this reason similar features are regrouped into classes, providing an abstraction layer. Thus K_r and K_l, which express the terrain type (or class) in both the remote and local feature space, are added to the model:

$$P(F_r, F_l, K_r, K_l).$$

The decomposed joint distribution is expressed as follows:

$$P(F_r, F_l, K_r, K_l) = P(K_r)P(K_l \mid K_r)P(F_l \mid K_l)P(F_r \mid K_r). \tag{2}$$

The probability distribution of the features is assumed to be dependent only on its corresponding class. This means that the features expressing different RTIs, F_l and remote terrain information F_r, are conditionally independent, given K_l and K_r. This is an important assumption that drives the RTI model and the corresponding features design. In the end, the idea is to use a more tractable expression to the following equation:

$$P(F_l \mid F_r), \tag{3}$$

which corresponds to the following question: What are the predicted F_l, based on the observed F_r?

2.4. Inference

Now that the probabilistic model has been expressed and its joint distribution described, the Bayes' rule can be used to develop Eq. (3):

$$\begin{aligned} P(F_l \mid F_r) &= \frac{\sum_{K_r}\sum_{K_l} P(K_r)P(K_l \mid K_r)P(F_l \mid K_l)P(F_r \mid K_r)}{P(F_r)} \\ &= \sum_{K_r}\left\{\left[\frac{P(K_r)P(F_r \mid Kr)}{P(F_r)}\right] \right. \\ &\quad \left. \times \sum_{K_l} P(F_l \mid K_l)P(K_l \mid K_r)\right\}. \end{aligned} \tag{4}$$

To simplify this expression, we can remark that the first part of this equation corresponds to the development of $P(K_r \mid F_r)$ using Bayes' rule:

$$P(K_r \mid F_r) = \frac{P(K_r)P(F_r \mid Kr)}{P(F_r)}. \tag{5}$$

Also, using the law of total probability, we can identify the second part of Eq. (4) as $P(F_l \mid K_r)$, the distribution over the local features knowing the remote class:

$$\begin{aligned} P(F_l \mid K_r) &= \sum_{K_l} P(F_l \mid K_l K_r)P(K_l \mid K_r) \\ &= \sum_{K_l} P(F_l \mid K_l)P(K_l \mid K_r). \end{aligned} \tag{6}$$

Note that this uses the assumption that the local features are conditionally independent of the remote class knowing the local class: $P(F_l \mid K_l K_r) = P(F_l \mid K_l)$.

Using Eqs. (5) and (6), Eq. (4) can be rewritten as

$$\begin{aligned} P(F_l \mid F_r) &= \sum_{K_r}\left[P(K_r \mid F_r)\sum_{K_l} P(F_l \mid K_l)P(K_l \mid K_r)\right] \\ &= \sum_{K_r} P(K_r \mid F_r)P(F_l \mid K_r). \end{aligned} \tag{7}$$

To simplify the double summation over K_r and K_l, we will consider only the most likely remote class based on the remote features. Let us define the most likely remote class $\tilde{k}_r$ as

$$\tilde{k}_r = \underset{K_r}{\operatorname{argmax}}[P(K_r \mid F_r)]. \tag{8}$$

Only considering $\tilde{k}_r$ is translated into the following approximation:

$$P(K_r = \tilde{k}_r \mid F_r) \cong 1 \quad \text{and} \quad P(K_r \neq \tilde{k}_r \mid F_r) \cong 0, \tag{9}$$

which is a good approximation, as long as the classes are well separated in the features space, meaning that their probability distributions have to be peaked.

Using this approximation, Eq. (7) becomes

$$P(F_l \mid F_r) \cong P(K_r = \tilde{k}_r \mid F_r) \sum_{K_l} P(F_l \mid K_l) P(K_l \mid K_r = \tilde{k}_r)$$
$$\cong \sum_{K_l} P(F_l \mid K_l) P(K_l \mid K_r = \tilde{k}_r) \quad (10)$$
$$\cong P(F_l \mid K_r = \tilde{k}_r).$$

Therefore Eqs. (7) and (10) allow us to determine the most probable predicted local features F_l based on the remote features F_r.

Note that summing all the probabilities in the computation of $P(F_l \mid K_r)$ is the mathematically exact way to proceed, but approximations could be considered, resulting in a pessimistic or an optimistic result. For example, a pessimistic approach would consider only the local feature F_l resulting in the worst robot behavior, among the ones that can be predicted from the remote feature class K_r. Of course, the meaning of worst has to be defined in the RTI context and can be assessed by the impact of the F_l on the global rover performance. Such an approach is an interesting way of preserving the hardware. In any case, asserting that the F_l is predicted from the F_r corresponds to predicting the RTI. The theoretical description of the approach is expressed involving several probability distribution functions, which are learned based on the rover experience. The learning process is the subject of the two next subsections.

2.5. Class Association

In this subsection, we expose the connection between the probability distribution of the local classes and the remote ones, which corresponds to expressing the link between the classes on the local and remote space, or $P(K_l \mid K_r)$ from Eq. (2). This information is acquired at the same time as the samples s_i are formed, because it contains the correspondence between the data in each subspace, as shown in Figure 3.

The samples s_i provide knowledge about how probable the connections between the various subspaces are. Thus assuming that A and B in Figure 3 are two subspaces

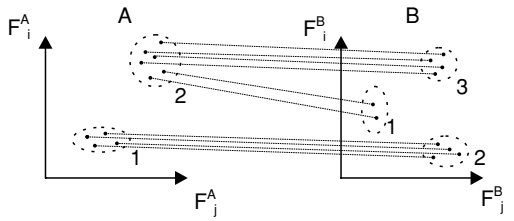


Figure 3. Connections between the classes of different subspaces allowing inference.

with, respectively, two and three classes,

$$P(K_B = 3 \mid K_A = 2) = \frac{N_{K_B=3K_A=2}}{N_{K_A=2}} = \frac{P(AB)}{P(A)},$$

with $N_{K_B=3K_A=2}$ being the number of connections between class 3 of B and class 2 of A and $N_{K_A=2}$ representing the number of samples collected as part of class 2 of A. Now assuming that A is a remote subspace and B is a local subspace, the class association can be performed and it is learned based on the data obtained by the rover.

2.6. Class Definition and Novelty Detection

Besides learning the class association, the ability to learn the classes themselves within the features space is fundamental. In other words, we need a method to decide whether new data are part of an existing class or whether they are eligible for creating a new one. An important aspect of the learning is to let the robot handle the data related to its experiments in the most suitable way. This means that the classes do not necessarily correspond to human defined criteria. In this sense, the approach relates closely to unsupervised learning.

The probability distributions of the features for a given class are defined as a Gaussian distribution. The inference and the expression of the prediction are described in the following part.

Assume that we have a feature space consisting of C classes, whose distributions are Gaussian, to which another distribution is added. This additional distribution refers to the unknown class, or class 0, and is based on a uniform distribution. Therefore, the probability distributions are the following:

$$P(F \mid K = k) = \begin{cases} G_{\mu_k,\sigma_k} & \text{if } 1 \le k \le C \\ U & \text{otherwise } (k = 0) \end{cases}, \quad (11)$$

with F and K being variables associated, respectively, with the features and the class number. The novelty detection aims to know the class corresponding to a given feature. If the class is not known yet, a new class can be created. Therefore, the question is the following:

$$P(K \mid F) \propto P(K) P(F \mid K).$$

And using the maximum likelihood, it can be expressed as

$$\tilde{k} = \underset{j \in [1,n]}{\text{argmax}} [P(F \mid K = j)].$$

Knowing that, the novelty detection concerns data whose classification results in $K = 0$. For example, considering a subspace of one dimension with j classes learned, new data $F = f$ are classified as follows:

$$K = \begin{cases} j & \text{if } P(K = \tilde{k} \mid F = f) > U \\ 0 & \text{otherwise} \end{cases}. \quad (12)$$

In case that the classification results in $K = 0$, the new data are considered as not being part of the existing class and instead are considered to create a new one. Most importantly, this decision is taken automatically, naturally when the new data do not fit into the current description of the classes in the subspace.

The data are processed in batches, at a specific moment (e.g., when the rover has traveled a given distance or reached a waypoint). At this point, the new pieces of data available are handled as follows:

- If they can be found as part of an existing class, they will then reinforce the knowledge of this class.
- If they can be found as part of class 0, or the unknown class, then the new data are part of no existing class in the *current* representation. Thus we have the following possibilities:
  - The new data f can be used, together with N other similar ones, to create a new class $K = C + 1$:

$$P(F \mid K = C + 1) = G_{\mu_{C+1},\sigma_{C+1}},$$

with

$$\mu_{C+1} = \frac{1}{N}\sum_{i=1}^{N} f_i \quad \text{and} \quad \sigma^2_{C+1} = \frac{1}{N}\sum_{i=1}^{N}(f_i - \mu_{C+1})^2.$$

  - If perceived as outliers, the data can be either stored for later use (when enough similar data are available) or simply discarded (in case of a transition between terrains, for example). New data can be perceived as outliers if the total number of samples of the unknown class is not big enough.

2.7. Subspaces

The feature space representation, in which the learning is performed, is divided into a set of subspaces. Each one of these is a feature space representing either remote data or an RTI (i.e., local data). This implementation is performed for the following reasons.

First, using all the inputs of the robot's sensors to characterize its interaction with the terrain into a single space leads to implementation issues. The number of dimensions of the feature space would be enormous, and it would be ineffective because they would most likely be sparse. Furthermore, the implementation would be difficult due to computational costs. Thus, in order to have useful results and distributions with a reasonable number of dimensions, a framework has to be defined.

Second, we are more interested in the meaning of the rover's interaction with the terrain than in the rover's sensor data themselves. Therefore, the feature space is subdivided into subspaces that correspond to given characteristics of the RTI. The features of the subspaces themselves correspond directly to the characteristics, and they are learned with the assumption that they are distributed according to a (multi)normal law. For each subspace, the metric that corresponds to a good or bad RTI is also known.

Finally, another advantage of subdividing the space is the flexibility it offers. The different RTI models can be treated one after the other and are independent. Thus adding a new type of RTI model, i.e., as a result of a new sensor integrated to the platform, does not affect what is already learned or the other subspaces. This is particularly useful in the context of this research.

We propose to define the subspaces according to the terrain characteristics (e.g., softness) and their effects on the robot. It can be interesting to have only a single type of sensor per subspace even though several sensors provide data regarding the same RTI model. In this case, several subspaces addressing the same RTI model can be used. Such a methodology improves the approach's reliability. Thus, if a sensor is damaged in operation, or if a sensor is modified or changed, part of the terrain representation can still be used.

The features of the subspaces modeling the RTI have to be designed with care. They are the elements that will capture the terrain representation and the class definitions within the subspaces.

For example, in the context of the Roomba robot, the RTI type could be defined as the noise generated by the robot cleaning the soil. The feature that could be used in this case is a root-mean-square (RMS) value of the noise signal. In this case a low RMS value would correspond to a better RTI, if a quiet behavior of the robot is preferred. To summarize, the subspaces have to be defined with respect to the robotic platform (the sensors available) and the application to which the robot is dedicated.

2.8. Path Planning

A path planner is used to drive the rover and reach the goal, using the terrains with the best predicted RTI model possible. This path planner is E*[4] (Philippsen, Jensen, & Siegwart, 2006). It is briefly described here in a first part, and then its use within RTILE is explained.

The E* algorithm is a grid-based path planner that is based on a weighted-region approach. The environment in which the rover is evolving is represented as a grid where the robot position and its goal position are known. A navigation function is computed for each cell, stating the path's cost to reach the goal from this cell. The underlying technique is expressed within the continuous domain, which corresponds to a wavefront propagating from the goal toward the rover. The path to reach the goal can be found by using a gradient descent over the navigation function.

[4] E* is very similar to the field D* (Ferguson & Stentz, 2005) regarding the functionalities. It differs mostly in terms of implementation.

The grid used by E*, named G_e, is formed of nodes, or cells c_e. Several interesting properties are linked with c_e:

- $r(c_e)$ is the difficulty or cost of traversing a given cell. This parameter corresponds intuitively to the wavefront speed of the navigation function. For example, a cell corresponding to an obstacle would block the wavefront.
- $v(c_e)$ represents the "height" of the navigation function of the cell. It can be computed based on the $v(c_e^n)$ of the neighbor cell closest to the goal, c_e^n, and based on the wavefront propagation cost $r(c_e)$.

In a typical application, the rover and goal positions are given to E*. The navigation function is then computed for each c_e, based on the $r(c_e)$. The cells corresponding to obstacles block the navigation function propagation, and the cells in the neighborhood of those have their propagation cost increased. This pushes the rover away from the obstacles without blocking the navigation function propagation. The trajectory reaching the goal can be then computed by using the gradient descent on the navigation function from the rover position to the goal position.

In our work the propagation cost is used in a different way. It is computed based on traversability, as well as on the predicted RTI. Thus, we have

$$r(c_e) = f(\Lambda, T). \tag{13}$$

The metric Λ depends on the evaluation of the RTI, called M_{RTI}, computed based on the local features. T depends mainly on the geometry of the terrain and can be considered as a Boolean value corresponding to the traversability. Even though other research considers the traversability as a continuous metric, a decision has still to be taken allowing the rover to traverse or not an area. T is the result of this decision:

$$\Lambda = h(M_{\mathrm{RTI}}) \text{ with } \Lambda \in [0;1], \tag{14}$$

$$T = \begin{cases} 1 & \text{if the cell is traversable} \\ 0 & \text{otherwise} \end{cases}. \tag{15}$$

Λ takes a high value for a terrain having a good rover interaction. Hence, a value of 1 for Λ means a perfect behavior of the rover according to the metric defined, whereas a value of 0 corresponds to a terrain to be avoided. For example, assume that the rover faces two terrains (named white and gray), as depicted in Figure 4. The rover, whose position is marked by the cross, has to reach the goal on the right-hand side. If the rover is able to identify the terrains according to

$$\Lambda_{\mathrm{white}} > \Lambda_{\mathrm{gray}},$$

then the resulting generated *trace* (dashed) provided by E* naturally avoids the gray terrain and is a result of a gradient descent performed on the navigation function (illustrated by the wavefront). In summary, E* offers a trade-off between the movement cost and the path length and it provides a trace to be followed to reach the goal.

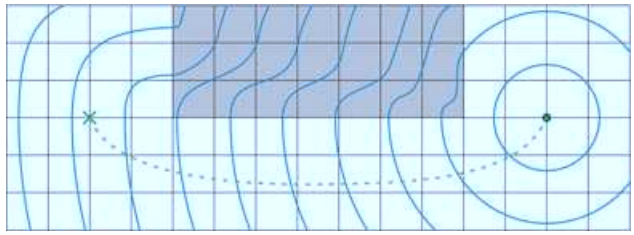


Figure 4. Wavefront propagation with E* (Philippsen, 2006), from the goal (marked with a circle) to the robot (marked with a cross). The *trace* proposed by E* using the gradient descent is also depicted in dashes.

3. IMPLEMENTATION

In this section, the RTILE approach is described in more detail, as well as the challenges and solutions implemented in the context of the CRAB rover platform. The most important assumptions are first described, and the rover is then presented. The learning process and its use to predict the RTI follow. Finally, the pseudo code of the approach and a summary conclude this section.

3.1. Assumptions

The most important assumptions driving the implementation of the approach on the CRAB rover, and therefore influencing the interpretation of the tests results, are summarized as follows:

- Although numerous works identify the traversability (T) of the rover surroundings, the goal here focuses on the RTI. The consequence is that the traversability is not taken into account, or rather the terrain surrounding the rover is assumed to be traversable.
- The terrain surrounding the rover is assumed to be globally flat. As we are interested in showing the adaptive behavior of the robot, the experiments conducted use flat grounds with varied Λ characteristics.
- Knowledge acquired during past tests can be reused, but this is not necessary. It is assumed that the approach is not dependent on any prior knowledge regarding the RTI or the remote data model. Thus, a predefined set of classes, such as grass, gravel, and so on, is not required.
- The adaptive behavior is driven by Λ [Eq. (14)], which is assumed to be known and provided by a user. In other words, the meaning of what a "good" and "bad" terrain is must be provided.

3.2. Hardware Platform

The platform used in the context of this project is the CRAB rover, depicted in Figures 1 and 5. The CRAB rover has six motorized wheels with a passive suspension mechanism. It is composed of a double–parallel bogie mechanism on each side, connected via a differential. The first parallel bogie is

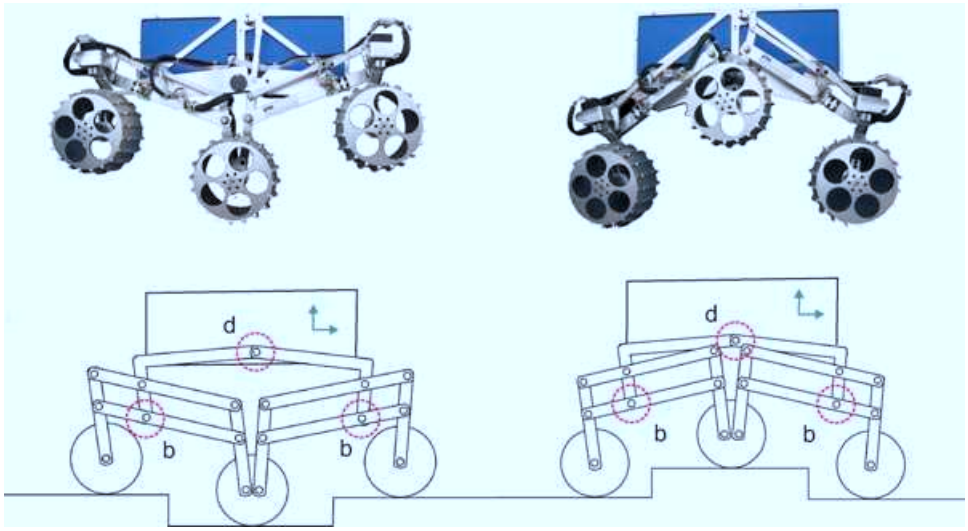


Figure 5. CRAB suspension mechanism movement. The position of the angular sensors is depicted with the dashed circles. The four MR_b (two on each side) are indicated with a b, and the two MR_d (one on each side) are marked with a d.

between the front and the middle wheels, and the second links the middle and the back wheels. The loop is closed via a rocker bogie that is connected to the chassis. As this link is made via a single pivot joint, a differential is necessary to control the chassis attitude. The kinematics of the passive suspension mechanism can be observed in Figure 5. The pivot joints on the parallel bogie are positioned so that the rover has an equal repartition of its mass on the wheels on a level ground. The six wheels are motorized with dc Maxon motors, as are the four steering units, which are linked to the four corner wheels. The control of the robot can be performed according to the kinematic constraints by implementing a double Ackermann steering, one for the front wheels and the other for the back wheels. Table I gives an overview of the most important CRAB dimensions.

The following sensors are available on the platform:

- An IMU, placed at the chassis level (represented by the arrows in Figure 5). It is an MT9-B from Xsens that provides Euler angles.

Table I. Dimensions of the CRAB rover.

Parameter	Value	Unit
Width	0.773	m
Distance front to middle	0.324	m
Distance middle to back	0.324	m
Ground clearance	0.2	m
Wheel diameter	0.196	m
Wheel width	0.1	m
Weight	37.25	kg
Speed autonomous	0.06	$m \cdot s^{-1}$

- Four angular sensors positioned on the parallel bogies. The sensors, MR_b, are homemade and based on a magnetoresistive technology to measure the pivot joint angle at positions shown in Figure 5 with b.
- Two angular sensors, MR_d, placed on the differential are of the same sensors as above. Their positions are shown in Figure 5 with d.
- An HD webcam from Logitech (Quickcam Pro 9000) provides two megapixel images of what lies ahead of the robot within a 53-deg field of view and is the only sensor that provides predictive data.

3.3. Rover Control

The algorithm described below is used on the CRAB to follow the *trace*, or planned path, provided by E*. In a first step, the rover is considered as a differential-drive rover (referred to as a differential rover). Then the computed commands are applied to the CRAB rover, taking its specific kinematic constraints into account.

The control for a differential rover can be achieved according to Siegwart and Nourbakhsh (2004). The translational (v_{trans}) and the rotational (v_{rot}) velocities are computed as follows:

$$v_{\text{trans}} = \begin{cases} k_\rho \rho & \text{if } \rho < \delta_1 \\ v_{\max} & \text{otherwise} \end{cases}, \tag{16}$$

$$v_{\text{rot}} = k_\alpha \alpha, \tag{17}$$

with ρ the distance to the goal and α the angle between the rover's orientation and the direct trajectory to the next trace

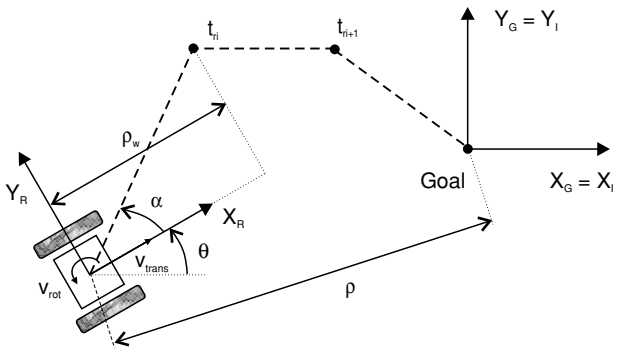


Figure 6. Path following for a differential-drive robot.

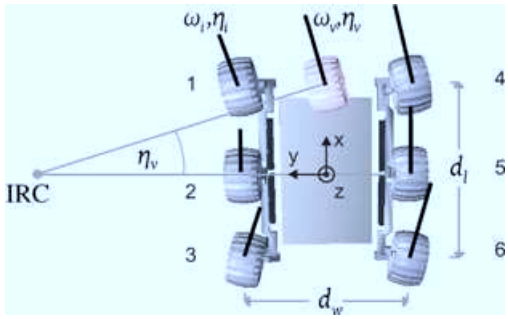


Figure 7. CRAB velocity control based on the virtual wheel. The IRC position is defined as the intersection of the middle wheels' axis and the virtual wheel axis.

waypoint t_r. These parameters are illustrated in Figure 6:

$$t_r = \begin{cases} t_{ri} & \text{if } \rho_w > \delta_2 \\ t_{ri+1} & \text{otherwise} \end{cases}. \tag{18}$$

t_r is defined as the next waypoint, t_{ri}, except if it is closer than a threshold, δ_2. In this case, the next waypoint is used. Thus, the rover reaches its goal following the trace provided by E*. The method described above is a standard approach that enables a differential robot to follow a *trace*. Table II shows the values of the control parameters used to drive the CRAB rover during the tests.

To retrieve the commands for each one of the 10 motors (six driving wheels and four steering units), v_{trans} and v_{rot} have to be transformed into commands for the CRAB's motors. The rover is basically controlled via a virtual wheel placed at the front of the rover, as depicted in Figure 7. The virtual wheel's steering angle η_v and velocity ω_v are computed as follows:

$$\eta_v = \arctan\left(\frac{v_{\text{rot}} * d_l/2}{v_{\text{trans}}}\right), \tag{19}$$

$$\omega_v = \frac{v_{\text{trans}}}{\cos(\eta_v)}. \tag{20}$$

An instantaneous rotation center (IRC) is defined by intersecting the virtual wheel axis and the middle wheel's axis. The position of the IRC and the relative distances to all the wheels is used to compute the required commands for all the motors. The corresponding set points, η_i and ω_i, for the six wheels are derived from η_v and ω_v using the geometry of the rover.

Table II. Control parameters for the CRAB rover.

Parameter	Value	Unit
k_α	0.2	s^{-1}
k_ρ	0.097	s^{-1}
δ_1	0.285	m
δ_2	0.507	m

3.4. Learning

This section describes the implementation of the required elements for the learning task. An illustration of this process is given in Figure 8. The subspaces used in this work are the following. The first two represent the RTI types, the so-called *softness* and *bumpiness*, which are local subspaces. The last subspace represents remote data and is called *appearance*, as linked with camera images. The creation of samples s_i associating local and remote features is based on a two-dimensional (2D) grid–based decomposition of the environment. The grid considered here is G_l, formed of cells c_l, which are used for the learning part.

For the remote features, the acquisition process is discrete and features are computed each time an image is taken. Thus the grid G_l is projected onto the image obtained, and the areas (called patches) defined by the visible cells c_l are processed. The corresponding data, characterizing the remote aspect of the patches, are saved within c_l.

The local feature processing differs because it depends on continuous sources, and their segmentation is driven by the wheel positions. Thus, when a wheel moves into a cell the data of the local source are recorded until the wheel moves out, after which the local features are computed.

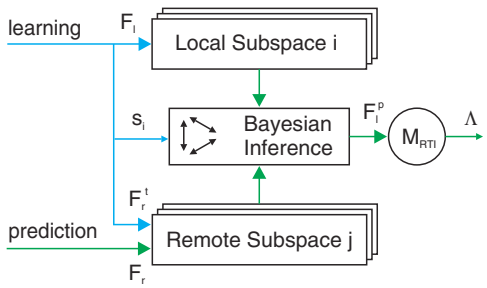


Figure 8. Schematic representing the learning and prediction mechanism in the representation chosen for this work (using subspaces). The metric Λ is the function that drives how the path is planned and can be related to the rover's behavior as expressed in Section 2.1.

Note that the wheels spend most of their time in different cells and each wheel is treated independently to generate the features. As an example, let us imagine the rover moving straight and parallel to one of the grid axes, the traversed cell (cl) end up with three local features in each subspace. The three features are acquired sequentially as the three wheels enter and leave the cell.

When the last wheel leaves the cell behind the rover, and the cell contains both local and remote features, a sample is created for use in the learning process. Note that ProBT,[5] a programming tool facilitating the creation of Bayesian models and their reusability, is used to create and learn the needed probability distributions. We now continue with a description of the subspaces used and their corresponding features.

3.4.1. Local Feature: Softness

The local softness subspace refers to whether the terrain has a soft soil (e.g., sand) or a hard one (such as asphalt), which can be measured by the shocks occurring to the rover structure. Owing to the particular suspension system of the CRAB rover and its metallic wheels, every shock resulting from the wheel–terrain interaction is directly transmitted to the chassis. Thus the shocks can be measured on the chassis with the IMU. As this specific IMU measures the Euler angles, ϕ_j with j referring to the three axes, one needs to process its data to retrieve the acceleration.[6] As we are interested in the magnitude of the shocks, the absolute value of the angular acceleration along the x and y rover axes are used as features:

$$F_l^{\text{softness}} = (\overline{\|\alpha_x\|}, \overline{\|\alpha_y\|}), \tag{21}$$

where α_j, $j = x, y$ is the angular acceleration of the robot. Note also that as $\|\alpha_j\|$ is a vector, the mean value is simply computed over the sequence corresponding to the grid cell treated. The angular acceleration along the z axis is not used as it is coupled with changes in the heading of the rover.

This RTI design is inspired by other research work, such as that of Brooks and Iagnemma (2005) and Weiss et al. (2006). These researchers have used this metric to classify the terrains the rover is traversing according to predefined classes. In these cases more complex features are used, such as the power spectral density (PSD) of the signal.

Finally, it must be noted that the data generated by the IMU are from a single source and express an RTI potentially influenced by all six wheels. To create the samples s_i, the IMU data must be related to the grid cells corresponding to the positions of all six wheels.

[5]http://www.probayes.com

[6]The IMU Xsense MT9-B can provide either the Euler angles, the quaternions, or the rotation matrix.

3.4.2. Local Feature: Bumpiness

The bumpiness subspace, which is also local, refers to the geometric characteristic of the terrain, whether it is flat (e.g., grass, at least on a soccer field) or bumpy (e.g., sand). This characteristic can be measured by the movement of the CRAB's suspension system, whose state is sensed by the angular sensors. Thus the MR_b angles (τ) are used to obtain the data of the suspension system movement. The average of the absolute value of the angular variation is proportional to the amount of displacement of the suspension system, which is defined as the bumpiness:

$$F_l^{\text{bumpiness}} = (\overline{\|\tau - \mu_x\|}). \tag{22}$$

As each suspension mechanism has two redundant MR_b sensors, a single data source can be computed as follows for the left—and similarly for the right—suspension system:

$$MR_b^{\text{left}} = \frac{MR_{\text{front}}^{\text{left}} - MR_{\text{back}}^{\text{left}}}{2}. \tag{23}$$

Because in this case the sensors are quite noisy, a low-pass filter is also used to smooth the data:

$$\tau_i = a \cdot MR_b^{\text{left}} + (1 - a)\tau_{i-1}. \tag{24}$$

A value of $a = 0.3$ was empirically determined to solve the sensors' noise issue. Note that the data of the MR_b^{left} and MR_b^{right} are indifferently mixed in the subspace as the structure is perfectly symmetrical. This leads to similar interpretations of the data if the metric is computed on the MR_b^{left} or MR_b^{right} signal.

In fact the bumpiness is fed by two continuous sources (MR_b^{left} and MR_b^{right}). This feature is specific to the CRAB rover. Owing to its highly compliant suspension mechanism, which adapts naturally to the terrain's shape, the suspension movement directly reflects the roughness of the terrain.

3.4.3. Remote Feature: Appearance

Although described last, the appearance subspace is not the least because it is the only remote one and hence fundamental and necessary to predict the RTIs. The idea here is to be able to visually identify the different terrains using a color-based criterion. The hue-saturation-value (HSV) color space is usually preferred because it is robust to illumination changes. Thus Brooks and Iagnemma (2007) used, among other elements, a color-based terrain descriptor consisting of four values: S, V, and the sine and cosine of H (to avoid any discontinuity problem). Such features were implemented and tested but, in the context of our approach, they lead to an unreliable representation of the asphalt terrain (probably due to its gray color, which has a poorly defined hue value).

In the present work, the appearance subspace makes use of a new color representation, which is normalized and

inspired from HSV, using the following features:

$$F_r^{\text{appearance}} = (\Delta RG, \Delta GB), \tag{25}$$

with

$$\Delta RG = \frac{R - G}{v},$$
$$\Delta GB = \frac{G - B}{v},$$
$$v = \max(R, G, B).$$

The divisor is named v as it is exactly how the V value is defined in the HSV color space. This operation imposes a fixed V value and sensibly solves the problems with changing illumination. The differences between the values R and G, and G and B, are enough to describe the color then. This representation is different from the normalized RGB color space due to this V value removal, which is a measure of where a particular color lies along the lightness–darkness axis. Removing this parameter allows the feature to be more robust to changes of illumination.

3.5. Prediction

This subsection concentrates on the use of the predicted M_{RTI} and the path planner E*. As both the learning part and path planning part are based on grid-based methods, the interaction is easy to foresee. In fact the cell sizes of both grids are subject to constraints that are different and contrary. On one side, the learning grid (G_l) cells' size must be sufficient to log a portion of signal from the local sources that is significant. On the other side, the path planning grid (G_e) cells must be fine enough to drive the rover efficiently. For these reasons, the two grids are aligned but have different resolutions. The details of both are presented in the following parts.

3.5.1. Grids Sizes

According to the position and orientation of the rover, the grid is overlaid onto every new image taken. Then the image is divided into patches corresponding exactly to the cells and the visual features are computed and stored into each G_l corresponding cell. The local sources are handled based on the position of the wheels as previously explained. The size of the grid is defined as the rover's radius, which is slightly larger than half its width:

$$S_{G_l} = \frac{\text{length}_{\text{tot}}}{2} = \frac{0.196 + 0.648}{2} = 0.422 \text{ m},$$
$$S_{G_e} = \frac{S_{G_l}}{\lambda_{le}}.$$

However, such a grid size is not fine enough for the planning and therefore a resolution λ_{le} times higher is used for G_e. In our implementation, $\lambda_{le} = 5$, which corresponds to an E* resolution smaller than 10 cm. Such a resolution is sufficient for accurately planning the rover trajectory while having a reasonable computational cost.

3.5.2. Differential Prediction

The visual features F_r are computed for each one of the patches extracted from the images. The patches correspond to a portion of the image defined by the projection of the G_l cells on the image. Based on those F_r and according to Eq. (10), the most probable F_l are assessed for each one of the cells and correspond to the expected RTI. An additional predicted cost $r^p(c_e)$ can be computed for each one of the observed cells of G_l, adjusting the E* propagation costs of the corresponding G_e cells, as follows:

$$r(c_e)^+ = r(c_e) + r^p(c_e). \tag{26}$$

The idea is to adapt the cells' cost $r(c_e)$ by a small amount (ϵ) each time a prediction can be performed. ϵ is either positive when the predicted RTI metric (M^p_{RTI}) is "good" or otherwise negative. As a reminder, note that the meaning of a "good" or "bad" RTI is assumed to be provided by the user.

Hence, let us assume that the RTI performance is measured by the softness and that softer soils are preferred. In this case, a "good" RTI corresponds to preferred lower values of the feature expressing the softness. The additional predicted cost [Eq. (27)] is then computed as follows, based on the RTI experienced from the beginning of the test [Eq. (28)]:

$$\text{if good} < \text{bad}$$
$$r^p(c_e) = \begin{cases} +\epsilon & \text{if } M^p_{\text{RTI}} < M^m_{\text{RTI}} \\ -\epsilon & \text{if } M^p_{\text{RTI}} > M^m_{\text{RTI}} \\ 0.0 & \text{if } M^p_{\text{RTI}} = M^m_{\text{RTI}} \text{ or if unknown} \end{cases}, \tag{27}$$

with

$$M^m_{\text{RTI}} = \frac{M^{\max}_{\text{RTI}} + M^{\min}_{\text{RTI}}}{2}. \tag{28}$$

The additional predicted cost is positive when the RTI predicted (F^p_l) corresponds to a characteristic smaller than the median RTI encountered during the test (until then) and negative otherwise. As a practical detail, $\epsilon = 0.05$ is used during our experiments.

3.5.3. Prediction Processing

Two steps are performed prior to integrating the additional predicted cost $r^p(c_e)$ within E*, as suggested by Eq. (26). First, a step dilating the additional predicted cost is needed. During the learning process, the RTI is computed associating the images patches with the wheels' positions. On the other hand, E* plans a path as a trajectory to be followed by the center of the rover. Therefore, the cost expected by E* has to characterize the RTI of a rover center and $r^p(c_e)$ must be updated accordingly. From the point of view of prediction, the additional predicted cost corresponds to a rover

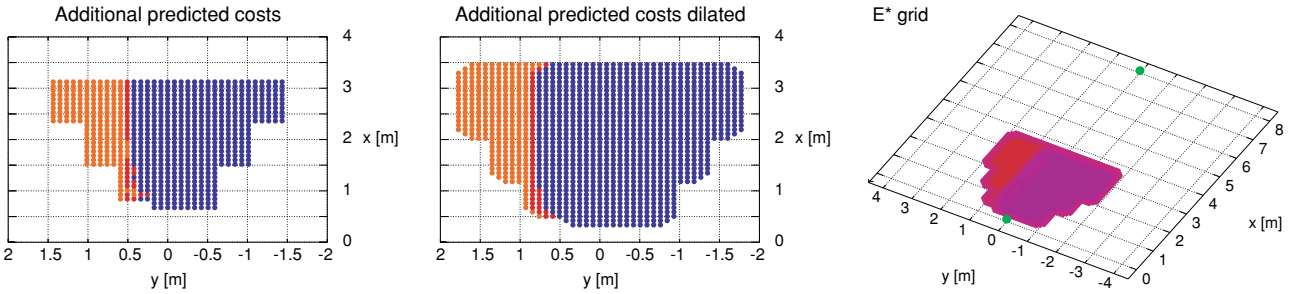


Figure 9. Different steps of the terrain prediction. Original prediction (left), dilated prediction (middle), and the resulting E* grid (right). Note that on the latter, the starting position and goal are depicted with the lighter dots.

whose position and orientation places one of its wheels into the cell. By considering the position of the wheels with respect to the rover's center, this problem is overcome by expending the worst additional predicted cost, which corresponds to applying a dilation mask of radius width/2 on $r^p(c_e)$.

The second step corresponds to a simple smoothing operation. In fact, discontinuities in $r(c_e)$ would result in unnatural paths planned by E*. Thus, having positive and negative additional predicted costs creates edges at the terrains' borders and results in traces planned that tend to follow these discontinuities. In the end, the resulting traces contain sharp turns and are very abrupt. The use of a very simple Gaussian filter of size λ_{le}, corresponding to one cell G_l, solves this issue.

Figure 9 depicts $r^p(c_e)$ on the left, the result of the dilation operation (middle), and the resulting E* grid (right). The additional predicted costs sent are based on the principle of trying to minimize the vibrations within the rover's structure. Following this rule, grass is better than asphalt. The specific situation depicted here corresponds to Figure 10. The grass, on the left-hand side, leads to a positive $r^p(c_e)$. The asphalt terrain receives, on the contrary, a negative one.

3.6. Summary

The entire implementation of the current approach is described in this subsection and is summarized in pseudo code in Algorithm 1, corresponding to the whole software depicted in Figure 2, and results can be observed in Figure 10. The rover is placed in a situation in which two terrains can be observed. These two terrains were previously encountered by the rover, and therefore the RTI can be predicted ahead of the rover. The grass and asphalt areas correspond, respectively, to positive and negative $r^p(c_e)$. The patches in between are unknown (or class 0). Note the dual resolution that is used to refine the border between different areas. When a cell c_l is classified as unknown and surrounded by two different known neighbors, the image patch is divided into subpatches corresponding to its cells c_e. Those subpatches are processed to classify them and predict their RTI. This ensures a good detection of the transition of the two terrain types. Note that the learning part is not affected by this procedure and does not take the subpatches into account.

It can be noted that the prediction of the local features is pretty straightforward, especially because there is only one remote subspace; however, it is not difficult to use several remote subspaces because their final prediction would be the result of naive Bayesian fusion.

Algorithm 1 RTILE pseudo code

Input $Pose_{rover}$; $Data \leftarrow$ Pose estimate and Sensors data
Output: $Ctrl_{rover} \leftarrow$ Rover commands
while *Waypoint not reached* **do**
  Update the rover's pose within G_l and G_e;
  Compute d, the distance to the last position where an image was taken;

  If *cell containing wheel pose changed from* $c_{l,i}$ *to* $c_{l,j}$ **then**
    Compute RTI, of $c_{l,i} \rightarrow F_{l,i}$;
  end

  If $d > S_{G_l}$ **then**
    Take an image;
    Extract patches from image ($n_{patches}$);

    Compute $F_{r,n}$ with $1 \leq n \leq n_{patches}$;
    Predict RTI $\rightarrow F^p_{l,n} = f(F_{r,n})$;

    Compute $r^p(c_e)$, predicted additional cost;
    Process $r^p(c_e)$, dilate and smooth;
    Send $r^p(c_e)$ to E* $\rightarrow r(c_e)^+ = r(c_e) + r^p(c_e)$;
    Update E* *trace*;
  end

  If c_{ln} *moved behind the rover* **then**
    Create s_{in} corresponding;
    Save s_{in} for learning;
  end
  Compute $Ctrl_{rover}$ based on *trace*;
  Send $Ctrl_{rover}$ to the rover control module;
end

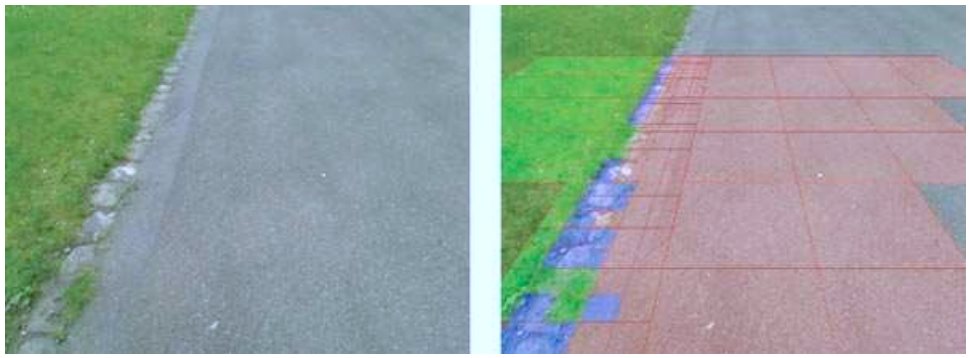


Figure 10. Original image taken from the CRAB (left) and its terrain cell decomposition and prediction using two resolutions (right). The overlay corresponds to the additional predicted cost attributed.

4. RESULTS

This section is dedicated to the description of the tests performed, their focus and the corresponding results. Four tests are presented here showing different aspects of the approach. The first and second tests illustrate the learning capability of the approach, in both controlled and natural environments. The third test shows the action of the whole method in a controlled environment, whereas the fourth test is performed in a more realistic environment.

4.1. Learning: Controlled Environment Trial

This test is designed to show the learning capabilities based on the rover's knowledge. In this context, the three subspaces presented in the preceding section are used. The RTILE approach enables the rover to learn its RTI models in an unsupervised manner within each subspace, which means that the classes learned are not based on a human-defined criterion. Therefore, it can be difficult to show the learning capabilities if the output is not understandable. For this reason, this first test places the rover in a controlled environment where its RTIs are sufficiently distinctive and expected to be learned into different classes. This experiment is also referred to as test 1.

4.1.1. Setup

The tests were performed indoors, using carpets to control the terrain appearance (Figure 11). The softness of the terrain can be controlled by the number of carpet layers used, whereas the bumpiness is changed by using wooden sticks placed under the carpets. Thus the appearance, the bumpiness, and the softness could be defined independently. The test consists of four successive runs labeled A–D. In each run, the rover is placed in a situation with a given *appearance*, *bumpiness*, and *softness*. The rover is then commanded to move forward on a 5-m-long trajectory and is expected to learn correctly from the data logged during the run. The rover moves at a speed of 10 cm/s. According to Figure 2, the *near-to-far* and *learning* parts are used, whereas the *path planning* and the *prediction* part are dismissed. Except for the first run, in which no prior information is available for the learning algorithm, the knowledge resulting from the learning performed on the previous runs is available.

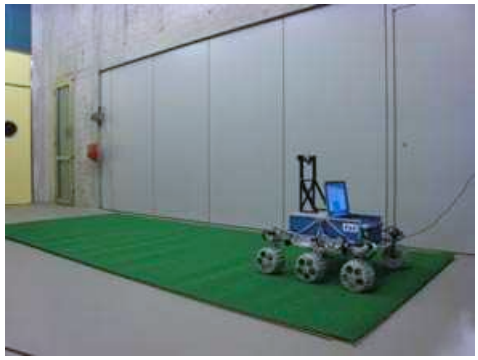


Figure 11. CRAB performing one of the runs of test 1.

The following runs are performed, and the associated terrain can be seen in Figure 12:

- In run A, the rover moves on the concrete, hardest surface, which is flat.
- In B, a single layer of grass-like carpet is used. The terrain is flat.
- For run C, a brownish carpet with wooden sticks is used. The wooden sticks are 2.4 cm in height and 4 cm in length and are placed 70 cm apart.
- Finally in run D, the grass-like carpet is reused. The terrain is flat, but this test makes use of three layers of carpet.

In a first step, the sequence of runs A, B, C, and D, named *ABCD*, is analyzed in detail. Then the knowledge acquired after the four runs for a different sequence is also analysed. It is interesting because the learning is based on processing the data on the basis of what is already learned. The different sequences analyzed are the following: *ABCD*, *DCBA*,

Figure 12. Appearance of the various terrain types used for test 1. Runs A (left), B and D (middle), and C (right).

CDAB, *CADB*, and *BADC*. The first run of every sequence is performed without any prior knowledge. Note that the length of the trajectory is equal for each run and the number of samples generated for each run is also equal. Thus, over a 5-m trajectory, 18 samples are generated.

4.1.2. Results

The progress of the rover knowledge resulting from the learning performed on the successive runs data of sequence *ABCD* can be seen in Figure 13.

The first run results in learning an initial class for each one of the subspaces, which is normal because no prior information is available at this point and therefore all the samples can be labeled only as unknown and used to learn the first class.

The second run presents a different appearance, as well as a softer terrain. As expected, a second class was learned in both subspaces but the bumpiness model remains as a single class, meaning that the bumpiness features were correctly classified.

Figure 13. Progress of the rover knowledge during the first test sequence, *ABCD*. The *softness* (left) and *bumpiness* (right) subspaces can be seen in parallel or independently. Their number of elements is alike as it depends on the *appearance* classes.

Run C offers a different appearance and a different bumpiness. Both were correctly learned, and the softness is correctly recognized as equivalent to the previous test; both have only a single layer of carpet.

Finally, the last run has only a softer surface, but it was correctly learned because a third softness class is added at this point. Both the appearance and the bumpiness are correctly interpreted as previously learned.

The final distributions have the data depicted in Figure 14. The class number is written beside the corresponding distribution, followed by the mean values of the corresponding features. Note also that the whole features space is not shown.

The rover knowledge resulting from the learning performed for the different sequences can be seen in Table III. It shows that the different sequences end up with very similar results, but with little variation. The sequences *DCBA*, *CDAB*, and *CADB* have a marginal additional class appearing in the *softness* subspace, whereas the sequence *BADC* has one appearing in the *bumpiness* subspace.

4.1.3. Summary

The learning approach is successfully validated in this test, and several elements can be highlighted. First, the sequence *ABCD*, presented in more detail, shows interesting elements. A new class is learned each time it is expected. Then, on the other hand, the previously learned classes are correctly recognized if the rover is confronted to a characteristic that was learned. This can be seen for each one of the subspaces, the appearance (D), the bumpiness (B and D), and the softness (C). This proves that the features in each of the subspaces are good. The approach is also capable of relearning as the inference of the softness has evolved after the fourth run. The subspaces are being learned independently; hence the approach is flexible.

Second, considering the different sequences tested, Table III shows that the different sequences end up with the same results, with very few variations, which proves that the features used characterize the terrains well and that the knowledge acquired is meaningful.

Also note that the classification of the different runs into the expected classes is the result of well-defined subspaces and a specifically designed experimental setup. The

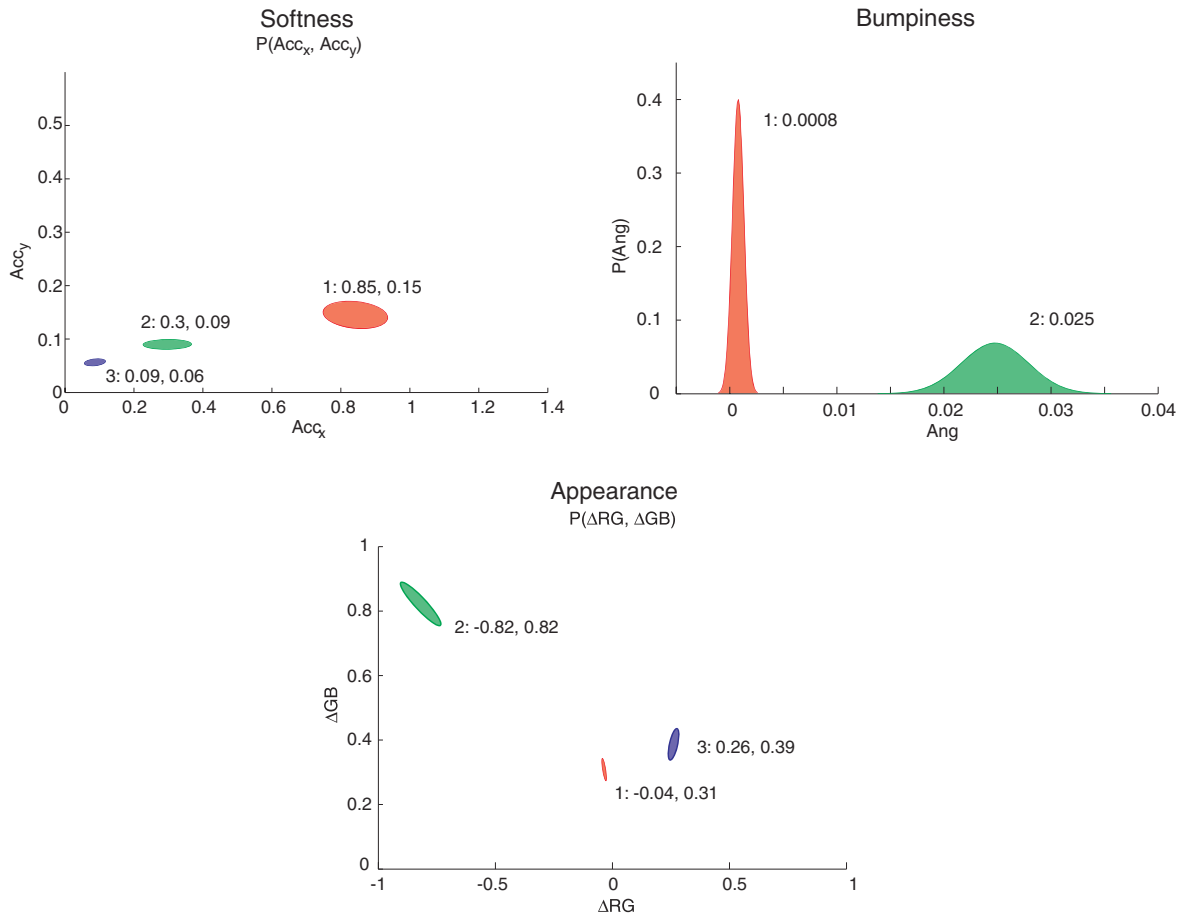


Figure 14. Representation of the learned distributions. *Softness* (upper left), *bumpiness* (upper right), and *appearance* (bottom). Note that the whole feature space is not represented. In each plot, the classes' numbers are written close to the corresponding probability distribution, in addition to the mean values.

Table III. Test 1 classification of samples.

		Softness (%)				Bumpiness (%)		
Sequence	Appearance class	1	2	3	4	1	2	3
ABCD	1	100.0	0.0	0.0	0.0	100.0	0.0	0.0
	2	0.0	50.0	50.0	0.0	100.0	0.0	0.0
	3	0.0	100.0	0.0	0.0	0.0	100.0	0.0
DCBA	1	100.0	0.0	0.0	0.0	100.0	0.0	0.0
	2	0.0	44.4	50.0	5.6	100.0	0.0	0.0
	3	0.0	100.0	0.0	0.0	0.0	100.0	0.0
CDAB	1	100.0	0.0	0.0	0.0	100.0	0.0	0.0
	2	0.0	44.4	50.0	5.6	100.0	0.0	0.0
	3	0.0	100.0	0.0	0.0	0.0	100.0	0.0
CADB	1	100.0	0.0	0.0	0.0	100.0	0.0	0.0
	2	0.0	44.4	50.0	5.6	100.0	0.0	0.0
	3	0.0	100.0	0.0	0.0	0.0	100.0	0.0
BADC	1	100.0	0.0	0.0	0.0	100.0	0.0	0.0
	2	0.0	50.0	50.0	0.0	94.4	0.0	5.6
	3	0.0	100.0	0.0	0.0	0.0	100.0	0.0

subspaces are well defined because they recognize specific terrain characteristics and their impact on the rover. The different runs were designed with knowledge of the subspaces to achieve the terrain representation. It is obvious that using different subspaces (e.g., using other sensors) or using a different setup would result in a different terrain representation.

Finally, as Figure 14 shows, the learned distributions of the *appearance* subspace are peaked. This corroborates the approximation assumed in Eq. (8). Therefore the learning method used in the context of this work offers all the required properties.

4.2. Learning: Natural Environment Trial

This test is presented to analyze the capabilities of the learning aspects of the approach in a real environment. Therefore, it is very similar to the previous test except for the lack of clear ground truth regarding what has to be learned and how. Using a natural environment allows the algorithm to really learn according to its own terrain representation. Nevertheless the results can be analyzed in terms of segmentation and sensitivity. This test is referred to as test 2.

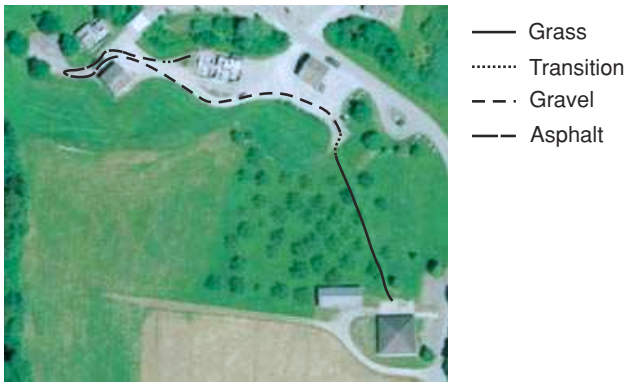


Figure 15. Trajectory of the CRAB rover during test 2. Different types of terrains as well as the transition in between can be observed. The terrain classes leave a lot of room for interpretation, because the ground truth is difficult to define.

4.2.1. Setup

The CRAB rover is driven manually over a 310-m-long trajectory at a speed of 10 cm/s in the test environment of a little village in the countryside. During the test, the rover was driven on various terrain types such as thick grass, gravel, and asphalt. The transitions in between these types can also be counted as separate terrains. The rover trajectory and information regarding the terrains can be observed in Figure 15. Note that the surface encountered by the rover is generally flat.

4.2.2. Results

The test has the following result with respect to the learning algorithm. We will focus here on the *appearance* subspace, as it is then the entry point to differentiate terrains and reuse the knowledge acquired.

In the first section, the algorithm learned the grass as the first class. The second class learned is the transition between the grass and the gravel. Then the gravel resulted in learning a new class, class 3, but this class also includes asphalt, which is not distinguished from gravel. In the gravel section, some small patches of grass resulted in learning a fourth class. At the end of the test, all the patches acquired from the images are classified. These results can be observed in Figure 16.

Figure 16. Typical terrains observed during the test (top) and their corresponding classification results (bottom). The overlay colors have the following meaning: red, class 1; green, 2; blue, 3; and black, 4. The class 0, or "unknown," is in yellow (in the right-hand-side picture).

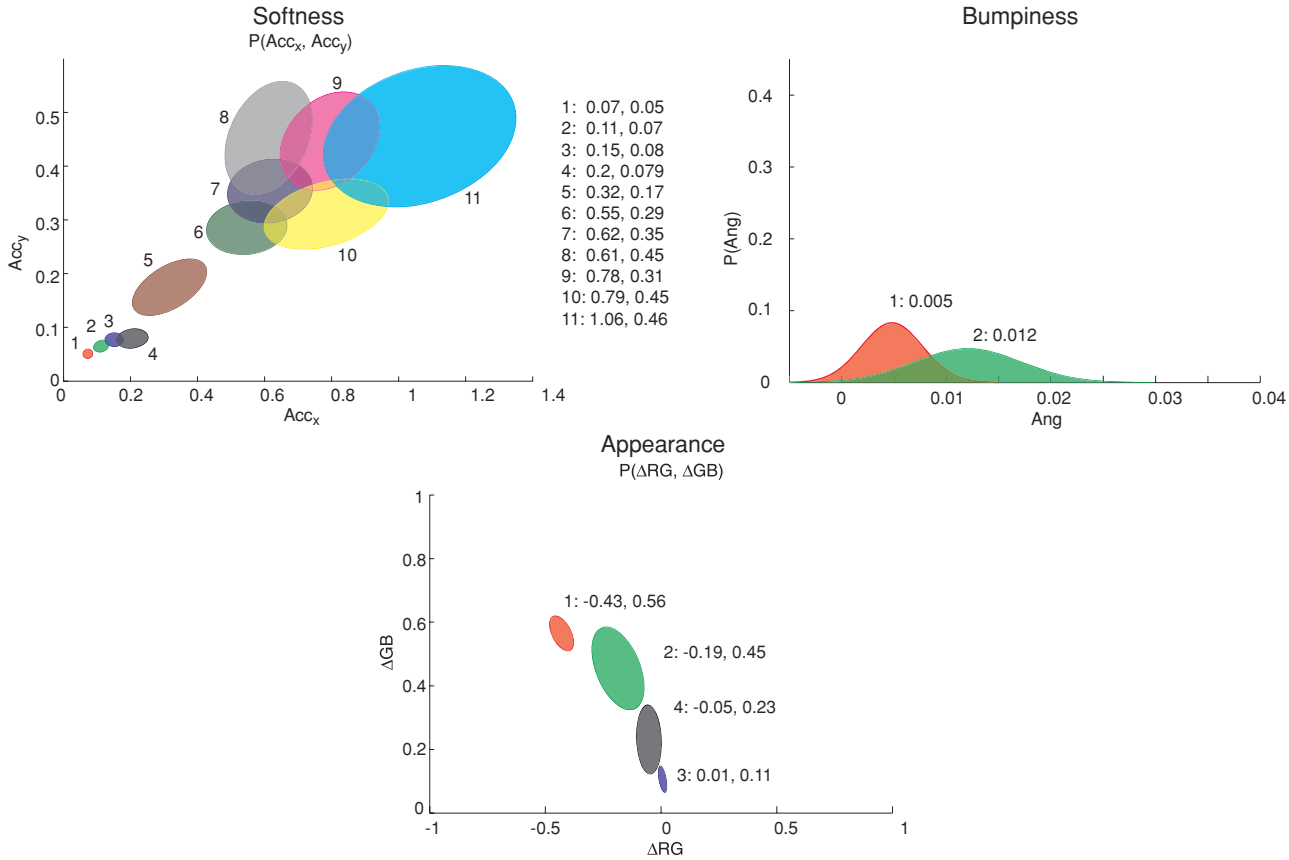


Figure 17. Representation of the learned distributions. *Softness* (upper left), *bumpiness* (upper right), and *appearance* (bottom). Note that the whole feature space is not represented. In each plot, the classes' numbers are written close to the corresponding probability distribution, in addition to the mean values.

Regarding the RTI subspaces, the *softness* proved to be very sensitive as no fewer than 11 classes were obtained from the test. The *bumpiness* resulted in only two classes, but this can be expected because the CRAB did not face any obstacle whatsoever. The details of the probability distributions learned from this test can be observed in Figure 17.

4.2.3. Summary

The learning algorithm works well, and the results for each subspace are good. In the appearance subspace, the asphalt terrain is not distinguished from gravel due to the lack of features representing the texture. On the contrary, although the features used are fairly simple, a lot of different terrain types are "seen" in the softness subspace. This experiment shows that it is important to use the right features to represent the terrain in such an unsupervised and generic approach.

As for the previous test, and as Figure 17 shows, the learned distributions of the *appearance* subspace are peaked. This shows again that the approximation of Eq. (8) is well founded.

4.3. RTILE: Controlled Environment Trial

The test is designed to verify the validity of the whole approach and especially the rover's behavior influence based on the knowledge available and learned during the test. This implies that we need to choose the function that rates the RTI, M_{RTI}, and also depends on the test environment. These tests are aimed at showing the impact of the overall method on the rover control, so the function is chosen to minimize the amount of vibration within the structure, thus defining

$$M_{\mathrm{RTI}} = \sqrt{F_{l,1}^{\mathrm{softness}^2} + F_{l,2}^{\mathrm{softness}^2}}. \tag{29}$$

Therefore, in this test, the path is planned according to the *softness* prediction of the terrain ahead of the rover. Note that the *bumpiness* is still acquired and handled but it is not

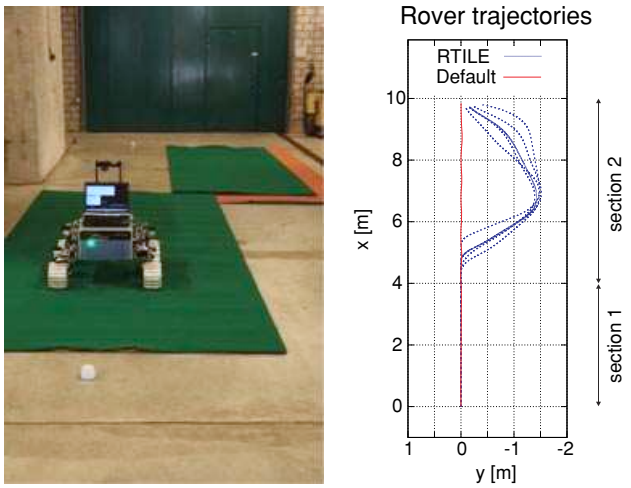


Figure 18. Setup of second test and resulting trajectories.

taken into account to evaluate the M_{RTI} driving the rover's path.

Following the successful results of the *learning trial* and in order to validate the approach step by step, the present test is performed indoors, in a controlled environment. Every component of the RTILE approach described in Section 2 is used. This test is referred to as test 3.

4.3.1. Setup

The rover is initially commanded to move toward a waypoint 4 m ahead and then another 10 m ahead of the starting position. The configuration of the environment can be seen in Figure 18, and two types of terrain can be observed. The first terrain type is concrete, whereas the second is grass-like carpets. The straight trajectory between the starting position and first waypoint drives the CRAB mostly on the grass-like carpet, and a straight trajectory between the first waypoint and the goal is only on concrete. Such straight trajectories can be considered as the default trajectories because the rover would move along them to reach the waypoint without taking the environment into account. The rover starts the test with a prior regarding the concrete and its corresponding softness, which means that the rover was driven for a few meters on this surface and it was able to characterize it. Therefore, concrete is "known" at the beginning of each test run. The test is repeated five times, and the rover drives autonomously at a speed of 6 cm/s.

4.3.2. Results

The first section of the test, which reaches the first waypoint, is performed in a straight line. The prior is not sufficient to challenge the obvious straight movement on a surface that is completely unknown. Upon reaching the waypoint, the samples acquired are processed and learned. This results in the creation of a new appearance and softness class corresponding to the grass-like carpet. In the second section of the test, the default trajectory is challenged by the opportunity of moving almost all way through on the grass-like carpet, which is softer than the concrete. As shown in Figure 18, the rover takes a slightly longer path to remain on the carpet.

To express what happened during the test, it is necessary to define metrics summarizing the rover behavior. Thus, two metrics are defined here, one corresponding to the distance traveled and the other corresponding to the softness of the terrain:

$$M^{\text{Dist}} = \sum \sqrt{\Delta x^2 + \Delta y^2}, \tag{30}$$

$$M^{\text{Soft}} = \sqrt{\overline{\|\alpha_x\|}^2 + \overline{\|\alpha_y\|}^2}, \tag{31}$$

with i being the trajectory section between waypoints i and $i+1$. As this test is composed of two sections, the first one being the same for the RTILE approach as well as the default approach, three pairs of results are presented in Table IV.

4.3.3. Summary

The *softness* of the second section has a value significantly lower using the RTILE approach. This is a success as a terrain minimizing the chassis vibrations is preferred. This result is consistent and repeatable, as the very low standard deviation over M^{Soft} shows. Figure 18 indicates that the trajectories of the five runs have globally the same shape, but with some variation, which is reflected by the slightly higher standard deviation over M^{Dist}, corresponding to the rover trajectory in the end of Section 2.

Table IV. RTILE controlled environment (test 3) results.

	M^{Soft} (rad/s^2)			M^{Dist} (m)		
		Section 2			Section 2	
	Section 1	Default	RTILE	Section 1	Default	RTILE
Mean	0.14	0.42	0.24	3.74	6.23	6.90
Std. dev.	0.02	0.02	0.02	0.02	0.02	0.17

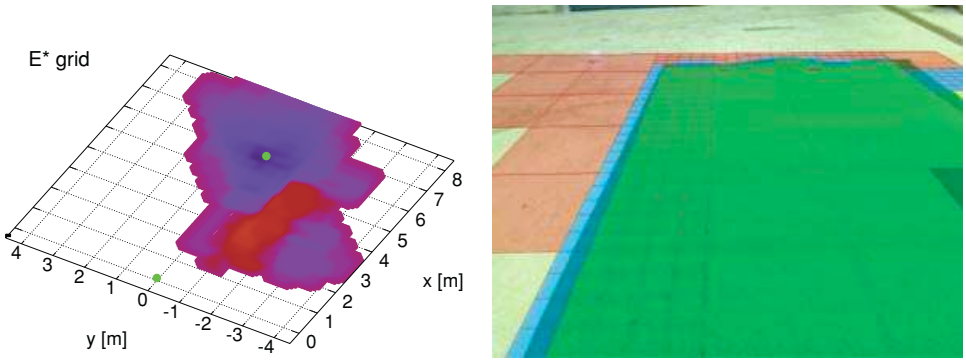


Figure 19. Final prediction map (left) and prediction on the carpet (right). The prediction map shows the value of Λ for each one of the observed c_l, which corresponds to the sum of the multiple rewards computed. The area depicted corresponds only to the second test section, the first bringing nothing as all the F_r are unknown at this time. The second carpet can be well identified on the prediction map. On the right-hand side, the prediction performed during the second test section on the carpet is shown. The result is very satisfying as only the transition cells are unknown.

When using RTILE, sections 1 and 2 correspond to the same terrain (grass-like carpet), but their M^{Soft} values are quite different. This difference is induced by the very last part of the trajectory, just before the goal, where the rover has to move a couple of meters on concrete. This raises the value of the $M^{\text{Soft}}_{2,\text{RTILE}}$.

In despite of these variations, the paths show that the rover behavior is influenced by the learned RTI, and this knowledge allows the path of the rover to be adapted. The E* path planner is successfully used and integrated to the approach, and the inference of the RTI on the terrain is successful.

Finally, to have an insight into the test, the map rating the RTI in this environment and the processing of an image to recognize previously met terrain are shown in Figure 19.

4.4. RTILE: Natural Environment Trial

The test is very similar to test 3. It aims also at verifying the entire approach but goes a step further by using a natural environment. Here, natural means that the environment is not as controlled as before, and some additional uncertainties are therefore taken into account. Therefore, the same M_{RTI} function is used as with the previously presented trial. Thus, the path is planned according to the *softness* prediction of the terrain ahead of the rover, focusing on minimizing the vibration within the rover chassis. Note that for the same reasons as for the previous test, the bumpiness is not taken into account. This test is referred to as test 4.

4.4.1. Setup

The test is divided into two parts. The setups of both parts are depicted in Figure 20.

In the first part, the rover is commanded to reach a first waypoint 5 m ahead, and then another 10 m ahead of the starting position. Performing a straight, direct trajectory, the rover moves first on asphalt and then on grass. In the second part, the rover aims for a goal positioned 8 m ahead

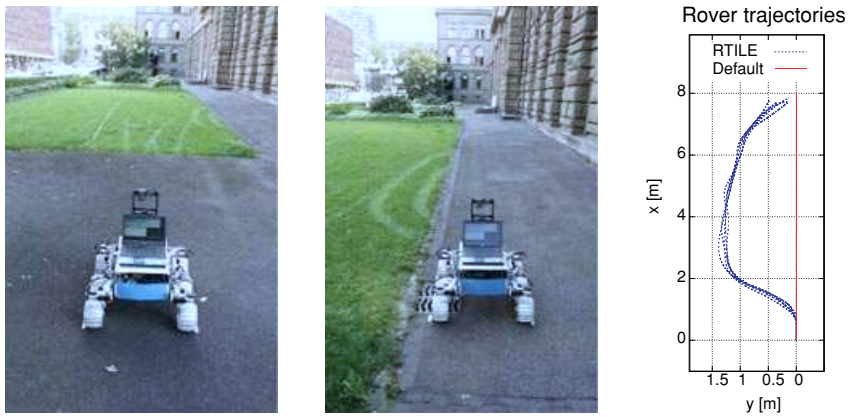


Figure 20. Test 4: CRAB rover at the starting point of first part (learning phase on the left-hand side), of the second part (middle), and resulting test trajectories (right). The traces left by the rover in the grass during the previous trial can be observed.

Table V. RTILE natural environment (test 4) results.

	M^{Soft} (rad/s^2)		M^{Dist} (m)	
	Default	RTILE	Default	RTILE
Mean	0.40	0.24	7.72	8.44
Std. dev.	0.04	0.02	0.02	0.07

and a straight trajectory would drive the rover on asphalt only. The test is also repeated five times to have more reliable results, and the rover drives autonomously at a speed of 6 cm/s.

4.4.2. Results

The first part is executed without a prior, and the waypoints are reached via a straight trajectory. It gives the opportunity to learn the asphalt and grass terrains, in terms of both appearance and softness. The second part of the test makes use of the knowledge acquired in the first part. In this case, and as depicted in Figure 20, the default trajectory is not used and a area) trajectory moving on the grass terrain is preferred. The resulting metrics of this test are presented in Table V. Figure 21 shows the E* propagation costs, $r(c_e)$, resulting from the test.

4.4.3. Summary

As the results above show, the use of RTILE allows the rover to select a softer terrain to reach the goal. As a side effect, this increases the distance traveled because the trajectory is longer.

This test shows again the rover's behavior influenced as a result of the learned RTI, which influences the path of the rover. The test started without any prior knowledge regarding the rover behavior; a short 5-m trajectory on both terrains allowed the rover to sufficiently learn both terrains.

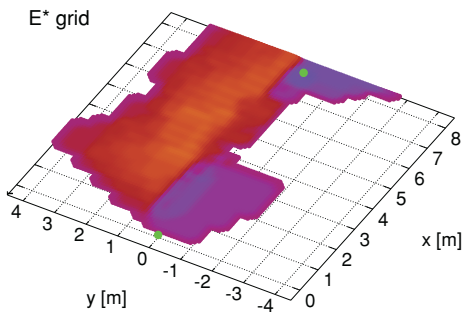


Figure 21. Final prediction map of test 4 showing the value of Λ for each one of the observed c_l. The grass terrain is well recognized and predicted. It corresponds to the area on the left-hand side.

4.5. Lessons Learned

The *learning trials* show that it is possible to learn from the rover's experiments, which can be done without any prior and in an unsupervised way, except for the definition of the cost regarding the learning part. First, the features have to be designed with care, to represent a specific and meaningful RTI model. As the results of test 2 show, the features must also be sensitive enough to be able to express different RTI models according to different terrain types. Second, in order to have reasonable computational time and memory usage, the features spaces must necessarily have a reduced number of dimensions, which also underlines the need for carefully designed features. To summarize, using an unsupervised approach requires the transfer of much more knowledge within the features. From a general point of view, it can also be noted that too many classes are generally better than too few classes in the subspaces. A finer representation of the terrains encountered is achieved by having more classes modeling the RTI.

The two RTILE trials show that the knowledge acquired to qualify the RTI can be used to influence the rover behavior or its path. Using RTILE has a cost regarding the rover behavior, as the resulting trajectory is longer in both tests. This raises the following questions: To what extent can the rover trajectory be influenced by the RTI? And what is the maximal deviation from the default path in order to have an optimal rover behavior?

Another point is that the remote subspaces are particularly important and are a key element of RTILE. They are the entry point to use the knowledge acquired and make any prediction as they are used to recognize the previously met terrains. If the *appearance* subspace, in the context of this article, regroups too many terrain types within the same class, the RTI predicted does not give any useful information. Then, the remote feature classes are better being too specific, rather than too largely defined.

Finally and similarly to the previous point, the camera's field of view has a big influence over the RTI predicted. The camera used in this research has a reduced field of view, and therefore all the information available in the images is used to predict the RTI. In this context it is very important to be able to detect as precisely as possible the transition between different terrains, and in this perspective, the use of a dual-resolution grid improves the results.

5. CONCLUSION

The RTILE approach aims to learn the RTI based on the rover's experiments, without any prior. The knowledge acquired is subsequently used to form RTI predictions on terrains lying ahead of the rover, which results in an influence

on the rover trajectory. The approach is implemented and successfully tested on the CRAB rover.

The *near-to-far* technique generates samples associating remote and local data, whose probability distribution can be learned with the ProBT tool. The E* path planner, combined with the predicted RTI, allows the rover's behavior to be influenced. The notion of subspaces, representing various RTI models, is also introduced, and three of them, namely *softness*, *bumpiness*, and *appearance*, are presented.

Finally the tests conducted with the CRAB rover show the feasibility of the approach as well as its successful implementation. The unsupervised learning indicates good results in detecting new classes within subspaces, and the entire approach is successfully tested in both controlled and natural environments. The tests presented in this article show the interest and versatility of the approach.

The method proposed in this article is not foreseen as a replacement to reduce design activity or to get rid of any wheel–ground interaction model or even to avoid considering the terrain traversability. The approach proposed here has to be considered as a complementary tool, a performance enhancer for a well-designed rover.

A few words about slip are necessary, as using it as a subspace can be very appropriate. Slip is important as several aspects of a rover mission demand as little slip as possible. First, navigation is more accurate if the rover does not slip. Second, because slipping wheels do not contribute to the rover's movement, slip is a loss of energy. Finally, potential slip increases the risk of operation failure due to loss of control of the vehicle. Despite all these elements, slip is not considered in this work, and this is mainly due to two reasons.

- Slip can be measured only if a reference (or ground truth) is available. This is not the case for the CRAB rover (but is being developed).
- The goal of the current research is to show the rover under the influence of RTI models learned from its experiments and to show a methodology to do this. In this context, the types of RTI models used are not as important as the approach itself. Thus, the authors do not feel compelled to use slip in this approach.

Henceforth, although the approach description is sound and the tests corroborate its potential, some questions are still open. Among them, two are of importance and are the focus of upcoming work. The first one concerns the learning aspect. In the work presented here, all the new samples that could not be classified (or that are classified "unknown") are used to generate a new class. The problem is that nothing proves that only a single new terrain was discovered, and this is what is implicitly assumed. To solve this potential problem, a clustering step has to be added before the learning. It can also be noted that this is not a critical problem if the learning process is performed more frequently than there are terrain changes in the environment. The second issue is linked with the characterization of the approach. It is known that the E* path planner proposes a trade-off between the path length and the trafficability cost (in the case of RTILE). In this context the values assigned to $r(c_e)$ have to be characterized, answering the following questions: In an environment with several terrains, what is the impact of a change of $r(c_e)$ on the E* proposed path? What is the relation between the variation of propagation cost and the amplitude of the difference between the RTILE and the *default* path? At the moment, it is assumed that the user provides RTILE with the metric M_{RTI} driving the cost. The next step is to refine the interaction with E* in order to be able to use an input such as the following: A detour of amplitude δm can be performed for a predicted δM_{RTI} between two terrains.

ACKNOWLEDGMENTS

The authors gratefully acknowledge Mark Höpflinger and Majid Nili for their help and expert advice in the early phase of the project. Thanks also to Dr. Francis Colas and Dr. Mike Bosse for their inputs regarding the redaction of this article. This project was partially funded by the European Space Agency (ESA). Finally, a special thank you to our trustworthy Cunégonde and Bröiseli, who are always ready to give a hand.

REFERENCES

Angelova, A., Matthies, L., Helmick, D., & Petrona, P. (2007). Learning and prediction of slip from visual information. Journal of Field Robotics, 24(3), 205–231.

Apostolopoulos, D. (2001). Analytical configuration of wheeled robotic locomotion. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA.

Bajracharya, M., Howard, A., Matthies, L., Tang, B., & Turmon, M. (2009). Autonomous off-road navigation with end-to-end learning for the LAGR program. Journal of Field Robotics, 26(1), 3–25.

Bessière, P., Laugier, C., & Siegwart, R. (Eds.). (2008). Probabilistic reasoning and decision making in sensory-motor systems. Springer Tracts in Advanced Robotics. Berlin: Springer.

Brooks, C., & Iagnemma, K. (2005). Vibration-based terrain classification for planetary exploration rovers. IEEE Transactions on Robotics, 21(6), 1185–1191.

Brooks, C., & Iagnemma, K. (2007, March). Self-supervised classification for planetary rover terrain sensing. In Proceedings of the 2007 IEEE Aerospace Conference, Big Sky, MT.

Dupont, E.M., Moore, C.A., Collins, E.G., & Coyle, E. (2008). Frequency response method for terrain classification in autonomous ground vehicles. Autonomous Robots, 24(4), 337–347.

Ferguson, D., & Stentz, A. (2005, October). Field D*: An interpolation-based path planner and replanner. In ISRR, San Francisco, CA (pp. 239–253).

Hadsell, R., Sermanet, P., Ben, J., Erkan, A., Scoffier, M., Kavukcuoglu, K., Muller, U., & LeCun, Y. (2009). Learning long-range vision for autonomous off-road driving. Journal of Field Robotics, 26(2), 120–144.

Halatci, I., Brooks, C., & Iagnemma, K. (2008). A study of visual and tactile terrain classification and classifier fusion for planetary rovers. Robotica, 26(6), 767–779.

Happold, M., Ollis, M., & Johnson, N. (2006, August). Enhancing supervised terrain classification with predictive unsupervised learning. In RSS, Philadelphia, PA.

Helmick, D., Angelova, A., & Matthies, L. (2009). Terrain adaptive navigation for planetary rovers. Journal of Field Robotics, 26(4), 391–410.

Huang, W.H., Ollis, M., Happold, M., & Stancil, B.A. (2009). Image-based path planning for outdoor mobile robots. Journal of Field Robotics, 26(2), 196–211.

Iagnemma, K., Kang, S., Shibly, H., & Dubowski, S. (2004). On-line terrain parameter estimation for planetary rovers. IEEE Transactions on Robotics, 20(5), 921–927.

Ishigami, G., Miwa, A., Keiji, N., & Kazuya, Y. (2006, June). Terramechanics-based analysis on slope traversability for a planetary exploration rover. In ISTS, Kanazawa City, Japan.

Kim, D., Sun, J., Min Oh, S., Rehg, J.M., & Bobick, A.F. (2006, May). Traversability classification using unsupervised online learning. In ICRA, Orlando, FL.

Konolige, K., Agrawal, M., Blas, M.R., Bolles, R.C., Gerkey, B., Sola, J., & Sundaresan, A. (2009). Mapping, navigation, and learning for off-road traversal. Journal of Field Robotics, 26(1), 88–113.

Lalonde, J.-F., Vandapel, N., Huber, D., & Hebert, M. (2006). Natural terrain classification using three-dimensional ladar data for ground robot mobility. Journal of Field Robotics, 23(10), 839–861.

Manduchi, R., Castano, A., Talukder, A., & Matthies, L. (2005). Obstacle detection and terrain classification for autonomous off-road navigation. Autonomous Robots, 18, 81–102.

Mulligan, J., & Grudic, G. (2006a). Editorial for Journal of Field Robotics—Special issue on machine learning based robotics in unstructured environments. Journal of Field Robotics, 23(11–12), 943–944.

Mulligan, J., & Grudic, G. (Eds.). (2006b). Special issue on machine learning based robotics in unstructured environments. Journal of Field Robotics, 23(11–12).

Ojeda, L., Borenstein, J., Witus, G., & Karlsen, R. (2006). Terrain characterization and classification with a mobile robot. Journal of Field Robotics, 23(2), 103–122.

Otte, M.W., Richardson, S.G., Mulligan, J., & Grudic, G. (2009). Path planning in image space for autonomous robot navigation in unstructured environments. Journal of Field Robotics, 26(2), 212–240.

Philippsen, R. (2006). A light formulation of the e* interpolation path replanner. Technical report. Lausanne, Switzerland: EPFL.

Philippsen, R., Jensen, B., & Siegwart, R. (2006). Towards real-time sensor-based path planning in highly dynamic environments. Springer Tracts on Advanced Robotics. Berlin: Springer.

Poppinga, J., Birk, A., & Pathak, K. (2008). Hough based terrain classification for realtime detection of drivable ground. Journal of Field Robotics, 25(1–2), 67–88.

Procopio, M.J., Mulligan, J., & Grudic, G. (2009). Learning terrain segmentation with classifier ensembles for autonomous robot navigation in unstructured environments. Journal of Field Robotics, 26(2), 145–175.

Sermanet, P., Hadsell, R., Scoffier, M., Grimes, M., Ben, J., Erkan, A., Miller, U., Crudele, C., & LeCun, Y. (2009). A multirange architecture for collision-free off-road robot navigation. Journal of Field Robotics, 26(1), 52–87.

Siegwart, R., & Nourbakhsh, R.I. (2004). Mobile robot kinematics. In Introduction to autonomous mobile robots. Cambridge, MA: MIT Press.

Vandapel, N., Huber, D.F., Kapuria, A., & Herbert, M. (2004, May). Natural terrain classification using 3-D ladar data. In ICRA, New Orleans, LA.

Ward, C.C., & Iagnemma, K. (2009). Speed-independent vibration-based terrain classification for passenger vehicles. Vehicle System Dynamics, 47(9), 1095–1113.

Weiss, C., Fechner, N., Stark, M., & Zell, A. (2007, September). Comparison of different approaches to vibration-based terrain classification. In ECMR 07, Freiburg, Germany.

Weiss, C., Froehlich, H., & Zell, A. (2006, October). Vibration-based terrain classification using support vector machines. In IROS, Beijing, China.

Weiss, C., & Zell, A. (2008, July). Novelty detection and online learning for vibration-based terrain classification. In Proceedings of the 10th International Conference on Intelligent Autonomous Systems (IAS 2008), Baden-Baden, Germany (pp. 16–25).

Wellington, C., & Stentz, A. (2004, May). Online adaptive rough-terrain navigation in vegetation. In ICRA, New Orleans, LA.

Appendix E

Pradalier et al. [2008]

Cédric Pradalier, Ashley Tews, and Jonathan Roberts. Vision-based operations of a large industrial vehicle: Autonomous hot metal carrier. *Journal of Field Robotics*, 25(4-5):243–267, 2008

Vision-Based Operations of a Large Industrial Vehicle: Autonomous Hot Metal Carrier

Cédric Pradalier, Ashley Tews, and Jonathan Roberts
Autonomous Systems Laboratory
CSIRO ICT Centre
P.O. Box 883, Kenmore
Queensland 4069, Australia
e-mail: cedric.pradalier@csiro.au, ashley.tews@csiro.au, jonathan.roberts@csiro.au

Received 2 May 2007; accepted 7 February 2008

Hot metal carriers (HMCs) are large forklift-type vehicles used to move molten metal in aluminum smelters. This paper reports on field experiments that demonstrate that HMCs can operate autonomously and in particular can use vision as a primary sensor to locate the load of aluminum. We present our complete system but focus on the vision system elements and also detail experiments demonstrating reliable operation of the materials handling task. Two key experiments are described, lasting 2 and 5 h, in which the HMC traveled 15 km in total and handled the load 80 times.

1. INTRODUCTION

Autonomous ground vehicles (AGVs) have been routinely operating in factories around the world since the 1970s. Thousands of units now deliver products in warehouses, car plants, and aircraft factories. These environments are indoors (free of significant changes in environmental conditions such as lighting) and generally clean and dry. These AGVs typically operate at low speed, in flat driving areas and where the driving surface is normally smooth.

The next frontier for AGVs is that of the outdoor/indoor industrial work site. Examples include steel works, aluminum smelters, large construction sites, and ports. These are environments that contain outdoor areas exposed to the weather and where the driving area may contain hills and the driving surface may be rough. There are very few examples of operational outdoor industrial AGVs anywhere in the world. One notable example is that of the automated shipping port in Brisbane, Australia (Nelmes, 2006). There, a container handling operation has been operating autonomously for the past two years using large autonomous straddle carrier vehicles (AutoStrad AGVs) to stack and move containers. This application is simple, in terms of typical outdoor materials handling tasks, but is impressive in that the AGVs operate in all weather conditions, 24 h a day. The autonomous port is also isolated from people, making the operation inherently safe.

Our project aims to push the technology of AGVs to the next stage: that of an industrial work site with a more difficult materials handling task, harsher environmental conditions, and the issue of people operating around the vehicles.

Figure 1. HMC in the process of picking up the crucible.

We are focusing our research using the example of molten aluminum handling at a smelter. In the aluminum industry, hot metal carriers (HMCs) perform the task of transporting molten aluminum from the smelter (where the aluminum is made) to the casting shed, where it is turned into block products. There are estimated to be 400 HMC vehicles worldwide with a 50/50 ratio of modified forklift-type vehicles to purpose-built vehicles. Our project has been tasked with the automation of the class of forklift-type HMCs. The vehicles weigh approximately 20 t unloaded and resemble forklifts except that they have a dedicated hook for manipulating the load rather than fork tines (Figure 1). The molten aluminum is carried in large metal crucibles. The crucibles weigh approximately 2 t, and they can hold 8 t of molten aluminum usually superheated to above 700°C. Therefore, HMC operations are considered heavy, hot, and hazardous, with safety of operation a significant issue. This application also shares many issues with other applications such as vehicle transport in the steel industry and the large-scale construction industry (large commercial building sites).

1.1. The Challenge: Reliability and Safety

There are many challenges in the HMC indoor/outdoor operating environment. Inside, there is a vast amount of infrastructure, other mobile machines, and people. In various areas, there are large magnetic fields and high temperatures near the molten aluminum pots. Outside, an HMC's path may be surrounded by infrastructure (buildings, fences, stored materials, and other vehicles) and their operation may be affected by environmental conditions such as rain, fog, snow, and heat. Research into automating these vehicles and their operations needs to consider the variability in operating conditions to produce repeatable and reliable performance of the task.

At a typical smelter, a handful of HMCs are used to carry the aluminum from the 500-m-long pot lines to a handful of furnace and casting sites, which can be up to 1 km away. Figure 2 shows an aerial view of a typical smelter. HMCs operate on their own road (the hot metal road) due to the criticality of their operation to the supply chain and the hazardous nature of their cargo. The hot metal road may not be used or crossed by other vehicles without explicit authorization. Smelter staff may cross the road but should not disrupt the HMC's work flow by doing so. In this context obstacle detection is a critical part of any autonomous HMC; however, obstacle avoidance is not. Any unexpected obstacle must be dealt with safely by stopping the vehicle, eventually signaling with the

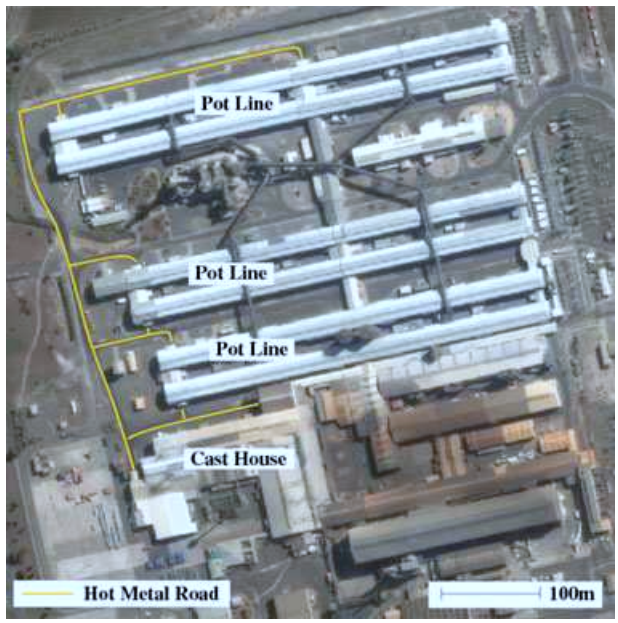


Figure 2. Aerial view of a smelter showing the pot line sheds and the hot metal road.

horn. Replanning a path around the obstacle is neither necessary nor desirable and is potentially unsafe.

1.2. Aims: Outdoor Vision and Visual Servoing for an Autonomous HMC

The HMC must first accurately locate the crucible in order to pick it up. Crucibles are typically dropped off by a crane, after being filled with the hot metal, ready for the HMC to pick up. The exact drop-off point will vary slightly each time, and so the HMC must locate the crucible each time. Owing to the extremely high temperatures, it is not practical to locate any electronic tag-like devices (RFID) on the crucible. Vision is an obvious way to address this location problem and has been a major focus of the research in our project. A human operator uses his/her eyes to servo the hook to the crucible handle's eye, and so vision is a promising sensing method to try. However, there are very few examples of the reliable use of vision for load detection and load localization in outdoor (all weathers) or indoor environments that have strong "outdoor light" projecting through shutters or windows.

Our aim was therefore to develop a reliable vision-based method for crucible detection that could operate in all typical lighting conditions under which a human operator is expected to perform the task, including cloudy and sunny skies, rain, darkness, and indoors under artificial lights.

We also aimed to borrow another method from the human operator, that of visual servoing. The task of hook insertion requires high precision, with the hook having to be driven into a 20-cm-wide target area (the so-called hook-eye). It is clear that operators visually servo the hook of the HMC in order to insert it into the hook-eye of the crucible handle. Our research aimed to determine whether we could mimic this human method of crucible pickup in a robust way.

1.3. Paper Outline

The remainder of this paper is structured as follows. Section 2 presents the work related to automating industrial vehicles and previous work in the areas of vision for automated materials handling tasks. Section 3 outlines the architecture and technical components of our vehicle's systems. Section 4 details the vision algorithms developed for reliably identifying the crucible. Section 5 provides details and performance of various long-duration experiments conducted at our work site. Section 6 describes an analysis of the failure modes of the numerous subsystems of our architecture. Finally, Section 7 concludes the paper with a brief discussion of the significance of the research and outlines our future work plan.

2. RELATED WORK

There has been much research into automating industrial vehicles for cargo transport. Mora, Suesta, Armesto, and Tornero (2003) present a complete system for controlling autonomous forklifts in a warehouse. The forklifts are scheduled from a centralized controller and can be operated autonomously or remotely. Localization of the vehicles is provided by a web-cam sensing lines painted on the floor.

Hasunuma et al. (2003) demonstrate a different approach to automating vehicles by using a humanoid robot to operate the controls of a conventional vehicle. The advantages of using a humanoid are that the vehicle does not necessarily have to be modified to allow pseudo-autonomous operation and the robot can be used for other tasks. The disadvantages are that the current standard of humanoid technology makes controlling a vehicle overly challenging and unreliable. Furthermore, vehicle control has to be encoded in the humanoid, which would be difficult for a closed-loop system considering the complexity of a human conducting the same tasks.

In 1999, our research team demonstrated the autonomous operation of an underground mining vehicle, a load-haul-dump (LHD) vehicle (Roberts et al., 2000; Roberts, Duff, & Corke, 2002) This work showed that two-dimensional (2D) scanning lasers could be used to navigate a vehicle at 20 km/h with little clearance (approximately 0.5 m) between the vehicle and tunnel walls. The system developed worked on the principle of relative or reactive navigation in which the LHD was steered based on the open space observed immediately in front of it. Higher-level turning commands, such as "turn left," "go straight," and so on, were issued by a navigation layer that had a coarse representation of location in the mine tunnel system. The navigator kept track of which section of the tunnel the LHD was in by observing key features such as intersections. The system is now available commercially and has been deployed in a number of mines around the world.

With respect to the load handling task and, in particular, pallet handling by a forklift, several

important research works must be noticed. Garibotto et al. (Garibotto et al., 1998; Garibotto, Masciangelo, Ilic, & Basino, 1996, 1997) present ROBOLIFT, a robotic forklift able to pick up/drop off pallets using computer vision. This work was conducted indoors and used specially designed fiducials. In our case, we are aiming at a minimally modified outdoor setting where these fiducials may not be discriminative enough. In more recent research, Nygards, Hogstrom, and Wernersson (2000) used the image of a visible laser in a camera image to localize a pallet and dock a forklift to it. After some experiments, we found that an eye-safe laser is not powerful enough to be reliably visible by a camera in bright sunlight. Finally, Seelinger and Yoder (2005) described another method of vision-based pallet sensing. Again, this work was aimed at an indoor forklift, and it used motion capture fiducials.

Most of the control algorithms discussed in the above-mentioned research into load handling could be applied to our crucible handling task. However, the pallet sensing methods are not suitable due to their low saliency and reliability in an outdoor, industrial setting.

If we consider now the use of computer vision for mobile robotics in outdoor environments, many examples are found in the literature, including state-of-the-art deployments such as the DARPA Grand Challenge (Buehler, 2006; Iagnema & Buehler, 2006) or the LAGR Program (Mulligan & Grudic, 2006). However, most reported work used computer vision for navigation, obstacle avoidance, or lane tracking and not for load identification or load localization: this is the key problem in our application and has not been addressed in a meaningful way before in other similar applications. The field of visual servoing (e.g., Corke, 1996; Espiau, Chaumette, & Rives, 1992; Mezouar & Chaumette, 2002) aims specifically at controlling the interaction between robots and their environment using vision. Most of the control techniques used in this article come from this field, with specific care given to implementability and reliability.

One of the important achievements of our system is the ability to use a pan tilt zoom (PTZ) camera to exhaustively scan a search area while guaranteeing that any part of the area will be seen with enough resolution (Section 4.6). This problem is related to the mosaicing problem (Teller et al., 2001) because creating a mosaic requires acquiring a tiling of images. Compared to our problem, mosaicing typically requires a regular tiling of the sphere around the camera and does not consider the choice of zoom to guarantee observation resolution. Building a covering tiling for a polygon also has attracted interest in computational geometry [see Grünbaum & Shephard (1986) for instance]. Most tiling methods attempt to create an arrangement of regular polygons (the tiles) that will cover an infinite plane without gaps or overlaps between the polygonal tiles. In our context, as we will detail later, a given camera configuration provides a way to observe a polygonal area of the scanned region. In this area, the camera resolution per unit of surface will vary, and only a subset of the area may provide enough resolution to accurately detect our crucible. To exhaustively scan a search region, we have to select a set of camera configurations resulting in a set of observed areas covering the search region with enough resolution at any point. In computational geometry terms we are trying to cover a polygonal region with a minimal selection of irregular polygons minimally overlapping. As a consequence, techniques from the geometrical tiling literature are not readily applicable.

It is also important to note that a number of companies (including Corecon and Omnitech robotics) provide autonomous forklifts for indoor, warehouse environments. Most use laser-based solutions, mainly due to the high reliability of features detected with these sensors. But to the authors' knowledge, no generic, vision-based, outdoor forklift is commercially available today.

3. SYSTEM DESCRIPTION

Our HMC has been automated to the level where it can carry out all the operations of a conventionally operated vehicle with a driver onboard. However, whereas the driver of a conventional HMC is responsible for the efficiency, safety, and sensing for the operations, the autonomous HMC has onboard systems to take this role. Apart from the obvious internal sensors that provide information about the state of the vehicle (e.g., temperature, oil pressure, odometry, hook height, and mast tilt), the vehicle has external environment sensors to assist with navigation, obstacle management, and crucible handling tasks. Four scanning laser range finders (SICK LMS 291[1]) are positioned around the vehicle (Figure 3) and are

[1]For interested readers, we use the LMS 291-S05. This laser scanner provides 0.5-deg resolution over a field of view of 180 deg with an effective range of 30 m with 0.01-m resolution. In addition to the

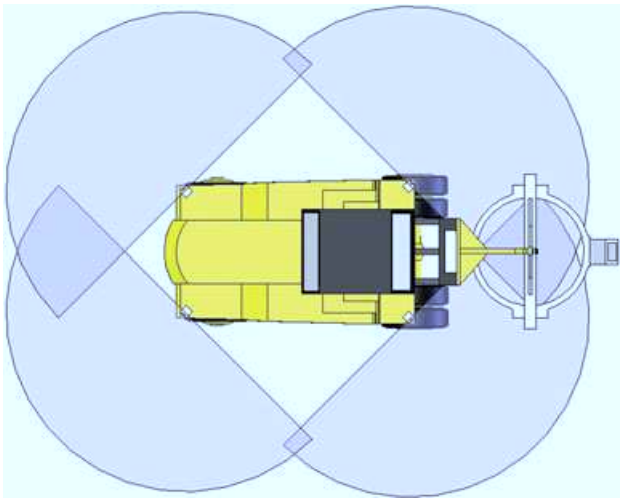


Figure 3. The HMC's lasers are located at each corner of the vehicle, tilted down by approximately 4 deg. They offer overlapping coverage out to approximately 30 m. They are used for localization in the navigation phases and for obstacle avoidance.

tilted down by 4 deg to provide 360 deg of coverage to a distance of approximately 30 m. However, there are still blind spots close to the vehicle (Figure 3). These lasers are used to provide beacon-based localization and obstacle detection [see Section 3.1 and Tews, Pradalier, & Roberts (2007)]. Localization using laser and beacons would be accurate enough to achieve the crucible handling task if only the crucible was guaranteed to be delivered always at the same place, with an accuracy better than 5 cm. As this is not a realistic requirement when crucibles are handled by industrial cranes and swinging hooks, a more flexible method was needed. To this end, a pair of PTZ web-cams (Figure 4) was attached to the mast, to be used as primary sensors for locating the crucible via markers on its handle. The use of computer vision for crucible handling is the main topic of this paper.

The autonomous HMC's safety system consists of a number of physical interlocks, emergency stops (E-stops), obstacle detection, and on- and off-board radio frequency (RF) remote fail-safe and software watchdogs. The E-stops are located around the vehicle, inside and on the portable remote RF device. Activating an E-stop brings the vehicle to a quick halt, and the engine is shut down. Hydraulic controls are frozen at this point. Door interlocks are also included in the E-stop loop to prevent access to the vehicle when it is running autonomously. The software safety systems consist of high-level velocity control when objects are detected close to the vehicle and low-level watchdog checks between interface level software and the low-level control software. A timeout on the watchdog initiates an E-stop.

Figure 5 provides a high-level view of the software and hardware architecture of the autonomous HMC's systems. Low-level components such as throttle, brakes, steering, hook, and mast controls are controlled through programmable logic controllers (PLCs). The critical safety components, such as the E-stop buttons and the watchdog monitor, are controlled through higher grade, fail-safe PLCs. These PLCs provide redundancy checks of relay connections and continuously monitor the input and output state of hardware connections.

The Hardware (H/W) Abstraction program converts the internal vehicle state sensors to human-readable signals and manages the vehicle demands in an opposite manner. High-level programs work directly with the external sensors and vehicle state to control the vehicle. Vehicle-level programs control and monitor the vehicle hardware systems.

3.1. Localization and Navigation

The localization system (Tews et al., 2007) is composed of laser range finders detecting retroreflective beacons placed around the environment. It uses the vehicle's encoder-based odometry as a motion reference but provides better accuracy because odometry suffers from drift and inaccuracies, depending on the tire pressures and load and road surface conditions. A full payload for the HMC weighs approximately 10 t, which distorts the tires and affects odometry readings. In many applications, global positioning system (GPS) is a useful sensor for outdoor-only operations. Differential, wide-area augmentation system (WAAS), and real-time kinematics (RTK) GPS can provide higher accuracy than normal GPS with precisions in the range of 2 cm to several meters. However, GPS accuracy depends on many factors, including visibility of a significant number of satellites in the GPS constellation and a relatively clear path from the GPS and differential base stations to the vehicle's receiver. Around our work site (and in a typical smelter), none of these factors are maintained because the vehicle operates inside and between

range data, the laser provides a reflectivity flag that is nonzero only on retroreflective material.

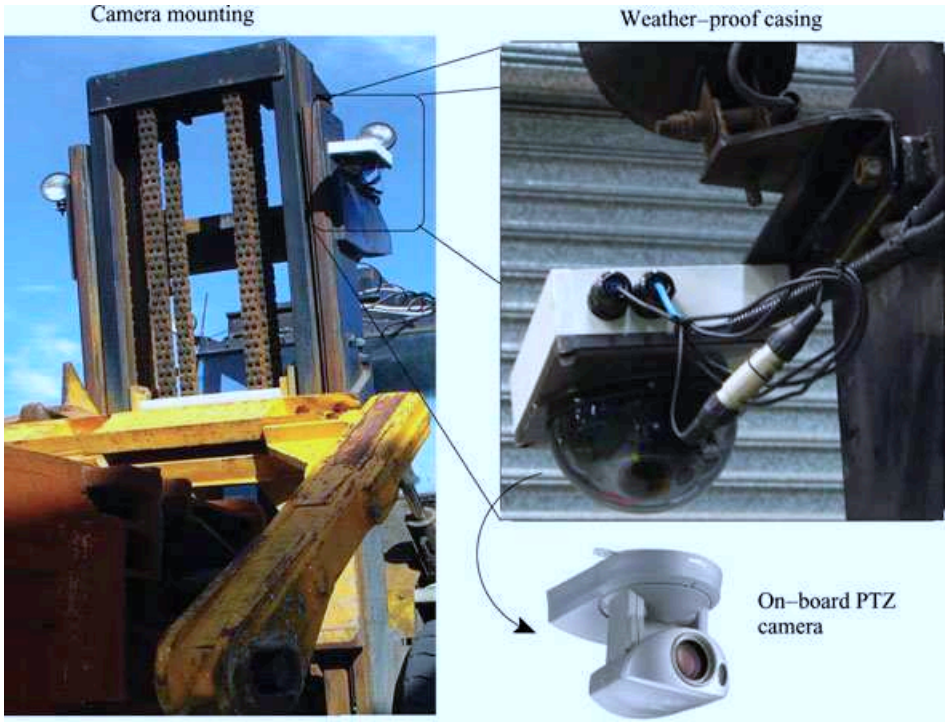


Figure 4. The mast-mounted PTZ web-camera used for locating the crucible. Left: placement on the forklift mast; right: zoomed-in photo of the camera inside its weatherproof casing.

large buildings and around roadways surrounded by tall trees. This results in significant multipathing and signal loss or complete dropout in some areas. Therefore, a local rather than global localization solution is required.

The navigation system uses way points derived automatically by manually driving the required route of operations. Way points are recorded after a certain change in distance since the last way point or a certain change in vehicle heading. Each way point also contains a velocity so that ramping speeds can be utilized for smoother navigation. The resulting way-point list is split into task segments with each segment being a homogeneous action such as a forward traverse (used for normal navigation) or backward traverse (used for crucible manipulation tasks). Within a segment, the navigation system switches to the next way point on the list when it is close to the current way point. The mission program (Section 3.3) handles switching between tasks.

Currently, the obstacle management system is simply a velocity-reduced gradient envelope surrounding the vehicle. As obstacles get closer to the vehicle, the vehicle slows and will eventually stop if they are too close. This behavior can be disabled when operating close to infrastructure such as when entering a narrow doorway. When reversing toward the crucible, the sector surrounding it is blanked so that it will not be considered as an obstacle. These simple rules have worked adequately for our current operations, but we are improving the system to be more flexible to the current environment and task state. For example, the sensitivity of the field for considering an object to inhibit HMC operations will be different when operating in open areas and when it is operating in the confined space of a shed.

3.2. Crucible Operations

The key functionality of a HMC is its ability to handle the crucible. Two main operational phases can be distinguished: crucible pickup and crucible drop-off.

Crucible drop-off Drop-off is an easy maneuver from an automation point of view. No sensing is required, and a simple ballistic maneuver is sufficient

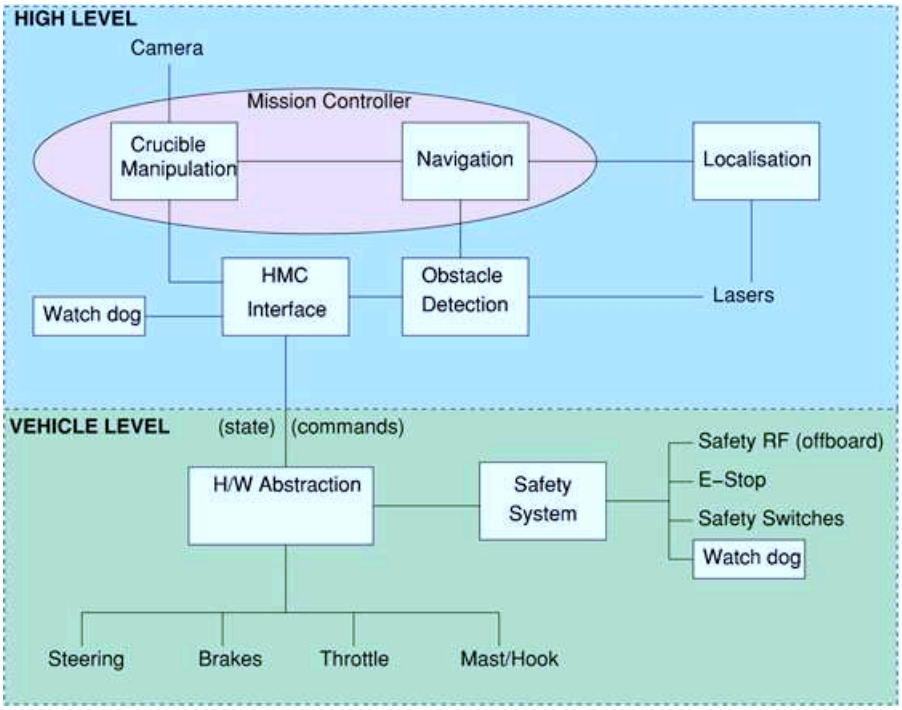


Figure 5. HMC system architecture. The program blocks are shown in boxes or ellipses with leaves representing physical parts of the system.

(Figure 6). The maneuver can be decomposed into three steps: first, lowering the hook so as to lower the crucible on the ground, then moving away from the crucible while lowering the hook to bring the handle to its rest position, and finally moving 15 cm away from the crucible to clear the hook from the crucible.

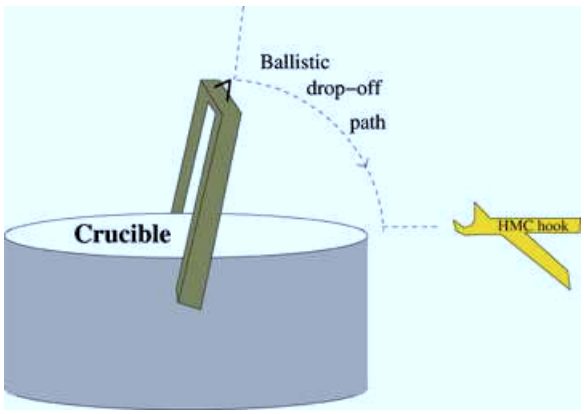


Figure 6. Schematic of the hook movement during a drop-off maneuver.

Crucible pickup The pickup maneuver is more challenging than the drop-off. It can be divided into two steps: first, an approach step in which the hook is visually guided toward the pickup point in the middle of the crucible handle (Figure 7), and then the actual pick up. The latter is an easy maneuver, again a ballistic movement, similar to a drop-off (Figure 6).

Approaching the crucible requires a continuous estimation of the crucible's pose relative to the HMC's hook. An onboard camera provides this information using techniques presented in Section 4. Considering the nonholonomic properties of the vehicle,

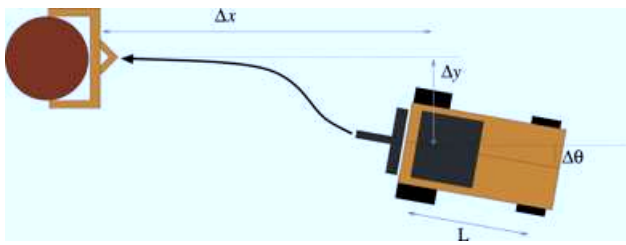


Figure 7. Crucible approach trajectories and variables.

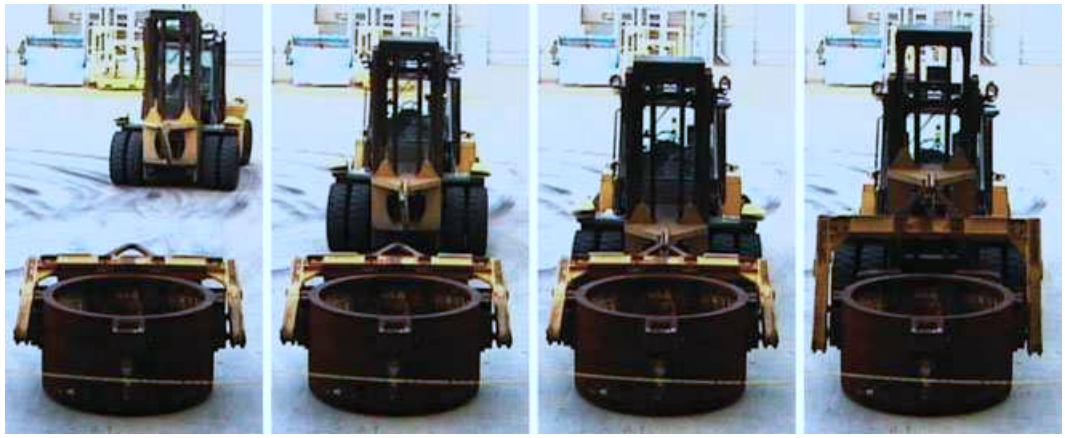


Figure 8. Example of a real approach trajectory.

the HMC servos to the line orthogonal to the crucible, passing through the pickup point (Figure 7).

The approach trajectory is executed at constant speed v (e.g., 0.4 m/s) while the steering of the vehicle is controlled using the standard pure pursuit control law (Hebert, Thorpe, & Stentz, 1997):

$$\dot{\theta} = K_\theta \Delta\theta + K_y v \frac{\sin \Delta\theta}{\Delta\theta} \Delta y, \quad (1)$$

where K_θ and K_y are tunable gains, $\dot{\theta}$ is the rotation speed from which a steering angle can be computed, and other variables are illustrated in Figure 7. Depending on the initial error, this controller may bring the vehicle to the pickup line. This usually requires two conditions: the initial alignment error ($|\Delta y|$ and $|\Delta\theta|$) must not be too big, and the distance to the crucible ($|\Delta x|$) must be sufficient to implement the maneuver. Then the pickup can continue successfully. Otherwise, we need to reliably detect this failure and generate a mitigation strategy. In our implementation, we must ensure that the final position of the vehicle presents a value of $|\Delta y|$ smaller than 0.15 m and an orientation error smaller than 20 deg. These values are arbitrary and adapted empirically to our system. This cannot be considered a reliable failure detection yet because these errors can be the result of camera misalignment or other inconsistencies in the crucible sensing process. We are currently adding sensing modalities to obtain several ways to confirm, with increased dependability, that the hook engagement has been successful.

To illustrate the pickup maneuver, Figure 8 gives an example of an approach trajectory captured during our 5-h continuous trial (Section 5).

3.3. Mission Control and Recovery

The mission controller is responsible for switching between tasks and monitoring their performance. A task may be "drive along a section of road," "drop off the crucible," "start up the engine," or even "blow the horn." Currently a mission is a sequence of tasks with each task returning its status during execution. Once a task has finished, the mission controller selects the next task. Contingencies occurring during task execution cause the mission controller to select the contingency subtask for that task. For example, a missed crucible pickup will trigger a "missed approach" signal and the HMC will move away from the crucible and retry the approach maneuver.

Practically, the mission controller is composed of two parts: a task executer and a task scheduler, which interact as illustrated in Figure 9. The task executer, written in C or C++, contains modular code to implement each registered task. For instance, the "start up the engine" task sends the start-up signal to the low-level interface, waits for the engine revolutions per minute (RPM) to stabilize to a usable level, and terminates with a success signal. If the engine RPM stay null for too long, the task terminates with a failure signal. The task executer is also responsible for receiving task execution requests and handling the initialization and completion of the tasks.

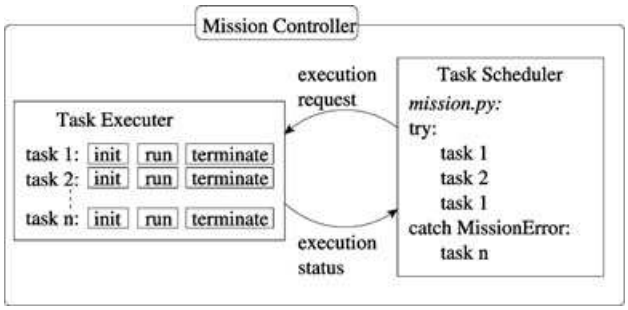


Figure 9. Overview of the mission control architecture.

The task scheduler (written in Python) uses the mission description to schedule each task in sequence. To this end, it sends execution requests to the task executer and waits for termination, raising an exception if the status is an error.

The mission controller is a generic component of our system: only the task implementations are specific to the HMC. For this reason, it is currently used on several of our platforms, including an autonomous submarine (Negre, Pradalier, & Dunbabbin, 2007).

This framework allows for automatic mission planning and even dynamic on-line mission replanning. Nevertheless, these functionalities were not part of the initial requirements for the HMC application. Missions described in Section 5 were hand planned, with built-in contingency plans. Algorithm 1 gives an excerpt of a plan included in a long-duration mission.

Algorithm 1 Example of mission plan, with contingency management

```
# Read current pose
pose.read()
try:
    # First try to servo to recorded
    # position of the crucible
    mi.servoto_mem()
    # Pick up the crucible
    mi.pickup()
except MissionError,me:
    try:
        # If it fails, go back to scan
        # location
        mi.servoto_point(pose.x,pose.y,pose.
        heading)
        # Try a different type of pick up
        # (long range pick up)
        mi.lrpickup(scanregion.x,
        scanregion.y)
    except MissionError,me:
        # If this fails, then ask for human
        # intervention
        print "Cannot find crucible for
        pick up, please pick up manually"
        mi.wait_manual_reset()
```

4. VISION-BASED OPERATIONS

4.1. Challenges and Objectives

To perform a reliable crucible pickup, it is necessary to have a reliable crucible localization system. Several solutions can be envisioned. The more obvious one requires using the vehicle's localization system and a memory of the last crucible drop-off location. Unfortunately, this will fail if crucibles are handled by other vehicles without a localization system such as a crane in smelter operations. It would also be possible to use the laser range finders to detect and localize the crucible from the shape of its hull. This solution would certainly be reliable, but in our current setting, the lasers are located too high on the vehicle to get a good view on the crucible. Changing their position would reduce the efficiency of the vehicle localization and obstacle detection systems.

The crucible localization system we present relies on computer vision. This, in itself, creates challenges and opportunities. The challenges occur because outdoor computer vision is known to be complex due to such difficulties as uncontrolled lighting conditions or unstructured and uncontrolled environments. Implementing an accurate and reliable computer vision solution in these conditions requires a carefully engineered solution. On the other hand, a camera is a much more interesting perceptual medium than a laser range finder. It has a much larger field of view and provides details more densely reported in its field of view.

In summary, this section will describe how we used PTZ cameras to localize the crucible handle, and more specifically the handle eye, in which the HMC hook must be inserted before lifting the crucible. This requires localizing the handle eye within $\pm$10-cm lateral accuracy and $\pm$5-cm longitudinal accuracy. It is important to realize here that the hook can be controlled only vertically from the vehicle (tilting

the mast does not significantly move the tip of the hook) and as a consequence these accuracies can be achieved only by accurately positioning the whole vehicle, which is 6 m long and 2 m wide, with strong nonholonomic constraints.

4.2. Landmark Choice

We chose to maximize the resilience of our system to external perturbation by using artificial fiducials attached to the crucible. These fiducials will be used to provide the position of orientation of the crucible handle. In our application, it is not required to obtain a full 3D position and rotation: as we will detail further on, in our specific setting, the localization problem can be reduced to a planar problem. Also, although obtaining a crucible identification from the fiducials will be useful in future deployment of the system, it was not required for our setup and has not been addressed in the context of this paper.

Most existing vision-based industrial applications use motion capture fiducials, that is, small disks with black-and-white quadrants (Garibotto et al., 1998; Seelinger & Yoder, 2006). These fiducials provide strong corners and may be good enough for an indoor controlled environment, but they are not distinctive enough for reliable outdoor detection.

In the context of industrial computer vision, some more informative landmarks have also been proposed (e.g., ARTag and QRCode) and comparative studies presented: see Claus and Fitzgibbon (2004) or Zhang, Fronz, and Navab (2002) for instance. These landmarks are designed to be very informative; some of them even carry information bits or error correction codes. On the other hand, they are sensitive to occlusion and scratching and we estimated that they would not be detected reliably enough over the intended range of distances.

Instead, we chose to use self-similar landmarks, as described by Briggs et al. (2000) and shown in Figure 10. These have been shown to be very reliably detected, especially in an indoor setting. Formally, the requirements are as follows: a pixel line of the camera must cut the black/white transitions in the image of these fiducials with an angle bigger than 45 deg. Furthermore, the fiducial must appear with a width larger than the detection window in the image, i.e., typically 40 pixels. Once these two conditions are fulfilled, the detection is not sensitive to perspective effects, shearing, or small rotations. Additionally, this landmark is very resilient to lighting condition changes, change of scale, partial occlusion, and scratches. This is especially valuable for a "ruggedized" implementation, as required in a real industrial setting. It is also important to note that the fiducial detector will not detect a mirror image of the landmark.

Nevertheless, in an outdoor environment, we found that the landmark suffers from two main defects. First, it is salient horizontally (or more generally, on a direction perpendicular to the stripes), but the estimation of its vertical position is more sensitive to noise and more uncertain: strong contrasts, JPEG encoding, sensor saturation, or image slant can make the vertical position of the fiducial very unreliable. Then, if we consider an application in an industrial environment, corrugated iron, used to create shed walls, can create a nearly self-similar pattern when observed with enough perspective. In this infrequent case, this can generate large numbers of false positives.

To increase the robustness of our landmark, we have tested two alternatives: an orthogonal landmark made of two self-similar patterns patched together with a 90-deg rotational offset and a circular landmark where the self-similarity is radial instead of being axial. The orthogonal landmark is depicted in Figure 11. The circular landmark is less sensitive to rotation and motion blur, but more sensitive to occlusions than the orthogonal one. Also it is easier to align the orthogonal landmark with an existing structure because its reference point is located along one

Figure 10. Self-similar landmark as described in Briggs, Scharstein, Braziunas, Dima, and Wall (2000).

Figure 11. Orthogonal landmark (right version): The small disc is the reference point of the landmark.

of its edges. Orthogonal landmarks can be made in two versions, with each being easily distinguishable: a *left* version, where the horizontal landmark goes from left to right, on the right of an up-to-down vertical landmark, and a *right* version, resulting from a flip of the other one over the vertical axis. The latter is shown in Figure 11. The possibility to create two easily distinguishable landmarks and the accurate alignment of the orthogonal landmarks were the compelling arguments for their use in this application.

To detect an orthogonal landmark, we use the self-similar landmark detector described in Briggs et al. (2002). This gives the position of the horizontal and vertical landmark. A pair of landmarks can be considered an orthogonal landmark if the distance between them is small and the difference of orientation is close to 90 deg. If these conditions are verified we can estimate the location of the orthogonal landmark reference point (Figure 11) in the image.

As a final note on our landmark design, we implemented our landmarks by having them printed with UV-resistant inks on rigid surfaces. This increases their cost but is critical for items that will spend a major part of their life span outdoors. To date, our landmarks have been on our crucible for 18 months, and they endured this time in Australian subtropical sun and heavy rainfalls without significant performance degradation.

4.3. Landmark Projection

To localize the crucible, we need to detect the fiducials attached to it, and from their location in the image, estimate their position with respect to the HMC's hook. Figure 12 shows the crucible handle and the two fiducials attached to it, as seen from the mast camera.

Let us first define the *target plane* as a plane parallel to the ground passing through the fiducial reference points. We can assume that when the HMC is approaching the crucible, the hook is also located in this plane.[2] Using this definition, and assuming that the ground is locally flat around the crucible, we can reduce the general fiducial localization problem to a simpler localization in the target plane.

Our camera is mounted on a PTZ head attached to the forklift mast. Knowing the camera model, the PTZ parameters, and the pose of the camera with respect to the hook, it is possible to build a one-to-one mapping between image pixels and their projection onto the target plane. In addition, given a point in the target plane, it may be possible to find a unique pan-tilt configuration that makes this point visible in the center of the image frame. Our PTZ head is installed such that this mapping exists for any point of the target plane located between the HMC and the crucible. The PTZ system provides a pan/tilt accuracy of the order of 0.5 deg, resulting in landmark localization accuracy of about 2.5 cm when the crucible is closest. Using our camera with a constant field of view of 42 deg, we were able to detect the visual landmarks until 7 m from the HMC hook point. Misalignment of the pan/tilt encoder can happen (although very seldom), resulting in a considerably reduced system reliability. As a consequence, an alignment checking function would be required. Implementation of this function and mitigation strategies are the subject of ongoing research.

4.4. Crucible Localization and Tracking

4.4.1. Localization

The approach to the handle must be done on a trajectory perpendicular to the handle; consequently, we need to estimate not only the pose of the handle's eye, but also the orientation of the handle bar. To reliably and accurately determine the orientation of the handle, we set up two orthogonal self-similar landmarks (one left and one right).

Using the landmark extractor described by Briggs et al. (2000) and the mapping from the image frame to the target plane, it is straightforward to

[2]The crucible handle is a large object whose height to the ground is known and constant.

Figure 12. Fiducials and their placement on the crucible (as seen from the PTZ camera).

localize the handle. To be robust to multiple possible perturbations, the handle localization is integrated into a particle filter (Doucet, De Freitas, & Gordon, 2001) and fused with movement information coming from the vehicle odometry.

In principle, a particle filter uses a set of samples (particles) to represent a probability distribution resulting for a Bayesian filtering process. Formally, if we let $X_t = (x, y)$ be the state of one of the handle ends at time t, Z_t the observation of a fiducial at time t, and Δ_t^u the vehicle displacement between times t and u, then the goal of the particle filter is to estimate

$$P\left(X_t | Z_{t_1} \ldots Z_{t_n}\ \Delta_{\tau_1}^{\tau_2} \ldots \Delta_{\tau_{m-1}}^{\tau_m}\right).$$

To this end, the filter relies on two models: the observation model $P(Z_t|X_t)$ that predicts a fiducial observation and a displacement model $P(X_{t_k}|X_{t_{k-1}}\ \Delta_{t_{k-1}}^{t_k})$ that predicts the motion of the particles.

It must be noted that any recursive state estimation technique would be suitable for this task, especially a Kalman filter. The choice of a particle filter was motivated by the nonlinearity of the system and by technical aspects such as the availability of the source code from another part of the project and the ease of visualization. The heavier computational requirements of a particle filter were not critical in comparison with the fiducial detection part of the system.

The localization is implemented as follows. We use two instances of a particle filter. Each one uses 200 particles to track the position (x, y), in the target plane, of one side of the handle. From the segment formed by these two estimates, it is then possible to evaluate the orientation of the handle and the position of the hook insertion point, given as the middle of the segment. All the coordinates are expressed in the vehicle frame, as if the vehicle were static and the crucible moving. This representation has been chosen to simplify the expression of the hook control as a servoing to zero.

When a landmark is identified in the image frame, its position is mapped to the target plane and used as an observation to update the particle filter estimate. Formally, this means that Z_t is the projected position (x_f, y_f) in the target plane of a detected fiducial. The observation model used is then

$$P(Z_t = (x_f, y_f)|X_t = (x, y)) \rightsquigarrow \mathcal{G}(X_t, \Sigma_Z),$$

where $\mathcal{G}(\mu, \sigma)$ is the Gaussian distribution of mean μ and covariance σ. Another possible model would be to consider directly the fiducials position in the image as observations. In our implementation, the additional complexity was deemed unnecessary.

In between visual observations, the vehicle odometry is used to infer the motion of the handle in the target plane. Between t_1 and t_2, the vehicle displacement is a translation T_v and a rotation R_v. By changing the reference frame, this can be expressed as a translation $\Delta_{t_1}^{t_2}$ of the observed fiducial. The resulting displacement model is

$$P\left(X_{t_k}|X_{t_{k-1}}\ \Delta_{t_{k-1}}^{t_k}\right) \rightsquigarrow \mathcal{G}\left(X_{t_{k-1}} + \Delta_{t_{k-1}}^{t_k}, \Sigma_\Delta\right).$$

In practice, both fiducial observation and odometry measurements run at different rates: the odometry information is sampled at 20 Hz, whereas the information on landmark localization is generated at 2–3 Hz, when the camera is focusing on the tracked landmark.

In this implementation, we had to take special care in the management of the processing delays, especially when receiving two video streams. Owing to time sharing on the CPU, and to the shear computational cost of the operation, there can be a significant delay, up to 0.5 s, between the availability of an image and the end of the self-similar landmark extraction. It is then necessary to backtrack the particle filter localization estimate, in order to integrate the landmark observation at the time the image was captured.

The output of both particle filters is a probabilistic estimation of the handle ends. Using the a priori knowledge of the handle length and the spread of the particles, we can evaluate whether the estimated handle length is compatible with the reality and wait for this estimation to converge to a consistent value before allowing the HMC to move. It is also possible to use the handle length to constrain the position of one landmark when only the other one is being observed. This type of nonlinear constraint would be the only compelling reason for choosing a tracking implementation based on a particle filter over an extended Kalman filter, given the otherwise low nonlinearity of the system. In practice, we tested the use of this constraint while using a single camera and found it added little accuracy given the low update rate. When using two cameras, we disabled this constraint because it is much less frequent to observe only one landmark.

4.4.2. Visual Tracking

While the HMC is approaching the handle, the camera is controlled in pan, tilt, and zoom to ensure that the landmark stays in the image frame and is visible with enough resolution. This is implemented by controlling the PTZ head so as to put the current position estimate in the center of the image.

To observe both sides of the handle, we make the camera focus for 2 s on each landmark. While the vehicle is far from the handle, both landmarks stay visible together. For the last meter of the approach, the landmarks cannot be seen together because the maximum field of view of the camera becomes too small. Consequently, each landmark can be tracked for only two seconds out of four.

In the current stage of our development, with only one crucible available, we have not developed any specific identification software. Our landmarks are distinctive enough to avoid any false positives and to identify each side of the handle without ambiguity.

Nevertheless, in a real industrial site, where all crucibles would be marked with similar fiducials, the handle identification would have to be dealt with properly. Using geometric constraints in a way similar as in the Joint Compatibility Data Association (Neira & Tardos, 2001) is likely to be successful here. Adding a unique tag to the handle would also help in identifying individual crucibles.

4.5. Multicamera Integration

To improve the accuracy and robustness of our system, we decided to use two PTZ cameras. Using multiple sources of observations integrates naturally in the particle filter framework. Two possible uses of the multicamera/multilandmark system are possible: Either each camera stares at only one landmark, or each camera observes each landmark alternatively.

The advantage of having one camera per landmark is that it requires fewer PTZ movements and consequently increases the efficiency of camera usage. On the other hand, if one of the cameras fails or becomes inaccurate, then the associated landmark is no longer tracked accurately.

To improve robustness, we chose to have both cameras looking alternatively to each landmark. Ideally, we would want to detect that one camera is becoming inconsistent or unreliable, and only then start panning the other camera between landmarks. Future work therefore includes more robust identification and mitigation of camera failures.

4.6. Long-Range Crucible Discovery and Pickup

The approach presented in the preceding sections is adapted to the tracking of a pair of orthogonal self-similar landmarks. It does not address locating the crucible in the environment when only an approximate location is known, for example, it is somewhere within a 20×20 m area. If the area is small enough, the discovery may be achieved by setting the camera at a known fixed position, but in most cases, a scanning strategy must be implemented.

In this section, we address the problem of finding the crucible when the only available knowledge is a range of locations where the crucible may be, given as a 2D polygon, called the *candidate polygon*. As for the crucible orientation, we assume that it is

compatible with a detection by the camera. We consider that the polygon is big enough to prevent finding the crucible with a single PTZ configuration. We also assume that the polygon may extend far enough away from the camera to require that various level of zoom be tried to detect the self-similar landmarks.

With these assumptions, our objective is to select an optimally small set of PTZ configurations guaranteeing that any place in the polygon will be observed with enough image resolution. We refer to this problem as the "viewpoint problem."

4.6.1. Optimal Viewpoint

Consider a point P in the candidate polygon. We can assume that there exists a pan angle and a tilt angle such that P is visible in the center of the image frame. The camera zoom is set such that a hypothetical fiducial located at P, ideally orthogonal to the optical axis, would appear as an object of width w in the image frame. This construction is depicted in Figure 13. This tuple of pan, tilt, and zoom will be defined as the *optimal viewpoint* for P.

One of the limitations of this optimal viewpoint definition is the assumption of orthogonality. In reality, achieving this configuration would require moving either the camera or the fiducial, both infeasible in practice. Nevertheless, we justify this assumption by the fact that in order to be observable by the camera, the fiducial must be close enough from orthogonal to the optical axis (an angle greater than 45 deg is usually challenging for the vision system). Finally this assumption provides a practical way to compute a desired zoom level for the camera to observe the region around P with a good chance of being able to detect any fiducial within.

4.6.2. Covisibility

A point Q is said to be covisible to P if it satisfies two conditions: (i) when P is observed on its optimal viewpoint, Q also appears in the image frame, outside of a w-pixel margin around the image border, and (ii) if a fiducial is located at Q, it appears as an object of width at least $w_p = 0.8\, w$ in the image frame (see Figure 13, right). Obviously the distance to the border and the width of the projected targets are tunable values. Empirically, we use $w = 50$ pixels, and we define the other parameters as a function of w. The width of the projected target (w_p) must be such that a target can be detected if its size is at least w_p.

4.6.3. The Covisibility Graph

To solve the viewpoint selection problem, we first discretize it. The candidate polygon is divided into cells with sides of length similar to the landmark size. In our case we use 0.5×0.5 m cells. For each cell C, the cell center P is used to compute an optimal viewpoint for the cell. Then for all cells D around P, we evaluate whether the center Q of D is covisible to P. This enumeration is done by spiraling around C until no covisible cells are found on a complete revolution.

We define a covisibility graph as a set of nodes and directed edges:

- a node is the center of a cell,
- an edge from node P to node Q indicates that Q is covisible to P.

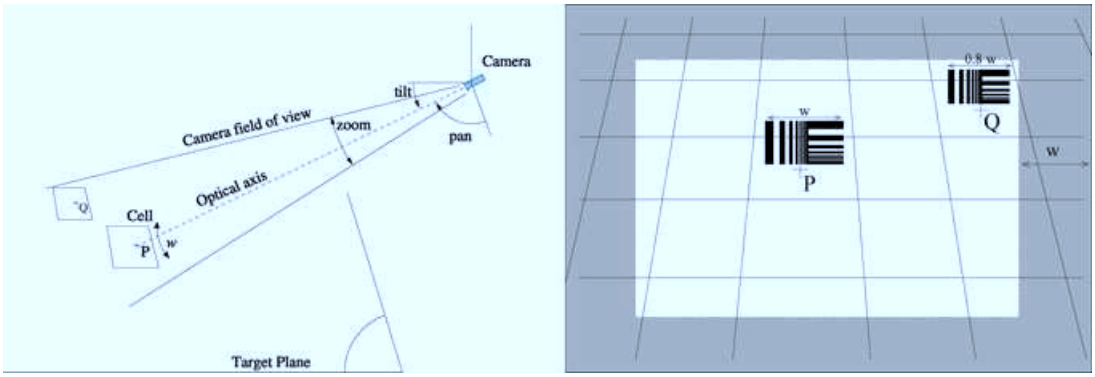
The minimum acceptable distance d_m to the border of the image frame is linked to w_p and also to the apparent size σ of cells in the image frame. If d_m is smaller than σ and σ is of the order of w_p, then there is a possibility that a fiducial located exactly at the border between two cells may not be visible in any of the image frame observing these cells.

4.7. Solving the Viewpoint Problem

Let us call $\mathcal{G}$ the covisibility graph built upon the discretized candidate polygon. Solving the viewpoint problem is finding a minimal set of nodes $\mathcal{S}$ in $\mathcal{G}$, such that for any node v in $\mathcal{G}$, there exists an edge of $\mathcal{G}$ between a node of $\mathcal{S}$ and v. The resulting set of nodes is equivalent to a set of viewpoints by mapping the nodes to their optimal viewpoint.

Algorithm 2 Selection of a set of view points in order to scan a candidate polygon using one camera

```
1:  Let O be an order of the nodes of G.
2:  Let V be the list of nodes of G sorted according to O.
3:  Mark all nodes in V as unvisited.
4:  for all nodes v in V do
5:    if v has been visited then
6:      continue
7:    end if
8:    find a unvisited node w such that v is
      co-visible to w and the size of the set of
      unvisited nodes co-visible to w is maximal.
9:    add w to the selection of view points.
10:   mark all nodes co-visible to w as visited.
11: end for
```



Optimal pan, tilt, and zoom Covisibility in the image frame

Figure 13. Optimal viewpoint selection and covisibility (Q is covisible to P).

This problem can be modeled as the well-known minimum set cover problem (Vazirani, 2004) by associating to each node the set of its neighbors and itself. The minimum set cover problem is known to be *NP hard*.

To solve it in a reasonable time on our system we use the greedy heuristic described in Algorithm 2.

Experimentally, this heuristic can be implemented to run very fast. The complexity of this heuristic is $O(n \log n + e)$, where n is the number of nodes and e the number of edges. This comes from the following reasoning. First, the set of nodes must be ordered, which is at most $O(n \log n)$. Then all nodes are visited once and only once, and for each node v (cf. lines 4–11 in Algorithm 2), finding w (line 8) requires considering all neighbors of v. During this process, all edges of the graph are traversed no more than twice.

When testing this algorithm with various cell-ordering heuristics, we found very little difference in the size of the resulting viewpoint set. Nevertheless, the heuristic is fast enough that we can run it with various cell-ordering heuristics and select the smallest set of viewpoints.

The cell-ordering heuristics we chose to use, without proof of optimality, are the following: the distance to the camera, increasing or decreasing; the covisibility cardinal, increasing or decreasing; the natural order of the cell (lexical order of the coordinates); and random.

Shown in Table I is the size of the resulting viewpoint set for the different cell-ordering heuristic for a square polygon, of 10 × 10 m located 10 m away from the camera. It should be observed that in this example very representative of the real use of the algorithm, the choice of order has little influence on the required number of viewpoints. Experimentally, if we try to move this square candidate area within 15 m left and right and from 5 to 20 m in front of the vehicle, the difference between the best and the worst cell ordering is on average 3 viewpoint only with a standard deviation of 2. This is consistent with what we have observed for any real experiments. Nevertheless, our test showed that none of the ordering heuristics in this table was always best. As a result of the low computational cost of the viewpoint selection heuristic, we can test these orders and pick the best results.

Table I. Influence of cell ordering on the size of the selected viewpoint set.

Order	Plan size
Increasing distance	9
Decreasing distance	8
Increasing covisibility	8
Decreasing covisibility	12
Random order	10
Increasing lexical	8

An example of a resulting viewpoint selection is shown in Figure 14. Each viewpoint is associated with a set of visible cells. The convex hull of this set of cells defines the projected polygon associated to a viewpoint. Owing to the resolution constraint and the candidate polygon discretization, the projected polygon is not a quadrilateral, as would be expected from the homographic projection of the image frame.

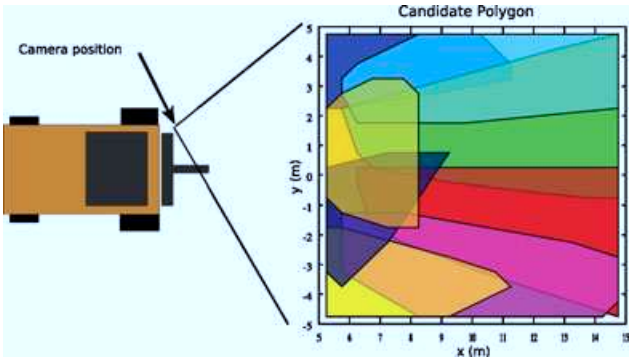


Figure 14. Projection of image frames corresponding to selected viewpoint onto the target plane: each polygon shows which part of the target plane is visible with enough resolution in a given frame.

Figure 14 shows how the candidate polygon is covered by the projected polygons generated from the set of viewpoints. In this figure, each polygon is represented transparently to show how the viewpoints overlap. It is obvious that many cells are observed from at least two viewpoints (Figure 15 shows how many viewpoints observe a given cell), and in this respect, the heuristic appears to be potentially suboptimal. In practice it is fast enough for the size of problem we are considering, especially when implemented with multiple cameras (Section 4.7.2).

It may be argued that only a limited set of candidate polygons need to be scanned and consequently that the corresponding optimal viewpoint sets could be computed off-line once and for all. In reality, the shape of the candidate polygon may be unique, but its position with respect to the vehicle reference point will change for any search, depending on where the vehicle stops to perform the search. Given the accuracy of the vehicle localization and control, it may be valid to ignore these changes and to always perform the same scanning pattern, with the advantage of more repeatability and determinism. On the other hand, computing the search pattern on-line provides a more flexible solution at the cost of very little computation time. This might become useful when dealing with an environment where multiple vehicles handle the crucibles with various degrees of accuracy.

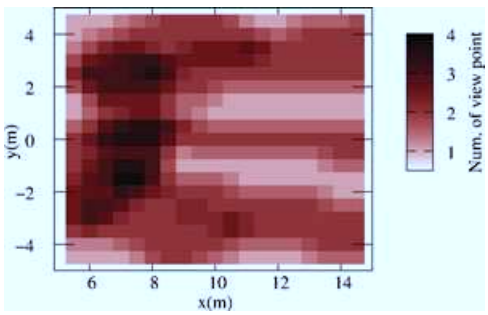


Figure 15. Number of viewpoints (color scale from 0 to 4) observing each cell in the target plane with enough resolution, for the viewpoint selection in Figure 14.

4.7.1. Viewpoint Plan Execution

Once a viewpoint set has been selected it is executed by actually moving the camera to each viewpoint in sequence until both sides of the handle have been observed. At each position, an image is captured and processed to extract orthogonal self-similar landmarks. At this point, it is essential not to have false positives, and fortunately the orthogonal self-similar landmarks are distinctive enough that we have never observed any. The detected landmarks are then fed into the particle filter, as described in Section 4.4.

Once the plan has been completed, the landmark tracker (Section 4.4) is activated and subsequent observations are used to refine the landmark position estimation.

It is essential to be able to evaluate whether the plan execution has effectively been successful in finding the crucible. In our current implementation, in which the crucible is the only object with attached landmarks, this is achieved by simply checking that both extremities of the handle have been observed during the scan.

4.7.2. Multicamera Viewpoint Selection

The approach described previously can be adapted to plan a set of viewpoints for multiple cameras. Several approaches can be envisioned: we can either build an independent plan for each camera or build a plan with a minimally small set of viewpoints shared between the cameras. We can also try to build a plan that can be achieved in minimum time, even if it may have a suboptimal number of viewpoints.

Building an independent plan for each camera This solution provides the most reliability because all places in the candidate polygon will be scanned

twice, once by each camera. It can be implemented by simply running the previous heuristic twice and then executing the scan plans independently. It is also the slowest approach because it takes as long as the longest camera.

Finding an optimally small set of viewpoints using both cameras Algorithm 2 can be adapted to share the set of viewpoints to cover between each camera. It can then find the smallest set of viewpoints of the three approaches proposed in this section, but it does not try to balance the number of viewpoints given to each camera. As a result, if implemented in parallel, the global scanning duration is the scanning duration of the camera with the biggest set of viewpoints to process.

Finding the shortest plan If the cameras are controlled in parallel, the shortest time will be achieved by distributing the viewpoints evenly to the cameras. To implement this, we need to modify the definition of the covisibility graph and adapt the heuristic. An edge in the covisibility graph will now be associated with a camera: a node Q is covisible to P using camera $\mathcal{C}$ if Q is visible when P is observed by $\mathcal{C}$ on its optimal viewpoint. The resulting heuristic is described in Algorithm 3.

Algorithm 3 Finding a view point selection to scan a candidate polygon with several cameras

1: Let $\mathcal{O}$ be an order of the nodes of $\mathcal{G}$.
2: Let $\mathcal{V}$ be the list of node of $\mathcal{G}$ sorted according to $\mathcal{O}$.
3: Mark all node in $\mathcal{V}$ as unvisited.
4: $\mathcal{C}$:= first camera.
5: **for all** node v in $\mathcal{V}$ **do**
6: **if** v has been visited **then**
7: continue
8: **end if**
9: $\mathcal{C}$:= next camera that can see v
10: find an unvisited node w such that v is co-visible to w using $\mathcal{C}$ and the size of the set of unvisited nodes co-visible to w using $\mathcal{C}$ is maximal.
11: add the pair $(w, \mathcal{C})$ to the selection of view point.
12: mark all nodes co-visible to w using $\mathcal{C}$ as visited.
13: **end for**

Figure 16 shows a typical result obtained using this approach and two cameras. The final number of viewpoints is still eight, but it can be appreciated that each camera will deal with four viewpoints. A parallel implementation will have scanned the candidate polygon in only four camera movements.

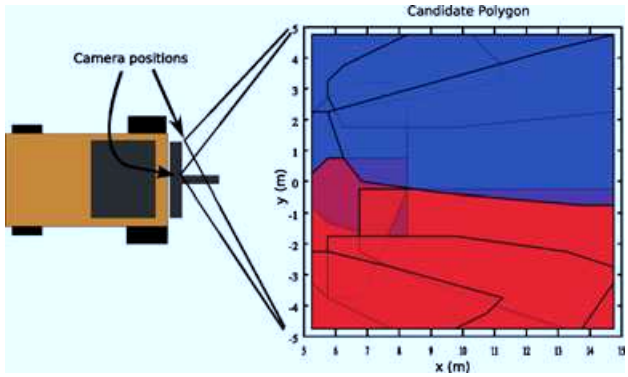


Figure 16. Projection of image frames corresponding to selected viewpoint onto the target plane: optimal viewpoint for two cameras while trying to balance the camera use, blue polygons correspond to the area viewed by camera 1; red ones are viewed by camera 2.

4.7.3. Discussion

We have identified two aspects that need refinement in the future: First, similar to the approach tracker, the long-range crucible detector needs to include geometric information (e.g., the known distance between landmarks) to validate the detections. This would also help in the development of better identification of the nondetection cases. Second, when the plan execution fails to find the crucible, some actions are required to try to find it. The definition of mitigation strategies is a problem that is difficult to address in a reliable and robust manner, but we consider this and the previous points as critical future work directions.

As discussed previously (Section 4.5), to take advantage of multiple cameras, proper management of camera failures should be implemented. If one camera failed, the current viewpoint selection should be aborted and a new plan for the remaining camera should be evaluated. Alternatively, the new plan may cover only the part of the candidate polygon that is still to be scanned. This strategy has not been evaluated yet on our system.

5. SIGNIFICANT EXPERIMENTS

Repeatability and reliability are paramount to using an autonomous vehicle for continuous operations.

We have successfully conducted numerous scheduled and unscheduled demonstrations of the autonomous HMC. We have also scheduled demonstrations of capability over prolonged periods toward creating dependable operations. The autonomous HMC has been operational for more than 1 year, and we have now conducted more than 200 h of autonomous missions. In this section, results from the two most significant demonstrations are presented. These consist of 5- and 2-h experiments in which the vehicle was completely independent of human control except for a safety supervisor. The crucible was filled with 3 t of solid concrete to simulate a partial payload. We determined that a partial load was sufficient for our demonstrations and is less stressful on the vehicle than a full payload.

5.1. Setup and Goals

The goals of the 5-h experiment were to demonstrate short-distance, highly repeatable navigation and crucible operations. It also allowed us to examine the effects of continuous operations on the accuracy of navigation and crucible operations. Vehicles operating over long time periods will have subtle changes in their dynamics; for example, as the vehicle and computer hardware get hotter, the response times to commands can change. The vehicle operations were conducted in a large constrained concrete area (inside the blue box in Figure 17) running a basic continuous mission of navigation and crucible operations. All crucible detections were conducted by a single camera.

Figure 17. The path (yellow) of the 2-h experiment. The crucible pickup and drop-off occurred in the open area at the end of the path on the left, and the in-shed operations were conducted in the large shed in the upper right. The 5-h experiment was conducted in the large area surrounded by buildings indicated by the blue box.

Table II. Key statistics from the 5- and 2-h experiments.

Experiment	Total dist.	Cycle dist.	Velocity range	Cruc. ops.
5 h	8.5 km	0.3 km	−1.1:1.6 m/s	58
2 h	6.5 km	0.93 km	−1.4:3.0 m/s	14

The goals of the 2-h experiment were to demonstrate operations over a longer distance in various types of environment with in-shed crucible operations. Figure 17 shows the vehicle path (yellow) for the 2-h experiment on an aerial map of the environment. Note that the path traverses buildings and a narrow roadway surrounded by bushland. The mission was set up for a complete set of vehicle operations, including start-up and shutdown, and the challenge of having to initially find the crucible in the environment because its precise location was not known. There were two locations for crucible operations: an open area and the confined space of a storage shed. The difference with the 5-h experiment was that it used two cameras for all crucible operations. This is mostly because the relevant hardware and software were developed and installed during the 6 months between the two experiments.

5.2. Results

Table 2 summarizes the main statistics from the experiments. The Total dist. is how far the HMC traveled during the experiment. The Cycle dist. is the distance traveled in each iteration. The Velocity range is the maximum and minimum velocity the vehicle traveled, and the Cruc. ops. field is the total number of times the crucible was picked up and dropped off.

5.2.1. Reliability

In each experiment, only a single intervention was required by the supervisor. For the 5-h experiment, the battery in the supervisor's safety remote went flat after approximately 4 h. This caused an E-stop to be triggered on the vehicle, which brought it to an immediate halt, and it shut down. The battery was replaced and the vehicle restarted. At the time of the stoppage, it was about to pick up the crucible, and after the restart, it executed a "missed approach" recovery procedure and continued uninterrupted operations until the end of the experiment. During the

2-h experiment, the vehicle failed once to pick up the crucible. This was mainly due to the vehicle performance changing as it got warmer (the day was particularly hot). The rising temperatures affected the control limits of the vehicle, causing the brakes to be more responsive and the engine to rev higher during a pickup. An increase in engine RPM was used in the criterion to determine contact with the crucible, which triggered prematurely in this case. A manual pickup was performed, and the vehicle continued its autonomous mission. This problem did not manifest again throughout the remainder of the mission. The failure highlights how even though systematic failures can be anticipated, there are also the changes in the vehicle's physical performance due to environment conditions to consider. Since this experiment, we have changed the pickup to not rely on engine revolutions.

5.2.2. Navigation

A thorough analysis of the navigation system is beyond the scope of this paper, and empirical evidence is provided instead. In both experiments, the navigation system (described in Section 3.1) was accurate and reliable from the perspective of vehicle performance and an empirical postanalysis of the recorded vehicle tracks. In the 5-h experiment, the HMC was at most 15 cm away from its nominal path, which highlighted the navigation consistency and accuracy.

In the 2-h experiment, there were several areas where highly accurate navigation was required: driving into and out of the HMC's parking location, traveling along the narrow back road, and entering the shed where the crucible was being dropped off. In the HMC's parking spot, there is a raised maintenance pit over which it drives. The clearance on either side of the pit to the inside of the wheels is approximately 20 cm. Along the back road, there is only 0.5-m clearance on either side of the road where either soft ground or drop-offs occur. Although this seems like ample clearance, it represents the longest part of a cycle at the highest velocity, so any deviations in vehicle control can result in a significant displacement of the vehicle on the path. Operations in the shed where the crucible was dropped off and picked up had to be within 10 cm because the entry to the shed is quite narrow, as can be seen in Figure 18. Once inside the shed, there is infrastructure on either side of the vehicle, which meant there was very little room for maneuvering (Figure 19).

Figure 18. The doorway into the storage shed was narrow for the HMC to enter. There was approximately 30-cm clearance on either side.

Figure 19. In the storage shed, space was limited for HMC operations.

5.2.3. Obstacle Management

The vehicle's obstacle management system raised some interesting issues. In normal operations, HMCs go close to infrastructure and contact the crucible. They also travel along long roads where no obstacles should be present. This requires a multivariate

strategy for determining what constitutes an obstacle, how to handle obstacles, and what ranges should be considered for obstacle detection. In our experience, the answers to these questions are primarily site and area specific and secondarily vehicle and task specific. For instance, at our site, if an obstacle is detected along the long back road, the vehicle needs to halt because it cannot avoid it, whereas in the open compound area, it may be able to choose a different path around a detected obstacle. Also, obstacle detection needs to be smart during a crucible pickup because the vehicle needs to contact the crucible. Our current simple strategy is to have a different obstacle detection profile for when the crucible is on or off the vehicle, and to turn off obstacle detection when entering an area where the vehicle will go close to infrastructure, such as when entering the storage shed. We are in the process of enhancing the obstacle management system to address the issues mentioned above.

5.2.4. Crucible Pickup

Five-hour experiment The reliability and predictability of the HMC autonomous maneuver is critical to its acceptance by industry. As an illustration of the repeatability of our current implementation, the vision-based pickup was used for all pickups in our 5-h experiment. Figure 20 shows the superimposition of half of the 58 pickups performed during this time; the other half happening at the other end of the path could not be displayed in this picture. Each line on this figure shows the path of the tip of the hook during a pickup maneuver. These paths were acquired by hand marking the position of the hook on a video sequence captured by a static camera. At the same time, for each pickup, the position of the crucible in the image was recorded, also by hand marking it. This specific mission being very repetitive, we expect the crucible to be always dropped in the same location. In Figure 20, the dots overlaid on the crucible handle show the recorded crucible positions and it can be seen that they are contained in a small ellipse of approximately 10-cm radius. It can also be clearly seen that the paths are well contained in an envelope whose width is correlated with the ellipse size. Apart from this analysis, it is difficult to analyze the performance of the crucible pickup operation, because it is either successful or it fails.

Regarding the lighting conditions, this experiment started at noon and lasted until 5 p.m., close to sunset at this time of the year in Queensland, Australia. The weather was fine with passing clouds (cumulus), and the light condition ranged from bright midday sunlight to reddish sunset illumination. Shadows were sharp under the sunlight and very smooth under the clouds. For all the pickups, both fiducials on the handle had the same lighting conditions. If one fiducial was in the shade and the other in bright sunlight, it is likely that due to the limited dynamic range of the camera, the fiducials may not be detected successfully.

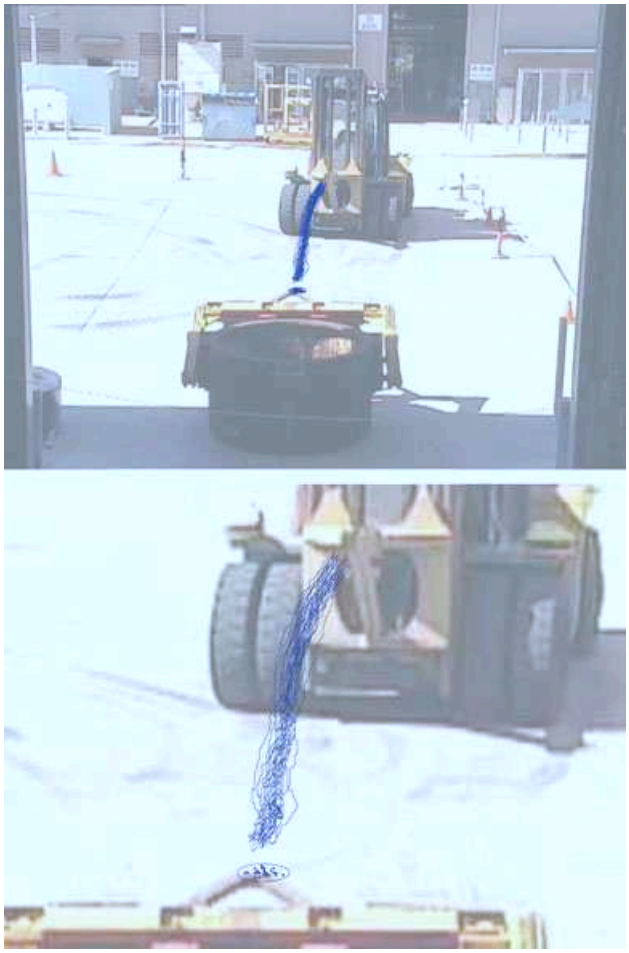


Figure 20. Pickup approach trajectories: superimposition of 29 pickups. The dots and the ellipse around the crucible handle show the variability of the crucible location across this set of maneuvers. The top figure shows an overview of the experiment area, and the bottom figure shows a close-up on the paths themselves. For reference, the handle width is about 2 m and the pickup hole approximately 20 cm across.

For the final pickup, the vehicle had to pick up the crucible outside of a shed where the lights had been turned on. The camera was not capable of handling this dynamic range, and we had to turn the shed light off to let the vehicle find the crucible and pick it up. A more advanced autonomous system would have solved the problem by turning its own lights on, but unfortunately the light control had not been automated in our system.

Two-hour experiment In the 2-h experiment, three types of pickup were performed: long-range camera based, short-range camera based, and known location navigation based. When picking up the crucible based on its known location, the difference between the current vehicle location and the crucible location is used as an input to the control law sketched in Section 3.2. As a consequence, no vision detection is performed and the overall pickup can be done much faster, but this solution would not be robust enough in a situation in which various vehicles are handling the crucibles.

In most of the crucible operations, the location of the crucible is recorded on every drop-off and for our experiments, the crucible is not perturbed from drop-off to pickup, so this location can also be used for the subsequent pickup. Nevertheless, this solution was used in only half of the pickups in this experiment because the drop-off point was in a tight corridor, and there was not enough maneuvering space to allow a vision-based pickup.

For the long-range camera-based pickup, the initial crucible detection was conducted without the location of the crucible being known. Instead, a 20×20 m area was defined to contain the crucible in the environment. The two onboard cameras applied the technique described in Section 4.6 to successfully find the crucible, and then the standard visual servoing pickup was undertaken. This search for the crucible was required only for the first pickup. In the following vision-based pickups, the recorded position of the crucible was used as a hint as to where the camera should initially be pointed. This assumed that the various sources of uncertainty would not be enough to prevent the crucible from being visible in this initial image frame.

The short-range camera-based and known location navigation-based pickups were successful for the duration of the 2-h experiment. Half of the pickups were performed using the vision-based method, and the other half based on the known location of the crucible. The navigation pickup was of particular interest because it demonstrated the accuracy of the localization and vehicle control systems. All the pickup and drop-off locations were inside an ellipsis less than 15-cm in radius, as can be seen in Figure 21.

6. FAILURE MODE ANALYSIS

As is often the case in complex autonomous systems, there are various sources of failure, with some being detectable while others are not. In this section, we list some of the possible failures of the different sub-systems, identifying their properties and solutions.

6.1. Navigation

Vehicle hardware and interface It is possible that components of the vehicle hardware may fail during an autonomous run. Some of these events can be detected, but there may be no solution except for aborting the mission. For example, in one particular situation, a main pneumatic hose was damaged. This can be detected by monitoring the system pressure, but no corrective action can be performed by the autonomous system because, as a redundant safety system, the vehicle is designed so that a lack of pressure engages the brakes. Other hardware failures such as alternator shutdown and battery failure can also be detected with the onboard sensing, but they cannot be corrected.

Hardware interfaces (green part in Figure 5) can fail in three ways: stop provide data, provide incorrect data, or implement an incorrect control. We have currently implemented solutions to the first case, in which the mission will be aborted, but have yet to do so for the second and third. Failure detection based on data consistency is not expected to be difficult, but this is a subject for further research.

Localization The localization system relies on the laser range finder and on correct identification of the reflective beacons. In case one laser range finder fails, we can continue to localize reliably with the three remaining. For localization, only two working lasers are required to have sufficient field of view in our environment. In some cases, the laser scanner failure can be solved by power-cycling them. We have installed the hardware to implement this kind of remediation, but we have not tested it at the time of this writing.

Obstacle detection Similarly to the localization subsystem, the obstacle detection subsystem relies on the

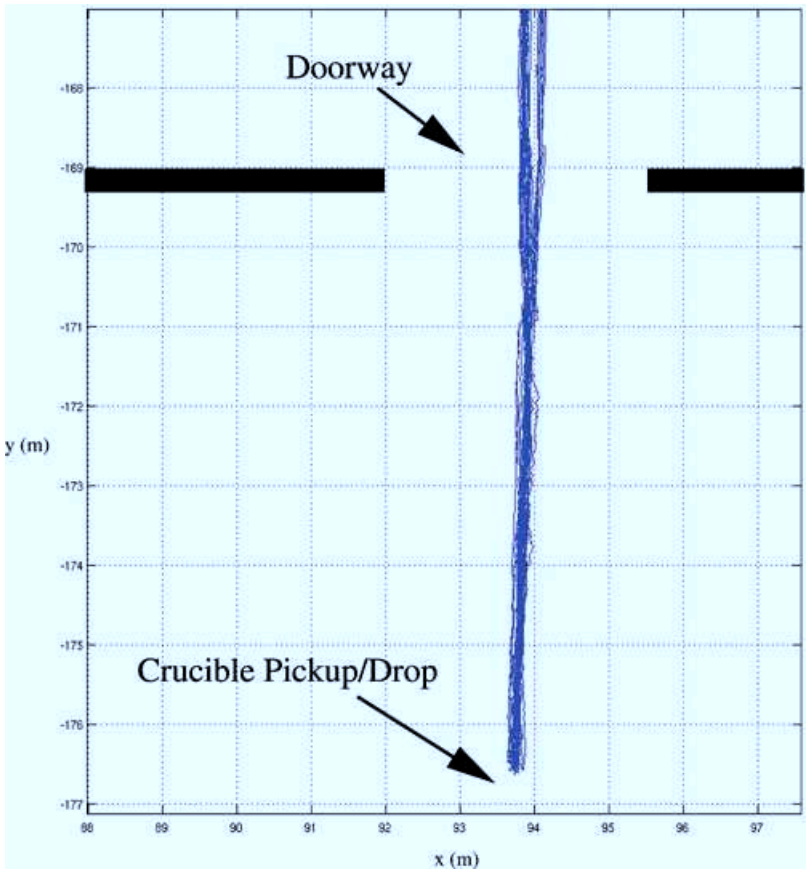


Figure 21. The 28 traverses into and out of the storage shed during the 2-h experiment. These paths are contained in an envelope whose radius is less than 15 cm.

laser scanners. If one of them fails, then the vehicle field of view is reduced to less than 360 deg and this subsystem cannot be used reliably anymore. As a consequence, the collision avoidance system will stop the vehicle.

Control The navigation subsystem is responsible for steering the vehicle along its paths. It can fail in three main cases: if the path are badly designed, if the localization fails, or if the vehicle hardware fails. The two latter cases have been discussed above, so we will discuss only the case of badly designed paths. To be tracked reliably, paths must be designed with a bounded curvature. If the path curvature is not feasible for the vehicle, the path tracking control law will fail to stabilize the vehicle on the path. This can be detected by monitoring the tracking error. In practice, we avoid this problem by defining only feasible paths or learning the path from the logs of a human driver.

6.2. Crucible Handling

Pickup and drop-off The implementation of our crucible pickup and drop-off are ballistic movements. As a consequence, they can fail if the hook and mast controls are behaving unexpectedly or if the hook position sensing is failing. Both cases cannot always be detected. We have implemented our state machines with timeouts in order to catch some of the control failure, but we are aware that this is only a partial solution. Redundant and multimodal sensing would certainly be part of a more robust solution to this subsystem.

Crucible visual detection and tracking Being one of the most complex components of the system, the crucible detection subsystem is the most prone to failure. It relies on the mast control to be accurate, the pan-tilt-zoom control to be working correctly, the camera sensors to be exporting data at a high enough frame rate, the crucible fiducials to be set up at the correct

position and visible, the vehicle to report accurate odometry, and most important, the vision algorithms to be able to detect the fiducials.

Because the hardware failure modes have already been discussed, the main issues affecting the vision system are listed below:

- The PTZ control of the camera relies on an accurate calibration of its internal encoders. This may drift after a few weeks of inactivity or a period of brown power. We handle this case by a specific daily start-up procedure that checks that the control is accurate.
- If the camera frame rate slows, this can be detected easily but not corrected by the autonomous system.
- The fiducials have been attached to the crucible with industrial-strength tape. In our setting this provides secure positioning on the crucible and has been in use for more than 2 years of outdoor exposure. On a real crucible, which can be heated close to 700°C, heat-resistant paint would be used.
- To ensure that the fiducials are visible, we design our missions and vehicle paths appropriately, and we assume that the crucible is not moved significantly between drop-off and pickup.
- The fiducial detection algorithm can fail in various ways: when the sun (particularly at sunset) is facing the camera, the whole environment is too bright (particularly around midday on a light-colored concrete patch), or if it is too dark. Currently, there is no solution to these problems, but we can ensure that the vehicle goes into a safe failure state when the fiducials cannot be detected. Several remedial actions are foreseen but not implemented: the vehicle could try to switch its headlights on, or we could try some smart control of the camera exposure.
- The internal calibration of the camera also plays an important role in the crucible localization system. We currently rely on this calibration to stay constant, and we check its validity only in our start-up procedure.
- If the fiducials are too far away from the camera, the resolution at which they appear in the image can be insufficient to detect them properly. There are two ways to deal with this problem. The simple and reliable one is to ensure that the vehicle is close enough before initiating the visual servoing maneuver. It could also be possible to use the camera zoom as in the long-range search (Section 4.6), but changing the zoom is a very slow control and this severely reduces the field of view.
- When the system is very close to the crucible, the fiducials may appear highly slanted, which makes their detection challenging (due to the nature of the detection algorithm). A circular self-similar landmark would help in this case. Another possible problem at short range is the very important angular distance between the two sides of the handle. This makes controlling the pan and tilt from one fiducial to the other a long operation, during which the fiducials are not tracked. Unfortunately, this is the time when we need the most accuracy in the tracking, because the hook must be raised as soon as it is engaged in the handle. To add some redundancy, and to be less sensitive to localization delays, we have added a contact sensor on the hook. This sensor guarantees the correct engagement of the handle.

If any of the above failures occur during a visual servoing to the crucible, the autonomous solution is to retry the maneuver, possibly using the long-range search to ensure that the crucible localization is found.

Localization-based crucible engagement When the crucible is dropped off, its position is recorded and can be used to make a quick pickup maneuver without using the vision system. This approach can fail if the laser scanners fail, if the vehicle control hardware fails, or if the localization system is incorrect either at drop-off time or at engagement time. Hardware failures have already been discussed above. One way to detect that this engagement has failed is to check that a contact has been detected no more than 20 cm away from the prerecorded position. In case of failure, corrective actions can be achieved only at mission level: the vehicle can move back to a safe scanning position and start the vision system to detect and track the crucible.

6.3. Summary

As can be seen above, there are many ways our system can fail. To deal with some of them, we

have introduced exception management in the mission control system. Unfortunately, only a few failure modes can be reliably detected, and even fewer can lead to a smart recovery action: stopping the vehicle and aborting the mission or calling for help cannot be considered smart. The only failure modes we can currently handle reliably and autonomously are the cases in which the localization-based crucible engagement or the visual crucible engagement fails.

7. CONCLUSIONS AND FUTURE WORK

We have shown in this paper that it is possible to fully automate a large industrial materials handling operation: that of hot metal movement in an aluminum smelter. At our work site, we have fully automated a HMC and have demonstrated typical operations of a production vehicle. Our vehicle is capable of autonomous start-up, shutdown, navigation, obstacle management, and crucible pickup and drop-off. The work reported here is focused on the use of vision as the primary sensor for detecting the load, a 2-t crucible containing an 8-t load (normally molten aluminum, but 3 t of concrete deadweight in our case).

We have addressed the challenge of using a vision system outdoors, in ever-changing lighting and weather conditions, by using artificial visual landmarks (fiducials) mounted on the handle of the crucible. A novel landmark was created using two orthogonal self-similar landmarks that proved to be robust in the outdoor environment.

First, the problem of accurately locating and tracking the crucible handle so that the HMC could be visually servoed to pick it up was tackled using a particle filter–based tracker that also used vehicle odometry data. This method was further enhanced with the development of a multi-PTZ camera system that increased the accuracy and robustness of the close-range approach tracker.

Second, the problem of long-range crucible detection was addressed. A technique was developed that used the same PTZ cameras as used for the approach tracker to locate the crucible at a range of 20 m and where only an approximate estimate of the crucible's location was available.

Our system has conducted more than 200 h of autonomous operations and demonstrated long periods of high reliability and repeatability. In this paper, we have presented results from a 5-h-duration, fully autonomous run. This experiment is a significant achievement in field robotics as it is one of the first long-duration autonomous demonstrations that has a challenging vision-sensed manipulation task every few minutes. During the 5-h run, the crucible was handled 58 times and the vehicle traveled nearly 8.5 km. The route distance for this experiment was relatively short at 0.3 km, and so another experiment with a longer route was undertaken. This experiment lasted for 2 h with a route distance of 0.93 km and a total mission distance of 6.5 km.

Future work on this project includes the testing of the vision system in the dark (using lights on the HMC), the development of a more sophisticated obstacle detection system that builds 3D maps of the terrain ahead of the vehicle, and the development of a robust personnel safety system to automatically detect people nearby. The ultimate goal of the project is to deploy and test the system at an operational aluminum smelter.

ACKNOWLEDGMENTS

This work was funded in part by CSIRO's Light Metals Flagship project and by the CSIRO ICT Centre's ROVER and Dependable Field Robotics projects. The authors gratefully acknowledge the contribution of the rest of the Autonomous Systems Lab's team and, in particular, Polly Alexander, Stephen Brosnan, Matthew Dunbabin, Paul Flick, Leslie Overs, Pavan Sikka, Kane Usher, John Whitham, and Graeme Winstanley, who have all contributed to the project. We also acknowledge the contribution of the CSIRO Exploration and Mining Automation team, in particular, Brendon Stichbury, Zak Jecny, and Andrew Castleden, who assisted in converting our HMC to a drive-by-wire vehicle.

REFERENCES

Briggs, A., Scharstein, D., Braziunas, D., Dima, C., & Wall, P. (2000). Mobile robot navigation using self-similar landmarks (pp. 1428–1434). In Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA. New York: IEEE.

Buehler, M. (2006). Summary of dgc 2005 results. Journal of Field Robotics, 23(8–9), 465–466.

Claus, D., and Fitzgibbon, A. (2004). Reliable fiducial detection in natural scenes (volume 3024, pp. 469–480). In Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic. New York: Springer-Verlag.

Corke, P. (1996). Visual control of robots: High performance visual servoing. Baldock, UK: Research Studies Press.

Doucet, A., De Freitas, N., & Gordon, N. (Eds.). (2001). Sequential Monte Carlo methods in practice. New York: Springer-Verlag.

Espiau, B., Chaumette, F., & Rives, P. (1992). A new approach to visual servoing in robotics. IEEE Transactions on Robotics and Automation, 8(3), 313–326.

Garibotto, G., Masciangelo, S., Bassino, P., Coelho, C., Pavan, A., Marson, M., & Elsag Bailey, G. (1998). Industrial exploitation of computer vision in logistic automation:autonomous control of an intelligent forklift truck (volume 2, pp. 1459–1464). In Proceedings of the IEEE International Conference on Robotics and Automation, Leuven, Belgium. New York: IEEE.

Garibotto, G., Masciangelo, S., Ilic, M., & Basino, P. (1996). Robolift: A vision guided autonomous fork-lift for pallet handling (pp. 656–663). In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Osaka, Japan. New York: IEEE.

Garibotto, G., Masciangelo, S., Ilic, M., & Bassino, P. (1997). Service robotics in logistic automation: Robolift: Vision based autonomous navigation of a conventional fork-lift for pallet handling (pp. 781–786). In Proceedings of the 8th International Conference on Advanced Robotics, Monterey, CA. New York: IEEE.

Grünbaum, B., & Shephard, G. (1986). Tilings and patterns. New York: W. H. Freeman.

Hasunuma, H., Nakashima, K., Kobayashi, M., Mifune, F., Yanagihara, Y., Ueno, T., Ohya, K., & Yokoi, K. (2003). A tele-operated humanoid robot drives a backhoe (pp. 2998–3004). In Proceedings of the IEEE International Conference on Robotics and Automation, Taipei, Taiwan. New York: IEEE.

Hebert, M., Thorpe, C., & Stentz, A. (1997). Intelligent unmanned ground vehicles: Autonomous navigation research at Carnegie Mellon. Norwell, MA: Kluwer Academic.

Iagnema, K., & Buehler, M. (2006). Editorial for Journal of Field Robotics—Special issue on DARPA grand challenge. Journal of Field Robotics, 23(8–9), 461–462.

Mezouar, Y., & Chaumette, F. (2002). Path planning for robust image-based control. IEEE Transactions on Robotics and Automation, 18(4), 534–549.

Mora, M., Suesta, V., Armesto, L., & Tornero, J. (2003). Factory management and transport automation (pp. 508–515). In Proceedings of Emerging Technologies and Factory Automation. New York: IEEE.

Mulligan, J., & Grudic, G. (2006). Editorial for Journal of Field Robotics—Special issue on machine learning based robotics in unstructured environments. Journal of Field Robotics, 23(11–12), 943–944.

Negre, A., Pradalier, C., & Dunbabbin, M. (2007). Robust vision-based underwater target identification & homing using self-similar landmarks. In Proceedings of International Conference on Field and Service Robotics, Chamony, France. Berlin: Springer-Verlag.

Neira, J., & Tardos, J. (2001). Data association in stochastic mapping using the joint compatibility test. IEEE Transactions on Robotics and Automation, 17(6), 890–897.

Nelmes, G. (2006). Container port automation (pp. 3–8). In Proceedings of Field & Service Robotics. New York: Springer.

Nygards, J., Hogstrom, T., & Wernersson, A. (2000). Docking to pallets with feedback from a sheet-of-light range camera (volume 3, pp. 1853–1859). In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Takamatsu, Japan. New York: IEEE.

Roberts, J., Duff, E., & Corke, P. (2002). Reactive navigation and opportunistic localization for autonomous underground mining vehicles. International Journal of Information Sciences, 145, 127–146.

Roberts, J., Duff, E., Corke, P., Sikka, P., Winstanley, G., & Cunningham, J. (2000). Autonomous control of underground mining vehicles using reactive navigation (pp. 3790–3795). In Proceedings of IEEE International Conference on Robotics and Automation, San Francisco. New York: IEEE.

Seelinger, M., & Yoder, J.-D. (2005). Automatic pallet engagment by a vision guided forklift (pp. 4068–4073). In Proceedings of the IEEE International Conference on Robotics and Automation, Barcelona, Spain. New York: IEEE.

Seelinger, M., & Yoder, J.-D. (2006). Automatic visual guidance of a forklift engaging a pallet. Robotics and Autonomous Systems, 54(12), 1026–1038.

Teller, S., Antone, M., Bodnar, Z., Bosse, M., Coorg, S., Jethwa, M., and Master, N. (2001). Calibrated, registered images of an extended urban area (pp. 813–820). In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii. New York: IEEE.

Tews, A., Pradalier, C., & Roberts, J. (2007). Autonomous hot metal carrier (pp. 1176–1182). In Proceedings of the IEEE International Conference on Robotics and Automation. New York: IEEE.

Vazirani, V. V. (2004). Approximation algorithms. New York: Springer.

Zhang, X., Fronz, S., & Navab, N. (2002). Visual marker detection and decoding in ar systems: A comparative study (pp. 97–106). In Proceeding of International Symposium on Mixed and Augmented Reality. New York: IEEE.

Appendix F

Pradalier and Usher [2008]

Cédric Pradalier and Kane Usher. Robust trajectory tracking for a reversing tractor trailer. *Journal of Field Robotics*, 25(6-7):378–399, 2008

Robust Trajectory Tracking for a Reversing Tractor Trailer

Cédric Pradalier and Kane Usher
Autonomous Systems Laboratory
CSIRO ICT Centre
P.O. Box 883
Kenmore, QLD 4069, Australia
e-mail: cedric.pradalier@mavt.ethz.ch, k.usher@uq.edu.au

Received 11 November 2007; accepted 15 March 2008

Tractor-trailer reversing is a classical nonlinear control problem in which many of the solutions proposed in the literature perform poorly in the presence of real-world constraints such as steering angle, rate limits, and lags. In this paper we describe a new method in which an inner loop controls the hitch angle of the trailer, creating a virtual articulated vehicle to which existing control techniques can be applied. We provide an analysis of the stability and convergence properties of this control approach, as well as experimental results that illustrate the robustness of this approach to model estimation errors, low-level control loop dynamics, and other disturbances introduced by, for example, state estimation errors. © 2008 Wiley Periodicals, Inc.

1. INTRODUCTION

Reversing of tractor-trailer systems is a common task in both recreational and industrial settings. In the recreational domain, examples of tractor-trailer reversing tasks include parking of transportation trailers and the reversing of boat trailers onto boat ramps. A driver-aid or automation system for performing such tasks could be of significant benefit for inexperienced or unskilled drivers.

In industrial settings, examples of tractor-trailer systems include agricultural systems, transportation, and cargo handling. Generally, drivers in industry are well trained and experience little difficulty in achieving a desired tractor-trailer reversing trajectory. However, a growing number of these tasks are being automated, for which a robust and reliable control system is essential.

Most existing approaches address the problem through the development of complex, nonlinear controllers, which can prove to be difficult to implement and tune. Furthermore, few, if any, of these approaches can deal with slow steering loop dynamics and limits, and many of these methods also fail in the face of significant noise in the system state estimation.

By reformulating the problem such that the input to the system is the angle between the tractor and trailer (i.e., the hitch angle) rather than the tractor steering angle, the system can be treated as a "virtual" articulated vehicle. Closing the inner loop on the hitch angle can be achieved using a proportional–integral (PI) controller, and the outer loop on the trajectory can be closed with very simple methods based on path lateral, longitudinal, and orientation errors. This paper extends earlier work (Pradalier & Usher,

2007a, 2007b) by providing a more thorough theoretical analysis of the properties of this control approach.

The remainder of this paper is structured as follows: Section 2 defines the problem addressed, including the kinematic model of the tractor-trailer system; Section 3 reviews existing approaches and discusses the problems with these approaches as applied to platforms with significant underlying control loop dynamics; Section 4 outlines our approach to trajectory control and discusses in depth the hitch-angle stabilization loop; Section 5 describes the platform and the properties of the system considered in this paper, Section 6 outlines comprehensive experiments validating the approach, and Section 7 concludes the paper.

2. PROBLEM MOTIVATION

The problem addressed in this work is that of reversing a tractor-trailer system onto a preplanned trajectory in the face of very slow steering loop dynamics and steering angle range limitations. The system should also be tolerant to noise in the available pose estimates. Additionally, as the system is targeted as a driver aid or as an automation system for industrial robots, it should rely on minimal computational and sensory resources, be easy to implement and tune, and be tolerant to plant variation.

The geometry of the tractor-trailer system considered here is depicted in Figure 1. The lengths referred to in this figure are defined as follows: L, the distance between the front and rear axles of the tractor; L_1, the distance between the rear axle of the trailer and the hitch joint; and L_2, the distance between the trailer axle and the hitch joint.

The system states are described by (x, y), the Cartesian coordinate of point P in the global frame; θ_1 and θ_2, the heading of the tractor and trailer, respectively; and ψ, the hitch angle (note that $\theta_2 = \theta_1 + \psi$). The system is controlled via two inputs: v, the linear velocity of point P ($v = \|\dot{P}\|$); and ϕ, the steering angle of the front wheels. With this notation, the system can be described by the following equations:

$$\dot{x} = v\cos\theta_1, \tag{1}$$

$$\dot{y} = v\sin\theta_1, \tag{2}$$

$$\dot{\theta}_1 = \frac{v\tan\phi}{L}, \tag{3}$$

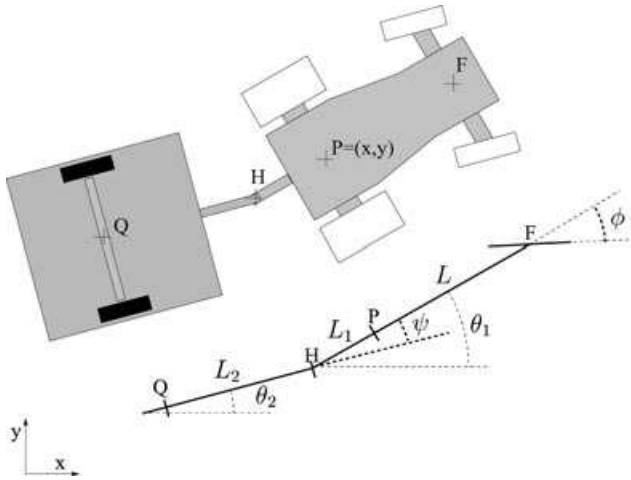


Figure 1. Kinematic model of a tractor-trailer system. It is particularly important to note the definition of the steering angle ϕ and the hitch angle ψ, i.e., the angle between the tractor and the trailer.

$$\dot{\psi} = \frac{-v\tan\phi}{L}\left(\frac{L_1}{L_2}\cos\psi + 1\right) - \frac{v}{L_2}\sin\psi. \tag{4}$$

The first three equations of this system are the standard kinematics for a car-like vehicle; see, e.g., De Luca and Oriolo (1995) for a derivation. Proving the expression for $\dot{\psi}$ requires consideration of the speed V_H of point H (resp. P or Q) in the global frame. First, we define $\Omega_1 = [0, 0, \dot{\theta}_1]^T$ and $\Omega_2 = [0, 0, \dot{\theta}_2]^T$. Then,

$$V_H = V_P + \Omega_1 \times \boldsymbol{PH}, \tag{5}$$

$$V_H = V_Q + \Omega_2 \times \boldsymbol{QH}. \tag{6}$$

with $\theta_2 = \theta_1 + \psi$ and where bold italic notation represents vectors. The expression for $\dot{\psi}$ is derived by solving

$$V_P + \Omega_1 \times \boldsymbol{PH} = V_Q + \Omega_2 \times \boldsymbol{QH}. \tag{7}$$

3. REVIEW OF EXISTING APPROACHES

The control of tractor-trailer systems has received much attention in the scientific and patent literature (see, e.g., Carter & Lormor, 2004; Karl, 1987; Robert, 2004) as it has clear industrial applications and is also interesting due to its inherent nonlinear

nature. In this section, we review several approaches to the problem and present representative examples of our testing of these approaches. To do this, we use the kinematic model of the tractor-trailer system [Eqs. (1)–(4)] and introduce additional dynamics in the steering response through rate limiting.

3.1. Stabilization Using a Virtual Tractor

The first method tested in this review relies on the idea that there exists a unique one-to-one mapping between the speed vector of a point on the tractor and the speed vector of a point on the trailer (Sekhavat, Lamiraux, Laumond, Bauzil, & Ferrand, 1997) (see Figure 1). Knowing this relation, it is possible to "virtually" exchange the role of the tractor and trailer. Given the distance from the trailer to the reference trajectory, a simple controller, such as pure pursuit (Hebert, Thorpe, & Stentz, 1997), can give the desired translational and rotational speed for the trailer. This control vector can then be mapped to a control vector for the tractor.

Experimentally, this method is able to stabilize the system, provided that we assume there are no limits on the steering angle rate. When even modest steering rate limits are introduced, the performance quickly degrades. Figure 2 illustrates this degradation: at a steering rate limit of 110 deg/s, control is still effective if not oscillatory, and at 100 deg/s, stabilization is no longer possible.

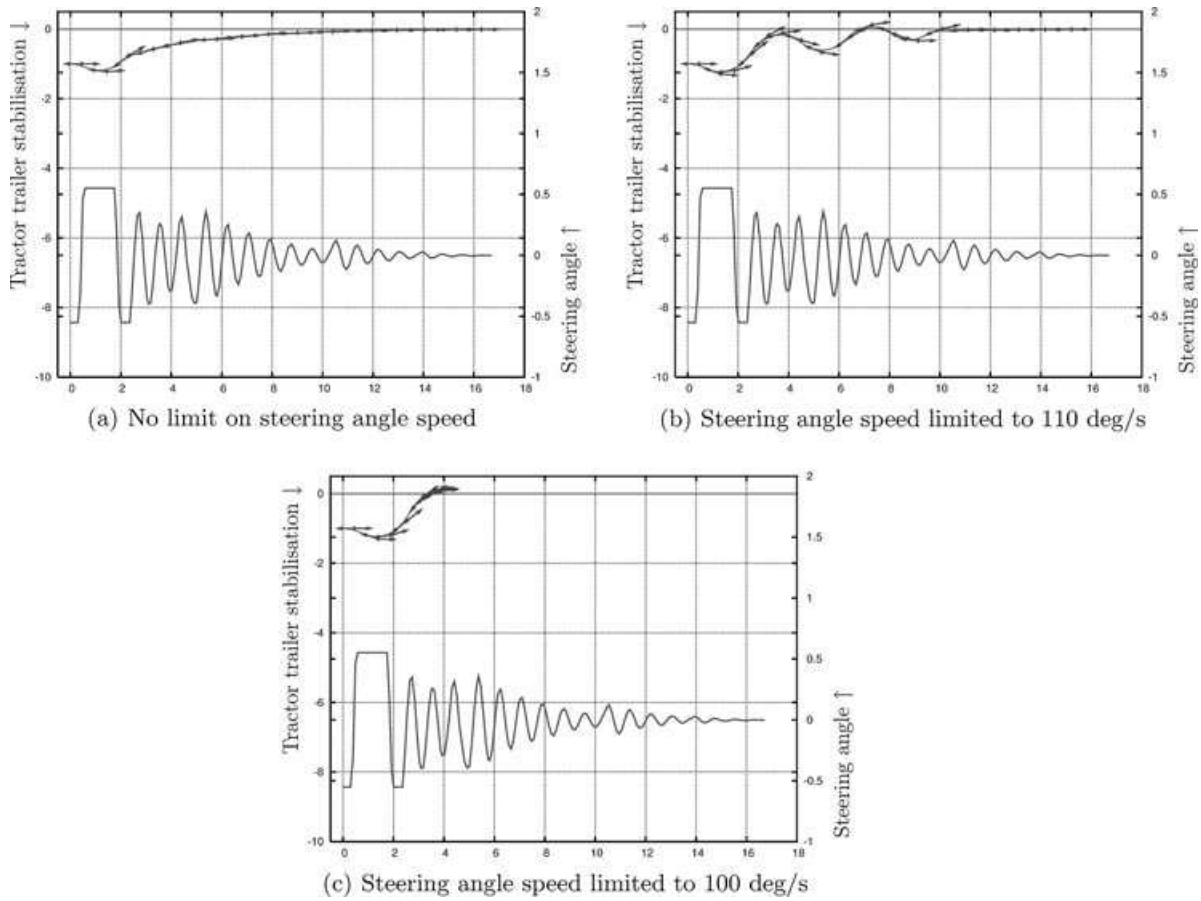


Figure 2. Stabilization of a tractor trailer to a straight trajectory ($y = 0$) using the HILARE controller (Sekhavat et al., 1997), while varying the steering angle rate (simulation). In each graph, the top part of the plot is demand trajectory (straight line) and the path of the vehicle moving from left to right, the arrows showing the heading of the tractor and trailer, and the bottom part shows the steering angle during the maneuver.

3.2. Differential Flatness

In the work of Rouchon, Fliess, Lévine, and Martin (1993), the tractor-trailer system depicted in Figure 1 is classed as a *general one-trailer system*, which has been proved to be differentially flat. Using this property, and inspired by Hermosillo and Sekhavat (2003) and Pradalier et al. (2005), we can convert the trajectory of the tractor-trailer system into the trajectory of its flat output (y_1, y_2):

$$y_1 = x + L_2\cos(\theta+\psi) + L(\psi)\frac{L_2\sin(\theta+\psi) - L_1\sin(\theta)}{\sqrt{L_1^2 + L_2^2 - 2L_1L_2\cos\psi}}, \quad (8)$$

$$y_2 = y + L_2\sin(\theta+\psi) + L(\psi)\frac{L_2\cos(\theta+\psi) - L_1\cos(\theta)}{\sqrt{L_1^2 + L_2^2 - 2L_1L_2\cos\psi}}, \quad (9)$$

with

$$L(\psi) = L_1L_2\int_{\pi}^{2\pi-\psi}\frac{\cos\sigma}{\sqrt{L_1+L_2-2L_1L_2\cos\sigma}}d\sigma.$$

Following Hermosillo and Sekhavat (2003) and Pradalier et al. (2005), a stabilizing control law is

$$y_{i=1,2}^{(3)} = (y_i^\star)^{(3)} + \sum_{j=0}^{2} k_{i,j}\left[y_i^{(j)} - (y_i^\star)^{(j)}\right], \quad (10)$$

where $(y_1^\star, y_2^\star)$ is the flat output corresponding to the state on the reference trajectory and $(.)^{(p)}$ is the time derivative operator of order p.

This technique requires the estimation of the two first derivatives of the state's flat output and then integration of the resulting control in order to deduce the actual steering and speed from the third derivative of the flat output, all of this while using and inverting Eqs. (8) and (9). Even if theoretically stable, the multiple derivatives and integrations make this method extremely sensitive to noise in the state estimation. In a real implementation, the state estimate can be quite noisy, making the second derivative of its flat output close to meaningless.

Our experience (Pradalier et al., 2005) in implementing this type of control law confirms this predicted sensitivity. Moreover, the tuning of the parameters $k_{i,j}$ is difficult due to their sheer number, their coupled effect on all the vehicle states, and the absence of any physical meaning of these parameters.

3.3. Trajectory Stabilization Using Precomputed Gains

Walsh, Tilbury, Sastry, Murray, and Laumond (1994) proposed a control law that, in theory, can exponentially stabilize a tractor-trailer system onto a reference trajectory. In principle, this control scheme estimates a gain for a linear control law on each point of the reference trajectory, using the shape of the trajectory over the next Δt seconds.

In our implementation, this approach presented two main difficulties. First, the precomputation of the gains involves complex integral estimations that can be prohibitively long if the trajectory is longer than a few meters. Second, this method does not provide a way to prioritize the control: When steering a trailer, it is essential to keep the hitch angle stable and controllable because entering a jackknife situation is irrecoverable unless forward motion is instigated. Unfortunately, this method does not provide any mechanism to deal with this constraint.

3.4. Chained-Form Representation

To obtain the chained form for this system, we first use Altafini's (2001) result, which shows that a car-like tractor pulling an off-centered single trailer can be modeled as a standard three-trailer system (that is, one tractor pulling three trailers). From this, the work of Sørdalen (1993) and Tilbury, Sørdalen, Bushnell, and Sostry (1995) gives the transformation of this system into a chained-form representation. The equations for this representation run to several pages in length and are omitted here for brevity.

One of the primary problems with the chained-form representation is that it creates very strong coupling between all the angles of the system; as a consequence, designing a controller and a tuning strategy is as hard as, or even harder than, when using the flatness property of the system.

3.5. Optimal Control

The *optimal control*–based methods use optimization schemes to derive a sequence of demands that will drive the tractor-trailer system onto the desired trajectory or path; see, for example, Altafini, Speranzon, and Wahlberg (2001) and Divelbiss and Wen (1997).

In simple terms, these methods use a vehicle model and a simulation process to compute the control commands that will lead to the best tracking of the trajectory. The required control trajectory can be optimized over some time horizon for minimal time, minimal control effort, or a combination of these and any other relevant "costs."

These methods rely on being able to adequately simulate the vehicle's behavior, which requires considerably more computational resource than would be required for the approach proposed in this paper. Deviations from the model, which in real-world implementations are inevitable, can lead to errors, which requires reestimation of the optimal control sequence. Furthermore, these methods are open loop, meaning that replanning is necessary to deal with errors in localization.

3.6. Learning-Based Approaches

The *learning-based* controllers seek to ease the computational burdens of the previous methods by providing a mapping between the current vehicle state, the desired state, and the required inputs to reach the desired state; see, for example, Koza (1992). Essentially, a simulated model of the tractor-trailer system is used to try many different possible methods and parameters. "Learning" occurs by searching the parameter space for the best set of methods/parameters, which are then encoded into, for example, a neural network or look-up table, which maps the "current" to "desired" configuration.

The main drawback of these techniques is the learning itself: If learning occurs from a model of the system, then errors in the model are clearly problematic; if learning occurs on the real vehicle, then there are safety issues because it is not possible to predict the behavior of the vehicle in the learning phase. Also, even if the best set of parameters performs well in practice, it is hard to guarantee its performance, which can be problematic in an application in which reliability is an issue.

4. OUR APPROACH

Our approach to stabilizing a tractor-trailer system to a trajectory is based on the idea of closing an inner loop around the trailer hitch angle and then treating the vehicle as a "virtual" articulated vehicle for which there are provably stable algorithms for stabilizing to a trajectory (Pradalier & Usher 2007b). We begin the discussion with the outer loop, which stabilizes the vehicle to the trajectory, and then we discuss in more detail the inner, hitch-angle control loop and its stability properties.

4.1. Trajectory Stabilization

To stabilize the vehicle to a trajectory (represented by a sequence of vehicle states indexed by time), we use a path-tracking controller derived for an articulated vehicle and apply an additional controller on the vehicle speed to ensure that the vehicle progresses along the trajectory.

The first important aspect of a tractor-trailer system is that its dynamics are asymmetric. When driving forward, the trailer angle is naturally exponentially stable. When reversing, it is naturally unstable. Consequently we use different control laws for the two situations.

When driving forward, we use a standard trajectory-tracking control law, such as *pure pursuit* (Hebert et al., 1997), and we ignore the trailer. When reversing, we use the trajectory control law presented in Ridley and Corke (2003), where a load haul dump (LHD) vehicle was considered. LHDs are four-wheeled, center-articulated vehicles that are used in underground metalliferous (non-coal) mining operations for the transport of ore.

The control law presented in Ridley and Corke (2003) aims at stabilizing the vehicle on a path, i.e., a two-dimensional curve in the plane. It relies on three error measurements, as depicted in Figure 3: ξ_y, the lateral error; ξ_θ, the heading error; and ξ_κ, the curvature error. The control law is defined as

$$\dot{\psi}^\star = K_y \xi_y + K_\theta \xi_\theta + K_\kappa \xi_\kappa, \tag{11}$$

where K_y, K_θ, and K_κ are tuning parameters.[1] From the hitch-angle derivative, a desired hitch angle $\psi^\star$ is computed by integration.[2] Here, this desired hitch

[1] For operations at higher velocities, these gains are speed dependent (Ridley & Corke, 2003).

[2] In discussions with the implementers of Ridley and Corke (2003) on a real LHD (Duff, Roberts, Corke, Sikka, & Winstanley, 2000; Roberts, Duff, & Corke, 2002), it was found that it may sometimes be necessary to directly set $\psi^\star = K_y \xi_y + K_\theta \xi_\theta + K_\kappa \xi_\kappa$. With this change, the vehicle will tend to straighten instead of bending to correct minor errors as would occur with the integrated version. This can be useful when minor errors are mostly the result of localization noise. In practice, we use the nonintegrated version of the control, and the success of this strategy is borne out in the results.

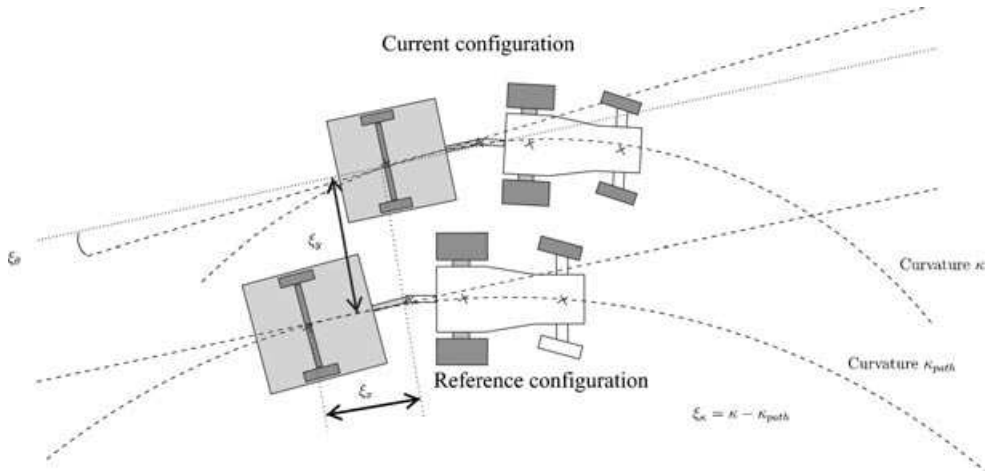


Figure 3. Trajectory error definitions.

angle is then fed into the hitch-angle stabilization law described in the next section. Speed is controlled using a standard pursuit approach to maintain the vehicle's progression along the trajectory and the system switch from the forward control law to the reversing control law when a change of direction is required, as occurs when the onset of jackknifing is detected or there is a cusp in the trajectory [see Pradalier and Usher (2007b) for details]. The trajectory-tracking system is described in block diagram form in Figure 4.

A common problem for such switching controllers is that of "chattering," in which the system gets trapped on the threshold between the two controllers. In this work, we rely on the concept of hysteresis to overcome this problem. For example, on detecting a jackknife situation that requires forward motion for correction, we ensure that the forward motion continues until the hitch-angle error converges to a value 10 times smaller than the jackknife detection value. This is an acceptable behavior because convergence is exponential when driving forward, as shown in Lamiraux (1997).

Of course, more sophisticated switching strategies are available [see, e.g., Campi, Hespanha, and Prandini (2004) and Hespanha, Liberzon, and Morse (2003a, 2003b)], but the hysteresis concept has proven to be sufficient in practice in this instance. However, this strategy assumes that the trajectory has been planned as a smooth path with a relatively low maximum curvature and, consequently, the desired hitch angle $\psi^\star$ is always small. In cases in which this assumption is not valid, it is certainly possible to create situations in which the system will chatter.

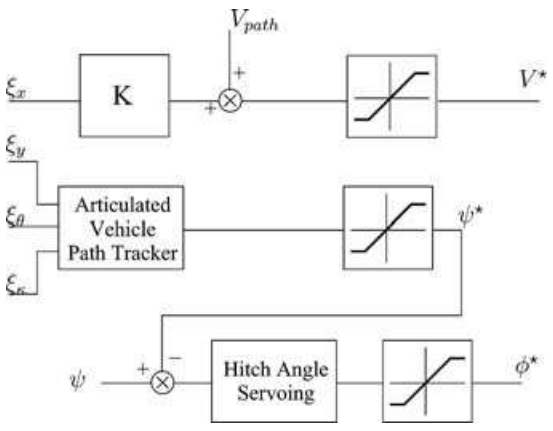


Figure 4. Block diagram for the reversing control law.

4.2. Hitch-Angle Stabilization

We now turn to the critical element in our approach, which is the hitch-angle stabilization loop. The details of this method appear in our previous work on the topic (Pradalier & Usher, 2007a, 2007b), and only an outline is provided here. However, a more detailed formal analysis of the properties of this loop will be presented, from which a nonlinear control

law arises that allows operation over a broader hitch-angle range.

In short, a simple PI controller is enough to achieve the hitch-angle stabilization task:

$$\phi = K_p(\psi^\star - \psi) + K_I \int_0^t (\psi^\star - \psi)du,$$

where $\psi^\star$ is the demanded hitch angle and K_p and K_I are the proportional and integral gains.

Our previous analysis of this controller showed that for nonzero operating points, the control demand needs to be modified in order to compensate for the steady-state error introduced by nonlinearities at these points. Of course, the integral term in the controller would compensate for these effects, but this premodification of the demand speeds up the system response. The expression for modifying the demand such that the system converges to the true demand is

$$\psi^d(\psi^\star) = \frac{K_P L_1 - L + K_P L_2}{K_P(L_1 + L_2)}\psi^\star. \tag{12}$$

Using the above relations, the control law to obtain theoretical convergence on $\psi^\star$ is

$$\phi = K_p[\psi^d(\psi^\star) - \psi] + K_I \int_0^t (\psi^\star - \psi)du. \tag{13}$$

As will be highlighted in the following analysis, the integral term is required to account for minor inaccuracies remaining after the proportional control that can result from the linearization leading to $\psi^d(\psi^\star)$ or from errors in the vehicle model (L, L_1, and L_2).

4.2.1. Stability Analysis

First we assign a Lyapunov function to the system that, in order to prove stability, must be positive semidefinite for all values of the system state. We choose

$$V = \tfrac{1}{2}[\psi^d(\psi^\star) - \psi]^2. \tag{14}$$

The stability of the system can be determined by analyzing $\dot{V} = dV/dt$, i.e.,

$$\dot{V} = \dot{\psi}[\psi - \psi^d(\psi^\star)], \tag{15}$$

where we have used the approximation that $\dot{\psi}^d(\psi^\star) = 0$. This assumption means that we are considering very smooth trajectories, where the rate of variation of the desired trailer angle is small with respect to the variation of the actual angle.

Inserting the control law of Eq. (13) (the proportional component only) and the expression for $\dot{\psi}$ [Eq. (4)] into the above relation leads to

$$\dot{V} = -\frac{v\tan\left\{K_p\left[\psi^d(\psi^\star) - \psi\right]\right\}}{L}\left(\frac{L_1}{L_2}\cos\psi + 1\right) - \frac{v}{L_2}\sin(\psi)$$

after some rearrangement. Figure 5(a) shows a plot of the value of this function across the range of inputs $\psi^\star = [-\pi/2, \pi/2]$ and the output state $\psi = [-\pi/2, \pi/2]$, where $K_P = 1$ has been substituted, and Figure 5(b) highlights the instability points.

These plots indicate that the system is stable for most of the operating region. Exceptions include the "ripples" as indicated in Figure 5(b), but these are avoided in practice by restricting the operating regime. Furthermore, Eq. (16) can be used to monitor the system, altering behavior as appropriate. For example, if Eq. (16) evaluates to a positive number, the reversing maneuver will inevitably lead to a jackknife condition and should be ceased, and a forward correcting motion initiated.

4.2.2. Alternative, Nonlinear Controller

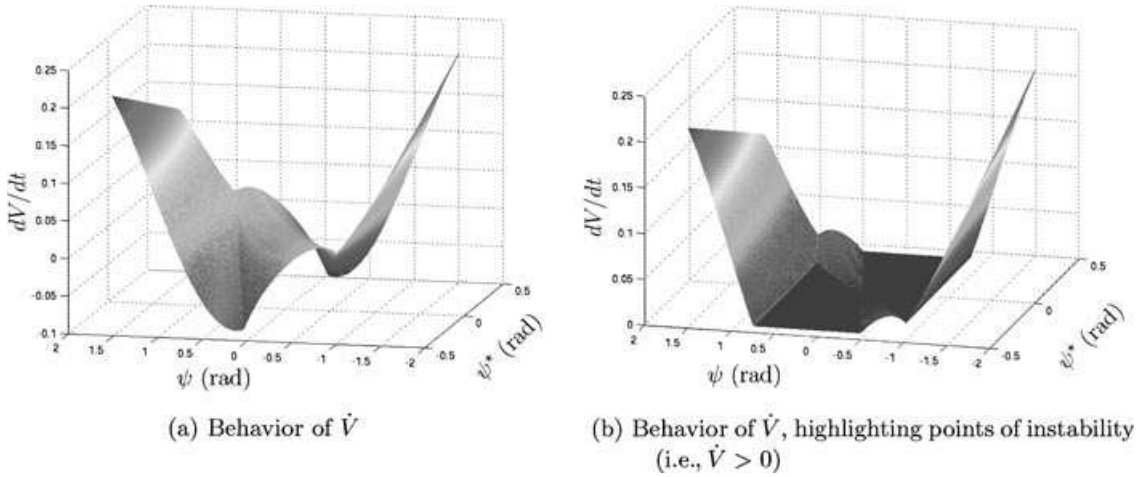
The Lyapunov function analysis from the preceding section can be used to derive an alternative, nonlinear, hitch-angle controller. First we note that the system is stable for $\dot{V} \leq 0$ [where $\dot{V}$ is from Eq. (15)], which can be achieved by setting

$$\dot{\psi} = -K(\psi - \psi^\star), \tag{16}$$

where K is a positive gain. Inserting the expression for $\dot{\psi}$ [Eq. (4)] and rearranging this equation for ϕ leads to the controller

$$\phi = \arctan\left[\frac{L}{v}\frac{K(\psi - \psi^\star) - \frac{v\sin\psi}{L_2}}{\frac{L_1}{L_2}\cos\psi + 1}\right]. \tag{17}$$

This version of the hitch-angle controller theoretically ensures convergence across the range of hitch angle from $-\pi/2$ to $\pi/2$. Additionally, this controller does



(a) Behavior of $\dot{V}$

(b) Behavior of $\dot{V}$, highlighting points of instability (i.e., $\dot{V} > 0$)

Figure 5. Surface showing the closed-loop behavior of the time derivative of the Lyapunov function $\dot{V}$. Color scale highlights the value of $\dot{V}$.

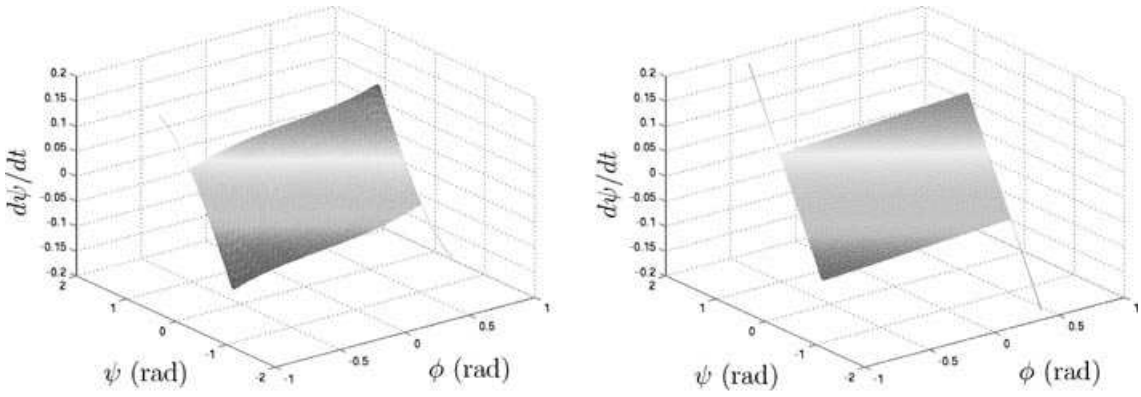
not require a modification of the demanded hitch angle $\psi^\star$ for nonzero operating points. However, like the controller of Eq. (13), this control law is also sensitive to errors in the system's geometric parameters. Thus, an integral term is required to ensure convergence without steady-state error. In practice, as will be illustrated in Section 6.1., due to the restricted operating range in hitch angle, the performance of this controller is about equivalent to that of Eq. (13).

4.2.3. Can the System Be Linearized?

Analysis of the system is substantially simplified if we can remove its nonlinear elements. To evaluate the linearity of the system, we first look at the closed-loop behavior of the $\dot{\psi}$ dynamics by inserting the control law of Eq. (13) into Eq. (4). Figure 6(a) shows a plot of these dynamics over a feasible range of the state and input space.

If instead we linearize the system about $\phi = 0$ and $\psi = 0$, we have $\tan\phi \approx \phi$, $\cos\psi \approx 1$, and $\sin\psi \approx \psi$, and we obtain the approximated (open-loop) ψ dynamics:

$$\dot{\psi} = -\left[\frac{v}{L}\phi\left(\frac{L_1}{L_2}+1\right) + \frac{v}{L_2}\psi\right], \tag{18}$$



(a) Full nonlinear closed-loop behavior of $\dot{\psi}$

(b) Linearized closed-loop approximation of $\dot{\psi}$

Figure 6. Comparison of full and linearized $\dot{\psi}$ dynamics. Color scale highlights the value of $\dot{\psi}$.

and again inserting the control law of Eq. (13), we obtain the closed-loop behavior for $\dot{\psi}$. Figure 6(a) illustrates the linearized $\dot{\psi}$ dynamics. Comparison with Figure 6(b) shows that the linearized response closely follows the true dynamics for the majority of the workspace and, even at the extremities, exhibits similar behavior.

The ability to linearize the system allows for the use of classical linear control tools for stability analysis and tuning purposes.

4.3. Tuning Strategy

A key advantage of this controller over existing methods is its simplicity, making it easy to implement and tune. The tuning strategy we use consists of three steps:

Tuning of the hitch-angle controller This requires tuning of the parameters K_P and K_I in Eq. (13). A standard proportional integral derivative (PID) tuning technique can be used, based on an analysis of the linearized system. In practice, we first start with $K_P = 1$ and $K_I = 0$, and then initialize the system with a nonzero hitch angle, stabilizing to zero while reversing at constant speed. We then increase K_P until oscillation appears. When K_P is optimal, we increase K_I, and the process is repeated, except in this case it is preferable to stabilize the system to a nonzero hitch angle in order to set the integral gain high enough to account for model errors. This procedure should also give practical bounds on the controllable hitch angles.

Tuning of K_x, K_y, and K_θ These parameters can be tuned by subjecting the system to lateral steps: From a starting position with a null hitch angle, the system must stabilize itself to a straight trajectory with a lateral offset (1 m, for instance), a process illustrated in Figure 7. The initial values of these parameters can be set so as to generate the maximum hitch angle for a given error. For instance, we may want to use the maximum hitch angle when the lateral error $|\xi_y|$ reaches 1.5 m or the heading error $|\xi_\theta|$ reaches 30 deg. During this stage, K_κ should be set to zero.

Tuning of K_κ K_κ has an influence only when driving on curved trajectories. As a consequence, it should be tuned by controlling the system to a circular trajectory with a feasible curvature. Using only K_y and K_θ, the system should be able to closely track a circular path. K_κ adds "look-ahead" by keeping the hitch angle above zero even if the other errors are null. In practice K_κ is less important than the other gains because changing the curvature is, in effect, how we control the trajectory of our system.

4.4. Simulations

Figure 7 shows the resilience of our controller to steering angle rate limitations. From an unlimited steering rate to a maximum of 20 deg/s, the controller was able to correct a 1-m lateral error with a very smooth path. With a maximum steering rate of 15 deg/s, a limit cycle starts to appear but the system still converges. At a rate limit of 10 deg/s, the controller can no longer stabilize the trailer onto the reference path. These results should be compared to those of Figure 2, where the minimum acceptable steering rate was roughly 100 deg/s.

5. EXPERIMENTAL PLATFORM

The platform used in these experiments is the CSIRO Autonomous Tractor (AT), as shown in Figure 8. It is an Ackerman-steered, ride-on mower that has been retrofitted with an array of actuators, sensors, and a computer system enabling the implementation and testing of control and navigation algorithms. The trailer hitch angle is sensed using a set of string pot encoders. Table I summarizes the system geometry with reference to Figure 1. For full details of the vehicle's design, refer to Usher (2005).

5.1. System Dynamics

The dynamics of the underlying control loops are extremely important in the application of trajectory and pose control algorithms to nonholonomic systems. In this section we identify models of the AT's response to steering and velocity inputs, which are subsequently used for system design and analysis purposes.

Table I. Geometric parameters of the CSIRO tractor-trailer system.

Parameter	Value (m)
L	1.2
L_1	0.45
L_2	1.2

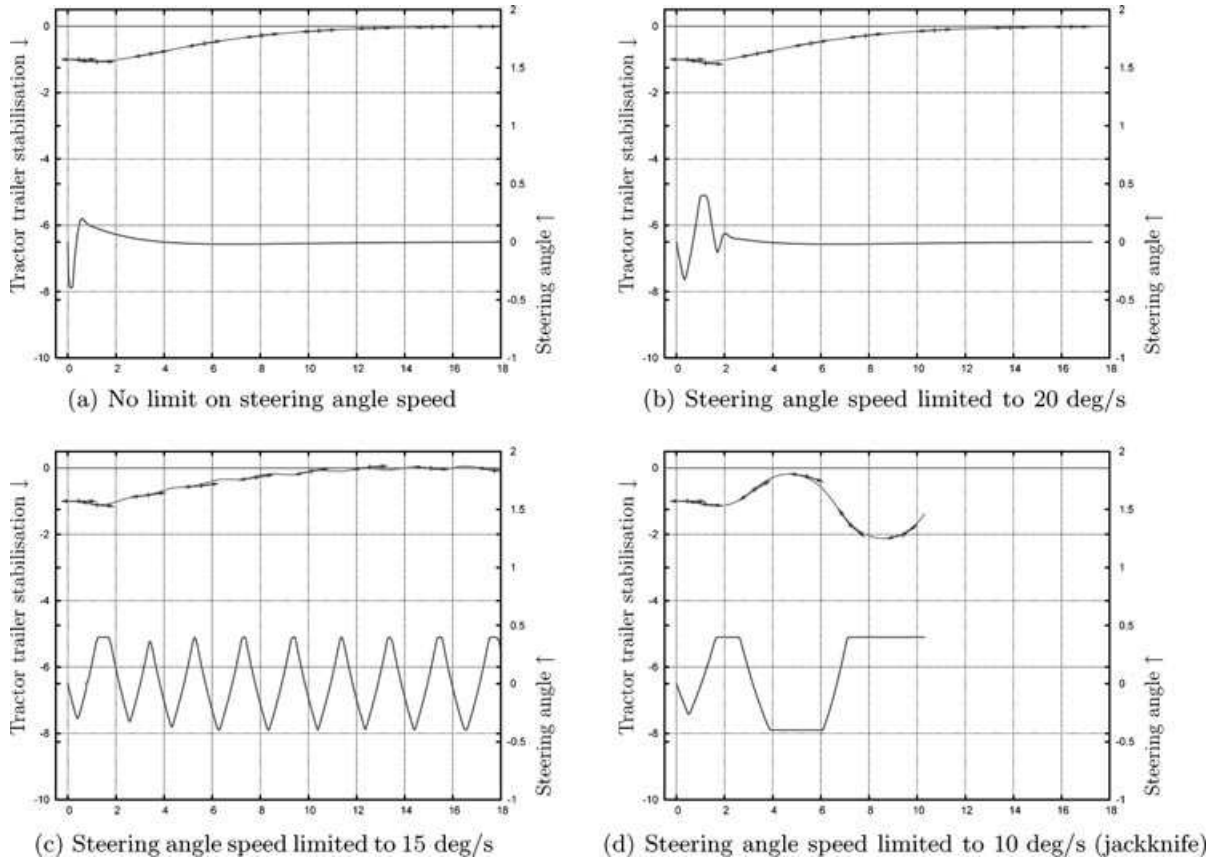


Figure 7. Stabilization of a tractor trailer to a straight trajectory using our approach while varying the steering angle speed (simulation). In each graph, the top part of the plot is the path of the vehicle moving from left to right, arrows showing the heading of the tractor and trailer, and the bottom part shows the steering angle during the maneuver.

5.1.1. Steering

An approximate model of the steering dynamics was experimentally identified from the response of the AT's steering loop to step changes in desired steering angle when traveling in reverse at 0.3 ms^{-1}. The response was determined to be approximately second order of the form (in the Laplace domain)

$$\frac{\phi(s)}{\phi^*(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n + \omega_n^2}. \tag{19}$$

The parameters ω_n and ζ vary due to the complexity of the interactions between the terrain and the wheels on different surfaces and also with the vehicle's translational speed. For these experiments, parameter values of $\omega_n = 2.15$ and $\zeta = 1.0$ were found to model the system adequately. Additionally, the steering angle range is limited to $\phi_{\max} = \pm 30$ deg and it is rate limited at approximately $\dot{\phi}_{\max} = \pm 20$ deg/s. Figure 9(a) shows a plot of the actual and modeled response of the vehicle to a step input.

5.1.2. Velocity

The velocity loop was empirically determined to have a first-order response, which is represented in the Laplace domain as

$$\frac{v(s)}{v^*(s)} = \frac{K_v}{\tau_v s + 1}. \tag{20}$$

Figure 9(b) illustrates the AT's response to a unit step change in velocity while traveling on level ground (concrete) and the response of the first-order model where $K_v = 1$ and $\tau_v = 1.33$. Again, the model

Figure 8. The AT with its trailer.

parameters will vary on sloping terrain, on different surfaces, and under different loading conditions. In addition, the speed of the AT is constrained to the range $-1.5\ \text{ms}^{-1} < v < 3\ \text{ms}^{-1}$.

In practice, the velocity loop has little impact on the overall system dynamics, and thus the primary focus in this work is on the analysis of the system, including the steering-loop dynamics.

5.2. Hitch-Angle Control System Modeling

We now consider a linear model of the hitch-angle control loop for the AT. In this analysis, we consider the controller of Eq. (13) and, to ease the analysis, ignore the effects of the integral term in the controller.

The steering loop dynamics are modeled by Eq. (19), and the linearized hitch-angle dynamics are represented (after the introduction of an intermediary state for the steering) by

$$\begin{bmatrix} \dot{\rho} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & -\omega_n^2 & \omega_n^2 K_P \\ 1 & -2\zeta\omega_n & 0 \\ 0 & -\frac{v}{L}\left(\frac{L_1}{L_2}+1\right) & -\frac{v}{L_2} \end{bmatrix} \begin{bmatrix} \rho \\ \phi \\ \psi \end{bmatrix} + \begin{bmatrix} \omega_n^2 K_P \frac{(L_1+L_2)+L}{(L_1+L_2)} \\ 0 \\ 0 \end{bmatrix} \psi^*, \tag{21}$$

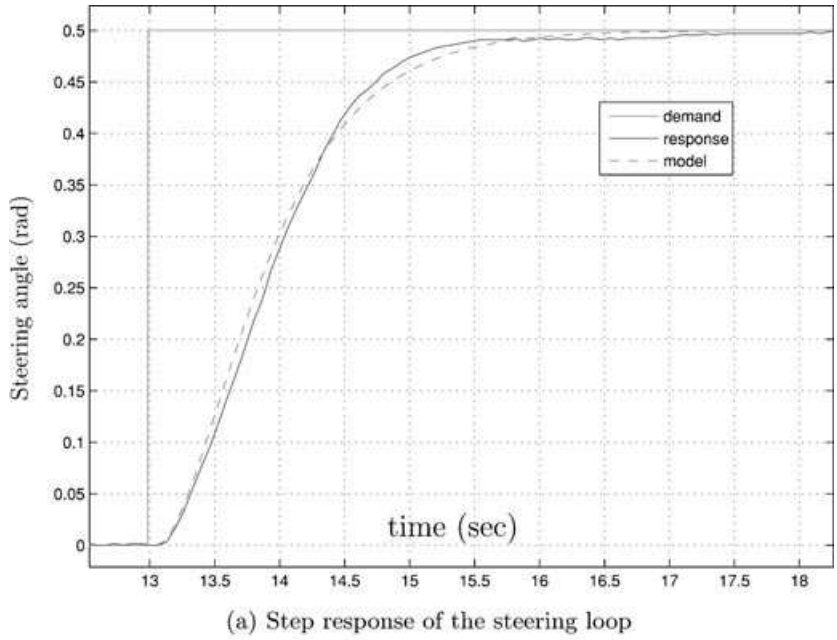
$$\dot{x} = A \cdot x + B \cdot u. \tag{22}$$

5.2.1. Root Locus

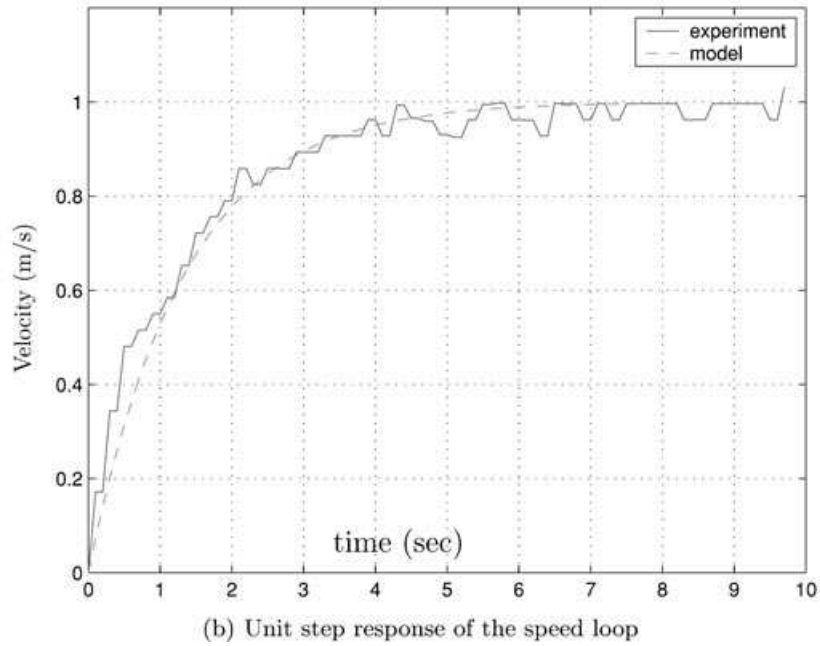
The characteristic equation for this system can be found by calculating the determinant of $sI - A$, where A is the state transition matrix. The characteristic equation for this system is

$$s^3 + \left(\frac{v}{L_2} + 2\zeta\omega_n\right)s^2 + \left(\frac{2\zeta\omega_n v}{L_2} + \omega_n^2\right)s - \omega_n^2 \frac{v}{L_2}\left[1 - \frac{K}{L}(L_1 + L_2)\right]. \tag{23}$$

By placing Eq. (23) in the form $1 + K_P GH = 0$, we can observe the behavior of the roots of the characteristic equation for various values of K_P. The resulting root locus plot is shown in Figure 10. The root locus indicates that the system is stable for $0.72 < K_P < 9.68$, and for a critically damped system, the gain should be set to $K_P = 1.65$.



(a) Step response of the steering loop



(b) Unit step response of the speed loop

Figure 9. Step response of the steering and velocity loops. Model data are plotted as a dashed line and experimental data as a solid line.

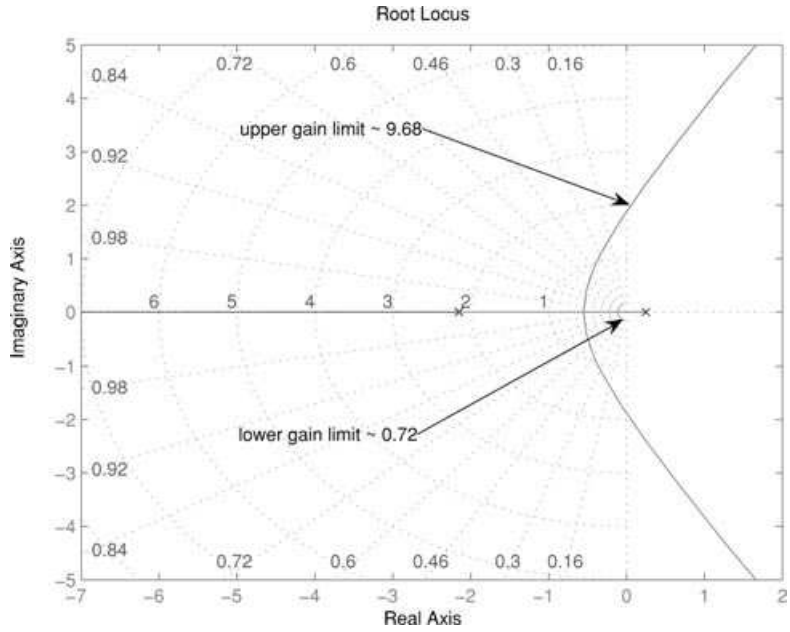


Figure 10. Root locus plot for the closed-loop system, including the steering loop dynamics, using the linear P controller.

5.2.2. Sensitivity Analysis

We now turn to an analysis of the sensitivity of the system to parameter variations. First, we present the case for the parameter L, following which a summary of the results for the remaining parameters is presented.

To determine the system's sensitivity to L, we evaluate

$$S_{s.L} = \frac{L}{s}\frac{\partial s}{\partial L}, \tag{24}$$

which results in

$$\frac{\frac{\omega_n^2 K_P v(L_1+L_2)}{LL_2}}{3s^3 + \left(\frac{2v}{L_2} + 4\zeta\omega_n\right)s^2 + \left(\omega_n^2 + \frac{2\zeta\omega_n v}{L_2}\right)s}.$$

We now evaluate this sensitivity at two representative levels of system gain K_P, using the root locus of Figure 10 to determine the dominant poles at these points:

$$K_P = 1.23: \quad s = -0.357, \qquad S_{s.L} = -5.27,$$
$$K_P = 3.10: \quad s = -0.362 \pm 1.0i, \; S_{s.L} = 0.40 \angle 141\text{deg}.$$

This indicates that the poles in the system are very sensitive to changes in the parameter L and that this sensitivity is substantially reduced with increasing gain.

We can also evaluate the effect of a change in L on the position of the dominant pole by rearranging Eq. (24):

$$\Delta s = s S_{s.L}\frac{\Delta L}{L}.$$

At the gain $K_P = 1.23$ (where the mathematics are more straightforward), a 10% increase in L yields $\Delta s = 0.19$, or a shift to the right of 0.19 units for a 10% increase in L.

Table II summarizes the results of this analysis for the two selected gains analyzed above. With the exception of L_2, which appears to bear little influence over the system sensitivity, the system is similarly sensitive to the other parameters at the lower gain, with a vastly reduced sensitivity at the higher gain. Practically, this analysis implies that the system parameters should be measured with reasonable accuracy and that some form of integral action is required in the controller to accommodate any errors in these measurements.

Table II. System sensitivity to variations in geometric parameters and vehicle velocity.

	$\|\|S_{s.P}\|\|$	
Parameter (P)	$K_P = 1.23$	$K_P = 3.10$
L	5.270	0.390
L_1	47.600	0.018
L_2	0.015	0.001
v	125.790	0.110

5.2.3. Gain Tuning

In practice, the system was tuned using the methods of Section 4.3. For the hitch-angle stabilization loop, gains of $K_P = 4$, $K_I = 0.03$ were found to provide both quick and accurate response with minimal overshoot, whereas for the trajectory-tracking loop, gains of $K_x = 2$, $K_y = 0.2$, $K_\theta = 1$, and $K_\kappa = 0.05$ were found to give the desired behavior.

6. RESULTS

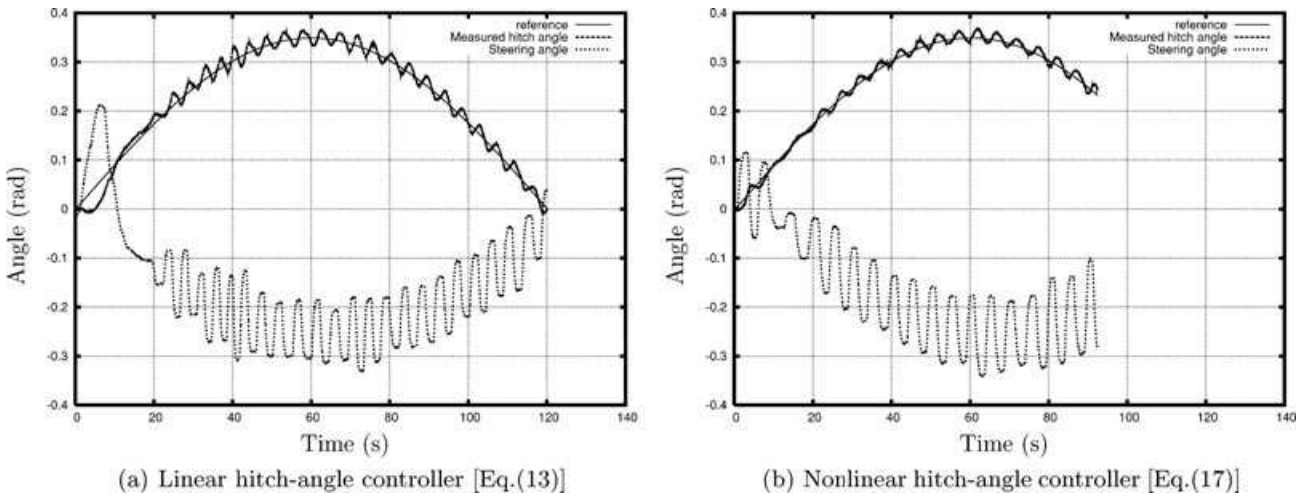
This section presents results obtained when tracking various reversing trajectories, first controlling only the hitch angle, then controlling the tractor's position using odometry-based localization, and then conducting several tests using an external localization estimate. The external localization estimates are provided by a particle-filter–based method using the vehicle odometry and sparse reflective beacons sensed with a front-mounted laser range finder [details on the localization system can be found in Duff, Usher, and Ridley (2006)]. This localization estimate is "drift-free" but comes at the cost of localization discontinuities when corrections are applied on spotting a beacon. Such discontinuities are especially challenging for the trajectory-tracking system.

6.1. Hitch-Angle Controller

Figure 11 shows experimental tests comparing the linear [Eq. (13)] and nonlinear [Eq. (17)] hitch-angle controllers, in which the tractor was reversing at a constant speed of 0.3 m/s. From these plots, we note that the linear and nonlinear controllers give very similar performance in terms of accuracy of the tracking and control of the oscillations. The oscillations of the hitch angle around the nominal trajectory are caused by slop in the hitch-angle sensing and a limit cycle caused by the steering rate limit. On the basis of these results, and for the sake of controller simplicity, we chose to use the linear controller for the following experiments.

6.2. Reversing on a Circle with Fixed Radius

In the first set of experiments, we define reference trajectories as arcs of circles of various radii. The pose of



(a) Linear hitch-angle controller [Eq.(13)] (b) Nonlinear hitch-angle controller [Eq.(17)]

Figure 11. Experimental performance comparison of the linear and nonlinear hitch-angle controllers in following a sinusoidal input. Reference trajectory is shown as a solid line, actual trajectory as a dashed line, and steering angle input as a dotted line.

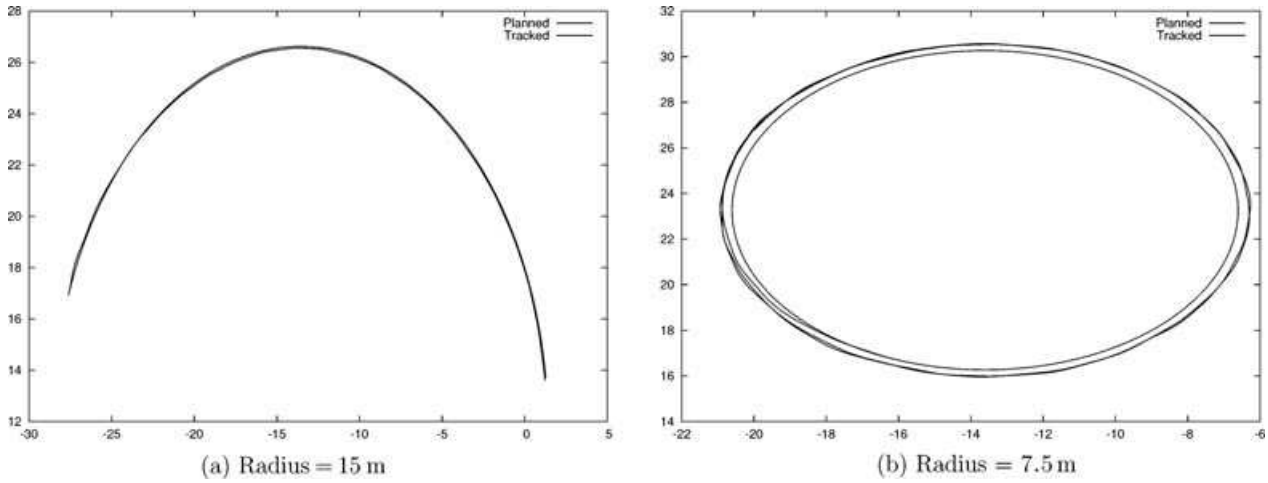


Figure 12. Trajectory tracking with odometry-based localization.

the vehicle is computed using odometry information. The advantage of this setting is that the vehicle pose is a very clean and smooth signal. The disadvantage is that odometry localization is known to drift over time and, consequently, is not suitable for real-life, long-range, robotic operations.

Figure 12 shows tracking results for a circular path, with radii of 7.5 and 15 m. As can be observed, the tracking is very accurate for radii of 15 m. For a radius of 7.5 m, the maximum achievable curvature is reached and the system cannot converge to the required trajectory. However, it should be noted that the system converges to a stable orbit, which is the best it can do to follow the required curvature.

6.3. Reversing on a Badly Planned Trajectory

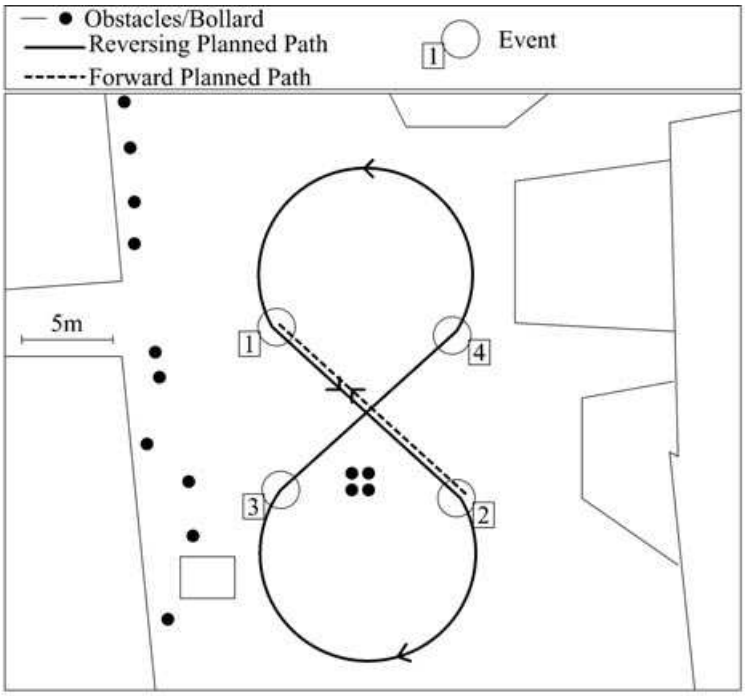
In the second set of experiments, we stretch the system by introducing a more complex path, as depicted in Figure 13(a). We consider this trajectory to be "badly planned" because it does not take into account the limitations of the vehicle, especially the limited turning radius and the very strong limitation on the hitch-angle rate. In addition, the trajectory contains path discontinuities in which straight lines join circular segments; such discontinuities are challenging for the control law because they require a very fast change of hitch angle. Our objective here is to show the performance of our system at the limits of its nominal specification. To further challenge the controller, we also use the external localization system rather than the vehicle odometry.

Figure 13(b) illustrates how the planned path is tracked by our tractor-trailer systems. From this experiment, we note that on straight-line segments the system converges reasonably fast. On the curved segments, the curvature is beyond the vehicle capabilities and the tracked trajectory is offset from the planned one. Transitions between the curved and straight segments of the path (Events 5, 8, and 9) introduce a discontinuity that the system struggles to deal with but nevertheless recovers from. Localization discontinuities (Events 6 and 7) are handled much more gracefully than the path discontinuities. Finally, Event 5 is an occurrence of a situation in which a beginning of jackknife was detected and the vehicle had to drive forward for approximately 2 m to restabilize the trailer, before continuing reversing.

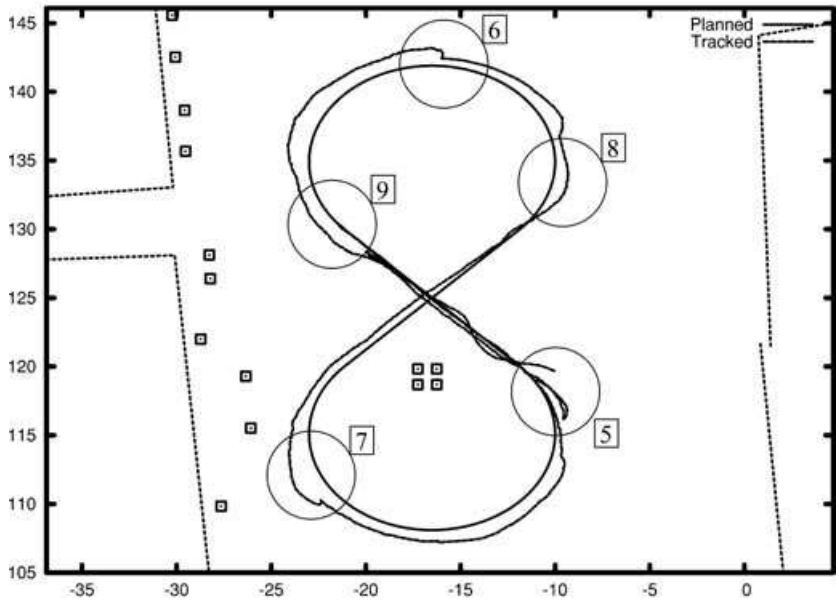
From these experiments, it is clear that in order to obtain very accurate tracking, the path planner needs to take into account the strong constraints of this system, in particular, the small maximum trackable curvature and the need for smooth curvature profiles. However, the trajectory control performed extremely well within these constraints.

6.4. Reversing on a Preplanned Trajectory

In this set of experiments, our objective was to demonstrate the performance of our system in a real-world situation. To this end, we designed a path across our research center. This path is approximately 170 m long and was planned manually using hand-selected way points and interpolating splines. This



(a) Schematic of the planned trajectory



(b) The tracked trajectory

Figure 13. Performance for a poorly planned trajectory that does not account for curvature limitations or path discontinuities. *Events* are denoted by the boxed numbers.

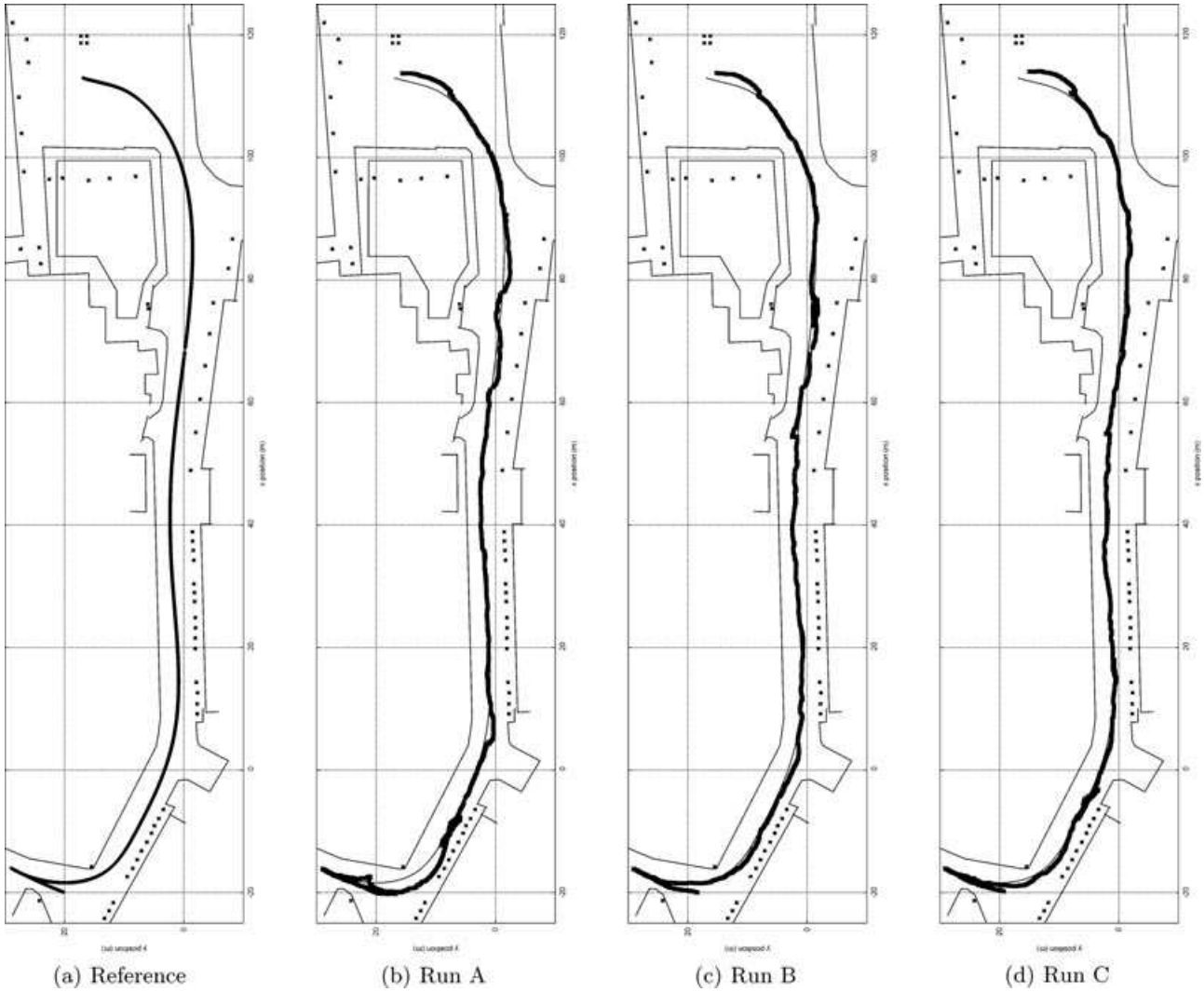


Figure 14. Reversing trajectories of the autonomous system on three independent runs.

approach guarantees a smooth and continuous trajectory. For these experiments, the system relied on the external localization estimates.

The resulting path is depicted in Figure 14(a). It is composed of two segments: a short forward motion to help align the system with the trajectory and a long reversing segment of approximately 160 m. The main constraints when designing this trajectory were to keep the vehicle close to the middle of the road and to minimize the path curvature.

Figures 14(b)–14(d) show the trajectories realized by our controller on three separate runs. More accurately, the trajectories represent the path of the center of the AT's rear axle. Examples of localization jumps (corrections) can be seen at the top end of all trajectories and around $(55, 0)$ in Runs B and C. It is important to note that our system stays stable around the reference trajectory, even when the localization estimation is very unstable. In some situations, e.g., close to $(-10, 10)$ in Runs A and C, or close to $(75, 0)$ in Run B, the localization estimate oscillates between the two sides of the reference trajectory. If this occurs at the wrong frequency, it can bring the system to a jackknife situation. In all cases, the preliminary signs of

Table III. Summary of experimental run statistics, for human and autonomous drivers.

Subject	Graph Label	Experience	Duration (s)	Corrective Fwd Motions	Avg Speed (m/s)
1	1	Professional	160	0	1.12
2	2	Beginner	300	4	0.60
3, Run 1	3	Beginner	322	3	0.55
3, Run 2	4	Beginner	230	2	0.78
4, Run 1	5	Medium	278	2	0.64
4, Run 2	6	Medium	290	2	0.62
5, Run 1	7	Medium	400	2	0.45
5, Run 2	8	Medium	250	0	0.72
6	9	First time	980	14+	0.18
Auto A	10	Autonomous	620	1	0.29
Auto B	11	Autonomous	600	1	0.3
Auto C	12	Autonomous	628	2	0.29

this event were detected by the system, and a short forward motion was initiated to realign the vehicle with the reference trajectory.

6.5. Evaluation: Comparison with Human Drivers

As a final experimental evaluation of our reversing system, we tested a number of human drivers in performing a trajectory similar to that performed by the autonomous system outlined in Section 6.4. Skill levels of these drivers were the following:

First time: first experience of trailer reversing
Beginner: driver having reversed a trailer a couple of times in recreational activities
Medium: driver who regularly reverses trailers in recreational activities
Professional: professional truck driver

For practical reasons, the drivers were required to keep the vehicle in the middle of the road, rather than following the same trajectory as the automated system.

We evaluated the drivers' ability to stabilize the trailer and to avoid jackknife situations and the statistical properties of their steering input. Figure 15 shows an example of human-controlled trajectories, in comparison with autonomous system ones.

6.5.1. Trajectories

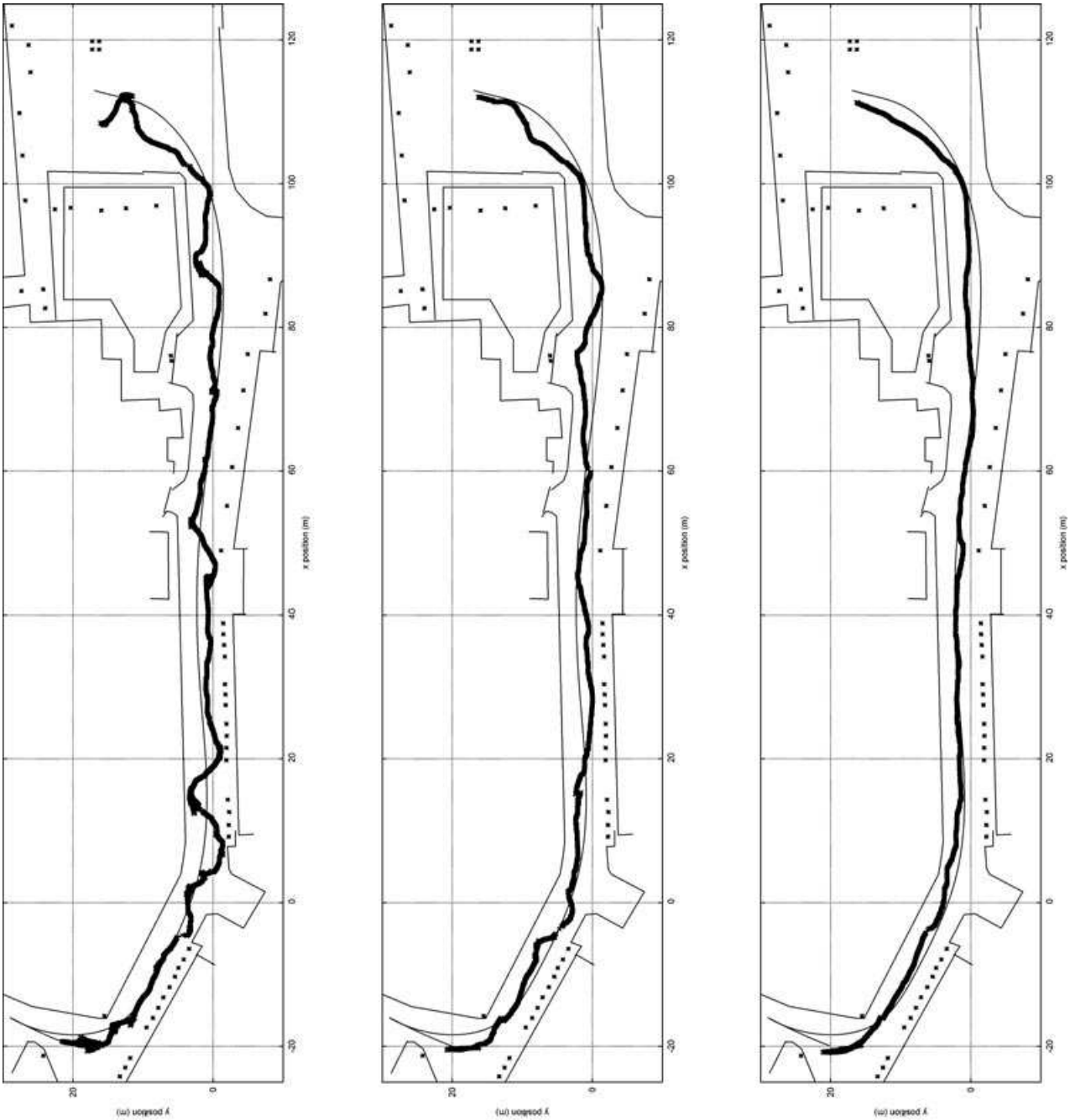
These experiments were conducted using six drivers, some of whom drove the trajectory twice. Some statistics about the drivers' performances are presented in Table III. Examples of trajectories realized by representative subjects are presented in Figure 14(d), where the paths were recorded using the external localization system.

When comparing the performance of human drivers with the autonomous control, one obvious difference is the travel speed. Except for the driver who was reversing a trailer for the first time, all human drivers were much faster than our system. As we will see later, this is mainly a result of the limited actuation capabilities of the autonomous system; to compensate for these limitations, the system has to be artificially slowed down.

Concerning the trajectories, the autonomous system realizes smoother paths than the inexperienced drivers and drives in a similar fashion as the more experienced human drivers. This is a definitive achievement given the limited actuation and the fact that the control relies on localization estimates, which can be discontinuous.

6.5.2. Hitch-Angle Stabilization

Figure 16(a) depicts the statistical properties of the measured hitch angle for each run as box diagrams. Each box represents the statistical distribution of measured hitch angles through its five-number summaries [the smallest observation, lower quartile (Q1), median, upper quartile (Q3), and largest observation]. If we call IQR the interquartile range, i.e., Q3 − Q1, then any data more than 1.5 IQR from the closest quartile (Q1 or Q3) are considered outliers. Graphically, the boxes in this diagram show the Q1 − Q3 interval: The "whiskers" show the farthest data



(a) Subject 6 (first time)

(b) Subject 4, Run 1 (medium skill)

(c) Subject 4, Run 2 (medium skill)

Figure 15. Reversing trajectories of inexperienced to medium drivers. Numbers correspond to those in Table III.

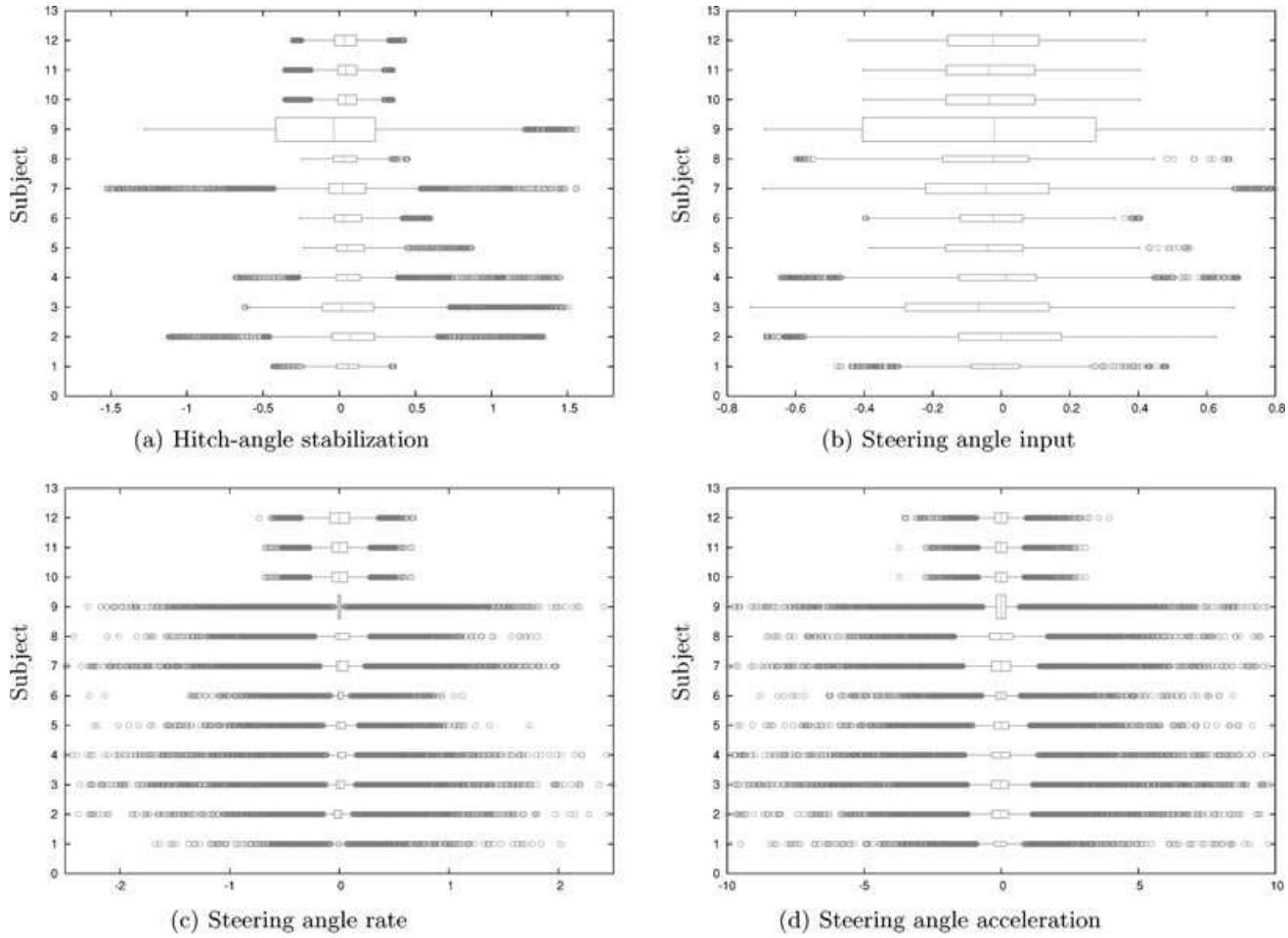


Figure 16. Comparisons of the capabilities of human drivers and the autonomous system. Numbers correspond to those given in the Graph label column of Table III. Lines 10–12 correspond to the three runs of the autonomous system.

point from the quartiles that is not considered an outlier, and the red circles show points considered outliers. The height of the boxes is not meaningful in our analysis.

In practice each line represents one run, with numbers referring to the different subjects in Table III. Lines 10–12 refer to the autonomous runs. The width of a box with its whiskers gives the range of steering angle regularly used by the driver. The red circles depict the steering angle used by the driver in infrequent events, such as the correction of the beginning of a jackknife situation. Jointly, the boxes and the circles give the range of control the driver (human or autonomous) used to achieve the trajectory.

For the autonomous runs, we note that the median hitch angle is slightly shifted toward positive values. This is expected because the trajectory is close to a long right-hand turn. When comparing the hitch-angle range of the autonomous system with that of the human drivers, it is clear that the autonomous system keeps much tighter control of the angle than most of the human drivers and is comparable with the professional driver (Driver 1). Nevertheless, it must be noted that this performance is achieved at the cost of a much lower ground speed (0.3 instead of 1.0 m/s).

Figures 16(b)–16(d) illustrate the properties of the steering angle and its first derivative, recorded for the

human drivers and the autonomous system. These graphs demonstrate the reduced control range and capabilities available to the autonomous system in comparison with the capabilities of human drivers: The accessible steering range available to the automation system is reduced by 40% in comparison to the human drivers, and the accessible steering speed and steering acceleration are reduced by about 70%. From these observations, it is clear that human drivers use their additional range of control to drive faster while demonstrating similar hitch-angle stabilization performance.

Finally, it is interesting to note that with driving capabilities reduced by 70%, the performance of our system is similar to that of a professional driver, albeit with a velocity reduced by about 70%. We are yet to determine whether this is a coincidence.

7. CONCLUSION

This article evaluated a new control scheme for a tractor-trailer system. This scheme is based on a two-layer control loop: First a hitch-angle stabilization loop controls the angle between tractor and trailer, and then a path-tracking control loop, initially designed for an articulated mining vehicle, is adapted to our tractor-trailer system.

The main advantage of this approach over traditional methods is its simplicity of implementation. Only a few parameters need to be tuned, and they all have a clear physical meaning. Although simple, this control scheme relies on a mathematically sound background.

Finally, this control law was implemented on a real vehicle and experiments were conducted on challenging trajectories. Given the limited dynamic performance of our platform (slow response time, loose components, low speed actuation), the control law exhibited excellent convergence and stability properties. Furthermore, our controller also compared well with a range of human drivers on a similar trajectory, even though the control system is significantly handicapped in terms of its actuation capabilities.

ACKNOWLEDGMENTS

This work was funded by the CSIRO ICT Centre under the ROVER and Dependable Field Robotics projects. The authors thank the Autonomous Systems Laboratory team for their support of this work. Special thanks go to Polly Alexander, Stephen Brosnan, Peter Corke, Elliot Duff, Paul Flick, Leslie Overs, Ashley Tews, John Whitham, and Graeme Winstanley, who all contributed to the development of our experimental autonomous tractor.

REFERENCES

Altafini, C. (2001). Some properties of the general n-trailer. International Journal of Control, 74(4), 409–424.

Altafini, C., Speranzon, A., & Wahlberg, B. (2001). A feedback control scheme for reversing a truck and trailer vehicle. IEEE Transactions on Robotics and Automation, 17(6), 915–922.

Campi, M. C., Hespanha, J. P., & Prandini, M. (2004). Cautious hierarchical switching control of stochastic linear systems. International Journal of Adaptive Control and Signal Processing, 18, 319–333.

Carter, D., & Lormor, J. (2004). Vehicle steering aid system. GB patent GB2398050.

De Luca, A., & Oriolo, G. (1995). Modelling and control of nonholonomic mechanical systems. In Kinematics and Dynamics of Multi-Body Systems. CISM Courses and Lectures, vol. 360, pp. 277–342. Vienna: Springer-Verlag.

Divelbiss, A., & Wen, J. (1997). Trajectory tracking control of a car-trailer system. IEEE Transactions on Control Systems Technology, 5(3), 269–278.

Duff, E., Roberts, J., Corke, P., Sikka, P., & Winstanley, G. (2000). Autonomous underground mining vehicle final report (Internal Tech. Rep. CMST-BCC2000-22). Brisbane, Australia: CSIRO.

Duff, E., Usher, K., & Ridley, P. (2006). Swing loader traffic control (Tech. Rep. ICT 06/142). Brisbane, Australia: CSIRO ICT Centre.

Hebert, M., Thorpe, C., & Stentz, A. (1997). Intelligent unmanned ground vehicles: Autonomous navigation research at Carnegie Mellon. New York: Kluwer Academic Publishers.

Hermosillo, J., & Sekhavat, S. (2003). Feedback control of a bi-steerable car using flatness: Application to trajectory tracking (volume 4, pp. 3567–3572). In Proceedings of the American Control Conference. New York: IEEE.

Hespanha, J. P., Liberzon, D., & Morse, A. S. (2003a). Hysteresis-based switching algorithms for supervisory control of uncertain systems. Automatica, 39, 263–272.

Hespanha, J. P., Liberzon, D., & Morse, A. S. (2003b). Overcoming the limitations of adaptive control by means of logic-based switching. Systems & Control Letters, 49, 49–65.

Karl, S. (1987). Automatic reverse steering system for passenger vehicles with trailer. DE patent DE3538338.

Koza, J. (1992). A genetic approach to finding a controller to back up a tractor-trailer truck (pp. 2307–2311). In Proceedings of the 1992 American Control Conference. IEEE.

Lamiraux, F. (1997). Robots mobiles remorque: de la planification de chemins l'execution de mouvements. Ph.D. thesis, Polytechnic National Institute of Toulouse, France.

Pradalier, C., Hermosillo, J., Koike, C., Braillon, C., Bessière, P., & Laugier, C. (2005). The cycab: A car-like robot

navigating autonomously and safely among pedestrians. Robotics and Autonomous Systems, 50(1), 51–68.

Pradalier, C., & Usher, K. (2007a). Experiments in autonomous reversing of a tractor trailer system. In Proceedings of the International Conference on Field and Service Robotics, Chamonix, France. New York: Springer-Verlag.

Pradalier, C., & Usher, K. (2007b). A simple and efficient control scheme to reverse a tractor-trailer system on a trajectory (pp. 2208–2214). In Proceedings of IEEE International Conference on Robotics and Automation, Rome, Italy. IEEE.

Ridley, P., & Corke, P. (2003). Load haul dump vehicle kinematics and control. Journal of Dynanic Systems, Measurement and Control, 125, 54–59.

Robert, S. (2004). Trailer backing up device and method. U.S. patent US2004215374.

Roberts, J., Duff, E., & Corke, P. (2002). Reactive navigation and opportunistic localization for autonomous underground mining vehicles. International Journal of Information Sciences, 20, 127–146.

Rouchon, P., Fliess, M., Lévine, J., & Martin, P. (1993). Flatness and motion planning: The car with n-trailers (pp. 1518–1522). In Proceedings of European Control Conference, Groningen.

Sekhavat, S., Lamiraux, F., Laumond, J., Bauzil, G., & Ferrand, A. (1997). Motion planning and control for hilare pulling a trailer: Experimental issues (volume 4, pp. 3306–3311). In Proceedings of IEEE International Conference on Robotics and Automation. New York: IEEE.

Sørdalen, O. (1993). Conversion of the kinematics of a car with *n* trailers into chained form (pp. 802–819). In Proceedings of the IEEE International Conference on Robotics and Automation. New York: IEEE.

Tilbury, D., Sørdalen, O., Bushnell, L., & Sastry, S. (1995). A multisteering trailer system: Conversion into chained form using dynamic feedback. IEEE Transactions on Robotics and Automation, 11(6), 807–818.

Usher, K. (2005). Visual homing for a car-like vehicle. Ph.D. thesis, Queensland University of Technology, Brisbane, Australia.

Walsh, G., Tilbury, D., Sastry, S., Murray, R., & Laumond, J. (1994). Stabilization of trajectories for systems with nonholonomic constraints. IEEE Transactions on Robotic and Automation, 39(1), 216–222.

Appendix G

Pradalier et al. [2005]

Cédric Pradalier, Jorge Hermosillo, Carla Koike, Christophe Braillon, Pierre Bessière, and Christian Laugier. The cycab: a car-like robot navigating autonomously and safely among pedestrians. *Robotics and Autonomous Systems*, 50(1):51–67, 2005

The CyCab: a car-like robot navigating autonomously and safely among pedestrians

Cédric Pradalier, Jorje Hermosillo, Carla Koike, Christophe Braillon, Pierre Bessiere, Christian Laugier

The *CyCab*: a Car-Like Robot Navigating Autonomously and Safely Among Pedestrians

Cédric Pradalier, Jorge Hermosillo, Carla Koike,
Christophe Braillon, Pierre Bessière, Christian Laugier

firstname.lastname@inrialpes.fr
GRAVIR – INRIA – INPG Grenoble
INRIA Rhône-Alpes, 38334 Saint Ismier cedex France

Abstract

The recent development of a new kind of public transportation system relies on a particular double-steering kinematic structure enhancing manoeuvrability in cluttered environments such as downtown areas. We call *bi-steerable car* a vehicle showing this kind of kinematics. Endowed with autonomy capacities, the bi-steerable car ought to combine suitably and safely a set of abilities: simultaneous localisation and environment modelling, motion planning and motion execution amidst moderately dynamic obstacles. In this paper we address the integration of these four essential autonomy abilities into a single application. Specifically, we aim at reactive execution of planned motion. We address the fusion of controls issued from the control law and the obstacle avoidance module using probabilistic techniques.

Key words: Car-like robot, navigation, path planning, obstacle avoidance, autonomous navigation.

1 Introduction

The development of new Intelligent Transportation Systems (ITS), more practical, safe and accounting for environmental concerns, is a technological issue of highly urbanised societies today [18]. One of the long run objectives is to reduce the use of the private automobile in downtown areas, by offering new modern and convenient public transportation systems. Examples of these, are the CyCab robot – designed at INRIA and currently traded by the Robosoft company (see www.robosoft.fr) – and the pi-Car prototype of IEF (Institut d'Electronique Fondamentale, Université Paris-Sud).

The kinematic structure of these robots differs from that of a car-like vehicle in that it allows the steering of both the front axle and the rear one. We call a vehicle showing this feature a bi-steerable car (or BiS-car for short).

Endowed with autonomy capacities, the bi-steerable car ought to combine suitably and safely a set of abilities that eventually could come to the relief of the end-user in complex tasks (e.g. parking the vehicle). Part of these abilities have been tackled separately in previous work: simultaneous localisation and environment modelling, motion planning execution amidst static obstacles and obstacle avoidance in a moderately dynamic environment without accounting for a planned motion.

In this paper we address the integration of these four essential autonomy abilities into a single application. Specifically, we aim at reactive execution of planned motion. We address the fusion of controls issued from the control law and the obstacle avoidance module using probabilistic techniques. We are convinced that these results represent a step further towards the motion autonomy of this kind of transportation system. The structure of the paper follows.

In section 2, we sketch the environment reconstruction and localisation methods we used and we recall how the central issue regarding the motion planning and execution problem for the general BiS-car was solved. Section 3 explains how our obstacle avoidance system was designed and section 4 how it was adapted to the trajectory tracking system. In section 5 we present experimental settings showing the fusion of these essential autonomy capacities in our bi-steerable platform the CyCab robot. We close the paper with some concluding remarks and guidelines on future work in section 6.

2 Localisation, Environment modelling, Motion planning and execution

In the design of an autonomous car-like robot, we are convinced that localisation, modelling of the environment, path planning and trajectory tracking are of fundamental importance.

2.1 Map-building and Localisation

The CyCab robot is the size of a golf-cab capable of attaining up to 30Km/h. Its “natural” environment is the car-park area of the INRIA Rhône-Alpes (about $10000m^2$). For localisation purposes, we did not want to focus on the detection of natural features in the environment, since such detection is often subject to failure and not very accurate. So, in order to ensure reliability, we decided to install artificial landmarks in the environment. These landmarks had to be detected easily and

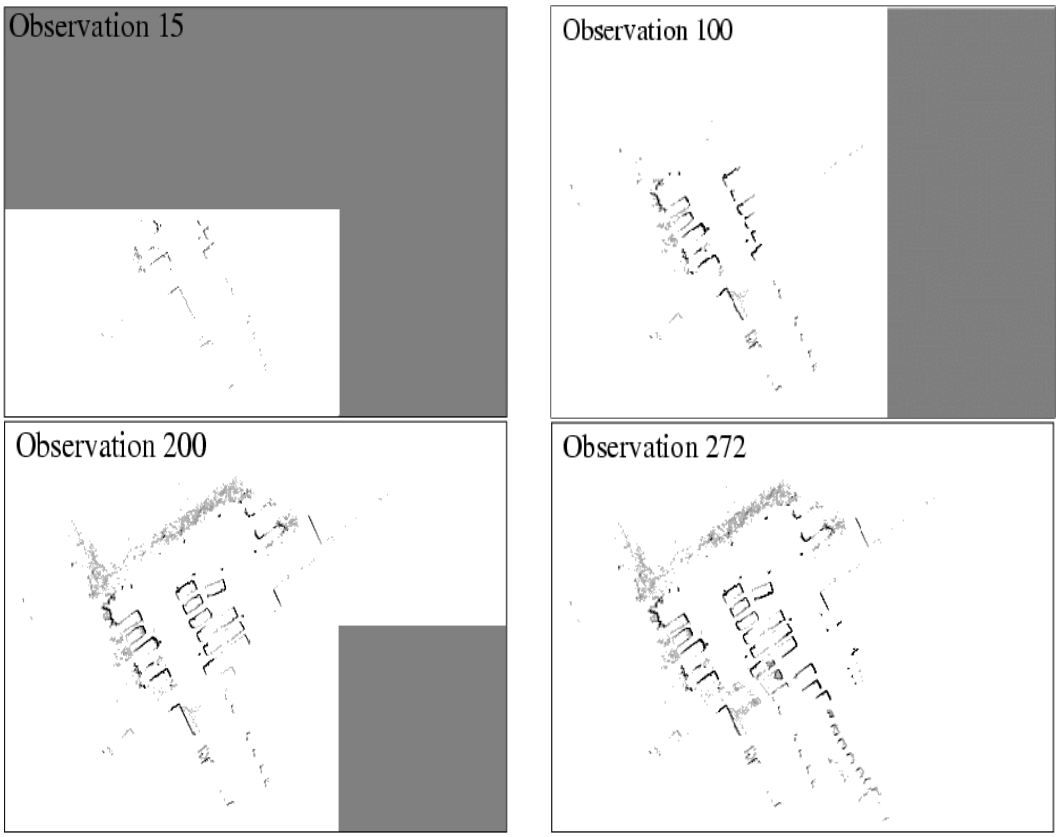


Fig. 1. Obstacle map evolution: Experimental images during the obstacle map-building phase. The vehicle is driven within the car-park area as long as needed. Simultaneously, the laser range sensor is used to detect the landmarks to build-up the localisation map.

accurately, and they should be identified with a reasonable computation effort. Fig. 2 shows our robot, its sensor and the landmarks : cylinder covered with reflector sheets, specially designed for our Sick laser range finder.

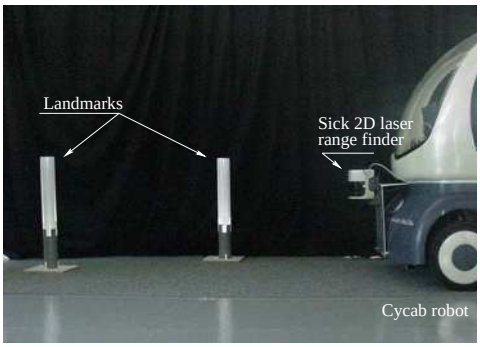


Fig. 2. Cycab robot and its landmarks for localization

Moreover, in order to keep flexibility, we wanted to be able to equip the environment with non permanent beacons. For this reason, we could not rely on a definitive landmark map, and we had to build a system able to learn the current state of the

car-park area. This led us to use SLAM [1] methods. The method which was best suited to our needs was the Geometric Projection Filter (see [21] for reference, and [24] for implementation details). It consists in building a map of features uncorrelated with the robot state. Such features are, for instance, the distance between landmarks or angles between three of them.

Owing to the accuracy of the laser range finder, to the good choice of our landmarks, and to the strength of the SLAM methods we use, we evaluate the worst case accuracy of our localisation system to the following value: about 10 centimetres in position and 2 degrees in orientation. We refer the reader to [24] for more details about the way we evaluate these values.

2.2 The Obstacle Map

The previous method localises the robot and builds a landmark map. But, we still miss a map of observed obstacles in order to plan safe paths. To achieve this goal, we build a kind of simplified occupancy grid[8] on the environment. This structure gives us informations correlated with the probability that a given place is the boundary of an obstacle.

Both maps are built online, in real-time, by the robot during the construction phase. Fig. 1 shows how the obstacle map evolves while we are exploring the environment. This map is made of small patches which are added according to the need of the application. In this way, the map can be extended in any direction, as long as memory is available. Once the map-building phase has finished, the obstacle map is converted into a pixmap and passed to the Motion Planning stage.

2.3 Motion Planning Amidst Static Obstacles

The Motion Planner adopted for the CyCab was presented in [26]. Essentially, it is a two step approach, dealing separately with the physical constraints (the obstacles) and with the kinematic constraints (the non-holonomy). The planner first builds a collision-free path without taking into account the non-holonomic constraints of the system. Then, this path is approximated by a sequence of collision-free feasible sub-paths computed by a *suitable* [2] steering method. Finally, the resulting path is smoothed.

A key issue in non-holonomic motion planning is to find a steering method accounting for the kinematics of the robot. One way of designing steering methods for a

[1] Simultaneous Localisation And Mapping

[2] i.e. Verifying the topological property as explained in [26].

non-holonomic system is to use its ***flatness*** property [10] allowing also for feedback linearisation of the nonlinear system (this is discussed in section 2.6). This is what we did for the general BiS-car for which a flat output—or linearising output—was given in [26].

2.4 Steering a BiS-car

The kinematics model of a general bi-steerable vehicle and its flat output are shown in Fig. 3.

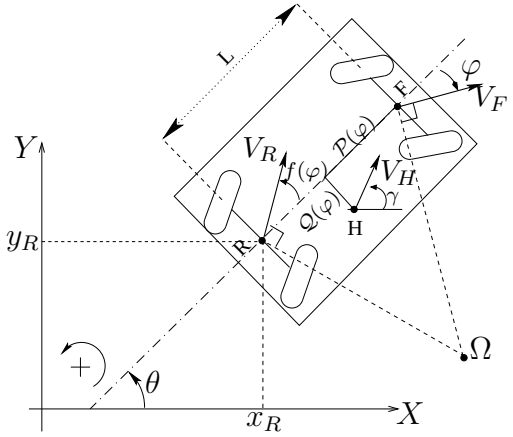


Fig. 3. Cycab robot, its landmarks and its kinematics model showing the coordinates of the flat output (point H) with respect to the reference frame of the robot placed at point F. In our case we have that $(x_{\scriptscriptstyle F}, y_{\scriptscriptstyle F}, \theta, \varphi)$ is the state of the robot.

The striking advantage of planning a path in the flat space is that we only need to parameterise a 2-dimensional curve whose points and derivatives define everywhere the current n-dimensional state[3] of the robot (in the case of the BiS-car $n = 4$). The main characteristic of such a curve is its curvature κ from which the steering angle can be computed.

Fig. 4 shows the outcome of the motion planner using an obstacle map generated as described in the previous section.

2.5 User-Planner Interface

The User-Planner interface in the CyCab is achieved through a ***touch-screen*** superposed to a 640×480 pixels LCD display. Additionally, we use the keyboard to allow for the entrance of data.

[3] The configuration space in robotics is called the ***state space*** in control theory, so we will use indistinctly both terms.

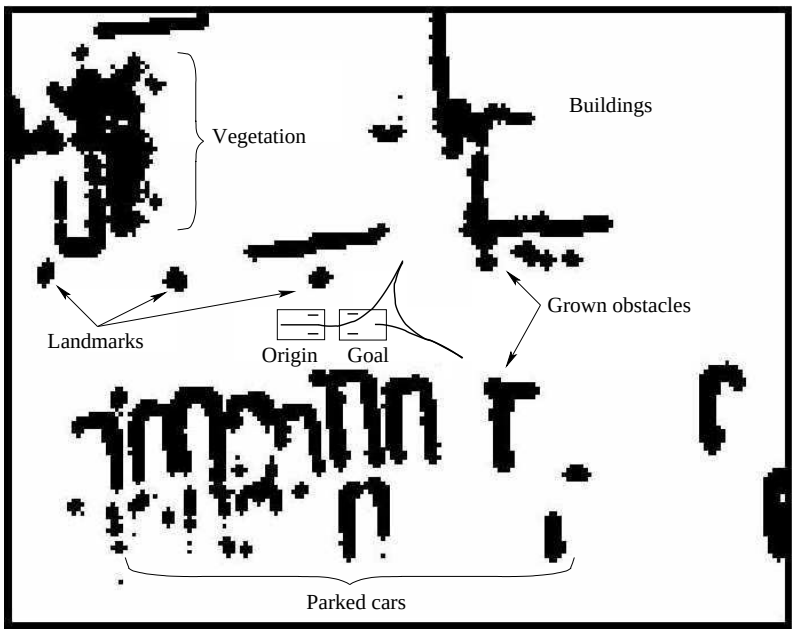


Fig. 4. Path computed by the motion planner using a real obstacle map. The obstacles are grown as well as the robot before computing the path.

The interface is used to display the current position of the robot within its environment and to capture the goal position entered by the user. These positions together with the obstacle map is passed to the motion planner. The output path is then displayed allowing the user to validate the path or start a new search.

Finally, the reference trajectory is generated using a regular parameterisation of the path [16] and the user is requested to accept to start the execution of the trajectory.

2.6 Trajectory tracking using flatness

It is well known that a non-holonomic system cannot be stabilised using only smooth state static feedbacks [6]. Ever since then, time-varying feedbacks [25] and dynamic feedbacks have been successfully used in particular for the canonical tractor-trailer and car-like robots [9].

Flat systems are feedback linearisable by means of a restricted class of dynamic feedback called *endogenous* [10]. The interest is that we are able to use state-of-the-art linear control techniques to stabilise the system. We present here results coming from recent work on feedback linearisation of the general BiS-car.

For a reference frame of the robot placed at point F in Fig. 3, the flat output $\mathbf{y} = (y_1, y_2)^T$ of a BiS-car are the coordinates of a point $H = (x_H, y_H)^T = (y_1, y_2)^T$,

computed as a function of the state as follows:

$$H = F + \mathcal{P}(\varphi)\vec{u}_\theta + \mathcal{Q}(\varphi)\vec{u}_{\theta^\perp}$$

where $\mathcal{P}(\varphi)$ and $\mathcal{Q}(\varphi)$ are coordinate functions relative to the robot's reference frame (see [26] for details) and where $\vec{u}_\theta$ (resp. $\vec{u}_{\theta^\perp}$) is the unitary vector in the direction θ (resp. the direction $\theta + \frac{\pi}{2}$).

Looking for a tractable relation between the controls of the robot and the linearising output, we found an expression giving the flat output dynamics with respect to a more convenient reference frame placed at the middle of the front axle of the robot (point F) and having orientation $\gamma = [\theta + \beta(\varphi)] \pm \pi$ where the function $\beta(\varphi)$ is the characteristic angle of the velocity vector of the flat output.

The convenience of this new reference frame relies on the fact that the velocity of the flat output has a single component in it. More precisely—assuming that $\gamma = \theta + \beta(\varphi) + \pi$—one can show that, in this reference frame, the flat output dynamics is given by the following expression [14]:

$$\begin{aligned} \frac{\partial H}{\partial t} &= v_{_H}\, \vec{u}_\gamma \\ v_{_H} &= v_{_F}[\cos(\varphi - \beta - \pi) - \mathcal{Q}\mathcal{F}] + \omega_\varphi[\frac{\partial \mathcal{P}}{\partial \varphi} - \frac{\partial \beta}{\partial \varphi}\mathcal{Q}] \\ \mathcal{F}(\varphi) &= \frac{\sin(\varphi - f(\varphi))}{L\cos(f(\varphi))} \end{aligned} \tag{1}$$

where $(v_{_F}, \omega_\varphi)$ are the controls of the robot (i.e. the heading and the front-steering speeds), $(\varphi - \beta - \pi)$ is the angle subtended between the velocity vector of the robot $\vec{V}_F$ and the velocity vector of the flat output $\vec{V}_H$ (see Fig. 3).

From expression (1) the open-loop controls of the robot can be found as soon as the trajectory of point H is known. As we are interested in stabilising the BiS-car around a reference trajectory, we explored the fact that, owing to the flatness property, the system is diffeomorphic to a linear controllable one [10]. The endogenous dynamic feedback that linearises the general bi-steerable system is presented in [14]. Then, from linear control theory, it can be shown that the closed-loop control stabilising the reference trajectory $\mathbf{y}^*$ has the following form :

$$y_i^{(3)} = (y_i^*)^{(3)} - \sum_{j=0}^{2} k_{i,j}\left(y_i^{(j)} - (y_i^*)^{(j)}\right) \quad i = 1, 2 \tag{2}$$

Where $(.)^{(p)}$ stands for the total derivative of order p. See [7] for details.

3 Obstacle avoidance using probabilistic reasoning

The previous approach considers trajectories in a static environment. In order to make the execution of these trajectories more robust, an obstacle avoidance system should be prepared to react to unpredicted changes in the environment. This section presents the principles of our obstacle avoidance module.

3.1 State of the art on reactive trajectory tracking

Most of the approaches for obstacle avoidance are local([11,5]), that is they do not try to model the whole environment. They goal is rather to use sensor measures to deduce secure commands. Being simpler and less computationnaly intensive, they seem more appropriate to fast reactions in a non-static environment. On the other hand, we can not expect optimal solutions from a local method. It is possible that some peculiar obstacle configuration create a dead-end from which the robot cannot escape with obstacle avoidance only.

3.1.1 Potential fields

The general idea of potential fields methods, proposed initially by O. Khatib in 1986, is to build a function representing both the navigation goals and the need for obstacle avoidance. This function is built so has to decrease when going closer to the goal and to increase near obstacles. Then, the navigation problems is reduced to an optimisation problem, that is, to find the commands that brings the robot to the global minimum of the function. This later can be defined with respect to the goal and the obstacles but other constraints can also be added therein.

Numerous extensions to the potential fields have been proposed since 1986. Among others, we can cite the Virtual Force Fields [3], the Vector Field Histograms [4] and their extensions VFH+[28] and VFH*[29]. Basically, these methods try to find the best path to the goal among the secure ones.

3.1.2 Steering Angle Field (SAF)

The SAF method, proposed by *Feiten et al.* in 1994, use obstacles to constrain steering angle in a continuous domain. Simultaneously, speed control is an iterative negociation process between the high-level driving module and the local obstacle-avoidance module.

One of the first extension to this method was published in [27]. It express the collision avoidance problem as an optimisation problem in the robot controls space

(linear and rotational speeds).

3.1.3 Dynamic Window

The Dynamic Window approach[11] propose to avoid obstacles by exploring command space in order to maximise an objective function. This later accounts for the progression toward the goal, the position of closer obstacles and current robot controls. Being directly derived from the robot dynamic, this methode is particularly well adapted to high speed movements.

The computational cost of the optimization process is reduced using the dynamic caracteristics of the robot (bounded linear and angular acceleration) so as to reduce the searched space. This kind of constraints are called *Hard Constraints* since the must be respected. Conversely, when the objective function includes preferences on the robot movement, we call the resulting constraints *Soft Constraints*.

3.1.4 Dynamic environments and Velocity Obstacles

In the specific case of moving obstacles, special methods have been proposed[17,2] using the *Velocity Obstacle* notion. Basically, this notion consist in projecting perceived obstacles and their expected movement in the space of secure commands. So, each mobile object generates a set of obstacles in the command space. These obstacles represent the commands that will bring to a collision in the future.

In the general case, obstacle movement parameters are not known *a priori*, so they have to be deduced from sensor data. Obstacle avoidance controls are then computed in reaction to theses previsions. Currently, it is still quite difficult to get reliable previsions of the obstacles future trajectory. Consequently, these obstacle avoidance methods are not appliable in real situations yet.

3.1.5 Obstacle avoidance and trajectory following

When we want to perform obstacle avoidance manoeuvres while following a trajectory, a specific problem appear. On our non-holonomous robot, the path planning stage took into account the kinematic of the robot and planned a feasible path. When the reactive obstacle avoidance generates commands, the vehicle leaves its planned trajectory. Then, we cannot be sure anymore that the initial objective of the trajectory is still reachable.

A solution to this problem was proposed in [20]. This method tries to deform the global trajectory in order to avoid the obstacle, respect the kinematic constraints and ensure that the final goal is still reachable. Even if theoretically very interesting, this obstacle avoidance scheme is still difficult to apply in real situations

due to it computational complexities, especially on an autonomous car. In our experiments[20], the vehicle had to stop for several minutes in order to perform the trajectory deformation

3.2 Objectives

After all these results on obstacle avoidance, it seems obvious that our goal is not to propose a new solution to this problem. It has been shown[19,1] that probalities and bayesian inference are appropriate tools to deal with real world uncertainty and to model reactive behaviors. We this in mind, we wanted to think about the expression of the obstacle avoidance problem as a bayesian inference problem. Consequently, the originality of our approach is mainly its expression and the semantic we can express with it.

3.3 Specification

The CyCab can be commanded through a speed V and a steering angle Φ. It is equipped with π radians sweeping laser range finder. In order to limit the volume of the data we manipulate, we summarised the sensor output as 8 values : the distances to the nearest obstacle in a $\pi/8$ angular sector(see Fig. 5). We will call $D_k, k = 1 \dots 8$ the probabilistic variables corresponding to these measures.

Besides, we will assume that this robot is commanded by some high-level system (trajectory following for instance) which provides it with a pair of desired commands (V_d, Φ_d).

Our goal is to find commands to apply to the robot, guarantying the vehicle security while following the desired command as much as possible.

3.4 Sub-models definition

Given the distance D_i measured in an angular sector, we want to express a command to apply that is safe while tracking desired command. Nevertheless, since this sector only has limited information about robot surrounding, we choose to express the following conservative semantic: tracking the desired command should be a soft constraint whereas an obstacle avoidance command should be a hard constraint, the closer the obstacle, the harder the constraint.

We express this semantic using a probability distribution over the commands to apply (V, Φ) knowing the desired commands and the distance D_i measured in this

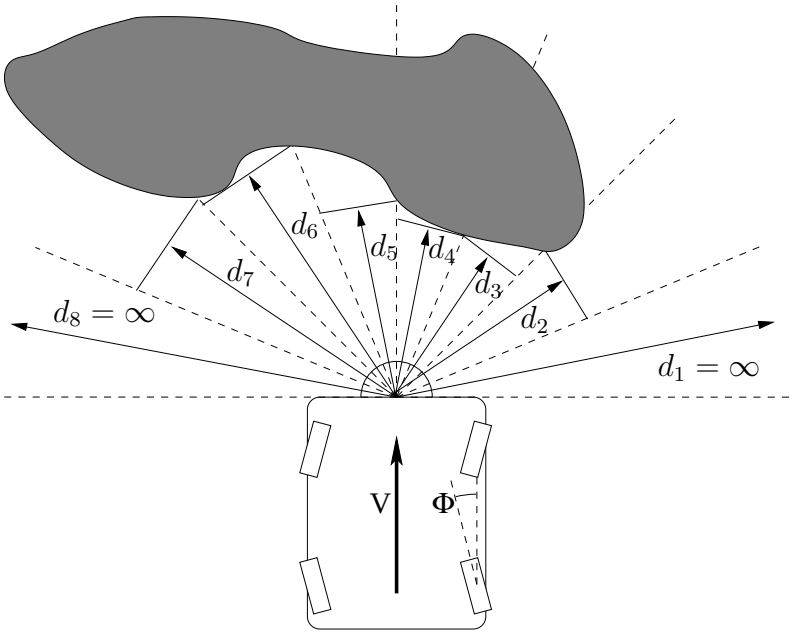


Fig. 5. Obstacle avoidance: situation

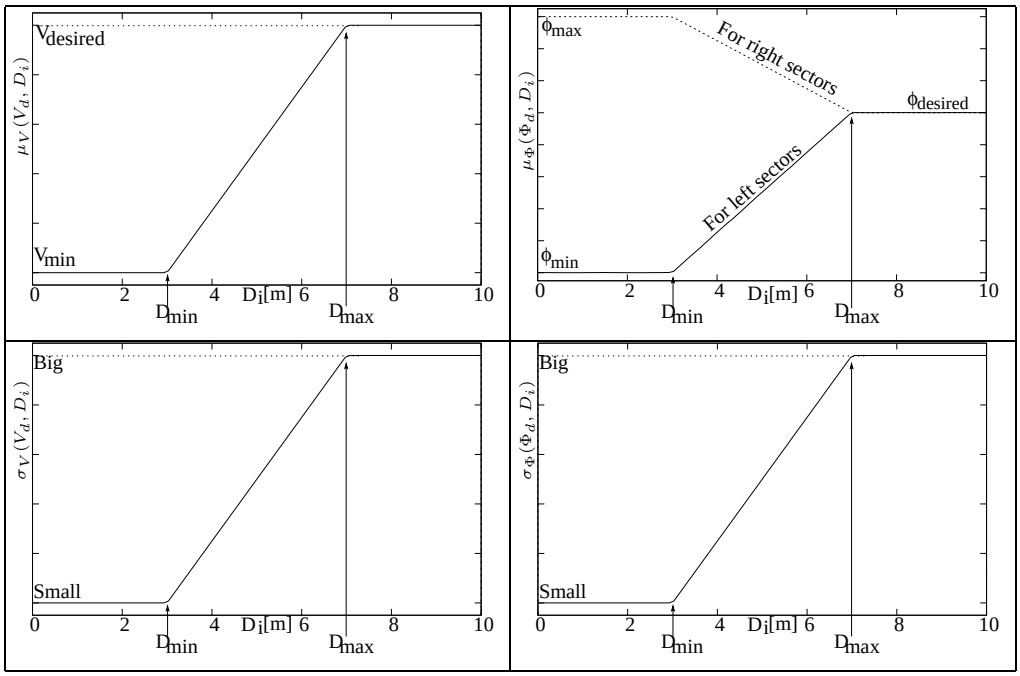


Fig. 6. Evolution of mean and standard deviation of $P_i(V \mid V_d D_i)$ and $P_i(\Phi \mid \Phi_d D_i)$ according to distance measured

sector:

$$P_i(V\Phi \mid V_d\Phi_d D_i) = P_i(V \mid V_d D_i) P_i(\Phi \mid \Phi_d D_i) \tag{3}$$

where $P_i(V \mid V_d D_i)$ and $P_i(\Phi \mid \Phi_d D_i)$ are Gaussian distributions respectively centred on $\mu_V(V_d, D_i)$ and $\mu_\Phi(\Phi_d, D_i)$ with standard deviation $\sigma_V(V_d, D_i)$ and $\sigma_\Phi(\Phi_d, D_i)$. Functions $\mu_V, \mu_\Phi, \sigma_V, \sigma_\Phi$ are defined with sigmoid shape as illustrated in Fig. 6. Example of resulting distributions are shown in Fig. 7.

There is two specific aspects to notice in Fig. 6 and 7. First, concerning the means μ_V and μ_Φ, we can see that, the farther the obstacle, the closer to the desired command μ will be, and conversely, the nearer the obstacle, the more secure μ: minimal speed, strong steering angle.

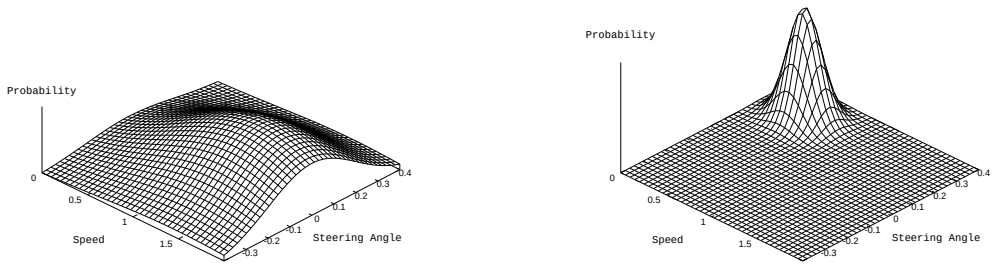


Fig. 7. Shape of $P_i(V\Phi \mid V_d\Phi_d D_i)$ for far and close obstacles

Second, the standard deviation can be seen as a constraint level. For instance, when an obstacle is very close to the robot (small D_i), its speed *must* be strongly constrained to zero, this is expressed by a small standard deviation. Conversely, when obstacle is far, robot speed *can* follow the desired command, but there is no damage risk in not applying exactly this command. This low level constraint is the result of a big standard deviation.

3.5 Command fusion

Knowing desired controls and distance to the nearest obstacle in its sector, each sub-model, defined by $P_i(V\Phi \mid V_d\Phi_d D_i)$, provides us with a probability distribution over the robot controls. As we have eight sectors, we will have to fuse the controls from eight sub-models. Then we will find the best control in term of security and desired control following.

To this end, we define the following joint distribution:

$$\begin{aligned} P(V\ \Phi\ V_d\ \Phi_d\ D_1 \ldots D_8\ S) = {} & P(D_1 \ldots D_8)\ P(V_d\ \Phi_d) \\ & P(S)\ P(V\ \Phi\ \mid\ V_d\ \Phi_d\ D_1 \ldots D_8\ S) \end{aligned} \tag{4}$$

where variable $S \in [1 \ldots 8]$ express which sector is considered. $P(D_1 \ldots D_8)$ and $P(V_d\Phi_d)$ are unknown distribution [4]. As there is no need to favour a specific sub-model, we define $P(S)$ as a uniform distribution. The semantic of S will be em-

[4] Actually, as we know we will not need them in future computation, we don't have to specify them.

phasised by the definition of $P(V\Phi \mid V_d\Phi_d D_1 \ldots D_8 S)$:

$$P(V\Phi \mid V_d\Phi_d D_1 \ldots D_8[S=i]) = P_i(V\Phi \mid V_d\Phi_d D_i)$$

In this equation, we can see that the variable S acts as model selector: given its value i, the distribution over the commands will be computed by the sub-model i, taking into account only distance D_i.

Using equation 4, we can now express the distribution we are really interested in, that is the distribution over the commands accounting for all the distances but not variable S:

$$P(V\ \Phi\ \mid\ V_d\ \Phi_d\ D_1 \ldots D_8) = \sum_S (P(S)P(V\ \Phi\ \mid\ V_d\ \Phi_d\ D_1\ \ldots\ D_8\ S)) \quad (5)$$

This equation is actually the place where the different constraint level expressed by functions σ_V and σ_Φ will be useful. The more security constraints there will be, the more peaked will be the sub-model control distribution. So sub-models who see no obstacles in their sector will contribute to the sum with quasi-flat distribution, and those who see perilous obstacles will add a peaky distribution, hence having more influence (see Fig. 8). Finally the command really executed by the robot is the one which maximise $P(V\ \Phi\ \mid\ V_d\ \Phi_d\ D_1 \ldots D_8)$ (eq. 5).

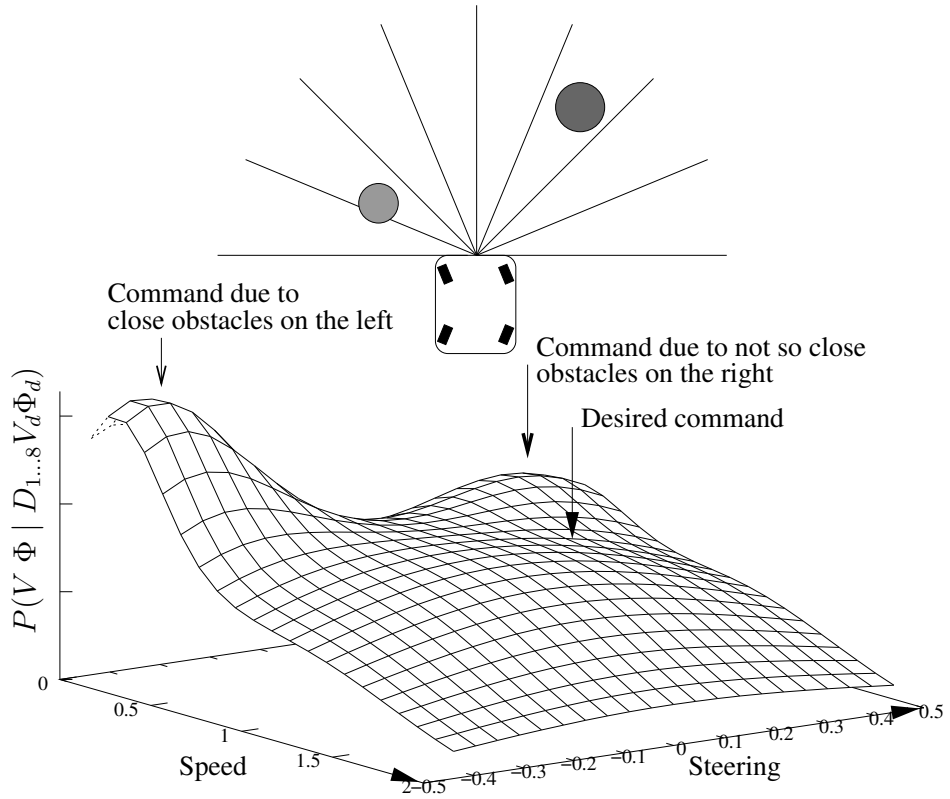


Fig. 8. Probability distribution over speed and steering, resulting from the obstacle avoidance system.

Fig. 9 illustrates the result of the obstacle avoidance system applied on a simulated example. The simulated CyCab is driven manually with a joystick in a square environment. In this specific situation, the driver is continuously asking for maximum speed, straight forward (null steering angle). We can observe on the dotted trajectory that, first obstacle avoidance module bends the trajectory in order to avoid the walls, and second, when there is no danger of collisions, desired commands are applied exactly as requested.

From the density of dots, we can figure out the robot speed: it breaks when it comes close to the walls and while its turning and try to follow desired speed when obstacles are not so threatening.

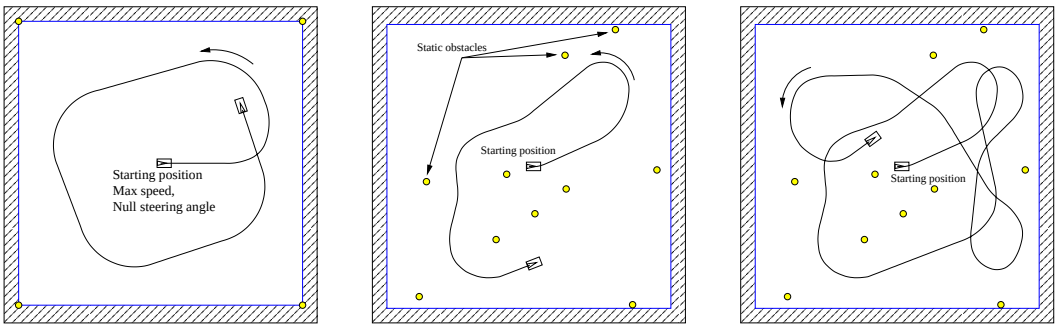


Fig. 9. Robot trajectory while driven manually with constant desired steering angle

3.7 Relation to fuzzy logic approaches

The design of our obstacle avoidance modules may remind some readers of a fuzzy logic controller[15,22,12]. It is rather difficult to say that one approach is better than the other. Both fuzzy logic and bayesian inference view themselves as extension of classical logic. Furthermore, both methods will deal with the same kind of problems, providing the same kind of solutions. Some will prefer the great freedom of fuzzy logic modelling and others will prefer to rely on the strong mathematical background behind bayesian inference.

As far as we can see, the choice between fuzzy logic and bayesian inference is rather an personal choice, similar to the choice of a programming language: it has more consequences on the way we express our solution than on the solution itself. To extend the analogy, one might relate fuzzy logic to the C language whereas Bayesian inference would be closer to Ada.

4 Trajectory tracking with obstacle avoidance

The method presented in the previous section provides us with an efficient way to fuse a security system and orders from a high level system. Nevertheless the perturbations introduced in the trajectory following system by obstacle avoidance are such that they can make it become unstable. In this section will show how we integrate trajectory tracking and obstacle avoidance.

While following the trajectory, obstacle avoidance will modify certain commands in order to follow as much as possible desired orders while granting security. These modifications may introduce delay or diversions in the control loop. If no appropriate action is taken to manage these delays the control law may generate extremely strong accelerations or even become unstable when obstacles are gone. This is typically the case when our system evolves among moving pedestrians. Thus we designed a specific behaviour to adapt smoothly our control system to the perturbations induced by obstacle avoidance.

4.1 Multiplexed trajectory tracking

4.1.1 Validity domain of flat control law

Experimentally, we found that the control law based on flatness can manage errors in a range of about 1 meter and 15 degrees around nominal trajectory. Furthermore, as this control law controls the third derivative of the flat output (eq. 2), it is a massively integrating system. For this reason, a constant perturbation such as immobilisation due to a pedestrian standing in front of the vehicle will result in a quadratic increase of the control law output. This phenomena is mainly due to the fact that when obstacle avoidance slows the robot down, it strongly breaks the dynamic rules around which the flat control law was built. So, there is no surprise in its failure.

4.1.2 Probabilistic control law

In order to deal with the situations that flat control law cannot manage, we designed a trajectory tracking behaviour (*TTB*) based again on probabilistic reasoning (section 4.2). As this behaviour has many similarities with a weighted sum of proportional control laws, we do not expect it to be sufficient to stabilise the robot on its trajectory. Nevertheless, it is sufficient to bring it back in the convergence domain of the flat control law when obstacle avoidance perturbations have occurred. Basically, the resulting behaviour is as follows: while the robot is close to its nominal position, it is commanded by flat control law. When, due to obstacle avoidance, it is too far from its nominal position, TTB takes control, and try to bring it back to

flat control law's convergence domain. When it enters this domain, flat control law is reinitialised and starts accurate trajectory tracking(this is illustrated in fig. 10).

4.1.3 Time control

Path resulting from path planning (section 2.3) is a list of robot configuration indexed by time. So when the robot is slowed down by a traversing pedestrian, it compensates its delay by accelerating. Nevertheless, when the robot is stopped during a longer time, let's say fifteen seconds, it should not consider to be delayed of fifteen seconds, otherwise it will try to reach a position fifteen second ahead, without tracking the intermediary trajectory. To tackle this difficulty, we introduced a third mode to the trajectory tracking: when the robot comes too far from its nominal position, we freeze the nominal position, and we use the TTB to reenter the domain where nominal position can be unfrozen.

The global system is illustrated by Fig. 10: we implemented some kind of multiplexer/demultiplexer which manage transitions between control laws. In order to avoid oscillating between control laws when at the interface between two domains of validity, we had to introduce some hysteresis mechanism in the switching. This is illustrated in Fig. 10.

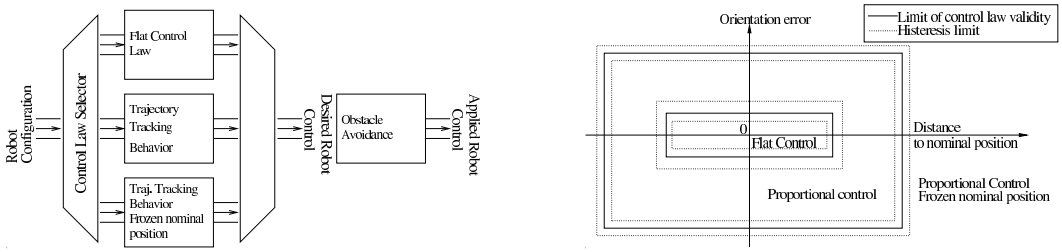


Fig. 10. Basic diagram of the control law selector mechanism and validity domains of the control laws

4.2 Trajectory tracking behaviour

Our trajectory tracking behaviour was built as a probabilistic reasoning, in a way similar to the obstacle avoidance presented above (section 3). Functionnaly, it is very similar to a fuzzy control scheme as presented in [15] and illustrated in [12].

To specify our module, we use a mechanism of fusion with diagnosis[23]. If A and B are two variables, we will define a diagnosis boolean variable I_A^B which express a consistency between A and B. Then, A and B will be called the ***diagnosed variables*** of I_A^B.

Our goal is to express the distribution over the desired controls (V_d, Φ_d) knowing reference controls (V_r, Φ_r) planned by the path planning stage, and error in position

$(\delta X, \delta Y)$ and orientation $\delta\theta$ with respect to the nominal position. Fig. 11 illustrates theses variables.

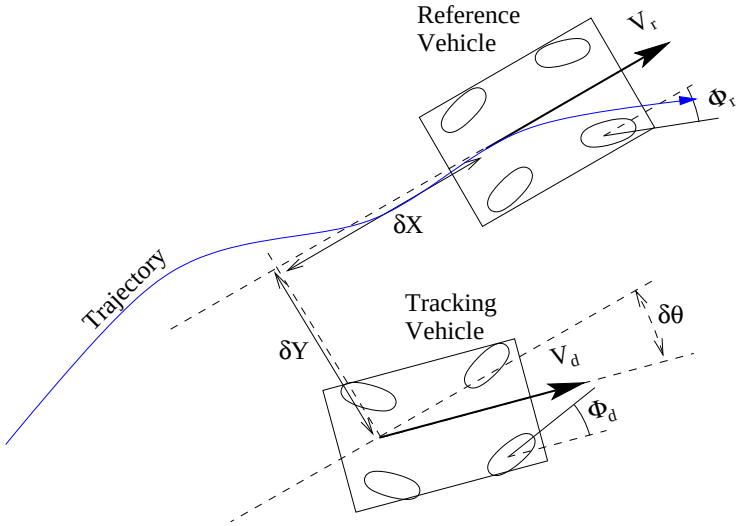


Fig. 11. Variables involved in trajectory tracking behaviour

In addition to the preceding variables, we will add five diagnosis variables $I^{\delta X}_{V_d}$, $I^{V_r}_{V_d}$, $I^{\delta Y}_{\Phi_d}$, $I^{\delta\theta}_{\Phi_d}$ and $I^{\Phi_r}_{\Phi_d}$. Variables linked to an error variable $(\delta X, \delta Y, \delta\theta)$ will diagnose if a given command helps correcting this error. Variables linked to reference commands evaluate if a command is similar to the reference one.

All these variables describe the relation between their diagnosed variables in the following joint distribution:

$$
\begin{aligned}
P(V_d\,\Phi_d\,V_r\,\Phi_r\,\delta X\,\delta Y\,\delta\theta\;I^{\delta X}_{V_d}\,I^{V_r}_{V_d}\,I^{\delta Y}_{\Phi_d}\,I^{\delta\theta}_{\Phi_d}\,I^{\Phi_r}_{\Phi_d}) = \\
P(V_d\;\Phi_d)\;P(V_r\;\Phi_r)\;P(\delta X\;\delta Y\;\delta\theta) \\
P(I^{\delta X}_{V_d}\mid V_d\;\delta X)\;P(I^{V_r}_{V_d}\mid V_d\;V_r) \\
P(I^{\delta Y}_{\Phi_d}\mid \Phi_d\;\delta Y)\;P(I^{\delta\theta}_{\Phi_d}\mid \Phi_d\;\delta\theta\;V_d)\;P(I^{\Phi_r}_{\Phi_d}\mid \Phi_d\;\Phi_r)
\end{aligned}
\tag{6}
$$

Using this joint distribution and Bayes rule, we will be able to infer

$$
\begin{aligned}
P(V_d\,\Phi_d\,|\,(V_r\,\Phi_r)\;(\delta X\,\delta Y\,\delta\theta) \\
[I^{\delta X}_{V_d}=1]\;[I^{V_r}_{V_d}=1]\;[I^{\delta Y}_{\Phi_d}=1]\;[I^{\delta\theta}_{\Phi_d}=1]\;[I^{\Phi_r}_{\Phi_d}=1])
\end{aligned}
\tag{7}
$$

Basically, this equation expresses the fact that we are looking for the most likely commands in order to correct tracking error while accounting for reference commands. Having all the diagnosis variables set to one enforces this semantic.

In the preceding joint distribution (eq. 6), all the diagnosed variables are assumed to be independent, and to have uniform distributions. All the information concerning the relation between them will be encoded in the distribution over diagnosis

variables. In order to define this distributions, we first define the function $d_\sigma(x, y)$ as a Mahalanobis distance between x and y:

$$d_\sigma(x, y) = e^{-\frac{2}{2}\left(\frac{x-y}{\sigma}\right)^2}$$

Then, for two variables A and B, we define

$$P([I_A^B = 1] \mid AB) = d_{S(A,B)}(A, f(B)).$$

Let's see how preceding functions S and f are defined in specific cases.

4.2.1 Proportional compensation of errors

In the case of $I_{V_d}^{\delta X}$, we set $f(\delta X) = \alpha.\delta X$ and

$$S(V_d, \delta X) = \max((1 - \beta.\delta X)\sigma_{\max}\ ,\ \sigma_{\min}).$$

Expression of f implies that the maximum of $P(I_{V_d}^{\delta X} \mid V_d\ \delta X)$ will be for a value of V_d proportional to the error δX. Expression of S defines the constraint level associated to this speed: the bigger the error, the more confident we are that a proportional correction will work, so the smaller σ.

The basic behaviour resulting from this definition is that when the robot is behind it nominal position, it will move forward to reduce its error: the bigger its error, the faster and with more confidence that this is the good control to apply.

For $I_{\Phi_d}^{\delta Y}$, we use a similar proportional scheme. Its basic meaning is that when the robot has a lateral error, it has to steer, left or right, depending on the sign of this error. Again, the bigger the error, the more confident we are that we have to steer.

Finally, the same apply for $I_{\Phi_d}^{\delta\theta}$, except that the steering direction depends not only of the orientation error, but also of the movement direction V_d.

4.2.2 Using planned controls

In the path planning stage, the trajectory was defined as a set of nominal position, associated with planned speed and steering angle. They have to be accounted for, especially when error is small.

Let's consider first $I_{V_d}^{V_r}$. We set f and S as follows: $f(V_r) = V_r$ and $S(V_d, V_r) = \sigma_{V_r} \in [\sigma_{\min}, \sigma_{\max}]$, rather close to $\sigma_{\max}$. By this way, planned speed is used as a indication to the trajectory following system. The distribution over $I_{\Phi_d}^{\Phi_r}$ is defined using the same reasoning.

4.3 Results

Fig. 12 illustrates the basic behaviour of our trajectory tracking behaviour. In both graphs, desired command will maximise either $P(V \mid \delta X\ V_c)$ or $P(\Phi \mid \delta Y\ \delta\theta\ \Phi_c)$. Since curve $P(V \mid \delta X\ V_c)$ is closer to $P(V \mid \delta X)$ than to $P(V \mid V_c)$, we can observe that longitudinal error (δX) has much more influence than reference command on the vehicle speed. In the same manner, steering angle is a trade-off between what should be done to correct lateral error (δY) and orientation error ($\delta\theta$), lightly influenced by reference steering angle.

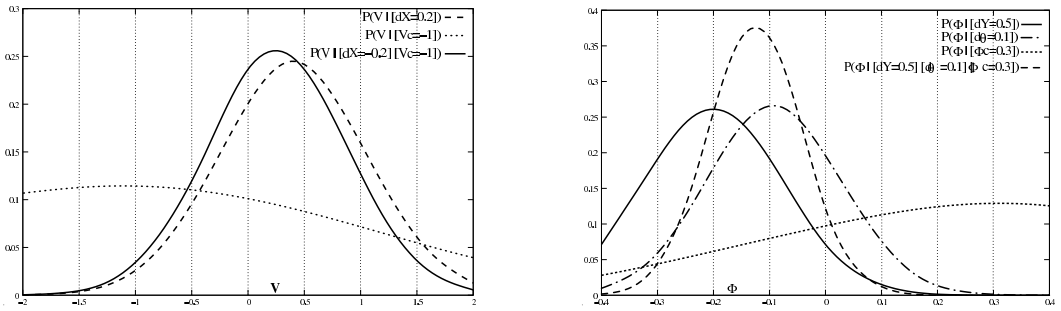


Fig. 12. Trajectory tracking : resulting command fusion

Fig. 13 shows the collaboration of obstacle avoidance and trajectory following on a simulated example. Planned trajectory passes through an obstacle which was not present at map building time. Obstacle avoidance modifies controls in order to grant security. When errors with respect to nominal trajectory is too big, our control law selector switch to the trajectory tracking behaviour. Here it is a big longitudinal error, due to obstacle avoidance slowing down the vehicle, which trigger the switching.

4.4 Discussion

Using the multiplexed control laws we managed to integrate, in the same control loop, our flat control, accurate but sensible to perturbation, with our TTB, less accurate but robust to perturbations. By this way we obtained a system capable of tracking trajectory generated by our path planner while accounting for unexpected object in the environment.

Finally, when the robot has gone too far from reference trajectory, or when reactive obstacle avoidance can not find suitable controls anymore, it may be necessary to re-plan a new trajectory to the goal. This has not been implemented on the robot yet, but this should not be considered neither a technical nor a scientific issue.

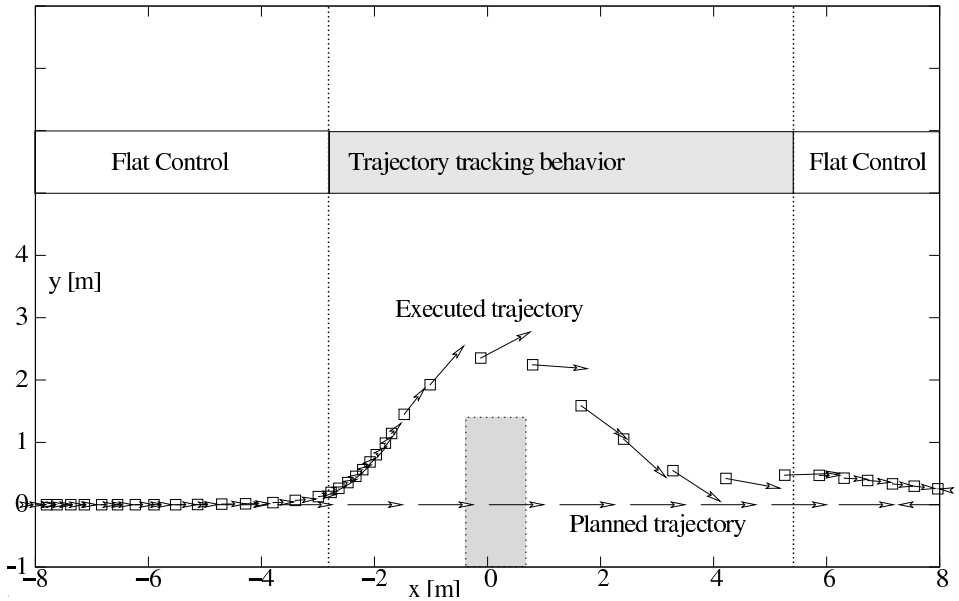


Fig. 13. Collaboration of trajectory tracking and obstacle avoidance on a simulated example

Fig. 14. An experimental setting showing from left to right: The arbitrary placing of the landmarks; the manual driving phase for landmark and obstacle map-building; the obstacle map generated together with the current position of the robot as seen on the LCD display; the capture of the goal position given by the user by means of the touch-screen; the execution of the found trajectory among aggressive pedestrians.

5 Experimental setup

We tested the integration of these essential autonomy capacities in our experimental platform the Cycab robot. The aim was to validate the theoretical considerations

made for the BiS-car and to get insight into the limitations of the whole motion scheme.

The computation power on-board the Cycab is a *Pentium II*^{TM} 233MHz running a Linux system. All programs were written in C/C++ language.

During the experiments the speed of the robot was limited to $1.5ms^{-1}$. The control rate of the robot was fixed at $50ms$. The throughput rate of the laser range-finder was limited to $140ms$ [5] ; therefore the control system has to rely momentarily in odometry[13] readings.

Fig. 14 is a set of pictures showing a complete application integrating the stages described throughout the paper.

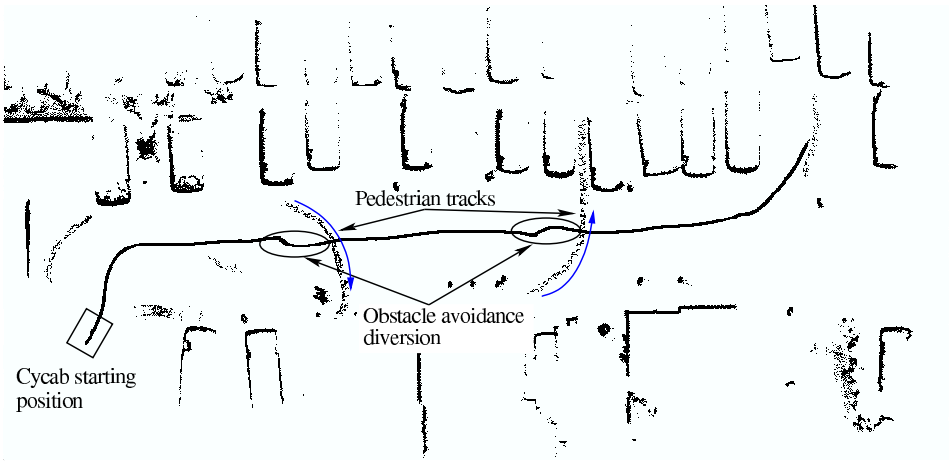


Fig. 15. Executed trajectory among static obstacles and moving pedestrians. Rear middle point (R in fig. 3) trajectory is drawn.

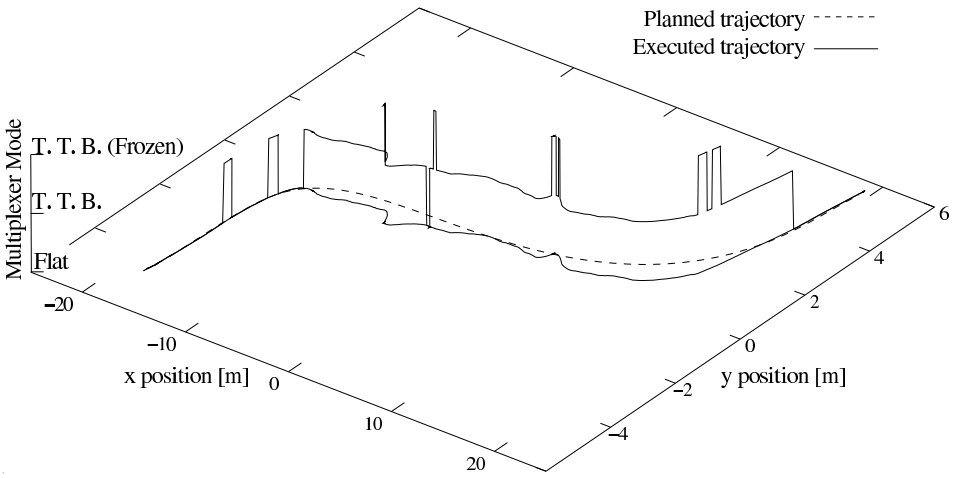


Fig. 16. Executed trajectory with respect to planned trajectory, and multiplexer mode.

[5] This rate is fair enough for our needs, even though we could use a real-time driver.

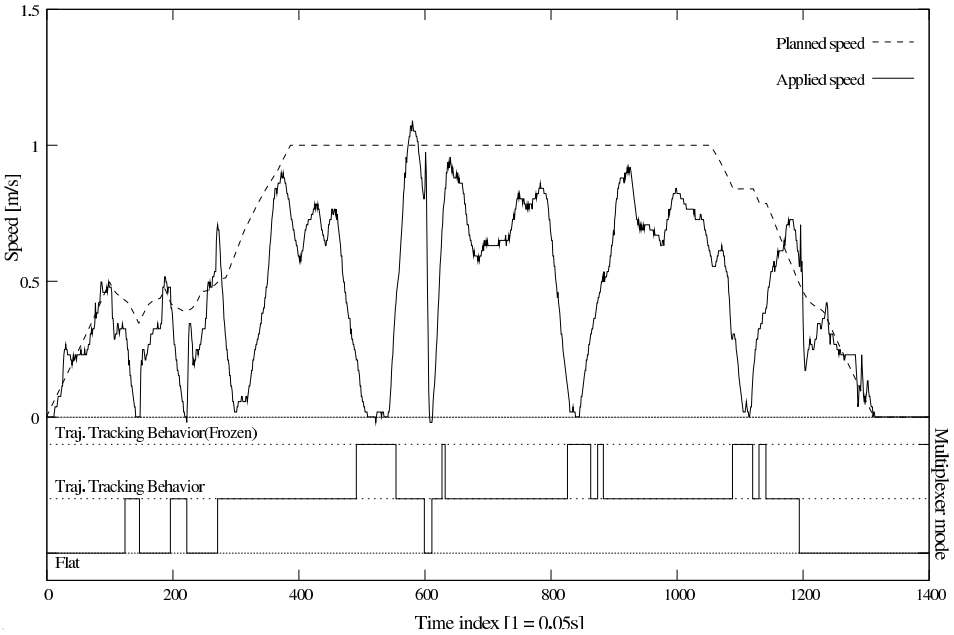


Fig. 17. Applied speeds with respect to planned speed, and multiplexer mode.

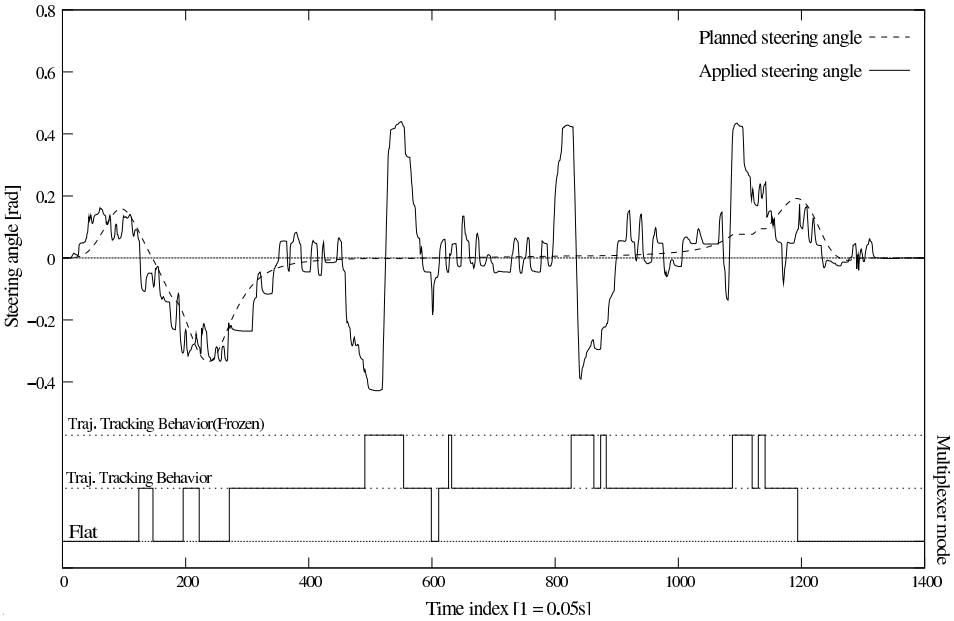


Fig. 18. Applied steering with respect to planned steering, and multiplexer mode.

Figs 15 to 18 illustrates how a planned trajectory is executed while avoiding moving pedestrians. In this environment, the control law using flatness could only be used at the beginning and at the end of the trajectory. On the remaining of the trajectory, speed and steering angle are adjusted in order to maintain security while keeping pace with the plan as much as possible.

6 Discussion & Conclusions

In this paper, we presented our new steps toward the autonomy of a bi-steerable car. The integration of localisation, map building, trajectory planning and execution in a moderately dynamic environment was discussed. Control law using the CyCab flatness property was found to be insufficient for trajectory tracking among moving pedestrians.

Even if this integration was successful and provides satisfactory results, we are convinced that a reactive behaviour cannot be sufficient for the autonomy of vehicle in a real urban environment. For this reason, we are working on the perception and identification of road users (pedestrians, cars, bikes or trucks). By this way, we will be able to predict future movement of "obstacles" and to react accordingly, in a *smarter* way than the simple scheme proposed in this paper.

References

[1] P. Bessière and BIBA-INRIA Research Group. Survey: Probabilistic methodology and techniques for artefact conception and development. Technical Report RR-4730, INRIA, Grenoble, France, February 2003. http://www.inria.fr/rrrt/rr-4730.html.

[2] S. Blondin. Planification de mouvements pour véhicule automatisé en environnement partiellement connu. Mémoire de Diplôme d'Etudes Approfondies, Inst. Nat. Polytechnique de Grenoble, Grenoble (FR), June 2002.

[3] J. Borenstein and Y. Koren. Real-time obstacle avoidance for fast mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(5):1179–1187, Sept/Oct 1989.

[4] J. Borenstein and Y. Koren. The vector field histogram - fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288, June 1991.

[5] O. Brock and O. Khatib. High-speed navigation using the global dynamic window approach. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Detroit, MI (US), May 1999.

[6] R.W. Brockett. Asymptotic stability and feedback stabilization. In R.W. Brockett, R.S. Millman, and H.J. Sussmann, editors, *Differential Geometric Control Theory*, pages 181–191, Boston, MA: Birkhäuser, 1983.

[7] Carlos Canudas de Wit, Bruno Siciliano, and George Bastin Eds. *Theory of Robot Control*. Springer-Verlag, 1996.

[8] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *IEEE Computer, Special Issue on Autonomous Intelligent Machines*, June 1989.

[9] M. Fliess, J. Lévine, P. Martin, and P. Rouchon. Design of trajectory stabilizing feedback for driftless flat systems. In *Proc. of the European Control Conference*, pages 1882–1887, Rome, Italy, september 1995.

[10] M. Fliess, J. Lévine, P. Martin, and P. Rouchon. Flatness and defects of nonlinear systems: introductory theory and examples. *Int. Journal of Control*, 61(6):1327–1361, 1995.

[11] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, 4(1):23–33, March 1997.

[12] Th. Fraichard and Ph. Garnier. Fuzzy control to drive car-like vehicles. *Robotics and Autonomous Systems*, 34(1):1–22, December 2000.

[13] J. Hermosillo, C. Pradalier, and S. Sekhavat. Modelling odometry and uncertainty propagation for a bi-steerable car. In *Proc. of the IEEE Intelligent Vehicle Symp.*, Versailles (FR), June 2002. Poster session.

[14] J. Hermosillo and S. Sekhavat. Feedback control of a bi-steerable car using flatness; application to trajectory tracking. In *Proc. of the American Control Conference*, Denver, CO (US), June 2003.

[15] Lawrence A. Klein. *Sensor Data Fusion Concepts and Applications*. SPIE, 1993.

[16] F. Lamiraux, S. Sekhavat, and J.-P. Laumond. Motion planning and control for hilare pulling a trailer. *IEEE Trans. Robotics and Automation*, 15(4):640–652, August 1999.

[17] F. Large, S. Sekhavat, Z. Shiller, and C. Laugier. Towards real-time global motion planning in a dynamic environment using the NLVO concept. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Lausanne (CH), September-October 2002.

[18] Ch. Laugier, S. Sekhavat, L. Large, J. Hermosillo, and Z. Shiller. Some steps towards autonomous cars. In *Proc. of the IFAC Symp. on Intelligent Autonomous Vehicles*, pages 10–18, Sapporo (JP), September 2001.

[19] O. Lebeltel, P. Bessière, J. Diard, and E. Mazer. Bayesian robots progamming. *Autonomous Robots*, 2003. In Press.

[20] O. Lefebvre, F. Lamiraux, C. Pradalier, and Th. Fraichard. Obstacles avoidance for car-like robots. integration and experimentation on two robots. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, New Orleans, LA (US), April 2004.

[21] P. Newman. *On the structures and solution of simultaneous localization and mapping problem*. PhD thesis, Australian Center for Field Robotics,Sidney, 1999.

[22] G. Oriolo, G. Ulivi, and M. Vendittelli. *Applications of Fuzzy Logic: Towards High Machine Intelligent Quotient Systems*. Prentice-Hall, 1997.

[23] C. Pradalier, F. Colas, and P. Bessiere. Expressing bayesian fusion as a product of distributions: Applications in robotics. In *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, 2003.

[24] C. Pradalier and S. Sekhavat. Concurrent localization, matching and map building using invariant features. In *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, 2002.

[25] C. Samson and K. Ait-Abderrahim. Feedback stabilization of a nonholonomic wheeled mobile robot. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1242–1246, Osaka (JP), November 1991.

[26] S. Sekhavat, J. Hermosillo, and P. Rouchon. Motion planning for a bi-steerable car. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 3294–3299, Seoul (KR), May 2001.

[27] R. Simmons. The curvature-velocity method for local obstacle avoidance. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 3375–3382, Minneapolis, MN (US), April 1996.

[28] I. Ulrich and J. Borenstein. VFH+: Reliable obstacle avoidance for fast mobile robots. In *IEEE Int. Conf. Robotics Automation*, pages 1572–1577, Leuven, Belgium, May 1998.

[29] I. Ulrich and J. Borenstein. VFH*: Local obstacle avoidance with look-ahead verification. In *IEEE Int. Conf. Robotics Automation*, pages 2505–2511, San Francisco, CA, April 2000.

Appendix H

Curriculum Vitae

Cédric Pradalier

Associate Professor

GeorgiaTech Lorraine

2 rue Marconi, 57000 Metz

France

I. Earned Degrees

- 2001-2004: PhD in Computer Science, specialization in Graphics, Computer Vision and Robotics. Title: “Intentional Navigation for a Mobile Robot”. Supervisor: Prof. Christian Laugier. Institution: National Polytechnic Institute of Grenoble / INRIA (France).
- 2000-2001: Master of Science in Computer Science, specialization in Graphics, Computer Vision and Robotics. Institution: National Polytechnic Institute of Grenoble / INRIA (France).
- 1998-2001: French Ingénieur Degree (very selective degree in the French educative system, usually considered equivalent to a master degree) at ENSIMAG, National School of Computer Science and Applied Mathematics in Grenoble, France. Specialization in compilers and languages.

II. Employment

- 2012-present: Associate Professor at GeorgiaTech Lorraine, Metz, France. Head of the DREAM lab: Data-driven Robotics for Environment Assessment and Monitoring.
- 2007-2012: Deputy Director of the Autonomous Systems Lab (ASL, Director: Prof. R. Siegwart), department of Mechanical Engineering, ETH Zürich (Federal Polytechnic Institute of Zürich), Switzerland. Responsibilities: daily operation of the ASL (approx.. 40 people), supervision of the third party grant and budget, research project development and supervision.
- 2009-present: Funding member of Skybotix AG, ETH spin-off specialized on the development of micro-air-vehicle for education and industrial application. Responsibilities: initially software development but purely advisorial role at this stage.
- 2004-2007: Post-doctoral fellow at the Autonomous Systems Lab (ASL), ICT Center, CSIRO (Commonwealth Scientific and Industrial Research Organisation), Brisbane, Australia. Responsibilities: software development for robotics and automation of industrial vehicles, submarines, etc...

III. Teaching

A. Individual Student Guidance

1. GeorgiaTech Lorraine

a) PhD students:

- Co-supervision of S. Griffith with Prof. F. Dellaert, PhD student co-funded by the Lorraine region and GTL, on the topic «Learning in Natural Environments». Expected completion: 2016

b) Graduate students supervised as part of the MSCS or MSECE or PhD Robotics

- Special problems (CS/ECE 8903):

Title	**Student(s)**	**Dates**
Visual Servoing for Stabilisation and Docking of an Autonomous Surface Vessel	Josephine Simon	2013-01-06 2013-05-01
Automation of an Autonomous Surface Vessel for Shore Following and Monitoring	Violette Louisot	2013-01-06 2013-05-01
Visual Servoing for Stabilisation and Docking of an Autonomous Surface Vessel	Paul Drews	2013-08-19 2013-12-15
3D visualisation of geo-localised image data-base	Naissa Conde	2014-01-06 2014-05-01

- Multidisciplinary Robotics Research (CS 8750/CS 8751)

Title	Student(s)	Dates
High Precision Take-off and Landing for Micro-Aerial Vehicles.	Adam Cantor	2013-05-15 2013-08-02
Kingfisher Station-keeping Controller	Paul Drews	2014-01-06 2014-05-02

2. ETH Zürich

50 student projects supervised over 7 semesters, among which 18 Master projects. In most cases, Master projects are focused on the design of robotic systems for specific applications. Some projects later lead to external funding, in particular for the Limnobotics project. Other led to conference publications. As the role of deputy-director of the ASL is requiring more and more time, my implication is now reduced to the inception of student projects which will later be supervised by PhD students.

a) PhD students:

- Co-supervision of G. Hitz with Prof. R. Siegwart, PhD student funded by the Swiss national fund, on the topic «Navigation of an Autonomous Surface Vessel for the environment monitoring in fresh-water environment». Expected defense: early 2014.
- General involvement in the supervision of most of the PhD students of the ASL, in collaboration with Prof. R. Siegwart. In Swiss laboratories, the director (professor) is always the person officially responsible for the PhD student supervision.

b) Master students supervised at ETH Zürich.

Title	Student(s)	Dates
Development of an autonomous surface vehicle	Corsin Gwerder, Lukas Limacher	2012-09-03 2013-03-29
Swimming pattern design, optimisation and evaluation for an underwater robotic turtle	Cédric Siegenthaler	2012-02-21 2012-08-31
Vision-based trajectory following for boats and ground robots	Hendrik Erckens	2011-10-01 2012-04-01
Design of a modular character animation tool	Oliver Glauser	2011-07-01 2012-06-30
Forward looking boom control for self propelled agricultural sprayers based on lasers sensor data	Valentin Gresch	2011-04-01 2011-09-30
RFID-based SLAM for the Mining Safety Platform	Christian Forster	2011-04-01 2011-09-30
Garden robotics	Linus Rohrer, Stefan Riesen	2011-02-21 2011-09-02

Supervisor design for the ExoMars control architecture	Ueli Halter	2010-10-17 2011-04-15
Autonomous indoor/outdoor operation of a micro-helicopter	Amadeo Knüsel, Sammy Omari	2010-10-03 2011-04-15
Beyond Simple Point Clouds: Volumetric, Surface, and Colour Representations for 3D Maps	Claude Holenstein	2010-10-01 2011-04-15
Visual Odometry for a Rover Using a Monocular Ground-facing Camera	Tim Kazik	2010-09-22 2011-03-31
Control optimisation for space rovers with deformable suspension	Pascal Strupler	2010-09-15 2011-03-31
Design and Implementation of a docking station for micro-helicopters	Fadri Furrer	2010-09-15 2011-03-31
Autonomous data collection, navigation and obstacle avoidance for the Limnobotics Autonomous Sampling Boat	Gregory Hitz	2010-04-18 2010-10-23
Modular Body Design for an Underwater Robot	Davinia Font Calafell	2010-02-22 2010-09-30
Simulation, Control and Testing of an Autonomous Sailboat	Mario Krucker	2010-02-22 2010-09-30
Design of an Autonomous Sampling Boat for the study of Algae Bloom in the Zurich Lake.	Manuel Baumann, Oliver Baur	2009-09-14 2010-02-28
Simulation and Control of an Autonomous Sailboat	Fabian Jenne	2009-09-01 2010-06-30
Preliminary Design of an autonomous boat for biological profiling	Tobias Hänggi	2009-04-20 2009-10-31

c) Summester Projects supervised (Graduate projects)

At ETH Zürich, in the department of Mechanical Engineering, MSc students must conduct a semester project before there master thesis. These projects account for approximately 40% of their time over a semester. The list below shows the projects I supervised:

Title	**Student(s)**	**Dates**
Visual tracker for stylized micro-helicopters	Endri Dibra	2011-11-14 2012-03-01
Robotic Floor Marking System using a Laser Measurement System	Bastian Bücheler	2011-09-18 2011-12-23
Mechatronic Design of an underwater robot	Lukas Limacher	2011-02-21 2011-06-30
Visual Position Control for Artificial Teeth Mounting Machine	Bernhard Mühlburger	2011-02-21 2011-06-04

Electrical design for the TRALOC snake-bot	Tobi Bleiker	2011-01-17 2011-02-28
Project Naro-Tortuga: design of a 2/3DoF actuation mechanism for underwater propulsion	Cédric Siegenthaler	2010-09-22 2010-12-25
Real-time long range communication and logging for the Limnobotics Autonomous Sampling Boat	Jürg Weber	2010-09-15 2010-12-25
Autonomous navigation and mapping for a micro-helicopter	Christian Gehring, Yves Pilat	2010-02-22 2010-06-30
Design and evaluation of a Wing-Leg for an underwater robot	Janine Stocker	2010-02-22 2010-06-30
Sensor integration and obstacle avoidance for an orientation assistant for Science City Hoenggerberg	Meng-Yu Yu	2010-02-22 2010-06-04
Satellite tracking from camera: the inverse GPS problem	Thomas Neuenschwander	2008-10-31 2009-03-11

d) Bachelor Projects supervised (Undergraduate projects)

During the Bachelor, students must also conduct a final project, accounting for 100% of their time during their 6th semester. The table below lists the project I supervised:

Title	**Student(s)**	**Dates**
Control of a Micro UAV using an external laser tracker	Andreas Michel, Dominic Jud	2011-02-21 2011-06-18
Autonomous docking for the Limnobotics Autonomous Sampling Boat	Nico Geiser	2010-02-22 2010-09-30
Embedded Obstacle Avoidance for the CoaX	Erik Voigt	2010-02-22 2010-06-30
Fully autonomous MAV-flight in avalanche regions	Luc Oth, Manuel Grauwiler	2010-02-22 2010-06-04
Path planning and navigation of an orientation assistant for Science City Hoenggerberg	Benedikt Mathis	2010-02-22 2010-06-04
User-Interface and –guiding behavior for an orientation assistant for Science City Hoenggerberg	Lorenz Oswald	2010-02-22 2010-06-04
Autonomous docking for the Limnobotics Autonomous Sampling Boat	Chrstoph Kammer	2010-02-22 2010-06-04
Implementation of a low level controller and design of a roll cage for the BOOGAR	Claudio Cavelti, Daniel Längle	2009-02-16 2009-05-30
Design of a Diving System for a Robotic Fish	Thomas Wuhrmann	2009-02-16 2009-05-29
iPhone-Robot Command Interface: A Case Study	Florian Müller	2009-02-16 2009-05-29

Sailing Strategy for a Robotic Yacht	Gion-Andri Büsser	2009-02-16 2009-05-29
Design and Evaluation of locomotion concepts for an Autonomous Underwater Robot.	Benjamin Peter, Roman Ratnaweera	2009-02-15 2009-06-01
Design of a communication system for an Autonomous Sailboat	Stefan Wismer	2009-02-15 2009-06-01
Design of a trajectory following controller for an Autonomous Sailboat	Hendrik Erckens	2009-02-15 2009-06-01
Energy Management for an Autonomous Sailboat	Jürg Weber	2009-02-15 2009-06-01

e) Studies of Mechatronics supervised (Undergraduate projects)

Also during the Bachelor, students must conduct a small literature research project called a "Study of Mechatronics". The table below lists the Studies of Mechatronics I supervised:

Title	**Student(s)**	**Dates**
Mechatronic history of computer input device	Lorenzo Garbani Marcantini	2011-10-03 2011-12-23
Autonomous docking for the Limnobotics Autonomous Sampling Boat	Nico Geiser	2010-02-22 2010-06-04
Embedded Obstacle Avoidance for the CoaX	Erik Voigt	2010-02-22 2010-05-31
Studies of Mechatronics: Adaptive control for flying robots	Luca Vanoni	2009-10-01 2009-12-18
Studies of Mechatronics: Adaptive control for ground/walking robots	Titus Haas	2009-10-01 2009-12-18
Studies of Mechatronics: methodologies for functional autonomy	Benedikt Mathis	2009-10-01 2009-12-18
Object Recoginition	Dani Eberli	2008-10-06 2008-12-19
LAGR project: overview of eight control strategies	Stefan Haag	2008-09-22 2008-12-19

f) «Fokus Projects»

The fokus projects aims at bringing together teams of 5 to 10 students at the end of the Bachelor. Over 12 months, their objectives are to face all the phases of a mechatronic project: mechanical design, electrical design, electronic design, software design and implementation of all these elements into a full system. Furthermore, the students must find their own funding and present their project at the end of the Bachelor as if

they were launching a product. I directly supervised the Avalon project in 2008-2009 with the objective to develop an autonomous sail boat to cross the Atlantic (this lead to one publication in the Robotic and Automation Magazine). As deputy-director of the ASL, I have been normally involved at a high-level of supervision every year for our 3 projects.

3. CSIRO

- Organization of the visit of French PhD students from the INRIA team e-Motion (Ch. Braillon, A. Negre). Each visit led to publications, in particular with a publication in the Journal of Field Robotics for A. Negre.
- Organization of the reciprocal visit of an Australian PhD student to INRIA (S. Nuske).

B. Other Teaching Activities

1. ETH Zürich

- Creation and responsibility and teaching of *Information Processing for Robotics* (42 hours in the Master of Robotics, System and Control) in collaboration with Dr. R. Triebel and Dr. F. Colas. Taught three times from 2009 to 2011.

IV. Scholarly Accomplishments

A. Published Books and Parts of Books

- **Pradalier, C. and Bessière, P.** Chapter 3 of *Probabilistic Reasoning and Decision Making in Sensory-Motor Systems*, Springer Tracts in Advanced Robotics, 2008, Bessière, Pierre; Laugier, Christian; Siegwart, Roland (Eds.)

B. Refereed Publications

1. International Journals

1) **Liu, M., Pradalier, C., Siegwart, R.** Visual homing from scale with an uncalibrated omnidirectional, camera, IEEE Transactions on Robotics, 29(6), pp. 1353-1365, **2013**

2) **MÈ Garneau, T Posch, G Hitz, F Pomerleau, C Pradalier, R Siegwart** , Short-term displacement of Planktothrix rubescens (cyanobacteria) in a pre-alpine lake observed using an autonomous sampling platform, Limnography and Oceanography 58 (5), **2013**

3) **Stumm, E.; Breitenmoser, A.; Pomerleau, F.; Pradalier, C. and Siegwart, R.** Tensor-voting-based navigation for robotic inspection of 3D surfaces using lidar point clouds, *The International Journal of Robotics Research,* **2012**

4) **Gonzalez, R.; Rodriguez, F.; Guzman, J.L.; Pradalier, C. and Siegwart,R.** Combined visual odometry and visual compass for off-road mobile robots localization, *Robotica,* **2012**

5) **Hitz, G.; Pomerleau, F.; Garneau, M.-E.; Pradalier, C.; Posch, T.; Pernthaler, J. and Siegwart, R.** Autonomous Inland Water Monitoring: Design and Application of a Surface Vessel, *IEEE Robotics and Automation Magazine*, **2012.**

6) **Krebs, A., Pradalier, C., and Siegwart, R.** Adaptive Rover Behaviour Based on Online Empirical Evaluation: Rover-Terrain Interaction and Near-to-Far Learning, *Journal of Field Robotics*, **2010.**

7) **Erckens, H., Busser G.A., Pradalier, C., Siegwart, R.** Avalon, *IEEE Robotics & Automation Magazine*, **2010**

8) **Pradalier, C., Tews, A. and Roberts, J.** Vision-based operations of a large industrial vehicle: autonomous hot metal carrier. *Journal of Field Robotics*. 2008.

9) **Pradalier, C. and Usher, K.** Robust trajectory tracking for a reversing tractor trailer. *Journal of Field Robotics.* 2008.

10) **Nègre, A.; Pradalier, C. and Dunbabin, M.** Robust vision-based underwater homing using self-similar landmarks. *Journal of Field Robotics.* 2008.

11) **Coué, C.; Pradalier,C.; Laugier,C.; Fraichard,T. and Bessière,P.** Bayesian Occupancy Filtering for Multitarget Tracking: an Automotive Application *Int. Journal of Robotics Research*, 2006.

12) **Pradalier, C.; Hermosillo, J.; Koike, C.; Braillon, C.; Bessière, P. and Laugier, C.** The CyCab: a car-like robot navigating autonomously and safely among pedestrians *Robotics and Autonomous Systems*, 2005

13) **Hermosillo, J.; Pradalier, C.; Sekhavat, S. and Laugier, C.** Autonomous Navigation of a Bi-steerable Car: Experimental Issues *Machine Intelligence and Robotic Control Journal*, 2004

2. International Conferences

a) 2014

1) **R Michalec, C Pradalier**, Sidescan Sonar Aided Inertial Drift Compensation in Autonomous Underwater Vehicles, *OCEANS'14 MTS/IEEE*, St. John's, September 2014

2) **A Jacobson, D Panozzo, O Glauser, C Pradalier, O Hilliges, O Sorkine-Hornung**, Tangible and Modular Input Device for Character Articulation, *41st conference*

on computer graphics and interactive techniques (SIGGRAPH), Vancouver, August 2014.

3) **S Griffith, P Drews, and C Pradalier**: Towards Autonomous Lakeshore Monitoring, International Symposium on Experimental Robotics (ISER), Marakesh, June 2014

4) **P Strupler, C Pradalier, R Siegwart**, Terrain Mapping and Control Optimization for a 6-Wheel Rover with Passive Suspension International Conference on Field and Service Robotics, 297-310, 2014

b) 2013

5) **C Siegenthaler, C Pradalier, F Gunther, G Hitz, R Siegwart**, System integration and fin trajectory Design for a robotic sea-turtle, IEEE/RSJ International Conference Intelligent Robots and Systems (IROS), 2013

6) **A Mcfadyen, L Mejias, P Corke, C Pradalier**, Aircraft collision avoidance using spherical visual predictive control and single point features, IEEE/RSJ International Conference Intelligent Robots and Systems (IROS), 2013

7) **R Gonzalez, F Rodriguez, JL Guzman, C Pradalier, R Siegwart** , Control of off-road mobile robots using visual odometry and slip compensation , Advanced Robotics 27 (11), 893-906,2013

8) **P Furgale, C. Pradalier et al.**, Toward automated driving in cities using close-to-market sensors: An overview of the V-Charge Project, IEEE Intelligent Vehicles Symposium (IV), 2013, 809-816

c) 2012

9) **Liu,M.; Pradalier,C.; Pomerleau,F.; Siegwart,R.** Scale-only Visual Homing from an Omnidirectional Camera. Proc. of The IEEE International Conference on Robotics and Automation (ICRA), May 2012.

10) **Pradalier,C.; Bouabdallah,S.; Gohl,P.; Egli,M.; Caprari,G.; Siegwart,R.** The CoaX Micro-helicopter: A Flying Platform for Education and Research Advances in Autonomous Mini Robots, 89-99

d) 2011

11) **D Font, M Tresanchez, C Siegentahler, T Pallejà, M Teixidó, C Pradalier,** Design and Implementation of a Biomimetic Turtle Hydrofoil for an Autonomous Underwater Vehicle *Sensors* 11 (12), 11168-11187, 2011

12) **A Bonchis, N Hillier, J Ryde, E Duff, C Pradalier,** Experiments in Autonomous Earth Moving *18th IFAC World Congress* 18 (1), 11588-11593, 2011

13) **Y Yu, C Pradalier, G Zong** , Appearance-based monocular visual odometry for ground vehicles *Advanced Intelligent Mechatronics (AIM), 2011 IEEE/ASME International Conference*, 2011

14) **D Font, M Tresanchez, C Siegentahler, M Teixidó, T Pallejà, C Pradalier**, Experimental determination of the hydrofoil's angle of attack in the case of a turtle-like Autonomous Underwater Vehicle *OCEANS*, 2011 IEEE-Spain, 2011

15) **B Xu, C Pradalier, A Krebs, R Siegwart, F Sun,** Composite control based on optimal torque control and adaptive Kriging control for the CRAB rover**,** *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 1752-1757, 2011

16) **X Perrin, F Colas, R Chavarriaga, C Pradalier, JR Millán, R Siegwart** , Learning User Habits for Semi-Autonomous Navigation using Low Throughput Interfaces *IEEE Int Conf Systems, Man, and Cybernetics (IEEE SMC 2011)*, 2011

17) A Medina, C Pradalier, G Paar, A Merlo, S Ferraris, L Mollinedo, A Servicing Rover for Planetary Outpost Assembly ASTRA, 2011

18) **B Xu, C Pradalier, R Siegwart,** Torque control with kriging estimation for the CRAB rover ASTRA, 2011

19) **CD Remy, O Baur, M Latta, A Lauber, M Hutter, MA Hoepflinger, C Pradalier,** Walking and crawling with ALoF: a robot for autonomous locomotion on four legs *Industrial Robot: An International Journal* 38 (3), 264-268, 2011

e) 2010

20) Liu, M.; Pradalier, C.; Siegwart, R. and Chen, Q. "A Bearing-Only 2D/3D-Homing Method under a Visual Servoing Framework", *Proc. of The IEEE International Conference on Robotics and Automation (ICRA)*, May 2010.

21) Peter, B.; Ratnaweera, R.; Fischer, W.; Pradalier, C. and Siegwart, R. "Design and Evaluation of a Fin-Based Underwater Propulsion System", *Proc. of The IEEE International Conference on Robotics and Automation (ICRA)*, May 2010.

22) Krebs, A.; Risch, F.; Thueer, R.; Maye, J.; Pradalier, C. and Siegwar, R. "Rover control based on an optimal torque distribution - Application to 6 motorized wheels passive rover", *Proc. of The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2010.

23) Krebs, A.; Pradalier, C. and Siegwart, R. "RTILE - Adaptive rover navigation based on online terrain learning", *Proc. of The 10th International Symposium on*

Artificial Intelligence, Robotics and Automation in Space (iSAIRAS), September 2010

24) **Remy, C.; Baur, O.; Latta, M.; Lauber, A.; Hutter, M.; Hoepflinger, M.; Pradalier, C. and Siegwart, R.** "Walking and Crawling with ALoF - A Robot for Autonomous Locomotion on Four Legs, *Proc. of the 13th International Conference on Climbing and Walking Robots (CLAWAR)*, September 2010.

f) 2009

25) L. Giger, S. Wismer, S. Boehl, G. Buesser, H. Erckens, J. Weber, P. Moser, P. Schwizer, C. Pradalier, R. Siegwart, "Design and Construction of the Autonomous Sailing Vessel AVALON", *Proc. of The World Robotic Sailing Championship and International Robotic Sailing Conference*, 2009.

26) X. Perrin, F. Colas, C. Pradalier, R. Siegwart, "Learning to identify users and predict their destination in a robotic guidance application", *Field and Service Robotics (FSR)*, 2009.

27) Liu, M.; Scaramuzza, D.; Pradalier, C.; Siegwart, R. and Chen, Q. "Scene Recognition with Omnidirectional Vision for Topological Map using Lightweight Adaptive Descriptors", Proc. of The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), October 2009.

28) **Shin, J.; Gachter, S.; Harati, A.; Pradalier, C. and Siegwart, R.** "Object Classification based on a Geometric Grammar with a Range Camera", *Proc. of The IEEE International Conference on Robotics and Automation (ICRA)*, May 2009.

g) 2008

29) Krebs, A,; Pradalier, C. and Siegwart R., "Comparison of Boosting based Terrain Classification using Proprioceptive and Exteroceptive Data", Proc. of The 11th International Symposium on Experimental Robotics (ISER), July 2008.

30) Hoepflinger, M.; Krebs, A.; Pradalier, C.; Lee, C.; Obstei, R. and Siegwart, R. "Description of the Locomotion Control Architecture on the ExoMars Rover Breadboard", *Proc. of The 10th ESA Workshop on Advanced Space Technologies for Robotics and Automation (ASTRA)*, 2008.

31) Krebs, A.; Pradalier, C. and Siegwart, R. "Strategy for adaptive rover behavior based on experiment", *Proc of the 10th ESA Worshop on Advanced Space Technologies for Robotics and Automation*, 2008.

32) **Pradalier, C.** Vision-based handling tasks for an autonomous outdoor forklift. In: Laugier, C. and Siegwart, R., eds. *Field and Service Robots: Results of the 6th International Conference* (FSR '07); Chamonix, France. Springer; 2008: 61-70. (Springer tracts in advanced robotics. v. 42). ISBN: 9783540754039.

33) **Pradalier, C. and Usher, K.** Experiments in autonomous reversing of a tractor-trailer system. In: Laugier, C. and Siegwart, R., eds. *Field and Service Robots: Results of the 6th International Conference* (FSR '07); Chamonix, France. Springer; 2008: 475-484. (Springer tracts in advanced robotics. v. 42). ISBN: 9783540754039.

34) **Braillon, C.; Pradalier, C.; Usher, K.; Crowley, J. L., and Laugier, C**. Occupancy grids from stereo and optical flow data. In: Khatib, O.; Kumar, V., and Rus, D., eds. *Experimental Robotics: the 10th International Symposium on Experimental Robotics* (ISER '06); Rio de Janeiro, Brazil. Springer; 2008: 367-376. (Springer tracts in advanced robotics. v. 39). ISBN: 9783540774563.

h) ***2007***

35) **Negre, A.; Pradalier, C., and Dunbabin, M.** Robust vision-based underwater target identification and homing using self-similar landmarks. In: Laugier, C. and Siegwart, R., eds. *Field and Service Robots: Results of the 6th International Conference* (FSR '07); Chamonix, France. Springer; 2008: 51-60. (Springer tracts in advanced robotics. v. 42). ISBN: 9783540754039.

36) **Nuevo, J.; Pradalier, C., and Bergasa, L. M.** Model-based load localisation for an autonomous Hot Metal Carrier. *IEEE/RSJ International Conference on Intelligent Robots and Systems* (IROS 2007); San Diego, Calif. IEEE; 2007: 247-252. ISBN: 9781424409129.

37) **Pradalier, C. and Usher, K.** A simple and efficient control scheme to reverse a tractor-trailer system on a trajectory. *IEEE International Conference on Robotics and Automation* (ICRA '07); Rome, Italy. IEEE; 2007: 2208-2214.

38) **Roberts, J.; Pradalier, C., and Tews, A.** Autonomous hot metal carrier. In: Grandfield. J. F. and Taylor, J. A., eds. *Aluminium Cast House Technology : 10th Australasian Conference & Exhibition*; Sydney, NSW. CSIRO Publ.; 2007: 247-256. ISBN: 9780643094741.

39) **Roberts, J.; Tews, A., and Pradalier, C.** Experiments and experiences with dependability for a large autonomous industrial vehicle. *5th IARP/IEEE-RAS/EURON International Workshop on Technical Challenges for Dependable Robots in Human Environments*; Rome, Italy. 2007.

40) Roberts, J.; Tews, A.; Pradalier, C., and Usher, K. Autonomous hot metal carrier: navigation and manipulation with a 20 tonne industrial vehicle. *IEEE International Conference on Robotics and Automation* (ICRA '07); Rome, Italy. IEEE; 2007: 2770-2771.

41) **Tews, A.; Pradalier, C., and Roberts, J.** Reliable autonomous industrial vehicle operations. I*EEE International Conference on Robotics and Automation* (ICRA '07); Rome, Italy. IEEE; 2007: 1176-1182.

3. 2006

42) Braillon, C.; Pradalier, C.; Crowley, J. L., and Laugier, C. Real-time moving obstacle detection using optical flow models. *IEEE Intelligent Vehicle Symposium* (IV 2006); Tokyo, Japan. IEEE; 2006: 466-471.

43) Braillon, C.; Pradalier, C.; Usher, K.; Crowley, J. L., and Laugier, C. Occupancy grids from stereo and optical flow data. *IEEE International Symposium on Experimental Robotics* (ISER '06); Rio de Janeiro, Brazil. 2006.

44) Braillon, C.; Usher, K.; Pradalier, C.; Crowley, J. L., and Laugier, C. Fusion of stereo and optical flow data using occupancy grids. *9th International IEEE Conference on Intelligent Transportation Systems* (ITSC 2006); Toronto, Ont. IEEE; 2006: 1240-1245.

45) **Braillon, C.; Usher, K.; Pradalier, C.; Crowley, J. L., and Laugier, C.** Real-time stereo and optical flow data fusion. *IEEE/RSJ International Conference on Intelligent Robots and Systems* (IROS 2006); Beijing, China. IEEE; 2006.

4. 2005

46) Tay, C.; Pradalier, C. and Laugier, C. Online reconstruction of vehicles in a car park. In: Corke, P. and Sukkarieh, S., eds. *Field and Service Robotics: Results of the 5th International Conference* (FSR 2005); Port Douglas, Qld. Springer; 2006: 207-218. (Springer tracts in advanced robotics. v. 25). ISBN: 3540334521.

47) Pradalier, C.; Vaussier, S. and Corke, P. Path planning for a parking assistance system: implementation and experimentation. *Australasian Conference on Robotics and Automation* (ACRA 2005); University of New South Wales. ARAA; 2005. CD ROM. ISBN: 0958758379.

48) **Tay,C.; Pradalier,C. and Laugier,C.** Vehicle Detection And Car Park Mapping Using Laser Scanner *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems* - Edmonton, Alberta, Canada - August 2005

5. 2004

49) Pradalier, C.; Hermosillo, J.; Koike, C.; Braillon, C.; Bessière, P. and Laugier, C. An Autonomous Car-Like Robot Navigating Safely Among Pedestrians *Proc. of the IEEE Int. Conf. on Robotics and Automation* - New Orleans, LA (US) - April 2004

50) Pradalier, C.; Bessière, P. and Laugier, C. Driving On A Known Sensori-Motor Trajectory With A Car-like Robot Cédric Pradalier, Pierre Bessière, Christian Laugier *Proc. of the Int. Symp. on Experimental Robotics* - Singapore (SG) - June 2004

51) Lefebvre, O.; Lamiraux, F.; Pradalier, C. and Fraichard, T. Obstacles Avoidance for Car-Like Robots. Integration And Experimentation on Two Robots *Proc. of the IEEE Int. Conf. on Robotics and Automation* - New Orleans, LA (US), page 4277--4282 - April 2004

52) **Pradalier, C. and Bessière, P.** Perceptual navigation around a sensori-motor trajectory *Proc. of the IEEE Int. Conf. on Robotics and Automation* - New Orleans, LA (US) - April 2004

6. 2003

53) Coué, C. ; Pradalier,C. and Laugier, C. Bayesian Programming for Multi-Target Tracking: an Automotive Application *Proc. of the Int. Conf. on Field and Service Robotics* - Lake Yamanaka (JP) - July 2003

54) Hermosillo, J.; Pradalier, C.; Sekhavat, S. and Laugier, C. Experimental Issues from Map Building to Trajectory Execution for a Bi-steerable Car. *Proc. of the IEEE Int. Conf. on Advanced Robotics* - Coimbra (PT) - July 2003

55) Pradalier, C.; Colas, F. and Bessière, P. Expressing Bayesian Fusion as a Product of Distributions: Application to Randomized Hough Transform *Proc. of the Conf. on Bayesian Methods and Maximum Entropy in Science and Engineering* - Jackson Hole, WY (US) - August 2003

56) Pradalier, C.; Colas, F. and Bessière, P. Expressing Bayesian Fusion as a Product of Distributions: Applications in Robotics *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems* - Las vegas, NV (US) - October 2003

57) Koike, C.; Pradalier, C.; Bessière, P. and Mazer, E. Obstacle Avoidance and Proscriptive Bayesian Programming *Proc. of the Workshop on Reasoning with Uncertainty in Robotics* - Acapulco (MX) - July 2003

58) **Koike, C.; Pradalier, C.; Bessière, P. and Mazer, E.** Proscriptive Bayesian Programming Application for Collision Avoidance *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems* - Las vegas, NV (US) - October 2003

59) **Pradalier, C.; Hermosillo, J.; Koike, C.; Braillon, C.; Bessière, P. and Laugier, C.** Safe and Autonomous Navigation for a Car-Like Robot Among Pedestrian *IARP Int. Workshop on Service, Assistive and Personal Robots* - Madrid (ES) - October 2003

60) **Hermosillo, J.; Pradalier, C.; Sekhavat, S.; Laugier, C. and Baille,G.** Towards Motion Autonomy of a Bi-steerable Car: Experimental Issues from Map-building to Trajectory Execution *Proc. of the IEEE Int. Conf. on Robotics and Automation* - Taipei (TW) - May 2003

7. 2002

61) **Pradalier,C. and Sekhavat,S.** «Localization Space»: a Framework for Localization and Planning, for Systems Using a Sensor/Landmarks Module, *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Washington DC, 2002

62) **Pradalier,C. and Sekhavat,S.** Concurrent matching, localization and map building using invariant features, *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, 2002

C. Editorial Contributions

63) **Pradalier,C.; Hirzinger,G.;Siegwart,R.** Robotics Research: The 14th International Symposium ISRR, Springer, 2011

64) **Pradalier,C.; Hirzinger,G.;Siegwart,R.** Editoria,l Special Issue on the Fourteenth International Symposium on Robotics Research, 2009 The International Journal of Robotics Research 30 (3), 263-264, published in 2011

65) **Laugier,C.; Martinelli,A.; Pradalier,C.; Siegwart,R.** The International Journal of Robotics Research (IJRR)-Special issue on``Field and Service Robotics'', 2009

66) **Pradalier,C.; Laugier,C.; Martinelli,A.; Siegwart,R.** Editorial for Journal of Field Robotics—Special Issue on Field and Service Robotics, Journal of Field Robotics 25 (6-7), 303-304, 2008

V. Service

A. Professional Contributions

- 2005: Member of the organization team for the Field and Service Robotics conference (FSR)
- 2006: Integration in the ICRA Associate Editors (in the team since then)
- 2007: Integration in the FSR Committee
- 2009: Integration in the Journal of Field Robotics Editorial board.
- 2009: Publication chair and local organizer for the International Symposium on Robotics Research 2009 (ISRR'09, September)
- 2012: Program chair for CARPI'12: Conference on Advance Robotics for the Power Industry
- 2012: Organizer for the ROScon'12: Technical conference around the ROS middleware.

VI. Grants and Contracts

A. As Principal and Co-Principal Investigator

Note that at ETH Zürich, the legal responsibility always lies with the professor leading a research lab, even if the proposal has been written and submitted by one of the lab member. In the projects below I listed myself as a PI when I was actually leading the project.

1. European funding (FP6/FP7/H2020):

- Flourish : RIA/H2020, 2015-2018, **PI**, Preparation of the proposal. Funding : Total 4'761'296€, Allocated to the UMI2958 457'650€.
- Noptilus: IP/FP7, 2010-2014, **PI**, Preparation of the proposal, Supervision of ASL responsibilities and technical contributions, supervision of 1 PhD students. Autonomous Underwater Vehicles. Funding: Total 3'784'022€ Allocated to ASL 411'140€
- V-Charge: IP/FP7, 2010-2014, **Co-PI**, Preparation of the proposal, technical coordinator[1], supervision of 1 PostDoc, 2-3 PhD students. Vision-based autonomous car navigation.

  Funding: Total 5'630'000€ Allocated to ASL 1'020'776€

[1] The coordinator of a European project is the partner who is interacting with the funding agency and leading the project. The technical coordinator is the person leading the technical activities of the consortium.

2. Funding from the European Spacial Agency (ESA):

- ExoMars, 2007-2011: **PI**, Proposals, development of various prototypes and tools for the ExoMars program. Continuous supervision of 1-2 PhD or engineers. Prime contractor: RUAG Space (Switzerland).

  Approximate funding for ASL: 300'000 CHF.

- Eurobot Rover Prototype, 2009: **PI**, proposal, design and development of a space rover for mobile manipulation experiments. Prime contractor: GMV (Spain). Supervision of 1 engineer. Funding: Total 150'000€ Allocated to ASL 75'000€

- Heavy Duty Planetary Chassis, 2011: **PI**, proposal, design and development of a scalable space rover. Supervision of 1 engineer. Prime contractor: RUAG Space (Switzerland).

  Funding: Allocated to ASL 70'000€

- Lunar Robotic Challenge, 2008: supervision of a team of students and PhD to build a robotic platform for lunar exploration over 8 months. Deployment and tests in a volcanic environment in Tenerife.

  Funding: Allocated to ASL 50'000€.

3. Funding from national agencies:

- Swiss National Fund: "Limnobotics" project, 2010-2013: **Co-PI**, supervision of 1 PhD (G. Hitz). Study of algae populations using an autonomous surface vessel.

  Funding: Total 439'174 CHF Allocated to ASL 160'744 CHF.

- Commission for Technology and Innovation: "Autonomous Robotic Inspection using High-Precision Localisation", 2011-2012, **PI**, proposal lead, collaboration with industrial partners: Alstom Inspection Robotics and Leica Geo-Systems. Supervision of 2 engineers.

  Funding: 265'107 CHF

B. As Investigator

1. European funding (FP6/FP7):

- Robots@Home: STREP/FP6, 2006-2009, Supervision of the ASL responsibilities and technical contributions, supervision of 1 PostDoc, 2PhD students. Mobile robot navigation in an indoor environment.

- NIFTI: IP/FP7, 2009-2013, Preparation of the proposal, general supervision of ASL side and overview, supervision of 1 PostDoc and 2PhD students. Robotic search and rescue.

- s-Fly: STREP/FP7, 2008-2011, Preparation of the proposal, general supervision of ASL side and overview (Coordinator: ASL), supervision of 1 PostDoc and 2PhD students. Swarm of flying robots.

- Europa: STREP/FP7, 2008-2011, Preparation of the proposal, general supervision of ASL side and overview, supervision of 1 PostDoc and 3PhD students. Navigation in outdoor urban environment.

VII. Honors and Awards

- Prix De Vigier, Juin 2011, received from fondation “De Vigier” (www.devigier.ch) for the spin-off Skybotix (founding member).
- TeamWork Award, 2007, CSIRO

VIII. Personal Data

Born: 1978 **Citizenship:** French

Email cedric.pradalier@georgiatech-metz.fr

WWW http://dream.georgiatech-metz.fr

Language: French (native), English (fluent), Spanish and German (basics)