Journal of Intelligent & Robotic Systems
https://doi.org/10.1007/s10846-017-0765-5

Map-Matching Algorithms for Robot Self-Localization: A Comparison Between Perfect Match, Iterative Closest Point and Normal Distributions Transform

Héber Sobreira[1] · Carlos M. Costa[1] · Ivo Sousa[1] · Luis Rocha[1] · José Lima[1,2] · P. C. M. A. Farias[1,3] · Paulo Costa[1,4] · A. Paulo Moreira[1,4]

Received: 16 October 2016 / Accepted: 18 December 2017
© Springer Science+Business Media B.V., part of Springer Nature 2018

Abstract
The self-localization of mobile robots in the environment is one of the most fundamental problems in the robotics navigation field. It is a complex and challenging problem due to the high requirements of autonomous mobile vehicles, particularly with regard to the algorithms accuracy, robustness and computational efficiency. In this paper, we present a comparison of three of the most used map-matching algorithms applied in localization based on natural landmarks: our implementation of the Perfect Match (PM) and the Point Cloud Library (PCL) implementation of the Iterative Closest Point (ICP) and the Normal Distribution Transform (NDT). For the purpose of this comparison we have considered a set of representative metrics, such as pose estimation accuracy, computational efficiency, convergence speed, maximum admissible initialization error and robustness to the presence of outliers in the robots sensors data. The test results were retrieved using our ROS natural landmark public dataset, containing several tests with simulated and real sensor data. The performance and robustness of the Perfect Match is highlighted throughout this article and is of paramount importance for real-time embedded systems with limited computing power that require accurate pose estimation and fast reaction times for high speed navigation. Moreover, we added to PCL a new algorithm for performing correspondence estimation using lookup tables that was inspired by the PM approach to solve this problem. This new method for computing the closest map point to a given sensor reading proved to be 40 to 60 times faster than the existing k-d tree approach in PCL and allowed the Iterative Closest Point algorithm to perform point cloud registration 5 to 9 times faster.

Keywords 2D laser scan · Map matching · Robot self-localization

✉ Héber Sobreira
heber.m.sobreira@inesctec.pt

Carlos M. Costa
carlos.m.costa@inesctec.pt

Ivo Sousa
ivo.e.sousa@inesctec.pt

Luis Rocha
luis.f.rocha@inesctec.pt

José Lima
jllima@ipb.pt

P. C. M. A. Farias
paulo.farias@ufba.br

Paulo Costa
paco@fe.up.pt

A. Paulo Moreira
amoreira@fe.up.pt

[1] INESC-TEC, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal

[2] Polytechnic Institute of Bragança, Campus Sta Apolónia, 5301-857 Bragança, Portugal

[3] Polytechnic School of Federal University of Bahia, Rua Aristides Novis, 40210-630 Salvador, Brazil

[4] Faculty of Engineering of University of Porto, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal

Published online: 23 January 2018

1 Introduction

Industrial mobile robots (AGVs, Automatic Guided Vehicle), are vehicles that can self-localize and move autonomously without human intervention. They are commonly used to transport materials between work stations in warehouses and production lines. They have been used in industrial environments for more than 50 years and both the algorithms and hardware in which they rely on have been evolving in order to increase the accuracy, robustness and flexibility while decreasing the costs of construction and operation.

Regarding the localization systems in industrial environments, it is common to use solutions that rely on artificial landmarks, such as magnetic tape following, line tracking or reflector based laser triangulation [12, 13]. On the other hand, service robotics applications tend to use map-matching algorithms that take advantage of natural landmarks [11, 15] in order to reduce installation time and decrease operation maintenance costs. These natural landmarks are composed by a set of distances and angles to the detected objects (such as doors, walls, furniture and many others) that can be acquired through an on-board laser range finder. This method has the main advantage of not requiring the installation of dedicated artificial landmarks in the environment, which in some factories shop floors might not be a viable option. However, it is expected that even without special markers and in straight corridors, the localization system remains accurate and robust. Despite its advantages, this approach needs to process a higher amount of sensor data efficiently in order to provide real-time localization and needs more advanced techniques in order to tolerate outliers in the sensor data. Therefore, the map-matching algorithms must be optimized in terms of accuracy, processing time, convergence speed and also sensor noise robustness.

Map-matching is a method of self-localization for mobile robots in which the sensor data of the local environment is matched with an already stored map. With this in mind, this paper addresses the comparison of three of the most used algorithms in localization systems based on environment natural landmarks, which are the Perfect Match (PM) [5], the Iterative Closest Point (ICP) [2] and the Normal Distributions Transform (NDT) [3].

The Perfect Match algorithm has increased its popularity within the robotics community mainly due to its successful application in the Middle Size League (MSL) robot soccer competitions, in which it was able to provide robust localization for robots that require high frequency decision and locomotion control. For its turn, ICP is a frequently used approach to solve the map matching problem for 2D and 3D point-clouds. The NDT algorithm is currently an alternative to ICP, that does not rely on the establishment of correspondences. As a result, the NDT is theoretically more immune to the sensors noise and sensors sampling resolution, as ICP assumes that sensor readings have been produced in the same position as the map reference points. Furthermore, another drawback of ICP is that in each new iteration a new function is minimized, since the correspondence information between points obtained from previous iterations are not used. As a result, a greater number of iterations may be needed to converge to a good solution [1]. The implementations of ICP and NDT that were used for this comparison are available in the Point Cloud Library (PCL), one of the most relevant open-source projects related to robotics perception.

The comparison and evaluation of these three algorithms were performed considering different metrics, namely, the computational weight of each algorithm, the speed of convergence, the maximum admissible initialization error (maximum rotation error introduced in the initial pose estimation of the robot that the map-matching algorithms can tolerate), and finally the robustness of the algorithms in the presence of outliers on the robot sensor data.

For the execution of this comparison we used the ROS (Robot Operating System) framework. In this regard, the work of Carlos et al. [4] was considered to make the interface between ROS and PCL, as it was designed to solve the robot localization problem and is completely parameterizable.

In terms of results, these were extracted both in a simulated environment (using the Stage Simulator) with virtually generated sensor information (laser range finder data), as well as using a real robotic platform (Jarvis robot), which has a SICK NAV350 laser range finder on board used for map-matching and ground-truth (relying on a reflector based triangulation system).

This paper is organized as follows: after the brief introduction given in this section, the Section 2 describes the algorithms (Perfect Match, Iterative Closest Point and Normal Distribution Transform). Then, Section 3 addresses the comparison of experimental results retrieved with each algorithm for each of the evaluation metrics. Finally, Section 4 concludes this paper and presents some future work.

2 Algorithms Definition

In this section it is presented the main concepts of the map-matching algorithms used in this experimental comparison.

2.1 Perfect Match

The Perfect Match is a light computational algorithm that was proposed by M. Lauer et al. in [5]. In this algorithm, the vehicle pose is computed using 2D distance information from the surrounding robot environment which can be acquired using LIDARs or CCD cameras. The main goal of the algorithm is to minimize the matching error (fitting error between the data acquired and the environment map). Overall, the Perfect Match algorithm can be divided into three steps: 1) matching error and gradient computation; 2) optimization routine based on the Resilient Back-Propagation (RPROP); and 3) co-variance estimation using the second derivative. Using the acquired map of the environment, that is then converted to an occupancy grid map, it is possible to compute the distance and gradient maps of the environment [5]. In the case of the distance map, each cell gives the distance to the closest obstacle. For the gradient maps, there are two measurements, one for the x direction and another to the y direction. The first one gives the direction of the variation of the distance map with the variation of the x position. The second one shows the direction variation of the distance map with the y position variation. These three maps (distance map, and x/y gradient matrices) can be pre-computed in order to speed up the computation of the Perfect Match algorithm.

Considering now a list of points of a Laser Range Finder scan defined as $PList$. The point i of this list in the world referential frame is $PList(i) = (x_i, y_i)$. The cost value is given by equation (1) where d_i and E_i represents, respectively, the distance matrix and the cost function values of point i.

$$E = \sum_{i=1}^{PList.Count} E_i, \quad E_i = 1 - \frac{L_c^2}{L_c^2 + d_i^2} \tag{1}$$

The parameter L_c is used to discard points with large error E_i, increasing the robustness of the algorithm to outliers. To minimize the error function, the Perfect Match algorithm uses the RPROP optimizer. The output of this algorithm is the state of the robot, x, y and θ (robot pose) that minimizes the map-matching error. For a detailed description of the algorithm please refer to [5].

2.2 Iterative Closest Point

The Iterative Closest Point algorithm is a commonly used map-matching method which tries to minimize the Euclidean distance between the input data and a reference model (in the self-localization problem it corresponds to the sensor data and the map of the environment).

From a mathematical point of view, consider two sets of 2D points, source A (with n points) and target B (with m points) $\subseteq \mathbb{R}^2$. The objective is to find a transformation function $u : A \rightarrow B$ that minimizes the mean squared distances (MSD) between A and B (Eq. 2).

$$MSD(A, B, u) = \frac{1}{n} \sum_{a \in A, b \in B} \|b - u(a)\|^2 \tag{2}$$

Incorporating the rotation (R) and translation (t) matrices into the matching function, the minimization problem can be written using Eq. 3.

$$\min_{u:A \rightarrow B} \frac{1}{n} \sum_{i=1}^{n} \|b_i - Ra_i - t\|^2 \tag{3}$$

With this mathematical formulation, the ICP tries in each iteration to minimize the $MSD(A, B, u)$ by switching between a matching and a transformation stage.

Matching Stage In this first stage, the objective is to minimize the mean squared distances $MSD(A, B, u)$ by finding the best correspondence between a point $a_i \in A$ and $b_i \in B$. This step is in its most basic form executed by selecting the point $b_i \in B$ with the minimum Euclidean distance to the point $a_i \in A$. Note that in the first iteration, t and R are normally set to $[0, 0]^T$ and to the identity matrix respectively.

Transformation Stage During the transformation stage, the objective is to compute the optimal R and t that minimizes (3), using the correspondences computed in the previous stage. ICP uses a simple least square solver to find the optimal linear transformation matrix $(R|t)$ that minimizes (3) [2]. For this purpose, the algorithm starts by subtracting from the reference and sensor point clouds their respective centroid, as shown in Eqs. 4 and 5.

$$a_i' = a_i - \frac{1}{n} \sum_{i=0}^{n} a_i \tag{4}$$

$$b_i' = b_i - \frac{1}{m} \sum_{i=0}^{m} b_i \tag{5}$$

This step helps simplifying the minimization problem [2]. Then, the cross-variance matrix is computed using the Eq. 6 with A' and B' being the set of points a_i' and b_i' respectively.

$$H = A'B'^T \tag{6}$$

Now, the rotation angle θ can be computed from Eq.7.

$$\theta = atan2((H(0,1) - H(1,0)), (H(0,0) + H(1,1)) \tag{7}$$

It can be shown that, the optimal solution for R and t that minimizes the objective function is given by Eqs. 8 and 9, where $\bar{a}$ and $\bar{b}$ are the points centroid computed in Eqs. 4 and 5.

In the end of the transformation stage, the source / sensor data is transformed using the estimated $(R|t)$ matrix and the algorithm goes back to the matching stage (with $(R|t)$ set to the identity matrix), unless a stopping criteria is verified (e.g. number of iterations, Euclidean error improvement between iterations, etc).

$$R = m \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \tag{8}$$

$$t = \bar{b} - R\bar{a} \tag{9}$$

The ICP 2D implementation used in this paper was the one available in the Point Cloud Library (PCL).

2.3 Normal Distributions Transform

The Normal Distributions Transform was introduced by P. Biber and W. Straßer [3] as a method for 2D scan registration and it was later extended for 3D scan matching [6, 7] by M. Magnusson. This map-matching algorithm creates a smooth surface representation of the environment that is modeled by a set of local probability density functions (PDFs). This representation is built from a set of reference points that are grouped into a set of fixed sized cells forming a voxel grid. Then, for each voxel grid cell that has at least a group of 6 points it is computed the mean (q) and covariance matrix (Σ) using the following equations:

$$q = \frac{1}{n}\sum_{i=1}^{n} k_i, \quad \Sigma = \frac{1}{n}\sum_{i=1}^{n}(k_i - q)(k_i - q)^T \tag{10}$$

with $k_{i=1..n}$ representing all the points inside each voxel grid cell. After initializing the 3D representation of the environment, the initial set of references points can be discarded and the probability of measuring a sample in a given voxel grid cell region is given by:

$$p(k) \sim -d_1 e^{-\frac{d_2 (k-q)^T \Sigma^{-1} (k-q)}{2}}. \tag{11}$$

In this equation, d_1 and d_2 are constants used to bound the effect of the sensor outliers [7], while q is the mean vector and Σ is the covariance matrix of the points contained within the respective cell.

To use the NDT approach for 3D position tracking, it is defined a number of parameters ($\overrightarrow{w} = \{tx, ty, tz, \phi_x, \phi_y, \phi_z\}$) to estimate and optimize. The 3D transformation function between two robot coordinate frames is represented by $\{tx, ty, tz\}$ for the translation and z-y-x Euler angles $\{\phi_x, \phi_y, \phi_z\}$ for the rotation, which can be represented as:

$$T(\overrightarrow{w}, \overrightarrow{k}) = R_x R_y R_z \overrightarrow{k} + \overrightarrow{t}. \tag{12}$$

Variable t represents the $\{tx, ty, tz\}$ translation parameters and $R_x R_y R_z$ is the rotation matrix built from the $\{\phi_x, \phi_y, \phi_z\}$ Euler angles. The goal of a scan registration is to estimate these 6 parameters from the sensor data given a pre-computed 3D representation of the environment. This is done using the Newton's optimization algorithm [10] in order to minimize the *score* function:

$$score(p(k')) = \sum_{i=1}^{n} -d_1 e^{-\frac{d_2 (k'_i - q_i)^T \Sigma_i^{-1} (k'_i - q_i)}{2}} \tag{13}$$

This *score* is calculated by evaluating the normal distribution of all points k'_i and summing the result ($score$) while also updating the gradient (g) and hessian matrix (H) in order to retrieve the necessary corrections (Δk) for estimating the robot pose. These corrections are computed using:

$$H\Delta k = -g \tag{14}$$

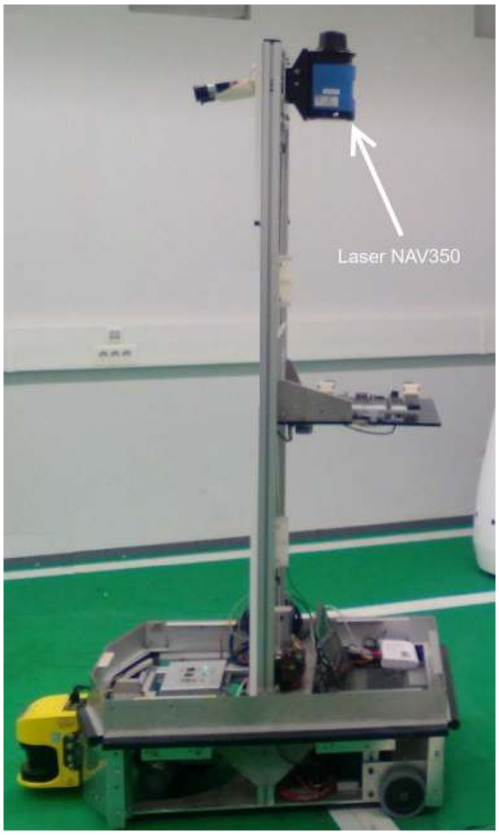

Fig. 1 Jarvis robot

3 Experimental Results

Having introduced the theoretical foundations of each of the three algorithms in the previous section, we will now present the experimental set-up (e.g. framework and simulator used, the algorithms parameterization, etc) and the results obtained in our experiments (the dataset can be found at[1]).

3.1 Framework description and evaluation tests

The ROS framework was used to perform the comparison of the three algorithms. Considering the simulation case, the Stage simulator was selected since it allows to model a virtual world where it can be introduced robots, sensors and objects. Besides the simulation, it is also presented results with a real robot platform (Jarvis). This robot is equipped with a commercial navigation laser (SICK NAV350), seen in Fig. 1.

3.2 Set-up of the Experiment

The initial parameterization of the algorithms for all the experiments was made as follows:

– All algorithms process the same sensor data.

[1]https://github.com/carlosmccosta/dynamic_robot_localization_tests

- All algorithms use the same reference map.
- For the evaluation of the three presented algorithms, it was ensured that the same stopping criteria was used for each evaluation metric. Considering the computational weight metric, we set as the stopping criteria to be the run of 50 iterations. This value was chosen since we verified experimentally that all matching algorithms converge to the final solution during this set of iterations. For the results on the converge speed, we defined a criteria that evaluated the position and orientation correction made by the algorithms between each iteration. When this correction is below a certain value the algorithm is considered to have converged. For the maximum initialization error metric, the same initial positions and orientations were considered for the three algorithms. Then, for each of these poses, the orientation error was incremented in each iteration and we stop the evaluation when the matching algorithm diverged to a bad matching solution. Finally, for the evaluation of the algorithms robustness to outliers, the same reference map and sensors data was used.
- All algorithms do not have access to data from odometry. Therefore the pose error is caused by the robots movement, traveling at 0.5 m/s.
- The version of the ICP available in PCL used in the tests was the "iterative_closest_point_2d" available in [4], without RANSAC for outlier rejection ("max_number_of_ransac iterations: 0"), and unless stated otherwise, the maximum distance search value was set to a high value for forcing the establishment of correspondences for all sensor data ("max_correspondence_distance: 9999.0"). Also, we set the parameter "use_reciprocal_correspondences: false" in all the tests performed.
- The NDT implementation used was the 3D version available in PCL.
- The computer used had a Intel Core i5 450M @ 2.40 GHz.

We choose to use the 3D implementation of NDT present in PCL (in relation to the 2D implementation also available in PCL) given that in preliminary tests it achieved superior robustness against initialization errors and sensor outliers while also being able to track the robot pose with higher accuracy. Moreover, the 2D implementation seems to have numerical / registration instability, which causes sudden loss of pose tracking even when the robot was performing only linear motions (the 3D implementation

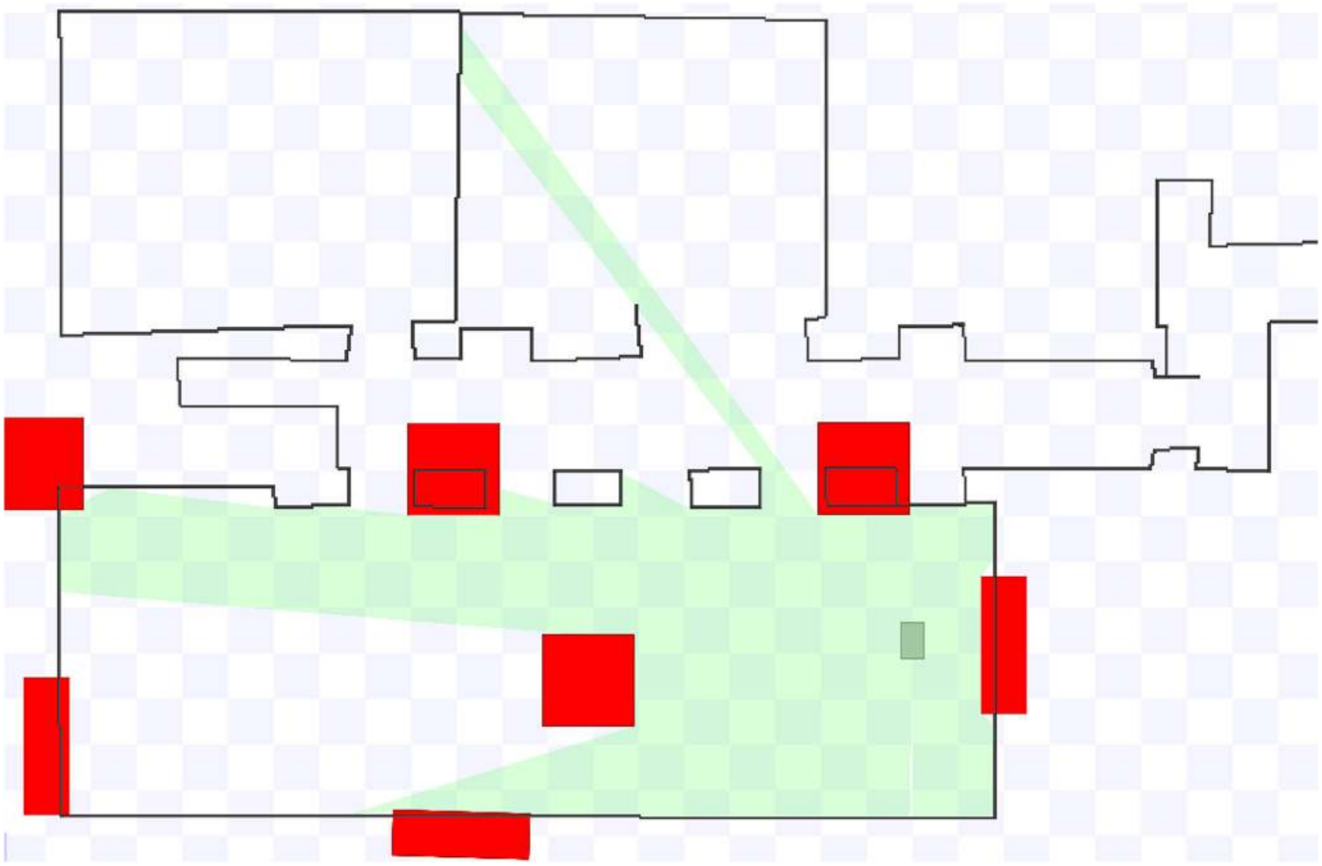

Fig. 2 Stage simulator with robot initial pose and unknown / not mapped objects presented as red rectangles. These objects introduce wrong measurements and occlusions in the LIDAR data

did not suffer from these issues). We acknowledge that the 3D implementation of NDT in PCL is about 4 times slower than the 2D implementation, but it should be noted that the 2D implementation uses 4 overlapping grids to avoid discretization problems, which requires to compute the 2D-NDT score 4 times more, one for each grid for each sensor point (the 3D implementation relies on a single voxel grid). Given the superior robustness, stability and accuracy of the 3D implementation of NDT in PCL (in relation to the 2D implementation), we believe that its usage in this article provides a more useful comparison for the robotics community that uses PCL.

Regarding the ICP parameterization it should be noted that in our previous article [14] we set "use_reciprocal_correspondences: true", which was causing the correspondence estimation stage to reject up to 40% of sensor data (a correspondence [map $\rightarrow$ sensor-data] needed to be the same as [sensor-data $\rightarrow$ map]). This significant rejection of correspondences was reducing the robustness of the ICP against rotation errors, and as such, in this article we set "use_reciprocal_correspondences: false" in order to avoid this issue.

3.3 Results on Computational Weight

The main goal of these tests was to evaluate the computational weight of the fundamental processing stages of the three map-matching algorithms. In order to achieve a more equitable performance evaluation of each method, the number of iterations was fixed to 50 and any auxiliary algorithm or filter that could influence the results was turned off. It was evaluated how the number of LIDAR points, the reference map cell size, and the presence of outliers in the sensor data (disposed as shown in Fig. 2) influenced the computational weight of the map-matching algorithms.

Analyzing Table 1 it is possible to verify that the Perfect Match is computationally lighter in all the tests performed. Furthermore, it can also be seen that each Perfect Match and NDT iteration is mainly affected by the amount of the robot sensors data, whereas in the case of ICP it is affected by all the variables that were analyzed (amount of the robot sensors data, resolution of the reference map and the presence of outliers in the robot sensors data). This higher computational demand required by the ICP in each of its iterations can be explained by the usage of a k-d tree to store the reference map (in the current PCL implementation of ICP) which makes the access slower and less deterministic (when compared to lookup tables used by the Perfect Match). The main advantage of a k-d tree is the optimization of memory required to store the information of the reference map, sacrificing the search time to access the data. The Perfect Match uses approximately three times more memory than each of the other algorithms (ICP and NDT), but for the case under review (localization of a robot in 2D space), the operating speed is an important factor and the requirements in terms of memory are less critical.

By analyzing the distribution of the time used in each of the stages of ICP, we concluded that more than 90% of the total time is spent in the search for correspondences in the k-d tree of the map. In order to reduce the computational weight of ICP in the current PCL implementation, we replaced the k-d tree search approach with a lookup table search method. Besides reducing the computation cost, this approach provides constant and deterministic retrieval of

Table 1 Computational time (in ms) taken by each algorithm to perform 50 iterations

		Map 5cm				Map 1cm			
		Points 288		Points 1440		Points 288		Points 1440	
		No Out	Outliers	No Out	Outliers	No Out	Outliers	No Out	Outliers
PM	Mean Time	1	1	5	5	1	2	6	6
	Max Time	5	3	13	12	6	5	15	15
	Min Time	<1	<1	2	2	<1	<1	3	3
ICP	Mean Time	29	32	114	125	38	49	126	181
	Max Time	43	47	145	175	64	76	157	267
	Min Time	22	23	103	109	29	37	115	145
LUT-ICP	Mean Time	6	6	19	19	6	6	19	20
	Max Time	14	14	31	40	14	16	32	33
	Min Time	3	3	13	13	3	3	13	14
NDT	Mean Time	72	52	335	309	60	69	339	310
	Max Time	94	85	386	355	77	93	471	376
	Min Time	60	43	299	231	52	46	301	228

For these tests were considered two different angular resolutions for the laser sensor. One which produced 288 points and another generated 1440 points per scan

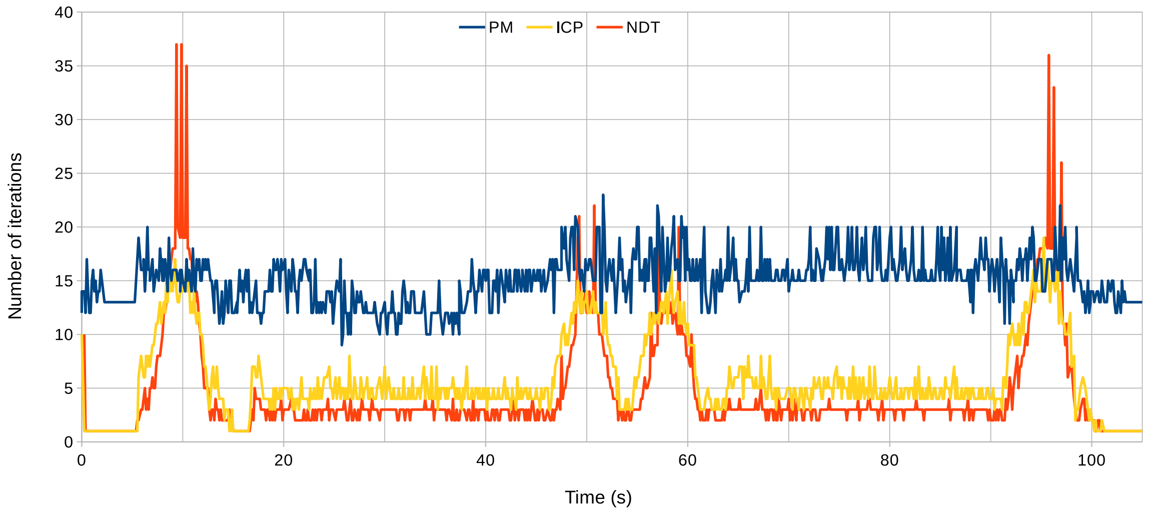

Fig. 3 Number of iterations performed by PM, ICP and NDT

the closest map point when a given sensor measurement is provided. This was achieved with the discretization of the environment into a 2D grid and the pre-computation of the closest map point for each of its cells. As such, when it is necessary to find the closest point to a given sensor measurement, instead of performing a computational intensive tree search, it is only required to use the sensor measurement 2D coordinates to compute the grid cell index in which the cached closest map point is stored and ready to be retrieved.

In Table 1 it is presented the computational weight results of this new proposed version of ICP (LUT-ICP), achieving performance of 5 to 9 times faster than the actual PCL implementation (relying on k-d trees). This new closest point search approach using a more deterministic approach reduced the correspondence estimation phase to a fast matrix access, boosting the ICP efficiency and cycle time while achieving computational times much closer to those of the PM algorithm. Although the operation speed was greatly optimized with this new implementation, the time needed to create the lookup table at the start of the program (using our implementation of the Meijster distance transform algorithm [9]) was slightly higher when compared with the k-d tree approach. However, given that this operation only needs to be calculated once for each map, its impact on the startup of the system is far outweighed by the performance gains that it provides during the runtime of a self-localization system. However, for SLAM applications in which it is required to dynamically update the map, it would be interesting to analyze how recomputing part of a lookup table compares in relation to rebuilding a k-d tree in terms of performance.

Considering the resolution of the reference map, from Table 1 it is possible to conclude that due to the NDT

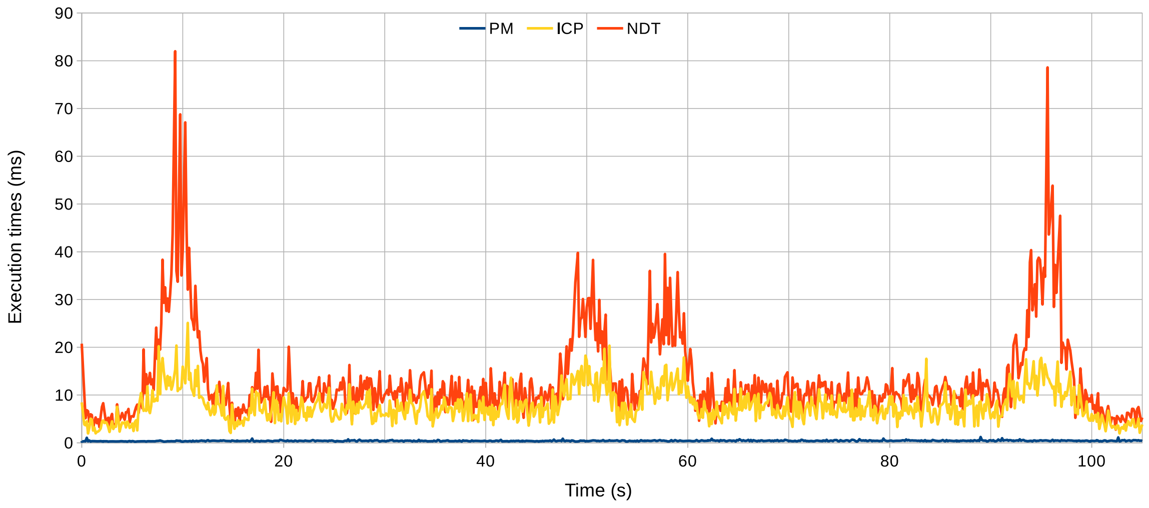

Fig. 4 Total execution time of PM, ICP and NDT

internal discretization of the environment as a set of normal distributions, the NDT algorithm is not affected by the map resolution. Moreover, it should be noted that the 3D implementation of NDT uses the Newton optimization method with a line search algorithm [10] in order to compute the optimization step length that guarantees sufficient decrease of the optimization score, which requires an internal loop of 10 iterations in which the NDT score must be computed. As such, the number of iterations reported in the tests requires the computation of the score by a factor of 10. It should also be noted that the 3D implementation of NDT in PCL uses a voxel grid for storing the normal distributions mean and covariance information and each cell of this voxel grid is indexed in a k-d tree in order to allow efficient radius search (required when computing the NDT score for each sensor point). Moreover, the 3D implementation of NDT in PCL does not include the extensions proposed in [7, 8], that would improve the robustness and accuracy of NDT at the cost of higher computational cost. The first extension introduces the iterative discretization add-on that runs the registration algorithm at successively finer cell resolution for higher pose estimation accuracy. The second improvement implements a trilinear interpolation approach that instead of using just one voxel grid over the 3D space, it relies on 8 overlapping grids for minimizing the discretization effects and have a smoother map representation. The last add-on proposes the concept of linked cells, in which every empty NDT cell stores a pointer to the closest cell (with covariance information), allowing the NDT algorithm to register point clouds with higher initial error of alignment.

3.4 Results on Convergence Speed

At this point we have concluded that each PM iteration, can be up to 72 times faster than a NDT iteration (considering the used PCL implementation) and up to 32 times faster than a ICP iteration. But this raises the question of convergence speed. Some algorithms may have a higher computational cost for each iteration, but require less iterations to converge to a good solution. In order to analyze the convergence speed, it was added another stopping criteria to the three map-matching algorithms. This criteria evaluates the position and orientation corrections made by the algorithms between two iterations. If they are below a certain value the optimization process stops. The parameters values used in this stopping criteria were 0.01m in translation and 0.8 degrees in rotation for all algorithms. The selected test scenario was the one with less density in robot sensor data, with the map with the lowest resolution and without outliers. Figures 3 and 4 present the results of the earlier described experiment. Figure 3 presents the number of iterations of PM (blue), ICP (yellow) and NDT (orange) over time, while the robot performs the path shown in Fig. 7. Figure 4 represents the PM (in blue), ICP (yellow) and NDT (orange) convergence time considering the same execution path. As can be seen in Fig. 3, the ICP and NDT algorithms need less iterations than the PM to perform map-matching. But this is not enough to be computationally lighter than Perfect Match. The NDT often converges in less iterations than PM due to the application of Newton's algorithm for minimization of the $-score$ (which uses first and second derivative information while PM only uses the first derivative) and also due to an adaptive step length

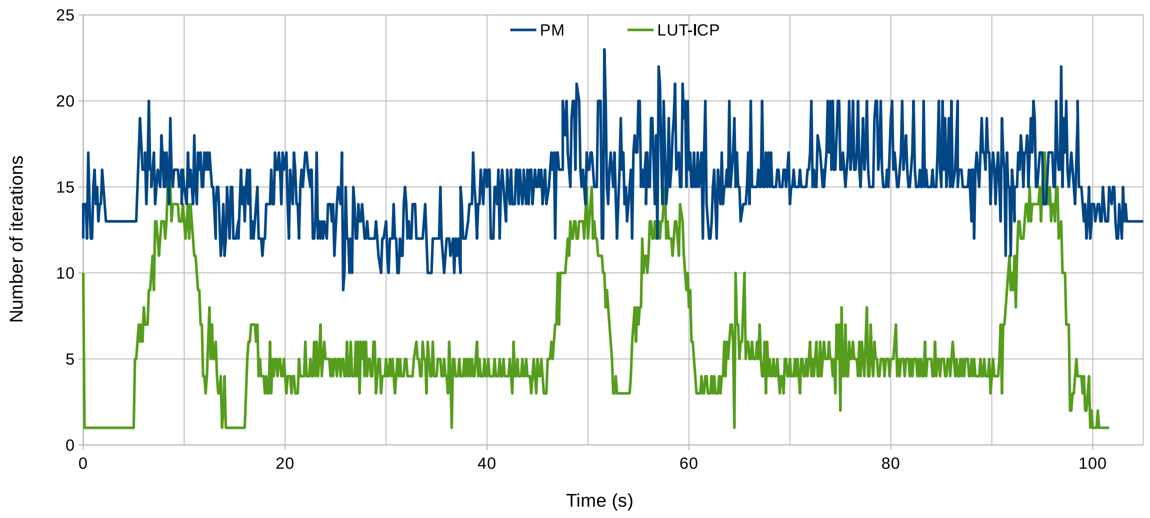

Fig. 5 Number of iterations performed by PM and LUT-ICP

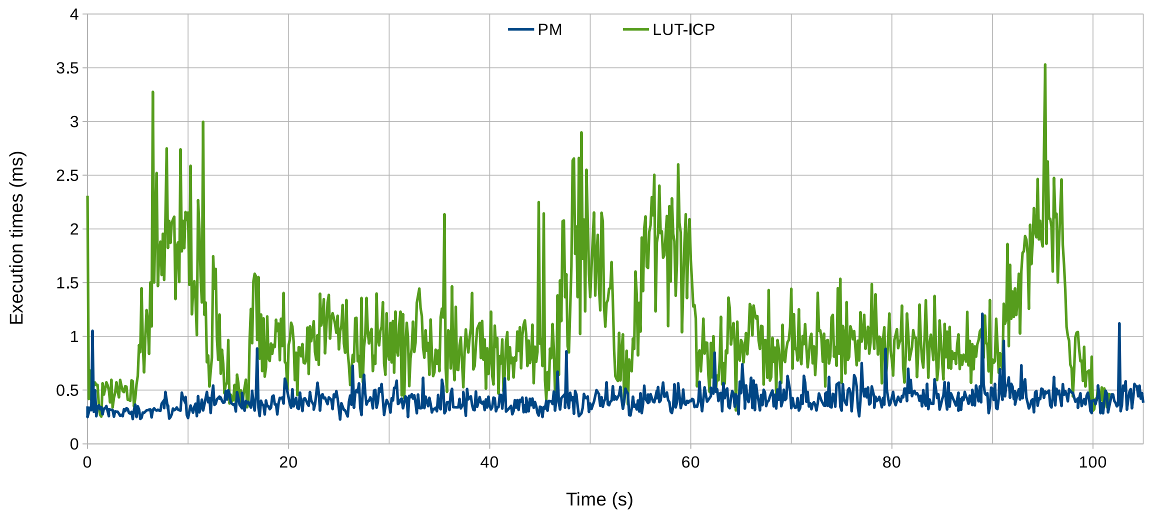

Fig. 6 Execution times of PM and LUT-ICP

computation within the Newton optimization to guarantee sufficient decrease of the score value.

In Fig. 3, it is also possible to identify three situations. A zone where ICP and NDT both perform only one iteration, which corresponds to the situation when the robot is stopped. A zone where ICP and NDT perform between two and eight iterations that corresponds to the situation where the robot is moving in a straight line. And another area where the number of iterations rises, even exceeding the PM number of iterations (mainly for the NDT case), corresponding to the situation when the robot is rotating. The above findings do not change if we repeat the experience for the scenario where there are outliers present in the map.

Figures 5 and 6 show a comparison between the Perfect Match algorithm and our ICP approach using lookup tables (LUT-ICP) regarding convergence speed.

As it can be seen in Fig. 5, the number of iterations performed by LUT-ICP are the same as the PCL implementation of ICP (Fig. 3), but the computational time used in each iteration is now much lower and similar to the PM's (Fig. 6). This new proposed approach does not negatively affect the ICP's performance and greatly reduces its cycle time, making it a more viable solution for real time operation.

3.5 Maximum Initialization Error

In the presented tests the Perfect Match has shown to be computationally lighter than ICP and NDT. In this section, we show test results for the three algorithms considering the initial localization error of the robot, in order to evaluate which algorithm is more robust to local minima. In these experiments it were used the same parameters of

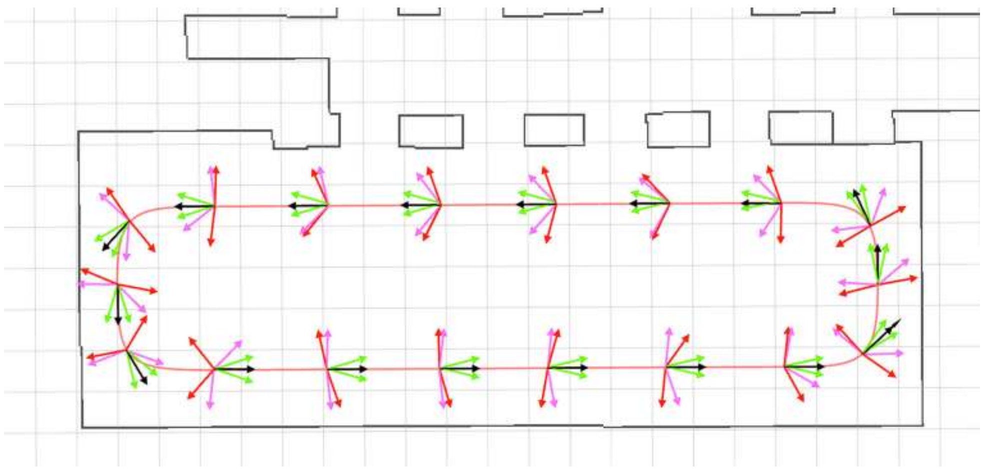

Fig. 7 Interval for the rotation initial error that the algorithms support and still converge to a good solution. In red results for the PM, in purple results for the ICP, and in green color results for NDT. The measured angle is overlapped with the robot pose ground truth represented by black arrows

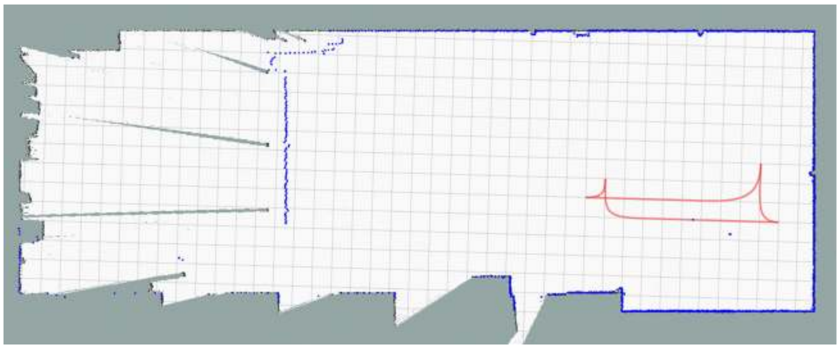

Fig. 8 Path (red line) performed by the Jarvis platform in a real environment, with a laser scan example (blue dots). In this laser scan it's possible to see the major obstacle of the environment (vertical blue dots forming a line, on the left side of the map)

Section 3.3, i.e, a fixed number of iterations was set, the maximum corresponding distance of ICP was set to a high value and all NDT filters were turned off. In order to see the orientation error tolerance of the analyzed algorithms we have stopped the robot in some points of the navigation trajectory and the three algorithms were reinitialized with an initial pose with a gradually greater orientation error (1 degree of resolution) until the matching algorithm diverges. In Fig. 7 it is presented the set of poses where these tests were performed (black arrows). Each of these poses have another three pairs of arrows (red, purple and green). These correspond, respectively, to the PM, ICP and NDT orientation error limits in which they still converge to a correct solution.

Analyzing the results in Fig. 7 it is clear that the PM, in the majority of cases, supports an initialization error for the rotation greater than both ICP and NDT.

One important aspect to refer is that the fact that the parameter "use_reciprocal_correspondences" was set to false had a crucial positive impact in the overall performance of the ICP algorithm, making it more robust. For this specific case, the maximum initialization error achieved by ICP was greatly improved (in relation to the results presented in [14]). These conclusions remain the same for the LUT-ICP, as the performance of the algorithm is only affected in terms of cycle

Also, preliminary tests were made regarding the position error, where we obtained similar results. We also performed the experience of setting the stopping criteria used in Section 3.4 but did not obtain better results. In addition, we also tested limiting the correspondence distance of ICP and PM. This test was performed in the start position of the path shown in Fig. 7 - middle of the right vertical segment. For this specific example and using Lc / max_correspondence_distance as 1m, the maximum initialization error of ICP is about 100 degrees as the PM's is 184. When decreasing the value of these parameters to 0.1m, the robustness to initialization errors in ICP drops in a higher factor than PM's, resulting in 13 and 62 degrees, respectively.

3.6 Evaluation of the Algorithms Robustness to Outliers

At this point we noted PM advantages in the computational weight. However, it raised the question of accuracy and robustness to outliers. There exists a large number of studies that address this issue. In particular the use of the RANSAC for identifying the presence of outliers in the sensor data. But many of these methods are transversal to all matching algorithms and so it can also be used in the PM algorithm. We are only interested for now in analyzing the core algorithms, and we want to examine whether the

Table 2 Statistic data of the position and orientation errors obtained in simulation (with outliers) and with the Jarvis platform in presence of natural outliers

	Simulation				Real dataset			
	Position error (m)		Orientation error (°)		Position error (m)		Orientation error (°)	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
PM	0.112	0.0715	0.333	0.207	0.0110	0.00310	0.0889	0.0521
ICP	0.110	0.0704	0.305	0.209	0.0120	0.00384	0.0448	0.0311
NDT	0.127	0.0819	0.378	0.325	0.00951	0.00357	0.0640	0.0428

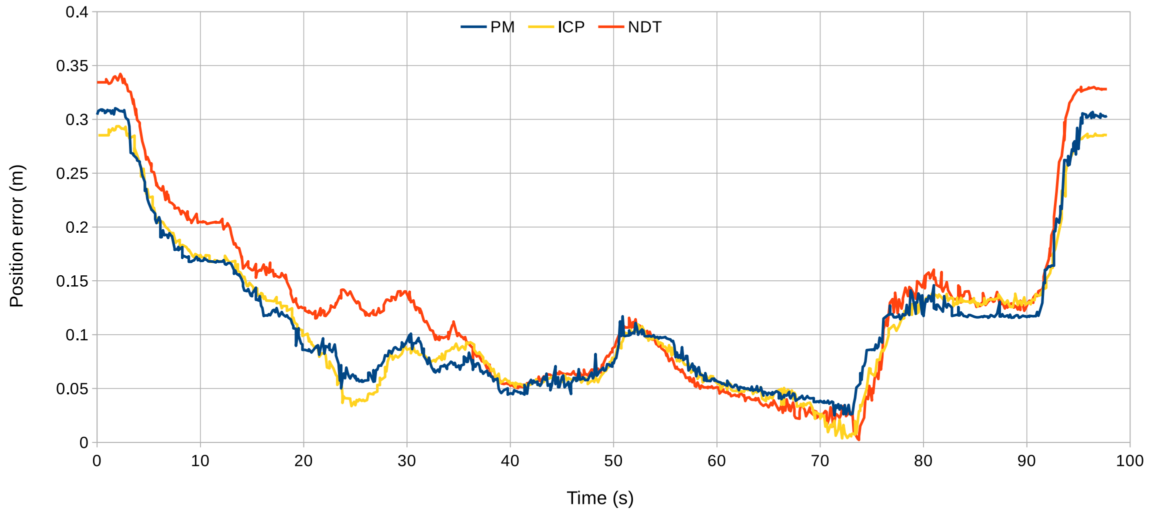

Fig. 9 Position error along the trajectory with outliers in simulation

PM computational efficiency is achieved by sacrificing accuracy/robustness to the result of matching. The PM has built in its optimization function a kind of outlier filter, tuned by the Lc parameter. In these experiments we used Lc = 0.1 meters. Moreover, in order to try to make this a more fair comparison we also changed the ICP parameter of the maximum correspondence distance to 0.1 meters. In relation to the NDT, the more related parameter is the ratio of expected outliers in the sensor data, which is set by default to 0.55.

As often simulators do not model important details that can make a significant difference in the performance of an algorithm, we also carried out the above experiments with a real robot equipped with a laser range finder with a reflective triangulation system installed on walls to serve as ground-truth. In this dataset the Jarvis robot was traveling at 0.05 m/s in a map with 20×8 m and performing the path shown in Fig. 8. We reduced the robot's velocity because a 8 Hz laser can have significant distortion when the robot is rotating fast, which would result in high error of localization. This problem can be mitigated by using sensors with high frequency motion estimation, such as wheels encoders, in order to compute the robot odometry that could be used to correct the laser distortion using spherical linear

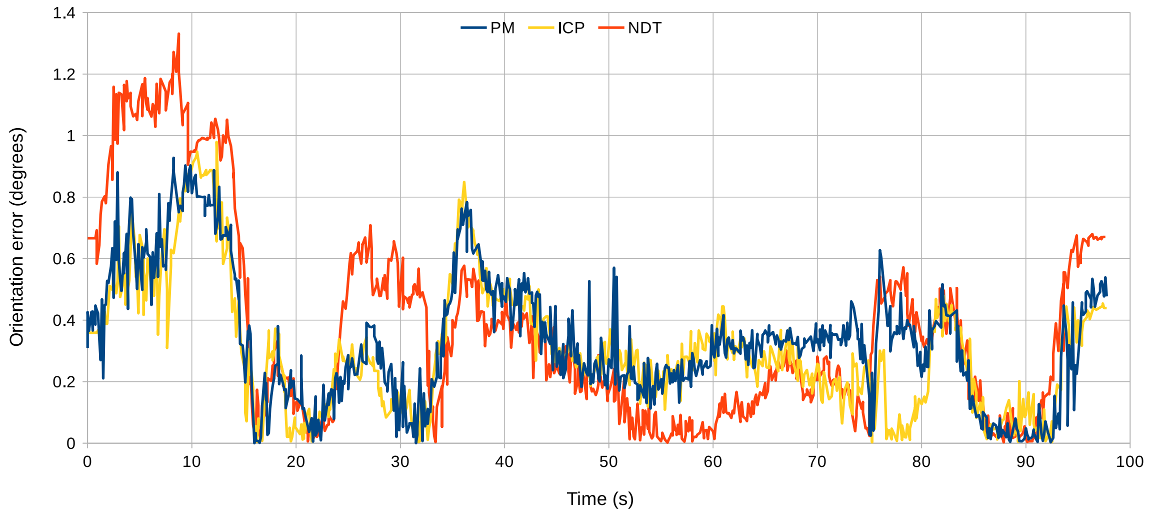

Fig. 10 Orientation error along the trajectory with outliers in simulation

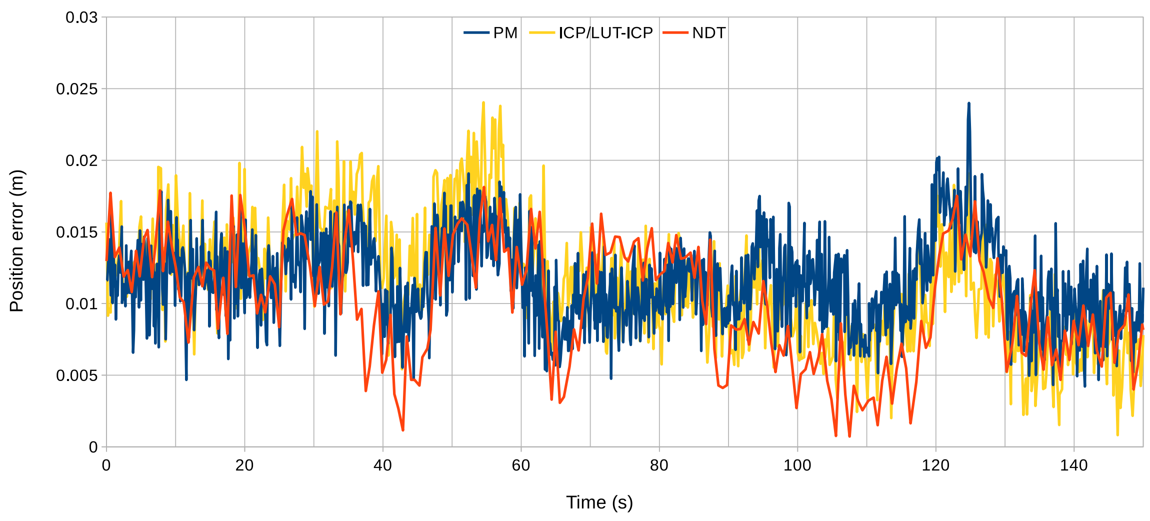

Fig. 11 Positional error along the trajectory using the Jarvis robot

interpolation. In order to illustrate the major sensor outliers of the robot environment, we also present a laser scan in Fig. 8 (laser measurements shown as blue dots).

Comparing the results presented in Table 2 we can conclude that all algorithms achieved similar results regarding position and orientation errors in the same circumstances and with equivalent parameterizations. With all the information to this point it is possible to conclude that the Perfect Match algorithm is equivalent to ICP and NDT in terms of accuracy and robustness but with a lower computational cost, leaving time for the application of more advanced filters in order to increase the efficiency of a localization system that relies on the PM. In this experiment, the LUT-ICP achieved similar results as the original ICP PCL's implementation presented in this section, which rivals the PM's computational weight advantage with similar accuracy and robustness results. Figures 9 and 10 present the precision and results for the PM, ICP and NDT in the simulated environment. While Figs. 11 and 12, present the same results but in a real environment with the natural presence of outliers, using the map and performing the path presented in Fig. 8.

4 Conclusions and Future Work

This paper presents a comparison between three well known map-matching algorithms, the Perfect Match, the Iterative Closest Point and the Normal Distribution Transform, considering four different evaluation metrics.

Starting with the computational weight, the Perfect Match has shown to be lighter than ICP and NDT. We

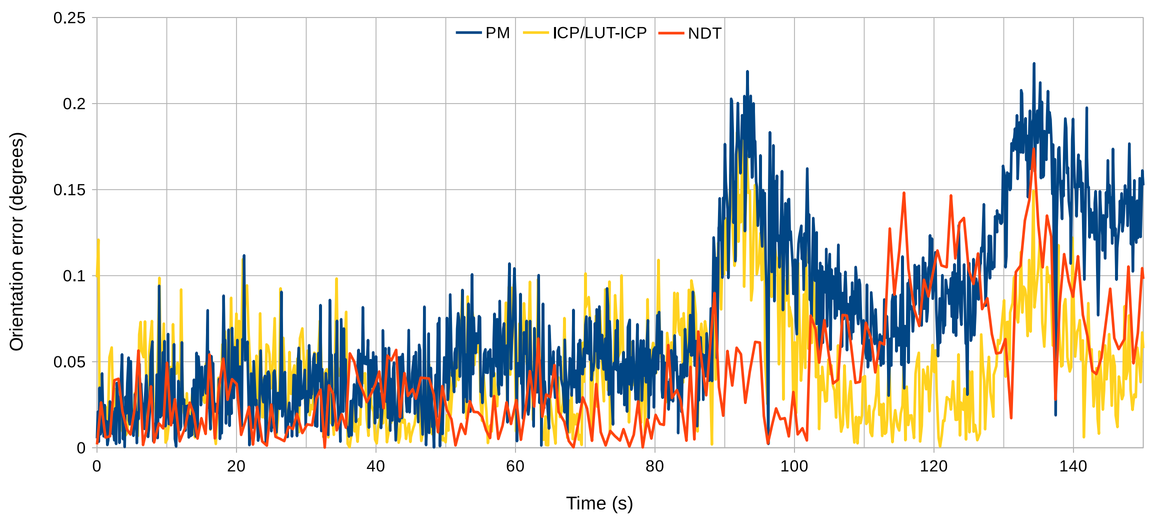

Fig. 12 Orientation error along the trajectory using the Jarvis robot

have concluded that this better performance could be mainly explained by the efficient PM correspondence search algorithm that is based on lookup tables. Thus, we have replaced the PCL ICP search approach based on a k-d tree with the a lookup table search method, which enabled us to greatly reduce the computational time of ICP, while maintaining the same matching performance (number of required iterations to converge, maximum initialization error, position and orientation errors in the presence of outliers). With this new search approach that we added to PCL, the LUT-ICP obtained closer results to those of Perfect Match, becoming a more viable solution for real time operation.

In terms of convergence speed, the ICP and NDT have shown that they require fewer iterations to converge when the robot is stationary or moving in a straight line. But their performance degrades significantly when rotations are made. In this case PM has shown, one more time, to be a viable option.

The analysis of the maximum initialization error was the most interesting result from the set of tests performed. The PM presented a higher tolerance to orientation errors than ICP and NDT. As future work we intend to refine this result in order to also include position error. However, some preliminary tests were made which showed a similar response of the three algorithms in terms of translation error. Furthermore, we should highlight the capacity of PM to handle outliers directly in the optimization function and support at the same time greater initialization error when compared with the other two analyzed algorithms.

In terms of accuracy, the three algorithms showed similar results, however ICP, NDT and the PM implementation handles the presence of outliers in different ways. The PM has contemplated in its cost function the possible presence of outliers. ICP handles outliers by trimming points that are farther than a given distance. NDT bounds the influence of outliers by computing the score as a mixture of a normal distribution and a uniform distribution in which we can specify the percentage of outliers that we are expecting to observe. One of the disadvantages of NDT is the discretization of the environment map into cells with a lower resolution than the original map. This approach introduces numerical errors that have a direct impact on the quality of the matching result. This issue takes special relevance when the map characteristic are not linear (such as the case in which the environment map has many curves and sharp edges), contributing to the growth of the matching error and consequently deteriorating the estimation of the AGV position (as high as 15 cm of translation error).

We acknowledge that there are a lot of other solutions in the state of art which could be included in this comparison (we only mentioned the ones implemented in PCL at the time). This study served to validate the relevance of the Perfect Match algorithm and we intend this to be the basis for future algorithm developments, improvements and comparisons.

Acknowledgments The research leading to these results has received funding from the European Union's Horizon 2020 - The EU Framework Programme for Research and Innovation 2014-2020, under grant agreement No. 688807. Project "TEC4Growth - Pervasive Intelligence, Enhancers and Proofs of Concept with Industrial Impact/NORTE-01-0145-FEDER-000020" is financed by the North Portugal Regional Operational. Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF). This work is also financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the FCT Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project POCI-01-0145-FEDER-006961. This work is financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the Portuguese funding agency, FCT -Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology), within project SAICTPAC/0034/2015-POCI-01-0145-FEDER-016418. The research leading to these results has received funding from the European Unions Horizon 2020 - The EU Framework Programme for Research and Innovation 20142020, under grant agreement No. 688807 ColRobot. P. C. M. A. Farias acknowledge support from CNPq/CsF PDE 233517/2014-6 for providing a scholarship.

References

1. Burguera, A., González, Y., Oliver, G.: Sonar scan matching by filtering scans using grids of normal distributions. Intell. Auton. Syst. **10**, 64–73 (2008)
2. Besl, P.J., McKay, H.D.: A method for registration of 3-D shapes. IEEE Trans. Pattern Anal. Mach. Intell. **14**(2), 239–256 (1992)
3. Biber, P., Strasser, W.: The normal distributions transform: a new approach to laser scan matching. In: EEE/RSJ International Conference on Intelligent Robots and Systems (IROS), vol. 3, pp. 2743–2748 (2003)
4. Costa, C.M., Sobreira, H.M., Sousa, A.J., Veiga, G.M.: Robust 3/6 DoF self-localization system with selective map update for mobile robot platforms. Robot. Auton. Syst. **76**, 113–140 (2016). ISSN 0921–8890
5. Lauer, M., Lange, S., Riedmiller, M.: Calculating the perfect match: an efficient and accurate approach for robot self-localization, Rob. 2005 Robot soccer world cup, no. c, pp. 142–153 (2006)
6. Magnusson, M., Lilienthal, A., Duckett, T.: Scan registration for autonomous mining vehicles using 3D-NDT. J. Field Rob. **24**, 803–827 (2007)
7. Magnusson, M.: The Three-Dimensional Normal-Distributions Transform - An Efficient Representation for Registration, Surface Analysis, and Loop Detection Örebro University (2009)
8. Magnusson, M., Nuchter, A., Lorken, C., Lilienthal, A.J., Hertzberg, J.: Evaluation of 3D registration reliability and speed - A comparison of ICP and NDT. In: IEEE international conference on robotics and automation (2009). ICRA 09
9. Meijster, A., Roerdink, J.B.T.M., Hesselink, W.H.: A general algorithm for computing distance transforms in linear time. In:

Mathematical morphology and its applications to image and signal processing, pp. 331–340 (2000)

10. Moré, J.J., Thuente, D.J.: Line search algorithms with guaranteed sufficient decrease. ACM Trans. Math. Softw. **20**, 286–307 (1994)
11. Pinto, M., Sobreira, H., Paulo Moreira, A., Mendonça, H., Matos, A.: Self-localisation of indoor mobile robots using multi-hypotheses and a matching algorithm. Mechatronics **23**(6), 727–737 (2013)
12. Schulze, L., Wullner, A.: The approach of automated guided vehicle systems. In: 2006 IEEE international conference on service operations and logistics, and informatics, pp. 522–527 (2006)
13. Schulze, L., Behling, S., Buhrs, S.: Automated guided vehicle systems: a driver for increased business performance. In: Proceedings of the international multiconference of engineers and computer scientists, pp. 19–21 (2008)
14. Sobreira, H.M., Rocha, L., Costa, C.M., Lima, J., Costa, P., Moreira, A.P.: 2D cloud template matching - a comparison between iterative closest point and perfect match. In: 2016 IEEE international conference on autonomous robot systems and competitions (2016)
15. Tomatis, N.: Bluebotics: navigation for the clever robot [Entrepreneur]. IEEE Robot. Autom. Mag. **18**(2), 14–16 (2011)

Héber Sobreira was born in Leiria, Portugal, in July 1985. He graduated with an M.Sc. degree (2009) and a Ph.D. degree (2017) in Electrical Engineering from the University of Porto. Since 2009, he has been developing his research within the Robotic and Intelligent Systems Unit of INESC-Porto (the Institute for Systems and Computer Engineering of Porto). His main research areas are navigation and control of indoor autonomous vehicles.

Carlos M. Costa received his MSc in Informatics and Computing Engineering at the Faculty of Engineering of the University of Porto in 2015 and is currently pursing a PhD at the same university at ProDEI while also researching at the Centre for Robotics in Industry and Intelligent Systems of INESC TEC. He has participated in several research projects dealing with the self-localization of autonomous vehicles (CARLoS and ColRobot) while also developing augmented reality and human machine interaction systems (SMErobotics CLARiSSA and CoopWeld). His current research areas include robotic assembly automation along with computer vision and 3D perception.

Ivo Sousa achieved B.Sc in Biomedical Engineering in 2013 from Higher Institute of Engineering of Coimbra (ISEC). He then extended studies and in 2016 completed an M.Sc degree in Electrical Engineering from the Faculty of Engineering of University of Porto (FEUP). In the present, he's working at the company TALUS Robotics in the development of innovative robotized solutions. His main research interest focus on mobile robotics (navigation and control).

Luis Rocha graduated with a MSc in Electrical and Computer Engineering at the Faculdade de Engenharia, Universidade do Porto, Portugal, in 2010 and received his PhD degree in Electrical and Computer Engineering in the same University in 2014. Since 2010 he is a researcher at the Centre for Robotics in Industry and Intelligent Systems of INESC TEC. His main research interests are focused in the flexibility enhancement of industrial robotic cells, as in terms of industrial manipulators programming procedure as on improving their perception skills. He has been involved in several FP7 and H2020 projects, such as CARLoS and ColRobot.

José Lima (male) received the M.Sc. and PhD in Electrical and Computer Engineering on Faculty of Engineering of University of Porto, Portugal in 2001 and 2009. He joined the Polytechnic Institute of Bragança in 2002, and currently he is a Professor in the Electrical Engineering Department of that school. He is also a senior researcher in Centre for Robotics in Industry and Intelligent Systems group of the INESC-TEC (Institute for Systems and Computer Engineering of Porto, Portugal). He has published more than 70 papers in international scientific journals and conference proceedings. In addition, he participated in some autonomous mobile robotics competitions and applications. Moreover, his research interests are in the field of robotics and automation: simulation, path planning, image processing, localization, navigation, obstacle avoidance and perception. He participated in some national, FP7 and H2020 funded projects such as Produtech, Grace, Arum, Carlos, Stamina and ColRobot.

P. C. M. A. Farias received the B.S. and M.Sc degrees in Electrical Engineering from the Federal University of Bahia (UFBA), in 1995 and 1999, respectively. He received his D.Sc. in Nuclear Engineering from the Federal University of Rio de Janeiro (COPPE/UFRJ), in 2006. He was a postdoctoral researcher at Faculty of Engineering of the University of Porto in 2016. He is currently a professor with the Department of Electrical and Computer Engineering, UFBA, Brazil. His research interests include robotics, high-energy physics, embedded digital signal processing and nondestructive testing.

Paulo Costa graduated with a degree in electrical engineering at the University of Oporto, in 1991. Then, he pursued graduate studies at University of Porto, obtaining a M.Sc. degree in Electrical Engineering Systems in 1995 and a Ph.D. degree in Electrical Engineering in 2000. Presently, he is Assistant Professor at the Faculty of Engineering of the University of Porto and researcher in the Robotics and Intelligent Systems Centre at INESC TEC. His main research interests are robotics, Simulators and Machine Vision.

A. Paulo Moreira graduated with a degree in electrical engineering at the University of Oporto, in 1986. Then, he pursued graduate studies at University of Porto, obtaining a M.Sc. degree in electrical engineering – systems in 1991 and a Ph.D. degree in electrical engineering in 1998. Presently, he is Associate Professor with tenure at the Faculty of Engineering of the University of Porto and researcher and manager of the Robotics and Intelligent Systems Centre at INESC TEC. His main research interests are process control and robotics.