IOP Conf. Series: Materials Science and Engineering **516** (2019) 012057 doi:10.1088/1757-899X/516/1/012057

Methods for robot localization on a map

M V Shikhman, S V Shidlovskiy

National Research Tomsk State University, Tomsk, Russia
E-mail: Shikhmar@gmail.com

Abstract. In many cases, to solve applied problems, the robot needs to know its real location, which is most often different from the data stored in the on-board system. For unmanned robotic devices, as well as for ground-based robots, it is most efficient to use local navigation algorithms, which consist in determining the coordinates of the device with respect to a certain starting point. The paper discusses localization algorithms provided that the map of the area is known in advance. Particular attention is paid to the Monte Carlo localization method because of several advantages. The paper presents an example of modeling the algorithm operation.

1. Introduction

The world has entered the digital era when new technologies are rapidly developing and changing the habitual way of life, new industries and professions are being formed, and new opportunities for development are appearing. Among the variety of areas in the digital industry, robotics is of particular interest. Robotic devices gradually penetrate into all spheres of human activity and have great prospects for further development. Many problems need to be solved to provide the subsequent evolution of such devices.

Mobile robots able to move independently in space are singled out the variety of robotic devices. For successful navigation, the robot must have a map of the area or skills to build it directly in the process of movement; pre-determine the route and be able to adjust it in the process of movement; control movement parameters to change its position; obtain environmental information and process it further; keep track of its location. In this paper, we consider methods for solving the latter problem, provided that the terrain map is known in advance. Localization is one of the most common tasks of robotic perception because knowing location of objects and the acting subject is the basis of any successful physical interaction.

2. The basic principles of building robot localization algorithms

The development of algorithms for robot localization consists of several areas of research including a mathematical and geometric description of navigation processes, computer technologies and technical devices that are necessary for implementing these processes. Gyroscopes, encoders, accelerometers, stereoscopic systems, laser range finders, and ultrasonic sonars can serve as technical devices for obtaining primary information about the environment, which is necessary for the operation of algorithms. Let us proceed to the description of the main algorithms used to localize the robot.

3. Iterative Closest Point

The most common algorithm for solving localization problems is the iterative closest point algorithm (ICP), which, in addition to determining the position of the robot, allows working with algorithms for finding the optimal route. The main idea of ICP is to combine images of the same object obtained

from different angles but with common areas, the so-called overlapping areas. This is implemented by means of iterative minimization of the average distance between two clouds of points. To do this, first, an initial assessment of a rough conversion of one cloud to another is made followed by refinement during the minimization process. As a result, the algorithm finds the best transformation of one cloud to another for two given clouds of points P_1 and P_2 [1].

The task of minimizing the average distance between the clouds of points P_1 and P_2 may be represented as follows:

$$F = \frac{1}{N}\sum_{i=1}^{N} d\left(p_i^1; p_i^2\right) \longrightarrow \min$$

where $\left[p_i^1;\ p_i^2\right]_{i=1...N}$ is a set of pairs of closest points;

$d\left(p_i^1; p_i^2\right)$ is Euclidean distance between the closest points.

To calculate the distance between the point p_1 belonging to the cloud P_1 and the cloud P_2, you can use the so-called 'point-to-point distance' metrics. When using them, the distance between points p_1 and p_2 is minimized.

Another metric is 'point-to-plane distance', which minimizes the sum of squares of distances from p_i^1 to a plane S'_1 that is perpendicular to the plane S_1 at a point p_i^2 for all pairs of closest points.

In general, the iterative closest point algorithm includes the following steps.

- search for the closest pairs of points $\left[p_i^1;\ p_i^2\right]_{i=1...N}$;
- search for shift and rotation parameters, which reduce the F error by minimizing the distance or by the method of least squares;
- application of the found parameters for the point cloud P_1;
- steps 1-3 are repeated until the F error becomes less than a certain threshold value.

4. The method of planes detected in point clouds

Presentation of the initial information received from the sensors in the form of point clouds is not always appropriate because it is associated with high costs of computing resources. To solve this problem, representation of points in the form of elementary geometric elements (vertices, lines and planes) is often used. In this case, objects are characterized by much smaller dimension and noise level; and finding various relations between them becomes simpler. Thus, the first step in using this method is detection of linear objects and their parameters, which can be done using the segmentation method by combining points with close normals.

Then it is necessary to find a match in the database for each detected object. The pair plane is found based on the rotation and movement of the sensors defining the plane. To do this, we consider the global and local coordinate system. The local coordinate system is connected directly to the sensor and has a clear description. We also know the approximate position of this system relative to the global one, which is described by the linear displacement vector and the rotation matrix allowing us to directly determine the global plane equation. However, this step may find several paired planes or not find them at all; therefore, the next step is necessary.

To determine the parameters of the robot motion, you need to use at least three pairs of planes. For n pairs of planes, you can write a nonlinear system consisting of n equations, each of which includes values of the rotation angles relative to each plane of the three-dimensional surface and the vector of the linear displacement. These parameters can be found by optimization methods. As a result, they must satisfy all pairs of planes with a certain given accuracy, which allows discarding erroneously proposed planes [2].

5. Algorithms based on space splitting using an octree type structure

The main idea of these algorithms is to represent the entire surrounding space as a cube and then divide it into 8 parts (8 small cubes). Such division continues until reaching certain conditions, usually including the volume of the cube and the number of elements contained in it. Information about the environment is compared with the initial (stored in the database) through hierarchical access. The robot processes only visible nodes and the nodes included in them, sequentially from top to bottom. When the environment is divided into a certain number of cubes, the robot only works with the visible part of the space, that is, it checks the child nodes only when the main node is visible to the camera [3].

6. Kalman filter-based localization algorithms

To solve the robot localization problems, the Kalman filter is often used, which applies information about the laws of the robot's motion at the forecast stage and information from the robot sensors at the correction stage. The state of the system is the navigation parameters of the robot position (usually, its coordinates and shift parameters). Predicting the evolution of the state is based on odometric data and measurements allowing you to calculate the robot position in accordance with the map. The Kalman filter allows you to evaluate the state of the system based on a noisy prediction of its evolution and noisy measurements of this state [4].

The Kalman filter algorithm consists of the following steps:

- prediction of the current state based on estimates at the previous time step and taking into account the control parameters for the current step;
- prediction of observation upon displacement based on the observation model and the assessment of the state;
- shift of the robot in space;
- correction of the predicted state taking into account the error between the predicted and implemented observation;
- space shift calculation;
- extraction of reference points from the surrounding space, observation of the state

7. Approximate robot localization algorithm

When describing the method of planes and detected points in clouds, we have already discussed the reasonability of representing a cloud of points in the form of elementary geometric cells. This principle is applicable to approximate algorithms for the robot localization. In this case, we may present a map of terrain in the form of a certain geometric figure, most often a polygon.

At some initial time, the robot is at some point *A*, and there is some area of visibility *S* at its disposal. To determine its current location, the robot needs to compare the area of visibility *S* with all areas of the original map stored in the database. However, there may be several similar areas on the map (Figure 1). Thus, hypothetically, the robot can be at any of the points A_n.

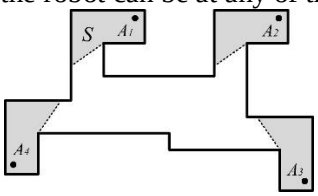

Figure 1. Variability of the robot position on the current map.

To determine exactly where the mobile robot is, it is most logical to use a probabilistic approach, which includes some kind of uncertainty of location. Thus, the probability of finding a robot at one or another point $P(A_n)$ is considered. During the process, the algorithm uses the methods of triangulation and construction of overlays [5].

8. Monte Carlo localization

In this case, to solve localization problems, the robot position is determined by the density of the distribution of probability of locating in a particular place approximated by a large set of particles. The particle is a hypothesis about the possible state of the robot at some point in time. The larger set of particles we have, the faster the program will find the right solution, that is the real position of the robot on the map.

The initial assumption of the algorithm is that there is usually a uniform random distribution of particles in the configuration space, that is, the robot does not have information about where it is located, and it is assumed that it can equally be at any point in space.

The algorithm includes two stages: prediction and correction. During the first stage, the robot motion from the previous time point to the current one and its angle of rotation are tracked. As a result, we obtain a preliminary set of approximating particles using a motion model. Here, the update of information about the location of each particle occurs. During the motion update, the robot predicts its new position based on the location of this trigger command, applying the simulated motion to each of the particles.

At the second stage, the search for paired planes is performed for each particle and there may exist several of them. To solve this uncertainty, a concept of 'particle weight' is introduced. The particle weight depends on how accurately the sensor readings coincide with the actual data and is calculated based on the sensors used. The final stage of correction is the formation of an approximating set of particles, in which a large particle belongs to the region with maximum weights [6].

8.1 Advantages of Monte Carlo localization

We should note that this method has several advantages. The particle filter, which is central to the Monte Carlo algorithm, can work with different probability distributions because it has a non-parametric representation. The time complexity of the particle filter is linear with respect to the number of particles, so you can find the optimal ratio of speed and accuracy. The implementation adapts to the available computing resources: the faster is the processor, the more particles can be generated and, therefore, we get a more accurate algorithm.

9. Implementation of the Monte Carlo localization

Based on the advantages of the Monte Carlo localization, the simulation was carried out specifically for this method. For this, we chose the following software: ROS (Robot Operating System), Gazebo software package, Rviz tool included in ROS, and MATLAB software package. In Gazebo, we used a TurtleBot robot simulator for simulation of the surrounding space. TurtleBot in Gazebo is a model of a real robotic device, which uses a Kinect sensor as a vision.

Then, using the Rviz tool, we created a map resulting from motion of the robot and receiving information from its sensor (Figure 2). Using the Matlab software package, the resulting map was transformed into a two-dimensional binary grid (Figure 3). Each cell in the grid has a value that represents true (1) or false (0) status of this cell.

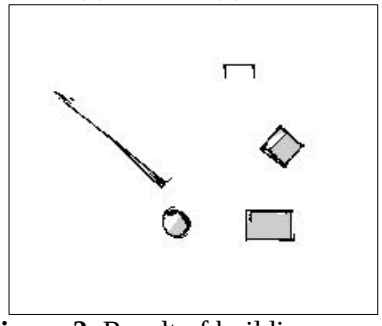

Figure 2. Result of building a map.

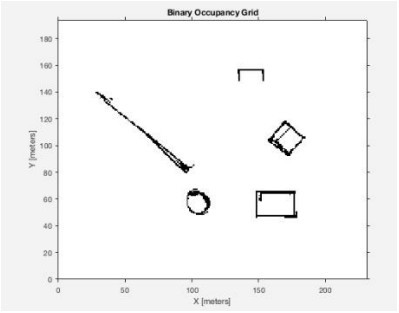

Figure 3. 2D binary grid.

The simulation of the Monte Carlo algorithm itself was carried out in the MatLab software environment. When applying this algorithm, there are two cases. First, when the approximate position of the robot is known and we just need to clarify it; and second, when the initial position of the robot is unknown. In this paper, we consider the second case because of its higher complexity. At the beginning, the algorithm assumes that the robot has an equal probability of being anywhere and generates uniformly distributed particles inside this space (Figure 4). The update occurs according to the algorithm and the current position of the robot is specified (Figure 5). As a result of the algorithm operation, the robot is localized (Figure 6).

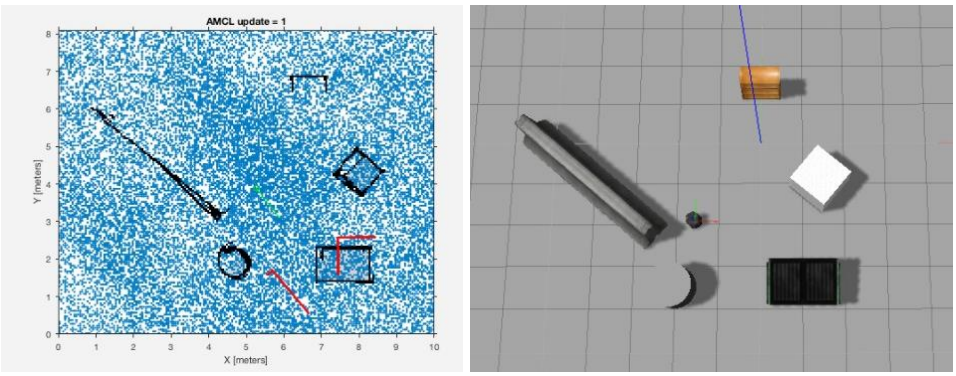

Figure 4. Start of the algorithm (third update).

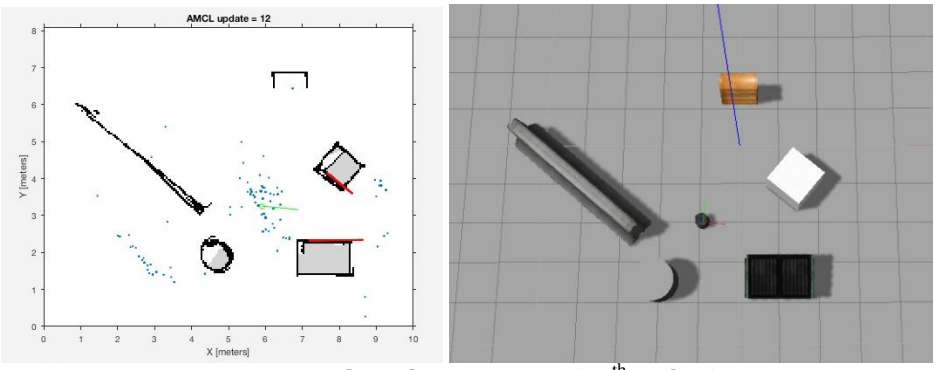

Figure 5. Algorithm operation (12th update).

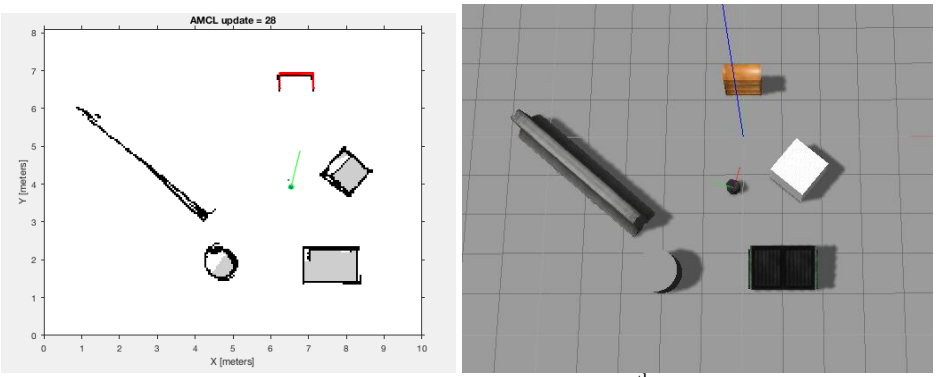

Figure 6. The end of the algorithm (28th update).

10.Conclusion

Monte Carlo localization allows specifying the current position of the robot when it moves and producing global localization in conditions of the absence of information about the robot location.

This algorithm has shown its effectiveness in solving the given problem, that is the robot localization provided that the terrain map is known in advance.

Acknowledgments
The research was carried out in Tomsk State University with the financial support of the Ministry of Education and Science of the Russian Federation, a unique project identifier: RFMEFI57817X0241. The authors are grateful to Tatiana B. Rumyantseva from Tomsk State University for English language editing.

References

[1] Borisov A. G., Gol S. A., Luksha S. S. 2013 *Vestnik RGRTU* **46 (4-3)** 35–42
[2] Kazmin V. N., Noskov V. P. 2015 *Proceedings of the Southern Federal University. Technical science* **8** 71–83
[3] Poslavskiy S. 2018 *Innovatika 2018* 118–120
[4] Gafurov O., Syryamkin V., Gafurov A. *et al.* 2012 *Telecommunications and Radio Engineering* **71**(17) 1565–1574 doi: 10.1615/TelecomRadEng.v71.i17.40
[5] Udovenko S., Sorokin A. 2014 *Information processing systems* **10** 248–254
[6] Dao Zuy Nam, Ivanovskiy S 2014 *Scientific Bulletin of NSTU* **55(2)** 109–121
[7] Sebastian Thrun *et al.* 2011 *Artificial Intelligence* **128** 99–141
[8] Kuznetsov D., Syryamkin V. 2015 *AIP Conference Proceedings* **1688** (040004) doi: 10.1063/1.4936037