Abstract- An online map building evolutionary algorithm is proposed using multi-agent mobile robots with odometric uncertainty. The control algorithm for map building in each robot is identical and trained by online evolutionary algorithm (EA). Each robot has configuration uncertainty which increases as it moves, and it perceives the surrounding environment information by the limited range sensors. It communicates with other robots and shares the information. The elementary behaviors are defined and they are used to build a map. EA is applied to the defined behavior set for optimizing the robot actions. To demonstrate the effectiveness of the proposed algorithm, computer simulations are conducted for various environments.

I. Introduction

Map building is a dynamic process which needs the interpretation of data supplied by external sensors [1, 2]. Map building process is the basis for an autonomous mobile robot to execute various works. The map building process has two kinds of problems which affect its performance: the limit of sensor range and the position error of mobile robot. Generally infrared light, ultrasound, or laser is used to perceive the environment information. But these devices can be used in a limited range. The larger range makes the more sparse and inaccurate environment information. The other problem comes from the mobile robot’s dead reckoning error [3]. Without landmarks or beacons, the position and orientation errors increase as robot moves. In the fully unknown environment, it is difficult to set up the beacons or to determine the landmark. To solve these problems we exploit the multi-agent robots which can communicate the environment data. The multi-agent robotics are studied and applied in many fields such as robot soccer, cleaning loom, and delivering objects [4, 5, 6, 7, 8, 9, 10]. By these studies, the effectiveness of multi-agent approach has been verified, which enables to carry out the complicated task with simpler and more inexpensive implementation than single-agent’s case. The multi agent system can be categorized, in terms of implemented structure, as centralized or decentralized. Centralized approach can be thought as a complicated single-agent approach. As the population becomes larger, each single agent becomes increasingly complicated and difficult to control [1]. On the other hand, in decentralized approach each agent behaves according to its own rules. Another category is homogeneous or heterogeneous system. If all the agents behave according to the same rules, the system is said to be homogeneous, otherwise it is heterogeneous.

In this paper we use a decentralized homogeneous multi-agent system. Each robot has the incremental position uncertainty, and the sensor range and communication range are limited. Only if the other robot is within the sensor range, it can take the position and environment information of the other robot. To simplify the uncertainty characteristics a compass is employed in each robot. To build the map, the agent must labor both to explore unvisited area and to reduce the uncertainty of visited area. Also, the communication with other robots is an important point to build the accurate map rapidly, which is a sophisticated problem. We propose an efficient map building algorithm considering these restrictions. First, elementary behaviors such as ‘patrol,’ ‘expand,’ ‘exchange,’ ‘share’ and ‘wait’ are designed and used to build the map. An online evolutionary algorithm is applied to the behavior set to optimize the robot actions [11]. As the control algorithm for building the map in each robot is decentralized and identical, it can be applied to the complex environment without increasing the control complexity. To demonstrate the effectiveness of the proposed scheme, computer simulations are carried out for various environments.

In Section II, the robot and environment models are presented, and the uncertainty of the robot and the map are discussed. In Section III, a new map building evolutionary algorithm is proposed. Section IV presents the evolutionary algorithm to train the behavior set of each robot. Section V includes several simulations to demonstrate the effectiveness of the proposed algorithm. Concluding remarks follow in Section VI.

II. Problem Formulation

A. Robot and its Workspace

A robot is assumed to be omnidirectional and point volume. It has abilities of sensing and communication in the circular range of sensor coverage. Additionally, it has a compass to reduce orientation uncertainty which is further discussed in the next subsection. As the objective of this paper is to make a
map from fully unknown environment, initially all the robots have no information about the environment, other robots, and even its own position.

Let \( p_i^0 \) be the nominal position \((x, y)\) of the robot in a planar workspace \( W \subset \mathbb{R}^2 \) as shown in Figure 1. Superscript \( i \) means the \( i \)th robot with \( i \in \{1, 2, \ldots, k\} \), where \( k \) is the number of robots. \( W_k^i \) is a workspace in the \( i \)th robot’s point of view. The origin of frame \( W_k^i \) is defined as the initial position of the \( i \)th robot. If there is no odometric error in robot motions, the transition matrix between \( W \) and \( W_k^i \) remains a fixed one. However, actually, it floats arbitrarily according to the robot odometric error. \( e^i \) is defined as a circular region covered by the \( i \)th robot’s sensor.

The control input for the \( i \)th robot is defined as a velocity vector \( V^i = [v_x, v_y]^T \) where \( v_x \) and \( v_y \) are the velocities to \( x \) and \( y \) directions, respectively. The control input vector \( V^i \) gives the following trajectory \( p_i^0(t) \) with respect to the frame \( W \):

\[
p_i^0(t) = p_i^0(0) + \int_0^t V^i \, dt. \tag{1}
\]

**B. Robot Uncertainty**

In this paper, odometric uncertainty of the robot is only considered as the uncertainty. Conventionally, the uncertainty dynamics by odometric error can be represented as follows:

\[
\begin{align*}
    x(t + \Delta t) &= x(t) + \hat{v} \Delta t \cos \hat{\theta} \\
    y(t + \Delta t) &= y(t) + \hat{v} \Delta t \sin \hat{\theta} \\
    \phi(t + \Delta t) &= \phi(t) + \hat{\xi}
\end{align*} \tag{2}
\]

where \( \hat{\theta} \) is determined by \( \hat{\theta} = \theta + \phi, \hat{v} \) and \( \hat{\xi} \) are bounded as \(|v - \hat{v}| < \delta v\) and \(|\phi - \hat{\phi}| < \delta \xi\) respectively. The term \( v \) and \( \theta \) are the magnitude and angle of the input vector \( V \), and \( \phi \) is the angle error between the direction of \( X \) axis of \( W \) and that of \( W_k^i \), \( \delta v \) and \( \delta \xi \) are bounded constants of the uncertainty.

By (2), the robot has a complex uncertainty boundary and the configuration uncertainties of \( x, y, \) and \( \phi \) are increasing as time goes on.

By using a compass, we can simplify the equation as follows:

\[
\begin{align*}
    x(t + \Delta t) &= x(t) + \hat{v} \Delta t \cos \hat{\theta} \\
    y(t + \Delta t) &= y(t) + \hat{v} \Delta t \sin \hat{\theta}
\end{align*}
\tag{3}
\]

where \( \hat{v} \) and \( \hat{\theta} \) are bounded as \(|v - \hat{v}| < \delta v\) and \(|\theta - \hat{\theta}| < \delta \theta\) respectively. Figure 2(a) shows the uncertainty region of (3).

Maintaining the validity, we can define a minimal circle covering the uncertain region as in Figure 2(b). Let \( U \) denote a random vector from the nominal velocity vector \( V \) to an arbitrary position within the circular region of radius \( d_w \). Then, we can rewrite the \( i \)th robot position at time \( t \) starting from a nominal position \( p_i^0(0) \) of (1) with uncertainty as follows:

\[
p_i^t(t) = p_i^0(0) + \int_0^t (V^i + U^i) \, dt = p_i^0(t) + \int_0^t U^i \, dt. \tag{4}
\]

Consequently, a region \( S_p^i(t) \) \((p_i(t) \in S_p^i(t))\) where the probability for the robot to exist is nonzero can be determined as follows:

\[
S_p^i(t) = \{p \in \mathbb{R} \mid |p - p_i^0(t)| < D \text{ with } D = d_w t \}. \tag{5}
\]

**C. The Environment and Map Uncertainty**

The environment information which the robot needs to know is the map shown at the standpoint of the robot. The position of a point obstacle \( p_o^i \) in the frame \( W_k^i \) can be derived from (4).

\[
p_o^i(t) = p_o^i(0) + \int_0^t U^i \, dt. \tag{6}
\]

A region \( S_o^i \) where the point obstacle is located is represented as follows.

\[
S_o^i(t) = \{p \mid |p - p_o^i(0)| < D \text{ with } D = d_w t \}. \tag{7}
\]

The above equation shows that the position uncertainty of a point, which is initially zero when it is within the sensor range, becomes circular region of which radius is proportional to the elapsed time after leaving from sensor range.

![Figure 1: A robot and workspace](image)

![Figure 2: Uncertainty region](image)
Using (7), the robot makes sensory data into the map on the frame $W^i_R$ with uncertainty. The Figure 3 shows an example of the process of reconstructing a map. The actual environment is shown at Figure 3(a). Initially, the only region which the robot can detect is the sensory region surrounding the robot (Figure 3(b)). As the robot moves, the area of certain region increases (Figure 3(c) and (d)). Simultaneously the obstacle boundaries are expanded by equation (7). The black area of the figures means the nominal obstacle region, and the gray means the region that has some probability of existence of obstacles. So, the white region only guarantees the obstacle free area.

![Figure 3: An example of constructed map](image)

The objective in this paper is to propose an algorithm to increase the certain region of reconstructed map. To clear up the objective we define some terminology in the following.

1. **Obstacle-free area** $A^i_f$
   The area where there is no obstacle, defined in the absolute frame $W$. The white region of Figure 3(a).

2. **Certainty area** $A^i_c$
   The area in which the probability for obstacles to exist is zero, defined in the frame $W^i_R$. The white region of Figure 3(b), (c), and (d).

3. **Certainty ratio** $\rho^i_c$
   The ratio of certainty area and obstacle-free area defined as follows:
   \[
   \rho^i_c = \frac{\text{size of } A^i_c}{\text{size of } A^i_f}
   \]

   The certainty ratio has a value from zero to one, and is used in Section IV as a cost function to be maximized.

**D. Problem statement**

Map building with uncertainty can now be stated as follows:

Find a searching algorithm of multi-agent mobile robots with uncertainty to expand the certainty area $A^i_c$ of each robot by making the certainty ratio $\rho^i_c$ be maximum for each robot.

**III. Map Building Algorithm**

Up to this point the uncertainty model of robot and environment and the process to determine the certainty area from sensory data are described. In this section we propose a new map building evolutionary algorithm considering the uncertainty. The algorithm is different from the others because of uncertainty minimization. The robot must labor both to explore unvisited area and to reduce the uncertainty of visited area. First we define several elementary behaviors, and then propose the algorithm.

**A. Areas**

Before we describe elementary behaviors, we introduce definitions for two kinds of areas as follows:

1. **Occupied area** $A^i_o$
   The area that $i$th agent occupies. It is obstacle free area. So, if uncertainty region of obstacles invades, its size decreases.

2. **Searched area** $A^i_s$
   The total area that $i$th agent has searched. As in occupied area, if uncertainty region of obstacles invades, its size decreases.

   These are defined in the frame $W^i_R$, and can be overlapped with those of other robots. They have the following relationship with the certainty area $A^i_c$:
   \[
   A^i_o \subset A^i_s \subset A^i_c \quad (8)
   \]

**B. The Elementary Behaviors**

Elementary behaviors are designed to modify $A^i_o$ and $A^i_s$, eventually to expand $A^i_c$. Several behaviors are defined as follows:

1. **Patrol**
   If the robot is in $A^i_o$, move to a nearest boundary point of $A_o$, and then follow the boundary of $A^i_o$. It is used to decrease the uncertainty surrounding the region of $A^i_o$. Figure 4 shows the patrol behavior, where the robot can keep or expand $A^i_s$ patrolling the boundary of $A^i_o$.

![Figure 4: A patrol behavior](image)
2. Expand
   To make \( A_o \) be \( A_i \) as follows:
   \[
   A_o^i \leftarrow A_s
   \]
   It expands the occupied area, and makes the robot explore unvisited region along with patrol behavior.

3. Exchange
   To exchange the certainty area with other robot as follows:
   \[
   A^i_o \leftarrow A^i_o \bigcup T_{ij} A^j_c
   \]
   \[
   A^j_c \leftarrow A^j_c \bigcup T_{ij} A^i_o
   \]
   where \( T_{ij} \) is a transition operator from the \( i \)th robot’s coordinate to the \( j \)th robot’s coordinate. This behavior is executed only when a robot is in the other’s sensor coverage.

4. Share
   To share the occupied area. When two robots are in the other’s sensor range, overlapped area of \( A^i_o \) and \( A^j_c \) is redistributed into two areas to make new area of \( A^i_o \) and \( A^j_c \) have the same size. A simple gradient descent method can be used to get the approximately same size.

5. Wait
   To wait until other robot comes into the sensor range.
   It is an important behavior that makes robots have synchronized motion with other robots.

C. The Map Building Algorithm

Using the elementary behaviors, ‘patrol’ and ‘explore,’ a robot can build the map under some uncertainty, although it can not overcome a certainty ratio. Moreover it has ill adaptability for environment changes. Using communication behaviors, ‘exchange’ and ‘share,’ robots can build a more accurate and large map.

To share the map information, synchronization with neighbor robots is needed. Appropriate synchronization timing is deeply intertwined with the characteristics of the environment. It is difficult to find a fixed algorithm as, by definition of robots, they have no preliminary information about the environment. To solve this problem, online EA is employed, introduced in the next section.

IV. Online Evolutionary Algorithms

The learning algorithm used here is a kind of hill-climbing algorithm as a simple EA. The number of population is one. In each robot the separate and homogeneous EA is processed. As the environment is unknown, only online learning algorithm is meaningful.

The structure of an individual is as shown in Figure 5. The individual makes a robot perform a periodic action. First two terms decide the characteristic of whole gene. The first term determines the patrol direction, and the second decides whether the occupied area is shared or not when it meets other robot. The array of other terms determines periodic actions of the robot. One behavior per time unit is executed. There can be three kinds of behaviors, ‘patrol,’ ‘wait’ and ‘expand.’ After executing all the behaviors, the robot repeats from the first term.

The mutation is executed to the individual to make offspring. Then each robot computes the average certainty ratio after it makes \( n \) rounds of its occupied area. It is used as a cost value of the individual. The values of the individual and its offspring are compared with, and the one with better certainty ratio is selected and mutated. The basic steps for evolution algorithm are as follows:

1. Initialize \( G \).
2. Mutate \( G \) and copy to \( S \).
3. Execute map building with \( G \) for \( n \) rounds and calculate the average certainty ratio.
4. Execute map building with \( S \) for \( n \) rounds and calculate the average certainty ratio.
5. Select a individual with the better certainty ratio and copy it to \( G \).
6. Proceed to step 2 unless available execution time is exhausted

where \( G \) and \( S \) are the individual and its offspring, respectively. The robots execute the evolution algorithm asynchronously.

V. Simulation Results

Computer simulations were carried out to demonstrate the effectiveness of the proposed algorithm. The grid map representation is used to build a map [12]. Resolution of the map is \( 180 \times 100 \) pixels\(^2 \). The size of one pixel is \( 2 \times 2 \) cm\(^2 \). Maximum robot speed is bounded by \( 50 \) cm/sec and robot uncertainty \( d_{\mu} \) is assumed as \( 0.5 \) cm/sec.

The used system is a Pentium processor with 433 MHz clock. Software for implementing the map building algorithm is developed in Visual C++ 6.0 programming language.
Figure 6: Simulation results

(a) The environment

(b) The initial map

(c) The formation of robots

(d) The reconstructed map

Figure 7: Comparison between EA and heuristic algorithm

(a) With EA

(b) Without EA

Figure 8: The change of environment

Figure 6(a) shows the environment and the robots’ initial positions. The points are the robot positions and the circles are the sensor ranges. In the simulation seven robots are used. Figure 6(b) shows a constructed map of a robot, which is the same as the sensor coverage. After 5 minutes of robot operation, the robots are located at the almost same interval as shown in Figure 6(c). Their occupied areas are also distributed into the approximately same size. The constructed map is expanded as shown in Figure 6(d), where the black region is the nominal obstacle area, the gray region the uncertain area. The hatched region is the occupied area. Figures 6(b) and 6(d) represent a certainty region of the robot marked as $R$ in Figures 6(a) and 6(c).

As previously stated, the robots share the occupied area among the others. When the robots patrol their occupied area, the certainty ratio remains in a fixed boundary. These figures show that proposed map building algorithm works very well.

Figure 7 shows the comparison between the algorithm with and without EA. It shows EA plays an important role. These graphs display the change of average certainty ratio. The graph without EA shows large ripples because the information sharing is not achieved harmoniously. Figures 8 and 9 show the adaptation of environment changes. In the right side of Figure 8, obstacles are removed (compare with the Figure 6(a)). The robot map has changed as in Figure 9 which rep-
VI. Conclusions

A new map-building evolutionary algorithm for multi-agent autonomous robots considering the robot uncertainty was developed and simulated. It is a decentralized and homogeneous algorithm. It was obtained by introducing elementary behaviors into the robot action. To obtain the optimal robot action, EA is used. The proposed learning algorithm is online and fully distributed so as to be applied to the real environment. As seen from computer simulations, the proposed method is useful for building a map under the robot uncertainty and for adapting the changing environment.

Bibliography


