Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 54 (2021) 154–159

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

10th CIRP Sponsored Conference on Digital Enterprise Technologies (DET 2021) – Digital Technologies as Enablers of Industrial Competitiveness and Sustainability

A simultaneous localization and mapping algorithm for sensors with low sampling rate and its application to autonomous mobile robots

Krisztián Balázs Kis[a], János Csempesz[a], Balázs Csanád Csáji[a,*]

[a]*SZTAKI: Institute for Computer Science and Control, Eötvös Loránd Research Network, Kende u. 13-17., Budapest H-1111, Hungary*

* Corresponding author. Tel.: +36-1-279-6231. *E-mail address:* balazs.csaji@sztaki.hu

Abstract

In this paper we suggest a Simultaneous Localization and Mapping (SLAM) algorithm for Autonomous Mobile Robots (AMRs) which have LiDAR (light detection and ranging) type planar sensors with low sampling rate, e.g., less than 1 Hz. The proposed method uses 2-dimensional point clouds for its internal occupancy map representation and applies Point Set Registration (PSR) algorithms for mapping and localization. The approach is validated on both synthetic and real-world data. The results demonstrate that the proposed method is efficient, even when the observations are imprecise as well as the difference between consecutive measurements is high in terms of position and orientation.

Keywords: SLAM; AMR; LiDAR; Point Set Registration; Point Cloud

1. Introduction

As the digital transformation of manufacturing gains momentum, building on the unprecedented progress of information and communication technologies, more and more emphasis is placed on the applications of advanced robotics, as one of the driving forces behind the fourth industrial revolution [7].

Several authors have argued that open networks of dynamic and reconfigurable cyber-physical systems of cooperative autonomous entities mean the future of manufacturing and logistics [3, 4]. These approaches have many advantages, such as increased reliability, robustness, performance, adaptiveness and flexibility, as well as reduced costs. On the other hand, such distributed approaches introduce several challenges which should be addressed. These include, for example, decentralized information, decision myopia, security and confidentiality, network stability, local autonomy, and communication overload [4].

Autonomous Mobile Robots (AMRs) constitute [8] an important part of the aforementioned (cooperative) autonomous cyber-physical systems paradigm and they play a crucial role in developing complex, adaptive, distributed logistic systems.

One of the fundamental problems for AMRs is to accurately sense their environment and effectively navigate inside of it, in order to reach a given goal. Simultaneous Localization and Mapping (SLAM) methods [2, 9] formulate a part of this problem as a continuous iteration of sensing the environment, building an internal representation of it and providing an accurate position and orientation of the robot inside of it. The first part is achieved through sensor measurements, which are transformed and merged together into an occupancy map. Simultaneously, any new measurement is matched against the map to derive the current location and orientation of the robot in the environment.

Although there are many solutions to the SLAM problem, ranging from (extended) Kálmán and particle filters and expectation maximization (EM) algorithms to various multi-robot solutions [2], their applicability can highly depend on the sensor type/accuracy and the environment itself. Furthermore there can also be other limiting factors like the balance between the sampling rate of the sensor and the speed of the agent. The motivation of this paper is to present a solution for cases, where the sampling rate of the measurement sensor is very low (less than one per second) and therefore the consecutive measurements can highly differ from each other with little overlap.

One of the direct motivations of our research came from the Industry 4.0 Robot Laboratory of SZTAKI (Institute for Computer Science and Control) located at the Széchenyi István University, Győr, Hungary. The laboratory has roughly 200 m^2

10.1016/j.promfg.2021.07.023

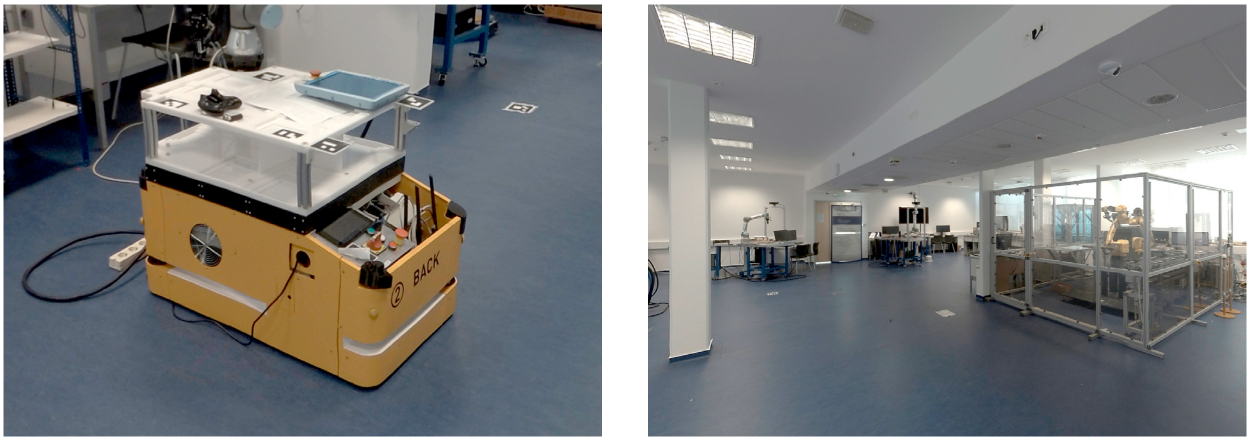
Fig. 1. (a) one of the AMRs with markers for the current visual localization; and (b) the experimental robot laboratory with ceiling-mounted cameras.

floor area with several robots, including AMRs equipped with 2D LiDAR (light detection and ranging) sensors. The main purpose of this laboratory is to support the research connected to human-machine interaction and provide an experimental setup for developments such as the one presented in this article.

The current localization of the AMRs is based on ceiling-mounted cameras which detect markers placed on the top of the AMRs, see Fig. 1. This approach has several drawbacks, such as: it can only be applied indoors, where the ceiling is accessible to install cameras, and the markers on the top of the AMRs limit what kind of equipment can be mounted on the vehicles. Moreover, relying on an external system for localization reduces the autonomy and the robustness of the logistic system.

The core idea is that a low-cost LiDAR based internal solution could provide an alternative to the expensive and restrictive camera-based external positioning system, especially, since such LiDAR sensors are already available on the AMRs for safety guarantees. That is, the AMRs are equipped with SICK sensors, whose primary function is proximity sensing: they ensure the vehicles against crashing into obstacles. These SICK planar LiDAR sensors can be queried, but only at a low sampling rate. The fine control of the robot movements could then be based on the combination of the low-frequency position estimates of a suitable SLAM algorithm and the high-frequency relative position estimates coming from a wheel odometry.

The LiDAR measurements of the AMRs are made at every direction within a 270° viewing angle with a step size of 0.5°. The obtained distance data have 1 cm accuracy and are obtained at a low, less than 1 Hz, sampling rate. Each AMR has two such LiDARs at the opposite sides of the vehicle, but even one sensor is enough to obtain efficient mapping and localization.

The main aim of the paper is to suggest a SLAM algorithm which can work with data where the consecutive measurements can be considerably different as a result of the potentially significantly changed position and orientation of the AMR.

The structure of the paper is as follows: in Section 2 we present the proposed SLAM algorithm, in Section 3 we discuss the tests and validations, and Section 4 concludes the paper.

2. The Proposed SLAM Algorithm

The proposed SLAM algorithm takes point cloud measurements (LiDAR) and also uses point clouds for the internal occupancy map representation. Therefore, this solution relies on multiple Point Set Registration (PSR) methods for calculating the optimal rigid transformation at any given point. Furthermore, different strategies were implemented to deal with the relative and absolute localization of the AMR and also to provide backup solutions, in case a strategy fails to localize the vehicle. As the algorithm incorporates multiple PSR methods, we need to introduce an error metric for measuring their performance in terms of accuracy. Formally, we apply

$$\begin{aligned} e_{psr}(X, Y, \epsilon) &\doteq \operatorname{mean}\left\{ d_{min}(x) : x \in X \wedge d_{min}(x) < \epsilon \right\}, \\ d_{min}(x) &\doteq \min_{y \in Y} \| x - y \|_2 \end{aligned} \tag{1}$$

Eq. (1) defines the e_{psr} registration error function, which takes $X, Y \subset \mathbb{R}^2$ finite 2-dimensional point sets, calculates the distance to the closest point in Y from each point of X and averages these distances only if they are below an $\epsilon > 0$ threshold. In the tests cases, presented in Section 3, we always set $\epsilon = 1$ (meter). This error metrics works especially well if X is a transformed subset of Y, which is practically consistent with X being the measurement and Y being the map, because those points in Y that are not represented in X are ignored in the error calculation.

Furthermore, we introduce the shift$_r$ function defined as

$$\begin{aligned} \text{shift}_r(\boldsymbol{h}) &\doteq S_r \cdot \boldsymbol{h} \\ S_r &\doteq \begin{cases} (s_{i,j} = 1), & \text{if } j = i + r \\ (s_{i,j} = 1), & \text{if } j = i + r - n \\ (s_{i,j} = 0), & \text{otherwise} \end{cases} \end{aligned} \tag{2}$$

where $\boldsymbol{h} \in \mathbb{R}^n$ and $S_r \in \mathbb{R}^{n \times n}$. This function shifts the components of a $\boldsymbol{h}$ vector by r positions in a circular manner.

The following sections may use the *pose* and *rigid transformation* expressions interchangeably as they represent the same thing in our context: a translation and rotation together.

2.1. Subcomponents

As it was mentioned, multiple PSR methods were utilized in the proposed algorithm in order to obtain accurate rigid transformations between any new sensor measurement and the internal occupancy map. The applied PSR methods are as follows:

- Iterative Closest Point (ICP) [1]
- Coherent Point Drift (CPD) [5]
- Relative Directional Neighbour Matching (RDNM)

ICP is a well-known and reliable PSR algorithm with many variants and applications. Rusinkiewicz and Levoy made a comprehensive summary on the different ICP variants, categorizing them based on what method they use in the six general stages of ICP [6]. This algorithm performs an iterative registration, where an iteration typically consists of (a) finding the closest point in the target point set for each point of the source point set and (b) solving the least-square problem minimizing the distance between the corresponding point pairs. Consequently, this algorithm works best if the two pint sets are sufficiently close to each other in terms of position and orientation.

CPD is a newer PSR method, which does not require one-to-one correspondence between the points for determining the rigid transformation between the point clouds, but uses a probabilistic approach instead. It fits Gaussian Mixture Model (GMM) centroids (representing the first point set) to the data (the second point set) by maximizing the likelihood. This algorithm inherently handles outliers and noise better compared to direct correspondence methods like ICP. However, as we found out in our experiments, the performance of CPD is not sufficient in cases, where the two point sets do not represent the same entity, e.g. one of them is a sufficiently small subset of the other.

We developed another PSR method, called RDNM, specifically to give a quick and rough estimate of the rigid transformation between the measurement and map point clouds. It has two major assumptions about the two input point clouds: a) it assumes that one of the point clouds is a subset of the other and b) it expects the point clouds to be LiDAR-like measurements (the singular measurement has a central vantage point and the map was built from such individual measurements); making it less robust compared to general registration methods like ICP and CPD. However, this specialization helps to improve the efficiency as most of the PSR methods handle subsets poorly.

The operation of Relative Directional Neighbour Matching is detailed in pseudocode 1. The *RDN* procedure takes $X \subset \mathbb{R}^2$ 2-dimensional finite point cloud, $c \in \mathbb{R}^2$ viewpoint, α section size and ϵ threshold, and returns $\boldsymbol{h} \in \mathbb{R}^{2\pi/\alpha}$. First, X is divided into α sized sections X_α as seen from c, then for each X_α section the distance d_{min} between the closest point and c is determined, and finally the mean distance h_i is calculated form each points of X_α, where the distance to c is between d_{min} and $d_{min} + \epsilon$.

The *RDNM* procedure takes $X, Y \subset \mathbb{R}^2$ finite point clouds, $P \subset \mathbb{R}^2$ candidate positions and the α and ϵ parameters which are directly forwarded to the *RDN* procedure calls, and returns T_{RDNM} rigid transformation, that transforms X onto Y with minimal $e_{PSR}(T_{RDNM}(X), Y, 1)$ registration error. First the *RDN* metric is calculated for X viewed from the 0 vector, then each candidate point in P is evaluated, which consists of calculating the *RDN* metric viewed from the candidate position and finding the r_p rotation value where $||\boldsymbol{h}_X - \text{shift}_{r_p}(\boldsymbol{h}_{Y,p})||$ is minimal (the shift$_r$ function corresponds to a rotation, as each component of $\boldsymbol{h}_X$ and $\boldsymbol{h}_{Y,p}$ is calculated from the points in a specific direction).

Algorithm 1 Relative Directional Neighbour Matching

```
1: procedure RDN($X, c, \alpha, \epsilon$)
2:     for all $i \in \{0, \alpha, 2\alpha..2\pi\}$ do
3:         $X_\alpha \leftarrow \{x \in X : \arctan x_y - c_y / x_x - c_x \in [i, i+\alpha)\}$
4:         $d_{min} \leftarrow \min_{x \in X_\alpha} ||x - c||$
5:         $D_{\alpha,\epsilon} \leftarrow \{||x-c|| : x \in X_\alpha \wedge ||x-c|| \in [d_{min}, d_{min}+\epsilon)\}$
6:         $h_i \leftarrow \text{mean}(D_{\alpha,\epsilon})$
7:     end for
8:     return $\boldsymbol{h}$
9: end procedure
10:
11: procedure RDNM($X, Y, P, \alpha, \epsilon$)
12:     $T_{RDNM} \leftarrow I$
13:     $\boldsymbol{h}_X \leftarrow \text{RDN}(X, 0, \alpha, \epsilon)$
14:     for all $p \in P$ do
15:         $\boldsymbol{h}_{Y,p} \leftarrow \text{RDN}(Y, p, \alpha, \epsilon)$
16:         $r_p \leftarrow \text{argmin}_{r \in \{0,\alpha,2\alpha..2\pi\}} ||\boldsymbol{h}_X - \text{shift}_r(\boldsymbol{h}_{Y,p})||$
17:     end for
18:     $p_{RDNM} \leftarrow \text{argmin}_{p \in P} ||\boldsymbol{h}_X - \text{shift}_{r_p}(\boldsymbol{h}_{Y,p})||$
19:     $T_{RDNM}(x) \leftarrow R_{r_p}(x) + p_{RDNM}$
20:     return $T_{RDNM}$
21: end procedure
```

Algorithm 2 SLAM absolute localization strategy

```
1: procedure SLAMAbsolute($X, Y, e_{max}$)
2:     $T_{abs} \leftarrow I$
3:     $P_{abs} \leftarrow$ generate global grid points in map
4:     $T_{RDNM} \leftarrow \text{RDNM}(X, Y, P_{abs})$
5:     $T_{ICP} \leftarrow \text{ICP}(X, Y, T_{RNDM})$
6:     if $e_{psr}(T_{ICP}(X), Y) < e_{max}$ then
7:         $T_{abs} \leftarrow T_{ICP}$
8:     end if
9:     return $T_{abs}$
10: end procedure
```

2.2. Absolute Localization

There are situations when the vehicle needs to position itself inside its internal map without the knowledge of its previous position. This could be due to emergency shutdowns or other reasons and it is known in the literature as the "*kidnapped robot*" problem. The proposed algorithm would deal with this problem by using a global version of the RDNM, where the candidate positions are generated along gridpoints of the internal map. This approach could be further improved in various ways, such as limiting the number of gridpoints based on an estimated position and obstacles, or iteratively refining the grid.

Algorithm 3 SLAM relative localization strategy

```
 1: procedure SLAMRelative($X, Y, e_{max}$)
 2:     $T_{rel} \leftarrow I$
 3:     $P_{rel} \leftarrow$ generate local grid points in map
 4:     $T_{RDNM} \leftarrow$ ANDM($X, Y, P_{rel}$)
 5:     $T_{ICP} \leftarrow$ ICP($X, Y, T_{ANDM}$)
 6:     if $e_{psr}(T_{ICP}(X), Y) < e_{max}$ then
 7:         $T_{rel} \leftarrow T_{ICP}$
 8:     else
 9:         $T_{CPD} \leftarrow$ CPD($X, Y$)
10:         $T_{ICP} \leftarrow$ ICP($X, Y, T_{CPD}$)
11:         if $e_{psr}(T_{ICP}(X), Y) < e_{max}$ then
12:             $T_{rel} \leftarrow T_{ICP}$
13:         else
14:             $T_{abs} \leftarrow$ SLAMabsolute($X, Y, e_{max}$)
15:             if $e_{psr}(T_{abs}(X), Y) < e_{max}$ then
16:                 $T_{rel} \leftarrow T_{abs}$
17:             end if
18:         end if
19:     end if
20:     return $T_{rel}$
21: end procedure
```

Pseudocode 2 presents the absolute localization method. The $SLAMabsolute$ procedure takes $X, Y \subset \mathbb{R}^2$ 2-dimensional finite point clouds and e_{max} error threshold, and returns T_{abs} rigid transformation. It first generates a set of global candidate points P_{abs} using a fix sized grid covering the Y point cloud which is the internal map of the AMR (some additional filtering can be done to remove points where the AMR would not fit), then uses the $RDNM$ algorithm to find the rough pose of the AMR, and finally it uses the ICP algorithm to provide an exact transformation and if $e_{psr}(T_{ICP}(X), Y)$ is less then the predefined e_{max}, the algorithm successfully terminates. Here, the ICP procedure (and the CPD procedure in the next section) takes a third argument, which is an initial guess for the transformation.

2.3. *Relative Localization*

The main aim of any SLAM algorithm is to continuously update the position of the vehicle and its internal map, which is achieved through incremental sensor measurements and their integration into the map representation. The proposed algorithm utilizes multiple strategies starting with the quickest one and moving toward slower, more robust methods after each registration failure. Naturally the last resort should always be the absolute localization approach presented in the previous section.

The relative localization scenario is presented in pseudocode 3. The $SLAMrelative$ procedure takes $X, Y \subset \mathbb{R}^2$ 2-dimensional finite point clouds and e_{max} error threshold, and returns T_{rel} rigid transformation. This procedure incorporates three registration strategies, each of them activated only if the previous one has failed (the registration error is more than the predefined e_{max} error threshold). The first (fast) strategy is the primary method, which uses the $RDNM$ algorithm with locally generated candidate points P_{rel} (around the assumed position of the AMR, e.g. 6 points in a 0.3 meter radius in our current implementation), then uses the ICP algorithm to provide an exact transformation. The second strategy uses CPD for a rough registration and finishes with ICP for the exact transformation (similarly to the first strategy). Finally if the first two strategies have both failed, then the $SLAMabsolute$ procedure is called (detailed in the previous section) as a last resort solution. This operation order assures that the registration will be fast in most of the cases, but it will still produce a result in the problematic cases (at the cost of computational time).

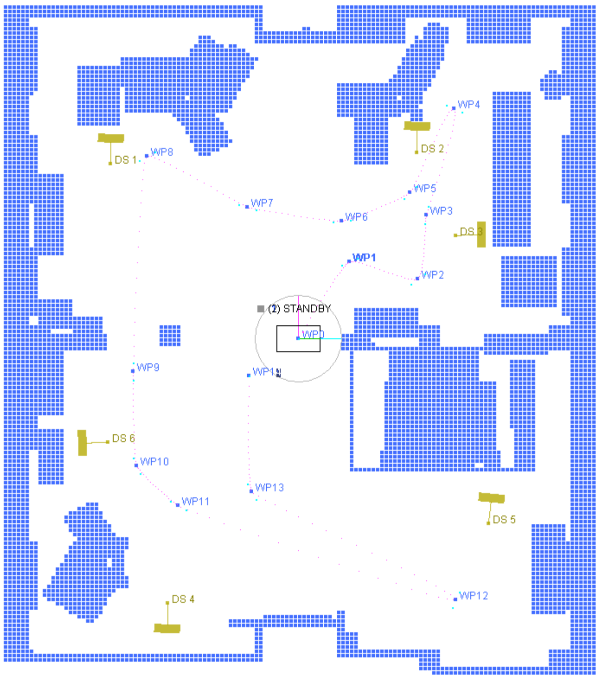

Fig. 2. virtual map of the AMR control software showing the laboratory layout.

3. Validation

In this section we present two validation cases which test the relative and absolute localization capabilities of the proposed SLAM algorithm. Both cases simulate the laboratory environment where the AMR will have to navigate in real life. This area has a fixed layout, where the bigger obstacles (e.g., machinery, tables) are static and small obstacles (e.g., chairs, people) could randomly appear or change position. In the relative localization cases the area was unknown, while in the absolute localization cases the area was known (it used the map built during relative localization). In both cases the AMR moved relatively slowly, around 0.3 m/s, due to some limitations of the measurement retrieving process, and the sampling rate was about 0.5 Hz.

3.1. *Synthetic Data*

During synthetic validation, we utilized an AMR control software, which models accurate differential drive vehicle dy-

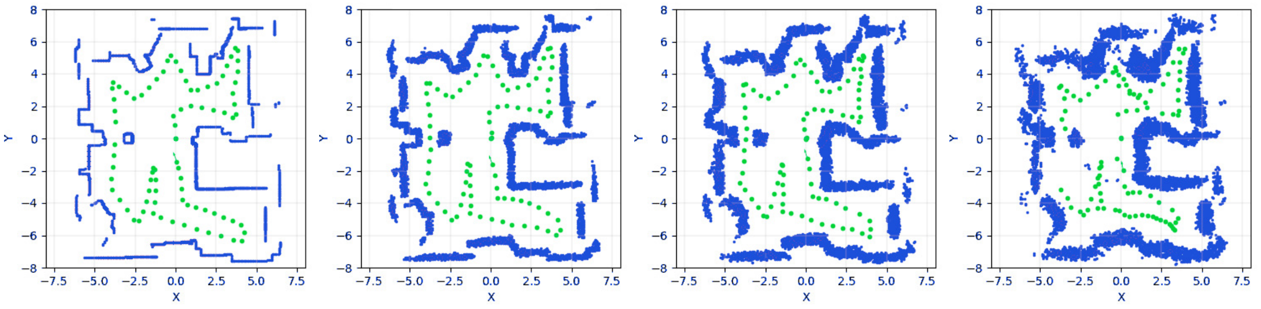

Fig. 3. internal occupancy maps for synthetic data with different Gaussian noise levels: from left to right σ was set to 0.0, 0.1, 0.2 and 0.3, respectively.

namics. It can be used to navigate the vehicle inside the laboratory. The control software can fully simulate the vehicle and take sensor measurements in the virtual environment, as well, so we connected it to the implemented solution and simulated the SLAM procedure by taking measurements in every few seconds, updating the map and calculating the exact position.

The basic layout of the laboratory can be seen in Fig 2, which is the main display of the control software. This map is ideal and noise-free, hence, we could simulate different noise levels of the sensor measurements in our tests. The virtual AMR was given a predefined path for mapping the environment, and made about 85 measurements from start to finish (note that with higher noise levels the path took a little bit longer to complete resulting in a few more additional measurements).

Fig 3 illustrates the internal map built by the proposed SLAM approach with the 4 different noise levels. The blue dots represent the contour of the obstacles and the green dots represent the path of the AMR. On each level, a Gaussian noise was added to both x and y coordinates of the virtual measurements simulating the inaccuracy of the sensor. The σ (deviation) parameter of the Gauss distribution was set to 0, 0.1, 0.2 and 0.3 for the 4 noise levels, respectively. As we can see, the higher the noise level is, the more unreliable the map becomes and the AMR has less confidence about its location. At the highest level, the path of the vehicle becomes jagged (as if it was “drunk”). Still, the algorithm managed to complete the map each time, even when the vehicle lost its position multiple times in a row (at the highest noise level we can see a gap at the center left side in the path of the AMR). We can also see that the shape of the map remains intact as the noise increases, which indicates that the algorithm is robust w.r.t. various noise levels.

The results of the experiments using synthetic data are summarized in Table 1. It presents the averages (*mean*) and the standard deviations (*std*) of the registration errors (as defined in Eq. 1) calculated for each noise level in the relative and absolute localization cases. Additionally, we defined a success rate for the absolute localization case, which shows how many times the algorithm managed to find the location of the AMR out of the 10 predefined setups, based on an expert diagnosis. What we did is checking each absolute localization setup manually (by visual inspection) and we only considered one a success if the measurement point cloud matched to the internal map perfectly. This was needed because the registration error does not always indicate sufficiently that a registration is correct, especially with higher noise levels. At the first two noise levels the absolute localization was perfect, while on the third level one, and on the forth level two (out of ten) localizations were unsuccessful.

3.2. Real Data

In the second validation scenario we created two distinct datasets from real LiDAR sensor measurements done by the AMR in the robot laboratory. Both datasets correspond to a general mapping scenario, where the AMR explores the laboratory along a predefined path, taking measurements typically every 2-3 seconds. The resulting series of measurement point sets, dataset A and B, consist of 83 and 69 samples, respectively. For each measurement, the assumed AMR position and orientation is stored, as well, which provides the starting point for the relative localization in each SLAM iteration. We simulated the vehicle movement and measurements based on this data and

Table 1. Relative and absolute localization registration errors on synthetic data.

	Relative Case		Absolute Case		
	mean (meter)	std (m)	mean (m)	std (m)	succes rate
σ=0.0	0.029	0.021	0.018	0.006	100.0%
σ=0.1	0.046	0.017	0.036	0.004	100.0%
σ=0.2	0.061	0.023	0.066	0.048	90.0%
σ=0.3	0.088	0.034	0.079	0.038	80.0%

Table 2. Relative and absolute localization registration errors on real data.

		Dataset A			Dataset B		
		mean (meter)	std (m)	succes rate	mean (m)	std (m)	succes rate
Relative Case	Initial	0.190	0.125	-	0.208	0.146	-
	Final	0.042	0.021	-	0.044	0.020	-
Absolute Case	g=0.2	0.037	0.019	98.6%	0.051	0.057	94.0%
	g=0.5	0.039	0.025	97.1%	0.053	0.059	90.4%
	g=0.8	0.044	0.034	92.8%	0.065	0.075	84.3%

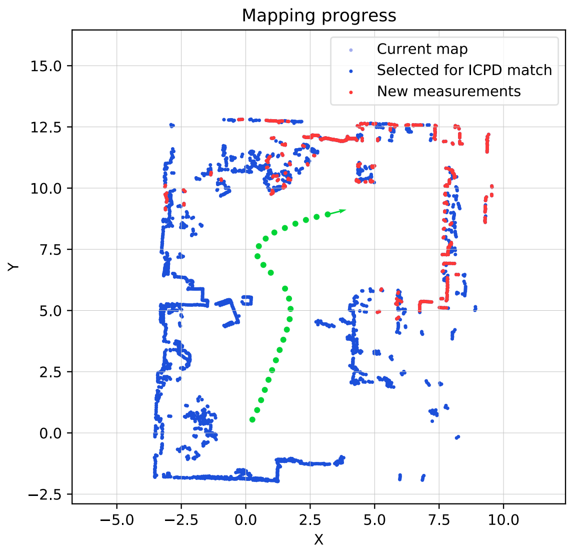

Fig. 4. internal occupancy map of the SLAM algorithm using real data.

created two test scenarios: building the internal map (relative localization) from one dataset and registering the measurements of the other dataset in the built map (absolute localization).

A partial internal map is shown in Fig. 4 taken during the operation of the SLAM algorithm. The blue dots represent the map, the green dots represent the AMR path and the red dots represent the actual measurements being registered. We can see that the map has some artefacts and additional noise compared to the synthetic maps seen in Fig. 3. This is expected in a real life condition with different materials influencing the sensor, various cables and other obstacles being present and even moving elements like people walking in the laboratory.

The results of the real validation case are summarized in Table 2. It presents the averages (*mean*) and the standard deviations (*std*) of the registration errors and the *succes rate*, just like in the synthetic case earlier. Additionally, we listed the initial (based on the assumed vehicle position) and final error averages (after the registration) for the relative localization, showing around 80% decrease after registration. Moreover, for the absolute localization case, we configured the RDNM algorithm with three grid resolution settings, g was set to 0.2, 0.5 and 0.8 meters. This means that the P input point set of the RDNM algorithm was constructed from the vertices of a g sized grid covering the map. We can see that the smaller the grid resolution is, the better the success rate of the localization is, however this also requires more computational resources. We found that 0.5 meter is a sweet spot for accuracy and time requirements.

4. Conclusions

In this paper we proposed a new Simultaneous Localization and Mapping (SLAM) approach for scenarios where the difference between consecutive measurements is high in terms of position and orientation. Such scenario arises, e.g., if the Autonomous Mobile Robot (AMR) is equipped with a LiDAR (light detection and ranging) type planar sensor having a low sampling rate. We introduced an effective Point Set Registration (PSR) method, called Relative Directional Neighbour Matching (RDNM), designed for this problem, more precisely, to perform an initial rough registration at the start of each SLAM iteration.

We have validated the proposed SLAM method through two test cases: one using synthetic data and another one based on real LiDAR sensor measurements. The results showed that the proposed method is effective in the relative localization (mapping) scenario, even when the consecutive measurements are noisy and/or are relatively far away from each other. The suggested algorithm performed well in the absolute localization case, as well, even though there were some cases where it was not able to find the correct absolute location of the vehicle.

Based on our results, we can define a clear path for moving forward in our research. We would like to extend our validation with a real world scenario in which the AMR directly uses the algorithm for its navigation. Furthermore, we would like to add extensive simulated testing scenarios, where the method is evaluated in (randomized) algorithmically generated maps having various sizes and using sensors with different noise levels. Finally, we would like to make detailed comparisons with other SLAM solutions in our experimental robot laboratory setting.

Acknowledgements

This research was supported by the Ministry of Innovation and Technology, Hungary, NRDI Office, within the framework of the Autonomous Systems National Laboratory Program.

References

[1] Besl, P.J., McKay, N.D., 1992. A method for registration of 3-d shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 14, 239–256. doi:10.1109/34.121791.
[2] Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., Leonard, J.J., 2016. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. IEEE Transactions on Robotics 32, 1309–1332. doi:10.1109/TRO.2016.2624754.
[3] Monostori, L., Csáji, B.Cs., Kádár, B., Pfeiffer, A., Ilie-Zudor, E., Kemény, Zs., Szathmári, M., 2010. Towards adaptive and digital manufacturing. Annual Reviews in Control 34, 118–128. doi:10.1016/j.arcontrol.2010.02.007.
[4] Monostori, L., Valckenaers, P., Dolgui, A., Panetto, H., Brdys, M., Csáji, B.Cs., 2015. Cooperative control in production and logistics. Annual Reviews in Control 39, 12–29. doi:10.1016/j.arcontrol.2015.03.001.
[5] Myronenko, A., Song, X., 2010. Point set registration: Coherent point drift. IEEE Transactions on Pattern Analysis and Machine Intelligence 32, 2262–2275. doi:10.1109/TPAMI.2010.46.
[6] Rusinkiewicz, S., Levoy, M., 2001. Efficient variants of the ICP algorithm, in: Proceedings Third International Conference on 3-D Digital Imaging and Modeling, pp. 145–152. doi:10.1109/IM.2001.924423.
[7] Schwab, K., 2017. The Fourth Industrial Revolution. World Economic Forum, Crown Publishing Group.
[8] Siegwart, R., Nourbakhsh, I.R., Scaramuzza, D., 2011. Introduction to Autonomous Mobile Robots. MIT press.
[9] Thrun, S., 2008. Simultaneous Localization and Mapping. Springer, Berlin, Heidelberg. pp. 13–41. doi:10.1007/978-3-540-75388-9_3.