МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Национальный исследовательский Томский государственный университет
Томский государственный университет систем управления и радиоэлектроники
Болгарская Академия наук
ООО «Научно исследовательское предприятие «Лазерные технологии»

ИННОВАТИКА-2019

СБОРНИК МАТЕРИАЛОВ

XV Международной школы-конференции студентов, аспирантов и молодых ученых
25–27 апреля 2019 г.
г. Томск, Россия

Под редакцией А.Н. Солдатова, С.Л. Минькова

Scientific & Technical Translations
STT
ИЗДАТЕЛЬСТВО
Томск – 2019

MONTE CARLO LOCALIZATION FOR MOBILE ROBOT

M.V. Shikhman
National Research Tomsk State University
Shikhmar@gmail.com

МЕТОД ЛОКАЛИЗАЦИИ МОНТЕ-КАРЛО ДЛЯ МОБИЛЬНОГО РОБОТА

М.В. Шихман
Национальный исследовательский Томский государственный университет

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. The article discusses Monte Carlo localization. The article presents the basic principles of the algorithm for the operation of a mobile robot.

Keywords: Monte Carlo localization, mobile robot, particle filter.

For successful navigation, the robot must constantly monitor its location, which is most often different from the data stored in the onboard system. For unmanned robotic devices, as well as ground-based robots, it's most effective to use local navigation algorithms, which consist in determining the coordinates of the device with respect to a certain starting point. In my work, I will talk about the advantages of the Monte Carlo localization method, the essence of the method and will show its implementation using the example of a simple robot.

Let us consider the 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 there is a trade-off between 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.

The essence of the algorithm itself is as follows. The algorithm uses a particle filter to represent the distribution of probable states. A particle is a possible state, that is, a hypothesis about where the robot is at some point in time. Most often, the initial representation of the algorithm is a uniform random distribution of particles in the configuration space. Whenever the robot moves, it moves the particles in order to predict its new state after movement. If the

robot determines familiar landmarks in the surrounding space, the particles are recalculated. Thus, it's determined how well the actual perceived data correlates with the predicted state. Ultimately, the particles must converge to the actual position of the robot [1].

Let us consider a robot in a one-dimensional circular corridor with identical doors. The robot uses a sensor that detects the presence or absence of a door in front of it. In this case, in general terms, we can describe the steps of the algorithm as follows. When the robot is located at a certain point in space, it determines the set of particles corresponding to its hypothetical locations. Then, for each particle, the robot calculates the probability that the indications about the environment on the map coincided with those of its sensors, if it were in this place. The weight is assigned to each particle. The weight is proportional to the specified probability. After that, the robot generates a set of new particles based on the previous representation with a probability that is proportional to the weights. Particles that are consistent with the readings of the sensors are chosen more often in contrast to particles incompatible with the readings of the sensors. Thus, the particles converge to the best estimate of the state of the robot. The robot is becoming more confident in its position [2].

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 1). Using the Matlab software package, the resulting map was transformed into a two-dimensional binary grid (Figure 2). Each cell in the grid has a value that represents true (1) or false (0) status of this cell.

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 3).

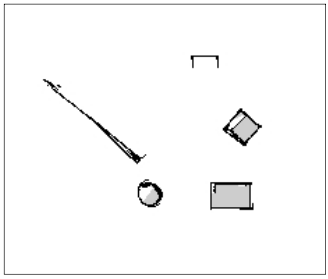


Fig. 1. Result of building a map

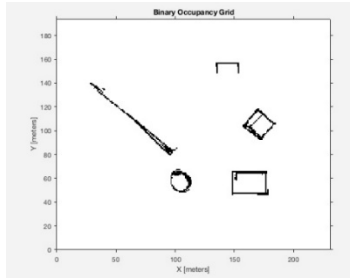
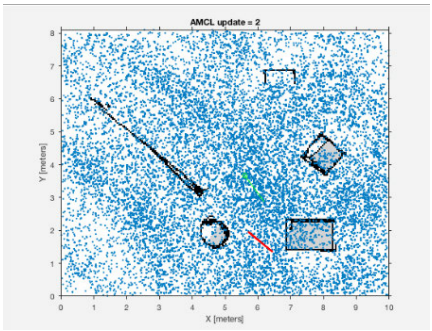


Fig. 2. 2D binary grid

The update occurs according to the algorithm and the current position of the robot is specified (Figure 4). As a result of the algorithm operation, the robot is localized (Figure 5).



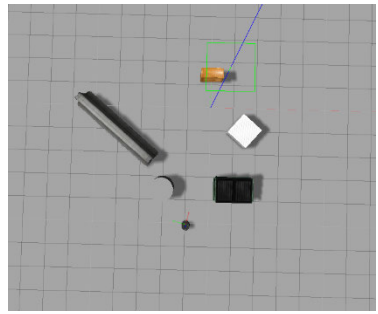
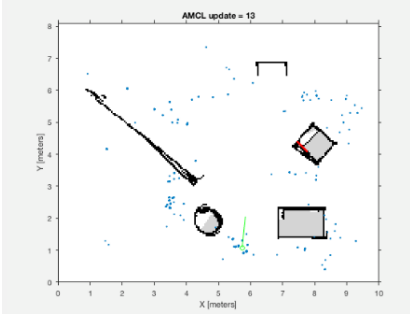


Fig. 3. Start of the algorithm (second update)



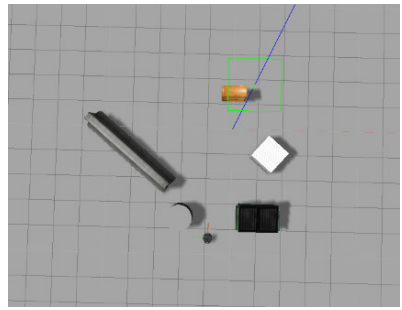


Fig. 4. Algorithm operation (13th update)

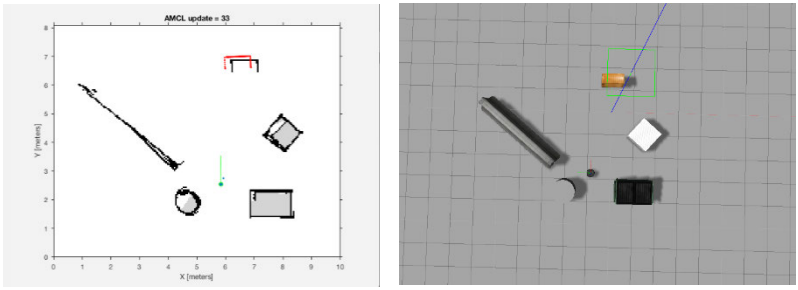


Fig. 5. The end of the algorithm (33th update).

In conclusion, it should be noted that the algorithm has been successfully implemented in practice. It can be used for various types of autonomous objects. Monte Carlo Localization allows you to specify the current position of the robot when it moves and to produce global localization in the complete absence of information about the location of the robot.

The research was carried out with the financial support of the Ministry of Education and Science of the Russian Federation, a unique project identifier: RFMEFI57817X0241.

Литература

1. Introduction to Monte Carlo Localization. [Электронный ресурс]. – URL: http://ais.informatik.uni-freiburg.de/teaching/ws12/practicalB/02-mcl.pdf (дата обращения: 05.11.2018).

2. Robust Monte Carlo Localization for Mobile Robots / Sebastian Thrun, Dieter Fox, Wolfram Burgard, Frank Dellaer. Artificial Intelligence. №128. 2011. С.99-141.