Artif Life Robotics (2006) 10:162–165
DOI 10.1007/s10015-005-0358-4

ORIGINAL ARTICLE

Byoung-Gi Jang · Tae-Yong Choi · Ju-Jang Lee

Adaptive occupancy grid mapping with clusters

Received and accepted: December 6, 2005

Abstract In this article, we describe an algorithm for acquiring occupancy grid maps with mobile robots. The standard occupancy grid mapping developed by Elfes and Moravec in the mid-1980s decomposes the high-dimensional mapping problem into many one-dimensional estimation problems, which are then tackled independently. Because of the independencies between neighboring grid cells, this often generates maps that are inconsistent with the sensor data. To overcome this, we propose a cluster that is a set of cells. The cells in the clusters are tackled dependently with another occupancy grid mapping with an expectation maximization (EM) algorithm. The occupancy grid mapping with an EM algorithm yields more consistent maps, especially in the cluster. As we use the mapping algorithm adaptively with clusters according to the sensor measurements, our mapping algorithm is faster and more accurate than previous mapping algorithms.

Key words Occupancy grid · Mobile robotics · Mapping · Bayes rule · Cluster

1 Introduction

Robotic mapping has been a highly active research area in robotics and AI for a few decades. Robotic mapping addresses the problem of acquiring spatial models of physical environments through mobile robots. There are a number of mapping algorithms. However, the occupancy grid mapping is more popular than others, since it has the reputation of being extremely robust and easy to implement. Mapping through occupancy grid mapping allows the use of various key functions necessary for mobile robot navigation, such as localization, path-planning, collision avoidance, and people-finding.

Occupancy maps have been built using various sensors such as sonar sensor, laser range finders, and stereo vision, etc. However, all these sensors are subject to errors, which are often referred to as measurement noise. In addition, sonar sensors cover an entire cone in space and form a single sonar measurement, and it is impossible to say where in the cone the object is. The sonar sensors are also sensitive to the angle of an object surface relative to the sensor and the reflective properties of the surface. These properties of sensors make a mapping problem difficult and lead to an inconsistent map.

The occupancy grid mapping resolves such problems by generating probabilistic maps. As the name suggests, occupancy grid maps are represented by grids. Namely, they decompose the high-dimensional mapping problem into many one-dimensional estimation problems, which are then tackled independently. Because of the independency of neighboring cells, they often generate maps that are inconsistent with the data, particularly in cluttered environments.

To overcome this, we define a cluster, which is a set of cells. The cluster is the region that has a high probability of being inconsistent with the sensor data when the standard occupancy grid mapping is used. Existing occupancy grid mapping algorithms do the task with the emphasis on individual cells. However, our approach maps with the emphasis on clusters. For making the cluster and choosing the optimal mapping algorithm according to the sensor measurements, maps generated by our approach are more accurate than ones generated by the previous occupancy grid mapping algorithm. Our mapping algorithm is also as fast as the standard occupancy grid mapping algorithm.

2 Standard occupancy grid mapping

The standard occupancy grid mapping approach[1,2] constitutes two main algorithms. First, it decomposes a

B.-G. Jang (✉) · T.-Y. Choi · J.-J. Lee
Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology (KAIST), 373-1 Guseong-dong, Yuseong-gu, Daejeon 305-701, Korea
e-mail: ee185zzang@kaist.ac.kr

This work was presented in part at the 10th International Symposium on Artificial Life and Robotics, Oita, Japan, February 4–6, 2005

multidimensional (typically 2D or 3D) tessellation of space into many independent cells. Second, each cell calculates a probabilistic estimate of its state. To calculate this estimate, techniques such as Bayesian reasoning are then employed on the grid cell level. Each cell is tackled independently.

Let m be the occupancy grid map. The grid cell has the index $\langle x, y\rangle$ to store a probabilistic occupancy, which is $m_{x,y}$. Occupancy grid maps are estimated from sensor measurements. Let $z_1, \ldots, z_T$ denote the measurements from time 1 through to time T. The measurement is composed of a sonar scan and the robot pose at which the measurement was taken. The robot pose, which is assumed to be known, is the xy coordinates of the robot and the heading direction. Each measurement carries information about the occupancy of many grid cells. Thus, the problem addressed by occupancy grid mapping is the problem of determining the probability of occupancy of each grid cell $m_{x,y}$ given the measurements $z_1, \ldots, z_T$.

$$p\left(m_{x,y} \middle| z_1, \ldots, z_T\right) \tag{1}$$

For computational reasons, it is common practice to calculate the *log-odds* instead of estimating the above posterior. The *log-odds* is defined as

$$l_{x,y}^{T} = \log \frac{p\left(m_{x,y} \middle| z_1, \ldots, z_T\right)}{1 - p\left(m_{x,y} \middle| z_1, \ldots, z_T\right)} \tag{2}$$

The assumption in standard occupancy grid mapping is the static world, and conditional independence gives knowledge of each individual grid cell $m_{x,y}$. Two assumptions and the Bayes rule allow us to simplify the posterior to

$$p\left(m_{x,y} \middle| z_1, \ldots, z_t\right) = \frac{p\left(m_{x,y} \middle| z_t\right) p\left(z_t\right) p\left(m_{x,y} \middle| z_1, \ldots, z_{t-1}\right)}{p\left(m_{x,y}\right) p\left(z_t \middle| z_1, \ldots, z_{t-1}\right)} \tag{3}$$

Let $\overline{m}_{x,y}$ be the freeness of the grid cell. The probability of the freeness of the grid cell can be calculated in the same way.

$$p\left(\overline{m}_{x,y} \middle| z_1, \ldots, z_t\right) = \frac{p\left(\overline{m}_{x,y} \middle| z_t\right) p\left(z_t\right) p\left(\overline{m}_{x,y} \middle| z_1, \ldots, z_{t-1}\right)}{p\left(\overline{m}_{x,y}\right) p\left(z_t \middle| z_1, \ldots, z_{t-1}\right)} \tag{4}$$

By dividing Eq. 3 by Eq. 4 and adapting the logarithm, the desired *log-odds* is expressed as

$$l_{x,y}^{t} = \log \frac{p\left(m_{x,y} \middle| z_t\right)}{1 - p\left(m_{x,y} \middle| z_t\right)} + \log \frac{1 - p\left(m_{x,y}\right)}{p\left(m_{x,y}\right)} + l_{x,y}^{t-1} \tag{5}$$

Finally, the desired posterior occupancy probability $p(m_{x,y}|z_1, \ldots, z_T)$ can be recovered from the log-odds representation of the map.

Standard occupancy grid mapping does not take the occupancy of neighboring cells into account. It makes the crucial independence assumption that the occupancy of a cell can be predicted regardless of a cell's neighbors. Herein lies a major problem of the standard occupancy approach. This leads to an incorrect map.

3 Adaptive occupancy grid mapping with clusters

This section presents an algorithm to improve the problems of the previous occupancy grid mapping. A key idea is adapting the cluster, which is a set of cells. The cells in the cluster have a high probability of being inconsistent with the sensor data when the standard occupancy grid mapping is used. Unlike the existing occupancy grid mapping algorithm, our approach does the mapping with the emphasis on the clusters. One cluster does not affect the others, since the clusters are independent of each other. The occupancy of the cells in the cluster is calculated with the occupancy grid mapping proposed by Thrun in 2003.[3] Using the expectation maximization (EM) algorithm, the alternative mapping algorithm solves the mapping problem by maintaining the dependencies between neighboring cells. Hence, it leads to more accurate maps than the standard occupancy grid mapping in the cluster. The clusters are made with neural networks,[4–6] which are powerful tools in pattern recognition.

To make the cluster, we use the neighboring sensor measurements, which are the inputs of neural networks.

$$P = \left[p_1, \ldots, p_R\right] \tag{6}$$

R is the number of sensor measurements used. The output of the neural networks, y, is "1" if the region swept by the sensors is a cluttered or erroneous place. Otherwise, y is "0". That is, if y is "1", we assemble the cells in that region and make a new cluster.

The occupancy of cells out of the cluster is calculated with the standard occupancy grid mapping algorithm explained in Sect. 2. The binary occupancy of cells in the cluster is calculated with the alternative occupancy grid mapping proposed by Thrun.

Let K_i be the number of obstacles in the sensor cone of the i-th measurement. Let $D_t = \{d_{t,1}, \ldots d_{t,Kt}\}$ denote the distances to these obstacles in increasing order. To describe the multiple causes of a sensor measurement z_i, the new variables, called correspondence variables, are defined as

$$c_t = \left\{c_{t,*}, c_{t,0}, c_{t,1}, \ldots, c_{t,K_t}\right\} \tag{7}$$

Each of these variables corresponds to exactly one cause of the measurement z_t. If $c_{t,k}$ is 1 for $1 \le k \le K_t$, the measurement is caused by the k-th obstacle. If $c_{t,0}$ is 1, none of the obstacles were detected and the sensor returns a max-range reading. The random variable $c_{t,*}$ corresponds to the case where a measurement was purely random. The log-likelihood of all data and correspondences is written as

$$\log p\left(Z, C \middle| m\right) = \sum_t \log p\left(z_t,\ c_t \middle| m\right) \tag{8}$$

Here, Z denotes the set of all measurements, and C is the set of all correspondences c_t for all data. Calculating not the probability of the correspondence variables but maximization of the likelihood of the data is important, since the probability of correspondence variables is unobservable. This is achieved by maximizing the expected log-likelihood $E[\log p(Z,C|m)|Z,m]$, where the expectation is taken over the correspondence variables C. The expected log-likelihood can be obtained as follows:

$$
\begin{aligned}
&E\left[\log p\left(Z, C|m\right)|Z, m\right] \\
&\quad = \sum_t \Bigg[E\left[\log p\left(c_t\right)|z_t, m\right] + \log \frac{1}{\sqrt{2\pi\sigma^2}} - \\
&\quad \frac{1}{2}\Bigg[E\left[c_{t,*}|z_t, m\right] \log \frac{z_{\max}^2}{2\pi\sigma^2} + \\
&\quad E\left[c_{t,0}|z_t, m\right] \frac{\left(z_t - z_{\max}\right)^2}{\sigma^2} + \\
&\quad \sum_{k=1}^{K_t} E\left[c_{t,k}|z_t, m\right] \frac{\left(z_t - z_{t,k}\right)^2}{\sigma^2} \Bigg]\Bigg]
\end{aligned}
\tag{9}
$$

Maximizing the above expected log-likelihood is the final goal. To do this, the *expectation maximization algorithm* (EM algorithm) is used. The EM algorithm is one such elaborate technique. The EM algorithm is a general method of finding the maximum-likelihood estimate of the parameters of an underlying distribution from a given data set when the data is incomplete or has missing values.

As above, we choose the optimal mapping algorithm according to the sensor measurements, namely clusters. Hence, maps generated by our approach are faster and more accurate than ones generated by the previous occupancy grid mapping algorithm.

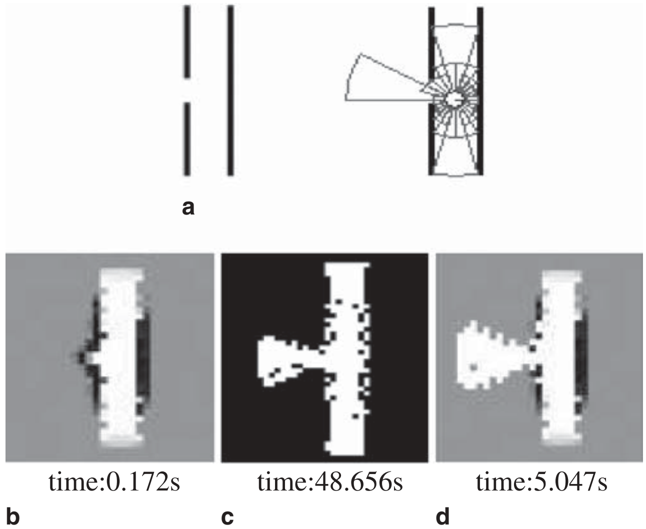

Fig. 1. Narrow open door without error

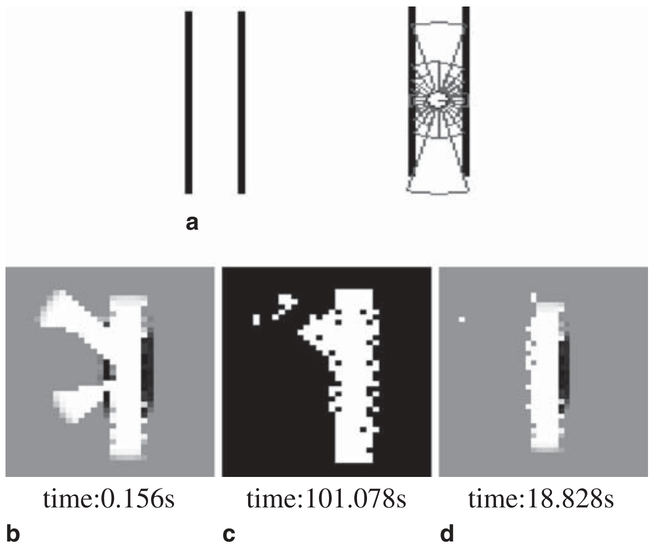

Fig. 2. Corridor with an error

4 Simulation

In order to test our approach, we applied it to learning grid maps using simulated data. Our main findings are that the maps generated by our approach are more accurate, and the approach takes less time, than the previous occupancy grid mapping algorithms such as the standard occupancy grid mapping algorithm and the alternative occupancy grid mapping algorithm with EM.

The sensor measurements are gathered in a corridor while driving by an open door. The mobile robot is equipped with a circular array of 24 sonar sensors. Figure 1a shows a narrow open door as a first example. The width of the door is twice as wide as the width of the mobile robot. Hence, the mobile robot can pass through the door, but it may be difficult to control. Figure 1b shows the result of the standard occupancy grid mapping algorithm. In the standard occupancy grid mapping, a narrow open door is not detected, but other places are similar to Fig. 1a. Fig. 1c is obtained by the alternative occupancy grid mapping with EM. In Fig. 1b, the door is detected, but it takes a long time to calculate. In Fig. 1d, generated by our approach, the door is detected and it takes less time than the occupancy grid mapping with EM. Figure 2 shows the result of a corridor with error measurements. Figure 2a is a simulated environment. Figure 2b shows a map of the standard occupancy grid mapping. The map is incorrect because of the sensor error. In Fig. 2c, the alternative occupancy grid mapping is detected incorrectly in one place, although it is better than Fig. 2b. Unlike Fig. 2b and c, Fig. 2d shows an accurate map. As Fig. 2d is generated by our approach, the map is similar to the environment in Fig. 2a. Because of clusters, our approach is more accurate than the occupancy grid mapping with EM with erroneous places. Our approach also takes less time than the others.

As a result, because our approach maps with the emphasis on clusters, maps generated with the adaptive occupancy grid mapping algorithm are more accurate and faster than the others.

5 Conclusion

In this article, an adaptive occupancy grid mapping algorithm is proposed. Unlike existing occupancy grid mapping algorithms, our approach relies on clusters. The clusters are the region that have the highest probability of being inconsistent with the sensor data. Neural networks are used to make a cluster. According to the cluster, we used the optimal occupancy grid mapping algorithm. As shown in the simulation result, we can map more accurately and faster than the previous occupancy grid mapping.

References

1. Elfes A (1989) Using occupancy grids for mobile robot perception and navigation. IEEE Comput 22:46–57
2. Moravec HP (1989) Sensor fusion in certainty grids for mobile robots. IEEE Comput 22:46–57
3. Thrun S (2003) Learning occupancy grid maps with forward sensor models. Auton Robots 15:111–127
4. Fausett L (1994) Fundamentals of neural netwoks: architectures, algorithms, and application. Prentice-Hall, Englewood chiffs
5. Riedmiller M (1993) A direct adaptive method for faster backpropagation learning: the RPROP algorithm. IEEE Neural Network 1:586–591
6. Kim CH (2002) Feature extration method for a robot map using neural networks. Artif Life Robotics 1:256–258