Engineering Applications of Artificial Intelligence 67 (2018) 436–442

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Registration of lines in 2D LIDAR scans via functions of angles

Branislav Holý

Department of Computer Science, FEECS, VŠB - Technical University of Ostrava, 17. listopadu 15, 708 33, Ostrava — Poruba, Czech Republic

ARTICLE INFO

Keywords:
Registration
Scan-matching
Lines
LIDAR
Transformation

ABSTRACT

A novel algorithm for registration of 2D LIDAR scans that combines two already known but different concepts has been proposed. The algorithm combines the registration of lines and the registration in polar coordinates. The main idea is a new representation of the surrounding area that makes the mentioned combination possible. The algorithm works with line segments in polar coordinates that are expressed as functions of angles. This representation significantly decreases the computation time and enhances the resulting alignment. The presented algorithm was designed for indoor environments that are rich in objects that can be represented by line segments. The algorithm has been tested with many indoor LIDAR scans and is able to align them with high accuracy and in real-time.

1. Introduction

Over the last years, the popularity of LIDAR (Light Detection and Ranging) data is still increasing and the LIDAR technology is used in a wide range of applications. Very often, LIDAR data provide information about the surrounding area for the tasks like object tracking, robot navigation, autonomous car driving, or even mapping of an unknown environment. The robots and cars are equipped with laser scanners to scan the surrounding area in two or three dimensions. The lasers emit the rays and their reflections from the closest obstacles provide the information about the surrounding area.

One of the applications is simultaneous localization and mapping (SLAM). In the SLAM technique, the final map is composed of a set of the LIDAR scans that need to be aligned. In other words, the locations of the robot, where the scans have been taken, need to be corrected. Odometry data can provide an initial guess of the robot location, but every move of the robot increases the uncertainty about the location. Registration of the scans can decrease the uncertainty of the possible location of robot for further usage in the SLAM algorithms.

The main goal of LIDAR scan registration is to find the best orthonormal transformation between two LIDAR scans. The transformation has to align the scans with the best fitness. However, the LIDAR scans are only partially overlapped because they are scanned from different locations and also the correspondences between the points are unknown. These two properties make the registration problem difficult. The state-of-the-art algorithms solve this problem in different ways. Mostly, the correspondence problem is solved by searching for the closest point in the second set. The presented algorithm eliminates this problem by making use a continuous function that is defined in polar coordinates. In addition to this, the usage of the line segments (instead of points) decreases the computation time because the parts of the continuous function can be defined and computed analytically. The line segment representation of the 2D LIDAR scans is perfectly suitable for the indoor environments because those environments are mostly composed of objects that have straight shapes (like walls, wardrobes, commodes, and furnitures in general) and therefore can be detected as line segments. The benefits of the proposed method are the following.

- No correspondence search due to the polar coordinates.
- Usage of less objects (few line segments instead of many points).
- Fast computation of the error function and gradient due to the analytical representation.

The most commonly used algorithm for registration of two sets of points is Iterative Closest Points (ICP) in Besl and McKay (1992) and Zhang (1992). The algorithm works iteratively and tries to minimize the error between two sets of points. There are many variants of this algorithm that improve some steps of the original version. For example, Turk and Levoy (1994) introduced uniform sampling in the selection step, and Masuda et al. (1996) used random sampling with a different set of sample points at each iteration. The matching step was accelerated by Zhang (1994) who used better structures like k-D tree to find the closest points in the input sets. Other variants of these steps are described in Rusinkiewicz and Levoy (2001).

The Trimmed Iterative Closest Point Algorithm (TrICP) by Chetverikov et al. (2002) is a modification of the original ICP that

E-mail address: branislav.holy@vsb.cz.

https://doi.org/10.1016/j.engappai.2017.09.017
Received 15 December 2016; Received in revised form 24 July 2017; Accepted 18 September 2017
Available online 31 October 2017
0952-1976/

sorts the weights in the ascending order before the rejecting step and takes only a few of them. By this modification, the TrICP is more robust to a low overlap. Fitzgibbon (2003) introduced another variant called LM-ICP. It uses the Levenberg–Marquardt algorithm for minimization, which does not affect speed significantly but allows the use of statistics for better robustness. Generalized-ICP by Segal et al. (2010) adds a probabilistic framework to the original ICP algorithm with point-to-point and point-to-plane variants. Every point in the sets is represented by a normal distribution of probability with the mean at the point coordinates and the covariance matrix describing the error of measurement. The method for registration of RGB-D data is described by Ekekrantz et al. (2013), which uses features (points of interest) for computing the proper transformation. One of the newest modifications is Sparse-ICP (see Bouaziz et al., 2013). As mentioned in Pomerleau et al. (2013), over 400 papers have been published about ICP and its variants.

All of these ICP variants need to find the correspondences between the points by searching for the closest points in the second set. Even the mentioned k-D tree has still the asymptotic complexity of $\mathcal{O}(n\log(n))$ to find the correspondences for all points.

The Normal Distributions Transform (NDT) by Biber and Strasser (2003) is a different kind of registration method for 2D space. This registration method divides the space covered by points from the scanner into cells, and for every occupied cell, a normal distribution of probability is computed. Then it finds minimum of the $-\,\mathrm{score}$ function with Newton's algorithm. A 3D variant is described in Magnusson et al. (2007).

The NDT algorithm finds the corresponding normal distribution of probability by selecting its cell according to the coordinates of the examined point. This searching is fast ($\mathcal{O}(n)$ for all points), but some points are excluded from the error computation because no corresponding cell is found. Increasing the cell size can help with the inclusion of more points but also the covariance of the cell becomes higher.

The method of Cox (1991), or Iterative Closest Line (ICL) by Alshawa (2007) are useful for the indoor environments. Both algorithms as well as the presented method use linear features but in a different way. The method of Cox uses lines only for a model (that can be created from the first scan) and aligns the points from the second set to the model. Our new method uses the line segments for both scans, which improves the computation time. Another improvement of our method is the searching for correspondences. The condition for corresponding features is the distance between them in both previous methods. Moreover, the corresponding lines in ICL have to have similar angles within some threshold. Searching for the corresponding lines needs mn iterations in the method of Cox and n^2 iterations in ICL. In contrast, our new algorithm needs only n iterations because it uses polar coordinates.

Polar coordinates were used in some previous papers (for example Lingemann et al. 2005, or Polar Scan Matching (PSM) by Diosi and Kleeman 2005) and in the presented algorithm as well. The usage has the same purpose, which is fast correspondence search ($\mathcal{O}(n)$) preserving the inclusion of all available (visible) points. However, the description of the surrounding space differs from the presented one. Both earlier papers use the sets of points in polar coordinates, but the presented algorithm uses a function as a more general and continuous description of the surrounding area. This generalization allows the use of line segments as functions of angles as described below. Lingemann et al. (2005) also used the line segments but only for better detection of feature points, not for registration.

In this section, the introduction to the registration problematic, disadvantages of the state of the art methods, and solving the disadvantages have been explained. The remainder of the paper describes the proposed algorithm in every detail in the following Section 2, covering the mathematical background and implementation details of the proposed method. Section 3 shows some results of the experiments that have been used to prove benefits of the new algorithm, specifically its speed and accuracy. The conclusion and future plans are presented in the final section.

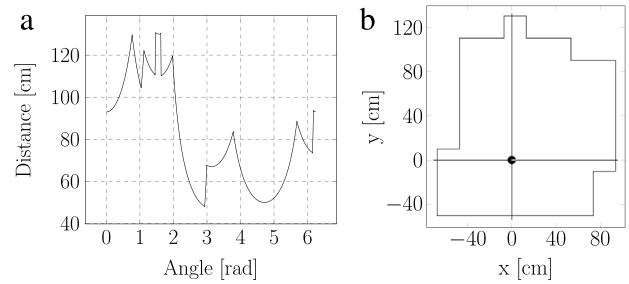

Fig. 1. The LIDAR function (a) of area (b).

2. New algorithm

The mentioned algorithms use the sets of points for the description of surrounding area. However, a set of points is too general and can contain more points that seem to be scanned in one angle. Since the LIDAR technology is able to scan only one distance in every angle, it is not necessary to use the sets of points. We can take advantage of this special property of the LIDAR scans and represent the surrounding area with a function called the LIDAR function defined as

$$\text{surrounding area} = \int_0^{2\pi} \mathrm{F}(\varphi)\,\mathrm{d}\,\varphi, \tag{1}$$

where the LIDAR function $\mathrm{F}(\varphi)$ returns a distance to the closest obstacle in angle φ (Fig. 1(a)). Sampling of the LIDAR function results in a set of points in the polar coordinate system, which is used in PSM as mentioned above.

The error of alignment between two LIDAR scans (F and F') can be expressed with an error function E as

$$\mathrm{E}(\mathcal{T}) = \int_0^{2\pi} (\mathrm{F}(\varphi) - \hat{\mathrm{F}}'(\varphi))^2\,\mathrm{d}\,\varphi, \tag{2}$$

where $\hat{\mathrm{F}}'$ is the transformed LIDAR function F' according to the orthonormal transformation denoted by $\mathcal{T}$ defined as follows

$$M = \left\{\left(\hat{\varphi}, \widetilde{\mathrm{F}}(\hat{\varphi})\right)\right\} = \left\{\mathcal{T}\left\{(\varphi, \mathrm{F}(\varphi))\right\}\right\}, \tag{3}$$

$$N(\gamma) = \left\{\widetilde{\mathrm{F}}(\gamma) \,|\, \left(\gamma, \widetilde{\mathrm{F}}(\gamma)\right) \in M\right\}, \tag{4}$$

$$\mathcal{T} = \begin{bmatrix} [r]\cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{5}$$

which saves every transformed point (rotated by θ and translated by t_x and t_y) of the LIDAR function into the set M. Since some transformed points would share the angle $\hat{\varphi}$, we need to choose only one of them that has the smallest value $\widetilde{\mathrm{F}}(\hat{\varphi})$. For this purpose, we introduce the function $N(\gamma)$ that returns the set of all distances $\widetilde{\mathrm{F}}(\gamma)$ that correspond to the chosen value of γ. The final transformed function $\hat{\mathrm{F}}$ is shown in (6) where we choose the $\min$ from the set returned by the function N

$$P = \left\{\left(\hat{\varphi}, \hat{F}(\hat{\varphi})\right) \,|\, \left(\hat{\varphi}, \hat{F}(\hat{\varphi})\right) = (\hat{\varphi}, \min N(\hat{\varphi}))\,;\, \left(\hat{\varphi}, \hat{F}(\hat{\varphi})\right) \in M\right\}. \tag{6}$$

The following conditions must be met (for small ε) to keep the order of transformed points. It also removes invisible points that seem to be scanned from the opposite side

$$\left(\hat{\varphi} + \hat{\varepsilon}, \hat{\mathrm{F}}(\hat{\varphi} + \hat{\varepsilon})\right) = \mathcal{T}\left\{(\varphi + \varepsilon, \mathrm{F}(\varphi + \varepsilon))\right\}, \tag{7}$$

$$\varphi < \varphi + \varepsilon \iff \hat{\varphi} < \hat{\varphi} + \hat{\varepsilon}, \tag{8}$$

where ε is a small number ($\varepsilon \to 0$) that describes the nearest surrounding around φ. The detailed explanation is covered in Section 2.3.

Unlike other environments, the indoor environments (consisting of doors, furniture, etc.) are ideal for the description of area via line segments. The presented method uses the line segments in the polar

coordinate system for the fast correspondence search. The line segments also change their end points dynamically according to the visible parts of the underlying line. The final error is then derived from the errors between the corresponding line segments. Its minimization finds the parameters of the orthonormal transformation with the best fitness value. Based on this, the algorithm can be divided into the following steps: line segment detection, transformation of the LIDAR function, line segment visibility, and minimization. All steps are repeated until some of the convergence criteria are met.

2.1. Line segment detection

The LIDAR lasers output the sets of angles and distances. These angle–distance pairs (points in a polar coordinate system) can be transformed into a Cartesian coordinate system. Thereafter, a line detection algorithm can be used to extract the lines from the scans. The previous version of this algorithm (Holý, 2016) uses the Split-and-Merge algorithm (Pavlidis and Horowitz, 1974) as the only line detection algorithm. The selection was made based on the comparison in Nguyen et al. (2007) for its superior speed, correctness, and suitability for LIDAR scans. However, we can improve the line detection even more by using a secondary line detection algorithm. After the points detected by the Split-and-Merge algorithm are removed from the sets, the remaining points are used by RANSAC (Fischler and Bolles, 1981) to detect more line segments. Since every line is constructed from a set of points, the first and last point (A and B) define the end points of the line segment.

A line in polar coordinates can be expressed as a function

$$\mathrm{L}(\varphi) = \frac{l}{\cos(\varphi - \alpha)}, \tag{9}$$

where l is the distance from the origin and α is the angle between the normal vector of the line and the x axis. The function L returns the distance from the origin in angle φ and is defined in the domain $D(\mathrm{L}) \in \left(\alpha - \frac{\pi}{2}; \alpha + \frac{\pi}{2}\right)$. Since we work with the line segments, the domain is bounded by the end points A and B. After that, the LIDAR function F can be constructed of the detected line segments as the following piecewise function.

$$\mathrm{F}(\varphi) = \begin{cases} \mathrm{L}_0(\varphi), & \text{if } \varphi \in D(\mathrm{L}_0) \\ \mathrm{L}_1(\varphi), & \text{if } \varphi \in D(\mathrm{L}_1) \\ \vdots & \\ \mathrm{L}_n(\varphi), & \text{if } \varphi \in D(\mathrm{L}_n) \end{cases} \tag{10}$$

$$D(\mathrm{F}) = D(\mathrm{L}_0) \cup D(\mathrm{L}_1) \cup \cdots \cup D(\mathrm{L}_n), \tag{11}$$

where the domains $D(\mathrm{L}_i)$ are disjoint.

2.2. Transformation

The function L is defined by two components α and l. Eq. (6) transforms the components into $\hat{\alpha}$ and $\hat{l}$ as

$$\hat{\alpha} = \alpha + \theta, \tag{12}$$

$$\hat{l} = l + t_x \cos(\alpha + \theta) + t_y \sin(\alpha + \theta), \tag{13}$$

where t_x, t_y, and θ defines the transformation $\mathcal{T}$ in (5). The domain of the function L needs to be transformed as well via its endpoints.

2.3. Line segment visibility

Some transformations can make the LIDAR function invalid because the domains of partial line functions become not disjoint (a line segment become only partially visible), or some line segments seem to be scanned from the opposite side (both problems are shown in Fig. 2).

In the case of not disjoint domains, an ordered set can be created from the bounds of the domains $D(\mathrm{L}_i)$. Then it is needed to check which line segment function has the smallest value in the middle of the examined interval of two consecutive bounds. Since every function L_i is defined only in the domain $D(\mathrm{L}_i)$, it is necessary to check the function L_i only between the intervals of its domain. After this step, all domains $D(\mathrm{L}_i)$ are disjoint again. This solves the problem described by (4) and (6) for the line segments.

In the second case, line segments that seem to be scanned from the opposite side need to be rejected. These line segments can be easily found if their transformed domain $D(\hat{\mathrm{L}}) \in \langle a; b\rangle$ meets the condition $a > b$ as described in (7) and (8).

2.4. Error of alignment

The partial error $\mathrm{e}_i(\mathcal{T})$ of two corresponding line segments L_i and $\hat{\mathrm{L}}_i'$ (gray areas in Fig. 3) can be computed within the $\langle \bar{a}_i; \bar{b}_i \rangle$ interval where both functions are defined, since $\bar{a}_i = \max(a_i, \hat{a}_i')$ and $\bar{b}_i = \min(b_i, \hat{b}_i')$ as

$$\mathrm{e}_i(\mathcal{T}) = \int_{\bar{a}_i}^{\bar{b}_i} \left(\mathrm{L}_i(\varphi) - \hat{\mathrm{L}}_i'(\varphi)\right)^2 \mathrm{d}\,\varphi. \tag{14}$$

This integral can be analytically computed as follows

$$\mathrm{e}_i(\mathcal{T}) = \left[l_i^2 \tan(\varphi - \alpha_i) + \hat{l}_i' 2 \tan(\varphi - \hat{\alpha}_i') - 2 l_i \hat{l}_i' P_i(\varphi)\right]_{\bar{a}_i}^{\bar{b}_i}, \tag{15}$$

where P_i is defined as

$$P_i(\varphi) = \begin{cases} \tan(\varphi - \alpha_i) & \text{if } \alpha_i = \hat{\alpha}_i' \quad \text{(a)} \\ \dfrac{\ln\left(\frac{\cos(\varphi - \alpha_i)}{\cos(\varphi - \hat{\alpha}_i')}\right)}{\sin(\alpha_i - \hat{\alpha}_i')} & \text{otherwise.} \quad \text{(b)} \end{cases} \tag{16}$$

The final error (2) is then defined as a sum of the partial errors multiplied by a coverage coefficient

$$\mathrm{E}(\mathcal{T}) = C \sum_{i=0}^{n} \mathrm{e}_i(\mathcal{T}) \tag{17}$$

where the coverage coefficient C is defined as

$$C = \frac{2\pi}{\left(\sum_{i=0}^{n} \bar{b}_i - \bar{a}_i\right)^2}. \tag{18}$$

Unlike the previous version (Holý, 2016), we introduce the coverage coefficient that improves the error function and solves situations when the error function has a lower output value but the alignment is worse. It is necessary to realize that the error area between two corresponding line segments is also affected by the interval where both functions are defined. Based on this, we need the minimal area between the corresponding lines at the widest interval.

2.5. Minimization

To minimize (17), the method of gradient descent can be used. The values of gradient can be computed analytically as partial derivatives of the expression in (17). Since the transformation can change correspondences, it is necessary to transform the LIDAR function in every iteration of gradient descent. Because of this, the analytically computed gradient has the most accurate value at $(0, 0, 0)$. The gradient at $(0, 0, 0)$ can be computed as follows

$$\frac{\partial \mathrm{E}}{\partial t_x} = \sum_{i=0}^{n} \left[2\cos(\alpha_i')\left(l_i' \tan(\varphi - \alpha_i') - l_i P_i(\varphi)\right)\right]_{\bar{a}_i}^{\bar{b}_i}, \tag{19}$$

$$\frac{\partial \mathrm{E}}{\partial t_y} = \sum_{i=0}^{n} \left[2\sin(\alpha_i')\left(l_i' \tan(\varphi - \alpha_i') - l_i P_i(\varphi)\right)\right]_{\bar{a}_i}^{\bar{b}_i}, \tag{20}$$

$$\frac{\partial \mathrm{E}}{\partial \theta} = \sum_{i=0}^{n} \left[\frac{-l_i'^2}{\cos^2(\varphi - \alpha_i')} + 2 l_i l_i' \left(\frac{\tan(\varphi - \alpha_i')}{\sin(\alpha_i - \alpha_i')} - \cot(\alpha_i - \alpha_i') P_i(\varphi)\right)\right]_{\bar{a}_i}^{\bar{b}_i}. \tag{21}$$

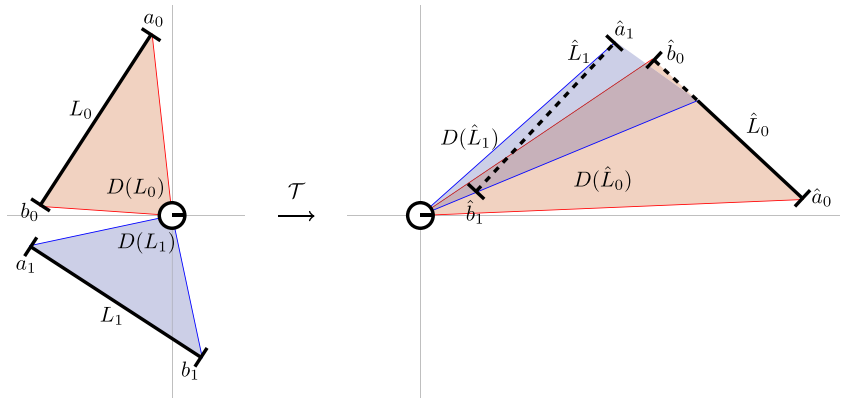

Fig. 2. The transformed line segment $\hat{L}_0$ is only partially visible because the domains $D(\hat{L}_0)$ and $D(\hat{L}_1)$ are not disjoint. And the transformed line segment $\hat{L}_1$ seems to be scanned from the opposite side ($\hat{a}_1 > \hat{b}_1$).

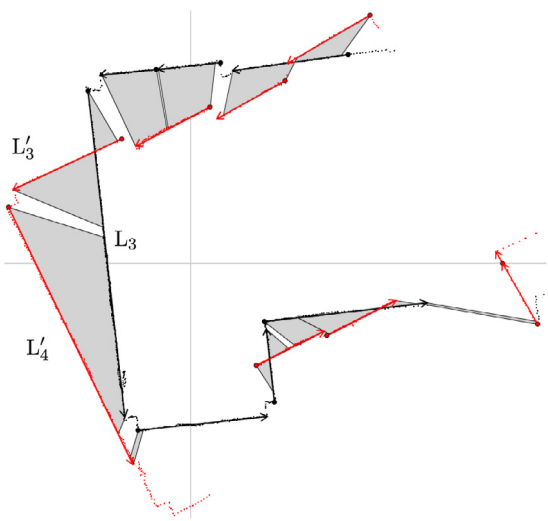

Fig. 3. Gray areas show the errors between the corresponding line segments. A line segment (e.g. L_3) can be corresponding for more line segments (L'_3 and L'_4) of the second LIDAR function.

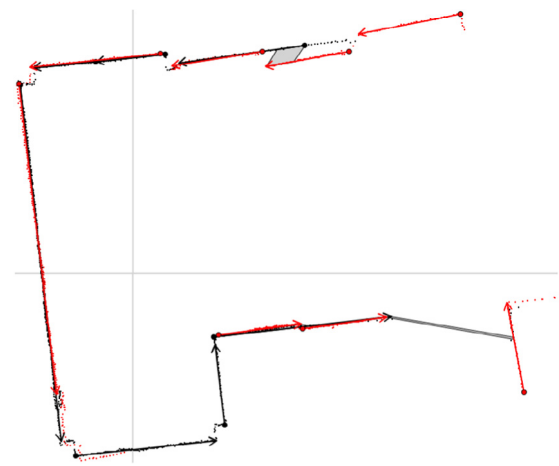

Fig. 4. The result of minimization of the LIDAR scans from Fig. 3 when the algorithm converged to a minimum.

The step size of the gradient descent is changed in every iteration. When the minimization starts, the initial guess of the step size is increased as long as the error is minimizing. The step size has to be decreased when the next error value is higher than the previous one.

The result of minimization of the LIDAR scans from Fig. 3 is shown in Fig. 4. The algorithm minimized the error between the LIDAR scans (gray areas in Fig. 3) and converged to an accepted minimum. Fig. 5 visualizes the error function before and after the minimization.

3. Results

The algorithm has been tested with many indoor datasets including Belgioioso castle (Haehnel, 2000a), Intel Research Lab (Haehnel, 2000b), own data scanned at our university, and also data from a simulated environment. Every test has compared the proposed method against PSM and two variants of ICP (TrICP and Sparse-ICP). The implementation of TrICP was taken from libpointmatcher (Pomerleau et al., 2013), the implementation of Sparse-ICP from libsparseicp (Bouaziz et al., 2013), and PSM from libpsm-v0.31 (Diosi and Kleeman, 2005). We present the results of Intel Research Lab dataset and Belgioioso castle dataset in this paper.

Since the scan matching algorithms are used in the early step of SLAM when no additional information (except odometry) are available, the algorithms have to be able to align two consecutive scans without any additional information as well. The experiments take this into account and align only two consecutive LIDAR scans to build a map with no SLAM technique. This testing approach was chosen deliberately to clearly show the precision of the registration algorithms. Testing the registration algorithms within a SLAM setup would show the final map aligned mostly by the SLAM technique and not by the registration algorithm itself. The SLAM algorithms are more robust in mapping because they often use loop-closing techniques and are designed for building a complete map. In contrast, the registration algorithms are designed to align only two consecutive scans and not every scan in the whole map. However, in this testing approach, we create a complete map with the registration algorithms only to cumulate the error of alignment so the final map can show how accurate the algorithms were.

The experiments measured the computation time and alignment of the compared algorithms. The final map was compared in several ways. At first, we used the mean squared error (MSE) to measure the error of the final alignment. The closest point to the measured point was considered as the true value for MSE. It was searched in all sets except the one containing the measured point. The second way used ground truth data and compared the estimated locations with the correct ones.

Fig. 6 shows one of the results of registration of all compared scan matching algorithms. We used the first 1,700 scans to create the loop around the center of Intel Research Lab. As the figure shows, the first 1,700 scans are enough to see the accumulated error because even our algorithm accumulated a little error that can be seen as the "double wall" on the top of Fig. 6(a). This is probably caused due to getting stuck in a local minimum during minimization of (17) for one pair of scans.

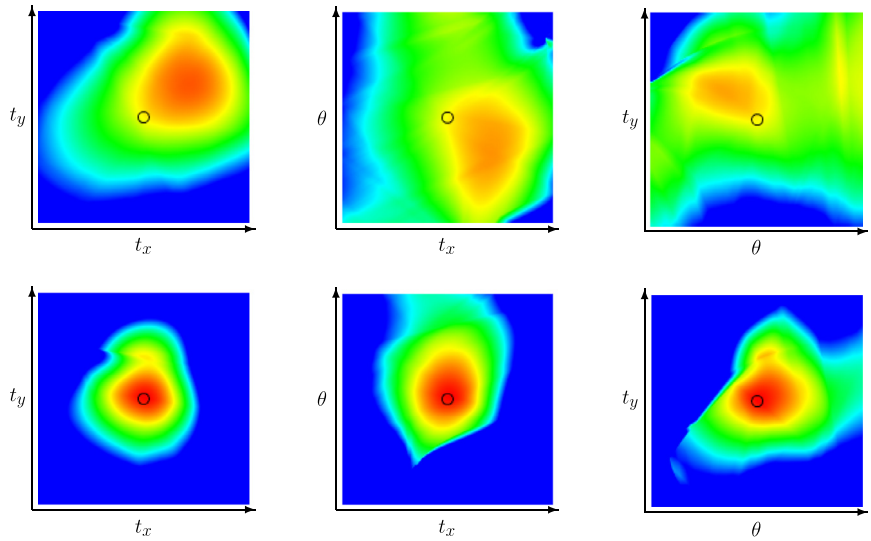

Fig. 5. Visualization of the error function before the minimization (the first row; Fig. 3) and after (the second row; Fig. 4). The circles in the center of images show the current error value. Red color means a lower value.

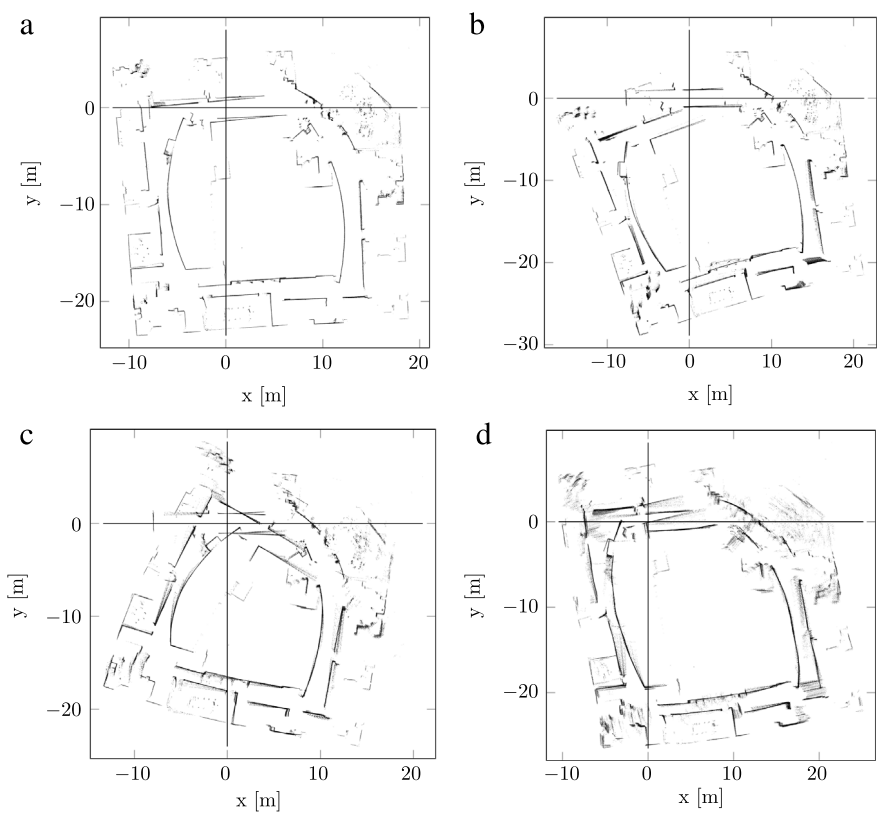

Fig. 6. The result of the first 1,700 scans of the Intel Research Lab aligned by the presented algorithm (a), Sparse-ICP (b), TrICP (c), and PSM (d). Every algorithm accumulated bigger error in the location and orientation than the new algorithm.

The other compared algorithms accumulated bigger error during the registration than the presented one. For instance, the results of Sparse-ICP (shown on Fig. 6(b)) shows a bigger error on every scan, which results in the mismatch of the beginning and the end of the final map. Similar problems have been found with the rest of the tested algorithms.

Since every scan-matching algorithm is accumulating error during map building, Fig. 7 shows cumulated location and orientation errors.

As mentioned above, the alignment of the final map was compared with MSE as well. The results of the computation time and MSE along with the averages of location and orientation errors are shown in Table 1. The bold numbers mean better results.

Similar experiments were performed with the Belgioioso castle as well. Fig. 8 shows one of the results with 400 consecutive scans of the castle that were captured during the move through three rooms.

Cumulated location and orientation errors of the Belgioioso castle are shown in Fig. 9.

Also the second presented experiment was compared by MSE, average time, location, and orientation errors as is shown in Table 2.

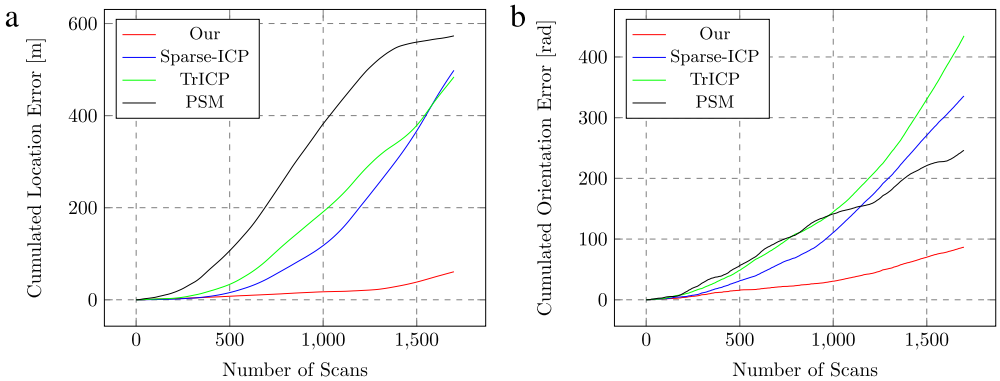

Fig. 7. The comparison of cumulated errors of the Intel Research Lab between the estimated and correct locations (a) and orientations (b) for all tested algorithms.

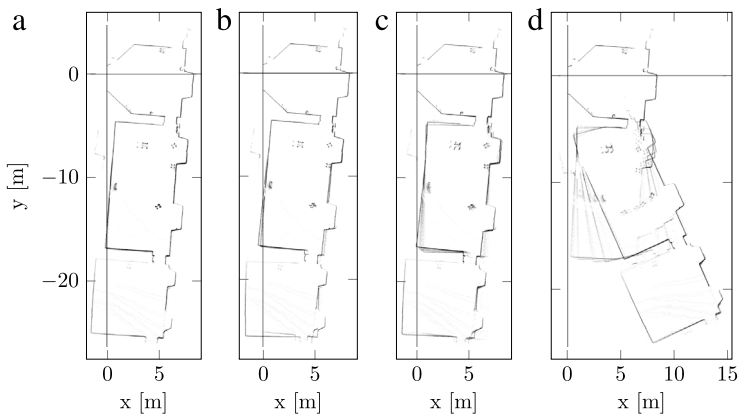

Fig. 8. The result of the 400 scans of the Belgioioso castle dataset aligned by the presented algorithm (a), Sparse-ICP (b), TrICP (c), and PSM (d). Every algorithm accumulated bigger error in the location and orientation than the new algorithm.

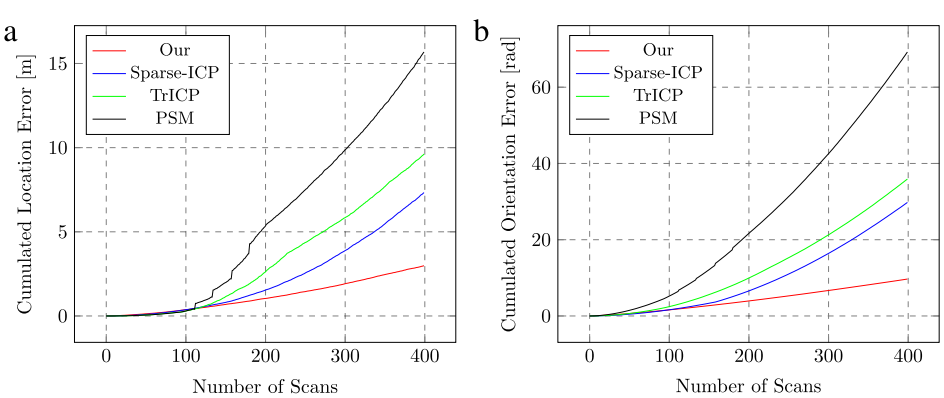

Fig. 9. The comparison of cumulated errors of the Belgioioso castle between the estimated and correct locations (a) and orientations (b) for all tested algorithms.

Table 1
The comparison of MSE and averages of computation time, location and orientation errors of the Intel Research Lab dataset.

Algorithm	MSE	Time [ms]	Loc. [cm]	Orien. [rad]
Our	**0.732**	2.11	**35.806**	**0.0509**
Sparse-ICP	1.044	81.75	293.274	0.1976
TrICP	2.06	10.43	284.917	0.2559
PSM	2.93	**1.95**	337.391	0.1449

Table 2
The comparison of MSE and averages of computation time, location and orientation errors of the Belgioioso dataset.

Algorithm	MSE	Time [ms]	Loc. [cm]	Orien. [rad]
Our	**0.918**	2.43	**7.472**	**0.0243**
Sparse-ICP	1.127	65.21	18.371	0.0746
TrICP	1.816	9.48	24.122	0.0901
PSM	3.081	**2.18**	39.301	0.1735

All of these experiments were performed on a single core of 1.90 GHz Intel® Core™ i7–3517U in C++ on Linux OS.

4. Conclusion

A novel algorithm for registration of LIDAR scans has been presented. The algorithm uses line segments that are expressed in polar coordinates as functions of angles. This description of the surrounding space around the LIDAR scanner with continuous LIDAR function in polar coordinates brings a better final alignment and lower computation time because there is no need for searching for the correspondences, and the gradient can be computed analytically. The experiments proved that the presented algorithm is accurate and fast. The future focus will be concentrated on a 3D variant of the algorithm that could bring even better improvements in the computation time and alignment.

Acknowledgments

This work was partially supported by the Grant of SGS No. SP2017/61, VŠB - Technical University of Ostrava, Czech Republic.

References

Alshawa, M., 2007. ICL: Iterative closest line. Ekscentar (10), 53–59.
Besl, P.J., McKay, N.D., 1992. A method for registration of 3-D shapes. IEEE Trans. Pattern Anal. Mach. Intell. 14 (2), 239–256.
Biber, P., Strasser, W., 2003. The normal distributions transform: A new approach to laser scan matching. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 3, pp. 2743–2748.
Bouaziz, S., Tagliasacchi, A., Pauly, M., 2013. Sparse iterative closest point. Comput. Graph. Forum 32 (5), 113–123.
Chetverikov, D., Svirko, D., Stepanov, D., Krsek, P., 2002. The trimmed iterative closest point algorithm. In: Proceedings of International Conference on Pattern Recognition, vol. 16, pp. 545–548.
Cox, I.J., 1991. Blanche–an experiment in guidance and navigation of an autonomous robot vehicle. IEEE Trans. Robot. Autom. 7 (2), 193–204.
Diosi, A., Kleeman, L., 2005. Laser scan matching in polar coordinates with application to slam. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1439–1444.
Ekekrantz, J., Pronobis, A., Folkesson, J., Jensfelt, P., 2013. Adaptive iterative closest keypoint. In: Proceedings of European Conference on Mobile Robots, pp. 80–87.
Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24 (6), 381–395.
Fitzgibbon, A.W., 2003. Robust registration of 2D and 3D point sets. Image Vis. Comput. 21 (13–14), 1145–1153.
Haehnel, D., 2000a. Belgioioso castle dataset. http://www2.informatik.uni-freiburg.de/~stachnis/datasets.html.
Haehnel, D., 2000b. Intel research lab dataset. http://www2.informatik.uni-freiburg.de/~stachnis/datasets.html.
Holý, B., 2016. Registration of lines in 2D LIDAR scans via functions of angles. IFAC-PapersOnLine 49 (5), 109–114.
Lingemann, K., Nüchter, A., Hertzberg, J., Surmann, H., 2005. High-speed laser localization for mobile robots. Robot. Auton. Syst. 51 (4), 275–296.
Magnusson, M., Lilienthal, A., Duckett, T., 2007. Scan registration for autonomous mining vehicles using 3D-NDT. J. Field Robot. 24 (10), 803–827.
Masuda, T., Sakaue, K., Yokoya, N., 1996. Registration and integration of multiple range images for 3-D model construction. In: Proceedings of International Conference on Pattern Recognition, vol. 1, pp. 879–883.
Nguyen, V., Gächter, S., Martinelli, A., Tomatis, N., Siegwart, R., 2007. A comparison of line extraction algorithms using 2D range data for indoor mobile robotics. Auton. Robots 23 (2), 97–111.
Pavlidis, T., Horowitz, S.L., 1974. Segmentation of plane curves. IEEE Trans. Comput. C-23 (8), 860–870.
Pomerleau, F., Colas, F., Siegwart, R., Magnenat, S., 2013. Comparing ICP variants on real-world data sets: Open-source library and experimental protocol. Auton. Robots 34 (3), 133–148.
Rusinkiewicz, S., Levoy, M., 2001. Efficient Variants of the ICP Algorithm. pp. 145–152.
Segal, A., Haehnel, D., Thrun, S., 2010. Generalized-ICP. In: Proceedings of Robotics: Science and Systems, vol. 5, pp. 161–168.
Turk, G., Levoy, M., 1994. Zippered polygon meshes from range images. In: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques. ACM, pp. 311–318.
Zhang, Z., 1992. On local matching of free-form curves. In: BMVC92: Proceedings of the British Machine Vision Conference. Springer, pp. 347–356.
Zhang, Z., 1994. Iterative point matching for registration of free-form curves and surfaces. Int. J. Comput. Vis. 13 (2), 119–152.