Lidar-based Teach-and-Repeat of Mobile Robot Trajectories

Christoph Sprunk* Gian Diego Tipaldi* Andrea Cherubini‡ Wolfram Burgard*

***Abstract*— Automation of logistics tasks for small lot sizes and flexible production processes requires intuitive and easy-to-use systems that allow non-expert shop floor workers to naturally instruct transportation systems. To this end, we present a novel laser-based scheme for teach-and-repeat of mobile robot trajectories that relies on scan matching to localize the robot relative to a taught trajectory, which is represented by a sequence of raw odometry and 2D laser data. This approach has two advantages. First, it does not require to build a globally consistent metrical map of the environment, which reduces setup time. Second, the direct use of raw sensor data avoids additional errors that might be introduced by the fact that grid maps only provide an approximation of the environment. Real-world experiments carried out with a holonomic and a differential drive platform demonstrate that our approach repeats trajectories with an accuracy of a few millimeters. A comparison with a standard Monte Carlo localization approach on grid maps furthermore reveals that our method yields lower tracking errors for teach-and-repeat tasks.**

I. Introduction

Autonomous transportation platforms are increasingly employed for logistics applications in industrial contexts. Initially, these systems were relying on guidance wires or optical markers to couple the robot to routes [1], [2]. In the last years, the paradigm shifted to the use of on-board sensors for localization and navigation, but for the most part still augmenting the environment with artificial markers [3]–[5]. While the well established systems are bound to their routes by the physical location of markers in the environment, more recent systems do not require such an augmentation of the environment and rather rely on a globally consistent metrical map to allow more flexible route planning.

However, all these systems require an expert user to adapt to changes of the environment or transportation routes. The global map has to be rebuilt and docking positions at route destinations have to be re-taught, e.g., using the approach of Röwekämper *et al.* [6]. At the same time, to make automation profitable for small lot sizes and frequently changing production processes one requires systems that can be intuitively used by non-expert shop floor workers. The system proposed in this paper is targeted towards a natural *one-button* approach that lets the user remotely control the robot to demonstrate a new trajectory that is then reproduced with high accuracy and precision.

In this paper we describe an approach that provides an intuitive teach-and-repeat framework for robot navigation.

* Department of Computer Science, University of Freiburg, Germany
‡ LIRMM, Université de Montpellier 2 CNRS, Montpellier, France

This work has partly been supported by the European Commission under contract numbers FP7-248258-First-MM and FP7-260026-TAPAS.

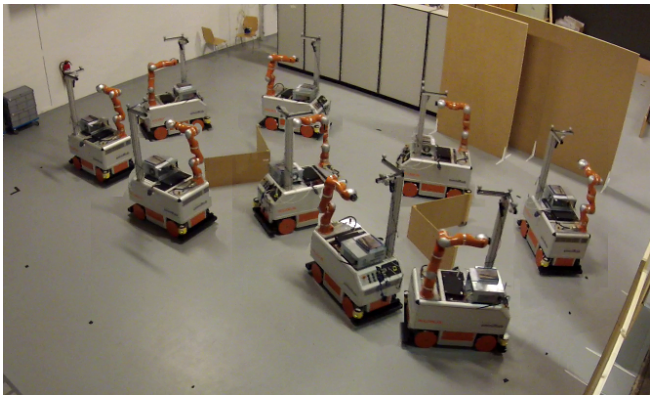

Fig. 1. Time-lapsed view of the holonomic omniRob robot reproducing the user-taught reference trajectory FigureEight with our approach without a globally consistent metrical map.

Our method represents user-taught trajectories by so-called anchor points consisting of raw odometry and 2D laser data. It applies a scan matching routine to estimate the actual offset of the current robot position from these anchor points. By comparing the reference offset with the actual offset, our algorithm calculates the error in the configuration space of the robot, which is then used for any desired feedback controller, e.g., a linear quadratic regulator or dynamic feedback linearization. With respect to standard appearance-based navigation, our approach differs in the sensor modality (laser scans vs. camera images) and in the feedback (in configuration space vs. in sensor space). Practical experiments carried out in complex real-world scenarios demonstrate that with our navigation scheme, robots can repeat taught trajectories more accurately than with the standard Monte Carlo localization approach using a global grid map.

II. Related Work

The most widespread approaches to mobile robot navigation are model- and appearance-based approaches [7]. Model-based approaches usually rely on a map of the environment, whereas appearance-based approaches do not require such a model and work directly in the space of the sensor data. The environment is usually described by a topological graph, in which each node corresponds to the description of a position, and a link between two nodes defines the possibility for the robot to move autonomously between the two positions that the link connects. Our focus here is on appearance-based navigation.

Recently, many researchers [7]–[14] have developed appearance-based frameworks for realizing teach-and-repeat navigation. During a preliminary *teach* phase, the robot

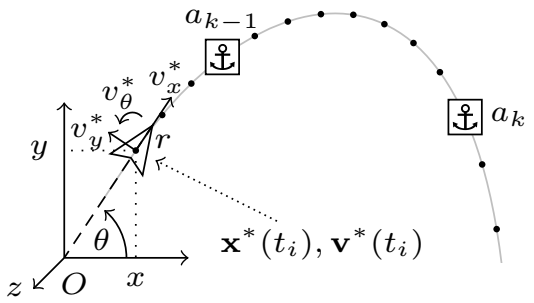

Fig. 2. During the teaching phase, we record the velocities $\mathbf{v}^*(t_i)$ at discrete time steps t_i, and use them to derive the taught trajectory poses $\mathbf{x}^*(t_i)$ (black dots). Along this trajectory, we insert anchor points. This schematic figure also illustrates the used coordinate systems.

motion is controlled by a human operator, and a sequence of sensor data is acquired and stored in a database. Then, during the *repeat* phase, the robot is required to repeat the same path, by comparing the currently observed and the previously recorded sensor data, relying on visual data in most works.

Pioneering work [8] associated a particular motion (e.g., "go forward", "turn left") to each recorded image, in order to move from the current to the next image in the database. In [9], a three dimensional representation of the taught path is built from the image sequence, and a classic path following controller is used for navigation. A similar approach, but using omnidirectional cameras, has been proposed in [10]. There, a path is represented by a sequence of images and a local 3D map is reconstructed from them. Path following is then done with a sequence of homing vectors to each image and feature tracking is used to localize the robot. The controller presented in [11] exploits angular information regarding the features matched in panoramic images. In [12], a novel varying reference based teach-and-repeat framework is proposed for outdoor navigation with a monocular camera.

In industrial applications, impressive results have been obtained with an infrastructure-free robot that uses three cameras to navigate with naturally occurring visual cues learned during operation [13]. Another teach-and-repeat framework for a mass customization manufacturing scenario relies on high-intensity visual markers and stereovision to enable quick and easy use and configuration for non-skilled users [14]. Furthermore, camera images have been replaced by a laser reflectivity image in [15], which extends the authors' previous work on visual teach-and-repeat [16].

While all these works rely on images, our objective here is to validate a similar approach, based solely on odometry and 2D laser scanners, that are generally mounted on industrial robots for safety reasons. Closely related is the approach of Marshall *et al.* [17]. The main idea of their work is to construct a sequence of local grid maps rather than having a globally consistent, monolithic map. During the teach phase the maps are built and a path is recorded. The recorded path is then repeated by localizing the robot in the local maps using the unscented Kalman filter. Our approach differs from this method as we do not rely on grid maps and instead operate directly on the sensor data.

In previous work [18], we also considered teach-and-repeat tasks with mobile robots. There, however, we expressed the paths as splines in a global coordinate frame and applied Monte Carlo localization to localize the robot in a global grid map. Here, we strive to repeat the user input as exactly as possible, without relying on a global, metrically consistent map of the environment.

III. Problem definition

The objective of our work is to repeat, with a wheeled mobile robot and no map, a time trajectory previously taught by user demonstration. The ground is assumed planar, and the robot may be either holonomic or subject to nonholonomic constraints (e.g., a differential drive robot), and it is equipped with a laser scanner mounted parallel to the ground.

We name r the robot center of rotation, which should track the trajectory. With reference to Fig. 2, we define the world frame $\mathcal{F}(O, x, y, z)$. The robot configuration is:

$$\mathbf{x} = [x, y, \theta]^\top \in SE(2)\,, \tag{1}$$

where x and y represent the Cartesian position of r in $\mathcal{F}$, and $\theta \in (-\pi, \pi]$ is the positive counterclockwise orientation of the robot with respect to x. The control inputs are:

$$\mathbf{v} = [v_x, v_y, v_\theta]^\top \in \mathbb{R}^3\,. \tag{2}$$

These are respectively the longitudinal translational, lateral translational, and rotational velocities, with sign conventions as shown in Fig. 2. For a differential drive robot, $v_y = 0$.

The goal of this work is to drive, at time $t \in [0, T]$, the robot configuration $\mathbf{x}(t)$ to the taught trajectory $\mathbf{x}^*(t)$. This trajectory is defined using the robot recorded odometry, as explained below. To realize this task, we rely on two sensors:

Wheel encoders: Measurements of the revolutions of the robot's wheels are available at discrete timesteps t_i ($t_0 = 0$). The robot velocities $\mathbf{v}$ are estimated by differentiating the encoder readings.

Laser scanner: The robot on-board laser scanner provides, at discrete timesteps t_j, a local 2D scan of the environment surrounding the robot.

The robot velocities estimated from the wheel encoders can be used to localize the robot at any time $t \in (t_i, t_{i+1}]$, employing the well known odometry equation [19]:

$$\mathbf{x}(t) = \mathbf{x}(t_i) \oplus \mathbf{R}_z(\theta(t_i))\mathbf{v}(t_i)\,(t - t_i)\,, \tag{3}$$

with $\mathbf{R}_z$ the rotation matrix about the z axis, and $\oplus$ the compounding operator [20]. During the teach phase, the estimated robot velocities are recorded. By plugging these recorded velocities $\mathbf{v}^*(0), \mathbf{v}^*(t_1), \mathbf{v}^*(t_2), \ldots, \mathbf{v}^*(T)$, and the initial pose $\mathbf{x}(0)$ into Eq. (3), we obtain the taught trajectory $\mathbf{x}^*(t)$ to track. Although it has piece-wise constant velocity, this trajectory will provide a sufficiently accurate estimation of the real user-taught one, if the encoder frequency is sufficiently high. A schematic illustration of a recorded reference trajectory is given in Fig. 2, in which the black dots correspond to the poses $\mathbf{x}^*(t_i)$ at timesteps t_i.

The local 2D scans of the environment are used by our approach in conjunction with a scan matcher to provide

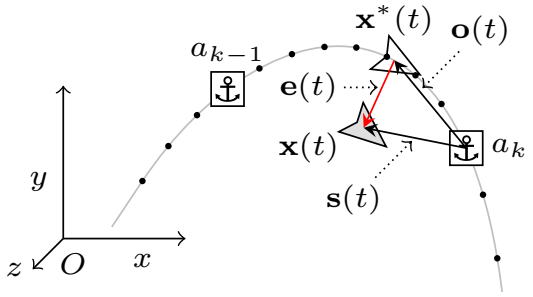

Fig. 3. During the repeat phase, the reference pose of the robot (hollow dart) at time t, $\mathbf{x}^*(t)$, is expressed as relative offset $\mathbf{o}_{a_k}(t)$ with respect to the current anchor point a_k, see Eq. (5). The actual offset $\mathbf{s}_{a_k}(t)$ of the current robot pose $\mathbf{x}(t)$ (shaded dart) from the anchor point a_k is measured by a scan matcher (Eq. (6)) and used to compute the error $\mathbf{e}_{a_k}(t)$ as in Eq. (7). In the figure, the indices of these vectors have been dropped.

relative estimates of the robot's pose. Note that in our approach, those estimates are not integrated and therefore not prone to accumulating drift. During the teach phase, scanner readings are recorded at regular intervals along the trajectory: we record a new scan as soon as the distance covered since the last one exceeds a fixed threshold $\tau_l > 0$ or the angular displacement exceeds $\tau_\theta > 0$. The rationale behind this is that we want to limit the required memory resources, while uniformly scanning the environment (e.g., avoiding multiple scan recordings with a standing robot). Based on these scans, we construct the *anchor points* that will be used for the repeat phase. An anchor point $a_k = (l_k, \mathbf{x}(t_k))$ consists of a laser scan l_k with the associated robot pose $\mathbf{x}(t_k)$, computed via Eq. (3). Fig. 2 shows two anchor points along a recorded trajectory. During the repeat phase all available laser scans l_j are used. They are matched against the scans associated with the anchor points to compute the feedback error.

IV. Repeating the taught trajectory

In this section, we explain how the error signal is computed and utilized in our trajectory tracking controller. The proposed task is to regulate to zero, at all times, the error between the current and the taught reference poses:

$$\mathbf{e}(t) = \mathbf{x}(t) \ominus \mathbf{x}^*(t)\,, \tag{4}$$

where $\ominus$ is the reverse compounding operator [20].

The computation of this error signal requires precise localization of the robot at time t, i.e., a precise estimate of $\mathbf{x}(t)$. However, as mentioned before, one of the main objectives of this paper is the realization of an intuitive map-free navigation procedure. An alternative to map-based localization consists of estimating $\mathbf{x}(t)$ from odometry alone, i.e., *dead reckoning*. However, this approach is subject to a drift, which becomes significant over long paths.

To deal with this issue, we exploit the anchor points introduced in the previous section. More specifically, during navigation, we rely on the anchor point, noted a, closest to the current robot pose. Fig. 3 shows a schematic example with the actual robot pose indicated by a shaded dart and the reference pose on the taught trajectory indicated by a hollow dart. Here, anchor point a_k is selected as the nearest to the robot. In the following, we drop the index k for readability.

Given the current anchor point $a = (l_a, \mathbf{x}_a)$, two steps are required to estimate the task error $\mathbf{e}(t) = \mathbf{e}_a(t)$. First, we define the reference offset $\mathbf{o}_a(t)$ between the reference pose $\mathbf{x}^*(t)$ and the pose of a at time t as:

$$\mathbf{o}_a(t) = \mathbf{x}^*(t) \ominus \mathbf{x}_a\,. \tag{5}$$

This vector is outlined in Fig. 3 by the arrow pointing to the hollow dart. Second, a scan matcher compares the current laser scan with the anchor point scan. Given anchor point $a = (l_a, \mathbf{x}_a)$ and the current laser scanner reading l, the output of the scan matcher is the offset between the respective robot poses at which the scans were taken:

$$\mathbf{s}_a(t) = \mathbf{x}(t) \ominus \mathbf{x}_a\,. \tag{6}$$

This vector is the arrow pointing from the anchor to the shaded dart in Fig. 3. We initialize the scan matcher by compounding the previously computed offset with the robot motion, to ensure quick convergence to the correct minimum.

Analyzing Eq. (5) and Eq. (6), we see that the reference offset $\mathbf{o}_a(t)$ and the measured one $\mathbf{s}_a(t)$ now represent the reference pose $\mathbf{x}(t)$ and the estimated pose $\mathbf{x}^*(t)$ in the reference frame of the same anchor point a. Combining these equations with Eq. (4), we obtain the final expression required for the feedback control:

$$\mathbf{e}(t) = \mathbf{s}_a(t) \ominus \mathbf{o}_a(t)\,. \tag{7}$$

This vector is represented by the red arrow connecting the two darts in Fig. 3. This error measure does not require a precise global grid map, but relies on the closest anchor point. Therefore, drift cannot accumulate between more than two consecutive anchor points. To choose the closest anchor, we keep track of the offsets between subsequent anchors during teaching. This way, given the offset to the current anchor, we can determine the offset to the next few anchor points and switch to the closest one.

A. Controllers for repeating the taught trajectory

The task error defined above can be used by a cornucopia of controllers and the achievable tracking performance will obviously depend on the chosen controller and on its parametrization. The choice of the controller is orthogonal to the proposed feedback error scheme and out of the focus of this work. Nevertheless, to test our feedback control scheme, we implemented two classical controllers, one for holonomic, and one for nonholonomic kinematics.

For both controllers, we use the velocities $\mathbf{v}^*(t_i)$, recorded during teaching, as the feedforward part. For $t \in [t_i, t_{i+1}]$, we get $\mathbf{v}^*(t)$ by linear interpolation of $\mathbf{v}^*(t_i)$ and $\mathbf{v}^*(t_{i+1})$. For the holonomic robot, we employ linear feedback:

$$\mathbf{v} = \mathbf{v}^* - \Lambda\mathbf{e}\,, \tag{8}$$

with Λ a positive definite gain matrix. Plugging this into the derivative of (4) yields $\dot{\mathbf{e}} = -\Lambda\mathbf{e}$, for which, as desired, $\mathbf{x}^*$ is a globally asymptotically stable equilibrium.

For the nonholonomic robot we use:

$$\begin{cases} v_x = v_x^* \cos e_\theta - \lambda_x e_x \\ v_\theta = v_\theta^* - \lambda_y e_y - \lambda_\theta e_\theta\,, \end{cases} \tag{9}$$

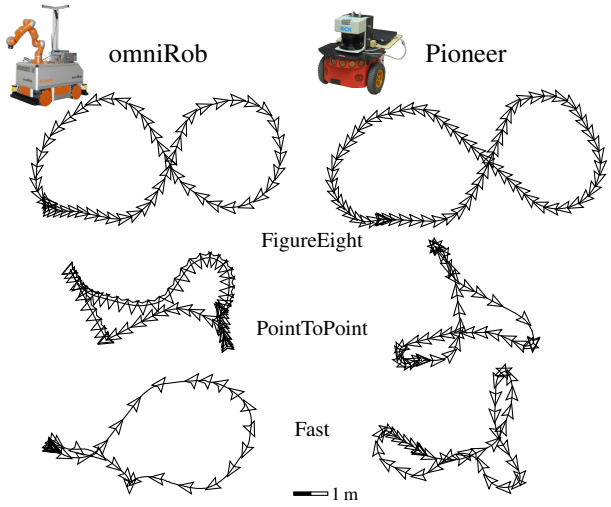

Fig. 4. The six different taught reference trajectories from our experiments. The darts indicate robot configurations and are drawn every second to convey velocity. All paths are drawn with the scale indicated in the bottom.

with λ_x, λ_y and λ_θ positive scalar gains. This controller is known to give good performance for trajectory tracking [19].

For the implementation on our robots, we choose odometry as the heartbeat of our controllers. As it is not synchronized with the laser on our robots, we compound the offset for the last processed scan with the subsequent motion of the robot to retrieve the offset for the current odometry reading.

V. Experiments

We implemented the proposed algorithms in C++ and tested them on two real robots: a holonomic KUKA omniRob (shown in Fig. 1) and a differential drive Pioneer P3-DX robot (shown in Fig. 4). Both robots are equipped with laser scanners, calibrated to the robot center. The omniRob is equipped with two Sick S300 with a 270 degree field of view and 541 beams. The scanner on the Pioneer is a Sick LMS 291 with a 180 degree field of view and 181 beams. To highlight the performance of the proposed system, we chose to use only the front laser of the omniRob. The approach, however, can easily be extended to multiple laser scanners.

For scan matching we rely on a variant of the iterative closest point algorithm with a point-to-line metric as proposed by Censi [21]. We chose this approach for its high accuracy and its ability to work in complex environments with a moderate amount of changes [6].

The environment in which the experiments were conducted is shown in Fig. 1. The ground truth for our evaluations is provided by a MotionAnalysis motion capture studio with 10 Raptor-E cameras. During both teaching and repeating, the system tracks our marker-equipped robots at 100 Hz, typically with sub-millimeter precision. We repeat the trajectories, using respectively controllers (8) for the omniRob, and (9) for the Pioneer, with the error calculated using Eq. (7). We compare the performance with the same controllers when $\mathbf{e}$ is derived from Monte Carlo localization in a global grid map [6], and from wheel encoders alone.

To account for communication and execution delays encountered with our robots, we retrieve the feedforward component from the taught trajectory with a lookahead of Δt_{delay} into the future, i.e., we use $\mathbf{v}^*(t + \Delta t_{delay})$ instead of $\mathbf{v}^*(t)$ in controllers (8) and (9). Unless stated otherwise, we used the following parameters for our controllers: for the omniRob, the gain matrix was set to $\Lambda = \mathrm{diag}\,(2.0, 2.0, 1.0)$ and $\Delta t_{delay} = 0.15$, whereas for the Pioneer we used $\lambda_x = 0.6$, $\lambda_y = 14$, $\lambda_\theta = 3.5$ and $\Delta t_{delay} = 0.2$.

For the omniRob, we inserted anchor points every $\tau_l = 0.07\,\mathrm{m}$ or $\tau_\theta = 0.05\,\mathrm{rad}$ (2.9 degrees) along trajectories. For the Pioneer, laser data was available at a higher frequency allowing to insert anchor points approximately every $\tau_l = 0.03\,\mathrm{m}$ or $\tau_\theta = 0.05\,\mathrm{rad}$ (2.9 degrees).

To assess the quality of trajectory reproduction with our proposed error design from Eq. (7), we compare the motion capture studio recordings of the repetitions with the recordings of the user demonstrations. Since sufficiently accurate time alignment of capture data from teach and reference trajectories was not available, we compute the minimal distance to the polyline path given by the captured points of the reference trajectory as error measure. We then report statistics on these minimal distances over multiple repetitions. To show that our approach is indeed able to track a trajectory and not only a path, we manually aligned the recordings for one set of runs to also show the evolution of the error over time.

A. Experimental setup

We test our approach on three different reference trajectories for the omniRob and the Pioneer, yielding a total of six different trajectories. The reference trajectories as recorded by our motion capture studio are shown in Fig. 4 and are named *FigureEight*, *PointToPoint*, and *Fast* for future reference. The trajectories last between 36 and 70 seconds with velocities of up to 1 m/s (omniRob) and 0.5 m/s (Pioneer).

The chosen trajectories aim at representing typical navigation patterns for different tasks. The FigureEight represents a general navigation pattern in which the robot is required to navigate smoothly for long distances, or tasks such as patrolling or monitoring, in which continuous operation is needed. The reference trajectory PointToPoint is less smooth than the FigureEight and contains turns on the spot. The idea behind this type of trajectory is to represent typical pick-and-place tasks or more general point-to-point navigation, in which the robot is required to reach a predetermined location, perform a manipulation task and then move to the next location. Finally, we wanted to test the performance of the proposed system when the robot travels near its maximum controllable speed and with more dynamic maneuvers. This test is performed with the Fast reference trajectory.

We also compare our feedback error scheme to the one based on Monte Carlo localization (MCL) and to the one based only on wheel encoders. For these schemes, we only change the way of measuring the offset $\mathbf{s}_a$ to the current anchor point. For the MCL scheme, we built a global gridmap of the robot environment with a resolution of 0.01 m and augmented the anchor points with the respective MCL estimate of the robot's location during the teach phase.

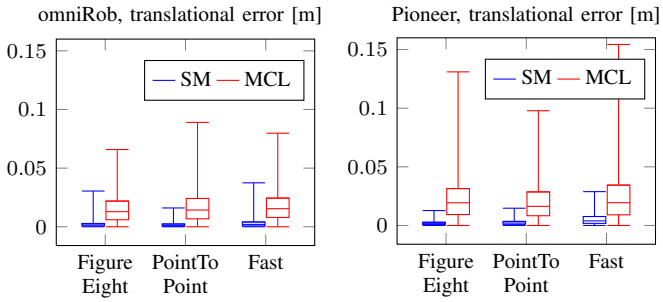

Fig. 5. Boxplots for the translational tracking error (see text) when reproducing the taught reference trajectories shown in Fig. 4. In addition to the lower and upper quartile, the plots show the median and maximum error over all runs. The data is gathered over 98 runs for the proposed scan matching based feedback error (SM) and over 92 runs for the Monte Carlo localization based feedback error (MCL).

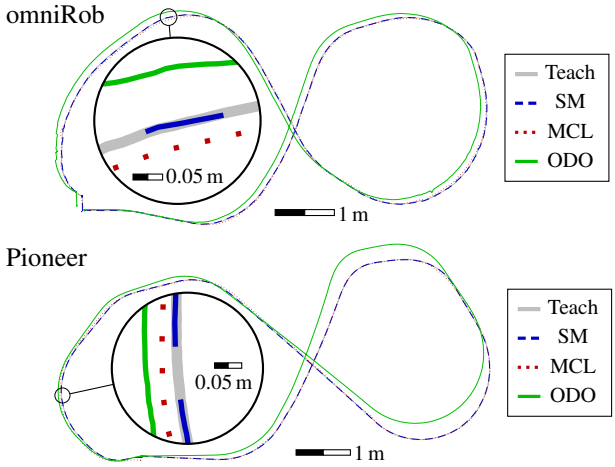

Fig. 6. Taught and repeated FigureEight trajectories for each approach to feedback error computation with the omniRob and the Pioneer.

During repetition, the system then determines $\mathbf{s}_a$ through the current MCL localization. Note that this latter scheme corresponds to the widespread approach for laser-based navigation. For repeating the trajectory with odometry alone, only one anchor point at the start of the trajectory is used. The offset to this anchor is determined by keeping track of the relative motion of the robot since the start.

B. Comparison with Monte Carlo localization

Statistics over the minimum distance error measure (introduced above) for trajectory reproduction are shown in Fig. 5 for both robots and all reference trajectories. In total, 190 runs have been executed to gather the data reported in this figure. To avoid biasing the evaluation with the user errors made in positioning the robot at the start point of the trajectory, we discard data generated in the first 1.0 s (omniRob) and 2.0 s (Pioneer) of the trajectory reproduction.

Fig. 6 shows example paths taken by the omniRob and the Pioneer for the FigureEight reference trajectories. For these particular examples, Fig. 7 depicts the norm of the tracking error over time as a result of a manual time alignment of motion capture data from the teach and repeat phases. The low errors over the whole trajectory indicate that we are indeed tracking the reference trajectory and not just the geometric

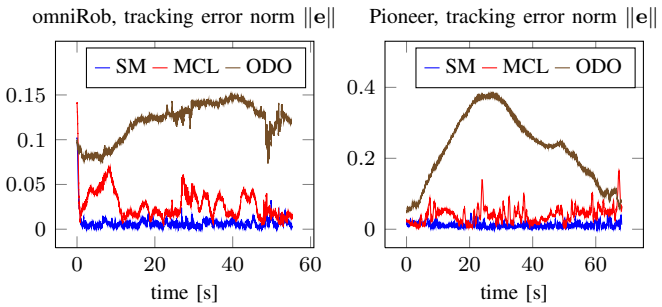

Fig. 7. Norm of the tracking error over time for FigureEight for omniRob (left) and Pioneer (right) as measured by the motion capture studio for the runs also shown in Fig. 6.

path, i.e., that our approach is also accurate with respect to the demonstrated velocities. Typically, the errors computed at a specific point in time are higher than the ones considering only the geometry of the tracked path. Moreover, the figure also includes the very start of the trajectory reproduction. As can be seen, especially in the omniRob experiments, there are higher errors at the beginning, which were introduced by the user who could not accurately position the robot.

The plots show that both our approach and a Monte Carlo localization based computation of the feedback error (MCL) are able to track the trajectory with considerable accuracy. However, when analyzing the error, we can see that our scheme achieves an accuracy that is higher by one order of magnitude compared to MCL. For the Pioneer PointToPoint experiment, the higher errors made it necessary to adapt the controller gains to $\lambda_x = 4.0, \lambda_\theta = 1.0$ for MCL, as the original values lead to oscillation and worse results.

As the plots show, the proposed approach is able to reproduce the taught trajectories with high accuracy and precision, not only on average but also in the worst case. These results are general and do not depend on the type of the robot or on the specific path driven. In other words, our feedback error scheme is able to effectively solve the desired task without relying on a globally consistent metrical map. The reference trajectories taught by the user are tracked with high precision for both, a holonomic and a differential drive robotic platform. The higher tracking errors of the MCL tracking scheme can be attributed to the discretization introduced by the grid map and to the local residual inaccuracies resulting from the optimization process for establishing global consistency in the map.

C. Comparison with wheel encoders

For further analysis and insight, we also compare against the tracking performance of 31 runs with a purely odometry based feedback error computation (ODO) for the FigureEight experiments. The drift of odometry becomes apparent in Fig. 6 and explains the substantially larger errors in the boxplots in Fig. 8 and in the tracking error in Fig. 7. That the errors still remain moderate can be explained by the path shape of the FigureEight experiment: the errors introduced by odometry drift tend to cancel out by the maneuvers, leading to longer periods of moderate tracking error. The analysis of pure wheel encoder based tracking also confirms that the

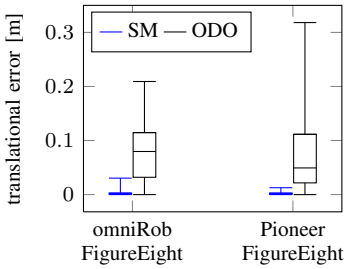

Fig. 8. Boxplots comparing tracking errors for our scan matching based feedback error computation (SM) and feedback error generation based on odometry only (ODO) for the FigureEight trajectories. The still moderate errors for odometry are explained by the trajectory shapes (see text).

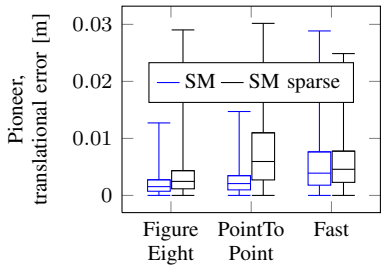

Fig. 9. Boxplot comparing the performance of our approach (SM) against a version with a substantially reduced number of anchor points (SM sparse).

high accuracy achieved by our approach is not simply caused by decent wheel encoders and floor conditions.

D. Evaluation with respect to anchor point density

In this experiment we analyze the effects of substantially reduced anchor point density along the taught trajectory. The experiment was performed with seven runs each for the Pioneer reference trajectories with anchor points reduced to occur approximately every $\tau_l = 0.5\,\text{m}$ or $\tau_\theta = 0.5$ rad (29 degrees). This decreased the number of anchor points from 810 to 46 for FigureEight, from 567 to 40 for PointToPoint, and from 444 to 41 for Fast. The results of this evaluation are shown in Fig. 9. Despite the reduction of anchor points by an order of magnitude, the tracking errors increase only moderately. The increase is the highest for the PointToPoint trajectory which contains prominent turns on the spot. Here, the new rotational offset of 29 degrees between anchor points has the highest impact, since it severely reduces the overlap between laser scans for the scan matcher.

The stable results for reference trajectories with substantially sparsified anchor points show that the high performance of the proposed approach is due to the underlying principle and cannot simply be explained by a huge amount of data expended to tackle the task.

VI. Conclusion

In this paper, we presented a framework for intuitive specification of teach-and-repeat robot navigation tasks. With our approach, the user only needs to demonstrate the desired trajectory once during a teaching phase and does not need to build a globally consistent map of the environment. Our approach stores 2D laser data and raw odometry in anchor points during teaching and then uses them for feedback control of the robot during the repeat phase. We evaluated our approach with real robots and compared it to alternative ways of feedback error generation for several trajectories. Our approach is able to track the taught trajectories with millimeter accuracy and outperforms feedback generation by Monte Carlo localization based on a grid map.

References

[1] H. Mäkelä and T. von Numers, "Development of a navigation and control system for an autonomous outdoor vehicle in a steel plant," *Control Engineering Practice*, vol. 9, no. 5, pp. 573–583, 2001.

[2] S.-Y. Leea and H.-W. Yang, "Navigation of automated guided vehicles using magnet spot guidance method," *Robotics and Computer-Integrated Manufacturing*, 2012.

[3] M. Beinhofer, J. Müller, and W. Burgard, "Near-optimal landmark selection for mobile robot navigation," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2011.

[4] D. Meyer-Delius, M. Beinhofer, A. Kleiner, and W. Burgard, "Using artificial landmarks to reduce the ambiguity in the environment of a mobile robot," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2011.

[5] M. Vitus and C. Tomlin, "Sensor placement for improved robotic navigation," in *Proc. of Robotics: Science and Systems (RSS)*, 2010.

[6] J. Röwekämper, C. Sprunk, G. D. Tipaldi, C. Stachniss, P. Pfaff, and W. Burgard, "On the position accuracy of mobile robot localization based on particle filters combined with scan matching," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2012.

[7] A. Cherubini, M. Colafrancesco, G. Oriolo, L. Freda, and F. Chaumette, "Comparing appearance-based controllers for nonholonomic navigation from a visual memory," in *Workshop on safe navigation in open and dynamic environments: application to autonomous vehicles at the IEEE Int. Conf. on Robotics & Automation*, 2009.

[8] Y. Matsumoto, M. Inaba, and H. Inoue, "Visual navigation using viewsequenced route representation," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1996.

[9] E. Royer, M. Lhuillier, M. Dhome, and J.-M. Lavest, "Monocular vision for mobile robot localization and autonomous navigation," *Int. Journal of Computer Vision*, vol. 74, no. 3, pp. 237–260, 2007.

[10] T. Goedemé, M. Nuttin, T. Tuytelaars, and L. V. Gool, "Omnidirectional vision based topological navigation," *Int. Journal of Computer Vision*, vol. 74, no. 3, pp. 219–236, 2007.

[11] A. A. Argyros, K. E. Bekris, S. C. Orphanoudakis, and L. E. Kavraki, "Robot homing by exploiting panoramic vision," *Journal of Autonomous Robots*, vol. 19, no. 1, pp. 7–25, 2005.

[12] A. Cherubini and F. Chaumette, "Visual navigation with a time-independent varying reference," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009.

[13] A. Kelly, B. Nagy, D. Stager, and R. Unnikrishnan, "An infrastructure-free automated guided vehicle based on computer vision," *IEEE Robotics and Automation Magazine*, 2007.

[14] P. Malheiros, P. Costa, A. Moreira, and M. Ferreira, "Robust and real-time teaching of industrial robots for mass customisation manufacturing using stereoscopic vision," in *Industrial Electronics. IECON '09. 35th Annual Conference of IEEE*, 2009.

[15] C. McManus, P. Furgale, B. Stenning, and T. D. Barfoot, "Lighting-invariant visual teach and repeat using appearance-based lidar," *J. of Field Robotics*, vol. 30, no. 2, pp. 254–287, 2013.

[16] P. Furgale and T. D. Barfoot, "Visual teach and repeat for long-range rover autonomy," *J. of Field Robotics*, vol. 27, no. 5, pp. 534–560, 2010.

[17] J. Marshall, T. Barfoot, and J. Larsson, "Autonomous underground tramming for center-articulated vehicles," *J. of Field Robotics*, vol. 25, no. 6-7, pp. 400–421, 2008.

[18] C. Sprunk, B. Lau, and W. Burgard, "Improved non-linear spline fitting for teaching trajectories to mobile robots," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2012.

[19] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer Verlag, 2009.

[20] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous robot vehicles*, I. J. Cox and G. T. Wilfong, Eds. Springer, 1990, pp. 167–193.

[21] A. Censi, "An ICP variant using a point-to-line metric," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2008.