Evolutionary Design of Mobile Robot Behaviors for Action-Based Environment Modeling

YAMADA Seiji

CISS, IGSSE, Tokyo Institute of Technology
4259 Nagatsuta, Midori, Yokohama 226-8502, Japan
E-mail: yamada@ymd.dis.titech.ac.jp
URL: http://www.ymd.dis.titech.ac.jp/~yamada/index-e.html

Abstract. This paper describes an evolutionary way to design behaviors of a mobile robot for recognizing environments. We have proposed an action-based approach (called AEM) for a mobile robot to recognize environments. In AEM, a behavior-based mobile robot acts in each environments and action sequences are obtained. The action sequences are transformed into vectors characterizing the environments, and the robot identifies the environments with the vectors. The design of suitable behaviors for AEM is very difficult for human because the search space is huge and intuitive understanding is hard. Thus we develop the evolutionary design of such behaviors using genetic algorithm.

1 Introduction

The most studies to recognize environments have tried to build a precise geometric map using a robot with high-sensitive and global sensors. However, just to recognize environments, such a strict map may be unnecessary. Thus we have tried to build a mobile robot which recognizes environments only with low-sensitive and local sensors, and proposed approach that a mobile robot can recognize the environment with *action sequences*. We call this approach AEM (Action-based Environment Modeling) [5]. In AEM, a mobile robot acts using *given* suitable behaviors like wall-following in environments. Then the action sequences executed in each environment are obtained, and transformed into environment vectors. A robot identifies the environments by comparing them.

Through the research on AEM, we recognized a significant problem: *where the suitable behaviors come from?*. An easy solution is that human designs the behaviors. However the task becomes quite difficult for a human designer as the variety of environments increases. In this paper, we propose the evolutionary design method of such behaviors using GA (Genetic Algorithm) and make experiments for evaluation.

In the similar approach to AEM, several studies have been done in robotics [2] and artificial life [4]. In most researches, wall-following has been used as suitable behaviors in [2][4][5]. Unfortunately the behaviors were described by human designers, and fixed.

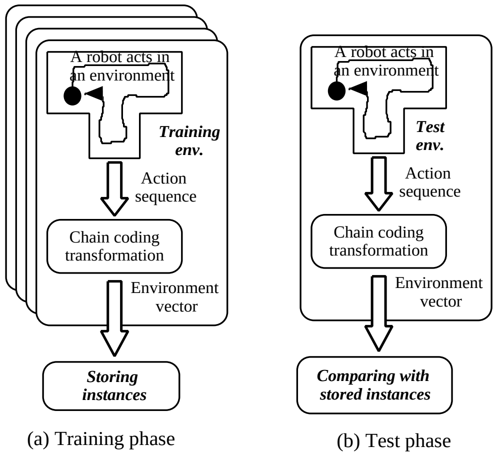

A robot acts in an environment
Training env.
Action sequence
Chain coding transformation
Environment vector
Storing instances
(a) Training phase
A robot acts in an environment
Test env.
Action sequence
Chain coding transformation
Environment vector
Comparing with stored instances
(b) Test phase

Fig. 1. Overview of AEM

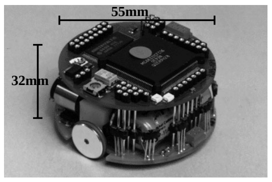


Fig. 2. Khepera

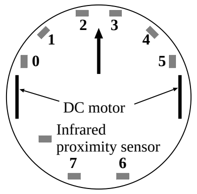


Fig. 3. Sensor positions

2 Task: Action-based Environment Modeling

In AEM [5], a mobile robot is designed in a behavior-based approach [1]. The *behavior* means mapping from states to actions. An AEM procedure consists of two stages: a training phase and a test phase (Fig.1). In the training phase, *training environments* having a *class* are given to a robot. The class means a category in which the environment should be included. The mobile robot acts in the environments using given behaviors and obtains sequences of executed actions (called *action sequence*) for each of them. They are transformed into real-valued vectors (called *environment vectors*) using a chain coding-like method. The environment vectors are stored as *instances*, and the training phase finishes.

In the test phase, a robot is placed in a *test environment* which is one of training environments. The robot tries to identify the test environment with one of training environments. The identification is done using 1-Nearest Neighbor method, i.e. the robot selects the most similar instance to the test environment, and considers that the class of the instance is that of the test environment. The similarity is evaluated with Euclidean distance between environment vectors.

Since the suitable behaviors depend on environment structure which a robot should recognize, they have been described by human designers thus far. However the task is very difficult because of a huge search space.

3 States, actions and environment vectors

Using real mobile robots as individuals in GA is not practical because it is impossible to operate several tens of real robots over more than one hundred generations. Thus we use a simulator of Khepera (Fig.2). As shown in Fig.3, it has two DC motors as actuators and eight Infra-Red proximity sensors which measure both distance to obstacles and light strength.

We describe a state with the range of a sensed value. For reducing the search space of behaviors, we restrict states and actions. A sensor on Khepera returns 10 bit values for distance and light strength. Thus we transform the distance into binary values 0 or 1. The "0" means an obstacle exists within 3cm from a robot, and "1" means it does not exist. Furthermore only three (front, left and right) of eight sensors are used. Next states for light strength are described using only 4 sensors (front, left, right and back). We describe a state using the sensor with the strongest light value and its binary values which mean a light is "near" or "far". A state in which all of the sensors has almost same values is also considered. As a result, the number of states for light is nine, and the total number of states is 72 ($= 2^3 \times 9$). We also describe four actions; *A1*: Go 5mm straight on, *A2*: Turn 30° left, *A3*: Turn 30° right, *A4*: Turn 180° left.

The generated action-sequence is transformed into an environment vector. Let an action-sequence and its environment vector be $[a_1, a_2, \cdots, a_n]$ ($a_i \in \{A1, A2, A3, A4\}$, $a_0 = 0$) and $\boldsymbol{V} = (\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_m)$ ($m \geq n$) respectively. The vector values of $\boldsymbol{V}$ are determined by four rules: If $a_i = A1$ then $\boldsymbol{v}_i = \boldsymbol{v}_{i-1}$, If $a_i = A2$ then $\boldsymbol{v}_i = \boldsymbol{v}_{i-1} + 1$, If $a_i = A3$ then $\boldsymbol{v}_i = \boldsymbol{v}_{i-1} - 1$, If $a_i = A4$ then $\boldsymbol{v}_i = -\boldsymbol{v}_{i-1}$. They change the vector value when the direction of movement changes in the similar way to *chain coding* which is a popular method in pattern recognition.

4 Applying GA to acquire behaviors

GA procedure and coding We use a simple GA procedure and parameters in the followings. We use the coding in which one of actions $\{A1, \cdots, A4\}$ is assigned to each state.

- *Population size*: 50, *Crossover operator*: Uniform crossover.
- *Selection method*: Elite strategy and tournament selection (the size = 2).
- *Crossover rate* P_{cross}: 0.8, *Mutation rate* P_{mut}: 0.05

Defining a fitness function We introduce three conditions for suitable behaviors to AEM: termination of actions, accuracy and efficiency of recognition. The fitness functions for each conditions are defined, and integrated.

A robot has to stop when it returns to the neighborhood of a start point. The termination is evaluated with $g = \frac{\text{(No. of E-trial) + (No. of H-trial)}}{2\times\text{(Total No. of trials)}}$, where E-trial and H-trial means trials in which a robot escaped from the neighborhood of the start point and trials in which it succeeded in returning. Accuracy of identifying environments is also important. It is evaluated with $h = \frac{\text{No. of successful test env.}}{\text{Total No. of test env.}}$, where $h = 0$ when $g \neq 1$. In AEM, the actions should be as small as possible for efficiency. Hence we introduce $k = 1 - \frac{\sum_{i=1}^{n} S_i}{n * S_{max}}$, where S_i is the size of an action sequence in an environment i, S_{max} is the given limited

Table 1. Experimental results in Exp-1

Env.	Train. env.	GN	MaxF
(a)	{emp, L}	1.0 (0)	2.80 (0.106)
(b)	(a) + L2	2.6 (1.84)	2.43 (0.131)
(c)	(b) + iL	2.8 (1.69)	2.44 (0.025)
(d)	(c) + s-emp	2.8 (1.75)	2.44 (0.093)
(e)	10 env.	2.1 (0.738)	2.51 (0.024)
(f)	12 env.	5.2 (3.08)	2.48 (0.07)

Table 2. Experimental results in Exp-2

Env.	Train. env.	GN	MaxF
(g)	emp, 1-la	1.6 (0.966)	2.68 (0.196)
(h)	(g) + 2-la	4.8 (2.78)	2.59 (0.131)
(i)	(h) + 3-la	9.3 (5.19)	2.59 (0.111)
(j)	(i) + 4-la	10.0 (5.94)	2.62 (0.075)

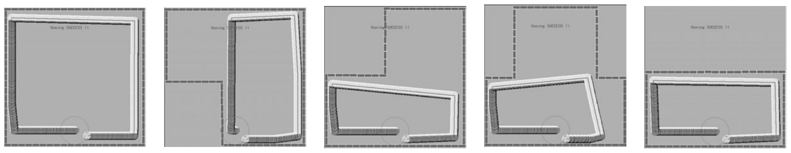
size of an action sequence, and $k = 0$ when $h \neq 1$. We finally integrate three fitness function into $f = g + h + k$ having range [0, 3]. The utility of the fitness function is investigated through successful experiments.

5 Experiments with simulation

We implement a system using a Khepera simulator [3], and make experiments. In all experiments, we give each of training environments to a robot once. The robot acts in the environments, and the environment vectors transformed from the action sequences are stored as instances. Next each of the *training* environment is given to the robot as a *test* environment, and the robot identifies each of the test environments with one of the training environments. In all the experiments, we had 10 trials having different initial population, and investigated the averages and standard deviations of generation number in which GA stopped.

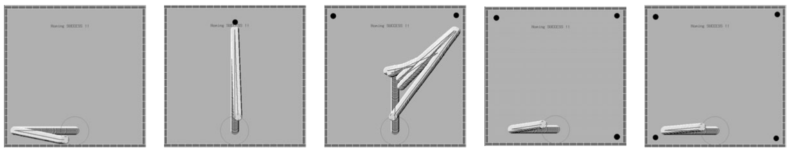
Exp-1: Environments with different contours in shape First we made experiments Exp-1 using environments with different contours in shape. The experimental results are shown in Table 1. Four parts ((a) $\sim$ (d) in Table 1) of five environments: {emp, L, L2, iL, s-emp} were given to a robot. Additionally 10 and 12 different shape environments were used ((e) and (f) in Table 1). The "GN" is the generation number in which GA stopped, and "MaxF" means the maximum fitness value at GN. The numbers in GN and MaxF stand for averages, and the numbers in brackets are standard deviations. This format is common in all the experimental results. Fig.4 indicates the action traces of the best individuals at GN in (d). In such simple environments, the suitable behaviors for AEM were obtained within few generations. The standard deviations was large in GN and small in MaxF, and this tendency was observed through all the experiments. Seeing from Fig.4, different action sequences were obtained depending on the structure of the environments.

Exp-2: Environments with different lights Next, by adding different lights to environments in number and position, we made five environments: {emp, 1-la, 2-la, 3-la, 4-la}. Exp-2 is made by using parts of the environments. Light was so strong that a robot can detect the light direction in any place. The experimental results are shown in Table 2. Fig.5 indicates the action trace of the best individual at GN in (j). In the figures, a black circle stands for a light.



(a) emp (b) L (c) L2 (d) iL (e) s-emp

Fig. 4. Trace of actions in Exp-1 (Five environments)



(a) emp (b) 1-la (c) 2-la (d) 3-la (e) 4-la

Fig. 5. Trace of actions in Exp-2

Though the GN increased more than ones in Exp-1, the suitable behaviors were obtained. Note that we cannot intuitively understand the behaviors in Fig.5. This means that it is very hard for human to design such behaviors by hand-coding and this automatic design method is quite effective.

6 Conclusion

We proposed evolutionary design of suitable behaviors to AEM. GA was applied to search the behaviors, and the simulated mobile robots were used as individuals. States and actions were described for coding chromosomes, and we carefully defined the fitness function. We made experiments using different environments in shape and lights, and found out our approach is promising to learn suitable behaviors for AEM.

References

1. R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Transaction on Robotics and Automation*, 2(1):14–23, 1986.
2. Maja J. Mataric. Integration of representation into goal-driven behavior-based robot. *IEEE Transaction on Robotics and Automation*, 8(3):14–23, 1992.
3. O. Michel. *Khepera Simulator v.2 User Manual*, 1996.
4. U. Nehmzow and T. Smithers. Map-building using self-organizing networks in really useful robots. In *Proceedings of the First International Conference on Simulation of Adaptive Behavior*, pages 152–159, 1991.
5. S. Yamada and M. Murota. Unsupervised learning to recognize environments from behavior sequences in a mobile robot. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 1871–1876, 1998.

This article was processed using the LaTeX macro package with LLNCS style