Learning Intelligent Behavior

Mohamed Salah Hamdi and Karl Kaiser

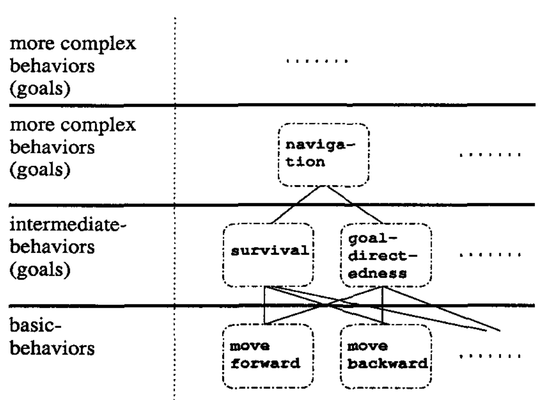
Fachbereich Informatik, Universitaet Hamburg
D-22527 Hamburg, Germany.
e-mail: hamdi/kaiser@informatik.uni-hamburg.de

Abstract. In this paper we present a method for extending the capabilities of a reactive agent using learning. The method relies on the emergence of more global behavior from the interaction of smaller behavioral units. To coordinate behaviors we use a dynamic self-organizing feature map and reinforcement learning. The dynamic self-organizing map is used to partition the space of sequences of situations into different regions. Reinforcement learning refines the content of the regions based on the experience of the agent. We show the effectiveness of the method and evaluate it through several simulation studies.

1 Introduction

The approach we propose for designing autonomous agents, consists of starting with a simple initial control structure and then extending it using learning and adaptability (see figure 1). The extension process consists of three main parts: first, improving the robustness of the system, second, improving the performance of the agent with regard to the individual goals separately, and third, coordinating the individual goals to solve more general and complex tasks. In previous work [1–3] we described methods to achieve the first and second part. This paper deals with the third part. Sections 2 describes the details of the coordination method. Section 3 presents several experiments that evaluate the system. Section 4 draws some conclusions.

Fig. 1. *Organization of the behaviors (goals) in different complexity levels: the lowest level consists of a set of basic-behaviors which define the basic abilities of the agent. The next higher level consists of a set of intermediate-behaviors or goals that can be specified by switching basic-behaviors on and off using priorities. All subsequent levels consist of more and more complex goals.*



2 Synthesizing more global behavior

2.1 The Target Problem

The target problem consists of extending the capabilities of an agent by combining behaviors it is already able to cope with to produce more complex emergent ones. Consider an agent with a set of basic-behaviors $\{B_1, B_2, \ldots, B_m\}$. Suppose that the agent has to deal with the complex task G by coordinating the less complex goals $G_1, G_2, \ldots, G_n$. Suppose also that the agent is already able to solve each individual less complex goal G_i, $i = 1, \ldots, n$, i.e., at each time-step t the agent is able to compute $P_{G_i}(B_j, t)$, $j = 1, \ldots, m$: the priority of each basic-behavior B_j with regard to the goal G_i.

To fulfill the task G, the agent has to compute at each time-step the priorities of the basic-behaviors with regard to G. However, since the agent is able to solve each individual goal G_i, this can be achieved by combining the priorities of the basic-behaviors with regard to the less complex goals:

$$P_G(B_j, t) = \Omega_1(t) P_{G_1}(B_j, t) + \ldots + \Omega_n(t) P_{G_n}(B_j, t) \quad \text{for} \quad j = 1, \ldots, m \tag{1}$$

In equation 1, $\Omega(t) = (\Omega_1(t), \ldots, \Omega_n(t))^T$ with $\Omega_1(t) + \ldots + \Omega_n(t) = 1$, is a vector of coordination parameters varying over time. It describes how, at each time-step, the individual goals G_i contribute to the solution of the task G. The operation "$.^T$" denotes vector transposition.

To deal with the task G, it is necessary to determine for each situation, i.e., a constellation of the priorities $P_{G_i}(B_j, t)$, the appropriate vector $\Omega(t)$. This can be achieved automatically using the learning techniques described below.

2.2 Map-building

By map-building we mean something more like *taking notes* of particular experiences, rather than constructing a geographical map. The aim of map-building is to carry out feature discovery or clustering, i.e., to build a mapping from inputs to statistically salient features of the input population that permit establishing clusters of input patterns with similar features. For this a self-organizing network is used (see [4]). We next briefly review the details of this kind of network and describe the necessary changes we made in order to deal with the coordination task we are trying to solve.

Self-organizing networks. Consider the one-dimensional self-organizing network given in figure 2(A). Each cell i of the self-organizing network has an individual weight vector W_i. Each input vector, I, is fed to all cells. The best matching cell or center of excitation (the cell most close (its weight vector has the smallest distance) to the input vector), i.e., the cell k satisfying the condition:

$$d_k = \|I - W_k\| \leq d_i = \|I - W_i\| \quad \text{for all cells} \quad i \tag{2}$$

is selected. This cell as well as all neighboring cells within a defined neighborhood region are then modified according to the following equation:

$$W_k^{new} = W_k^{old} + \eta(I - W_k^{old}) \tag{3}$$