High-accuracy vehicle localization for autonomous warehousing

Goran Vasiljević, Damjan Miklić*, Ivica Draganjac*, Zdenko Kovačić*

LARICS Laboratory, Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia

Paolo Lista*

Euroimpianti S.r.l., Schio (VI), Italy

Abstract

The research presented in this paper aims to bridge the gap between the latest scientific advances in autonomous vehicle localization and the industrial state of the art in autonomous warehousing. Notwithstanding great scientific progress in the past decades, industrial autonomous warehousing systems still rely on external infrastructure for obtaining their precise location. This approach increases warehouse installation costs and decreases system reliability, as it is sensitive to measurement outliers and the external localization infrastructure can get dirty or damaged. Several approaches, well studied in scientific literature, are capable of determining vehicle position based only on information provided by on board sensors, most commonly wheel encoders and laser scanners. However, scientific results published to date either do not provide sufficient accuracy for industrial applications, or have not been extensively tested in realistic, industrial-like operating conditions. In this paper, we combine several well established algorithms into a high-precision localization pipeline, capable of computing the pose of an autonomous forklift to sub-centimeter precision. The algorithms use only odometry information from wheel encoders and range readings from an on board laser scanner. The effectiveness of the proposed solution is evaluated by

*Corresponding author
Email address: damjan.miklic@fer.hr (Damjan Miklić)

an extensive experiment that lasted for several days, and was performed in a realistic industrial-like environment.
Keywords: high-accuracy localization, autonomous warehousing, autonomous ground vehicle

1. Introduction

Notwithstanding the recent very successful examples of large scale use of autonomous mobile delivery vehicles such as Kiva Systems [1], today's manufacturing and logistics facilities are still largely dependent on manually operated vehicles [2]. Although numerous commercial solutions using autonomous ground vehicles (AGVs) do exist, e.g., automated material handling vehicles by Swisslog[1] and Euroimpianti[2] or Komatsu's autonomous haul systems for the mining industry[3], significant improvements can be made in terms of their level of autonomy and deployment cost. In particular, the industrial state of the art for vehicle localization requires additional infrastructure, usually in the form of reflective markers or electromagnetic guides, for accurately determining vehicle pose [3]. This approach suffers from numerous disadvantages, as the markers have a high installation cost, requiring many man-hours of work by qualified personnel, they are sensitive to false positive readings and to changes in the environment which may obstruct the field of view of the vehicle, and they can get damaged or dirty in harsh industrial environments. Relevant contemporary literature, such as [4, 5, 6] unanimously identifies the ability of AGVs to self-localize without additional infrastructure as one of the key technologies that will increase their performance and flexibility, thus enabling their widespread use.

Because self-localization is a basic prerequisite for autonomous vehicle operation, this subject has been receiving significant scientific attention from the

[1] http://www.swisslog.com/en/Solutions/HCS/Material-Handling-Automation
[2] http://www.skilledrobots.com/products/agv
[3] http://www.komatsu.com.au/AboutKomatsu/Technology/Pages/AHS.aspx

earliest beginnings of mobile robotics. Different sensors have been used for this purpose, the most common ones being wheel encoders, contact switches, sonar arrays, 2D and 3D laser scanners, mono, stereo and RGBD cameras. A nice overview of relevant references for the different approaches can be found in [7]. For localization in the plane, i.e. on flat terrain, 2D laser scanners have been shown to provide the best accuracy, robustness and speed [8]. Additionally, the vast majority of AGVs deployed today is already equipped with laser scanners for safety and localization (using artificial landmarks) purposes, so we are focusing on this type of sensor throughout the rest of the paper. Advances in computational power at the turn of the century have made new classes of probabilistic methods computationally feasible. Since then, the AMCL[4] method [9] has established itself as the de facto standard in academic research. The method is based on a particle filter which fuses odometry information provided by wheel encoders with laser scanner range readings to provide robust localization with reported accuracy between 0.05m and 0.1m [10, 11]. Because it is a multi-hypothesis method, it is able to deal with situations when the robot is temporarily "lost". These features make the method suitable for most academic research work in mobile robotics. Its effectiveness has led researchers to consider robot localization in the plane to be a "solved problem" and move on to other research topics, such as localization in 3D space. However, the accuracy required by material handling applications is typically within 0.01m and 0.5°, so the industry has continued relying on additional infrastructure to ensure the required accuracy.

In the past five years, the prospect of widespread industrial adoption of autonomous vehicles, with enhanced performance and flexibility and reduced deployment costs, has spurred vigorous research activity related to precise localization without artificial landmarks. A popular approach for obtaining high precision while maintaining robustness is to combine AMCL with scan matching. Estimates obtained by AMCL are refined by matching the laser readings

[4]Adaptive Monte Carlo localization

to map features. This approach is used very successfully in [7]. The authors performed extensive experiments with a holonomic vehicle in a laboratory environment, using a vision-based tracking system for evaluating the results with millimetre accuracy. In a static environment, they were able to achieve localization and positioning errors at taught-in locations well below 0.005m and 0.2°. Slightly higher, but still sub-centimetre errors were reported in the presence of dynamic obstacles (people walking by). In a similar setup, tracking accuracy of taught-in trajectories was examined [12]. By manually teaching-in the trajectories, the authors avoid using a global map. Point-to-line scan matching [13] is used for trajectory tracking in the robot's local coordinate system. Accuracy was evaluated by measuring the minimal distance to the reference trajectory and milimeter accuracy is reported. It should be noted that the experimental environment was small and completely static. The work presented in [14] focuses on seamless transition between marker based, map based and pure SLAM (localization without an a priori map) scenarios. The reported preliminary evaluation in a large scale warehouse, on a single trajectory, had errors within 0.05m and 0.5° most of the time. In their later work, the same authors examine the accuracy of 2D localization as a prerequisite for 3D mapping [15], in extensive experiments in a large scale warehouse. They do not use odometry information from wheel encoders. Instead they use Point-to-Line Iterative Closest Point (P-L-ICP) [13] to simulate odometry. They do not use the localization estimate as feedback for the position controller (they rely on the artificial landmarks for control) and report a mean positioning error of 0.052m and 0.012°. Another very promising approach is based on the Normal distributions transform (NDT) introduced by [16]. The NDT is a piecewise continuous representation, which represents the space as a set of normal distributions. It enables a more compact and more accurate representation of the environment, compared to grid-based representations. Authors in [17] formulate Monte Carlo localization (MCL) using the Normal distributions transform (NDT) as underlying representation for map and data. They evaluate their approach using offline data sets, in closed loop with a smaller AGV in a laboratory environment and in a

real warehouse with an industrial AGV in open loop (localization estimate is not used as control feedback for the vehicle). They achieve localization accuracy of 0.014m and 0.074° in laboratory conditions and below 0.03m in the industrial setting. The NDT-MCL algorithm is extended in [18] to dual-timescales, i.e., a dynamically updated short-term map is used for localization in addition to the a priori provided static map. This approach improves localization performance in highly dynamic environments, reducing localization errors below 0.02m in a laboratory setting. In our previous work [19], we evaluated an approach which fused readings from several laser scanners mounted at different heights. The average localization error was below 0.06m, while the maximum error was kept within 0.1m at all times during an experiment in an industrial setting, which does not quite satisfy industrial requirements.

The approach pursued in this paper combines AMCL, scan matching and Discrete Fourier Transform (DFT)-based pose estimate refinement into one algorithm stack for high-precision localization in industrial indoor environments. We describe all components of the localization algorithm and provide extensive experimental results, which confirm the accuracy, robustness and reliability of our approach. The experiments have been performed in an industrial warehouse setting, with a full sized autonomous forklift. The localization module is working in closed loop with the vehicle control module, so any significant localization error would cause a failure in path execution. During three days and 19 hours of total travel time, the vehicle has logged over eight kilometers, relying only on map information and its sensor readings, without a single failure or operator intervention. The results are evaluated using a methodology similar to [7]. Because vehicle positioning is most critical at docking stations, when picking up and delivering pallets, we defined 6 docking stations in the warehouse layout. The vehicle was repeatedly visiting the docking stations and these positions were used for evaluating localization accuracy. Making a fair comparison with results from other authors is difficult, mainly because of a lack of standardization. Some attempts at standardization are beeing made [20, 21], however, these are currently not applicable to industrial systems. In state of the art re-

search mentioned above, the vehicles, experimental environment and evaluation methods vary vastly. Experimental platforms range from holonomic laboratory robots to non holonomic industrial forklifts weighting several tons. Some experiments are performed in laboratory conditions, while others are performed in actual in-production warehouses. Different evaluation methods include absolute errors with respect to a reference localization system, relative errors computed with respect to taught-in poses, static errors, dynamic errors or Absolute Trajectory Errors (ATE) [20]. Notwithstanding the lack of standardized methods for comparison, to the best of our knowledge, no localization algorithm to date has been evaluated in comparably realistic industrial conditions which provides better localization accuracy at taught-in docking positions. Therefore, the main contribution of this paper is a rigorously validated AGV localization algorithm, which satisfies demanding industrial requirements and provides an improvement over the state of the art. Furthermore, in addition to the localization algorithm, we also describe the path planning and tracking algorithms that have been implemented on the vehicle and used for the experiment. Together with the localization module, they constitute a complete, experimentally verified positioning solution for autonomous warehousing.

The paper is organized as follows. The focal point of the paper is Section 2, where we provide a detailed description of our localization algorithm stack. In Section 3 we describe the industrial platform and the environment in which we performed our experiments. The positioning algorithm used for experimental validation is described in Section 4. The results of extensive experimental validation are described in Section 5. Concluding remarks and an outline of our future work plans are provided in Section 6.

2. The localization algorithm stack

In this section, we present the main contribution of our work, a localization algorithm stack capable of providing sub-centimeter localization accuracy in industrial environments, in real time. It consists of three algorithms which are

executed sequentially, each one providing a better and more accurate vehicle pose estimate. The described algorithm stack is general, in the sense that it makes minimal assumptions on the properties of the vehicle and of the environment. No assumptions are made regarding the vehicle kinematics, shape or size, or the layout of the environment. We assume that the vehicle is moving on flat terrain and that an occupancy grid map of the environment is available a priori. Furthermore, we assume that the vehicle is equipped with wheel encoders and a laser range scanner. These assumptions are valid in the vast majority of industrial warehousing scenarios.

2.1. Localization overview

A high-level overview of the localization system is depicted in Figure 1. The first step is using the robust AMCL (Adaptive Monte Carlo Localization) algorithm [9], which fuses odometry data with laser range readings to provide a robot pose estimate with a known covariance. That result is used as the initial estimate for the scan matching ICP algorithm [13], similarly to the approach in [7]. Finally, the obtained result, which typically has a very good orientation estimate, is used as the initial estimate in a discrete Fourier transform method, which returns the final result. These steps, along with the input data used at each step, are shown in Figure 2.

2.2. Map of the environment

In order to localize itself in the testing area, a mobile robot must have a known map of its environment. In this work we use the `gmapping` ROS package, which implements the algorithm described in [22]. The method features a highly efficient Rao-Blackwellized particle filter for learning grid maps from laser range data. In our experiments, we used a map built with a constant resolution of 5cm (shown in Figure 8). Even though the size of a single map cell is 5 cm, it is possible to achieve much better localization accuracy, because the localization algorithms use laser distance measurements to hundreds of points from different directions, which are implicitly averaged to return the location of the vehicle with an accuracy much better than the 5cm resolution of map.

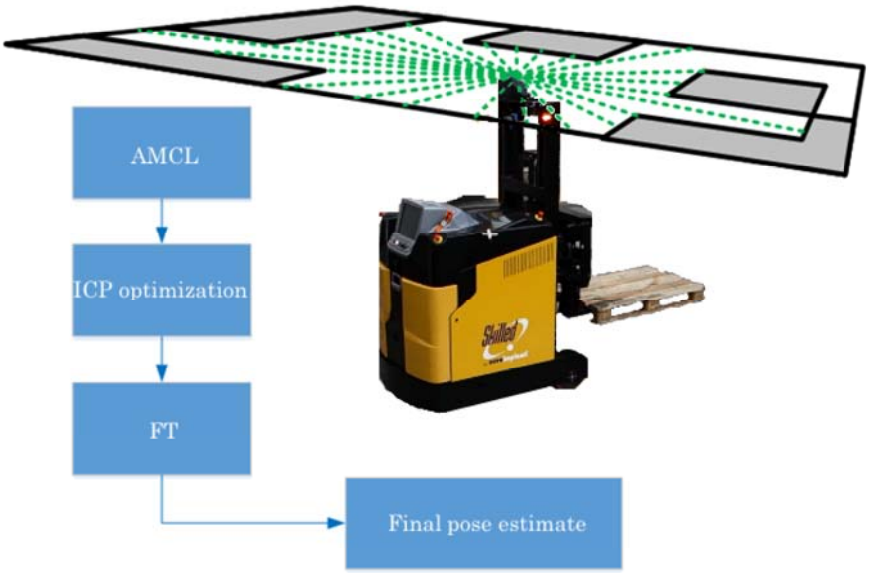


Figure 1: An overview of the proposed localization system. It is an algorithm stack consisting of three steps: Adaptive Monte Carlo Localization, Iterative Closest Point optimization and a Fourier Transform-based position refinement, yielding the final pose estimate.

2.3. Adaptive Monte Carlo Localization

We use Adaptive Monte Carlo Localization (AMCL) algorithm as described in [9] and implemented in the `AMCL` ROS package. The method is also known as Kullback-Leibler Distance (KLD) sampling for particle filters. This method can be applied to any mobile robot moving in the 2-dimensional space, providing odometry and range sensor measurements (laser, sonar, etc.).

Each step of the AMCL method is composed of two parts: motion update and sensor update. When the robot moves, odometry data is used to predict its next position. Because of uncertainties in odometry measurements, the estimated position is represented by a number of particles distributed around the estimated position depending on the measurement uncertainty. The number of particles varies depending on their density. Distances measured by the range scanner are used to calculate the probability that a robot is at a position represented by each of the particles generated by the motion prediction. Laser sensor calibration is

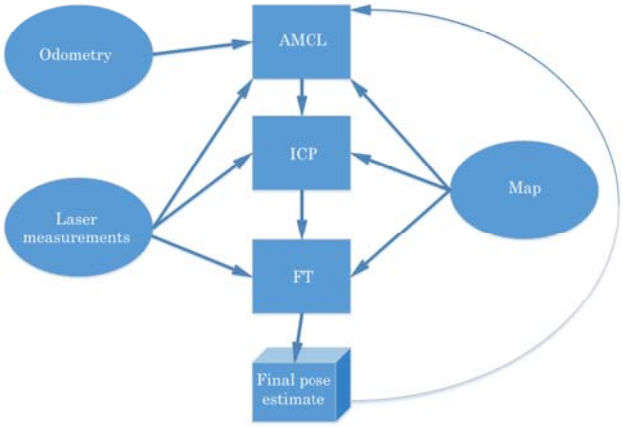


Figure 2: A view of the localization system structure showing the data used by each particular component of the localization stack. The initial AMCL step fuses odometry and laser readings, matched against a 2D map of the environment, in order to provide a rough pose estimate, based on the previous estimate. The ICP performs high-precision matching of the laser measurements against the 2D map, providing very accurate orientation and position estimates. Small jitters in the position estimate are finally filtered by the Fourier Transform-based step. The final, highly accurate pose estimate is fed back to the AMCL algorithm.

a prerequisite for highly accurate pose estimation.

The DFT result, if found, is fed back to the AMCL by updating its pose estimate periodically. In the case when the DFT result is not found (vehicle is lost), AMCL works independently until the robot find its approximate location, and the DFT returns a result. In this case AMCL guarantees are kept.

2.4. Iterative closest point

Iterative closest point (ICP) is well known algorithm, extensively used in computer vision. The implementation used within this paper is the one presented in [13] using point to line metrics. The algorithm uses two point clouds and tries to find the transformation that will best fit one to the other using a certain type of metrics. The first point cloud is the one acquired using laser range scanner:

$$\begin{aligned} x_{mi} &= d_{mi} \cdot \cos(\alpha_{min} + \Delta \cdot i) \\ y_{mi} &= d_{mi} \cdot \sin(\alpha_{min} + \Delta \cdot i) \end{aligned} \tag{1}$$

where x_{mi} and y_{mi} are the coordinates of the i-th point, d_{mi} is the i-th distance returned by range scanner, α_{min} is the angular offset of the first beam w.r.t. the orientation of the laser range finder and Δ is the angular distance between two beams. The second data cloud represents virtual measurements acquired from the map and the current estimate of the location:

$$\begin{aligned} x_{oi} &= x_e + d_{vi} \cdot \cos(\alpha_{min} + \Delta \cdot i + \theta_e) \\ y_{oi} &= y_e + d_{vi} \cdot \sin(\alpha_{min} + \Delta \cdot i + \theta_e) \end{aligned} \tag{2}$$

where d_{vi} is the minimal positive value for which the map position (x_{oi}, y_{oi}) is occupied, x_e, y_e and θ_e are the estimated position and orientation of the vehicle respectively. d_{vi} represents virtual measurements, and to get the second data cloud, the following equations are used:

$$\begin{aligned} x_{vi} &= d_{vi} \cdot \cos(\alpha_{min} + \Delta \cdot i) \\ y_{vi} &= d_{vi} \cdot \sin(\alpha_{min} + \Delta \cdot i) \end{aligned} \tag{3}$$

where x_{vi} and y_{vi} represent the i-th virtual measurement data cloud point.

The ICP algorithm represents an efficient method of fitting the two data clouds as best as possible. Since, in reality, there is a difference between the map and real working area, in order to reduce possible errors, some of the points in the point cloud are discarded if they do not satisfy the following conditions:

$$|d_{mi} - d_{vi}| < \epsilon \tag{4}$$

The choice of ϵ is crucial for the correct working of the algorithm because it determines the maximal allowed error between the estimated and the real position of the vehicle, but on the other hand it can have an influence on the precision of the cloud fitting in situations when there are small differences between the map and the working area. The solution is to have an adaptive ϵ that is reduced during different iterations of the ICP algorithm, since every iteration of the algorithm should theoretically reduce the position error. Unlike the outlier rejection method presented in [13], it is not necessary to define the exact number of points to be used for ICP, which could in many cases improve the accuracy of the pose estimate.

The number of cloud points that satisfy (4) is a measure that determines the success of scan matching, and in case when the number of matching points is too small, it is considered that localization is unsuccessful, and the obtained result is discarded.

2.5. Localization result improvement using Discrete Fourier Transform

The results obtained using AMCL with ICP are much more precise than the ones obtained using only AMCL, with very good orientation estimate, but a slightly unstable estimate of position. To stabilize the position estimate, a method based on the Discrete Fourier Transform is used, which assumes a very accurate orientation estimate.

The basic idea is illustrated by Figure 3, under the assumption that the position of the vehicle is in the middle of a perfectly round room, as shown in Figure 3a. If the initial estimate of the vehicle is moved toward the upper right, as shown in Figure 3b, then the $N = 1440$ distance measurements of d_{mi} and d_{vi} have shapes as depicted in Figure 3c. The difference $d_{mi} - d_{vi}$ has sinusoidal shape with a period of 1440 where the amplitude of the sinusoid represents the offset of the real position with respect to the estimated position and the phase offset represents the direction. The amplitude and the phase of the sinusoidal function with the period equal to the number of data points can actually be obtained as the first element of the discrete Fourier transform.

The first element of the discrete Fourier transform of $d_{mi} - d_{vi}$ is:

$$X_1 = \sum_{n=0}^{N-1} (d_{mn} - d_{vn}) \cdot e^{-2 \cdot \pi \cdot i \cdot n/N} \tag{5}$$

$$= \sum_{n=0}^{N-1} (d_{mn} \cdot e^{-2 \cdot \pi \cdot i \cdot n/N} - d_{vn} \cdot e^{-2 \cdot \pi \cdot i \cdot n/N}) \tag{6}$$

Using equations (1) and (3), under the assumption that the scan range is 2π

and that the starting angle is $-\pi$, the following equations stand:

$$\begin{aligned}
&\Delta = 2 \cdot \pi / N \\
&\alpha_{min} = -\pi \\
&x_{mn} = d_{mn} \cdot \cos(-\pi + 2\pi n/N) = -d_{mn} \cdot \cos(2\pi n/N) \\
&y_{mn} = d_{mn} \cdot \sin(-\pi + 2\pi n/N) = -d_{mn} \cdot \sin(2\pi n/N) \\
&x_{vn} = d_{vi} \cdot \cos(-\pi + 2\pi n/N) = -d_{vi} \cdot \cos(2\pi n/N) \\
&y_{vn} = d_{vi} \cdot \sin(-\pi + 2\pi n/N) = -d_{vi} \cdot \sin(2\pi n/N)
\end{aligned} \tag{7}$$

From equations (7) the following equations can be derived:

$$d_{vn} e^{-2\pi i n/N} = d_{vn} \cos(2\pi n/N) - i \cdot d_{vn} \sin(2\pi n/N) \tag{8}$$

$$= -x_{vn} + i y_{vn} \tag{9}$$

$$d_{mn} e^{-2\pi i n/N} = d_{mn} \cos(2\pi n/N) - i \cdot d_{mn} \sin(2\pi n/N) \tag{10}$$

$$= -x_{mn} + i y_{mn} \tag{11}$$

then equation (6) becomes:

$$\begin{aligned}
X_1 \quad &= \textstyle\sum_{n=0}^{N-1} \left(x_m(n) + i \cdot y_m(n)\right) \\
&- \textstyle\sum_{n=0}^{N-1} \left(x_v(n) + i \cdot y_v(n)\right)
\end{aligned} \tag{12}$$

Under the assumption that the distance between the real and assumed position is small compared to the measured area, the following stands:

$$\begin{aligned}
\textstyle\sum_{n=0}^{N-1} \left(x_v(n) + i \cdot y_v(n)\right) \approx \quad &\textstyle\sum_{n=0}^{N-1} \left(x_m(n) + i \cdot y_m(n)\right) \\
&+ N \cdot x_{err} + N \cdot y_{err} \cdot i
\end{aligned} \tag{13}$$

where x_{err} and y_{err} are offsets from the virtual to the real position of the vehicle. From equations (12) and (13), it follows:

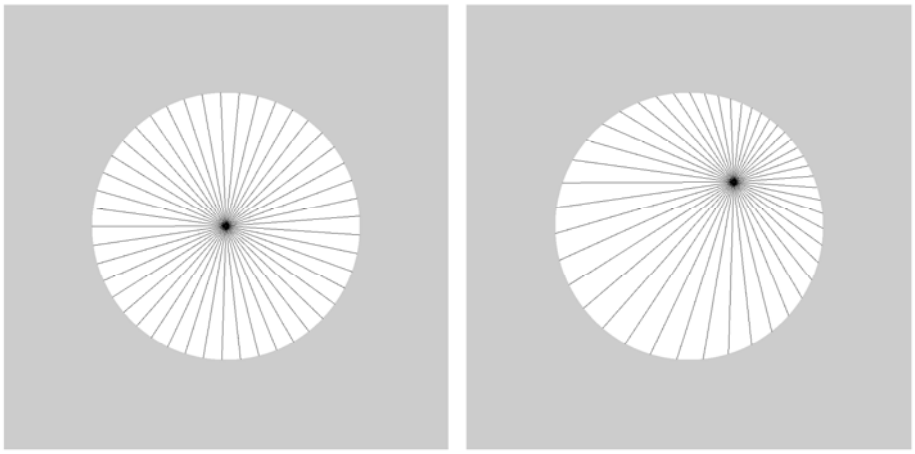
$$X_1 \approx -N \cdot x_{err} - N \cdot y_{err} \cdot i \tag{14}$$

From the discussion above, we conclude that the calculation of the first element of the Discrete Fourier Transform returns the position offset from the estimated to the real position. Since there is a difference between real and virtual measurements, only measurements that satisfy the condition (4) are taken into

account. The procedure is iterated several times in order to achieve sufficient precision of the localization estimate. Figure 4 demonstrates the effectiveness of the method in the rectangular room. In the event when the estimated position is moved for 5m in x and y direction, the amplitude and phase of a discrete Fourier transform are shown in Figure 4e, and show that the offset from estimated to the real position of the vehicle corresponds to the amplitude of the first element in DFT ($\sqrt{5^2+5^2}$, 2.36 rad = 135°). The precision reached after three iterations is shown in Table 1.

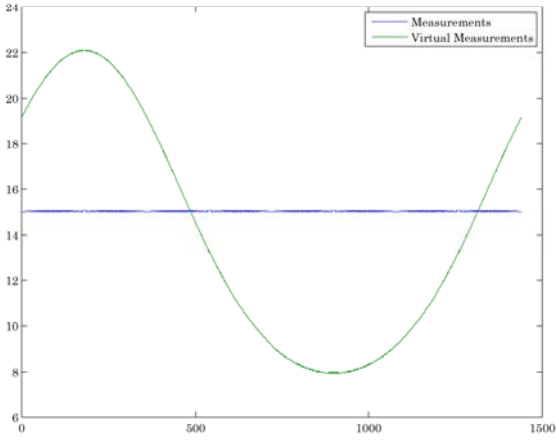
Table 1: Position error by algorithm iteration, for the example depicted in Figure 4.

Step	err_x/m	err_y/m
0	5	5
1	0.11843	0.1183
2	0.0003	0.0003



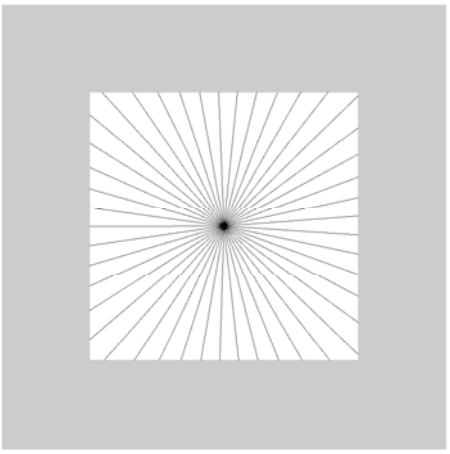
(a) The "real" scan, obtained by a sensor placed in the center of a perfectly round room.

(b) The "virtual" scan, which would be obtained under the (false) assumption that the sensor is displaced from the room center.

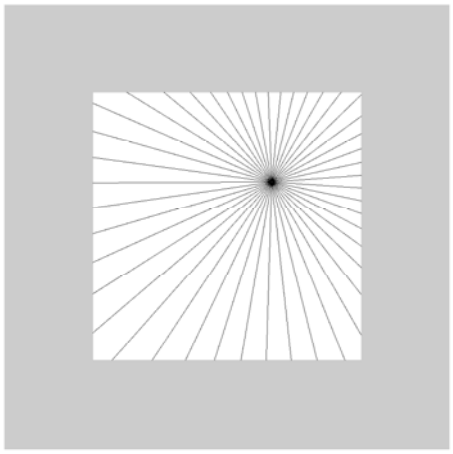


(c) Graphs of the "real" and "virtual" measurements, i.e., scans from (a) and (b). The difference between the two scans has the shape of the green graph.

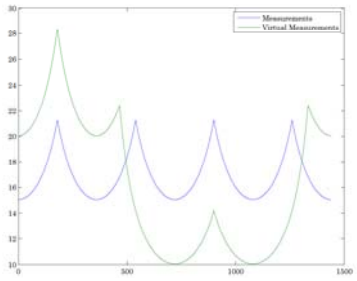
Figure 3: An illustrative example of the DFT-based localization estimate improvement, in a circular workspace. By examining the amplitude and phase of the difference between the two scans (c), we can estimate the magnitude and direction of the offset between the actual sensor position (a) and the (false) initial estimate (b).



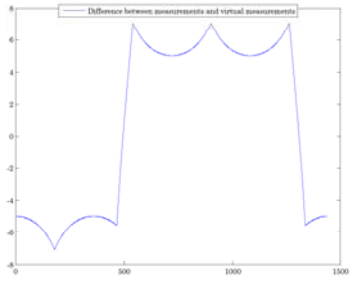
(a) The "real" scan, obtained by a sensor placed in the center of a square-shaped room.



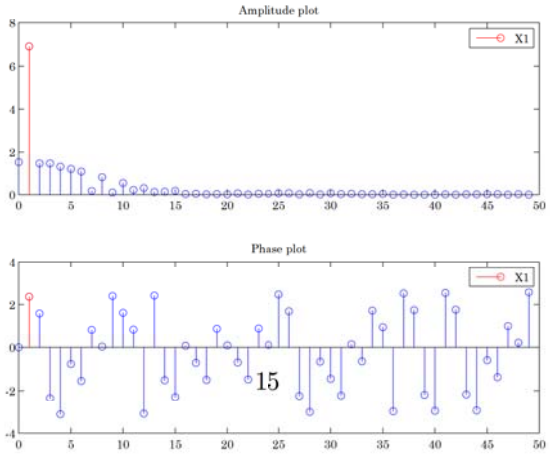
(b) The "virtual" scan, which would be obtained under the (false) assumption that the sensor is displaced from the room center.



(c) Graphs of the "real" and "virtual" measurements, i.e., scans from (a) and (b).



(d) The difference between the two scans, i.e., the blue and green graphs depicted in (c).



(e) Fourier transform of the difference signal (d). The ampllitude and phase of the first component, used to compute the position estimate offset, is highlighted in red.

Figure 5: The skilled 1000 autonomous forklift at the Euroimpianti testing facility in Schio.

3. System description

In this section, we introduce the vehicle, present its mathematical model and describe the environment in which the experiments were performed.

3.1. The Skilled 1000 autonomous forklift

The experiments presented within this paper were conducted using the Skilled 1000 autonomous forklift, shown in Figure 5. This is a next-generation prototype vehicle manufactured by Euroimpianti company. The vehicle has one steering wheel in the front and two support wheels at the rear, configured in a tricycle steering system. It is controlled by controlling the velocity and the angle of the front wheel. Figure 6 shows the steering configuration of the vehicle.

The following equations describe the motion of the vehicle in its local coordinate system.

$$\begin{aligned} v_x &= cos(\theta) \cdot v_1 \\ v_y &= 0 \\ \dot{\psi} &= sin(\theta) \cdot \frac{v_1}{a} \end{aligned} \tag{15}$$

where v_x and v_y represent translational velocities in x and y direction, $\dot{\psi}$ is the vehicle yaw rate, v_1 is the linear velocity of the front wheel (traction), θ is the

steering angle of the front wheel and a is the distance between passive wheels and driven wheel (see Figure 6). The distance a is directly measurable from the vehicle's geometry, whereas v_1 and θ can be obtained from motor encoder values using the following equations:

$$\begin{aligned} v_1 &= \omega_d \cdot s_1 \\ \theta &= (\alpha_s - s_2) \cdot s_3 \end{aligned} \tag{16}$$

where ω_d is the rotational velocity of the traction motor, α_s is the angle travelled by the steering motor and s_1, s_2 and s_3 are scaling parameters obtained by vehicle calibration. Velocities transformed into the global coordinate system are expressed as:

$$\begin{aligned} v_{xg} &= cos(\psi) \cdot v_x \\ v_{yg} &= sin(\psi) \cdot v_x \end{aligned} \tag{17}$$

where v_{xg}, v_{yg} are velocities in the global coordinate system in the x and y directions respectively. The odometry-based position estimate is obtained by integrating velocities. Since the controller operates on discrete time samples, integration is approximated by:

$$\begin{aligned} x_k &= x_{k-1} + v_{xg}(k) \cdot T_s \\ y_k &= y_{k-1} + v_{yg}(k) \cdot T_s \\ \psi_k &= \psi_{k-1} + \dot{\psi}(k) \cdot T_s \end{aligned} \tag{18}$$

where T_s is the sampling time and x_k, y_k and ψ_k represent the odometry-based pose of the vehicle at time step k.

The vehicle is equipped with four on-board lasers, three of those are safety lasers Sick S300 located at the ground level used to safely stop the vehicle if it comes too close to a moving or static obstacle. The fourth laser is the navigation laser Sick NAV350, a laser range scanner with the view angle of 360°. It provides the functionality of localizing the vehicle in 2-D space using reflective markers mounted in the environment. Range data from the navigation laser are used for the localization method presented within this paper.

Due to the prototype nature of the presented navigation and localization system, all of the algorithms were implemented on notebook PC that was inter-

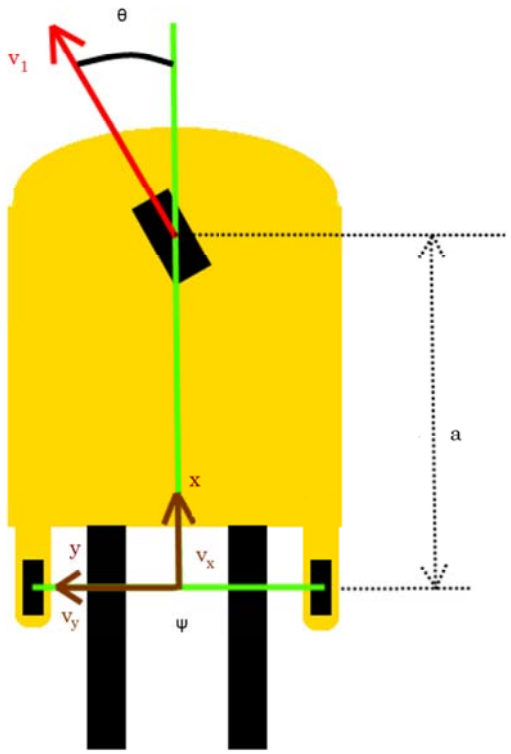


Figure 6: The car-like kinematics of the LGV. The front wheel is used for both steering and traction. The unactuated rear wheels, located sideways from the forks, are for stabilization only.

faced to several systems of the vehicle (Figure 7). Communication to the motor drives over CAN bus is used for gathering odometry data and for controlling vehicle motion. The ethernet interface was used for receiving range data from the NAV350 scanner. The ModbusTCP protocol was used for communicating with the system computer in order to enable and disable particular safety features when approaching a docking station. Data from the safety scanners (not used in the work presented in this paper) was retrieved through the RS422 interface.

All of the algorithms described within this paper were running on the laptop PC under the Ubuntu 14.04 operating system. The algorithms were imple-

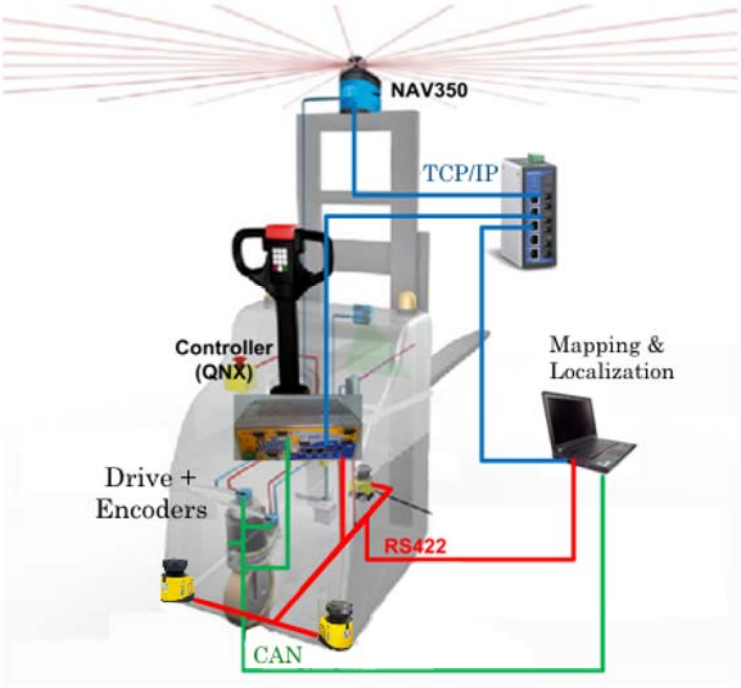


Figure 7: Interfacing of the control computer to the vehicle. By connecting our computer to the CAN Bus we can override the commands sent by the standard QNX-based controller and have full control over vehicle motion. Laser range readings from the NAV350 and the S300 are received over TCP/IP and RS422 respectively.

mented within the Robot Operating System (ROS) framework [23].

3.2. The testing environment

The localization experiments have been performed at the Euroimpianti manufacturing and testing facility in Schio (VI), Italy. The floorplan of the facility is shown in Figure 8. It consists of a 80m by 50m main hall and a 20m by 50m storage area. The main hall is used for assembling and testing palletization lines and AGVs, with over 30 people working there on a typical day. This facility provided a realistic unstructured operating environment for evaluating the accuracy of our localization algorithms. At the time our experiments were performed, two tire storage lines were being assembled in the southern and

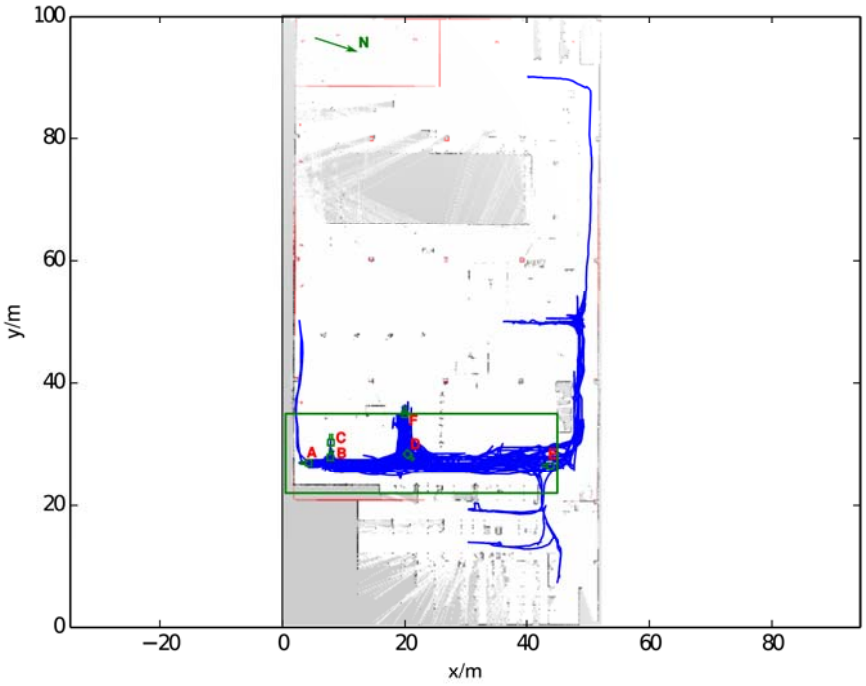


Figure 8: Floorplan of the Euroimpianti facility. The occupancy grid map used in the experiments is overlaid with the architectural CAD drawing (red lines). The resolution of the map is 5cm per pixel. Blue lines correspond to all paths travelled by the vehicle during the experiments. Red capital letters denote the docking stations, i.e. locations at which localization accuracy is evaluated. The green rectangle denotes the approximate area where ground truth information was provided by the reflective marker system.

south-western corridors, and this entire area was inaccessible to our vehicle. An approximately 45m by 15m area (marked by the green rectangle in Figure 8) was equipped with reflective markers providing ground truth information on vehicle pose.

Our experiments were performed in the eastern part of the facility. Six locations within the marker equipped area were chosen as docking stations where the localization system accuracy was evaluated. We used the northern hallway to drive the vehicle over longer distances, in order to check the robustness of our system to extensive vehicle motion. All paths taken by the vehicle during

(a) Testing area equipped with reflective mark-(b) Northern hallway with the high stack storage ers. The occasional motion of the big palletiz-shelf. Occasional loading and unloading items ing robots is a small source of dynamics at 2.5mwas another source of dynamics at 2.5m. height.

Figure 9: Photographs of the testing environment at the Euroimpianti facility. The environment at 2.5m height is mostly static, however, some occasional dynamics are present.

the experiments are shown with blue lines in Figure 8. Photographs of the experimental area are shown in Figure 9.

4. Vehicle positioning system

In order to experimentally verify the proposed localization algorithm on the LGV1000, a properly designed vehicle navigation system is necessary. Since the existing vehicle navigation system only works with predefined waypoints and paths, and we wanted to have more freedom in experimenting with the vehicle, we implemented a navigation system capable of autonomously planning its path. Forklift delivery missions usually consist of two distinct parts: point-to-point navigation and docking. In the navigation phase, the vehicle should take the shortest path from its current location to the docking station. In the docking phase, the vehicle must precisely align its orientation in order to position its forks correctly for pallet pickup or delivery. In this section, we describe the implementation details of motion control components that handle both of these tasks.

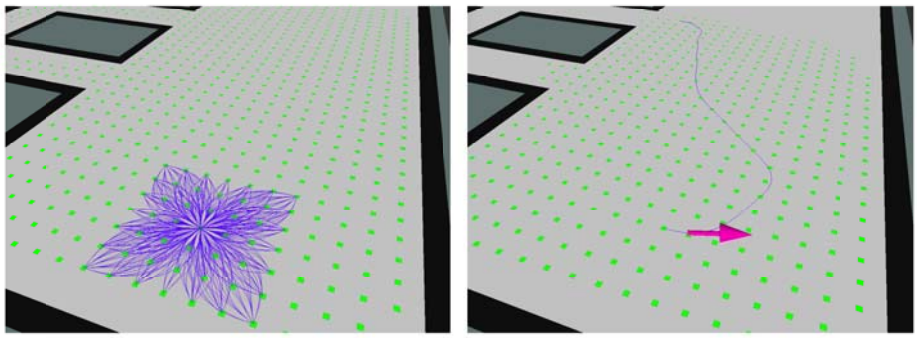
4.1. Point-to-point navigation

The primary goal of point to point motion controllers is to safely navigate the vehicle along the shortest obstacle-free path from an initial location towards a desired destination in the vehicle working space. Obviously, this functionality can be achieved by integration of the path planning and the path tracking algorithms. Therefore, the first step at the navigation system design phase is the selection and implementation of an appropriate path planning algorithm. Unlike most practical AGV systems in which the vehicles follow paths along predetermined path network, we decided to implement a free-ranging motion scheme which enables planning and execution of motions within the entire obstacle-free space. The adoption of the free-ranging motion scheme is more suitable for performing localization experiments as it allows for easy definition of arbitrary motion sequences within any part of the dynamic working environment. An important aspect that had to be considered during the design of the path planing algorithm is related to path feasibility due to the non-holonomic vehicle constraints.

Considering the desired free-ranging properties together with the path feasibility requirements, we decided to implement a path planning method based on the use of a *state lattice*. The state lattice, introduced in [24], represents a sampled vehicle state space which encodes feasible motions by design. The first step in the design of a path planning algorithm based on the use of a state lattice is the state lattice construction process. In this process, a vehicle state vector $[x, y, \psi]$ is uniformly sampled all over the vehicle workspace and the interval $[0, 2\pi]$, respectively. In this work we use the state lattice with $\pi/8$ orientation resolution and $0.25m$ distance between any two adjacent states in the $x - y$ plane. At the second step a boundary value problem (BVP) is solved in order to generate feasible motions (motion primitives) connecting one sampled state to a certain number of nearby states (Figure 10a). The number of states that each state is connected to, as well as the state sampling density is chosen based on the size of the vehicle, the size of workspace, and vehicle steering limitations. Since states are sampled uniformly over x and y, the calculated motion primi-

tives can easily be translated to any other discrete state in the working space. The overall path planning problem is then reduced to finding an appropriate sequence of motion primitives $\pi = (p_0, \cdots, p_n)$, which represents the resulting vehicle path connecting the initial and desired vehicle states (Figure 10b). The constructed state lattice can easily be represented in a form of a directed graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$. Each node $v \in \mathcal{V}$ represents a valid vehicle pose, while each edge $e \in \mathcal{E}$ represents a feasible motion with respect to the non-holonomic vehicle constraints. In this way, the overall path planning problem is reduced to a graph search problem and thus any graph search algorithm can be used to find a path between any two sampled states. In this work we have implemented the well-known A* search algorithm which uses Euclidean distance as a heuristic function. During the process of searching for the shortest path, each candidate path segment is also checked for collision against static obstacles in the working space. For this purpose the vehicle workspace is divided into cells. A cell is considered occupied if an obstacle is present in any part of it. The implemented collision detection mechanism detects collisions by checking the availability of all cells that are to be occupied by the vehicle during the execution of the candidate path segment. In case a collision has been detected, the corresponding path segment is excluded from the set of candidate segments. This procedure yields an obstacle-free path towards the desired vehicle's pose.

Once the path planning algorithm has been implemented, it was necessary to implement an appropriate path tracking algorithm to ensure that the vehicle can accurately follow the calculated path. For this purpose we have implemented the *pure pursuit* path tracking algorithm [25]. The parameters of this algorithm have been experimentally adjusted to values which ensure a precise path following as well as accurate vehicle positioning. The described path planning and path tracking algorithms have been implemented within a custom ROS package intended for execution on the on-board PC of the vehicle.



(a) Visualization of the state lattice around one of the discrete states in the working space.

(b) Example of resulting vehicle path comprising a sequence of state lattice segments.

Figure 10: Path planning in free space, based on the use of a state lattice. The final path is obtained by "stitching together" a sequence of lattice segments, ensuring continuous change of vehicle orientation.

4.2. Docking manoeuvres

The main requirement for docking is that the vehicle goes into the docking station following the *docking line* (line going out of the docking station perpendicular to the dock). Since the dock can have an arbitrary orientation and by entering a docking station, the vehicle is in reality going into an obstacle, the path planning presented in Section 4.1 cannot be used for docking manoeuvres and it is necessary to develop new path planning procedures suitable for these special cases. A possible solution is to use the path planning procedure from Section 4.1 to arrive at the graph node as close as possible to the docking line with the closest possible orientation and then calculate a continuous path consisting of two segments. The first *transtion segment* connects the pose of the mentioned graph node to the docking line using smooth continuous curve where the orientation of the end of the segment corresponds to the orientation of the docking line. The second segment follows the docking line until the vehicle arrives at the docking station. General path of the docking manoeuvre is shown in Figure 11.

In the case when the orientation of the vehicle w.r.t to the orientation of

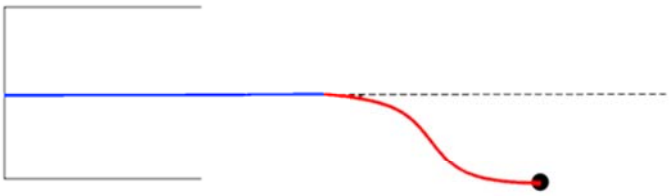


Figure 11: The two-segment docking path. The docking line (blue) ensures straight vehicle motion for picking up and delivering pallets. The transition segment (red) ensures a smooth transition between the initial pose of the vehicle and the docking line.

the goal point is greater than δ_{max}(dependent on the configuration of the vehicle) or when distance from the vehicle to the docking line is greater then d_{max}(dependent on the configuration of the vehicle), the motion is assumed not feasible.

The only problem remaining is the choice of a nonlinear function for the first segment. Given that the starting and ending orientation are going to be similar, the following function is chosen:

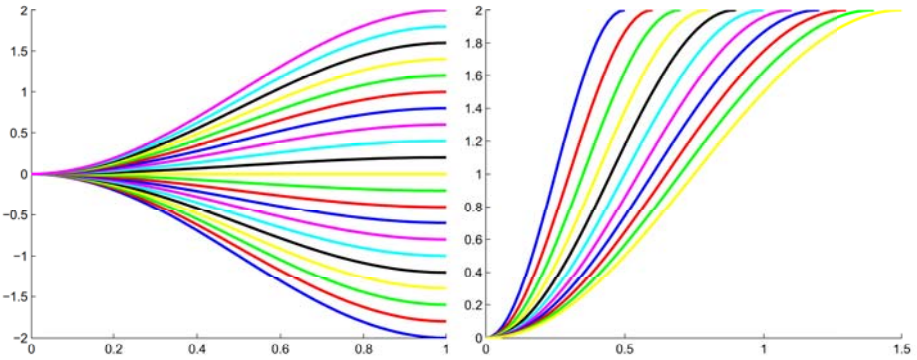
$$f(x) = a(1 - \cos(b \cdot x)). \tag{19}$$

If (x_s, y_s, ψ_s) is the starting pose and (x_e, y_e, ψ_e) is the end pose of the segment, then the following is the relative pose of the end pose with respect to the start pose:

$$\begin{aligned} x_r &= (x_e - x_s)\cos(-\psi_s) - (y_e - y_s)\sin(-\psi_s) \\ y_r &= (x_e - x_s)\sin(-\psi_s) + (y_e - y_s)\cos(-\psi_s) \\ \psi_r &= \psi_e - \psi_s \end{aligned} \tag{20}$$

The problem comes down to the following: find parameters a and b such that:

$$\begin{aligned} y_r &= a(1 - \cos(b \cdot x_r)) \\ \tfrac{dy}{dt} &= b\sin(b \cdot x_r) \\ \tfrac{dy}{dt} &= \tan(\psi_r) \end{aligned} \tag{21}$$



(a) Transition segment shapes for a range of different initial offsets in the x coordinate.

(b) Transition segment shapes for a range of different initial offsets in the y coordinate.

Figure 12: Transition segment shapes for different initial offsets x and y coordinates respectively.

Since equation (19) is periodic, there is an infinite number of solutions to the equation (21). To avoid this problem, the range of a and b is limited in such a way that only the first solution is found:

$$\begin{aligned} a &\in [-0.5y_r, -1.5y_r] \cup [0.5y_r, 1.5y_r] \\ b &\in [\tfrac{\pi}{2x_r}, \tfrac{3\pi}{2x_r}] \end{aligned} \tag{22}$$

The solution to the equation (21) (parameters a, b) with constraints (22) is found numerically.

Figure 12 shows graphs of the *transition segment* for different x and y coordinates. The path segment, defined by a set of points distributed from starting pose to the ending pose, can be calculated by selecting N equally distributed points in the range $[0, x_r]$, and using equation (19) to calculate the y coordinate of each point. This procedure returns a set of points (x_{ri}, y_{ri}) that represent path points in the coordinate frame of the starting pose. To acquire the path segment in the global coordinate frame, points (x_{ri}, y_{ri}) have to be converted

using the inverse of the equation (20):

$$\begin{aligned} x_i &= x_{ri}\cos(\psi_s) - y_{ri}\sin(\psi_s) + x_s \\ y_i &= x_{ri}\sin(\psi_s) + y_{ri}\cos(\psi_s) + y_s \end{aligned} \tag{23}$$

During pallet pickup, the assumption that the pallet orientation and position are known is sound, since it is aligned before reaching the pickup location. Planning of the docking maneouver starts when the vehicle reaches the position assumed to be close to the docking line and with the similar orientation. Unlike the method presented in 4.1, which is calculated offline, approach presented within this section is fast enough to be calculated online.

5. Experimental evaluation

In the period between March 10 and March 13 2015, we performed continuous localization experiments with a LGV1000 vehicle prototype at the Euroimpianti facility in Schio (Vicenza, Italy). By continuous, we mean that there was no manual intervention in the localization estimate during the experimental period. During this time, the vehicle was in motion for over 19 hours, logging a total path of over 8.5km and performing 287 docking manoeuvres at designated docking stations. All the relevant data was recorded using the `rosbag` tool from ROS. In the remainder of this section we explain the methodology used to evaluate the accuracy of our localization algorithm and present the experimental results.

5.1. Methodology description

The goal of our experiment was twofold:

1. verify that the localization accuracy of our algorithm stack is within 0.015m and 0.5° at designated docking stations
2. assert that the localization error does not systematically increase with long-term and long-range vehicle motion

These requirements stem from industry practice, as AGVs perform repetitive tasks over extended periods of time. The necessary localization accuracy is set by commonplace warehousing system design practice, where (de)palletizing manipulators operate on predefined trajectories, without measuring the exact position of the pallets. Therefore, pallet placement must be performed with high accuracy.

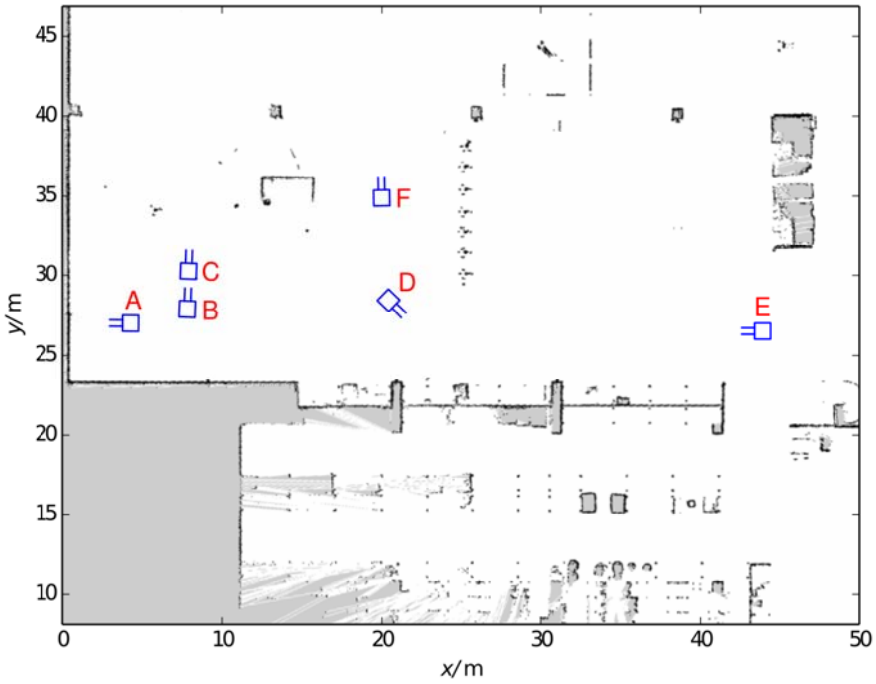


Figure 13: Docking poses at which localization accuracy is evaluated. The positions were chosen to span the whole area covered by the reflective markers, where accuracy evaluation can be performed.

To verify the accuracy of our localization algorithm, we performed repeated docking manoeuvres at six different poses in the environment, depicted in Figure 13. At these poses we evaluate both *localization accuracy* and *positioning accuracy*. A state of the art reflective marker based system, installed in the eastern corridor (area marked by green rectangle in Figure 8) was used for ob-

taining ground truth information on vehicle pose. In the evaluation of results, we follow the methodology presented in [7]. In order to avoid the inaccuracies introduced by the non real-time nature of the computer running the algorithm and the lack of timestamp synchronization with the NAV350 scanner, accuracy is evaluated once the vehicle reaches its target pose and stops its motion. Additional inaccuracies could be introduced by the mapping procedure, as the map was generated from odometry and laser scanner data only, without using ground truth information, as well as by the limited accuracy of the marker based system itself. Therefore, we analyse *relative errors*, with respect to the reference poses depicted in Figure 13. The pose reached by the vehicle upon the first docking manoeuvre at each respective docking station is taken as the *reference pose*. For every subsequent pose, the positioning error is computed as the deviation from the reference pose, as reported by the marker based localization system. The localization error is computed as the difference between this positioning error, and the positioning error reported by our localization algorithm.

During the experiment, the vehicle was operated in three modes: manual, semi-automatic with goals provided manually by the operator, and fully automatic, cyclically executing a sequence of missions loaded from a file. The goal was to verify that the localization error does not systematically increase as the vehicle operates for extended periods of time and traverses greater distances. Furthermore, by changing mission sequences from time to time, we made sure that different approach paths were taken by the vehicle, thus verifying the robustness of our approach. It is also important to point out that the testing facility was fully operational for the whole duration of the experiment. Workers and other vehicles, both autonomous and human operated, were constantly present in the environment. However, due to the fact that the NAV350 scanner is mounted at 2.5m, which is standard for this vehicle model, the dynamics of the environment visible to the scanner was low. Only occasional changes were visible, when objects were moved to/from high stack storage shelves.

5.2. Localization accuracy

The localization accuracy results are summarized in Table 2, Table 3 and in Figure 14 (red boxplots). Out of the total 287 docking manoeuvres, the localization estimate provided by our algorithm was within required tolerance 276 times, that is in 96% of cases. The average error is 0.87cm with a standard deviation of 0.42cm. As Figure 14b shows, the orientation estimate was accurate at all times.

Table 2: Average localization errors and corresponding standard deviaitons (Sd).

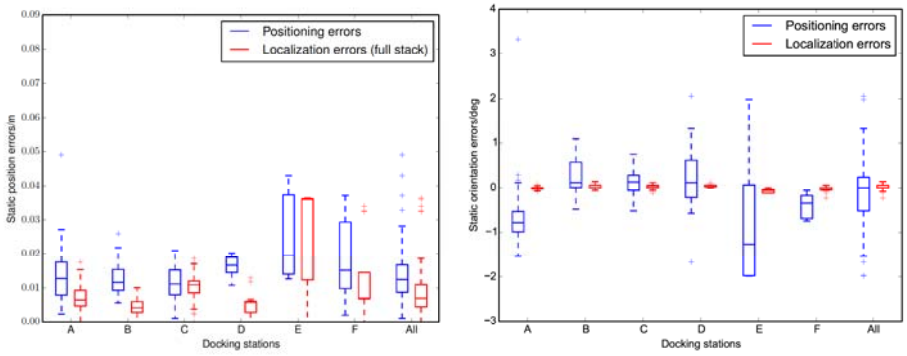
	Position error		Orientation error	
Algorithm	Average/cm	Sd/cm	Average/°	Sd/°
AMCL	3.89	3.28	1.27	1.04
AMCL with ICP	1.61	1.03	0.13	0.07
Full stack	0.87	0.42	0.13	0.07

Out of the six docking stations, localization accuracy was over 95% at four stations. The worst accuracy was achieved at docks F and E. These docks are distinguished by two features. Firstly, they lie at the border of the reflective marker area, so the ground truth information reported at these docks suffers from outliers. Furthermore, the number of measurements at these docks is significantly lower than at the other docks. This is due to the fact that both docks are located in parts of the facility which were for a significant amount of time allocated to other activities, and were therefore inaccessible for our experiments. The small number of measurements makes these locations inadequate for statistical analysis of the results, however we decided to include them for completeness.

To examine the evolution of localization errors in time, in Figure 15 we display errors from all docking maneuvers as individual data points, ordered by their time stamp. Again, it is obvious that the orientation errors are almost an order of magnitude smaller than the industrial requirement. A total of 11

Table 3: Localization accuracy summary per docking station.

Dock	Number of dockings	Number of localizations within requirements	Percentage
A	82	78	95.12%
B	71	70	98.59%
C	100	99	99.00%
D	25	24	96.00%
E	5	2	40.00%
F	10	8	80.00%
All	287	276	96.17%



(a) Boxplots of position errors, by docking station.

(b) Boxplots of orientation errors, by docking station.

Figure 14: Box plots of positioning and localization errors at the docking stations. Positioning errors (blue) have larger average values and a higher dispersion, caused by inaccuracies in the path following implementation.

475 distance errors are greater than the required 1.5cm, with two significant outliers around 3cm, one around 2cm and the remainder well below 2cm. There is no visible trend that would indicate a systematic increase of errors after 19 hours of continuous operation. This can also be interpreted as a property of the environment, i.e., the absence of a systematic error that increases with time

indicates that the environment is "static enough" to allow our algorithm to run successfully with a static map.

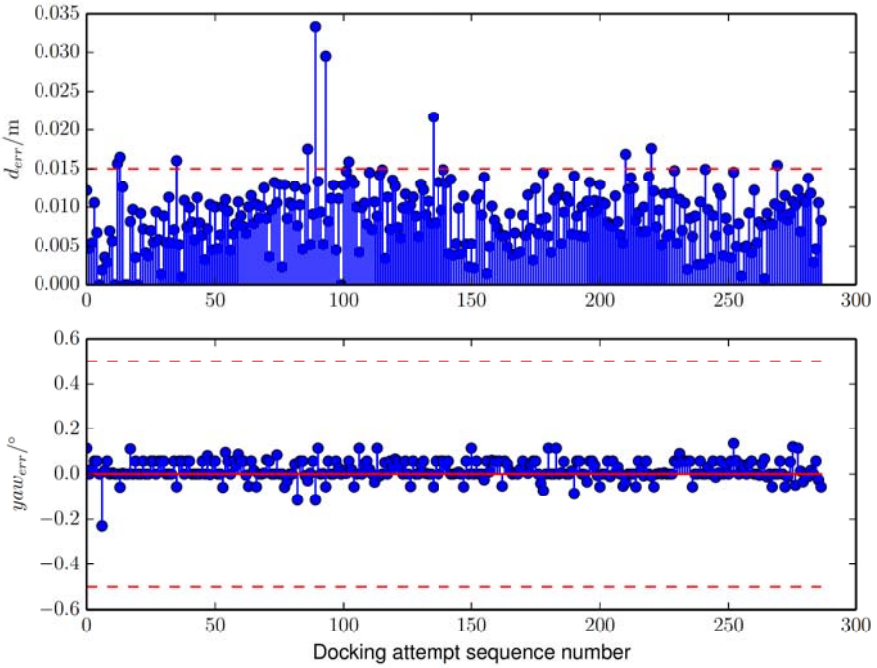


Figure 15: Evolution of localization errors (position and orientation) in time. Each data point represents one docking manoeuvre. There is no visible trend that would indicate a systematic increase in localization accuracy over time.

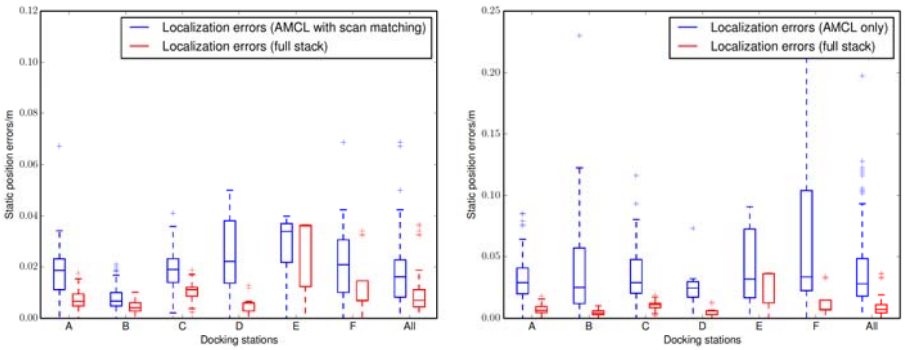
5.3. Accuracy improvements through scan matching and DFT

Our localization algorithm stack is composed of three components: AMCL, scan matching and DFT. In this section, we analyse the contribution of each individual component to the overall accuracy of our stack. To this end, we have extracted the localization estimates provided by AMCL only, and by AMCL with scan matching (without the DFT step) and compared their accuracy to the accuracy of the full localization stack.

The results of the comparison are summarized in Table 2, Table 4 and Figure 16. They indicate that each component of the localization stack delivers

Table 4: Localization accuracy by docking station, for individual steps of the algorithm.

	Percentage of localizations within requirements		
Dock	AMCL only	AMCL with ICP	Full stack
A	15.85	35.37	95.12
B	29.58	87.32	98.59
C	13.00	35.00	99.00
D	16.00	32.00	96.00
E	20.00	20.00	40.00
F	20.00	40.00	80.00
All	18.22	49.15	96.17



(a) Position estimate errors obtained by using AMCL with scan matching, compared to errors obtained by using the full localization stack.

(b) Position estimate errors obtained by using only AMCL, compared to errors obtained by using the full localization stack.

Figure 16: Boxplots of localization errors at the docking stations, using AMCL with scan matching and only AMCL respectively, compared to the errors obtained by using the full localization stack (AMCL with scan matching and DFT).

a significant increase of accuracy with respect to the previous step. Both the average error and the standard deviation are reduced roughly by half with each algorithm step. This observation is consistent over all docking stations. We conclude that with our implementations of the scan matching and DFT algo-

rithms, all three algorithm stack components are necessary to achieve a high percentage of localizations that satisfy industrial requirements.

5.4. Positioning accuracy

Although the main contribution of this paper lies in the localization algorithm, and the positioning system was implemented simply as a means of validating localization accuracy, for the sake of completeness we also evaluate positioning accuracy. During the run of the experiment, positioning accuracy was occasionally checked by placing pallets at the docking stations and having the vehicle transport them between different stations. Two snapshots of a delivery mission between dock C and dock D are shown in Figure 17.

(a) Pallet pickup at docking station C.

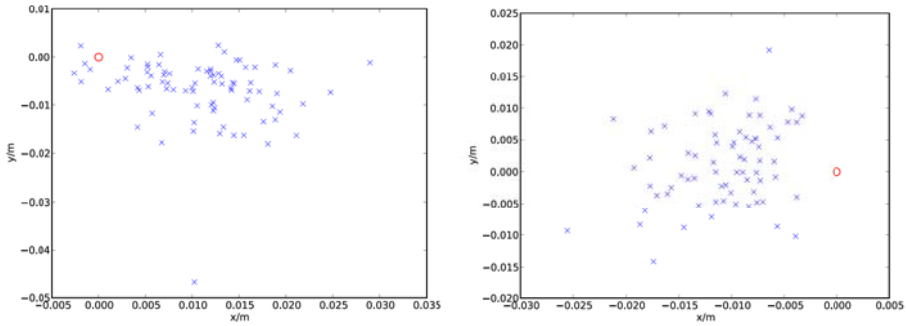
(b) Pallet stacking at docking station D.

Figure 17: Snapshots of a pallet delivery mission between docking station C and docking station D. This is one of the delivery missions that was repeated cyclically during the course of the experiment. On some runs, real pallets were used in order to visually demonstrate the accuracy of the localization and positioning system.

The analysis results are depicted by the blue boxplots in Figure 14 and the vehicle positioning errors at the six docking stations are depicted in Figure 18. The average positioning error is 1.3cm with 0.6cm standard deviation, which comes very close to satisfying the industrial requirements. Scatter plots in Figure 18 also reveal that the vast majority of positioning errors have acceptably small values. However, the discrepancy between localization and positioning is very obvious in the case of orientation errors, depicted in Figure 14b. The main

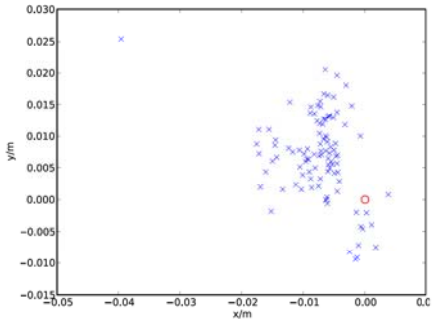
reason for the relatively poor performance of the positioning system with respect to orientation accuracy is a flaw in the path following algorithm design. Path following is implemented as a pure pursuit algorithm, which uses a *look-ahead distance* parameter to steer the vehicle towards a point on the path which is at some distance from the current position of the vehicle. With a straight path (which the final approach segment of a docking maneuver always is), bigger look-ahead distances result in more stable orientation of the vehicle. However, in our current implementation, as the vehicle approaches the final point on its trajectory, the look-ahead distance starts shrinking (because there are no further points to track), steering the vehicle more aggressively towards the goal. The result is that even very small position errors will result in significant changes in orientation when close to the goal.

Looking at the positioning accuracy scatter plots in Figure 18, there is a noticeable bias, i.e., the blue crosses are predominantly grouped on one side of the origin (zero error), marked by the red circle. This is a consequence of the tolerance margin of the positioning system and can be better understood by looking at the docking paths, depicted in Figure 19 for docking station A. In our experimental procedure, when we sent a vehicle to a docking station for the first time, the pose obtained after this initial manoeuvre was used as the pose reference for all subsequent dockings. Because the positioning algorithm requires a small tolerance value, it would stop the vehicle shortly before reaching the goal pose.

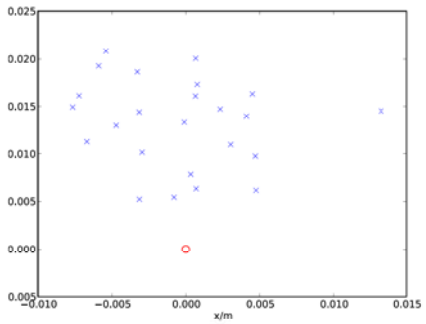


(a) Docking station A.

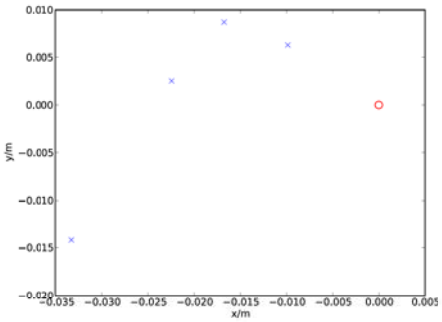
(b) Docking station B.



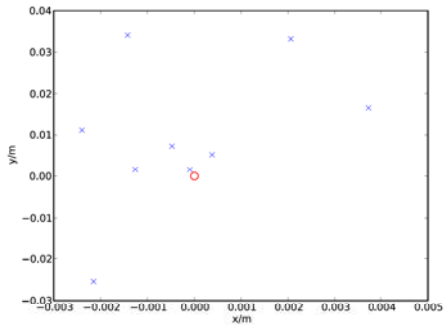
(c) Docking station C.



(d) Docking station D.

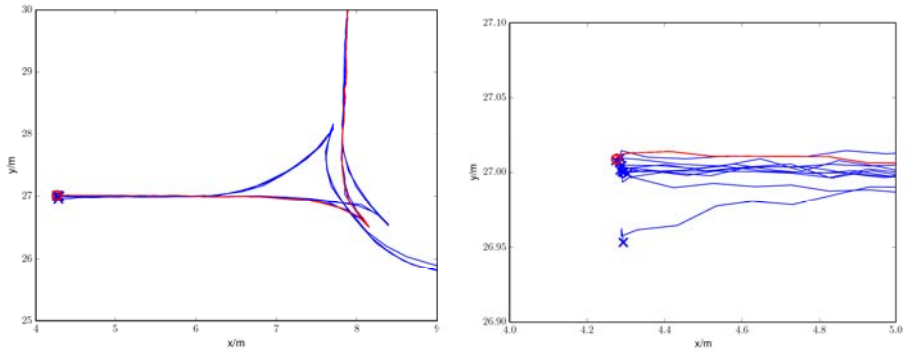


(e) Docking station E.



(f) Docking station F.

Figure 18: Vehicle positioning accuracy at the docking stations. Red circles represent the initial docking manoeuvre and the blue crosses represent subsequent dockings. The bias is due to the limited accuracy of the positioning algorithm implementation.



(a) A wider view of some of the paths executed by the vehicle to approach docking station A.

(b) A zoomed-in view of some of the paths executed by the vehicle to approach docking station A.

Figure 19: Several paths executed by the vehicle at docking station A. This docking station was always approached along the negative x-axis. The zoomed-in view reveals the accuracy limits of the marker-based localization system, which is around 2cm.

6. Conclusions and future work

In this paper we have presented an algorithm stack for high-precision localization of autonomous ground vehicles in industrial environments. Through an extensive experiment in a realistic industrial environment, we have demonstrated that the algorithm is capable of maintaining a localization estimate over extended periods of time, that is accurate to sub-centimetre and sub-degree levels. This algorithm presents a clear improvement over the industrial state of the art, as it can provide the required levels of accuracy and robustness, while requiring no external infrastructure. At the same time, it is competitive with respect to the most recent scientific research results. In our implementation, the DFT-based position estimate refinement that we propose has enabled a 50% reduction in the average localization error. Moreover, this improvement has enabled our system to satisfy industrial requirements for localization accuracy in over 96% of evaluated measurements.

In future work, we plan to improve the algorithm performance during vehicle motion by implementing an accurate time stamping mechanism for odometry and laser data. Furthermore, we plan to implement several improvements to the navigation and path following algorithms, in order to raise the vehicle positioning accuracy to the level that can satisfy industrial requirements. Finally, we plan to deploy the described algorithm to a customer site in order to evaluate it on a time scale of weeks and months. The data gathered during such an extended timespan would enable us to address the challenging and still unanswered questions regarding long-term autonomous vehicle operation in a changing environment.

7. Acknowledgments

This work has been supported by the European Commission FP7-ICT-2011-7 project Estimation and Control for Safe Wireless High Mobility Cooperative Industrial Systems (EC-SAFEMOBIL, Project No. 288082).

References

[1] P. R. Wurman, R. D'Andrea, M. Mountz, Coordinating hundreds of cooperative, autonomous vehicles in warehouses, AI magazine 29 (1) (2008) 9.

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

[3] D. Ronzoni, R. Olmi, C. Secchi, C. Fantuzzi, AGV global localization using indistinguishable artificial landmarks, in: Robotics and Automation (ICRA), 2011 IEEE International Conference on, IEEE, 2011, pp. 287–292.

[4] R. D'Andrea, A revolution in the warehouse: A retrospective on kiva systems and the grand challenges ahead (guest editorial), Automation Science and Engineering, IEEE Transactions on 9 (4) (2012) 638–639.

[5] H. Andreasson, A. Bouguerra, M. Cirillo, D. N. Dimitrov, D. Driankov, L. Karlsson, A. J. Lilienthal, F. Pecora, J. P. Saarinen, A. Sherikov, et al., Autonomous transport vehicles: where we are and what is missing, Robotics & Automation Magazine, IEEE 22 (1) (2015) 64–75.

[6] IFR Statistical Department, World robotics: Service robots 2014, Tech. rep., VDMA Robotics + Automation Association, http://www.worldrobotics.org/index.php?id=home&news_id=275 (2014).

[7] J. Rowekamper, C. Sprunk, G. D. Tipaldi, C. Stachniss, P. Pfaff, W. Burgard, On the position accuracy of mobile robot localization based on particle filters combined with scan matching, in: Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, IEEE, 2012, pp. 3158–3164.

[8] J.-S. Gutmann, W. Burgard, D. Fox, K. Konolige, An experimental comparison of localization methods, in: Intelligent Robots and Systems, 1998.

Proceedings., 1998 IEEE/RSJ International Conference on, Vol. 2, IEEE, 1998, pp. 736–743.

[9] F. Dellaert, D. Fox, W. Burgard, S. Thrun, Monte carlo localization for mobile robots, in: Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on, Vol. 2, IEEE, 1999, pp. 1322–1328.

[10] S. Thrun, D. Fox, W. Burgard, F. Dellaert, Robust Monte Carlo localization for mobile robots, Artificial intelligence 128 (1) (2001) 99–141.

[11] D. Brscic, H. Hashimoto, Comparison of robot localization methods using distributed and onboard laser range finders, in: Advanced Intelligent Mechatronics, 2008. AIM 2008. IEEE/ASME International Conference on, IEEE, 2008, pp. 746–751.

[12] C. Sprunk, G. D. Tipaldi, A. Cherubini, W. Burgard, Lidar-based teach-and-repeat of mobile robot trajectories, in: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, IEEE, 2013, pp. 3144–3149.

[13] A. Censi, An ICP variant using a point-to-line metric, in: Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on, IEEE, 2008, pp. 19–25.

[14] C. Reinke, 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.

[15] P. Beinschob, C. Reinke, Advances in 3D data acquisition, mapping and localization in modern large-scale warehouses, in: Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on, IEEE, 2014, pp. 265–271.

[16] P. Biber, W. Straßer, The normal distributions transform: A new approach to laser scan matching, in: Intelligent Robots and Systems, 2003.(IROS

2003). Proceedings. 2003 IEEE/RSJ International Conference on, Vol. 3, IEEE, 2003, pp. 2743–2748.

[17] J. Saarinen, H. Andreasson, T. Stoyanov, A. J. Lilienthal, Normal distributions transform monte-carlo localization (NDT-MCL), in: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, IEEE, 2013, pp. 382–389.

[18] R. Valencia, J. Saarinen, H. Andreasson, J. Vallvé, J. Andrade-Cetto, A. J. Lilienthal, Localization in highly dynamic environments using dual-timescale NDT-MCL, in: Robotics and Automation (ICRA), 2014 IEEE International Conference on, IEEE, 2014, pp. 3956–3962.

[19] G. Vasiljevic, F. Petric, Z. Kovacic, Multi-layer mapping-based autonomous forklift localization in an industrial environment, in: Control and Automation (MED), 2014 22nd Mediterranean Conference of, IEEE, 2014, pp. 1134–1139.

[20] J. Sturm, N. Engelhard, F. Endres, W. Burgard, D. Cremers, A benchmark for the evaluation of RGB-D SLAM systems, in: Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, IEEE, 2012, pp. 573–580.

[21] C. Sprunk, J. Röwekämper, G. Parent, L. Spinello, G. D. Tipaldi, W. Burgard, M. Jalobeanu, An experimental protocol for benchmarking robotic indoor navigation, in: Proc. of the International Symposium on Experimental Robotics (ISER), 2014.

[22] G. Grisetti, C. Stachniss, W. Burgard, Improved techniques for grid mapping with Rao-Blackwellized particle filters, IEEE Transactions on Robotics 23 (1) (2007) 34–46. doi:10.1109/TRO.2006.889486.

[23] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, ROS: an open-source robot operating system, in: ICRA workshop on open source software, Vol. 3, 2009, p. 5.

[24] M. Pivtoraiko, R. A. Knepper, A. Kelly, Differentially constrained mobile robot motion planning in state lattices, Journal of Field Robotics 26 (3) (2009) 308–333.

[25] R. C. Coulter, Implementation of the pure pursuit path tracking algorithm, Tech. Rep. CMU-RI-TR-92-01, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA (January 1992).