Robust Robot Localization in a Complex Oil and Gas Industrial Environment

Pierre Merriaux, Yohan Dupuis, Rémi Boutteau, Pascal Vasseur, Xavier Savatier

Robust Robot Localization in a Complex Oil and Gas Industrial Environment

Pierre Merriaux
IRSEEM, ESIGELEC
76800 Saint-Etienne-du-Rouvray
pierre.merriaux@esigelec.fr

Yohan Dupuis
CEREMA, Department of Multimodal Transportation Infrastructure
76120 Le Grand Quevilly
yohan.dupuis@cerema.fr

Rémi Boutteau
IRSEEM, ESIGELEC
76800 Saint-Etienne-du-Rouvray
remi.boutteau@esigelec.fr

Pascal Vasseur
Normandie Univ, UNIROUEN, UNIHAVRE, INSA Rouen, LITIS
76000 Rouen, France
pascal.vasseur@univ-rouen.fr

Xavier Savatier
IRSEEM, ESIGELEC
76800 Saint-Etienne-du-Rouvray
xavier.savatier@esigelec.fr

Abstract

In this paper, we propose a LiDAR-based robot localization method in a complex oil and gas environment. Localization is achieved in six Degrees of Freedom (DoF) thanks to a particle filter framework. A new time-efficient likelihood function, based on a pre-calculated 3D likelihood field, is introduced. Experiments are carried out in real environments and their digitized point clouds. Six DoF real-time localization is achieved with spatial and angular errors of less than 2.5cm and 1° respectively in a real environment of $350m^3$. The proposed approach focuses on real-time performance on embedded platforms. It enabled the Vikings team to win the first two ARGOS Challenge contests.

1 INTRODUCTION

1.1 Argos Challenge

ARGOS (Autonomous Robot for Gas and Oil Sites) Challenge[1] is organized by Total in partnership with ANR. The ARGOS challenge addresses objectives detailed in the Robotics 2020 Multi-Annual Roadmap published by the SPARC(SPA, 2015). SPARC is a public-private partnership between the European Commission and the European Robotics Community. Several key topics have been identified in the roadmap in order to reach a massive deployment of autonomous robots in industrial facilities. The roadmap aims at increasing Technology Readiness Levels for topics such as robot motion, perception, localization or decisional autonomy for instance. The present paper tackles localization challenges highlighted in the Robotics 2020 Multi-Annual Roadmap.

Introduction of autonomous robots for oil and gas infrastructure patrolling are primarily motivated by industrial needs including :

- Constant safety standards. In fact, human tends to decrease their safety standards as they get used to patrolling. Robots will always apply the same safety rules and expectations.
- Fully automated infrastructures already exist. Companies want to keep the platforms without human. Moreover, in case of hazards (fire, gaz leaks, explosions), humans would not be able to access the facilities. Consequently, the robot must be conceived to fulfill the ATEX directives. The robots must be able to perform several tasks mentioned here-above.

Three contests have been organized in order to challenge five international teams. The final contest will be held in March, 2017.

Several tasks must be performed by the robots:

- Pressure reading in manometers.
- Control valve position estimation.
- Detection and localization of abnormal heat surfaces.
- Temperature measurement.
- Pump sound processing.
- Abnormal or unexpected objects in the 3D map of the site.
- Gas leak detection.

[1] http://www.argos-challenge.com/

(a) Floating production storage and offloading boat. ©Serge RUPERT Total S.A.

(b) ARGOS challenge test site

Figure 1: Complex environment targeted in ARGOS challenge : multi-storey building, pipes, stairs...

The tasks can be performed either autonomously or remotely controlled. The robot must be able to safely travel on the platform by detecting static and dynamic obstacles. It should be able to bypass obstacles or climb stairs. The robot also has to be able to detect the emergency alarm and a communication failure.

1.2 Autonomous Navigation

Offshore platforms have different layouts compared to other industrial sites found in the literature. In fact, studied storage warehouses or factories often have squarish shape with machines or racks and shelves and large alleys. A look at the facility blueprint would reveal a well organized structure. Offshore platform environments are different. As it can be seen in Figure 1a, equipments are everywhere. Alleys are narrow. Most of the time such facility has several storeys with stairs to travel from one to another. The floor itself is made of gratings, which makes it irregular. Figure 1b shows the full-scale test environment. It corresponds to a former production unit and scaffold. It is used for the fire department training.

Robot localization is a key step in autonomous robot deployment. Given its localization, a robot can decide how it will reach a given target, i.e. achieves path planning, path control and safely performs given tasks. Consequently, our robot should able to precisely find its location despite the environment complexity. Navigating in the offshore platform requires to climb stairs and consequently achieve 6 DoF motion. As a result, the state of the robot must be known along the six DoF. Figure 12 illustrates motion achieved by the robot on the platform.

The robot must be able to patrol all day long and its performance should not be influenced by weather or daytime conditions. As a consequence, LiDAR sensors were preferred to vision sensors. Safety concerns also require real time processing on embedded processing units. A particular focus has been put onto algorithm implementation and processing unit capability usage.

Oil and gas platforms are really complex facilities. Everyday, hundred of thousands of tons of highly inflammable and explosive liquids are processed. Consequently, safety standards are really high. All facilities are surveyed and digitized in order to keep track of installation and maintenance operations. CAD models are highly valuable tools for employee and fireman training.
As a result, autonomous robot navigation in such infrastructures can be regarded as a localization problem rather than a Simultaneous Localization And Mapping (SLAM) since maps are a-priori known .

2 Related works

2.1 LiDAR-based perception in industrial environments

Robot localization based on LiDAR has been largely investigated. This section is focused on recent works dedicated to robot localization in industrial environments. (Reinke and Beinschob, 2013) proposed to use features such as corners or large lines or planes extracted from the 3D map. However, as seen in Figure 1, offshore platform environments do not have such features in the 3D map. Researches are actually being focused on dynamic environments such as parking lot. (Tipaldi et al., 2013) developed a method robust used in a parking lot context. The approach is able to deal with car motions and changes in parking space occupancy. Static but challenging environments are still an important research topic. (Jagbrant et al., 2013) and (Underwood et al., 2015) investigated robot localization in orchards. The scene is locally complex with irregular structure caused by trees. However, irregularities induce key points. From a larger standpoints, orchards can be regarded as corridors delimited by trees. (Stoyanov et al., 2013) performed robot localization in a milk production facility with a Velodyne HDL32. This scene is indoor and rather complex due to a lot of machines and small pipes. However, it will result in a lot of key points and large planes induced by the surrounding walls. The robot motion is performed onto a plane (3 DoF).

Offshore platform environment includes several levels, sparse environments with few walls. Smaller and round objects are also more challenging. In fact, LiDAR beams are less likely to hit the object surface or the ray not being reflected. To the best of our knowledge, there exists no work publicly available that tackles problems faced in the ARGOS challenge.

2.2 Robot localization in industrial facilities

There exists two situations in robot localization:

- The environment is unknown. This problem is known as SLAM (Simultaneous Localiza-

tion And Mapping). The robot must simultaneously map the environment and locate itself with the environment. First works on SLAM tackles simple environments such as offices or corridors. Flat ground surfaces require 3 DoF to represent the robot state (x,y,θ). Industrial and outdoor applications require more DoF. For instance, 6D SLAM was investigated by Nuchter in order to patrol in mines (Nüchter et al., 2006). The main issue with SLAM is the map distortion resulting from approximated localization and mapping. Consequently, loop-closure is often required to obtain a decent map of the environment and consequently localization. The reader can refer to (Cadena et al., 2016) for an updated state-of-the art of current SLAM techniques.

- The environment is already known, we consider an available a priori map. Safety and accuracy required on the robot localization to patrol on offshore platform cannot allow the use of SLAM approaches. In fact, a wrong localization can result in the falling of the robot into the sea or from one level to another. Consequently, such facilities are often surveyed and a 3D maps often exist for operational and training purposes.

Most of the existing works tackling robot localization in industrial facilities are emphasized on indoor, simple and planar environments. There are numerous approaches to solve localization in such environments. Magnetic or optic lines (Olivares-Mendez et al., 2011; Taghaboni and Tanchoco, 1988) can be placed on the ground. The robot has to follow those lines. However, robot motions are constrained to line following. Robots cannot avoid an obstacle being placed on the line. In order to allow more freedoms, beacons were place onto the infrastructure. Beacons have two drawbacks. First, they must be detected by a sensor. Secondly, they also require to equip the facility. As mentioned by (Sabattini et al., 2013), localization based on the existing environment would allow to expand autonomous robot applications to a larger set of situations.

Contrary to existing industrial solutions, beacon-based localization cannot be applied on oil and gas facilities as :

1. It would be extremely complex and costly to install beacons on existing infrastructures.
2. Six degree-of-freedom motion would induce to place a huge number of beacons in order to ensure LiDAR beam reflections.

2.3 Map data storage

3D maps are nowadays widely available. Data gathered by LiDAR correspond to 3D points. Once a map is build from multiple surveying LiDAR measurement or LiDAR embedded on a moving vehicle, a 3D point cloud of the environment is available.

3D data storage implies to store the data position and its value. There exists three different ways to represent data position :

- 3D grids: the geometrical structure is implicit. Position information does not require extra space.

- Octree: the hierarchical structure between the tree nodes is stored with pointers, requiring extra space.
- Point array: 3D coordinates are directly stored in memory.

2.4 Contributions of our work

Our work is focused on localization with a known 3D environment. We must be able to achieve 6 DoF robot localization with constrains regarding robustness and processing capabilities. We propose an approach able to locate a robot in environments that could be regarded both as indoor and outdoor. Offshore platforms do not correspond to outdoor environments which are often encountered in the literature such as urban environments, fields or forests. The environment structure is itself challenging for LiDARs. Contrary to existing works ((Nuchter et al., 2004), (Nüchter et al., 2006) and (Zhuang et al., 2013)), we do not use single layer LiDARs rotating along one axis actuated by a motor. In fact, this approach often requires to stop the robot motion to perform a full scan of the scene. Such behavior is not compatible with the real time processing and operability required in the ARGOS challenge. (Zhang and Singh, 2014) tackles this issue by performing LiDAR scans rectification with an approach similar to SLAM based on key points matching. As mentioned earlier, only view key points exist in our environment. Finally, most of the works use different flavors of ICP in order to match the LiDAR measurements and the 3D map. Still, ICP is not robust to environment changes and, as an iterative algorithm, would not be compatible with a mobile platform with limited resources. Our work introduces a new representation of the environment enabling real time processing on a platform with limited memory and CPU capabilities. In order to overcome these limitations, we replaced the environment map with a 3D representation of LiDAR impact likelihood named *3D likelihood field*. This representation allows to highly reduce the CPU load. In order to store huge 3D look-up tables and tackle limited memory challenges, we introduce the concept of *hybrid octree*.

3 Methodology

3.1 Theory of localization

The aim of localization is to determine the most probable state vector $\mathbf{X_t}$, given a map $\mathcal{M}$, our prior knowledge of the environment, and a sensor unit providing measurements $\mathbf{Z_t}$ of this environment. The state vector $\mathbf{X_t}$ is defined as follows:

$$\mathbf{X_t} = \begin{bmatrix} x & y & z & \psi & \theta & \varphi \end{bmatrix}^T \tag{1}$$

where:
x, y, z : Position in meter (m)
ψ, θ, φ : Orientation in degree ($^\circ$)

Localization methods aim at finding the most likely state vector among several hypotheses.

The likelihood function expresses the probability $\mathbb{P}(\mathbf{Z}|\mathbf{X},\boldsymbol{\mathcal{M}})$ of obtaining a measurement $\mathbf{Z}$, given a state $\mathbf{X}$ a our prior knowledge of the environment $\boldsymbol{\mathcal{M}}$.

Ideally, the likelihood function should be :

- Able to discriminate between several hypotheses in order to be able to keep only the most likely one.
- Monotonic and without local maxima in order to converge to the correct solution.
- With a moderate gradient in the direction of the solution in order to facilitate the convergence of a hypothesis close to the solution
- Low cost in terms of calculating time in order to test a large number of hypotheses quickly.
- Independent between states in order to be able to converge in each dimension separately.

For localization applications, there are a number of filters in the literature :

- Kalman and extended Kalman filters. They require the uncertainty of state models to be Gaussian.
- Histogram filters. Their computational complexities grow quadratically with the number of dimensions. It is consequently difficult to use them with 6 DoF.
- Particle filters. They are not limited to unimodal probability distributions of state vectors. The state vector space is continuous contrary to histogram filters. The number of particles vs dimensions tends to grow exponentially rather than quadratically in the case of histogram filters (Gordon et al., 1993; Thrun et al., 2001; Arulampalam et al., 2002; Thrun et al., 2005).

We have therefore implemented a standard particle filter (Thrun et al., 2005) to estimate the state vector $\hat{\mathbf{X}}_\mathbf{t}$. It issues hypotheses for $\hat{\mathbf{X}}_\mathbf{t}$, called particles, and select them according to the likelihood function. $\hat{\mathbf{X}}_\mathbf{t}$ is the result of the barycenter of the particles.

The main steps in particle filtering for localization are as follows :

- *Motion update* : A kinematic model is used to move the particles (issues new hypotheses for $\mathbf{X}_\mathbf{t}$). In this work, we use a differential drive kinematic model. We consider that the motion measurements are contaminated with a noise proportional to the displacement, modeled by normal distribution $\mathcal{N}(\mathbf{0},\Sigma_p)$ and an additive noise defined by $\mathcal{N}(\mathbf{0},\Sigma_a)$. We considered the noises as being uncorrelated between the robot states. Consequently Σ_p and Σ_a are diagonal matrices defined by vectors σ_p^2 and σ_a^2 respectively.

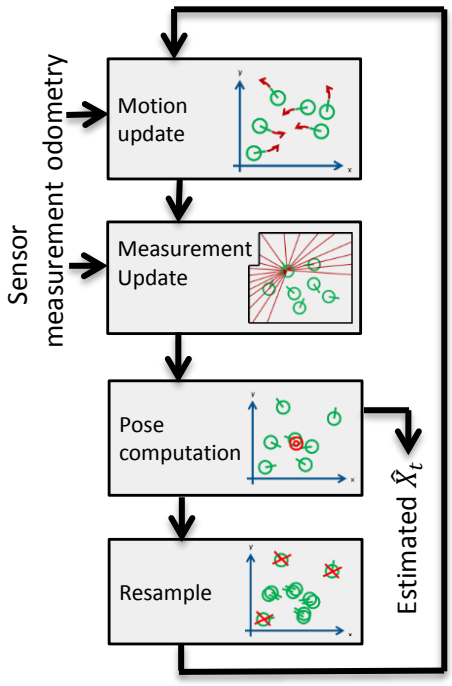


Figure 2: Main steps of particule filter-based localization

- *Measurement update* : a measurement from a sensor allows the likelihood function to evaluate a probability for each hypothesis (its weight). In our case $p(\mathbf{Z}|\mathbf{X}, \mathcal{M})$ is determined using Equation (5) from a LiDAR measurement.
- *Pose computation* : $\hat{\mathbf{X}}_\mathbf{t}$ is equal to the barycenter of each particle state $\mathbf{X}_\mathbf{t}$.
- *Resample* : The best particles are selected according to their weight and duplicated.

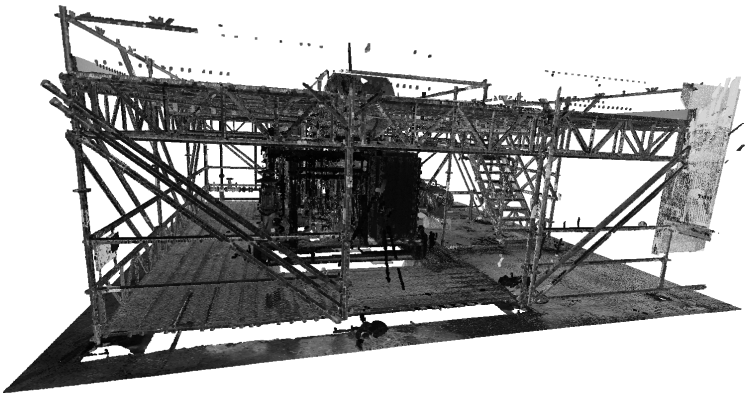
Section 3.2 describes a number of methods for evaluating the likelihood function. Then, we introduce a pre-calculated space of LiDAR impact probabilities called likelihood fields and propose a 3D variant.

3.2 A-piori map data representation and storage

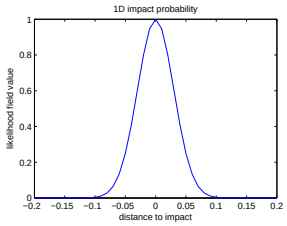
3.2.1 Likelihood theory

There are several ways of interpreting the information measured by a LiDAR. For example, (Levinson and Thrun, 2010) use the 3D information and the infrared reflectivity to localize a car within a pre-established map. The term "Multi-Level Surface Maps" (Triebel et al., 2006; Pfaff et al., 2007) is usually used for outdoor mobile robotics in a non-structured environment. It consists of comparing the height of the near environment to a pre-established map.

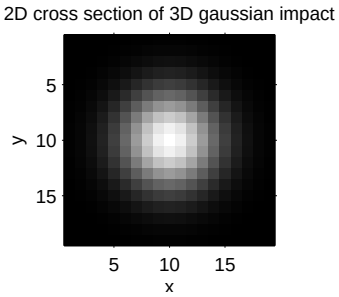
We started this work with an approach similar to (Fallon et al., 2012), i.e. a 3D simulation of the environment making it possible to calculate a distance error between the LiDAR measurements



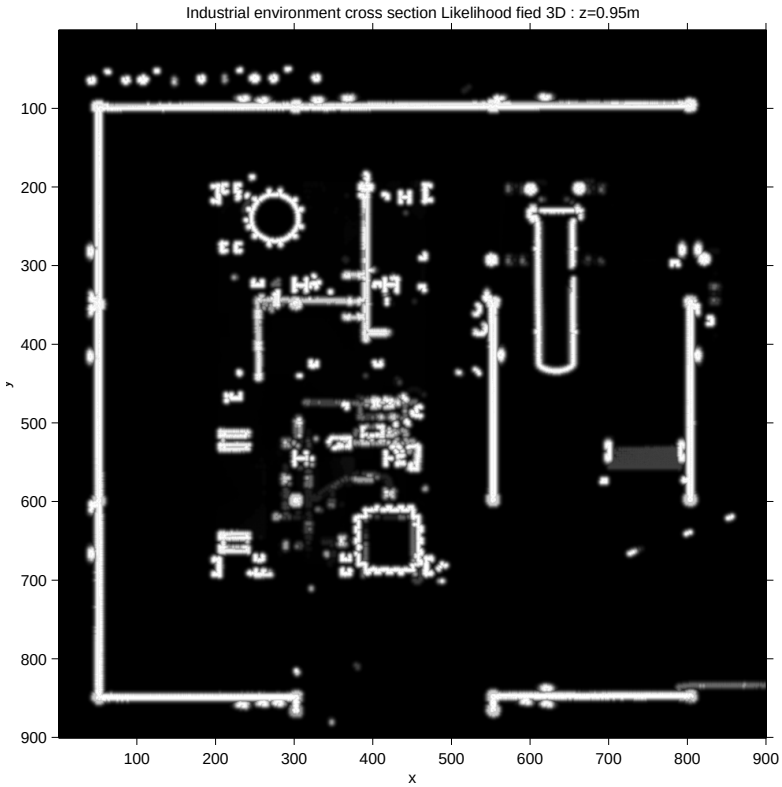
(a) 3D point cloud from a complex oil and gas site mapping. Point cloud data ©Total S.A.



(b) 1D likelihood LiDAR impact



(c) 2D cross section of 3D gaussian local likelihood LiDAR impact



(d) Cross section of the resulting likelihood field (voxel unit). Lines in the figure correspond to horizontal bars of the fence that exist at z=0.95m

Figure 3: 3D Likelihood field map construction : from environment mapping point cloud Fig 3a, we use the LiDAR variance Fig 3b, to compute a local 3D likelihood field Fig 3c, and merge all of them in a global likelihood field Fig 3d.

and the intersection of the facets of the scene: the scene is represented by a set of 3D facets. To evaluate a hypothesis of the particle filter, we simulate ray tracing for each LiDAR impact in order to determine the intersection with a facet of the scene. The product of the distances between the calculated intersection and the LiDAR impact gives $p(\mathbf{Z}|\mathbf{X},\boldsymbol{\mathcal{M}})$. Still, two problems arise:

- The environment being essentially made of a multitude of low diameter pipes, a large number of facets are needed to represent it which results in a costly ray tracing calculation for each LiDAR point.
- Another consequence of environment composed of small objects is a likelihood function sensitive to a variation in $\mathbf{X}$. For example a very small angular shift causes a very large variation in the likelihood score. As a result, despite a hypothesis close to the solution, the score suggests that the hypotheses is far from the true position.

In order to tackle these two issues, we propose to extend the use of likelihood field, proposed by (Thrun et al., 2005; El Hamzaoui, 2012), to 3D. The environment is discretized and represented by a 3D grid, and for each object making up the scene, we calculate the probability of LiDAR impact using a normal distribution :

$$p_{hit}(\mathbf{z}|m,O^k_{\boldsymbol{\mathcal{M}}}) = \frac{1}{\sigma_{map}\sqrt{2\pi}}\ \mathrm{e}^{-\frac{1}{2}\left(\frac{d}{\sigma_{map}}\right)^2} \tag{2}$$

where:

m	: likelihood cell
d	: Euclidean distance from cell m to k^{th} obstacle $O^k_{\boldsymbol{\mathcal{M}}}$ in $\boldsymbol{\mathcal{M}}$
σ_{map}	: map uncertainty

σ_{map} is set to take into account the map uncertainty, location uncertainty, measurement uncertainty (caused by raw measurement error and approximation of the measurement process) and ensures a smooth shaped likelihood function response.
Each cell m in the 3D grid of the map stores the impact likelihood as follows :

$$p_{hit}(m) = \max_{0\leq k\leq n_{obstacle}} p_{hit}(\mathbf{z}|m,O^k_{\boldsymbol{\mathcal{M}}}) \tag{3}$$

where:

$n_{obstacle}$	: number of obstacles in the environment.

The likelihood field $\mathscr{L}(\boldsymbol{\mathcal{M}})$, given the map $\boldsymbol{\mathcal{M}}$, is the 3D grid that stores all $p_{hit}(m)$ for any m resulting from the discretization on $\boldsymbol{\mathcal{M}}$.

The angular resolution of the LiDAR determines the number n of measured beams. Only a subset $\mathbb{L}$ of these beams will be valid and become 3D points. In fact, objects may be too far. Missing distances may be caused by absorbent surfaces or the low incident angles.

Thanks to Equation (3), the likelihood function $p(\mathbf{Z}|\mathbf{X},\boldsymbol{\mathcal{M}})$ is very easy to compute. First, it is necessary to start by projecting the LiDAR points $\mathbf{P}_{\boldsymbol{\mathcal{L}}}$ into the robot coordinate frame $\boldsymbol{\mathcal{R}}$ then in the map coordinate frame $\boldsymbol{\mathcal{M}}$ (Figure 16) :

$$\mathbf{P}_{\boldsymbol{\mathcal{M}}} = \begin{bmatrix} \mathbf{R}_{\boldsymbol{\mathcal{R}}\rightarrow\boldsymbol{\mathcal{M}}} & \mathbf{t}_{\boldsymbol{\mathcal{R}}\rightarrow\boldsymbol{\mathcal{M}}} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{\boldsymbol{\mathcal{L}}\rightarrow\boldsymbol{\mathcal{R}}} & \mathbf{t}_{\boldsymbol{\mathcal{L}}\rightarrow\boldsymbol{\mathcal{R}}} \\ 0 & 1 \end{bmatrix} \cdot \mathbf{P}_{\boldsymbol{\mathcal{L}}} \tag{4}$$

Secondly, values in $p_{\boldsymbol{\mathcal{M}}}$ are then allocated to a given m. Finally, one can calculate $\mathbb{P}(\mathbf{Z}|\mathbf{X},\boldsymbol{\mathcal{M}})$ directly by re-reading the likelihood field $\mathscr{L}(\boldsymbol{\mathcal{M}})$:

$$p(\mathbf{Z}|\mathbf{X},\boldsymbol{\mathcal{M}}) = \frac{(\sum_{i=\mathbb{L}} p_{hit}(m_i|\mathrm{p}_{i,\boldsymbol{\mathcal{M}}} \in m_i))^2}{n} \tag{5}$$

where:
$\mathrm{p}_{i,\boldsymbol{\mathcal{M}}}$: i^{th} beam of the LiDAR measurement $\mathbf{P}_{\boldsymbol{\mathcal{M}}}$.

Equation (5) takes into account coding capacity limits. In fact, LiDAR impacts being independent, the numerator should be a product. Still, as shown in (El Hamzaoui, 2012), sum operators should be used to ensure that the probability does not quickly converge to zero.

In this section, we have investigated how 3D likelihood fields can be computed. However, 3D grids are not the best data structure from a memory standpoint. In the following section, we investigate how octree-like data structure (Hornung et al., 2013) can be used to obtain a compact representation of the 3D likelihood field.

3.3 Likelihood field storage

Octrees are tree data structure.They are often used to represent 3D data. Each node has exactly 8 branches, known as octants. They can be used to store 3D point clouds of an environment (Hornung et al., 2013). Another tree structure is named KD-tree. Contrary to Octrees, kd-tree cells are not guaranteed to be cubical. kd-tree cells can have a large aspect ratio. Consequently, the concepts of neighbor and neighboring distance are different and cannot directly be obtained from the data structure. Tree structures enable to reduce the memory space occupied by the map as they do not store the empty space like 3D occupancy grids. Octrees enable to quickly find neighbors of a given point contrary to points that would be stored in an array.

The likelihood field represents the impact probability of a LiDAR ray. We have decided to store the probability on one byte. The smallest value that can be stored is 1/255. When $\sigma_{map} = 3cm$ is used, the probability goes down to zero when the ray is 19cm away from a real obstacle. Figure 4 shows none zero voxels of the likelihood field. Voxels 19cm away from objects in the map go to zero.

3D grids are really efficient when the 3D data structure is dense. Octrees are more interesting when the data is sparse. In fact, octrees require to store separately the values and positions represented by pointers.

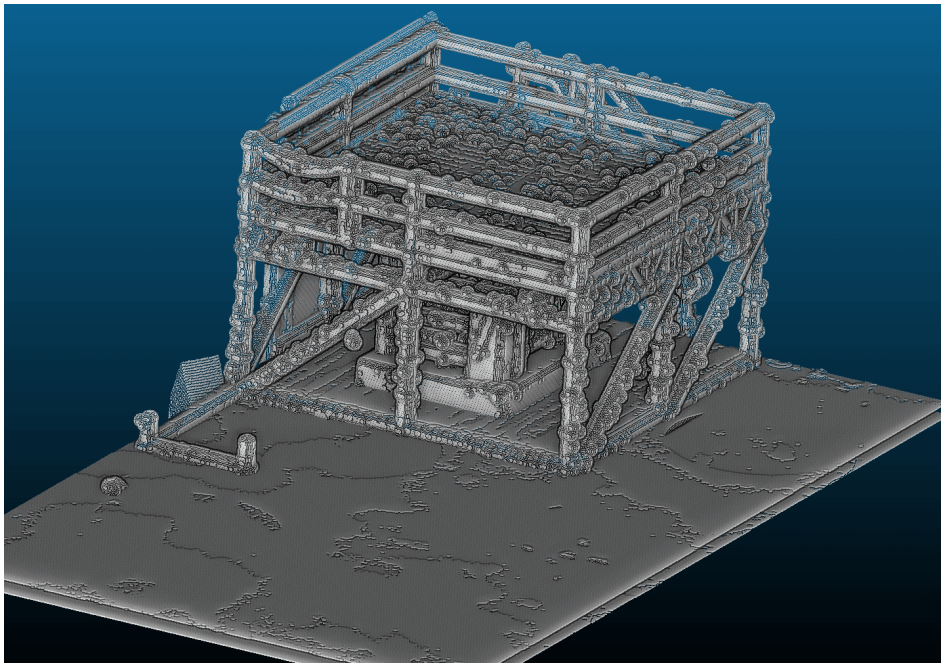


Figure 4: 3D representation of the environment stored by the octree (Figure 14a). Only regions where the likelihood is larger than zero are represented. Storage size of the likelihood : 8 bits, $\sigma_{map} = 3cm$. The likelihood field presented is the map used in our experiments

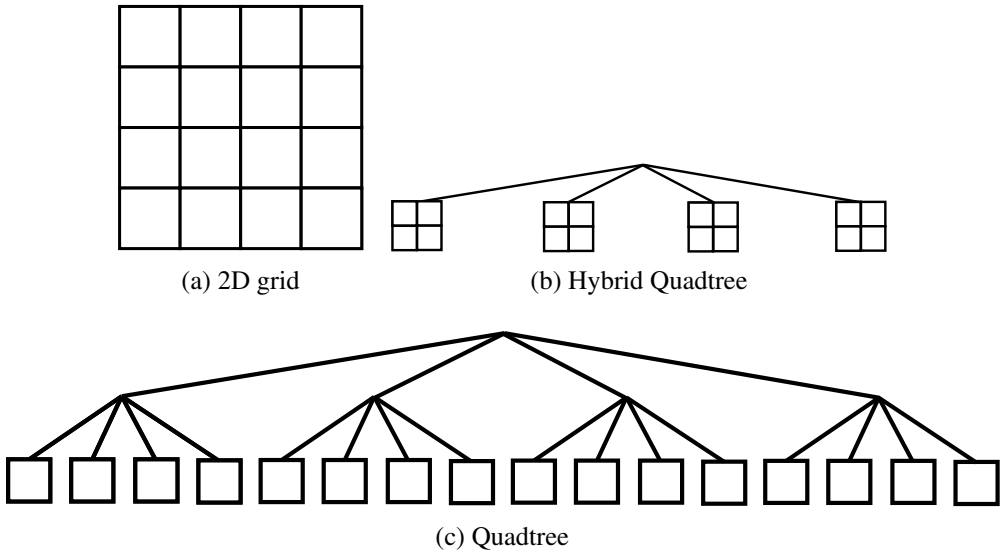


Figure 5: Illustration of the hybrid data structure concept in 2D. Figure 6 extends the concept in 3D.

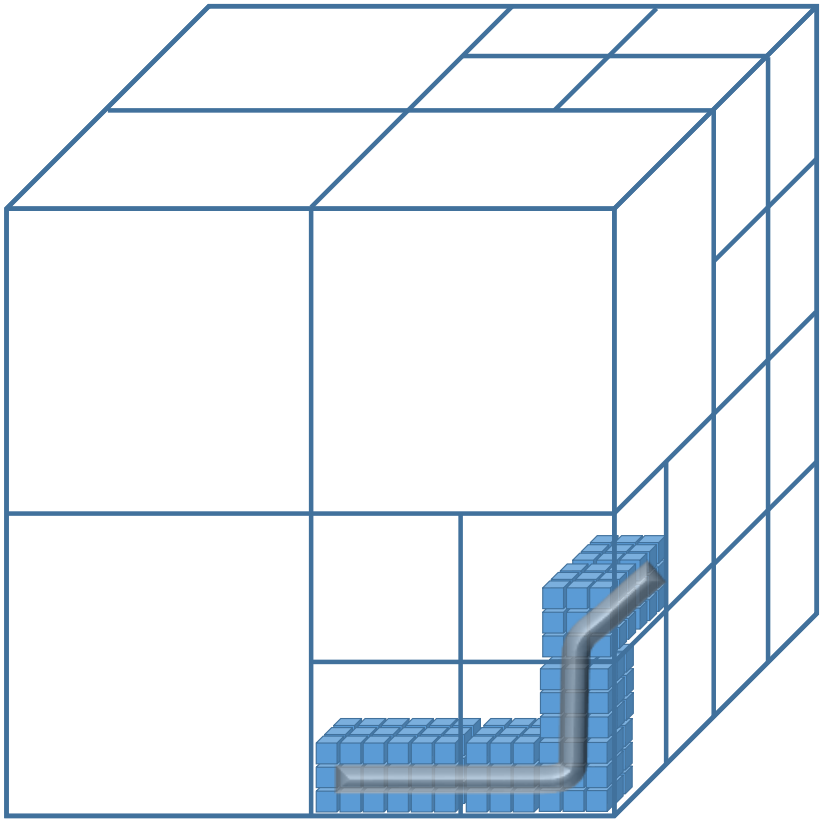


Figure 6: *Hybrid octree* optimized for likelihood field storage.The grey pipe storage is split in two parts. First, the octree is computed for a resolution R_o. Secondly, a 3D occupancy grid is used to store finer resolutions up to resolution R_a. As it can be seen in Table 1, the memory size is significantly reduced compared to a regular octree structure when applied to dense data such as likelihood fields

We are aiming at finding a data structure that would give enough flexibility to fit both memory and processing capability constraints. Consequently, we introduce a data structure that uses advantages of both octrees and 3D grids

The likelihood of the offshore platform is highly sparse as most of the non empty voxels are really close to real objects. Consequently, we propose to use a hybrid approach named hybrid octree. An octree is used to represent a given resolution R_o. Once R_o is reached, a 3D grid stores the information from R_o to R_a. Figure 5 shows the idea of hybrid data structure in 2D using Quadtrees. While regular quadtrees would store cells at resolution R_a. In this example, R_a is the finest resolution corresponding to individual cells of the 2D grid. Hybrid quadtree stops using octree structure at resolution R_o and then store 2D grids at the leaf level. In our illustration, $2x2$ grids are stored in the leafs. Figure 6 illustrates the extension of hybrid data structure to 3D, namely the *hybrid octree* concept.

We now focus on information access time. The map being built offline, we do not focus on map writing time. The data access time depends on the data structure used:

- 3D grid : 3 pointers independently of the map size stored.
- Octree : n pointers, with n being proportional to log_2 of the environment size.
- Hybrid Octree : $n - log_2(R_o)$ pointers plus one indice reading of the high resolution 3D grid. n representing the octree part is also proportional to log_2 of the environment size.

Memory usage and access time of the hybrid octree depend on the size of the 3D grid as well as the likelihood field type. Its performance is investigated in section 4.1.1.

4 Results

In this section, we first study the performance of the newly introduced data structure used to store the preprocessd a-priori map information. Secondly, robot localization performance is evaluted on several case studies.

4.1 Likelihood field storage performance

The performance is evaluated on the point cloud used for ARGOS lab test (Section 4.2.2). It corresponds to a volume of $261m^3$ (9,8mx7mx3,8m) (Figure 14).

4.1.1 Memory consumption

The likelihood field memory consumption depends on two factors : the environment itself and the likelihood field characteristics (spatial resolution and uncertainty in the map). The likelihood

3D occupancy grid size (R_o)	**Number of nodes**	**Number of 3D occupancy grids**	**Maximum # of 3D occupancy grids in the tree**	**Occupancy rate** (%)	**Memory size** (MB)
1^3	5105966	32747775	259940879	12.6	381.8
2^3	728712	4377254	32492610	13.5	83.4
4^3	110909	617803	4061576	15.2	45.3
8^3	18336	92573	507697	18.2	45.9
16^3	3468	14868	63462	23.4	58.3
32^3	727	2741	7933	34.6	85.7
Full 3D occupancy grid	–	–	–	–	247.8

Table 1: Memory size of the hybrid octree on a 64 bits operating system as a function of the occupancy grid size. Tests were carried-out in the build-up part of the likelihood in Figure 14. It includes $9.8 \times 7 \times 3.8m$ with $R_a = 1cm$ and 32.7 millions points to be stored.

field is built from a spatial resolution of 1cm and a σ_{map} equals to 3cm, resulting in 32,7 million likelihood of impacts to be stored. First, the impact of the final grid is evaluated. The grid size is changed with respect to the powers of 2 from 1 to 32 voxels. A grid size of one actually corresponds to an octomap structure that would store probabilities of impact in the final leafs. Strictly speaking, regular octomaps would store occupancy states and would require to compute probabilities of impacts. As the map is known a-priori, we directly store the impact probability in order to avoid extra computations during the localization process. This trick allows to highly reduce the computation cost.

Results are shown in Table 1. The occupancy rate is given by the number of grids used to store the likelihood field over the number of grids in a regular 3D occupancy grid. We can notice that the smallest sizes are achieved for a grid size of 4 and 8. Smaller sizes result in more pointers to be stored. Larger sizes produce a lot of empty cells in the grid. The point cloud becomes too sparse for larger sizes. We can notice that the hybrid octree with grid size equal to 4 or 8 enables to reduce the storage size by 82%.

The first line of Table 1 actually corresponds to a regular octree structure as the final leafs store a unique 8 bits probability. It can be seen that the required storage is larger than hybrid octrees. The large difference is mainly due to the fact that probabilities are spread around the actual location of the 3D points. The access time for a regular octree structure is displayed in Table 2, on the first line.

The full-scale test structure includes 471.6 million of points included in a volume of $11890m^3$. While a 3D occupancy grid and regular octree would use respectively 11.1GB and 4.75GB, a hybrid octree requires 595.2 MB, i.e. a reduction of 58% for the regular octree and 94.6% for the hybrid octree.

3D occupancy grid size (R_o)	**Tree depth**	**Acces time** (ms)	**Pointwise access time** (ns)
1^3	11	64.96	33.8
2^3	10	62.32	32.4
4^3	9	59.72	31.1
8^3	8	55.68	29.0
16^3	7	53.64	27.9
32^3	6	51.08	26.6
Full 3D occupancy grid	–	25.56	13.3

Table 2: Access time performance of the hybrid octree for 19.2 million point and one core of an *Intel i7-4700MQ @ 2.4GHz* CPU

To validate the storage efficiency, we also applied our approach to road environment. A HDL64e was used to scan an area of 398.7mx256.2mx14.7m around the lab. A 5cm resolution was used to build the likelihood field from 229 million of points. The hybrid octree uses 403.5MB while the 3D occupancy grid requires 11.2GB. The hybrid octree enables to save 96.4% of memory. We can notice that the larger the environment is, the more efficient the hybrid octree is with respect to a 3D occupancy grid.

4.1.2 Likelihood access time

In this section, we focus on the likelihood reading time as we consider that the likelihood field is built offline. Consequently, the robot missions would only require to read the likelihood of impacts. The theoretical access time depends on the octree depth. The hybrid octree depth depends on the environmental size and the choosen grid size :

$$P_{hybrid-octree} = \lceil log_2 \frac{\max_{\forall i} \mathrm{Dim}_i}{R_a} \rceil - log_2 R_o \tag{6}$$

With : R_a : Hybrid octree resolution. R_o : 3D grid size. Dim : Environment size

It is difficult to estimate the processing time as modern computers use cache memories and various hardware optimization techniques. Consequently, Equation 6 is theoretical as the performance actually depends on the octree implementation itself. Only a real measurement can give the actual performance.

As a result, we ran new experiments on the likelihood used in the previous section. The average access time is measured from 19.2 million LiDAR impacts simulated in the small-scale test structure. We discarded random 3D points as it could have resulted in empty space reading. It would result in less pointers and data to read and consequently faster access time. For instance, a 3D coordinate could result in the first pointer to be empty causing the reading to end.

In order to avoid any bias, the experiments were repeated 25 times. Figure 7 and Table 2 present

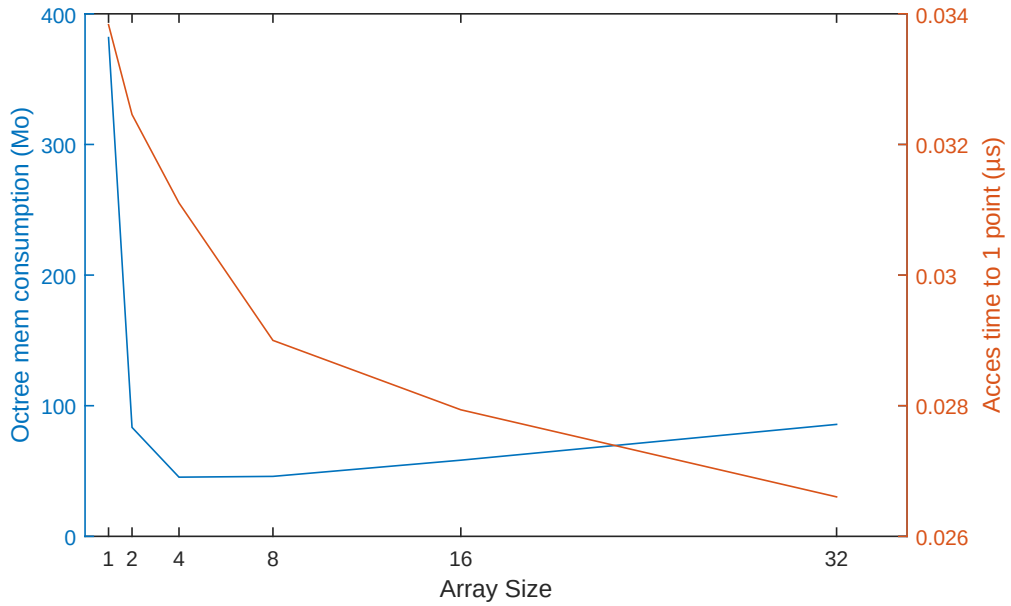


Figure 7: Memory consumption and processing time of the hybrid octree with respect to the 3D occupancy grid size. The best performance is achieved for occupancy grids of 4x4 or 8x8 elements. A regular grid map requires 248MB and results in a point access time of 13.3 microseconds

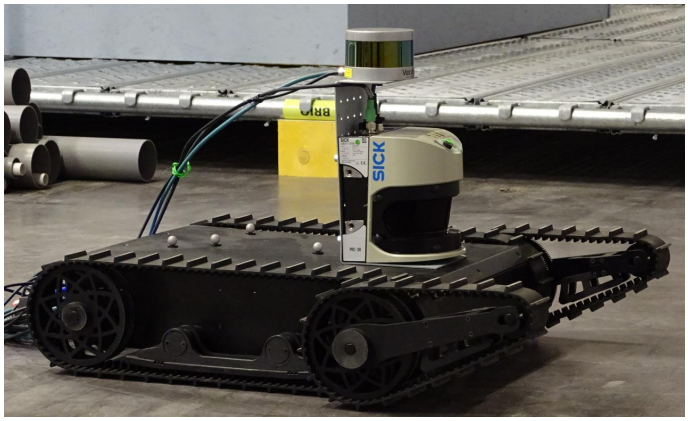
the results. As expected, a shallower tree produces a longer access time. Grid size of 4 and 8 gives a reading time 2.2 times larger than the 3D occupancy grid. This performance is fair with respect to the hybrid octree complexity. A grid size of 8 seems to be the best choice given our environment as it corresponds to the elbow criterion. Moreover, the hybrid octree, while requiring a memory space, gives reading times for single and multiple layers LiDAR compatible with real time applications:

- Sick LMS511, angular resolution: 0.5°, field of view: 190°, 381 rays per scan: 11 microseconds.
- Velodyne VLP16, angular resolution: 1°, field of view: 360x30, 5760 rays per scan: 167 microseconds.

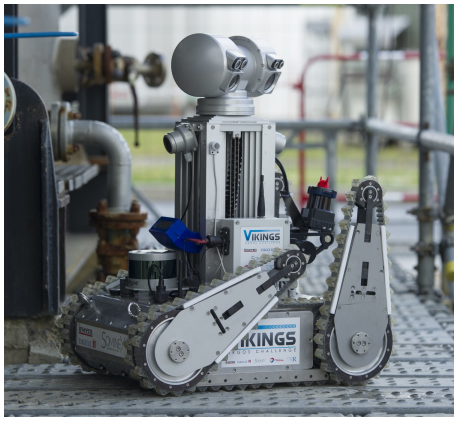
Reading times are given for these two specific models as they will be investigated in the next Section.

4.2 Robot localization performance

We developed this localization approach for the ARGOS challenge. First, we directly tested our approach on the real oil and gas site. Unfortunately, this site was not equipped with ground truth. As a result, we evaluated the localization precision in our lab. One mock-up of petrochemical



(a) Jaguar used for lab and single-layer versus multi-layer comparison



(b) Viking v1.2, VLP16 rear mounted, april 2016 at the competition site

Figure 8: Robots used to test our localization approach

facility was installed in the lab. Finally, we tune the algorithm main parameters with simulated trajectories in the real environment acquired with a Leica scanner.

The main steps are :

- Oil and gas site, contest in Lacq (France) : large real industrial facility on several levels, various lighting and weather conditions, numerous persons around the robot.
- Lab : precision evaluation in a mock-up of the petrochemical facility and ground truth with a Vicon system
- Simulations in the real environment : performance generalization and robustness to parameters and initial conditions.

Experiments were carried-out on two different platforms respectively named Jaguar and Viking (c.f Figure 8). The Viking robot was used during the ARGOS challenge contest. Experiments in the lab were achieved with the Jaguar robot[2]. Two LiDAR were evaluated in the Lab environment (c.f Figure 9). Number of layers measured by the LiDAR were investigated with a Sick LMS511 and Velodyne VLP16 respectively named SL LiDAR and ML LiDAR in the rest of this paper.

4.2.1 Oil and gas facility

As mentioned previously, the industrial site enables to perform a field test of the robot. The industrial site (c.f. Figure 1b) is full-scale facility. Extra challenges were introduced with dynamic objects such as people around the robot (c.f. Figure 10a) and difficult weather conditions including sunny weather variants and firemen spraying the robot (c.f. Figure 13). Extra static obstacles were also introduced randomly by the jury (c.f. Figure 10b)

[2] `http://jaguar.drrobot.com/specification_V4.asp`

(a) Single layer lidar, Sick LMS511 : 190°fov. ©Sick AG

(b) Multiple layers lidar: Velodyne VLP16, 16 layer, -15°to 15°, 360°fov. ©Velodyne Inc.

Figure 9: LiDARs used in our experiments

The method was used during the Argos Challenge on the Viking Platform. It took place outdoor in the environment which gave the surveyed point cloud in Figure 3a. The robot ran for about 16 hours correspoding to a distance of 7.8km, with a mix of manual and autonomous control (Figure 11). Localization was performed at 20Hz, with 18084 LiDAR points processed every second and 500 particles. The embedded CPU was a intel i7-4600U 2.1GHz, and CPU load was 16%. The Viking robot won the first two years of the ARGOS Challenge.

The particle filter was the same for all experiments. The number of particles was set to 500. Resampling was performed in order to avoid particle deprivation. We added Gaussian additive noise in order to add variability in the particle states.

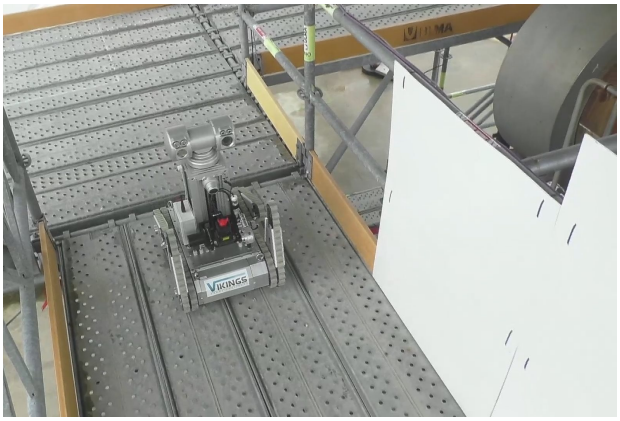
Lacq facility map data includes 471 millions of points for a corresponding hybrid octree structure of 595.5 MB.

No localization failure occurred despite large environment modifications added by the jury and dynamic obstacles such as numerous people in the scene. The localization performance seems fair enough to perform remote robot control and autonomous tasks such as obstacle localization and manometer readings from a PTZ camera. The next section will highlight the precision of our approach.

4.2.2 Lab performances quantification

The industrial test site did not provide a ground truth system. As a result, we performed complementary experiments in the lab equipped with a Vicon system to assess the localization accuracy (Figure 14b).

(a) Dynamic environment with numerous persons



(b) White boards not in the map introduced by the jury

Figure 10: Static and dynamic changes introduced by the jury

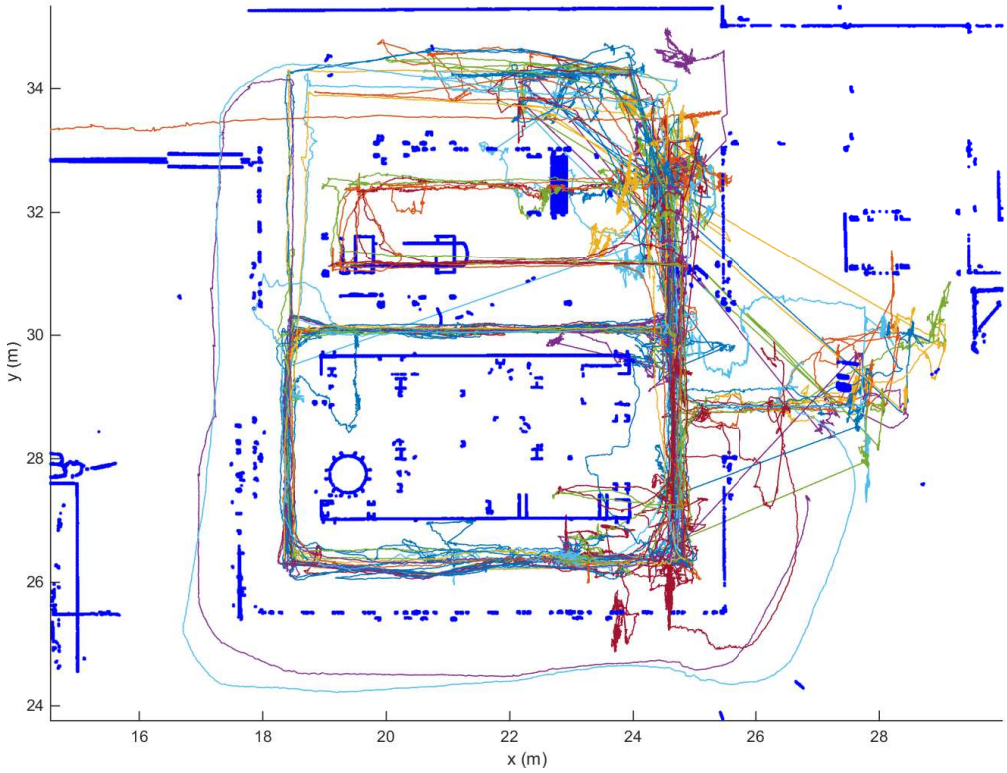
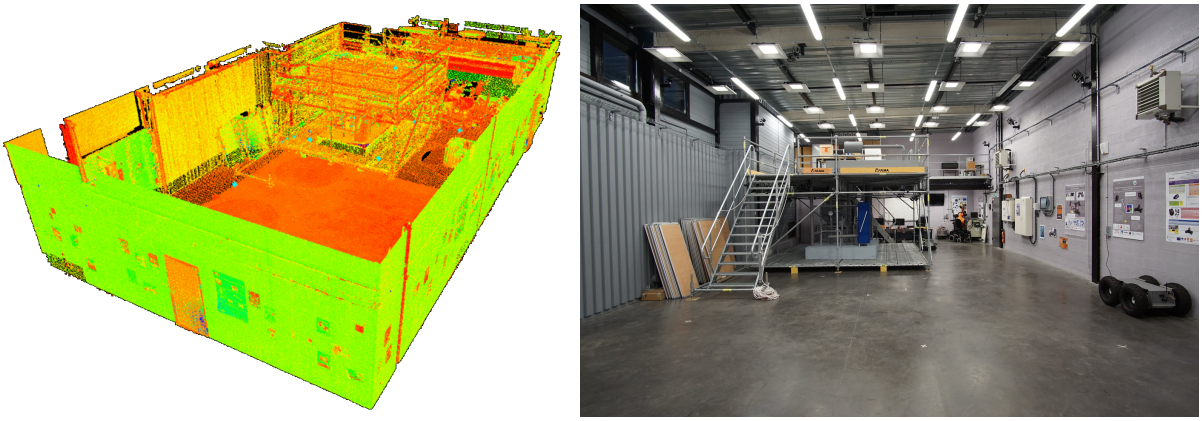


Figure 11: Bird-eye view of the trajectories of Viking robot. Thick blue lines and points correspond to map elements located at heights between 0.3m and 0.9m. Thin colored lines correspond to different trajectories. They represent 16h and 7.8km. Straight lines represent wrong robot location initializations. Some missions did not start at the default starting region of interest (ROI). Starting ROI were manually updated after robot starts.

Figure 12: 6 DoF motions faced by the robot during the ARGOS contest. Top: stair climbing in the contest site. Bottom: obstacle management in the lab

(a) (b) (c)

Figure 13: The robot in rainy conditions



(a) Point cloud from Leica C10

(b) Picture of our Lab

Figure 14: Irseem autonomous navigation laboratory including a oil and gas facility mock-up equipped with Vicon ground truth

Impact of the LiDAR technology (SL *versus* ML) was also investigated.

A Jaguar robot was used to test planar trajectories (Figure 8a), and the Viking robot for 6D motions (Figure 12). Contrary to the Jaguar, the VLP16 is installed looking backward. 110° of its horizontal FOV is occluded by a mast. The data was collected in the Autonomous Naviguation Laboratory of IRSEEM. As it can be seen in Figure 14b, a part of the room is equipped with a mock-up of oil and gas facilities. A 20-camera VICON T40S[3] is installed in the room and provides a 6 DoF localization ground truth. While the robot position ground truth was recorded at 100Hz, robot odometry and LiDAR measurements were recorded at 25Hz. RTMaps[4] was used to record and synchronize the data provided by the robot sensors. The $350m^3$ Autonomous Naviguation Laboratory was mapped with a Leica ScanStation C10[5]. The Leica measurement uncertainty is given with a 6mm standard deviation. The resulting point cloud was then processed in order to obtain its likelihood field (Section 3.2).

The similarity between the true state $\mathbf{X}_t$ and the state estimated by the particle filter $\hat{\mathbf{X}}_\mathbf{t}$ is computed separately for the position and orientation parts of the state vector. The Euclidean distance is used to compare the positions and the 3D solid angle is used for the orientations. The 3D solid angle Ω is computed as follows:

$$\Omega = \arccos \frac{\mathbf{X}_t \cdot \hat{\mathbf{X}}_t}{\|\mathbf{X}_t\| \cdot \|\hat{\mathbf{X}}_t\|} \tag{7}$$

Several planar trajectories (A-D, Figure 15) were recorded with a SL LMS511 LiDAR on the Jaguar platform. An other 6D trajectory E was performed with Viking platform, including step climbing/downing and obstacle crossing. Because of track drifting, the motion update is aided by

[3] http://www.vicon.com/System/TSeries

[4] http://intempora.com/

[5] http://www.leica-geosystems.fr/fr/Leica-ScanStation-C10_79411.htm

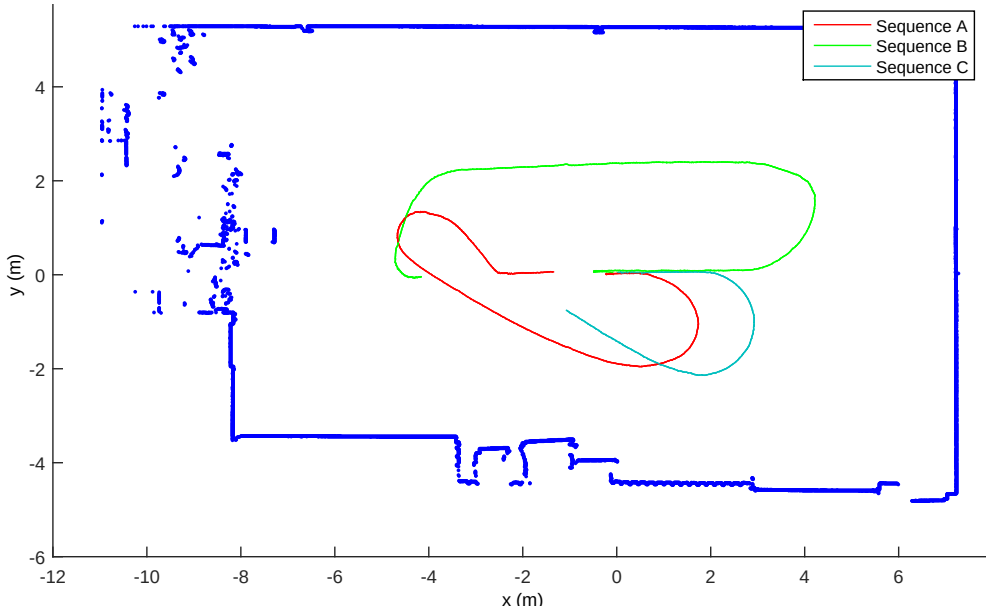


Figure 15: Trajectories used in lab experimentations

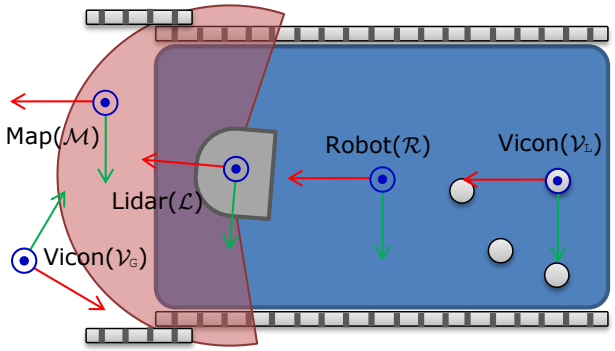


Figure 16: Reference frames used in the lab experiments

a low cost IMU. The IMU is only used during motion update to estimate the rotations about the x-axis and y-axis while rotations about the z-axis are estimated from the differential drive kinematic model. As it can be seen in Figure 16, the LiDAR reference frame is not at the same place than the robot mechanical center. Consequently, a transformation is required to project the LiDAR measurements into the robot reference frame.

500 particles were used to localize the robot. Results are presented in Table 3. In order to compare the performance in the first and second environments, we have adopted two data sampling approaches.

The first approach uses all the data provided by the sensors over four sequences (A-C,E). It results in a localization every 3 to 7mm. During sequences A-C, the performance is in the same order of magnitude than simulated robot trajectories in the first environment. In sequence E, with 6D motions and ML LiDAR, the performance is also close to simulation.

Sequence D tests the influence of spatial sampling instead of temporal sampling. Localization is done every 13cm. We used sequence A and subsampled the LiDAR measurements and integrated the odometry information in order to reach 13cm displacements. The position error is still really close to our first experiments. It can be explained by our lack of transformation between the LiDAR measurements and the robot mechanical center. Still, the error remains low. As a result, our localization approach is robust to both types of sampling and environments.

Table 3: Real experimentations results in 350m^3 Irseem autonomous navigation laboratory, 500 particles

Sequence number and length (m)	sampling type	Position error (m)				Orientation error (°)		
		initial	mean	std	RMSE	mean	std	RMSE
A : 14.97	25Hz	0.57	0.0106	0.0054	0.0119	0.963	0.567	1,117
B : 17.71		0.82	0.0192	0.0128	0.0231	0.804	0.389	0.893
C : 8.71		0.28	0.0178	0.0096	0.0202	0.890	0.435	0.991
D : 14.97	0.13m	0.57	0.0279	0.0185	0.0334	0.959	1.699	1.951
E : 22.5	25Hz	0.15	0.0236	0.0113	0.0261	0.292	0.162	0.334

This section confirms the localization precision obtained on the real test site. The localization method works on different robot platforms. Evaluation of algorithm intrinsic parameters being difficult to assess from real experiments, we simulated robot motions and measurements in the real environment point cloud.

4.2.3 Simulation evaluations

We ran simulation to assess the impact of various parameters in a decoupled manner. Simulation was preferred as real test conditions might not allow to purely decouple the following aspects :

- Discriminant property of the likelihood function.
- Localization convergence speed.
- Influence LiDAR measurement noise.
- Influence of the particle number.

Influence of such parameters was evaluated on both SL and ML LiDARs.

We simulated robot trajectories in a digitized environment resulting from the point cloud provided by the ARGOS Challenge organizers (Figure 3a). The resulting environment was also used as the

Table 4: Likelihood function robustness evaluation with SNR

		Pose_1		Pose_2	
		SL	ML	SL	ML
	Number of LiDAR beams per measurements	381	5760	381	5760
	Ratio of valid LiDAR beams	0.90	0.61	0.47	0.60
DoF variations	t_x, t_y	40.32	18.29	5.11	17.84
	t_x, t_z	31.69	41.66	22.55	44.10
	t_y, t_z	14.85	25.64	18.47	27.75
	t_x, R_z	36.77	14.24	3.54	14.14
	t_x, R_y	23.66	23.14	6.9	24.55
	t_x, R_x	24.75	19.19	11.26	17.79
	t_y, R_y	5.52	10.24	6.73	11.52
	t_y, R_x	6.56	11.28	11.16	10.52

map during the test performed in the real environment. Both LiDAR technologies were simulated using ray tracing algorithm with a Gaussian noise $\mathcal{N}(0, \sigma^2_{lidar})$.

Likelihood Function

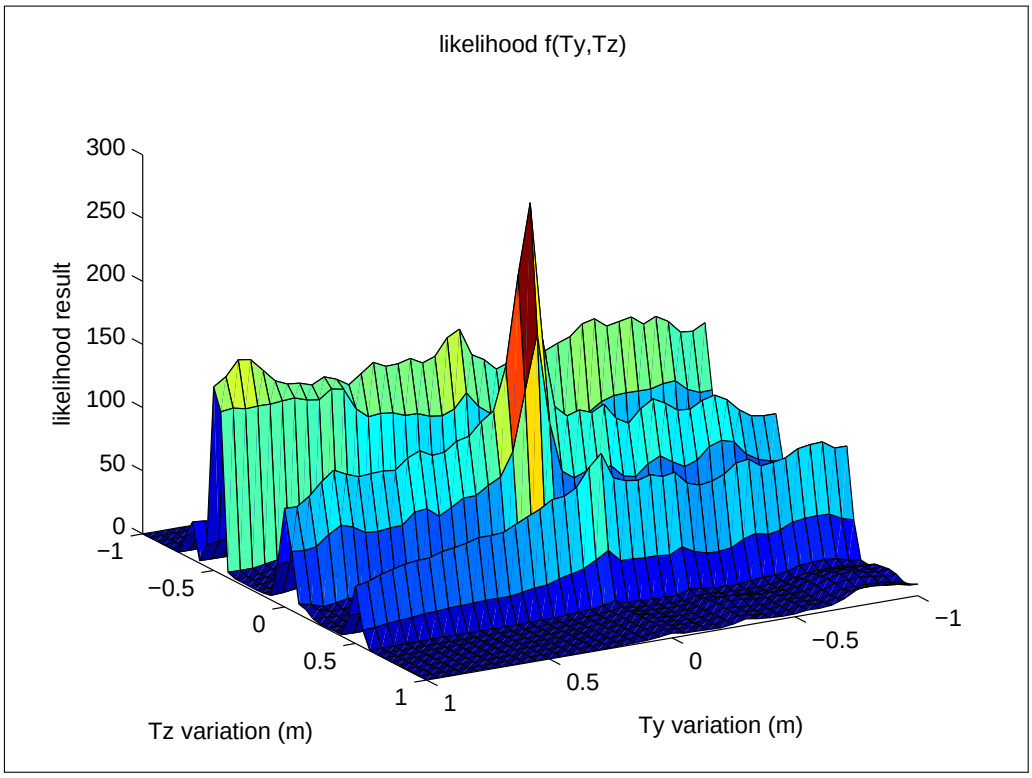
The ideal properties expected for the likelihood function used in the particle filter stated in Section 3.1 depends not only on the formula of the likelihood but also on both the environment and sensor. Figure 17 illustrates the case of a translation along the vertical axis. To highlight the characteristics of the likelihood function, we took a reference point in the environment and vary 2 DoF (from $[-1m, 1m]$ for translations t_i and $[-20°, 20°]$ for rotations R_i). We define this interval as $\mathbf{I}$. We use the signal-to-noise ratio (SNR) as an indicator of the likelihood robustness. It is computed as ratio between the value of the likelihood function at the reference point $\mathbf{X}_{ref}$ over the average on the interval considered :

$$\text{SNR} = \frac{p(\mathbf{Z}|\mathbf{X}_{ref}, \boldsymbol{\mathcal{M}})}{\overline{p}(\mathbf{Z}|\mathbf{X_i}, \boldsymbol{\mathcal{M}}, i \in \mathbf{I})} \tag{8}$$

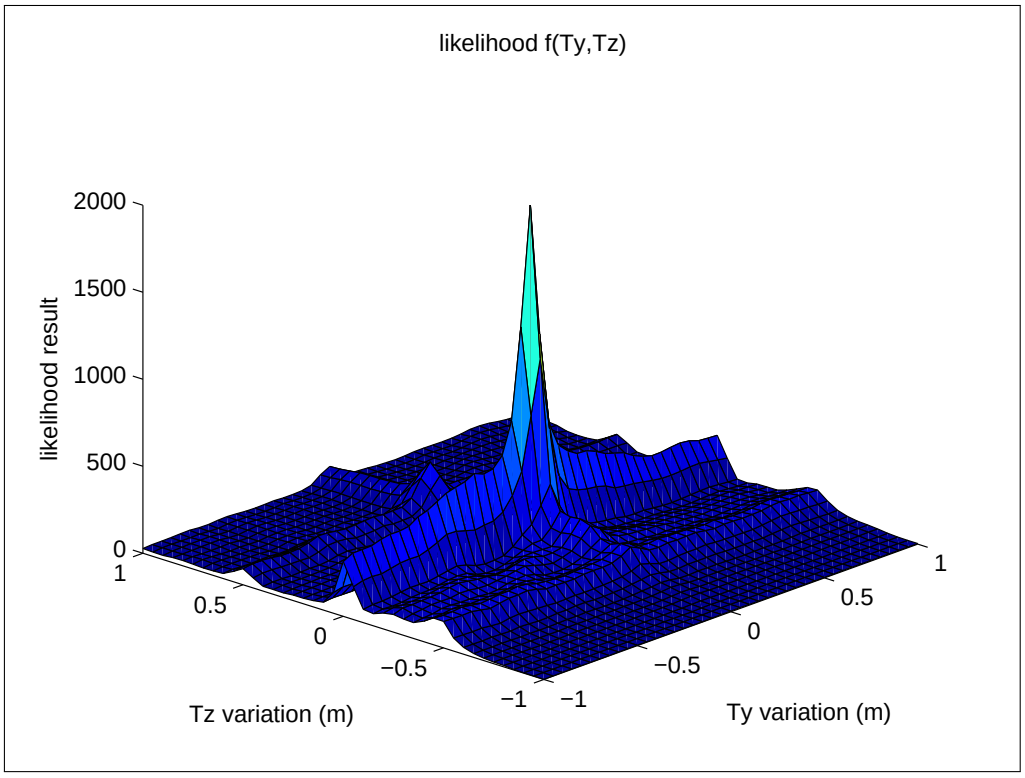
The evaluation in Table 4 was carried out for 2 different poses :

- Pose_1 : the single-layer LiDAR cuts the guard rail which goes around the structure.
- $\text{Pose}_2 = \text{Pose}_1 + \begin{bmatrix} 0 & 0 & -0.1 & 0 & 0 & 0 \end{bmatrix}^T$: under the guard rail, the single-layer only has the impacts on one side.

The SNR of the ML is much more constant between the two poses. In fact, the root mean squared



(a) SL vertical translations



(b) ML vertical translations

Figure 17: Likelihood evaluation for single layer (*SL*) and multi layers (*ML*) LiDAR in a corridor of the oil and gas environment Figure 3a

deviations between $Pose_1$ and $Pose_2$ are 15.52 and 1.54 for the SL and ML LiDARs respectively. The SL is disturbed by the distribution of the impacts as its field of view is limited to a line.

Figure 17a shows that the SL LiDAR in our environment is not very discriminating with regard to translations along the z-axis. The ML LiDAR achieves a better performance (Figure 17b). Overall, the likelihood computation with the ML LiDAR shows that the information provided is more robust and discriminant.

Convergence study

To test the convergence of our approach, we consider the situation of a "lost" robot. We initialize the particle filter to a given pose X_{init} with an error (ε_{pos}) from the true position X_{truth} and we observe the behavior of the localization algorithm as follows :

$$X_{init} = X_{truth} + \varepsilon_{pos} \tag{9}$$

The performance evaluation is done with the same metrics as in Section 4.2.2: euclidean distance for position and 3D solid angle for orientation.

In this case, there is no information for the *motion update*, but just an additive noise to ensure convergence. The particles are initially spread with $\mathcal{N}(\mathbf{0}, \Sigma_i)$ with Σ_i a diagonal covariance matrice defined but $\sigma_i = abs(\varepsilon_{pos})$. Noises in the state transition model are set to $\sigma_p = \mathbf{0}$ and $\sigma_a = \sigma_i/10$.

Figure 19 shows the accuracy of convergence with regard to position and orientation as a function of the number of particles for an initial positional error of $\varepsilon_{pos} = \begin{bmatrix} 0.2 & -0.2 & 0.05 & 2 & -2 & 5 \end{bmatrix}^T$. The convergence is calculated over 200 iterations of *measurement update*, and the result is determined with the mean and standard deviation of the last 50 iterations. These tests are repeated 8 times in order to overcome the variability of the results when the number of particles is low. The average of the 8 tests is shown. The LiDAR measurement noise is randomly sampled from $\mathcal{N}(0, \sigma_{lidar})$ with $\sigma_{lidar} = 0.01m$. The ideal number of particles is between 250 and 500.

Different studies are proposed in Figure 20. While the number of particles is set to 500, we vary the LiDAR noise σ_{lidar}. The methodology (convergence over 200 iterations and multiple tests) is similar in all respects. The accuracy of our method using the multi-layer LiDAR is hardly influenced at all by the noise of the sensor, only the localization error variance increases. On the other hand, the single layer LiDAR, which does not have a larger view of the environment is strongly disturbed.

Robustness to larger initial positionning error has also been carried out by significantly altering ε_{pos}. For example with $\varepsilon_{pos} = \begin{bmatrix} 1 & 1 & 0.1 & 5 & -5 & 5 \end{bmatrix}^T$, the ML always converges while the SL no longer converges. With a more moderate position error but the same orientation error, $\varepsilon_{pos} = \begin{bmatrix} 0.5 & 0.5 & 0.05 & 5 & -5 & 5 \end{bmatrix}^T$, the SL converges correctly for the orientation or position but rarely both at the same time. While for a strong position error and a smaller orientation error, $\varepsilon_{pos} = \begin{bmatrix} 0.5 & 0.5 & 0.05 & 1 & -1 & 5 \end{bmatrix}^T$ it converges every time. The single-layer LiDAR is much more sensitive than the multi-layer LiDAR to the DoFs over which it only partially perceives the environment : t_z, $\mathbf{R}_x$ and $\mathbf{R}_y$.

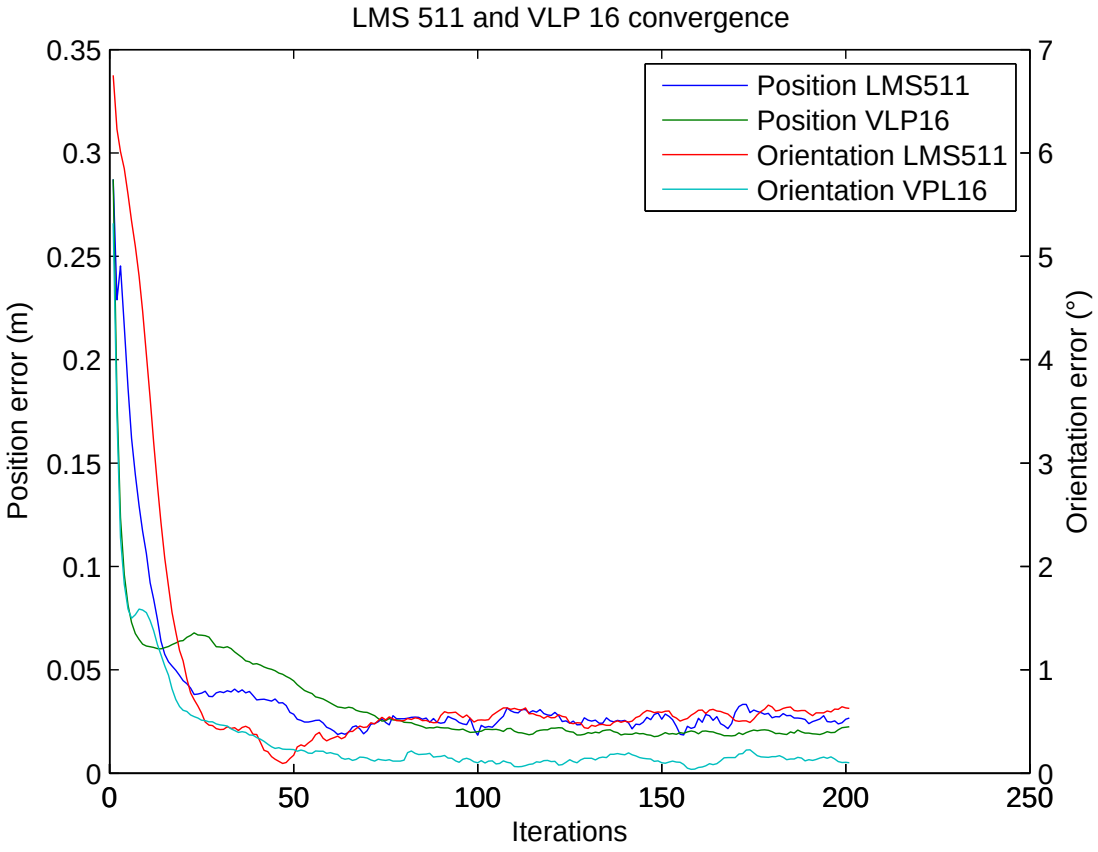
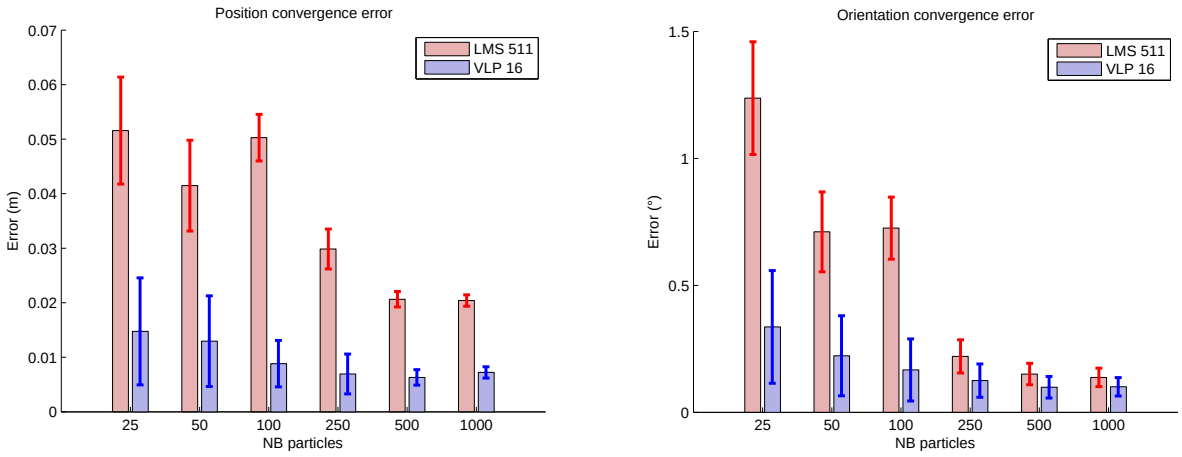


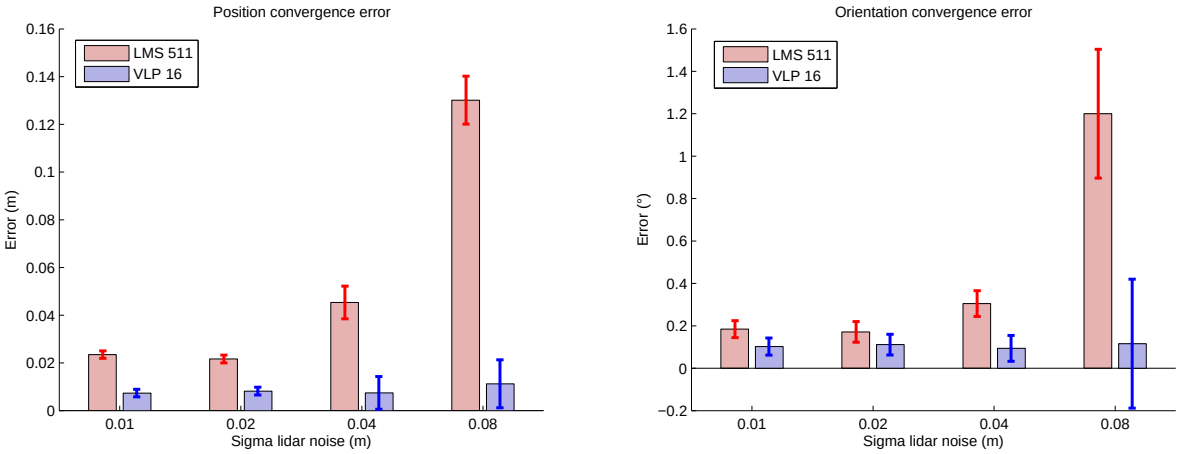
Figure 18: Single-layer and multi-layer position and orientation convergence : *1000 particles,* σ_{lidar}*=1cm*



(a) Position convergence error

(b) Orientation convergence error

Figure 19: Mean and standard deviation convergence precision with respect to the number of particles.



(a) Position convergence error

(b) Orientation convergence error

Figure 20: Mean and standard deviation convergence precision *versus* σ_{lidar}.

	SL LiDAR			ML LiDAR		
	mean	std	RMSE	mean	std	RMSE
x-y plane error (m)	0.0326	0.0041	0.0328	0.0050	0.0025	0.0055
z direction error (m)	0.0145	0.0049	0.0153	0.0046	0.0018	0.0049
3D error (m)	0.0369	0.0043	0.037	0.0071	0.0024	0.0075

Table 5: In-plane and out-of-plane mean absolute errors in meter averaged over eight runs during convergence. The numbers in brackets refers to the standard deviation. Particle set size set to 500.

Table 6: Localization performance on a 22.1m trajectory with 500 particles and $\sigma_{lidar} = 1cm$.

	Position error (m)			Orientation error (°)			computation time
	mean	std	RMSE	mean	std	RMSE	Intel i2640M-2.8GHz (ms)
SL	0.0223	0.0145	0.0265	0.45	0.81	0.926	0.9 (381 lidar impacts)
ML	0.0157	0.0120	0.0197	0.31	0.44	0.538	2.2 (5760 lidar impacts)

Table 5 shows the error components along the x-y plane and out-of-plane error. It corresponds to the average performance over 8 convergence studies. It can be seen that the SL LiDAR error with respect to the ML LiDAR error is in the same order of magnitude than in previous results. Still, as the ML LiDAR is based on several layers, allowing to have a wider FOV along the z-axis, one could imagine that the ratio z-axis error over x-y-plane error would be in favor of the ML LiDAR. Strictly speaking, this ratio is equal to 0.44 for the SL LiDAR and 0.92 for the ML LiDAR. Still, the error is so small for the ML LiDAR that we could not conclude that this result is truly meaningful.

Localization
To test the complete localization process, we simulated a robot loop trajectory in the first floor pathways. It results in 160 robot poses associated with the corresponding odometry (Figure 21a) and LiDAR measurements of the environment in Figure 3a. Its length is 22.1m. The LiDAR noise was set to $\sigma_{lidar} = 0.01m$. The odometry is contaminated with $\sigma_p = \begin{bmatrix}0.1 & 0 & 0 & 0 & 0 & 0.2\end{bmatrix}^T$ (Figure 21b). The results are obtained with 500 particles. The initial position is altered from $\varepsilon_{pos} = \begin{bmatrix}0.5 & -0.5 & 0.05 & 1 & -1 & 5\end{bmatrix}^T$ (Figure 21c).

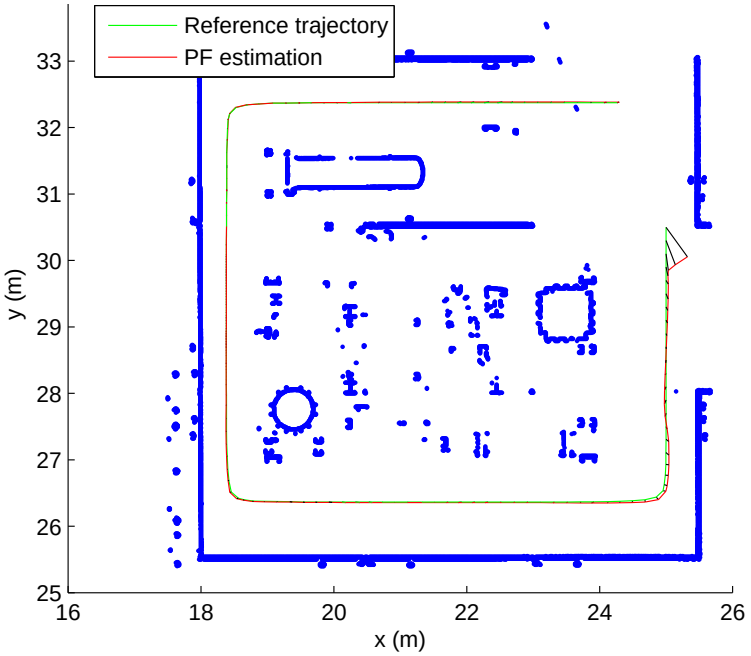
Despite an initial position error of 0.71m and orientation error of 5.1°, the average position error is less than 3cm and the average orientation error is less than a degree as shown in Table 6. The ML LiDAR outperforms the SL in mean position error and standard deviation.

As it can be seen in Table 6, the calculation times of the likelihood function obtained with Matlab is low.

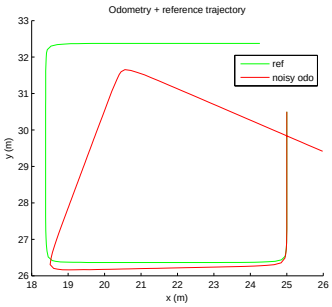
Results found in simulation show robustness of our approach confirming the absence of localization failure in the industrial facility (Section 4.2.3). This section also highlights that the ML LiDAR outperforms the SL LiDAR in complex environment with 6DoF trajectories.

5 Conclusion

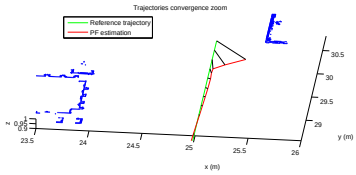
In this paper, we proposed a fast and robust 6 DoF localization approach based on an extension of likelihood fields to 3D spaces. The likelihood field is stored in an optimized data structure introduced in this paper. Our results suggest that the proposed data-structure and associated likelihood function are more suited to applications involving complex environments and embedded systems. The performance was successfully validated during the ARGOS challenge contest. Further investigations were carried-out in a lab as well as simulations in order to show the robustness.



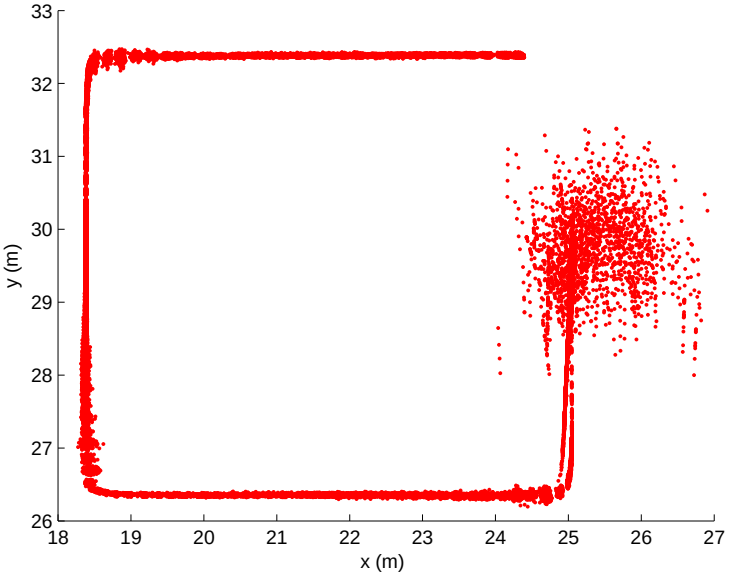
(a) Reference *(green)* and particle filter estimated *(red)* trajectory, cross section of environnement *(blue)*



(b) Reference trajectory and noisy odometry integration



(c) Particle filter convergence zoom



(d) particles cloud iterations

Figure 21: Localization study in simulated oil and gas environment for the single-layer LiDAR, with 500 particles and $\sigma_{lidar} = 1cm$.

Several experimental foundings were highlighted during this study. First, despite a challenging environment made of pipes and few reflective objects, LiDAR measurements are not so noisy and sufficient enough to give good results when used in the localization process. We used only LiDAR impacts to localize our robots. Out-of-sight measurements or the fact that a measurement was empty and expected as empty in the map was not used in the likelihood. Consequently, localization might be performed from few cues in the environment. Secondly, performing multitasking operation on an embedded system can be done while exploiting a large amount of LiDAR data to perform localization in real time while leaving enough processing power to achieve the other tasks. The performance was validated both in lab and on real infrastructures. Thirdly, 6DoF localization based on particle filters requires a lot of care. In fact, as shown in Section 4.2.3, the particle set size and the different noise levels involved in the particle filter must be properly tuned in order to maximize the performance. In our case, a set size of 500 particles was found to be enough as it does not improve the positioning performance. Forthly, a lot of effort must be focused on the measurement update of the particle filter in order to reach real-time processing with low CPU and memory usage. The proposed solution implies an innovative map data storage structure. It exploits the fact that a map of the environment is already available.

Future works will focus on managing huge scene such as petroleum refineries involving online loading of several octrees depending on the robot position. We will also investigate if our framework can be transposed to other domains such as underground parking garages.

Acknowledgement

This study is a part of ARGOS challenge which is organized by Total in partnership with the French National Research Agency (ANR). Point cloud used in the first environment (Figure 3a) belongs to TOTAL.

References

(2015). Robotics 2020 multi-annual roadmap. Technical report, SPARC.

Arulampalam, M. S., Maskell, S., Gordon, N., and Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188.

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I. D., and Leonard, J. J. (2016). Simultaneous localization and mapping: Present, future, and the robust-perception age. *CoRR*, abs/1606.05830.

El Hamzaoui, O. (2012). *Localisation et cartographie simultanes pour un robot mobile quip d'un laser balayage: CoreSLAM*. PhD thesis, Ecole Nationale Suprieure des Mines de Paris.

Fallon, M. F., Johannsson, H., and Leonard, J. J. (2012). Efficient scene simulation for robust monte carlo localization using an rgb-d camera. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1663–1670. IEEE.

Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F - Radar and Signal Processing*, 140(2):107–113.

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., and Burgard, W. (2013). Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Auton. Robots*, 34(3):189–206.

Jagbrant, G., Underwood, J., Nieto, J., and Sukkarieh, S. (2013). Lidar based localisation in almond orchards.

Levinson, J. and Thrun, S. (2010). Robust vehicle localization in urban environments using probabilistic maps. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 4372–4378. IEEE.

Nüchter, A., Lingemann, K., Hertzberg, J., and Surmann, H. (2006). 6d slam–mapping outdoor environments. In *IEEE International Workshop on Safety, Security, and Rescue Robotics (SSRR). IEEE Press*.

Nuchter, A., Surmann, H., Lingemann, K., Hertzberg, J., and Thrun, S. (2004). 6d slam with an application in autonomous mine mapping. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 2, pages 1998–2003 Vol.2.

Olivares-Mendez, M. A., Mellado, I., Campoy, P., Mondragon, I., and Martinez, C. (2011). A visual agv-urban car using fuzzy control. In *Automation, Robotics and Applications (ICARA), 2011 5th International Conference on*, pages 145–150. IEEE.

Pfaff, P., Kümmerle, R., Joho, D., Stachniss, C., Triebel, R., and Burgard, W. (2007). Navigation in combined outdoor and indoor environments using multi-level surface maps. *WS on Safe Navigation in Open and Dynamic Environments, IROS*, 7.

Reinke, C. and Beinschob, P. (2013). Strategies for contour-based self-localization in large-scale modern warehouses. In *Intelligent Computer Communication and Processing (ICCP), 2013 IEEE International Conference on*, pages 223–227. IEEE.

Sabattini, L., Digani, V., Secchi, C., Cotena, G., Ronzoni, D., Foppoli, M., and Oleari, F. (2013). Technological roadmap to boost the introduction of agvs in industrial applications. In *Intelligent Computer Communication and Processing (ICCP), 2013 IEEE International Conference on*, pages 203–208. IEEE.

Stoyanov, T., Saarinen, J., Andreasson, H., and Lilienthal, A. J. (2013). Normal distributions transform occupancy map fusion: Simultaneous mapping and tracking in large scale dynamic environments. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 4702–4708. IEEE.

Taghaboni, F. and Tanchoco, J. (1988). A lisp-based controller for free-ranging automated guided vehicle systems. *The International Journal Of Production Research*, 26(2):173–188.

Thrun, S., Burgard, W., Fox, D., et al. (2005). *Probabilistic robotics*, volume 1. MIT press Cambridge.

Thrun, S., Fox, D., Burgard, W., and Dellaert, F. (2001). Robust monte carlo localization for mobile robots. *Artificial Intelligence*, 128(1):99 – 141.

Tipaldi, G. D., Meyer-Delius, D., and Burgard, W. (2013). Lifelong localization in changing environments. *International Journal of Robotics Research*, 32(14):1662–1678.

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

Underwood, J. P., Jagbrant, G., Nieto, J. I., and Sukkarieh, S. (2015). Lidar-based tree recognition and platform localization in orchards. *Journal of Field Robotics*, 32(8):1056–1074.

Zhang, J. and Singh, S. (2014). Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems Conference (RSS)*.

Zhuang, Y., Jiang, N., Hu, H., and Yan, F. (2013). 3-d-laser-based scene measurement and place recognition for mobile robots in dynamic indoor environments. *Instrumentation and Measurement, IEEE Transactions on*, 62(2):438–450.