Simulation of a mobile robot with a LRF in a 2D environment and map building

Teslić L.[1], Klančar G.[2], and Škrjanc I.[3]

[1] Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, 1000 Ljubljana, SLOVENIA luka.teslic@fe.uni-lj.si
[2] Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, 1000 Ljubljana, SLOVENIA gregor.klancar@fe.uni-lj.si
[3] Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, 1000 Ljubljana, SLOVENIA igor.skrjanc@fe.uni-lj.si

1 Introduction

This artcile deals with the modelling and simulation of a mobile robot with a laser range finder in a 2D environment and map building. The simulator is built in the Matlab Simulink environment, therby taking advantage of the powerful Matlab toolboxes for developing mapping, localization, SLAM and navigation algorithms. A map-building algorithm is developed and tested with a simulation. The line segments, extracted from the LRF's output in each scan, are made up of polylines, which are merged with the existing global map to form a new global map. The global map of the environment is represented by unions of line segments, where each union represents an object in the environment. Map building, localization and navigation are important issues in mobile robotics. To develop and test algorithms for executing tasks of this kind, it is useful to have a simulator of a mobile robot equipped with sensors in a static environment. Since a Laser Range Finder (LRF) is often used as the basic interaction between the robot and the environment, the represented mobile robot model also includes a model of the LRF. The problem of robotic mapping and localization has been widely studied. A robot must know its own pose (localization problem) in order to build a map, and the robot also needs to know the environment map (mapping problem) to localize itself to its current pose. The problems of mapping and localization can be handled separately if the robot's pose is given to the robot by a human or from using GPS and INU sensors (outdoor environments) when map building. The map of the environment can then be used to solve the localization problem. To avoid the known robot's pose assumption, a SLAM (Simulta-

neous Localization and Mapping) algorithm must be built, where the problems of localization and mapping are merged. The robot can localize itself by odometric measurements and by comparing the local map, obtained from the current view of the robot, with an already built global environment map. In [1] a comprehensive survey of the SLAM problem is addressed. In the literature different approaches to the map building were proposed. A topological map [3] is composed of the nodes representing topological locations and the edges between the nodes. These nodes contain information about the way to reach a connected topological location. In [3] the metric and topological paradigm are integrated into a hybrid system for map building. A global topological map connects local metric maps, allowing a compact environment model, which does not require global metric consistency and provides both precision and robustness. The metric approach builds a map with occupancy grids or with simple geometrical features (e.g., line segments). Occupancy grids require a huge amount of computer memory and are therefore not appropriate when modelling a large environment [4]. In this paper we chose line segments for the environment model as they require a smaller amount of computer memory. In [5] a comparison of line-extraction algorithms using a 2D laser rangefinder is reported. In [6] the environment is represented by polygonal curves (polylines), possibly containing rich shape information for matching environment scans. However, some environment objects can not be represented by one polyline (consecutive line segments). Therefore, each environment object is represented by the union of inconsecutive line segments in this paper.

2 Simulator

The main reason to develop a new simulator instead of using one of the many already available is to study navigation, localization and mapping algorithms in the Matlab Simulink environment. Matlab and its tool-boxes (e.g., Fuzzy Logic, Control, etc.) represent a very powerful tool for developing all kinds of algorithms. The simulator includes the models of a mobile robot (Fig.1 (a)), a laser range finder and the environment. The purpose of the modelling is to create a simulation model where different algorithms for mapping can be tested. We assume a two-dimensional environment and that the robot knows its own pose

$$\mathbf{p}(t) = [x_{rob}(t), y_{rob}(t), \varphi_{rob}(t)] \tag{1}$$

at time t, in a global frame of reference (Fig.1 (a)). The denotation for time, t, will be subsequently abandoned for simplicity.

2.1 Robot Model

The kinematic model of the robot is given by the following equations

$$\dot{x}_{rob} = v\cos\varphi_{rob}, \quad \dot{y}_{rob} = v\sin\varphi_{rob}, \quad \dot{\varphi}_{rob} = \omega, \tag{2}$$

where the input v denotes the tangential speed, input ω denotes the angular speed, (x_{rob}, y_{rob}) denotes the position of the robot in global coordinates (x_G, y_G) and φ_{rob} denotes the orientation of the robot according to the global coordinate axis, x_G. The continuous model (Eq. (2)) is implemented in Matlab Simulink with a simulation scheme using the $ode45$ integration method. For simulation purposes it is enough to control the robot with the inputs v and ω. The pose (Eq. (2)) is the input in the S-function *Animation* for simulating the environment model and the LRF model and the input in the S-function for the map-building algorithm.

2.2 Environment Model

The two-dimensional model of the environment can be built with line segments. The line segment is defined with two points on the line and the normal line equation

$$x_G\cos\alpha_j + y_G\sin\alpha_j - p_j = 0, \tag{3}$$

where the parameter p_j denotes the distance of the line from the origin, parameter $\alpha_j \in (\pi-, \pi]$ denotes the direction of the line normal passing through the origin and x_G, y_G are the global coordinates of the points lying on the line.

2.3 Laser Range-Finder Model

The laser range finder in each time step gives the set of distances $\mathbf{d}_s = [d_{s0^\circ}, ..., d_{s180^\circ}]$ to obstacles (e.g., a wall) at angles $\boldsymbol{\theta}_s = [0^\circ, ..., 180^\circ]$. We simulate a reflection point by calculating the intersection points $(x_{ip}(i,j),$ $y_{ip}(i,j))$ between the i-th laser beam line (Fig. 1 (b)) and all $(j = 1, ..., N)$ the lines describing the environment line segments with determinants and calculate distances and angles

$$\begin{aligned} d(i,j) &= \sqrt{(x_{ip}(i,j) - x_{rob})^2 + (y_{ip}(i,j) - y_{rob})^2}, \\ \theta(i,j) &= atan2(y_{ip}(i,j) - y_{rob}, x_{ip}(i,j) - x_{rob}) - (\varphi_{rob} - 90^\circ). \end{aligned} \tag{4}$$

If there is no intersection point between the lines, this is labelled with

$$d(i,j) = D; \quad D > d_{max}, \tag{5}$$

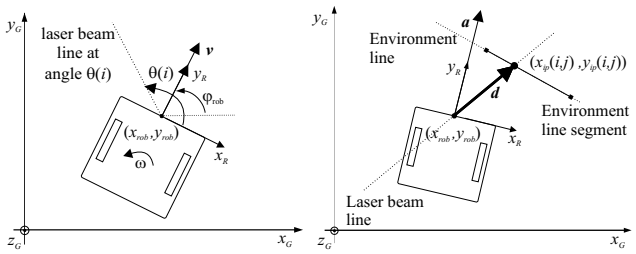


Fig. 1. (a) Coordinates of the robot according to the global coordinates. (b) Intersection point between the laser beam line and the environment line.

where d_{max} denotes the maximal range of the LRF (e.g., $80m$). In the case of concurring lines the nearest point of the environment line segment to the robot (x_{rob}, y_{rob}) is chosen as the reflecting point. Because the intersections between the lines are calculated, the intersections behind the robot and the intersections in front of the robot, which do not lie on the environment line segments, must be eliminated (labeled with Eq. (5)). Furthermore, we choose a point with the minimum distance from the robot $d(i) = min(d(i,1), ..., d(i,N))$ as the reflection point. So, if there is no intersection point between the $i-th$ laser beam and all the environment lines, the distances $d(i,:)$ take the value D and $d(i) = D$. If there are more environment lines in front of the robot, the nearest intersection point is chosen as the reflecting point of the laser beam. We define the vectors (Fig. 1 (b)) $\mathbf{a} = (\cos\varphi_{rob}, \sin\varphi_{rob}, 0)$ and $\mathbf{d} = (x_{ip}(i,j) - x_{rob}, y_{ip}(i,j) - y_{rob}, 0)$. If the dot product $\mathbf{a} \cdot \mathbf{d} < 0$, the intersection point lies behind the robot and it is eliminated. If $\theta(i)$ equals $0°$ ($180°$) and there is an environment line on the left (right) side of the robot, an intersection point between lines, which does not appear with the real LRF, is eliminated. This situation happens when

$$\mathbf{c} = \mathbf{a} \times \mathbf{d} = (0, 0, c_3); \quad c_3 > 0 \ (c_3 < 0), \tag{6}$$

where the operator $\times$ denotes the cross product. We assume the LRF noise model using

$$d_{noise}(i) = d(i) + \frac{d(i)}{d_{max}} N(0, \sigma), \tag{7}$$

where $N(0, \sigma)$ denotes the normal distributed noise with zero mean and σ variance. If the distance $d(i)$ is large, the influence of the noise is proportionally larger. In this way we can test the robustness of mapping algorithms without knowing the real noise distribution.

3 Mapping algorithm

Assuming that the robot knows its own pose (Eq. 1)) the calculated reflection point ($x_G(i),\ y_G(i)$) according to the global coordinates is

$$\begin{aligned} &x_G(i) = x_{rob} + d(i)\cos\theta_G(i), \quad y_G(i) = y_{rob} + d(i)\sin\theta_G(i), \text{ and} \\ &\theta_G(i) = (\varphi_{rob} - 90^\circ) + \theta_s(i), \quad i = 1, ..., n, \end{aligned} \tag{8}$$

where $\theta_G(i)$ denotes the laser-beam angle according to the global coordinate frame (Fig.1 (a)). All consecutive points (8) by which a reflection has occurred ($d(i) \leq D$) are clustered, other points ($d(i) > D$) are ignored. If there is only one point in a cluster, it is also ignored. Each cluster is then split into more clusters if the distance between two following points is greater than the threshold. Finally, every cluster is reduced in the set of consecutive line segments or polylines using the split-and-merge algorithm (it is fast and has good accurate) [5] and least-squares line fitting. The line segments are defined with edge points and line parameters p_k (the distance of the line from the origin) and $\alpha_k \in (\pi-, \pi]$ (the direction of the line normal passing through the origin).

3.1 Integrating the global map with the local map

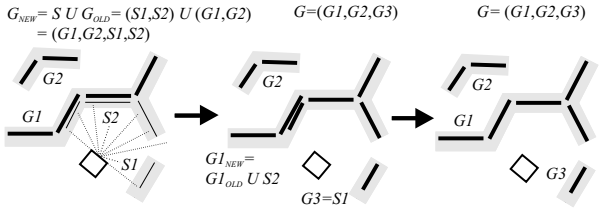
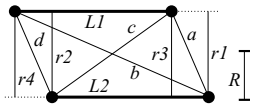


Fig. 2. Integrating the global map G =($G1, G2$) with the local map S =($S1, S2$).

Each local map S (Fig.2) represents a set of polylines $(S1, S2)$, and each polyline is composed of consecutive line segments described with line parameters and edge points. The global map G (Fig.2, right) is composed of the unions ($G1$, $G2$ and $G3$) of line segments, which represent objects in the environment. The local map S is united with the previously built global map G_{old} to get a new global map $G_{NEW} = S \cup G_{OLD}$. When a robot makes its second local map S, G_{OLD} is equal to the local map S_{FIRST}, obtained in the first environment scan. When uniting S and G_{OLD} each line segment of



$$c1 : a + b < l(L1) + T, \quad c2 : c + d < l(L1) + T,$$
$$c3 : a + c < l(L2) + T, \quad c4 : b + d < l(L2) + T,$$
$$c5 : r1 < R, \quad c6 : r2 < R,$$
$$c7 : r3 < R, \quad c8 : r4 < R,$$

Fig. 3. The conditions for merging the line segments $L1$ and $L2$.

the set (G_{OLD}, S) is compared to each line segment in one loop. We define the conditions for merging the line segments $L1$ and $L2$ (Fig. 3), which correspond to the same environment line segment for the thresholds T and R, where $l(Li)$ denotes the length of line segment Li. If at least two of conditions c_i $(i = 1, 2, 3, 4)$ are satisfied and if all conditions c_i $(i = 5, 6, 7, 8)$ are satisfied, two line segments are merged. If the conditions c_1 and c_2 or the conditions $c3$ and $c4$ are satisfied, two line segments are also merged. When merging two line segments, new line-segment parameters are computed by uniting the edge points of both segments, as indicated in [7]. If the merged line segments belong to different unions, the unions are merged (Fig.2, $G1_{NEW} = G1_{OLD} \cup S2$). The loop of comparison is then repeated. If the conditions for merging any two segments are not fulfilled in the next loop, the loop is stopped.

In [2] the SLAM algorithm for a line-based environment representation is described. The global environment map is composed of the set of the global environment lines (e.g., 1000) and the local environment map is also composed of a set (e.g., 10) of local environment lines. For localization purposes the line segments of the global map that correspond to the same environment line segments (e.g., a wall) as the line segments of the local map, must be found. The line parameters of the local (robot's coordinates) map (α_{Li}, p_{Li}) are recomputed to global coordinates according to the approximately known robot pose [2] (the prediction step of the Extended Kalman Filter). In general, parameters (α_{Gk}, p_{Gk}) of each global line must be compared to recomputed parameters (α'_{Li}, p'_{Li}) of each local line to find the corresponding pair of lines. In large environments this can take a huge number of comparisons (e.g., 1000×10). If the sum of the squared differences $(\alpha_{Gk} - \alpha'_{Li})^2 + (p_{Gk} - p'_{Li})^2$ is below a threshold, the lines can be chosen as a corresponding pair. In our mapping approach the map is composed of unions of lines, where each union corresponds to an environment object. It is very likely that objects seen by the robot in the previous environment scan are also seen in the current environment scan. There could be many line segments that are seen in the current environment scan that correspond to objects (unions Gi) seen in the previous environment scan. The search strategy can be faster if the line pairs for the comparisons are first found among all the global lines that correspond to the

environment objects seen in the previous environment scan and all the local lines (recomputed parameters) of the current scan.

4 Results

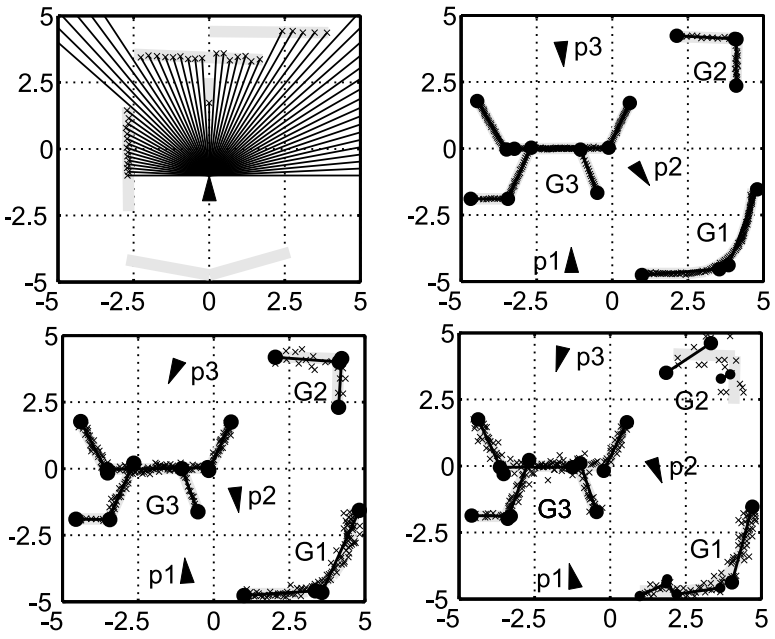


Fig. 4. (a) Reflecting points of the LRF in a 2D environment. (b), (c) and (d) Experiment of building map $G = (G1, G2, G3)$ at $d_{max} = 14m$ and (b) $\sigma = 0.07m$, (c) $\sigma = 0.42m$ and (d) $\sigma = 0.63m$.

Fig. (4) (a) shows a simulated LRF, a mobile robot and a 2D environment. Figures (4) (b), (c) and (d) show experiments at different values of LRF noise variance σ ($0.07m$, $0.42m$ and $0.63m$) and a fixed value $d_{max} = 14m$ (Eq. (7)), where the global environment map G is built. The environment scans are taken at poses $p1$, $p2$ and $p3$. The experiments shows that our mapping algorithm is robust, even at a lot of noise from the LRF. Our mapping approach builds a global map with many sets (unions) of line segments, which correspond to the environment objects. Compared to occupancy grids [4] the environment description with the line segments requires a much smaller amount of computer memory. In [6] the environment is represented by polygonal curves (polylines), possibly containing rich shape information, which

can be important when matching consecutive environment scans for localization purposes. The object represented by the union of the lines $G3$ (6 lines) in Fig. 4 (b) could not be represented by one, but four polylines (consecutive line segments) seen from the left, right, lower and upper sides of this object (14 lines). Environment representation with polylines could require more computer memory than our representation with unions of line segments.

5 Conclusion

In this paper we represent a simulator of a mobile robot with a LRF in a 2D static environment and propose a map-building algorithm, which is tested on the simulator. An environment map describes each environment object with a union of line segments. In this way the search strategy to find pairs of line segments for localization purposes could be faster than with an environment map, which is composed of only one set of line segments. The mapping algorithm is fast enough for real-time applications and will be integrated into the SLAM algorithm in the future.

References

1. Thrun S (2002) Robotic Mapping: A Survey. In: Lakemeyer G, Nebel B, (eds) Exploring Artificial Intelligence in the New Millenium. Morgan Kaufmann
2. Garulli A, Giannitrapani A, Rossi A, Vicino, A (2002) Mobile robot SLAM for line-based environment representation. 44th IEEE Conference on Decision and Control and European Control Conference.
3. Tomatis N, Nourbakhsh I, Siegwart R (2003) Hybrid simultaneous localization and map building: a natural integration of topological and metric. Robotics and Autonomous Systems 44:3–14
4. Veeck M, Veeck W (2004) Learning polyline maps from range scan data acquired with mobile robots. IEEE/RSJ International Conference on Intelligent Robots and Systems. Proceedings.
5. Nguyen V, Martinelli A, Tomatis N, Siegwart R, (2005) A Comparison of Line Extraction Algorithms using 2D Laser Rangefinder for Indoor Mobile Robotics. IEEE/RSJ International Conference on Intelligent Robots and Systems.
6. Latecki L J,Lakaemper R, Sun X, Wolter D (2004) Building Polygonal Maps from Laser Range Data. ECAI International Cognitive Robotics Workshop, Valencia, Spain, August.
7. Mazl R, Preucil L (2000) Building a 2D Environment Map from Laser Range-Finder Data. Proceedings of the IEEE Intelligent Vehicles Symposium.