The VIKINGS Autonomous Inspection Robot: Competing in the ARGOS Challenge

Merriaux Pierre, Romain Rossi, Rémi Boutteau, Vincent Vauchey, Lei Qin, Pailin Chanuc, Florent Rigaud, Florent Roger, Decoux Benoit, Xavier Savatier

VIKINGS autonomous inspection robot for the ARGOS challenge

Pierre Merriaux[1], Romain Rossi[1], Rémi Boutteau[1], Vincent Vauchey[1], Lei Qin[1], Pailin Chanuc[1], Florent Rigaud[1], Florent Roger[2], Benoit Decoux[1], Xavier Savatier[1]

***Abstract*— This paper presents the overall architecture of the VIKINGS robot, one of the five contenders in the ARGOS challenge and winner of two competitions. The VIKINGS robot is an autonomous or remote-operated robot for the inspection of oil and gas sites and is able to assess various petrochemical risks based on embedded sensors and processing. As described in this article, our robot is able to autonomously monitor all the elements of a petrochemical process on a multi-storey oil platform (reading gauges, state of the valves, proper functioning of the pumps) while facing many hazards (leaks, obstacles or holes in its path). The aim of this article is to present the major components of our robot's architecture and the algorithms we developed for certain functions (localization, gauge reading, etc). We also present the methodology that we adopted and that allowed us to succeed in this challenge.**

I. Introduction

Over the last decades, robotics in industrial environments has been confined to assembly lines to increase productivity and decrease costs. In recent years, traceability and flexibility have become major issues for our industry. In order to meet these needs, the connectivity of all the assembled parts (industry 4.0) and mobile robotics are the key technologies [1] [2].

More precisely, regarding mobile robotics in an industrial environment, most of the works focus on AGVs (Automated Guided Vehicles). These works are applied to warehouses [3], sometimes in more complex environments such as dairies [4], but rarely in unstructured environments that combine the difficulties of indoor and outdoor industries such as oil and gas sites. In the specific field of oil and gas, [5], [6] and [7] show us that there are basically three kinds of robots:

- pipe or pipeline inspection robots: with a maximum of one or two degrees of freedom, it is difficult to classify them as mobile robotics,
- remotely operated underwater vehicles (ROVs), not autonomous,
- flying drones to inspect the exteriors of large-scale structures, without real interaction with their environment.

After some evaluations of commercial products in situ, TOTAL came to the conclusion that nothing met their expectations: an offshore platform inspection robot with a sufficient level of autonomy to help an operator in case of a serious problem on the premises.

[1]Pierre Merriaux, Romain Rossi, Rémi Boutteau, Vincent Vauchey, Lei Qin, Pailin Chanuc, Florent Rigaud, Florent Roger, Benoit Decoux, Xavier Savatier are with Normandie Univ, UNIROUEN, ESIGELEC, IRSEEM, 76000 Rouen, France

[2]Florent Roger is with Sominex, Z.I 13, Rue de la Résistance, 14406 Bayeux

TOTAL, with the help of ANR (Agence Nationale de la Recherche), consequently launched in 2014 the ARGOS challenge (Autonomous Robot for Gas and Oil Sites) [8] to test the most advanced mobile robotics solutions in operational conditions. The main goal of this challenge is to foster the development of autonomous inspection robots to increase the safety of oil and gas production sites. A video presentation of the ARGOS challenge can be found at the link below.[1]

From June 2015 to March 2017, three competitions were organized in a decommissioned gas dehydration unit in Lacq (Southwest France), the UMAD site. Each competition had an increasingly difficult level with more realistic and complex test scenarios. During these competitions, the robots were evaluated on various missions : navigating in complete autonomy on multiple levels with stair negotiation in between, reading and checking values of pressure gauges, checking the state of valves, making thermal measurements on pipes.

In addition to autonomous monitoring of the factory process, the robots have to handle anomalies (gas or oil leak detection, unexpected heat sources, general platform alarms, cavitation noise detection in pumps), harsh environments (heavy rain, negative temperatures, mist, direct sunlight) and be safe for human co-workers, factory equipment and itself (obstacle detection and negotiation, negative obstacles, tolerance to wifi loss, safe behavior in all conditions). Indeed, the robot must remain operational and useful even in case of emergency or major incident and thus be able to operate in degraded environments or with damaged parts.

Another major requirement was the ATEX certificability of the system which is mandatory for this industrial application. This requirement implies strong impacts on the mechanical and electric design of the robot.

This paper presents VIKINGS, the robot developed for the ARGOS challenge by IRSEEM and SOMINEX. VIKINGS won the two first competitions and took second place in the final ranking.

II. Robot description and architecture

The VIKINGS project aims to propose an innovative and agile robotic platform for autonomous monitoring of oil and gas sites. It was developed by a French consortium: IRSEEM as lead partner and responsible for system design and software development; and SOMINEX, responsible for the mechanical design and manufacturing.

[1]`https://www.youtube.com/watch?v=kdx-DFI1VuA`

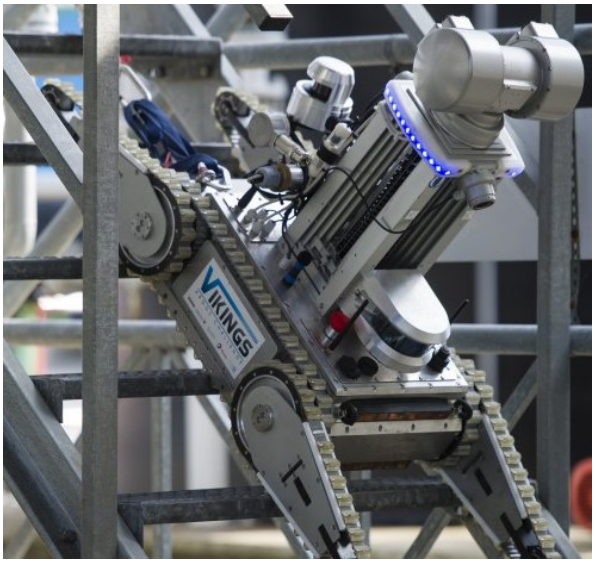


Fig. 1: VIKINGS climbing stairs in autonomous driving mode at the competition site during the last competition (March 2017).

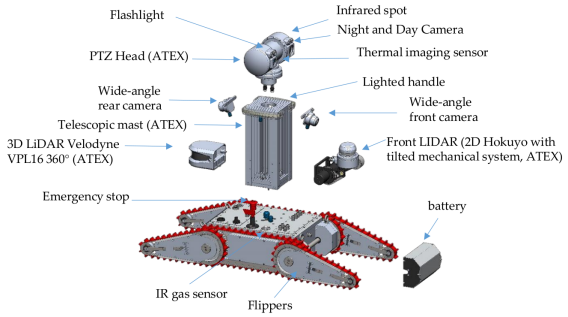


Fig. 2: Overview of the VIKINGS subsystems.

The system is built on a mobile base with two articulated caterpillar tracks and equipped with a telescopic mast enabling the robot to perform routine surveillance rounds in operational conditions. Fig. 1 shows the robot climbing a staircase during the last competition. The blue light on top of the mast indicates the autonomous driving mode. The tracks are fully extended to increase traction during this maneuver.

A. Mechanical design

The mobility is based on a differential drive system with tracks and mobile flippers. The mobile base has a width of 43 cm and a length of 60 cm, which is compact and allows easy maneuvering in narrow corridors. When the flippers are fully extended, the track length in contact with the floors goes up to 101cm which gives stability for stair-climbing. The mobile flippers are also used to overcome steps and other obstacles up to a height of 25 cm. The mobile base and other sub-systems are detailed in Fig. 2.

To perform measurements on difficult-to-reach areas, a telescopic mast was designed with a sensor head on the top. This telescopic mast makes it possible to move the head from a low position 70 cm above the ground, to a high position up to 180 cm above the ground.

The two main optical sensors are embedded in the sensor head: a Sony color camera equipped with a motorized optical zoom; and a Flir thermal camera. Both are used to assess the status of various systems in the factory: read pressure gauges, valves positions and measure pipe temperatures. The head is motorized along the pan and tilt axes with two Dynamixel servomotors controlled by the embedded computer.

B. Sensors

To fulfill its missions, the robot is equipped with many sensors listed below:

- Velodyne VLP16 LIDAR,
- Hokuyo UTM-30LX LIDAR,
- SBG Ellipse inertial measurement unit,
- Sony FCB-EH3410 color camera with motorized zoom,
- Flir A35 thermal camera,
- Two IDS UI-1241LE-C-HQ usb camera with fish-eye lenses,
- AXIS T83 microphone,
- Two Dodotronic ULTRAMIC384k ultra-sound sensors,
- Four TMP107 internal temperature sensors,
- Euro-gas infrared methane sensor,
- Encoders on tracks, flippers, mast and nodding LIDAR,
- Custom battery current and voltage sensor.

The VLP16 LIDAR is placed at the rear of the Vikings and is used for localization. The 16 LIDAR layers with high precision and range make this sensor well suited for this application. However its minimum sensing range of 70 cm makes it useless for obstacle detection near the robot.

A custom-made ”nodding LIDAR” sensor is placed at the front of the robot and is mainly used for obstacle detection (positives and negatives) in the main direction of motion. This sensor uses a single-layer Hokuyo UTM-30LX LIDAR, animated with a nodding motion by a small DC motor. An incremental encoder on the axis of rotation measures the vertical angle of the LIDAR. This measurement is taken into account by the software to build a 3D point cloud. Custom electronics takes care of DC motor regulation and synchronization between the LIDAR and the angle sensor. The Hokuyo LIDAR has a horizontal field of view of 270° and the nodding motion gives a vertical field of view of 70°. This nodding LIDAR provides a complete scan (bottom to top) at a frequency of 2Hz.

A pair of fish-eye cameras (front and rear-facing) is attached to the fixed part of the telescopic mast and allows a wide-angle view of the surrounding area, mainly for operator assistance during remote operations.

Two ultrasonic sensors are installed to detect and locate various noises, such as compressed air or gas leaks. A standard microphone is installed to detect the General Platform Alarm (GPA) sound. It is also used to detect malfunctions of electric pumps by performing sound analysis.

Other sensors are installed in the robot’s main body: temperature probes to monitor various systems, an inertial measurement unit used by the localization and incremental

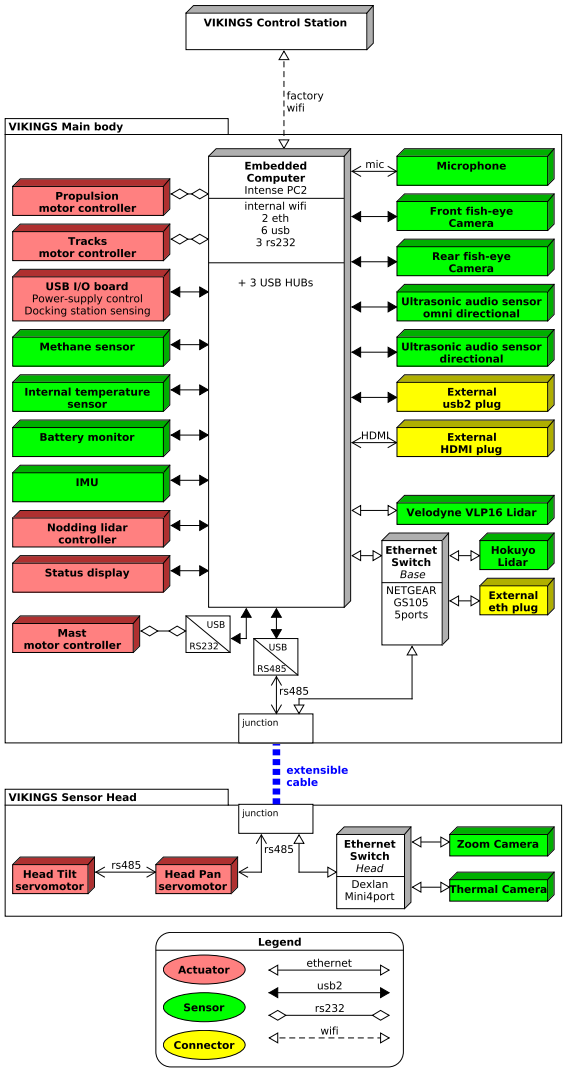


Fig. 3: System architecture.

encoders on actuators (motors, flippers, mast ...) for closed-loop control. A dedicated sensor measures the battery current and voltage to estimate the remaining capacity and monitor power usage.

C. System architecture

To facilitate programming, construction and maintenance, we designed a simple and straightforward control architecture where the embedded computer is directly connected to every sensor and actuator. This architecture is presented in Fig. 3. The embedded computer, with its integrated wireless network card, provides remote connection to the Control Station through the factory network.

To power all the electronics and actuators, a custom-made Lithium-Ion (48.1V, 168Ah) battery pack is used. This battery is located at the front of the mobile base and can be easily replaced. The Battery Management System is integrated into the robot to protect the battery against over-loading, over-current and under-voltage conditions. A special

Fig. 4: Overview of the HMI. The HMI is dynamic according to the events encountered and the requests of the user. The operator can ask it to display on the central area: direct camera views, localization map, 3D map, path-planning, thermal image, etc.

docking-station is also used for autonomous charging of the battery.

A custom power-supply board was made to power every internal part with the required voltage and protection. This power-supply can receive commands from the main computer to selectively enable each output. Power-hungry sensors and actuators can then be disabled when required to save energy.

The *Control Station* is a standard laptop computer with two USB joysticks allowing complete control of the robot. One joystick is used to control the pan-tilt head, camera zoom and height of the mast. The other is a game-pad with two mini-sticks and is used for locomotion control, flipper control and selecting the driving mode. On the control station HMI, as shown in Fig. 4, the data from all the robot's sensors are displayed to provide full situation awareness.

D. Software architecture

The software developed for the VIKINGS robot and its control station is built on the component-based middle-ware RTMaps[2]. This framework was selected because of its high efficiency and ease of programming. As each component is a compiled dynamic library written in C++, there is very little overhead. Moreover, the data exchange between components relies on a shared memory approach, which is very efficient.

E. Data architecture

The robot uses various files to describe its environment and the tasks to perform. These files are created for a specific site (a whole factory or off-shore platform); stored in the internal hard-drive of the robot and automatically loaded when needed. Here is a description of the main data files:

- Localization map,
- Routemap,
- Static map,
- Mission list.

The *Localization map* is automatically generated from a 3D scanning of the environment. The 3D scanning can

[2]https://intempora.com/products/rtmaps.html

be performed with high-resolution survey LIDAR sensor, or by manually controlling the robot. This map contains a likelihood field and is used for position and orientation estimation. The localization function is described in section III-B with additional information about the data structure.

The *Routemap* contains a set of 3D positions; a directed graph structure to link these positions and the checkpoint data recorded in a single XML document. This is used to define the interesting positions: predefined pathways, corridor intersections, gauges, valves and other checkpoint positions and data (expected pressure for a gauge, maximum temperature for a thermal measurement and so on). This file is created manually using an XML editor. Semi-automatic creation of this file is possible but has not been developed.

The *Static map* is a 2D map of each level of the environment and contains allowed and forbidden areas. The robot uses this map to check whether it can go to a specific position, both in tele-operated and autonomous driving modes. These maps are created automatically from the 3D scanning of the environment and the operator can manually add forbidden areas.

The *Mission List* contains multiple definitions of a robot's mission. Each mission is a set of measurements to perform and a definition of the various autonomous behaviors in case of a measurement error or anomaly detection (loss of wifi connection, General Platform Alarm, gas leak detection and so on). These behaviors are described using a Domain-Specific Language built on top of the LUA scripting language. This file is created manually and cannot be easily automatized as it contains know-how about the gas and oil site monitoring procedures. On the other hand, this file is very short thanks to the DSL: a typical mission is described in about 20 to 30 lines of code.

Although mostly manual, the creation of the data files to deploy the robot on a new site is actually quite fast. We had the opportunity to display the robot on various exposition booths and a typical setup can be performed in a couple of hours for a site of about $100m^2$ and a dozen checkpoints. Semi-automatic software tools can be developed to ease the file creation procedure and allow wider adoption of the robot. Moreover, these data files only need to be updated when a major modification of the environment is performed, as the algorithms are robust enough to handle small modifications.

III. Robotic functions

VIKINGS is a complex robot, able to operate in various environments and situations. Here is a description of the generic functions developed for the ARGOS challenge.

A. Operating modes

The VIKINGS robot is able to operate in various driving modes depending on the task to perform. Here is a description of each driving mode.

Autonomous
: In this mode, the robot operates in complete autonomy. A mission plan is specified by the operator, with a sequence of measurements to perform and reactions to apply in case of incorrect readings or emergency. After the operator instruction to start the mission, the robot is fully autonomous and navigates to perform the requested gauge and valve readings. In case of default (incorrect reading on a gauge/valve) or emergency (heat source detected, gas leak, wifi loss ...) the robot performs autonomous actions. At all times, the remote operator is able to modify the robot's behavior and mission plan.

Rail mode
: In this mode, the robot is basically performing autonomously, but the forward speed is manually controlled by the remote operator. This allows a very intuitive use of the robot for manual inspections. Every tedious and difficult maneuver (climbing steps or stairs, driving around obstacles, finding a path to the next objective) is performed automatically.

Manual mode
: This mode is intended for manual inspection and maneuvers, with all the safety features still enabled. This allows direct control of the robot, but prevents all dangerous situations (collisions with structures or people, falling down stairs). Complex maneuvers can be performed safely in this mode, as the operator has direct control of all actuators. However, this mode is challenging to use and needs a trained operator.

Unsafe Manual mode
: This last mode also gives full and direct control of each robot actuator, with all safety features disabled. This mode is mainly used to operate outside of the known environment in case of maintenance of the robot, or for interventions in a degraded environment.

The transition between operating modes is designed to be as transparent as possible. During a routine inspection in autonomous mode, the operator can take manual control, move the robot to another position, make manual measurements and resume the original mission plan seamlessly. The robot autonomously performs path-finding when needed and keeps track of visited checkpoints in the ongoing mission. A mission ends when there are no more checkpoints to control. In this case, the robot goes back autonomously to the docking station and reloads its batteries to be ready for the next mission.

B. Localization

Robot localization is a key function in autonomous robot deployment. The current robot's position is a mandatory input data for path-planning and trajectory control. Consequently, the robot should be able to precisely find its location despite the environment complexity. Navigating on an offshore platform requires climbing stairs, so the motion is not always on a flat surface. As a result, the current

position state of the robot must be estimated with 6 degrees-of-freedom.

Offshore platforms have different layouts from other industrial sites found in the literature. In fact, studied storage warehouses or factories often have squarish shapes with machines, racks or shelves and large gangways. A look at the facility blueprint would reveal a well-organized structure. Offshore platform environments are different. As shown in Fig. 11a, the area is crowded with equipment. Gangways are narrow. Most of the time such facility has several floors connected by staircases. The floor itself is made of gratings, which makes it irregular. Fig. 11b shows the full-scale test environment.

Most of the existing works tackling robot localization in industrial facilities focus on indoor, simple and planar environments. There are numerous approaches to solving localization in such environments. Magnetic or optic lines [9], [10] can be placed on the ground and the robot follows these lines. However, the robot motion is constrained by the line and it cannot avoid an obstacle on its trajectory. In order to allow more freedom, beacon-based localization can be used.

Such approaches would be very costly in oil and gas facilities because at all times, multiple beacons have to remain in direct line of sight to allow the robot to estimate its position. Consequently, a very large number of beacons has to be installed in the site. As mentioned by [11], localization based on the existing environment would expand autonomous robot applications to a larger set of situations, and would be useful in oil and gas sites.

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 meters (m)
ψ, θ, φ	: Orientation in degrees ($^\circ$)

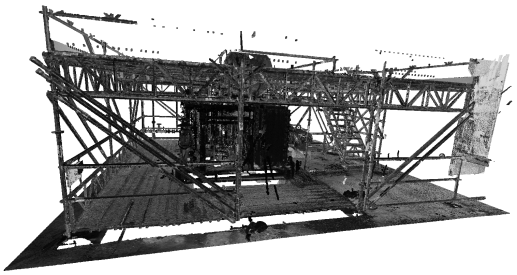
Localization methods aim to find the most likely state vector among several hypotheses. The likelihood function expresses the probability $\mathbb{P}(\mathbf{Z}|\mathbf{X}, \mathcal{M})$ of obtaining a measurement $\mathbf{Z}$, given a state $\mathbf{X}$ and our prior knowledge of the environment $\mathcal{M}$.

The likelihood function uses the 3D likelihood methodology presented in Fig. 5. This method offers two main advantages:

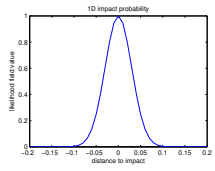
- As the LIDAR impact probabilities are pre-processed in the map, the run-time computing cost is limited, enabling real-time applications.
- As detailed in section III-C, it is possible to detect LIDAR impacts not belonging to the map with limited overhead for hole detection.

However, one drawback is the memory size of the likelihood field. To overcome this, a hybrid data structure was developed: an octree with regular 3D arrays as leaves. The octree structure allows very efficient storage of empty spaces, while the regular 3D arrays on the leaves makes possible to efficiently store the likelihood values. As the likelihood values are stored on 8bits, smaller than pointers (64bits), this hybrid structure allows us to find an optimal balance between the density of the likelihood field and the number of pointers used to store the octree. Fig. 6 shows the storage performances of this hybrid data structure.

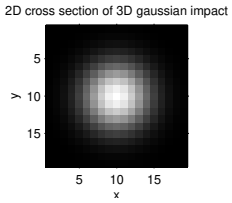
The localization performances were evaluated in our laboratory presented in section V-A.2 using our Vicon motion



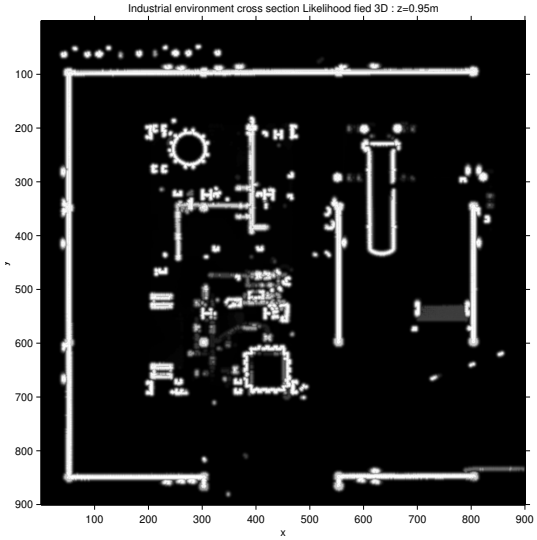
(a) 3D point cloud from a complex industrial site mapping. Point cloud data ©Total S.A.



(b) 1D likelihood LIDAR impact



(c) 2D cross section of 3D Gaussian local likelihood LIDAR impact



(d) Cross section of the resulting likelihood field

Fig. 5: 3D Likelihood field map construction : from environment mapping point cloud Fig. 5a, we use the LIDAR variance Fig. 5b, to compute a local 3D likelihood field Fig. 5c, and merge all of them in a global likelihood field Fig. 5d.

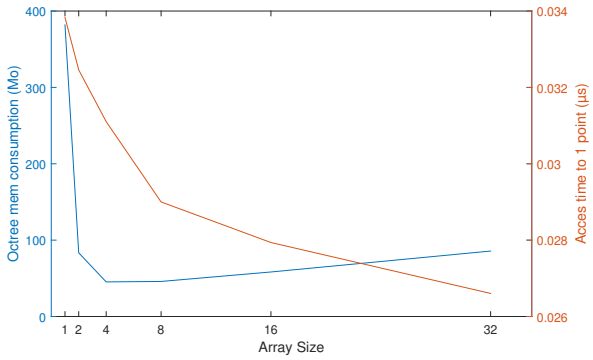


Fig. 6: Memory consumption and processing time of the hybrid octree for 19.2 million points with respect to the 3D occupancy grid size. The best performance is achieved for occupancy grids of 4x4 or 8x8 elements. A regular grid map requires 248MB and results in a point access time of 13.3 microseconds. The performance is measured on one core of an *Intel i7-4700MQ @ 2.4GHz* CPU.

capture system as ground truth [12]. For a complex 3D trajectory, with obstacle-crossing and staircase-climbing, the RMSE positioning accuracy is 2.6cm and 0.29° as detailed in Table I. A more complete description and evaluation of this localization method was published in [13].

TABLE I: Real experimentation results in the 350m^3 IRSEEM autonomous navigation laboratory, 500 particles on a 22.5m 3D trajectory.

	mean (std)	**RMSE**
Position error(m)	0.0236 (0.0113)	0.0261
Orientation error (°)	0.292 (0.162)	0.334

C. Obstacle detection

Obstacle detection is another critical function for a mobile robot evolving in complex and manned environments such as an oil platform. The VIKINGS robot is equipped with 3 detection systems based on a 3D point cloud. This 3D point cloud is built in real-time with the data from the two embedded LIDARs, merged in a common framework (see Fig. 2).

The 3 obstacle detection systems are:

- Positive obstacle detection : using the likelihood field, once the current location of the robot has been processed, it is straightforward to define a threshold to classify LIDAR impacts as belonging to the known map or not. The LIDAR points lying outside of the known map are then clustered using the DBSCAN algorithm [14].
- Negative obstacles (holes) : this algorithm counts the number of LIDAR points in the area in front of the robot, in the direction of travel. The density of the LIDAR points is used to assess the presence of negative obstacles.
- Human detection : As a safety feature, the robot is asked to pause its current actions whenever a human is in the vicinity (radius of 1.5m). To detect this condition, a specific algorithm searching for vertically-aligned circles representing the legs and torso was used. A RANSAC and a temporal filter were implemented to avoid false-positives. The circle detection method in LIDAR point clouds is explained in more detail in [15].

These 3 types of obstacles are detected and stored in a local 2D map used by the path-planning function as input data.

D. Path planning and path control

For safety reasons, the robot is required to use designated areas on pathways. To handle this, a two-layer path-planning algorithm was designed.

The first layer, the *global path planning* uses a connected graph representing the authorized parts of the walkways. This graph is created with human supervision for the site, as part of the initial setup and specifically indicates the path the robot should use. In this graph are also included interest points for measurements and specific actions such as climbing stairs. A Dijkstra algorithm is used in this graph to quickly process the shortest path between the current position of the robot and the destination. The nearest point in the graph is taken as the entry point in the graph and only allowed connections between pathways are followed. The global path planning ensures the robot only uses the designated areas in normal conditions.

For abnormal situations such as blocked pathways or obstacles, a second path-planning algorithm is used. This *local path-planning* algorithm dynamically creates a path between the current robot position and the nearest point in the global graph. This path may go outside of designated areas on pathways for short distances. The path is processed using an A* algorithm in a dynamically created graph and uses an occupation grid to handle nearby obstacles.

When an obstacle in the trajectory of the robot is encountered, the robot stops and an overcoming strategy is evaluated among three options :

1) ”New path” : completely avoid the blocked pathway by finding another path to the destination,
2) ”Drive around” : get around the obstacle and continue the current path,
3) ”Cross over” : runs over the obstacle using the flippers and continue the current path.

The ”Drive around” option is usually possible, except in narrow pathways, and is the default option. If the obstacle is big enough to prevent the robot from driving around it, the ”New path” option is executed. For safety reasons, the ”Cross over” option is only available on operator approval.

IV. Specific functions for Oil and gas environments

Specific functions developed specifically for oil and gas environments are described in this section.

A. Reading pressure gauges and valves

One of the main functions of the robot is autonomous monitoring of the factory chemical process, by automatic readings of the installed sensors: pressure manometers and other gauges or fluid level indicators. Checking of the orientation of manual valves and detecting a missing or modified sensor helps to increase safety. We used image processing and computer vision techniques to implements these functions.

For each checkpoint, the sensor (gauge or valve) to be read is known; we consequently have a reference image of it. The general procedure for processing a checkpoint is as follows. Firstly, the robot goes to the registered reference position for this checkpoint and takes a picture of the scene using its PTZ camera. We denote this picture as the test image. Secondly, we detect and localize the sensor in the test image by matching the image feature descriptors extracted from the reference image with the ones from the test image. In this way, the absence of a sensor can be detected when the number of matches falls below a predefined threshold. Thirdly, we transform the detected local region of interest (ROI) to align it with the reference image. Finally, sensor-specific processing is performed in the transformed ROI to read the value of the gauge or to determine the state of the valve.

To increase robustness, we register multiple reference positions in the map for each checkpoint. If the image analysis module is not able to find the sensor for a given checkpoint at the first position, the robot will try the other reference positions previously registered until the sensor is found. Eventually, if the sensor is not seen in any of these positions, it is declared missing and the appropriate actions will be taken (operator warning and reporting).

1) Pressure gauges: To read the value of a gauge from its image taken by the PTZ camera, some a priori information is required:

- A reference image of the gauge taken by a standard camera from the front view,
- The geometric relation (manually annotated) between the angle of the gauge's needle and its corresponding value in the reference image. Specifically, we annotate the minimum and the maximum values of the gauge and their corresponding needle angles in the reference image. As the reference image is carefully taken from the front view, it is reasonable to assume a linear relation between the needle angle and its corresponding gauge value. In this manner, reading the gauge value is reduced to measuring the needle angle in the image.

Our method for reading the gauge dial in a checkpoint can be summarized as follows:

1) Place the robot to access the checkpoint,
2) Control the PTZ camera so that its point of view is oriented towards the gauge,
3) Perform image acquisition using the camera,
4) Detect the gauge in the test image by the Scale-Invariant Feature Transform (SIFT) [16] keypoint detection and descriptor matching,
5) Perform image registration using homography between the reference image and the detected ROI,
6) Extract the needle of the gauge in the transformed ROI with a Hough-transform based line detection [17] with geometric verifications,
7) Compute the angle of the needle by calculating the slope of the detected line,
8) Read the value of the gauge using the angle and the geometric relation annotated previously.

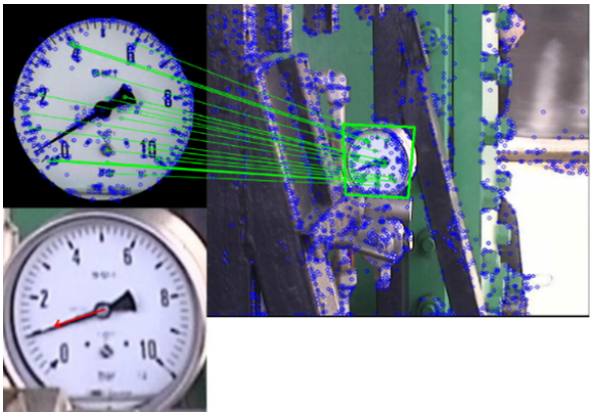


Fig. 7: Example of detection of position of the needle of a sensor dial.

At the end of the process, a confidence level of the result is calculated. If this confidence level is not high enough, the process is repeated again from step 3, with a change in camera zoom. Fig. 7 shows an example of a correct reading of a manometer value.

It may be worth noting some best practices that we found useful for increasing the robustness of the gauge reading.

- In step 4, when the gauge is small in the test image, the keypoint-based method may fail to detect it. As the dial of the gauge is a circle, it becomes an ellipse in the test image under the camera projection. Therefore, we also considered ellipse detection methods, e.g. [18], to complement the detection of the gauge.
- In step 6, we need to binarize the transformed ROI image before applying the Hough transform to detect lines. Instead of using a fixed threshold for binarization, we used adaptive thresholding methods, which have been observed to greatly robustify the resulting binary image against noise and irregular lighting.
- In step 6, geometric verifications are helpful for rejecting false positives detections of the needle. In our implementation, we considered several geometric constraints, namely the ratio of the length of the line with respect to the size of the reference image, the distance from

Fig. 8: Examples of correct needle detection in manometers in non-optimal conditions.

the image center to the line, the aperture angle of the two ends of the line segment with respect to the image center, etc. The needle is declared to be "detected" only when all these geometric constraints are satisfied within certain predefined thresholds.

All these processes allows us to improve the robustness of the detection against disturbances such as direct sunlight and dirt on the instruments to be read. Fig. 8 shows examples of correct needle detections in manometers in non-optimal conditions.

2) Valve positions: For reading the valve positions, we developed two methods by image analysis using different features. The first one is based on the color information of a given valve and was tested during the first competition. The second method is based on machine learning with no assumption of color and was implemented for the second competition.

Color-based method. The color-based detection of the state of a valve (open or closed) consists of two steps:

- Localization of the valve in the image taken by the camera,
- Determination of the valve's state by comparing the test image with the reference images.

The first step is based on the detection of keypoints in the images and the calculation of descriptors at the location of these keypoints. This step allows a global localization of the valve-object (not only the valve but the whole block which is associated with it). The second step consists of processing based on color, to precisely localize the valve itself. This process is first applied to 2 reference images: one of the valve in the open state and one in the closed state. Fig. 9 shows one example of correct detection using the color method.

Machine learning-based method. This second method is based on a learning step to help the machine to decide if the valve is open or closed. It is applied on grayscale images of the valves. The training set is based on the image collection we gathered during the first competition. Specifically, we extracted the Histogram Of Oriented Gradients (HOG) [19] feature and train a Support Vector Machine (SVM) [20] classifier for classification.

Arbitration. Both the color method and the machine learning method generate their own results with corresponding confidence levels. The results of the two methods can be different. To fuse the results, we employ a simple arbitration algorithm to determine the final output based on the results' confidence levels. The whole process for determining the

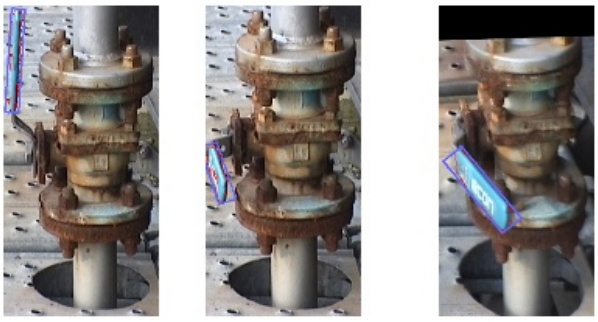


Fig. 9: Examples of detection of the position of a valve (left: reference image for open position; middle: reference image for closed position; right: result of processing on one image of the database, showing the detection of the valve)

position of a valve is as follows:

1) Perform image acquisition by the PTZ camera.
2) Detect the valve by keypoint matching using SIFT descriptors.
3) If the detection is successful, use the detected ROI for the subsequent processing; otherwise use the whole image.
4) Process the test image from step 3 using the color method; get a first result with its confidence level.
5) Process the test image from step 3 using the Machine learning-based method; get a second result with its confidence level.
6) Fuse the results of the two methods:
   - If the two results agree, use the result; the final confidence level is the greater of the two methods with a small increase of the confidence level;
   - Otherwise use the result with the greater confidence level; set the final confidence level to the greater one with a small decrease of the confidence level.

3) Detecting and measuring sensor displacement: The position displacement of a sensor, e.g. a pressure gauge, can be detected by comparing the actual image of the scene with its reference image. The idea is to compensate the global camera motion and then to detect and measure the local gauge movement. Global motion is represented by a perspective transformation matrix, which is computed by point correspondences in the two images. Point correspondences are achieved by optical flow if the motion is not intense or by keypoints detection and matching in general cases. After obtaining the transform matrix, we then transform the test image to align it with the reference image. The local gauge movement is then detected by a subtraction operation between the reference image and the transformed test image. Next, the pixel displacements of the gauge is computed from the subtracted images. Computation of the displacement distance is obtained as follows:

- Calibrate the camera to process the focal length f,
- Estimate the pixel displacement of the gauge in the

subtracted images p,
- Estimate the camera-to-gauge distance L,
- Process the displacement d in 3D world by $d = p \times L / f$.

4) Validation process: To evaluate the performance of the algorithms for reading valves and gauges, an automatic test tool was set up using Jenkins. Jenkins is an open source automation server that helps to automate the software development process with continuous integration. Each time a new build is uploaded, Jenkins runs automated tests on an image database to check if new evolutions actually improve the reading results and do not cause a crash on the whole database (robustness test). The database is composed of 2910 images of the different checkpoints, which can be split into two sets: gauges and valves. The valve set is made up of 991 images containing a valve, annotated with its real state (open or closed) whereas the gauge set contains 1919 images of manometers, annotated with the real value indicated by the needle. The images of this database were taken at all times of day and in different seasons (we enriched the database at each competition), which allowed us to test our algorithms in very different conditions. In addition, during the challenge, the gauges were deliberately degraded by the jury: water drops, dirt, pen marks, etc.

For quantitative assessment, we defined a threshold value to determine whether to take the result of the reading process into account or not: a minimal confidence level of 50%. Moreover, we defined a second threshold: a maximal error rate of 2% between the real value and the measured value. If the confidence level of the reading process is below the minimal confidence, the measured value is declared as "uncertain". Otherwise, the measured value is compared to the real value: if the absolute difference is greater than the maximal error, the result is declared as "false". Otherwise, it is declared as "true".

For the pressure gauges, with these evaluation criteria, we get 25.8% of uncertain readings and 97.4% of true results on the certain readings. For the valves, with the same evaluation criteria we get 19.4% of uncertain readings but 83.5% of true results on the certain readings.

B. Temperature detection and measurement

For temperature detection and measurement, we integrated a FLIR A35 camera. The sensor is mounted at the top of the mast to allow a 360° measurement around the robot. Using a geometric calibration of the thermal imaging sensor, any part of the structure can be targeted in the same manner as other sensors. Knowing the attitude of the PTZ head and using geometrical calibration of the thermal imaging sensor, the 3D position of hot spots can be evaluated in the same framework as robot navigation and obstacle detection.

C. Sound processing and detection

The robot has to deal with three kinds of sound processing: General Platform Alarm detection, pump analysis, and gas leak detection.

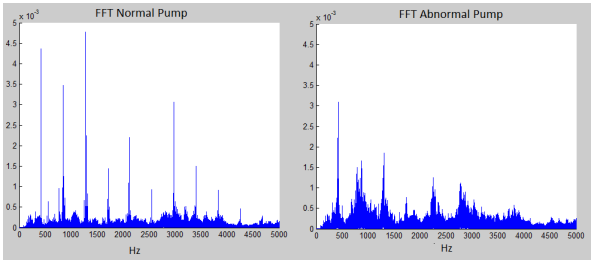


Fig. 10: Sound frequency spectrum for a normal (left) and an abnormal (right) pump.

1) General Platform Alarm (GPA) detection: The robot has to detect the GPA, which is a general alarm calling all the staff to go to their allocated muster stations. The GPA is a standardized intermittent signal of constant frequency. This sound measurement is done using the Axis microphone and the processing is based on synchronous detection using a pure sine wave as a model of the GPA sound.

2) Pump analysis: Pumps have to be monitored to determine if their operation is normal or abnormal. The robot goes close to the pump and an audio recording and analysis is performed. This sound measurement is carried out using the Axis microphone.

To set up our algorithm, we built a dataset by mixing the two samples provided (normal and abnormal pumps) with various industrial noises. Harmonic separation consists of maximizing the total signal power while maintaining a fundamental frequency within a certain tolerance. Fig. 10 shows an example of normal and abnormal pump sounds. A harmonic descriptor was built taking into account the total power as well as the signal-to-noise ratio in a frequency band around each harmonic. This descriptor is linearly separable according to 3 classes: pump not present, normal and abnormal behavior. The resulting algorithm is able to detect the presence or absence of a pump and to verify its proper functioning. The measurement is accompanied by a confidence level.

3) Gas leak sound detection: The robot is embedded with two ultrasonic microphones: an omni-directional one for leak detection and a directional one (thanks to an acoustic cone) for leak localization. The robot is continuously listening to ambient ultrasonic sound and performing sound leak detection. As soon as a leak is detected, the robot is rotated to find the leak origin using the directional microphone. Ultrasonic sensors are sampled at 384 kHz. The method consists in measuring the spectral power between 25 and 100 kHz after mapping and subtracting the ambient ultrasound signal.

D. Autonomous reactions

During operation, some events trigger a specific reaction of the robot. Here is the list of events which are continuously monitored and the corresponding default actions:

- Gas leak detection $\Rightarrow$ locate the leak,
- Heat source detection $\Rightarrow$ locate the source,
- Battery low $\Rightarrow$ go to the docking station,

- General Platform Alarm ⇒ go to the safe area,
- Wifi loss ⇒ go to the safe area if connection not restored after a given time.

The operator is continuously informed about such events and can manually change the reaction. The default reaction depends on the current driving mode. In manual mode, the operator receives a warning and a proposal to perform the action autonomously. The operator needs to select the reaction for the robot to switch back to autonomous mode and react. In autonomous or rail mode, the operator receives a warning and a proposal to abort the autonomous reaction. After a short time lapse, the robot reacts accordingly. In the Unsafe Manual mode, the operator receives a warning but no action can be performed autonomously since the robot is potentially in a dangerous position.

E. ATEX

The safety of our platform is based on the Ex-d protection mode, i.e. an extremely robust envelope contains the explosion inside the device. A theoretical analysis of the platform was carried out with a simulation of an explosion resistance with a pressure of 15 bars. All components requiring intrinsically safe signals (mics, LEDs, radio antennas) are powered by specifically designed circuits, developed in compliance with IEC 60079-11.

The PTZ head is also protected by the Ex-d protection mode. All moving parts (mast, PTZ head, caterpillar) are considered safe due to their very slow motion : a movement below 1m/s is considered safe in EN 134363-1.

Optical sensors (cameras, LIDARs) are installed in Ex-d enclosures with windows made of polymer material. The main issue is the effect of the window on the LIDAR measurements. Two enclosure shapes were tested: cylindrical and prismatic. Neither gave deformation of the point cloud. However, 36% of the LIDAR points are lost with the cylindrical enclosure and 24.7% with the prismatic one.

The uniformity of the point cloud depends on the shape. With the cylindrical enclosure, the points are not evenly distributed: there is almost no point on the ground, which makes it difficult to localize the robot along the Z axis. The uniformity is not affected with the prismatic enclosure, which makes it a better choice regarding LIDAR disturbances.

We were accompanied throughout the development by an ATEX expert to guide our technological choices. The result is a certifiable platform with a design folder including: ATEX Certification Readiness Assessment, Protection Mode Checklist, Mechanical Resistance Simulation and the VIKINGS Atex Certification Assessment. These elements make it possible to request the certification by an accredited organization, which was not requested in the competition. Moreover because of the selected protection mode, the certification tests require the destruction of several robots.

V. Implementation, tests and experiments

A. Software development and methods

One of the main aspects of the ARGOS challenge is an annual evaluation of the robot's performance on a test field during the 3 consecutive years of the project. This organization required us to have a functional system during each competition. To meet this requirement, we used state-of-the-art software development methods and tools, as well as a very realistic testing area with a layout similar to the competition site.

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

(b) ARGOS challenge test site (UMAD)

Fig. 11: Environments targeted by the ARGOS challenge...

This section describes the development methods as well as the various tools we used to test the system before going to the competitions.

1) Continuous integration: The source code is based on the RTMaps middleware. The source code was maintained under revision control with Mercurial SCM, and a collaboration server based on Redmine was used to synchronize the repositories of each developer. The Jenkins automation tool was used to continuously build and test the various critical software components, such as the computer vision components described in section IV-A.4. Using this approach, we were able to catch and correct numerous defects and regressions of software components during the three years of the project.

2) Simulation: For the development of high-level functions, we developed a simplified simulator focused on the behavior of the robot. This simulator provides a simplified 2D map of the robot location and numerous displays of internal states. It is comprehensive enough to enable the simulation of complete missions. The control station software is able to connect to this simulator and to control the simulated robot. However this simulator does not include 3D and physical simulations. A simplistic simulator such as this one is a valuable tool when developing high-level functions such as human-robot interaction, mission control and autonomous behaviors.

3) Lab tests: Our main tool to test the complete system was a scaled-down version of the UMAD Lacq testing site. A realistic reproduction of the competition environment using identical gratings, manometers and valves as well as a realistic mock-up of pipes, pressure vessels and other structural parts was built as shown in Fig. 12. To be able to test all locomotion conditions, our *mini-UMAD* test site has two floors and the same staircase as the one installed in the UMAD.

Fig. 12: Mini-UMAD reproduced in our laboratory for lab tests.

TABLE II: Project statistics.

Number of software developers	8
Total Source Code Lines	148,121
Average data record rate (logs & sensors)	920Mio/min
Maximum battery autonomy	4h20
Total weight of the robot	67kg
Total distance traveled at UMAD site	25km

This structure is installed in our Autonomous Navigation Lab, a $15m \times 10m \times 5m$ room equipped with a Vicon motion capture system allowing us to accurately localize the robot and thus to verify the embedded localization algorithm performance.

VI. Conclusion

The VIKINGS robot and the engineering team participated in the 3 competitions of the ARGOS Challenge with great success. Some statistics regarding the robot and the project are presented in Table II. Some videos of the robot in action are given in the links below[3,4,5,6].

In their final report, the jury noticed that the VIKINGS mobility was the most promising of the challenge with respect to the specifications of an oil and gas site: good trade-off between dimensions and weight to guarantee stability, very fast and smooth motions, very efficient and graceful crossing motion over steps, outstanding low energy consumption.

Based on state-of-the-art sensors such as the Velodyne VLP16 LIDAR, powerful middleware (RTMaps) and the most advanced algorithms for 3D perception, scene analysis and 6 DoF localization, VIKINGS achieved a high level of performance in the autonomous missions of the ARGOS challenge. Moreover VIKINGS is probably the most cost-effective solution of the Challenge.

VIKINGS was very successful, winner of two competitions and ranked second of the whole ARGOS Challenge. Some of the functions developed for this robot are currently been improved to use in other project. The LIDAR-based localization for example is currently used in autonomous car research.

[3] https://www.youtube.com/watch?v=wie3POxmbGI
[4] https://www.youtube.com/watch?v=W5EyAL0cGLM
[5] https://www.youtube.com/watch?v=-6xLqWv2t_0
[6] https://www.youtube.com/watch?v=21lZvVej2IQ

References

[1] A. Grau, M. Indri, L. L. Bello, and T. Sauter, "Industrial robotics in factory automation: From the early stage to the internet of things," in *IECON 2017-43rd Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2017, pp. 6159–6164.

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

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

[4] T. Stoyanov, J. Saarinen, H. Andreasson, and A. J. Lilienthal, "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*. IEEE, 2013, pp. 4702–4708.

[5] A. Shukla and H. Karki, "Application of robotics in onshore oil and gas industry—a review part i," *Robotics and Autonomous Systems*, vol. 75, pp. 490–507, 2016.

[6] ——, "Application of robotics in offshore oil and gas industry—a review part ii," *Robotics and Autonomous Systems*, vol. 75, pp. 508–524, 2016.

[7] A. S. Hashim, B. Grămescu, and C. Nițu, "State of the art survey on using robots in oil and gas industry," in *International Conference of Mechatronics and Cyber-Mixmechatronics*. Springer, 2017, pp. 177–185.

[8] The argos challenge website. [Online]. Available: http://argos-challenge.com

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

[10] F. Taghaboni and J. Tanchoco, "A lisp-based controller for free-ranging automated guided vehicle systems," *The International Journal Of Production Research*, vol. 26, no. 2, pp. 173–188, 1988.

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

[12] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, and X. Savatier, "A study of vicon system positioning performance," *Sensors*, vol. 17, no. 7, p. 1591, 2017.

[13] R. B. P. V. Pierre Merriaux, Yohan Dupuis and X. Savatier, "Robust robot localization in a complex oil and gas industrial environment," *Journal of Field Robotics*, vol. 35, no. 2, pp. 213–230, 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21735

[14] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[15] P. Merriaux, R. Boutteau, P. Vasseur, and X. Savatier, "Imu/lidar based positioning of a gangway for maintenance operations on wind farms," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 4213–4219.

[16] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[17] D.H. and Ballard, "Generalizing the hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111 – 122, 1981. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0031320381900091

[18] M. Fornaciari, A. Prati, and R. Cucchiara, "A fast and effective ellipse detector for embedded vision applications," *Pattern Recogn.*, vol. 47, no. 11, pp. 3693–3708, Nov. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2014.05.012

[19] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, ser. CVPR '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 886–893. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2005.177

[20] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995. [Online]. Available: https://doi.org/10.1023/A:1022627411411