Sensor Network and Localization methods for a mobile robot

João Ferreira, ISR-IST, Alberto Vale, IPFN-IST, and Rodrigo Ventura, ISR-IST

Abstract—This paper proposes a localization system for mobile vehicles, without sensors on board, in indoor environments. The sensory part is a network composed by laser scanner sensors placed in defined points on the building. This system is useful in cases where sensor installation directly on the vehicle is not advised, due to his or his job characteristics. An optimization method was developed to place the sensors on the building. Improving the network disposition on the building, the quality of gathered information increases. Using this information, two localization methods were developed and tested, one based on extended Kalman filter, and other on particle filter. Both were studied and compared, in simulation environment, concerning their performance and reliability. It was analyzed the impact of sensors disposition on the building in the localization system, evaluating the robustness to different sensor configurations, including a possible fail of some of them. From the experimental results, the approach that suits better this application is the particle filter localization method, the resulting estimations have good precision, it is robust to sensor failure and, facing a general failure, like power failure, it manages a short time global localization. Although these conclusions are based on simulation, the method shows promising results for a future real application.

Index Terms—Localization, Autonomous Mobile Vehicles, Sensor network, Laser scanners, Data Fusion

I. INTRODUCTION

One of the biggest technological challenges in the present is the production of clean energy, with small environmental impact but still supplying the ever growing consumption. Atomic fusion has potential to be a suitable alternative, it has the capability to generate abundant energy, releasing no carbon dioxide or greenhouse gases.

International Thermonuclear Experimental Reactor (ITER) is an experimental fusion reactor, used to test this process, and prove if this kind of energy generation is practicable.

ITER operation is based on a central component, the tokamak, that generates energy using nuclear fusion forming a plasma, kept in a Vacuum Vessel (VV), heat from this plasma is used to generate electric power.

During ITER operation, maintenance actions, like inspections or component replacement, are necessary, and can only be made remotely, due to rad-hard conditions. The transportation of equipment performed by a remote controlled vehicle, the Transfer Cask System (TCS) through narrow corridors. This is a very hard task that requires a precise TCS localization system due to the tight safety margins inside the building.

Due to a specific propriety of the task, the radioactive loads, it is not advised to install sensors in the vehicle because the radiation would rapidly damage exposed electronic hardware. The solution proposed is to strategically install sensors on the building, where the radiation has lower levels due to building shield, the sensors selected to observe the entire operation area are Laser Range Finder (LRF) sensors, these are precise and accurate from short to long ranges, the measures have no interference from the magnetic fields on the tokamak and sensor’s electronic part exposure to radiation can be minimized, leaving only the mirror exposed, which suits the current problem.

A network of LRF sensors are required to avoid not covered areas, occluded by scenario obstacles. The main problems addressed in this paper are the placement of these multiple sensors on the building and the posterior estimation of vehicle pose (position and orientation) based on these sensor’s measurements.

Although the motivation is the localization in ITER scenarios, this localization system is suitable to other indoor scenarios, providing an accurate localization to autonomous vehicles. Localization problem is well known in robotic field due to his importance in task execution, in particular, for guidance of an autonomous vehicle.

Mobile vehicle localization is a crucial area of robotics, many localization techniques are available, based on EKF (2), (17), (12]), or PF (6), (10), (16). General applications of these methods depend on sensors mounted on board of the vehicle, our solution has the constraint of leaving the vehicle without sensors. Now a distributed sensor network observes the vehicle and the surroundings instead of the typical framework where the vehicle observes the surroundings. Some localization systems with this principle have been implemented, resorting to cameras, (13), to WiFi, (3) or to RFID, (4), but these techniques are not very reliable and have low accuracy.

This solution uses LRF sensors (18), (21), these have great accuracy and long ranges. The LRF network layout will play a major role on the system’s performance, so, each sensor placement should be optimized. Simulated Annealing (SA), (22), is used for this task, retrieving an optimal coverage network layout from the set of possible sensor placements.

Up to date a similar work on localization with laser range finder laser network mounted on the building, with a mobile vehicle completely free of sensors, was not found, it is an innovative technique that allows a localization with sensor free robot. This have applications on industry and logistics,

J. Ferreira and R. Ventura are with Institute for Systems and Robotics, Instituto Superior Técnico, Lisbon, Portugal
A. Vale is with Instituto de Plasmas e Fusão Nuclear, Instituto Superior Técnico, Lisbon, Portugal
Manuscript received October 27, 2010
the common Autonomous Guided Vehicle (AGV) have no longer to be restricted to a stripe or cable on the floor, trajectories can be dynamically changed.

This paper is organized as follows. Section 1 introduces the problem and the proposed solution. Section 2 develops a method for sensor placement optimization, retrieving optimized sensor network configurations. Section 3 addresses and compares two localization methods based on standard methodologies, Extended Kalman Filter (EKF) and Particle Filter (PF). Section 4 tests the robustness of localization methods for various sensor network configurations. Section 5 concludes the paper and gives direction for further developments.

II. SENSOR PLACEMENT OPTIMIZATION

To implement the proposed location system, it is crucial that the sensory part covers the entire scenario. Indoor environments have, typically, obstacles creating occlusions, only one sensor can never cover the entire area. To have a complete coverage of the environment is required a network of sensors with different placements. Optimizing them is crucial to enhance performance.

This section proposes a method for optimizing sensors’ placements, maximizing the coverage, with the goal of minimizing the amount of sensors to be installed.

The sensor coverage in an indoor environment is similar to the art gallery problem stated by [5]. This problem inquires how many observation points are necessary to cover an entire area with a given number of walls. This is a NP-problem as proved by [1]. Any solution to these problems is very fast to verify but there is no fast solution known. The difficulties in optimization on a visibility problem like this are known, and some approximate solutions have been proposed, applied usually in visual sensors [19], like cameras in surveillance systems like [7]. There are developed methods to optimize coverage areas and enhance performance of visual sensors network ( [14], [15]). The improvement of visual sensor network coverage, like surveillance cameras [23] have similar optimizing difficulties, the main constraints are the same, the occlusions in indoor environments. A preliminary study on LRF network coverage is addressed in [8], it describes the optimization framework presented in this thesis.

To optimize the sensor placement is required a map of the scenario. The map is a simplified 2D layout since, usually, the vehicle operates on floor level. This way the map of the scenario consists of 2D cartesian points MP, defining the corners and a set of edges connecting these points ML (1).

\[
MP = \begin{bmatrix} p_{1x} & p_{1y} \\ \vdots & \vdots \\ p_{Nx} & p_{Ny} \end{bmatrix}, \quad ML = \begin{bmatrix} i_1 & f_1 & v_1 \\ \vdots & \vdots & \vdots \\ i_M & f_M & v_M \end{bmatrix}
\]

The sensor state \( \hat{s} \) (2) on Fig. 1(a), is composed by \( \lambda \in [0,1] \), a parameter of sensor position along the wall, \( \theta_s \), the sensor orientation, \( \beta \), the wall where the sensor is installed and \( \Phi_s \), the sensors Field of View (FoV). With \( \lambda = 0 \) the sensor is installed in the initial point of wall \( \beta \), as \( \lambda \) grows sensor moves along the wall reaching \( \lambda = 1 \) in the final point. An equivalent reference ti sensor placement is \( (\hat{x}_s, \hat{y}_s) \) that gives the absolute position on the map.

\[
\hat{s} = [\lambda \quad \beta \quad \theta_s \quad \Phi_s]^T
\]

Including multiple sensors, a network state is compiled, \( \hat{S}_L \) (3) including a state for each sensor. \( L \) is the total number of sensors used by the network.

\[
\hat{S}_L = \{ \hat{s}_1, \ldots, \hat{s}_L \}
\]

The areas where a sensor can be installed on the map must be predefined before the optimization, they represent the Sensor State Space \( \hat{SSS} \).

\( \hat{SSS} \) is defined as a path traveling trough all map walls where the sensor can be installed, this path, represented in Fig. 1(b), is discretized for computation purpose, forming \( dSSS \), and parameterized by an index, \( Tag \), that indexes each state on \( \hat{SSS} \). For each \( L \) there’s an optimal network configuration, the variables to optimize are the sensors’ placements. The optimization process must compute the optimal placement for \( L \) sensors, with \( \hat{s}_1 \ldots \hat{s}_L \in \hat{SSS} \) and respecting defined criteria.

A. Optimization Criteria

The most important criteria is the coverage that the sensor network have on the map. Localization system must be valid at all possible vehicle positions, it must cover the entire scenario or else, when the vehicle travels to an uncovered zone, with no sensor readings, there is no way to localize it.

The percentage of the total map area covered from the sensor network was the best way to measure this criteria. The optimization process consists in the maximization of this percentage having the sensors positions has variables.

To evaluate it, first a Visibility Polygon (VP) (4) is extracted with function \( Poly(\hat{s}) \), described in Fig. 2, this polygon represents the area covered by a single sensor. To combine the coverage from the entire network a simple boolean union between each sensor respective VP is performed, evaluating the area posteriorly (5). Evaluating the coverage along \( \hat{SSS} \) path for the example map, it yields the graph in Fig. 3. This is Coverage graph, where it plots the coverage percentage depending on the sensor state (Tag).

\[
Poly(\hat{s}) = [a_1 \quad a_2 \ldots \quad a_K]^T
\]
Meeting the most important criteria that we have establish, SA is the algorithm chosen to optimize sensor networks. To compute the network with optimal coverage using SA algorithm, the cost function is actually a fitness function because it s maximized instead of minimized. This is not a big challenge and is done on the algorithm implementation itself. From now on the SA is “inverted” and tries to maximize the fitness function. This function is the coverage of the network \( C(\hat{S}_L) \) (5). This function is evaluated on the search space \( dSSS_L \) during each run of SA. The best solution \( (\hat{S}_L^*) \) is the tuple (8) for which the coverage function gets the higher value \( (C(\hat{S}_L^*)) \). The objective of the optimization process is to find the global maximum of this function (7).

\[
C(\hat{S}_L) = \text{Area} \left[ \bigcup_{j=1}^{L} \text{Poly}(\hat{s}_j) \right], \quad \hat{s}_j \in \hat{S}_L \tag{5}
\]

Another criteria, not optimized on this paper, but very interesting is the Redundancy (6), this is defined as the area covered from more then one sensor. This is important in cases of sensor failure, or temporary occlusions of one sensor. It is shown in the results but it is not introduced in the algorithm evaluation function.

\[
R(\hat{S}_L) = \text{Area} \left\{ \bigcup_{j=1}^{L} \bigcup_{i=1}^{L} \text{Poly}(\hat{s}_j) \cap \text{Poly}(\hat{s}_i) \right\}, \quad i \neq j \tag{6}
\]

### B. Optimization Algorithm

Evaluating the coverage function along the state space it is possible to perceive that this function is non-smooth, non-convex, and consequently, very hard to optimize and to discover global maximum. Adding sensors, the optimization difficulties grow exponentially. The solution are Monte Carlo optimization methods. Analyzing the search space and the cost function characteristics, SA algorithm is suitable for this problem:

- Discrete state space, \( dSSS \). Normally used in SA implementations.
- Non-Convex cost function, the coverage area along the possible sensor configurations has many local minima and narrow valleys, as seen in the example of Figure 3, SA algorithm is suited for this problem as it can avoid being trapped in local minima.
- Wide search space, being impossible to search the entire space. SA is a MC method which is needed to compute a solution in a reasonable time.
- Implementation simplicity, SA is one of the most simple to implement MC methods. As this is just a first step towards the localization system itself the method simplicity is an important criteria to choose it.

Meeting the most important criteria that we have establish, SA is the algorithm chosen to optimize sensor networks.

For experimental proposes two maps were used, the first is the indoor map of a floor of the Tokamak Building(TB) in ITER shown on Figure 4(a) where the solid edges are walls and dashed ones are VV doors. The second map is a general propose WareHouse Building (WHB) on Figure 4(b), synthesized to prove the effectiveness of this method in other environments, solid edges are walls and dashed edges are shelves. Sensors can only be mounted on solid walls. Figures 4(a) and 4(b) show the path used to create the \( dSSS \) set, this was done with a discretization step of 0.2 m for translation and \( \pi/60 \) for rotations. The CG for these two maps are shown in Figure 5(a) and 5(b) , this is sufficient to notice the behavior of this functions for a network of just one single sensor, very hard to optimize as the sensor network grows. Figure 8 shows the coverage \( C(\hat{S}_L^*) \) and redundancy \( R(\hat{S}_L^*) \) depending on \( L \), for the network with maximum coverage. The graph also show the minimum coverage \( (\min(C(\hat{S}_L))) \), the minimum and the maximum redundancy \( (\min(R(\hat{S}_L))) \) and \( (\max(R(\hat{S}_L))) \) obtained in the ten SA runs., the behavior of maximum coverage as the number of sensors grow is similar on both maps, and is

\[
C(\hat{S}_L) = \text{max}_{S_i} C(\hat{S}_L) \tag{7}
\]

\[
(s_1, \ldots, s_L) = \hat{S}_L^* = \text{argmax}_{S_L} C(\hat{S}_L) \tag{8}
\]

The output of the algorithm is not always the global maximum, but for multiple runs\(^1\) of SA, the probability of finding the global maximum rises. With more runs of the algorithm more certainty we have about the location of the maximum.

The tuple that gives rise to the maximum coverage contains the configurations for the sensors on the network, placing them with those configurations guarantees the optimal coverage for a network with \( L \) sensors.

### C. Experimental Results

For experimental proposes two maps were used, the first is the indoor map of a floor of the Tokamak Building(TB) in ITER shown on Figure 4(a) where the solid edges are walls and dashed ones are VV doors. The second map is a general propose WareHouse Building (WHB) on Figure 4(b), synthesized to prove the effectiveness of this method in other environments, solid edges are walls and dashed edges are shelves. Sensors can only be mounted on solid walls. Figures 4(a) and 4(b) show the path used to create the \( dSSS \) set, this was done with a discretization step of 0.2 m for translation and \( \pi/60 \) for rotations. The CG for these two maps are shown in Figure 5(a) and 5(b) , this is sufficient to notice the behavior of this functions for a network of just one single sensor, very hard to optimize as the sensor network grows. Figure 8 shows the coverage \( C(\hat{S}_L^*) \) and redundancy \( R(\hat{S}_L^*) \) depending on \( L \), for the network with maximum coverage. The graph also show the minimum coverage \( (\min(C(\hat{S}_L))) \), the minimum and the maximum redundancy \( (\min(R(\hat{S}_L))) \) and \( (\max(R(\hat{S}_L))) \) obtained in the ten SA runs., the behavior of maximum coverage as the number of sensors grow is similar on both maps, and is

\(^1\)Executions of SA with the same state space, which increases the probability of finding the global maximum
Fig. 4. TB and WHB with path along $dS_{SS}$

$$G(L) = C(\hat{S}_L) - C(\hat{S}_{L-1})$$

For scenarios where there is no tolerance for failure, redundancy of the system is very important, although not optimized on this work. The redundancy grows with the coverage, but since it is not optimized has more variation then the coverage along the different runs. In some situations it is better not to choose the maximum coverage network, because there are solutions where, losing some coverage, the network gains a lot of redundancy. For the TB values (8(a)) for a network with eight sensors ($L = 8$), the coverage is almost the same on all runs of SA, but the maximum coverage network has a redundancy below 60% while there is a network with almost 90% redundancy, with nearly the same coverage, which is a better solution considering the two criteria. Naturally, in this case the best solution does not coincide with the maximum coverage criteria.

Optimal coverage network configurations $\hat{S}_L$ are shown in Figure 6 and 7, to notice the behavior when a new sensor is added, the tendency is to stay away from the others to maximize coverage over the entire scenario. When $L$ is high, the network attributes a sensors for very specific areas, where the others can not cover, and so the separation between them is less visible.

The redundancy is not optimized on this work because it takes a long time computing, the present values, are processed off-line, after the SA algorithm completes the coverage optimization.

III. VEHICLE LOCALIZATION

A Localization system requirement in a known environments is to localize the vehicle given the map layout, the vehicle layout and the correct sensors positions.. Usually this is made using on board sensors, with methods like EKF Localization or Monte Carlo Localization (MCL) ( [9]). In this work, the main difference is the installation of sensors in the scenario instead, leaving the vehicle just as an actuator.

This specification brings new challenges, in common methods, with the vehicle movement, all measures are expected to change, since they are observations of the environment. In this application some change and others stay the same because some observe the vehicle and others do not. There is no trivial way to decide if all measures are interesting or just the ones hitting the vehicle, and no trivial way to classify them as hitting or not hitting the vehicle. The main goal is the creation of a localizations system that, overcoming this issues, integrates the sensor measurements and estimate the vehicle pose correctly with a certain accuracy. The vehicle pose (10) tells the position of vehicles’ center and respective orientation.

$$x_t = [x_t^r, y_t^r, \theta_t^r]^T$$

$$\hat{x}_t = [\hat{x}_t^r, \hat{y}_t^r, \hat{\theta}_t^r]^T$$
Localization problem consists on estimating the correct pose, returning an estimation, $\bar{x}_t$ (11).

Sensor measurements come from a network of LRF sensors [11], each one with specific configurations ($s_i$) (12). $x_s^i$, $y_s^i$ and $\theta_s^i$ are the pose of the i-th sensor, $\Phi_s^i$ is the FoV, $\delta_s^i$ the angular resolution and $\sigma_s^i$ the standard deviation for distance measurement errors. Some of these parameters are imported from the optimization problem of section 7. Only $\delta_s^i$ and $\sigma_s^i$ are established here depending on equipment installed.

The integration of several sensors is mandatory to cover all possible vehicle positions, the sensor network is defined in $S$ (13) containing $L$ sensors.

$$s_i = \begin{bmatrix} x_s^i & y_s^i & \theta_s^i & \Phi_s^i & \delta_s^i & \sigma_s^i \end{bmatrix}^T$$

$$S = [s_1 \ s_2 \ \ldots \ s_L]$$

Measurements acquired from each sensor $s_i$ are arranged in the array $z(s_i)$ (14) where $d_j^i$ is a distance corresponding to laser beam with direction $\varphi_j^i$ acquired by the i-th sensor in scenario. Measures available on the network are compiled in $Z$, (15) The number of measurements for each sensor depends on his FoV and angular resolution.

$$z(s_i) = \begin{bmatrix} d_1^i \ \cdots \ d_{P_i}^i \ \varphi_1^i \ \cdots \ \varphi_{P_i}^i \end{bmatrix}^T$$

$$Z = \begin{bmatrix} z(s_1) \ \cdots \ z(s_L) \end{bmatrix}$$

Measurements coming from LRF are corrupted with noise, mainly in the ranges measured, the error in angular position is very small and neglected in this work. The error in ranges can be modeled as corrupted with zero mean gaussian noise as shown in [24]. The variance for this noise is described for each sensor by his standard deviation, $\sigma_s^i$.

Based on these problem framework, two Bayesin methods of estimation are tested, EKF and PF [20], to choose which is better suited to the task. They will integrate sensor measurements with dynamic information, like velocity commands, to achieve an estimation with a certain accuracy. This paper presents an adaptation of common methods to this new framework, the main differences to common approaches are on observation models used. The rest of the bayesian filters implementation is equal to standard applications (20).

### A. Observation Model

1) Extended Kalman Filter: The observation model $h(\bar{x}_t)$ is a non-linear function that predicts measurements based on a estimation of vehicle pose $\bar{x}_t$. This observation model predicts, for each sensor, the distances measured by a laser ray casted in predefined angular position $\varphi_j^i$. These angles are predefined by the sensor orientation $\theta_s^i$, angular resolution $\delta_s^i$ and FoV $\Phi_s^i$.

A ray is casted in each direction $\varphi_j^i$, all the possible intersections with vehicle’s edges are considered. Since the laser measurements hitting the map walls are not dynamic with the vehicle pose, there is no need for an observation model for them, so for each sensor, the model prediction, $h_i(\bar{x}_t)$ (16), is an array of distances, $\bar{d}_j$, one for each direction, $\varphi_j^i$, that hits the vehicle. The length of this array, $J = \#\chi(\bar{x}_t, s_i)$.
There are four distinct sets of measurements, the ones hitting $\bar{x}_t$, $\chi(x_t, s_i)$ is a set of angular positions, such that laser rays casted in those directions, from sensor $s_i$, hit the vehicle with pose $\bar{x}_t$. $\chi(x_t, s_i)$ is the set of angular positions which rays do not hit the vehicle. Figure 10 shows the parameters and variables used in observation model, for a general vehicle layout. So, $h(x_t)$ is a prediction of $Z$ but only for measures that hit the vehicle.

$$h_i(x_t) = \left[ \bar{d}_i^1, \ldots, \bar{d}_i^j, \varphi_i^1, \ldots, \varphi_i^\ell \right]$$  \hspace{1cm} (16)

$$h(x_t) = \left[ h_1(x_t), \ldots, h_L(x_t) \right]$$  \hspace{1cm} (17)

The problem with these approach is that, at time $t$, the EKF does not know the true pose of the vehicle, $x_t$, it knows the ápriori estimation, given by the movement model, $\bar{x}_t$. This will have some impact in the observation model, since in reality, the laser rays hitting the robot are not the same as the ones hitting in estimation.

There are four distinct sets of measurements, the ones hitting the robot in real pose and in estimated pose, $\chi(x_t, s_i)$ and $\chi(\bar{x}_t, s_i)$, and the ones hitting map walls or fixed obstacles in real and estimated pose, $\chi(x_t, s_i)$ and $\chi(\bar{x}_t, s_i)$, respectively. Figure 11 presents an example where the estimation is not far from real pose, but the sets does not coincide. This situation represents a non-gaussian and non-smooth behavior of the measurement model, being this the principal cause for a poor performance of EKF on this application.

2) Particle Filter: PF uses the observation model to attribute weights for each particle, observation model is a likelihood function, $h(\bar{x}_t^n) = p(z_t|\bar{x}_t^n)$, where $\bar{x}_t^n$ is the particle $n$ state at time $t$, it evaluates for each particle, the resemblance between measurement coming from sensors, $Z$, and predicted for particle.

PF has no restriction on the model, and the method predicts all possible measurements, $\varphi_j^i \in \chi() \cup \chi()$, the predicted range is given by $\bar{d}_j^n$ for every angular position, $\varphi_j^i$, making $y_i(x_t^n)$ (18) the same length as $z(s_i)$ (14).

Intuitively the $y_i(x_t^n)$ is the sensor, $s_i$, observation if the vehicle was in pose $x_t^n$, comparing $y_i(x_t^n)$ with $z(s_i)$ for all sensors, will give a good evaluator if particle $n$ state is close to the true state. The closer the state is, the higher the measurements’ likelihood and the higher is the particle weight

$$w_i^n = p(d_j^n|\bar{x}_t^n) = \frac{1}{\sqrt{2\pi\sigma^2_{\text{range}}}} e^{-\frac{(d_j^n - \bar{d}_j^n)^2}{2\sigma^2_{\text{range}}}}$$  \hspace{1cm} (19)

LRF sensors have a gaussian error on ranges measured, so a way to compute the likelihood of a measurement prediction $(w_i^n)$ is a 1D gaussian, with variance $\sigma^2_{\text{range}}$ and mean $\bar{d}_j^n$. The likelihood of this measurement is given by $19$

The weight of a particle $(w_i^n)$, is proportional to the likelihood $p(Z|x_t^n)$, the sum of all particle weights must be normalized to sum one (21), this way, the set of weights are a discrete probability distribution. To obtain this distribution, each measurement is assumed to be an independent random variable, so the joint probability is given by $20$. Figure 12(a) shows the appliance of this method, retrieving weights for several measures hitting the vehicle, for map walls, the procedure is the same.

The use of gaussian function (19),will result in a very low likelihood for these measurements and consequently, since we use 20 to estimate the particle weight, a very low weight to the respective particle. This situation is similar to have many particles around the correct pose, but their likelihood is very low, obviously the simple gaussian function is not suited for this situation, or else very few particles would have a sufficient weight to survive the resample step, making the PF perform very poorly.

The idea to overcome this problem is the use of a new likelihood function, instead of a gaussian centered in predicted range. This function will have in account this discontinuity and will smooth it. To implement it, classifying the measurement type is crucial.

Returning the previous set notation from Kalman filter implementation, $\chi(x_t, s_i)$ is a set of angular positions, such that laser rays casted in those directions, from sensor $s_i$, hit the vehicle with pose $x_t$. $\chi(\bar{x}_t, s_i)$ is the set of angular positions such that rays do not hit the vehicle. Applied to PF, there are several of these sets, for each particle predicted state $(\bar{x}_t^n, s_i)$ and $\chi(\bar{x}_t, s_i)$, and for real pose $(\chi(x_t, s_i)$ and $\chi(x_t, s_i)$). There is no need for compute the sets $\chi(x_t, s_i)$ and $\chi(\bar{x}_t, s_i)$, this effort is avoided by introducing the new likelihood function.

The two branches of $p(d_j^n|\bar{x}_t^n)$ function, are drawn in Figure 13, and their appliance is shown in Figure 12(b) similarly to gaussian model in Fig. 12(a). But now if the ray is expected to hit the vehicle it uses function on Fig. 13(a) if not it uses the one in Fig. 13(b). This is an adaptation of measurement model to this situation.

The weighting of these likelihood function must respect some rules, the first is that the weights must sum up one, this is to keep $p(d_j^n|\bar{x}_t^n)$ as a probability distribution over $d_j^n$, the other is the adjustment of the weights.
The methods were tested in multiple trajectories on TB, like the one in section 2, in this paper only an experiment with a clock wise (CW) trajectory is shown. TCS travels around the building corridors, like shown in Fig. 14(a) and the results for position error (norm), $||l_{err}||$ (22), and orientation error, $\theta_{err}$ (23), are on Fig. 15(a) for EKF and 15(b) for PF. Results were obtained in simulation environment, like the one shown in Fig. ??, with the same four sensors installed.

$$l_{err}|(t) = \sqrt{(\hat{x}_t - x_t)^2 + (\hat{y}_t - y_t)^2}$$  \hspace{1cm} (22)

$$\theta_{err}(t) = \theta_t - \theta^t$$  \hspace{1cm} (23)

The Vehicle Localization section makes an extensive description of both EKF and PF filter implementation for a distributed LRF sensor network, both methods have powerful capabilities, but by the results shown the PF is a better choice for this kind of problem. The accuracy, robustness and stability is better for PF estimation then for EKF, tests performed during this section are a good proof of this. The choice for the three trajectories shown CWtraj, CCWtraj and VVtraj, is directly connected to the functions of the TCS and the future application, the two CW and CCW trajectories on the corridors are a good way to evaluate the performance, since the corridors are the main area visited by the vehicle. The last, VVtraj, is an example real trajectory.

The performance in special situations, like loss of track, stopped vehicle is also very important, inside the rad-hard scenario there is no room for failures, and the system must be capable of self recovering from eventual malfunctions. Table I has a summary evaluation of the two methods, where the classification has 3 levels, - for bad performance, + for acceptable performance, with limitations or failures, ++ for good performance.

From the tests performed and results obtained, the logical
choice to integrate the system localization system is the PF, although it becomes hard to compute for a very high number of measurements. The computational effort pays back in performance.

IV. LOCALIZATION DEPENDENCE ON SENSOR PLACEMENT

The entire system proposed in this paper concerns mainly the localization system, but also an algorithm to optimize the sensor’s placements (Figure 16). This section studies the effects of sensors parameters on localization performances, analyzing the main error characteristics facing different network configurations. It is basically a section of experimental results with different network configurations. The objective is to conclude whether optimal sensor placement is crucial to have a good performance, or the algorithms are robust to multiple sensor placements and still perform at the same levels. Another consideration is the computational effort necessary as the network grows, being that an important topic because these methods should perform at real-time. Concerning the sensors characteristics itself, it is also studied how different angular resolution affects the localization performance. The results shown in this chapter should help to refine optimization criteria, if necessary, and contribute to the understanding of method robustness to failures on sensors.

Following results were performed in TB:

- Fig. 17 shows errors obtained for optimal networks with different number of sensors, and corresponding coverage, like optimized in section 2.
- Fig. 18 tests performance for different redundancies for an optimal network of 8 sensors with coverage 99% with multiple paths, yielding the mean errors shown.
- Table II shows values of errors for tests performed with an optimal network of 8 sensors with coverage 99% and redundancy 87%, but with failure of one sensor, for a network of 8 sensors, one was turned off while the vehicle travels along the corridors.

From results obtained in simulation concerning the robustness of a method, PF is clearly more robust.

V. CONCLUSION

The creation of a localization system for a vehicle, operating indoors, with no sensors on board was the main objective of this thesis. It was achieved resorting to a laser range finder sensor network, installed on building walls. This network observes the surroundings, and, integrating measurements with a Bayes filter, returns the estimated pose of the vehicle (position and orientation). Being the sensors installed on walls, they are static, and their placement is crucial to the system performance. The areas covered by the sensors, and the redundancy of the system depend critically on sensor placement. An optimization procedure was developed thinking in this critical dependency, the better the network, the better the performance. Simulated Annealing was the selected method to optimize the network placements, it optimizes an evaluation function that, on section 2, contemplates only the covered area. This is the main criteria to optimize, because if there are areas without coverage, observation on vehicle cannot be made, and the estimation is far away from real pose.

From the several experiences with the localization methods, we conclude that redundancy is also a very important factor to maximize. Besides allowing the system to work correctly during a sporadic sensor failure, during normal operation it gives more information to the system. If an area is observed from several sides, when the vehicle passes by, the uncertainty on estimation is much smaller than observing from just one side.

Localization algorithms are based on Bayes filters, two
proposes were evaluated, one based on EKF and the other on PF. From section 3 we can conclude that PF is the better suited method for this application. It is more accurate, and robust than EKF, in every situation tested. The better behavior of PF is due to its better adaptability to non-linear, non-smooth processes while the EKF is confined to gaussian representation. Filter belief representation is also limited to a gaussian function for EKF, but PF can have multi-modal distributions.

Having a global network also allows for fast global localization, since we have always measurements from the entire building. Computation effort is the weakness of PF, it is much slower than EKF, but, it still performs in real time, and pays back in estimation precision and accuracy. Performance for the overall system, network plus localization method, is evaluated in section 4, where it is proven that PF is better than EKF, but, it still performs in real time, and pays back in estimation precision and accuracy.

Performance for the overall system, network plus localization method, is evaluated in section 4, where it is proven that PF is more robust networks with few sensors, to sensor failures or to modification of sensor placements, decreasing redundancy. PF is the best method to apply in this case, but still has some problems to solve.

- **Redundancy optimization** — currently the SA algorithm only maximizes the coverage of the network, but as seen in this paper redundancy is also very important. So a method for multiple criteria optimization should be addressed in future work, although by including redundancy to the current evaluation function or by using methods of multi objective optimization.

- **Calibration system** — it was identified a problem on localization system, it is intolerant to sensor deviation from predefined placement, it introduces an offset on estimated poses. Has this is a static error, it can be corrected with a calibration system. After assemble stage there must be a system that given the map and the measurements from sensors can calibrate the system and retrieve the correct placements for each sensor.

- **PF measurement pre-selection** — one of the current problems with PF solution is the computational effort, as the sensor network grows the method becomes slower. But from operation, it is known that when the estimation is near the correct pose, the majority of measurement integrated are irrelevant to the system. Just the measurements around the pose contain information. A method to filter the interesting and irrelevant measurements can decrease a lot the effort needed to compute PF iterations.

- **Multiple vehicles** — If there are more than one vehicle traveling along the building, it should be possible to track it also, apart from occlusions between vehicles, the network is always capable to observe both. So a possible interesting development of this technique is the inclusion of multiple objects to track, vehicles with different shapes, etc.

- **Practical Implementation** — It is still far away from this, much more work have to be done, but still, it is a goal for future work, to implement this work on a practical experiment.

Although it has some failures, and is still only working in simulation environment, it yields promising results, not only for application in ITER, but for other application with this type of requirements, or even to implement an AGV system where the paths can change dynamically, there is no need to physically install cables or stripes on the floor. Today’s factories are in constant layout change, depending on the product in manufacturing, so an efficient way to have different paths for autonomous vehicles can be of industry interest as well.

### TABLE II

<table>
<thead>
<tr>
<th></th>
<th>EKF</th>
<th>PF</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Failures</td>
<td>$\sigma$ 58.9</td>
<td>$\sigma$ 24.2</td>
</tr>
<tr>
<td></td>
<td>$\mu$ 73.9</td>
<td>$\mu$ 35.8</td>
</tr>
<tr>
<td>Sensor Failure</td>
<td>$\sigma$ 754.6</td>
<td>$\sigma$ 37.2</td>
</tr>
<tr>
<td></td>
<td>$\mu$ 351.4</td>
<td>$\mu$ 40.1</td>
</tr>
</tbody>
</table>

### References


