SLAM Using Scan Matching

Hadi Givehchian, Pulak Sarangi, Sina Shahsavari
A53284450, A53097118, A53278804

I. Abstract

Localizing an agent in an unknown environment has received lots of attention from both industry and academia for years. A variety of sensors have been built in autonomous cars and robots to enable localization. Lidar sensors, in particular, provide dense and accurate point clouds. A variety of algorithms have been proposed to exploit these Lidar scans to localize the agent. Iterative Closest Point (ICP) family can estimate the pose variation of the agent between two different scans which can be further used to localize the agent given the starting point. In this work, we implement the PL-ICP algorithm to localize the agent. Moreover, we found PL-ICP sometimes does not estimate pose variation on real-world data correctly which can devastate the estimated trajectory. To mitigate this problem, we propose a simple yet effective outlier rejection mechanism which improves the pose estimation. Furthermore, it turns out that PL-ICP is highly sensitive to initialization. To this end, we incorporate an initialization mechanism to the algorithm to avoid unreasonable estimates. We extensively evaluate our implementation and modifications to the algorithm on both synthetic and real-world data (Oxford robotcar dataset) and compare our results with the scan matching implementation provided in the Matlab navigation toolbox.

II. Introduction

Over the last few years, the range of applications of mobile robots has significantly increased. In order to be fully functional, an autonomous mobile robot must be capable of navigating safely through an unknown environment while simultaneously carrying out other tasks. To reach this goal, the knowledge of its location is essential for the robot and helps it to make correct decisions about future actions. This crucial requirement leads to one of the most fundamental tasks which a robot should undertake, known as Robot Localization. Robot localization is the process of determining the state of a mobile robot with respect to its surrounding environment. Typically a robot is equipped with sensors that observe the environment as well as monitor the robot motion. Therefore, localization aims to estimate the robot position and orientation using information gathered from those sensors. The robot localization techniques should be robust to noisy observations and also generate a measure of uncertainty of its estimate. Moreover, they should be sufficiently robust to ensure an accurate estimates in unknown environments.

One famous technique to perform robot localization task is to employ sensors which monitor their own motion (e.g., wheel encoders sensors and Inertial Navigation System, INS) and can compute an estimate of the robot location relative to where it started given a mathematical model of the motion. This is known as Odometry or Dead Reckoning. As the robot navigates in its environment, due to errors present in the sensor measurements and the motion model, the robot location estimates obtained from odometry become more and more unreliable. Errors in odometry estimates can be corrected when the robot can observe its environment using information gathered by other sensors such as Lidar, Radar sensors or Global Positioning System (GSP).

Using GPS data can be beneficial for correcting the estimation error of odometry method. However, this approach suffers from two major deficiencies as well. Firstly, similar to INS sensors the large error in the data comes from GPS is unavoidable. Moreover, in many cases such as indoor localization, there is no GPS signal and it is not possible to correct the error of INS system without GPS data. Consequently, other sensors such as Lidar based sensors are exploited to scan the environment and deal with the localization problem.

A popular approach that has been employed to enable localization using Lidar scan data, is scan-matching. The objective of scan-matching is finding the relative translation and rotation between two consecutive scans. The scan-matching algorithms translate and rotate the actual scan to make the best overlap of the reference scan. If a scan-matching algorithm can provide a decent matching between points in the two consecutive scans, then the estimated translation and rotation is the relative distance and rotation between the reference position and the actual position. The relative distance and rotation can further be used to update the position of the robot and track its trajectory.

In this project we have considered a scenario in which a mobile agent is located in an unknown environment. Our goal is to a robustly and accurately estimate its location as well as tracking the trajectory it traverses. To realize this goal, we have implemented PL-ICP algorithm to estimates relative rotation and translation of the agent and then determine its trajectory. To enhance the performance of the algorithm, we have employed several approaches, such as outlier detection module and different mechanisms for initialization.

III. Prior Works

A. Localization

Mobile robot localization approaches can be distinguished by the kind of data available to the robot: vision sensors [1], sonar sensors [2], odometry [3], GPS [4], and ultrasonic beacons [5]. To handle position uncertainties associated with a navigating robot, a variety of approaches have been proposed in the literature some of them use a statistical approach, for example, Chatila and Laumond [6], Smith and Cheeseman [7], Kosaka and Kak [8], Crowley [9], Leonard and Durrant-Whyte [10], Watanabe and Yuta [11], Burlina, DeMenthon and Davis [12], and Matthies and Shafer [13].
In addition, several works in this field have considered Extended Kalman Filter (EKF) when there exists some prior knowledge about the. movement, such as map of the environment. A. de la Escalera, et. al in [14], have used the EKF algorithm to combine the information of a laser sensor and a camera to perform a dead reckoning based estimation. The experimental results show that the algorithm can work in real-time, and it actualizes the position of the robot continuously. The references [15], [16], [17], [18], [19], also use the EKF algorithm to process the sensor data, and at the same time, predict and update the position and orientation of the mobile robot.

Neural network based approaches also have been investigated by several researchers, for example, Meng and Kak in [20], presented a NEURO-NAV system for mobile robot navigation. In the NEURO-NAV, a feedforward neural network, which is driven by transformation of some features extracted from visual sensor's output, is employed to obtain the approximate relative angles between the heading directions of the robot. In [21], a neural network based camera calibration method was presented for the global localization of mobile robots using monocular vision. In [22], authors employed neural network to perform a fusion for data coming from different types of sensors.

B. Scan Matching

Several techniques have been proposed for scan-matching in the literature. ICP [23], [24] is one of those methods which has been used pervasively in scan matching. In ICP, each point in the current scan is associated with the reference scan according to a distance metric (most commonly Euclidean distance). Specialization of the ICP in the robotics community have flourished by [25], [26], [27], [28], [29]; Particularly, Lu and Milios [25] describe two scan matching methods based on ICP. The first method considers the rotational and translational components separately: alternately fixing one, then optimizing the other. Given rotation, least-squares is used for optimizing translation. Their second method, dubbed Iterative dual correspondence (IDC), combines two ICP-like algorithms with different point-matching heuristics.
Since the introduction of ICP by Chen and Medioni [30] and Besl and McKay [23], many ICP variants have been investigated, because the core algorithm can be slightly modified in many ways: which subset of points to use, how to define the error metric, how to discard outliers, etc. Iterative Closest Line or point to line ICP (PL-ICP) [31], [32], [33], is one of ICP versions which is similar to the vanilla ICP, except that instead of matching current points to the reference points, the current points

are matched to lines extracted from the reference points. The simplest PL-ICP vesrions interpolate lines between each pair of adjacent Lidar points, but this is undesirable in many cases (e.g. depth discontinuities). Alternatively, heuristic methods can be used to determine which pairs of points are likely to be part of a connected surface, or lines/splines can be extracted from larger sets of points [34], [31], [33]. In this project we consider the variant of PL-ICP introduced by Censi et. al, in [32].

The literature is filled with other scan matching heuristics as well. These include using polar coordinates [35] and Feature-based methods, such as Normal Distribution Transform (NDT) [31], [33], [36]. In the NDT [36], the 2D plane will be subdivided into cells. A normal distribution is assigned to each cell, which locally models the probability of measuring a point. That can be used to match another scan using Newton's algorithm. Thereby, there is no need to establish explicit correspondences. Moreover, Hough transforms [37] and histograms [38], [39] are some other heuristic methods have been used. Weiss and Puttkammer [40] used angular histograms to recover the rotation between two poses. Then x and y histograms, which were calculated after finding the most common direction were used to recover the translation.

IV. Problem Statement

This work aims at the conventional localization problem in an unknown environment using Lidar range data. More specifically, at each time step $t = 1, 2, 3, ..., T$, the agent measures range data for L different angles using Lidar and obtains the measurements $\{(x_1^t, y_1^t), ..., (x_L^t, y_L^t)\}$ for these L angles. Having these measurements, the goal is to find the location and trajectory of the agent from the starting point $t = 1$ to the end time step $t = T$. To achieve this goal, the first step is to estimate the change in the pose of the agent between two consecutive time steps. Specifically, for all $t = 2, ..., T$, we should find the rotation θ_t and translation t_t between the time steps $t-1$ and t using the measurements $\{(x_1^{t-1}, y_1^{t-1}), ..., (x_L^{t-1}, y_L^{t-1})\}$ and $\{(x_1^t, y_1^t), ..., (x_L^t, y_L^t)\}$. Thereafter, these roto-translations can be successively applied to yield the trajectory of the agent.

V. Methodology

As mentioned earlier, the first step to obtain the trajectory of the agent is to estimate the rotation θ_t and translation $\mathbf{t}_t$ between the time steps $t-1$ and t for all $t = 2, ..., T$. To this end, we employ the PL-ICP algorithm which is a point-to-line variant of the ICP algorithm. Given two sets of points, ICP solves a surface matching problem yielding the rotation and translation between those two sets of points. These rotations and translations can be further used to obtain the trajectory of the agent. Hence, we first elaborate on ICP and PL-ICP algorithm, and after that, we build the trajectory using the rotations and translations from the PL-ICP algorithm.

A. Iterative Closest Point (ICP)

Having two scans with unknown data association, ICP solves a surface matching problem. In particular, given a reference surface S_{ref} and a set of points $\{\mathbf{p}_i\}$, ICP tries to iteratively find (θ, t) that minimizes the distance of the points $\mathbf{p}_i$ roto-translated by rotation θ and translation $\mathbf{t}$ to their projection on surface S_{ref}. In other words, ICP iteratively solves the following optimization problem for which a closed-form solution does not exist in general:

$$\min_{\mathbf{t},\theta} \sum_{i=1}^{L} \|\mathbf{R}(\theta)\mathbf{p}_i + \mathbf{t} - \mathcal{P}_{S_{ref}}(\mathbf{R}(\theta)\mathbf{p}_i + \mathbf{t})\|_2 \quad (1)$$

where $\mathcal{P}_{S_{ref}}$ denotes the Euclidean projector on S_{ref}. ICP starts from an initialization of (θ, t). Then until convergence, it successively uses the values of $(\theta, \mathbf{t})$ from k-th iterations to compute the values of $(\theta, \mathbf{t})$ for the $k+1$-th iteration. Specifically, at each iteration k, ICP solves the following problem for which a closed form solution is available:

$$\min_{\mathbf{t}_k,\theta_k} \sum_{i=1}^{L} \|\mathbf{R}(\theta_k)\mathbf{p}_i + \mathbf{t}_k - \mathcal{P}_{S_{ref}}(\mathbf{R}(\theta_{k-1})\mathbf{p}_i + \mathbf{t}_{k-1})\|_2 \quad (2)$$

Although ICP uses the aforementioned Euclidean point-to-point distance, other distances or error metrics might be used. Furthermore, it is not necessary to use all points in a scan in the matching process. Besides, having outliers in scans or the matching process can significantly degrade the performance. Moreover, several weaknesses have become evident when using ICP for scan-matching. For instance, it may converge to a wrong solution for large initial errors; correspondence search is expensive; convergence is slow, and occlusions and outliers are frequent. As a result, there has been a huge body of work on which subset of points to use, how to define the error metric and how to discard outliers, each of which results in a variant of ICP algorithms. In this work, we use a fast point-to-line version of ICP called PL-ICP.

B. PL-ICP

PL-ICP [32] is a point-to-line variant of ICP. In fact, it employs the following point-to-line metric instead of the point-to-point metric used in the original ICP algorithm:

$$\min_{\mathbf{t}_{k+1},\theta_{k+1}} \sum_i \Big(\mathbf{n}_i^T\big[\mathbf{R}(\theta_{k+1})\mathbf{p}_i + \mathbf{t}_{k+1} - \mathcal{P}_{S_{ref}}(\mathbf{R}(\theta_k)\mathbf{p}_i + \mathbf{t}_k)\big]\Big)^2 \quad (3)$$

where n_i is the normal vector to the surface at the projected point. While ICP in general exhibits a linear convergence, the point-to-line metric makes the convergence of PL-ICP quadratic in the case of a zero-residual problem, and a good initialization. PL-ICP algorithm is described in details in algorithm 1.

It is worth mentioning that the closed form solution for error function $E(\theta_{k+1}, t_{k+1})$ (alg.1) can be obtained by reducing to quadratic form. The error function minimization problem can be written as follows

$$\min_{\mathbf{t}_{k+1},\theta_{k+1}} \sum_i \left(\mathbf{P}_i\mathbf{x}_{k+1} - \mathbf{q}_1^{(i)}\right)^T \mathbf{n}_i\mathbf{n}_i^T \left(\mathbf{P}_i\mathbf{x}_{k+1} - \mathbf{q}_1^{(i)}\right) \quad (4)$$

where $\mathbf{x}_{k+1} = [x_{t_{k+1}}, y_{t_{k+1}}, cos(\theta_{k+1}), sin(\theta_{k+1})]$ includes the roto-translation parameters, $\mathbf{q}_1^{(i)}$ is as defiined in alg.1 and

$$\mathbf{P}_i = \begin{bmatrix} 1 & 0 & x_{p_i} & -y_{p_i} \\ 0 & 1 & y_{p_i i} & -x_{p_i} \end{bmatrix} \quad (5)$$

By ignoring the constant terms, the error function to be minimized can be written as $E'(\theta_{k+1}, \mathbf{t}_{k+1}) = \mathbf{x}^T\mathbf{P}\mathbf{x} + \mathbf{g}^T\mathbf{x}$ where

$$\mathbf{P} = \sum_i \mathbf{P}_i^T\mathbf{n}_i\mathbf{n}_i^T\mathbf{P}_i, \quad \mathbf{g} = -\sum_i 2\mathbf{q}_1^{(i)T}\mathbf{n}_i\mathbf{n}_i^T\mathbf{P}_i \quad (6)$$

Furthermore, since $cos^2(\theta_{k+1}) + sin^2(\theta_{k+1})$, the optimization problem is constrained to $x_3^2 + x_4^2 = 1$. Hence, we reduced the minimization problem of $E(\theta_{k+1}, \mathbf{t}_{k+1})$ to a constrained quadratic optimization problem which can be solved using Lagrange's multipliers.

C. Trajectory

Having the rotation and translations for all consecutive time steps, obtaining the trajectory is as simple as applying these rotations and translations successively. In fact, having the current location $\mathbf{p}_{t-1}$ at time $t-1$ and rotation θ_t and translation t_t between the points $\{(x_1^{t-1}, y_1^{t-1}), ..., (x_L^{t-1}, y_L^{t-1})\}$ and $\{(x_1^t, y_1^t), ..., (x_L^t, y_L^t)\}$, by representing $\mathbf{p}_{t-1}$ in homogeneous coordinates, the location at the next time step can be computed as

$$\mathbf{p}_t = \mathbf{R}'(\theta_t, \mathbf{t}_t)\mathbf{p}_{t-1} \quad (7)$$

where R' is the 3×3 roto-translation matrix defined as follows

$$\mathbf{R}'(\theta_t, t_t) = \begin{bmatrix} \mathbf{R}(\theta_t) & \mathbf{t}_t \\ \mathbf{0} & 1 \end{bmatrix} \quad (8)$$

Therefore, assuming we know the starting point $\mathbf{p}_0$, the location at each time step $t = 1, ..., T$ can be computed as the following:

$$\mathbf{p}_t = \Big(\prod_{t'=1}^{t} \mathbf{R}'(\theta_{t'}, \mathbf{t}_{t'})\Big)\mathbf{p}_0 \quad (9)$$

VI. Experiments and Results

In this section, we evaluate the performance of our implementation of the PLICP algorithm through a series of experiments on both synthetically generated data as well as a real dataset Oxford [41]. Our experiments focus on the scan matching task that can be used to also track the pose of a mobile agent.

A. Synthetic Data

In the first experiment, we generate a synthetic scan assuming the robot is placed in a static surrounding where the range information is uniform in all directions. We assume that the Lidar has a resolution of 0.5 degrees and has a field of view of 180 degrees. We synthetically transform this scan using a known translation and rotation simulating the effect of a change in the robot pose. Our goal is the infer the relative pose between the two scans and compare the quality of inference for a wide range of transformations. We compare the performance of the PLICP algorithm against another scan matching algorithm the Normal Distributions Transform (NDT) algorithm [36] which is available

Algorithm 1: PL-ICP

Input: Two sets of range measurements for L angles uniformly distributed in range of $[-\pi/2, \pi/2]$
Output: Estimated $\widehat{\theta}$, $\widehat{\mathbf{t}}$
initialization: $\theta_0 \leftarrow$ proper initialization, $\mathbf{t}_0 \leftarrow$ proper initialization
step 1: extract Cartesian's points out of input ranges: $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_L\}$, $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_L\}$,
while $k \leqslant K$ **do**
- Transform the points in the 1st scan: $\mathbf{p}_i^{\omega} = \mathbf{R}(\theta_k)\mathbf{p}_i + \mathbf{t}_k$
- For each point $\mathbf{p}_i^{\omega}$, find the two closest point in the 2nd scan $\{\mathbf{q}_1^{(i)}, \mathbf{q}_2^{(i)}\}$
- Eliminate outliers
- Compute the normal vector, $\mathbf{n}_i$ to the line segment joining points $\mathbf{q}_1^{(i)}$ and $\mathbf{q}_2^{(i)}$
- Minimize the following error function to obtain $\theta_{k+1}, \mathbf{t}_{k+1}$

$$E(\theta_{k+1}, \mathbf{t}_{k+1}) = \sum_i \left(\mathbf{n}_i^T \left[\mathbf{R}(\theta_{k+1})\mathbf{p}_i + \mathbf{t}_{k+1} - \mathbf{q}_1^{(i)}\right]\right)^2$$

$k \leftarrow k+1$
end
return $\widehat{\theta} = \theta_K$, $\widehat{\mathbf{t}} = \mathbf{t}_K$

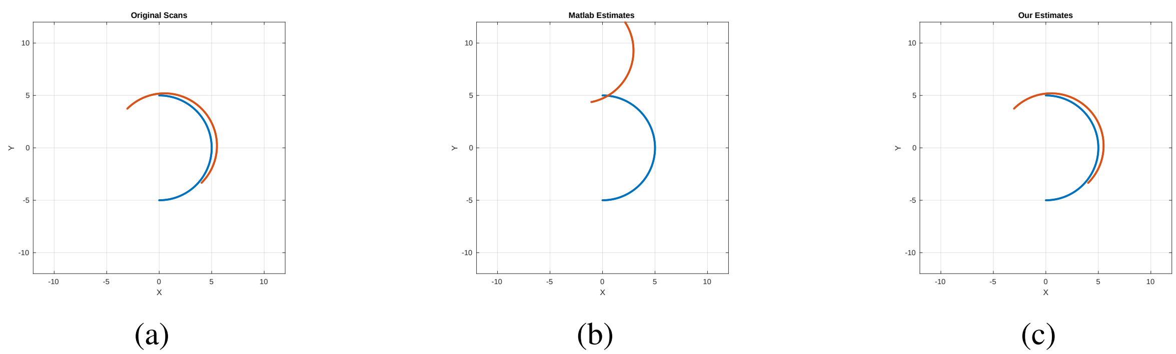


Fig. 1. (a) Ground Truth Transformation (b) Estimated transformation using NDT (c) Estimated transformation using PLICP

in the MATLAB navigation toolbox. In Figure 1 (a), we show the original scan in blue and the corresponding transformed scan which has been rotated by 45 degrees and translated by $[0.5, 0.2]$. It can be clearly seen that, when the rotation is large the NDT algorithm (in Figure 1 (b)) failed to identify the true transformation whereas our algorithm (in Figure 1 (c)) correctly identifies the relative transformation between the two scans.

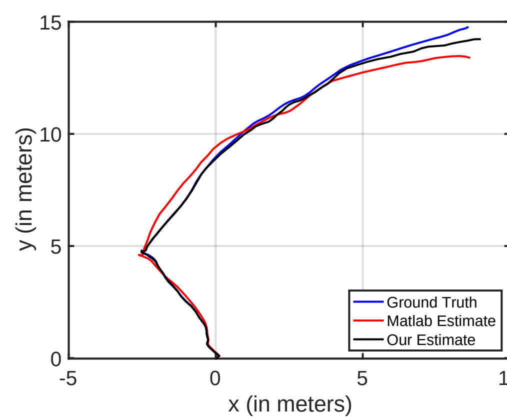


Fig. 2. Comparison of tracking performance using PLICP and NDT (MATLAB) under the constant velocity motion model

In the next experiment, we now acquire a series of scans as the robot is moving in the static environment. Here, our goal is to estimate the trajectory of the robot using the scan measurements obtained from the static surrounding. This can be useful to accurately track the robot measurement when the odometer sensors are unreliable due to slips or other uncertainties. We will assume the robot is moving according to a constant velocity model with the following state-transition model:

$$\mathbf{x}_n = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{n-1} + \begin{bmatrix} 0.5T^2 & 0 \\ 0 & 0.5T^2 \\ T & 0 \\ 0 & T \end{bmatrix} \mathbf{q}_n$$

where $\mathbf{x}_n$ is state vector consisting of robot location and velocity driven by noise $\mathbf{q}_n$. We acquire scans as the object is moving according this model and we estimate its trajectory using only scan matching and compare it to the ground truth trajectory available. The surrounding environment is assumed to have the same uniform range pattern used for experiment 1. It can be seen in Figure 2, the trajectory obtained by our pose estimates matches closely compared to the NDT implementation used in MATLAB. Both the scan matching algorithms are initialized using the estimate of the pose obtained from the previous time instance. For the first step, we initialize both the algorithms with $\mathbf{t} = [0, 0]$ and $\theta = 0$ for a fair comparison.

B. Real Data

In the first experiment on real data, we use a simple real dataset provided in the MATLAB Navigation toolbox. In Figure 3 (a), we show an approximate layout of the environment which the robot is exploring and its trajectory. The data is collected using a sensor with 270 degrees field of view with 1 degree resolution. Our goal is to accurately estimate the trajectory of this robot as it is moving in the indoor environment shown above. We obtain an estimate of the trajectory by assuming the robot started at $[0, 0]$ and we apply the estimated translation and rotation using consecutive scan matches to estimate the position of the robot at the next time instance. In Figure 3 (b), the estimated trajectory obtained using our scan matching algorithm matches with the true robot trajectory shown in Figure 3 (a). In our next experiment, we test the algorithm on real world data from the Oxford Robotcar dataset [41]. The dataset consists of following sensors Cameras, 2-D and 3-D LIDAR and ground truth data obtained using GPS and inertial navigation system (INS). For evaluating our scan matching algorithm, we will use the 3-D LIDAR data and front 2-D LIDAR sensor. This 3-D LIDAR sensor has a 85 degree HFoV, 3.2 degrees VFoV, 4 planes, 12.5Hz, 50m range with a 0.125 degree resolution. On the other hand, the

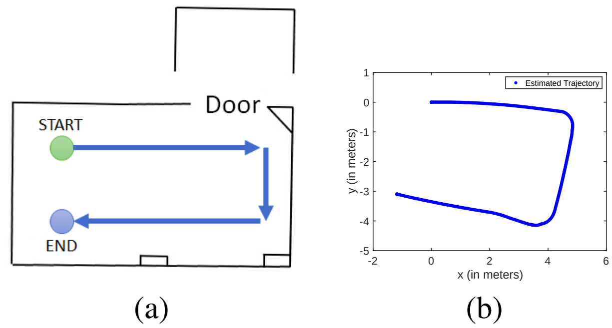
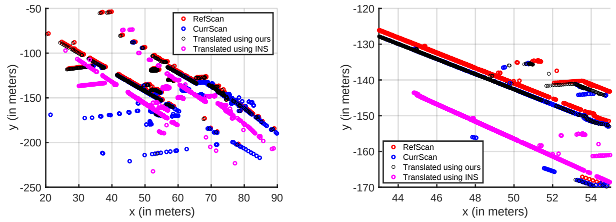


Fig. 3. (a) Layout of the indoor environment and Robot trajectory (b) Estimated trajectory using our Scan matching algorithm

2D LIDAR has 270 degrees FoV with 0.5 degree resolution and 50m range. However, for simplicity we will use only the 2-D information from the 3-D LIDAR data by simply projecting the 3-D points onto two dimensions and discarding points beyond $[-0.5, 0.5]$.

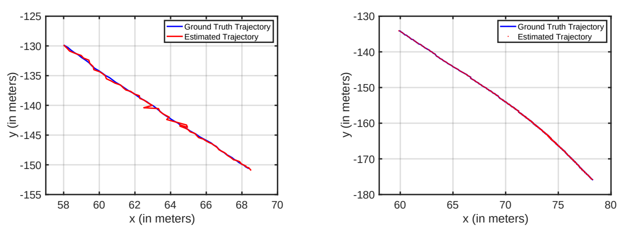
Similar to our synthetic experiments, we begin by evaluating the effectiveness of the algorithm in estimating the relative transformation between two scans acquired between a time interval of 400ms. We also have pose information available from the INS sensor and we compare the estimate obtained using our scan matching algorithm against the measurements from this INS sensor. However, we interpolate the INS measurements to align them to the time instances where the LIDAR data is acquired. We determine the quality of our estimate by transforming the scan using the estimated pose and comparing it with the second scan. We can also obtain similar transformed scan using the measurement of the INS sensor. In Figure 4(a), we show the point clouds of two scans acquired using 3-D lidar after projection onto the x-y plane and we wish to estimate the relative transformation between these two scans. As expected, we can observe in Figure 4 (a)-(b), the rotated scan using our estimate (shown in black) closely matches with the second scan (in blue). It is surprising that the transformed scan obtained using the INS measurements has an offset compared to the second scan. This might be a result of the simple linear interpolation scheme used to temporally align the INS data along with the LIDAR data. Now, we compare the trajectory of the robot using the scan matching algorithm. We estimate the position at a time interval using the position at the previous time and accounting for



(a) (b)

Fig. 4. (a) Comparison of rotated scans obtained using different pose estimates (b) Zoomed in view of the different rotated scans

the roto-translation estimated by our algorithm. In Figure 5, compare the trajectory obtained from INS data and using the estimated roto-translations from the 3-D (left) and 2-D (front) LIDAR data. Since the sampling frequency for both sensors is different, we perform the estimation between different starting and end times therefore the trajectories are not identical. Finally, we also tried a similar experiment



(a) (b)

Fig. 5. (a) Trajectory estimated using 3-D LIDAR sensor (b) Trajectory estimated using 2-D LIDAR sensor

on the Dataset [29] and observed consistent result as shown in Figure 6.

VII. Discussion

In this section, we briefly discuss some of the novel components that were proposed by our group in addition to the original paper [32]. In the original paper, the matching step was performed by

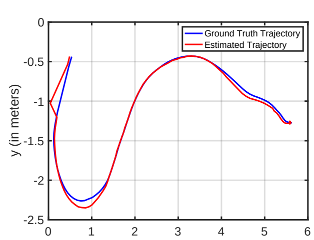
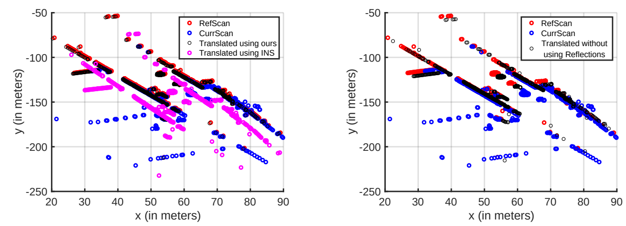


Fig. 6. Tracking performance on dataset from [29]

performing a simple comparison of the Euclidean norm between the points. However, such a simplistic approach does not perform well in case of real data due to potential incorrect matches. In the oxford dataset, in addition to the range information, we also have access to the reflectance value. Our idea was to rely on a combined metric for the outlier rejection module which considers weights for matched pairs based on the difference of reflectance value besides the euclidean distance to rejects poorly matched pairs among the points. We demonstrate the effectiveness of this additional step in further boosting our results. As shown in Figure 7 (b), when the scan



(a) (b)

Fig. 7. (a) Scan matching performance with reflectance based outlier detection (b) Scan matching performance without reflectance based outlier detection

is transformed using the estimate obtained without our outlier detection system (which uses reflectance information for matching) leads to a poor estimate in contrast to Figure 7 (a).

Our second modification focuses on combating the sensitivity of PLICP algorithm to initialization. Due to the non-convex nature of the optimization problem, the quality of the solution is a function of the initialization. In most cases, when estimating the robot trajectory we observed that initializing the scan matching algorithm using the previous estimate works reasonably well. However, in certain cases, it leads to a solution which is far from being reasonable. We included an additional re-initialization step which in every iteration of the algorithm checks the quality of the solution and if it violates a certain threshold condition then the algorithm is re-initialized. This step was critical in improving the performance of obtaining the robot trajectory.

VIII. Conclusion

In this work, we focused on solving the problem of location estimation as well as tracking for a mobile agent in an unknown environment. We adopted a scan matching based approach to estimate relative rotation and translation of the agent between consecutive Lidar scans. We have implemented the PL-ICP algorithm which is a fast and robust way of scan matching. Our experiments showed that the performance of PL-ICP on real data is highly dependent on a good initialization as well as perfect outlier detection. To address the first problem we Introduced a re-initialization mechanism when it is stuck in a bad local minimum. Moreover, To overcome the second problem we have employed a fusion module for outlier detection which works based on both Euclidean distance and reflectance value. Eventually, the combination of all the aforementioned methods and mechanisms lead to an adequately accurate solution.

REFERENCES

[1] M. Betke and L. Gurvits, "Mobile robot localization using landmarks," *IEEE transactions on robotics and automation*, vol. 13, no. 2, pp. 251–263, 1997.

[2] M. Drumheller, "Mobile robot localization using sonar," *IEEE transactions on pattern analysis and machine intelligence*, no. 2, pp. 325–332, 1987.

[3] F. Chenavier and J. Crowley, "Position estimation for a mobile robot using vision and odometry," in *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pp. 2588–2589, 1992.

[4] S. Cooper and H. Durrant-Whyte, "A kalman filter model for gps navigation of land vehicles," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94)*, vol. 1, pp. 157–163, IEEE, 1994.

[5] L. Kleeman *et al.*, "Optimal estimation of position and heading for mobile robots using ultrasonic beacons and dead-reckoning.," in *ICRA*, pp. 2582–2587, 1992.

[6] R. Chatila and J.-P. Laumond, "Position referencing and consistent world modeling for mobile robots," in *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 138–145, IEEE, 1985.

[7] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *The international journal of Robotics Research*, vol. 5, no. 4, pp. 56–68, 1986.

[8] A. Kosaka and A. C. Kak, "Fast vision-guided mobile robot navigation using model-based reasoning and prediction of uncertainties," *CVGIP: Image understanding*, vol. 56, no. 3, pp. 271–329, 1992.

[9] J. L. Crowley, "World modeling and position estimation for a mobile robot using ultrasonic ranging.," in *ICRA*, vol. 89, pp. 674–680, 1989.

[10] J. J. Leonard and H. F. Durrant-Whyte, "Simultaneous map building and localization for an autonomous mobile robot.," in *IROS*, vol. 3, pp. 1442–1447, 1991.

[11] Y. Watanabe and S. Yuta, "Position estimation of mobile robots with internal and external sensors using uncertainty evolution technique," in *Proceedings., IEEE International Conference on Robotics and Automation*, pp. 2011–2016, IEEE, 1990.

[12] P. Burlina, D. DeMenthon, and L. S. Davis, *Navigation with Uncertainty: Reaching a Goal in a High Collision Risk Region. I.* Citeseer, 1991.

[13] L. Matthies and S. Shafer, "Error modeling in stereo navigation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 3, pp. 239–248, 1987.

[14] A. D. L. Escalera, L. Moreno, M. A. Salichs, and J. M. Armingol, "Continuous mobile robot localization by using structured light and a geometric map," *International Journal of Systems Science*, vol. 27, no. 8, pp. 771–782, 1996.

[15] J. J. Leonard and H. F. Durrant-Whyte, "Mobile robot localization by tracking geometric beacons," *IEEE Transactions on robotics and Automation*, vol. 7, no. 3, pp. 376–382, 1991.

[16] J. J. Leonard and H. F. Durrant-Whyte, *Directed sonar sensing for mobile robot navigation*, vol. 175. Springer Science & Business Media, 2012.

[17] A. Curran and K. J. Kyriakopoulos, "Sensor-based self-localization for wheeled mobile robots," *Journal of Robotic Systems*, vol. 12, no. 3, pp. 163–176, 1995.

[18] A. Holenstein and E. Badreddin, "Mobile-robot position update using ultrasonic range measurements," *INTERNATIONAL JOURNAL OF ROBOTICS & AUTOMATION*, vol. 9, no. 2, pp. 72–80, 1994.

[19] S. Borthwick, M. Stevens, and H. F. Durrant-Whyte, "Position estimation and tracking using optical range data," in *Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'93)*, vol. 3, pp. 2172–2177, IEEE, 1993.

[20] M. Meng and A. Kak, "Fast vision-guided mobile robot navigation using neural networks," in *[Proceedings] 1992 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 111–116, IEEE, 1992.

[21] A. Zou, Z. Hou, L. Zhang, and M. Tan, "A neural network-based camera calibration method for mobile robot localization problems," in *International symposium on neural networks*, pp. 277–284, Springer, 2005.

[22] H. M. Barberá, A. G. Skarmeta, M. Z. Izquierdo, and J. B. Blaya, "Neural networks for sonar and infrared sensors fusion," in *Proceedings of the Third International Conference on Information Fusion*, vol. 2, pp. WEB4–18, IEEE, 2000.

[23] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611, pp. 586–606, International Society for Optics and Photonics, 1992.

[24] S. Thrun, M. Diel, and D. Hähnel, "Scan alignment and 3-d surface modeling with a helicopter platform," in *Field and Service Robotics*, pp. 287–297, Springer, 2003.

[25] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2d range scans," *Journal of Intelligent and Robotic systems*, vol. 18, no. 3, pp. 249–275, 1997.

[26] S. T. Pfister, K. L. Kriechbaum, S. I. Roumeliotis, and J. W. Burdick, "Weighted range sensor matching algorithms for mobile robot displacement estimation," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 2, pp. 1667–1674, IEEE, 2002.

[27] B. Jensen and R. Siegwart, "Scan alignment with probabilistic distance metric," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3, pp. 2191–2196, IEEE, 2004.

[28] L. Montesano, J. Minguez, and L. Montano, "Probabilistic scan matching for motion estimation in unstructured environments," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3499–3504, IEEE, 2005.

[29] J. Minguez, F. Lamiraux, and L. Montesano, "Metric-based scan matching algorithms for mobile robot displacement estimation," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 3557–3563, IEEE, 2005.

[30] Y. Chen and G. G. Medioni, "Object modeling by registration of

multiple range images.," *Image Vision Comput.*, vol. 10, no. 3, pp. 145–155, 1992.

[31] M. C. Bosse, *ATLAS: a framework for large scale automated mapping and localization*. PhD thesis, Massachusetts Institute of Technology, 2004.

[32] A. Censi, "An ICP variant using a point-to-line metric," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, (Pasadena, CA), May 2008.

[33] E. B. Olson and S. Teller, "Robust and efficient robotic mapping," 2008.

[34] R. S. Consensus, "A paradigm for model fitting with applications to image analysis and automated cartography," *MA Fischler, RC Bolles*, vol. 6, pp. 381–395, 1981.

[35] A. Diosi and L. Kleeman, "Fast laser scan matching using polar coordinates," *The International Journal of Robotics Research*, vol. 26, no. 10, pp. 1125–1153, 2007.

[36] P. Biber and W. Straßer, "The normal distributions transform: A new approach to laser scan matching," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, vol. 3, pp. 2743–2748, IEEE, 2003.

[37] A. Censi, L. Iocchi, and G. Grisetti, "Scan matching in the hough domain," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 2739–2744, IEEE, 2005.

[38] T. Rofer, "Using histogram correlation to create consistent laser scan maps," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, pp. 625–630, IEEE, 2002.

[39] M. Bosse and J. Roberts, "Histogram matching and global initialization for laser-only slam in large unstructured environments," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 4820–4826, IEEE, 2007.

[40] G. Weiß and E. v. Puttkamer, "A map based on laserscans without geometric interpretation," 1999.

[41] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The oxford robotcar dataset," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017.