Performance and Attention in Multi-Agent Tasks *

Yiming Ye

IBM T.J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598, USA
yiming@watson.ibm.com **

Abstract. A well designed cooperation strategy for a task oriented multi-agent team is important as it can improve performance. A challenging research issue in cooperation concerns the extent to which an agent should pay attention to the actions and effects of other agents. In this paper, we address this issue in the context of an object search team. We first propose the concept of an activity window which captures an agent's view of the activities and effects of the team. Then we pinpoint some criteria that can be used to determine whether it is beneficial for an agent to put an action of the team into its window. Finally, we present experimental results to test these criteria.

1 Introduction

An agent is a computational system that inhabits dynamic, unpredictable environments. It has sensors to gather data about the environment and can interpret this data to reflect events in the environment. Furthermore, it can execute motor commands that produce effects in the environment. Multi-agent systems are computational systems in which several autonomous agents interact and work together to perform tasks or satisfy goals. Many researchers are building agents that can work in complex, dynamic multi-agent domains[8]. Such domains include virtual theater[1], realistic virtual training environments [5][6][8], RoboCup robotic and virtual soccer [4], among others.

Coordinating the actions of the agents is very important because an agent that considers the activities of other agents when forming its own plan is usually better able to choose actions that lead to outcomes that it favors. On the one hand, it is obviously not a good strategy for the agents of a cooperative multi-agent team simply to ignore each other, because the intended effects of one

* First International Workshop on Collective Robotics, La Cite des Sciences, Paris, France, July, 4- 5, 1998, Proceedings published by Springer-Verlag in Lecture Notes on Artificial Intelligence volume 1456

** The author would like to thank John Tsotsos, Demetri Terzopoulos, Allan Jepson, Chris Brown, Hector Levesque, and Eugene Fiume for their valuable comments on his Ph.D. thesis, and Eric Harley and the reviewers of CRW for their valuable comments on this paper.

agent's action may already have been achieved by the actions of other agents. On the other hand, it is also not a good strategy for each agent to keep track of all the activities of the other agents, because the effort required might prevent the agent from doing useful work itself. For example, in the robotic soccer domain, ignoring teammates is ill-advised, but so is delaying action until one has complete knowledge of what all the other players are doing. A player that is going to shoot at the goal only needs to know its own surrounding and the situation around the opposing team's goalkeeper; while a goalkeeper only needs to observe the surrounding situation and the actions of the goal shooter in order to save the goal.

These observations motivate examination of how and to what extent an agent should consider the activities of other agents, and what factors are important in deciding local coordination strategy. Sen, Roychowdhury and Arora [7] study the effect of limited local knowledge on group behavior for the resource utilization problem where a number of agents are distributed between several identical resources. They conclude that an agent may benefit more from limited knowledge of the environment rather than complete global knowledge. Hogg, Huberman and Kephart [2] [3] analyze a similar problem and study the effects of local decisions on group behavior. They show that system parameters like decision rate can produce stable equilibria, damped oscillations, persistent oscillations, or chaos. Vidal and Durfee [9] present an algorithm for an agent to determine which of its nested, recursive models of other agents are important to consider when choosing an action.

In this paper, we address the issue of how and to what extent an object search agent should consider the activities of other agents during a multi-agent object search process — the process of searching for a 3D object in a 3D environment by a group of pan, tilt, and zoom cameras (or a group of robots). The goal of the team is to maximize the probability of detecting the target within a given time constraint. Given the real-world nature of the multi-agent object search task, the issues studied in this paper reflect many of the characteristics of other real-world tasks in a dynamic multi-agent environment.

2 The Multi-agent Object Search Team

In this section, we describe some concepts concerning the multi-agent object search system. These concepts are important for further discussions in the rest of the paper. We assume throughout the paper that there are in total m search agents a_1, a_2, ..., a_m available in the team.

The model of the search agent is based on Laser Eye - a pan, tilt, and zoom camera (Fig. 1(a)). The state s_a of a search agent a is uniquely determined by 7 parameters $(x_a, y_a, z_a, w_a, h_a, p_a, t_a)$, where (x_a, y_a, z_a) is the position of the camera center, w_a, h_a are the width and height of the solid viewing angle of the camera, p_a, t_a are the the camera's viewing direction (Figure 1(c)). An operation $\mathbf{f}(a_i, s_{a_i}, r_{a_i}^{(j)})$ for agent a_i entails two steps: (1) take a *perspective* projection image according to state s_{a_i}, and then (2) search the image for the target using

the recognition algorithm $r_{a_i}^{(j)}$. We assume that each agent can have several recognition algorithms that can be used to detect the target; $r_{a_i}^{(j)}$ refers to agent a_i's jth recognition algorithm. The cost $\mathbf{t}(\mathbf{f})$ for an action $\mathbf{f} = \mathbf{f}(a_i, s_{a_i}, r_{a_i}^{(j)})$ gives the total time needed for agent a_i to execute the action. It includes (1) time to manipulate the hardware to the state s_{a_i} specified by $\mathbf{f}$; (2) time to take a picture using the camera on a_i; and (3) time to run the recognition algorithm $r_{a_i}^{(j)}$ specified by $\mathbf{f}$.

To encode the agent's knowledge about the possible target position, the search environment Ω is tessellated into a series of elements c_i: $\Omega = \bigcup_{i=1}^{n} c_i$ and $c_i \bigcap c_j = 0$ for $i \neq j$ (Fig. 1(b)). In addition, we introduce another "cell" c_{out} to refer to the region that is outside the search region Ω. Each cell c is associated with a probability distribution $\mathbf{p}$ for each agent. The term $\mathbf{p}(a_i, c_j, \tau)$ gives the belief of agent a_i regarding the probability that the center of the target is within cell c_i at time τ. The term $\mathbf{p}(c_j, \tau)$ gives the real target probability distribution at time τ. Before the search process, $\mathbf{p}(c_j, \tau)$ and $\mathbf{p}(a_i, c_j, \tau)$ $(1 \leq i \leq m)$ are the same, but they may diverge during the search process.

To calculate the effects of applying a given action by a given agent a_i, we introduce the detection function $\mathbf{b}(a_i, c_j, \mathbf{f})$, which gives the conditional probability that agent a_i will detect the target given that the center of the target is located within cell c_j, and the operation is $\mathbf{f}$. The value of $\mathbf{b}(a_i, c_j, \mathbf{f})$ can be obtained by transformation from a pre-recorded standard detection function for the recognition algorithm used by $\mathbf{f}$ [10]. Obviously,

$$P(\mathbf{f}) = \sum_{j=1}^{n} \mathbf{p}(a_i, c_j, \tau_{\mathbf{f}}) \mathbf{b}(a_i, c_j, \mathbf{f}) \ , \tag{1}$$

gives agent a_i's belief on the probability of detecting the target if $\mathbf{f}$ is applied, where $\tau_{\mathbf{f}}$ is the time just before $\mathbf{f}$ is applied. The actual probability of detecting the target can be calculated by replacing $\mathbf{p}(a_i, c_j, \tau_{\mathbf{f}})$ of the above term with $\mathbf{p}(c_j, \tau_{\mathbf{f}})$.

For any agent a_i, its beliefs on the possible target positions $\mathbf{p}(a_i, c, \tau)$ (for all c) change over time as the multi-agent team perceives the world. Suppose another agent a_j executes an action $\mathbf{f}$, then if agent a_i does not care about the effects of $\mathbf{f}$, its belief will stay unchanged. Otherwise, its belief will be updated according to Bayes law:

$$\mathbf{p}(a_i, c_j, \tau_{\mathbf{f}+}) \leftarrow \frac{\mathbf{p}(a_i, c_j, \tau_{\mathbf{f}}) \left(1 - \mathbf{b}(a_i, c_j, \mathbf{f})\right)}{\sum_{k=1}^{n,\text{out}} \mathbf{p}(a_i, c_k, \tau_{\mathbf{f}}) \left(1 - \mathbf{b}(a_i, c_k, \mathbf{f})\right)} \ , \tag{2}$$

where $j = 1, \ldots, n, \text{out}$.

For a given agent a_i, one of the most important tasks during the search process is to select actions to search for the target. This action selection process is guided by the agent's knowledge $\mathbf{p}(a_i, c_k, \tau)$ — the agent always selects an action $\mathbf{f}$ with the maximum $P(\mathbf{f})$ (Formula (1)). Usually, there are a huge number of actions to be considered, most of which are not necessary. In [10], we develop a method that reduces the many possible actions to a limited set $\Pi[a_i]$ of actions

that must be considered. Thus, during the search process, agent a_i only needs to select the next action from $\Pi[a_i]$. The method is to calculate term (1) for each action in $\Pi[a_i]$. The first action that maximizes term (1) is selected.

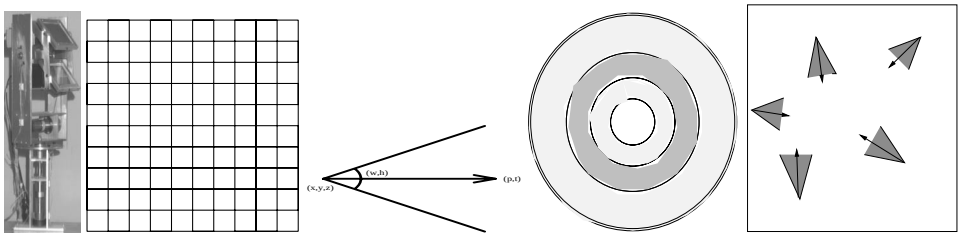

Fig. 1. *(a) The object search agent model: Laser Eye. (b) The tessellation of the environment. (c) The state parameters of an action. (d) The effective range (the dark layer) and the influence range (the dark and the light layers) (e) A scene of an object search team in an environment.*

The performance of the object search team is measured by the probability of detecting the target within a time constraint T by the multi-agent object search team (Fig. (1)(e)). Suppose $\mathbf{F}_{a_i} = \left\{\mathbf{f}_{a_i}^{(1)}, \mathbf{f}_{a_i}^{(2)}, \ldots, \mathbf{f}_{a_i}^{(N_{a_i})}\right\}$ is the set of actions actually selected by agent a_i $(1 \leq i \leq m)$ during the search process, where N_{a_i} is the number of actions selected by agent a_i. Then $\mathbf{F} = \mathbf{F}_{a_1} \bigcup \mathbf{F}_{a_2} \bigcup \ldots \bigcup \mathbf{F}_{a_m}$ is the set of all the actions applied by the team during the search process. The total number of elements in $\mathbf{F}$ is $|\mathbf{F}| = N_{a_1} + \ldots + N_{a_m}$. Let $\Omega(\mathbf{f}_{a_i}^{(j)}) = \{c \mid \mathbf{b}(a_i, c, f_{a_{i_1}}^{(j_1)}) \neq 0\}$ and $\Omega(\mathbf{f}_{a_{i_1}}^{j_1} \cdots \mathbf{f}_{a_{i_r}}^{j_r}) = \Omega(\mathbf{f}_{a_{i_1}}^{j_1}) \bigcap \cdots \bigcap \Omega(\mathbf{f}_{a_{i_r}}^{j_r})$. Let $\sum_{\mathbf{F}(\mathbf{f}_{a_{i_1}}^{(j_1)} \ldots \mathbf{f}_{a_{i_k}}^{(j_k)})}$ be the sum operation over all the different subset $\{\mathbf{f}_{a_{i_1}}^{(j_1)}, \ldots, \mathbf{f}_{a_{i_k}}^{(j_k)}\}$ of $\mathbf{F}$, where $1 \leq k \leq m$. Then the probability of detecting the target $P[\mathbf{F}]$ by the effort allocation $\mathbf{F}$ is calculated by the following formula [11]:

$$
\begin{aligned}
P(\mathbf{F}) = {} & (-1)^{1+1} \sum_{\mathbf{F}(\mathbf{f}_{a_i}^{(j)})} \left(\sum_{c \in \Omega(\mathbf{f}_{a_i}^{(j)})} \mathbf{p}(c, \tau_0) \mathbf{b}(a_i, c, \mathbf{f}_{a_i}^{(j)}) \right) \\
& + (-1)^{2+1} \sum_{\mathbf{F}(\mathbf{f}_{a_{i_1}}^{(j_1)} \mathbf{f}_{a_{i_2}}^{(j_2)})} \left(\sum_{c \in \Omega(\mathbf{f}_{a_{i_1}}^{(j_1)} \mathbf{f}_{a_{i_2}}^{(j_2)})} \mathbf{p}(c, \tau_0) \mathbf{b}(a_{i_1}, c, \mathbf{f}_{a_{i_1}}^{(j_1)}) \mathbf{b}(a_{i_2}, c, \mathbf{f}_{a_{i_2}}^{(j_2)}) \right) \\
& + \\
& \vdots \\
& + (-1)^{r+1} \sum_{\mathbf{F}(\mathbf{f}_{a_{i_1}}^{(j_1)} \ldots \mathbf{f}_{a_{i_r}}^{(j_r)})} \left(\sum_{c \in \Omega(\mathbf{f}_{a_{i_1}}^{(j_1)} \ldots \mathbf{f}_{a_{i_r}}^{(j_r)})} \mathbf{p}(c, \tau_0) \mathbf{b}(a_{i_1}, c, \mathbf{f}_{a_{i_1}}^{(j_1)}) \ldots \mathbf{b}(a_{i_r}, c, \mathbf{f}_{a_{i_r}}^{(j_r)}) \right)
\end{aligned}
$$

$$+$$
$$\vdots$$
$$+(-1)^{|\mathbf{F}|+1}\left(\sum_{c\in\Omega(\mathbf{f}_{a_1}^{(1)}\ldots\mathbf{f}_{a_m}^{(N_{a_m})})}\mathbf{p}(c,\tau_0)\mathbf{b}(a_1,c,\mathbf{f}_{a_1}^{(1)})\ldots\mathbf{b}(a_m,c,\mathbf{f}_{a_m}^{(N_{a_m})})\right) . \quad (3)$$

3 The Activity Window for a Given Agent

When an agent in a cooperative object search team (Fig. 1)(e) executes an action, it will also broadcast the parameters of the action (i.e., the viewing direction and the viewing angle size) to all the other members of the team (the communication time among agents for this purpose is small enough to be ignored). Thus, during the search process, an agent continually gets information on the action execution situations of other agents. The activity window $\mathbf{W}[a_i]$ for agent a_i refers to the views of agent a_i on the activities of the search team. By putting an action $\mathbf{f}$ executed by agent a_j into the activity window of agent a_i (represented as $\mathbf{f} \in \mathbf{W}[a_i]$) refers to the fact that agent a_i updates its target distribution according to Formula (2) when $\mathbf{f}$ is executed by a_j. As we have discussed before, an agent a selects actions based on its knowledge $\mathbf{p}(a, c, \tau)$ ($\forall c \in \Omega$). If it keeps track of every other agent's actions and updates its own knowledge (Formula (2)) accordingly, then its knowledge represents the true target distribution, thus it can select good quality actions during the search process. Otherwise, its knowledge will be different from the true distributions, thus it may not be able to always select good actions during the search process. Although it may seem that it is better for an agent to keep track of all the actions of other agents during the search process, this may not be a good strategy because it takes time to update the agent's knowledge. Thus, it is important for an agent in a multi-agent team to decide the extent to which it should heed the activities of other agents, or in other words, to decide the content of its activity window.

3.1 Factors that may influence team performance

As we have discussed above, good performance of the team depends on finding the proper balance between the benefit and the cost of updating an agent's knowledge based on what other agents are finding. Here, we pinpoint some factors that may influence this balance. Let n_{update} be the number of actions in the activity window of agent a_i which are heeded after the previous action is executed and before the beginning of the action selection process for the next action. Then there are three costs associated with an agent a_i's action $\mathbf{f}$: (A) $t_{select}^{[i]}$, the time needed for the agent to select an action; (B) $n_{update} \times t_{update}^{[i]}$, the total time needed for the agent to update the environment for the n_{update} actions in its activity window, assuming time $t_{update}^{[i]}$ for each; (C) $t_{execute}^{[i]}$, the time needed

for agent a_i to execute an action. If the cost $t^{[i]}_{execute}$ of executing an action is considerably higher than the update time $t^{[i]}_{update}$, then it is worth putting more actions in the activity window. Then more accurate knowledge about the target distribution is used and higher quality actions are selected. However, if $t^{[i]}_{execute}$ is considerably lower than $t^{[i]}_{update}$, then it is not worth spending time to keep track of the activities of other agents — better to devote the time to executing more actions.

Let T be the time used for search; n_i be the total number of actions applied by agent a_i within time T; and n_{ij} be the number of actions of agent a_j that are heeded by agent a_i. Then for a team in which every agent keeps track of all the actions of other agents, the following relations hold:

(A) $n_{ij} \leq n_j$ $(j \neq i)$ and $n_{ii} = n_i - 1$.

(B) $n_i(t^{[i]}_{select} + t^{[i]}_{execute}) + (n_{i1} + \ldots + n_{im})t^{[i]}_{update} \leq T$ (for $1 \leq i \leq m$).

3.2 Determining the contents of the activity window

The important task of deciding which actions should be put in to the activity window is is difficult, because both the benefit, (which is not obvious sometimes), and the cost must be considered. Here, we discuss some criteria that can be used to determine whether it is beneficial to attend to an action of another agent.

Clearly, there may be certain agents in the multi-agent environment whose activities have no effect on set $\Pi[a]$ of potential actions for agent a. These irrelevant activities should not be considered by agent a, because there are no benefits. In the following, we study which agents' activities are irrelevant to agent a. We first explain some concepts. For a given camera angle size $\langle w, h \rangle$, there is an *effective range* corresponding to a spherical layer surrounding the camera (Fig. 1(d)) such that if the target is within this layer, the possibility that it be detected by a correctly directed action with size $\langle w, h \rangle$ is high. The actions in $\Pi[a]$ with size $\langle w, h \rangle$ are partly determined by this layer (refer to [10] for details). There is also an *influence range* corresponding to a larger spherical layer surrounding the camera (Fig. 1(d)), such that if the target is outside this range, it cannot be detected by an action with camera angle size $\langle w, h \rangle$ [10]. The outer radius of the influence range is the *influence radius* and is represented as $\mathbf{R}_a(\langle w, h \rangle)$. The intersection of the viewing volume of a given action $\mathbf{f}$ with the influence range for the angle size $\langle w, h \rangle$ of $\mathbf{f}$ defines the *influence volume* $\Omega(\mathbf{f}) = \{c \mid \mathbf{b}(a, c, \mathbf{f}) \neq 0,$ where $\mathbf{f} \in \Pi[a]\}$ (Section 2). An agent a can have different zoom factors and thus different influence ranges. The smallest angle size that can be achieved by agent a produces the largest influence radius, denoted $\mathbf{R}_a$. If the target is outside $\mathbf{R}_a$, then no matter how a adjusts its state parameters, it will not be able to detect the target.

The following theorem and Properties can be used by agent a to select its activity window during the team search process.

Theorem: *Suppose* $\mathbf{f}$ *is an action applied by agent* a_j *during the search process. If* $\Omega(\mathbf{f}) \bigcap \Omega[a_i] = \emptyset$, *then there is no benefit in putting* $\mathbf{f}$ *in the activity*

window $\mathbf{W}[a_i]$ *of agent* a_i*. In other words, the actions selected by agent* a_i *will not be influenced whether* $\mathbf{f}$ *belongs to* $\mathbf{W}[a_i]$ *or not.*

Proof: Suppose $\mathbf{f}^*$ is the next action selected by agent a_i when $\mathbf{f}$ is not included in $\mathbf{W}[a_i]$. Then we have: $P(\mathbf{f}^*) = max\{P(\mathbf{f}^{'}) \mid \mathbf{f}^{'} \in \Pi[a]\}$ and $\mathbf{f}^*$ is the first such action chosen during a_i's action selection process.

Now suppose that $\mathbf{f}$ is put in $\mathbf{W}[a_i]$. We need to prove that after the probability updating process of a_i, $\mathbf{f}^*$ will also be selected as the next action to execute. Let $\mathbf{p}(a_i, c, \tau)$ (for all $c \in \Omega$) be the target distribution when a_i selects $\mathbf{f}^*$ in the situation that $\mathbf{f}$ is not in $\mathbf{W}[a_i]$; $\mathbf{p}^{'}(a_i, c, \tau)$ be the target distribution after the probability updating process when $\mathbf{f}$ is in $\mathbf{W}[a_i]$; and $P^{'}(\mathbf{f}^{'})$ be the calculated probability of detecting the target by $\mathbf{f}^{'}$ when $\mathbf{f}$ is in $\mathbf{W}[a_i]$. Then we have

$$
\begin{aligned}
P^{'}[\mathbf{f}^*] &= \sum_{j=1}^{n} \mathbf{p}^{'}(a_i, c_j, \tau)\mathbf{b}(a_i, c_j, \mathbf{f}^*) \\
&= \sum_{j=1}^{n} \frac{\mathbf{p}(a_i, c_j, \tau)\,(1 - \mathbf{b}(a_i, c_j, \mathbf{f}))}{\sum_{k=1}^{n,\text{out}} \mathbf{p}(a_i, c_k, \tau)\,(1 - \mathbf{b}(a_i, c_k, \mathbf{f}))}\mathbf{b}(a_i, c_j, \mathbf{f}^*) \\
&= \sum_{c \in \Omega(\mathbf{f})} \frac{\mathbf{p}(a_i, c, \tau)\,(1 - \mathbf{b}(a_i, c, \mathbf{f}))}{1 - P(\mathbf{f})}\mathbf{b}(a_i, c_j, \mathbf{f}^*) \\
&= \sum_{c \in \Omega(\mathbf{f})} \frac{\mathbf{p}(a_i, c, \tau)}{1 - P(\mathbf{f})}\mathbf{b}(a_i, c_j, \mathbf{f}^*) = \frac{P(\mathbf{f}^*)}{1 - P(\mathbf{f})} \geq \frac{P(\mathbf{f}^{'})}{1 - P(\mathbf{f})} = P^{'}(\mathbf{f}^{'}). \qquad (4)
\end{aligned}
$$

Thus, $P^{'}(\mathbf{f}^*) = max\{P^{'}(\mathbf{f}^{'}) \mid \mathbf{f}^{'} \in \Pi[a]\}$. Since the algorithm selects $\mathbf{f}^*$ as the next action to execute when $\mathbf{f}$ is not included in $\mathbf{W}[a_i]$, it will also select the same $\mathbf{f}^*$ as the next action when $\mathbf{f}$ is included in $\mathbf{W}[a_i]$. □

The above theorem is important because it states criteria for determining whether it is beneficial to include an action in an agent's activity window. However, it is not very convenient to use. The following properties give ways of conveniently using the theorem. Let d_{ij} be the distance between agent a_i and agent a_j.

Property A *If* $d_{ij} \geq \mathbf{R}_{a_i} + \mathbf{R}_{a_j}$*, then it is not necessary for agent* a_i *to consider actions executed by agent* a_j*.*

Property B *Let* $\mathbf{f}$ *be an action executed by agent* a_j *with visual angle size* $\langle w, h \rangle$*. If* $d_{ij} \geq \mathbf{R}_{a_i} + \mathbf{R}_{a_j}\langle w, h \rangle$*, then it is not necessary for agent* a_i *to put* $\mathbf{f}$ *into its activity window.*

4 Experiments

A $2D$ simulation of a multi-agent object search system is implemented to test the influence of the activity window on the performance of the multi-agent object search system and to examine various factors that should be considered in deciding the content of the activity window. The system is implemented in C on IBM RISC System/6000. The search environment is a $2D$ square with size

175×175 as shown in Figure 2(a). There is an obstacle in the environment. We assume that each camera agent has only one fixed camera angle size 40°. We also assume that the detection functions for each agent are the same. Figure 2(b) shows the detection function. It is obvious that the radius for any agent is 50. The time needed to select an action by an agent is determined by the size of the search region, the number of the candidate actions to be considered, and the speed of the machine used to run the code. The time needed to update the environment if an agent wants to incorporate an action's effects into its knowledge (that is, to put an action into its activity window) is determined by the size of the environment and the speed of the machine used to run the code. In our experimental setting, the average time needed to select an action is 21" seconds, the average time needed to update the environment is 7" seconds.

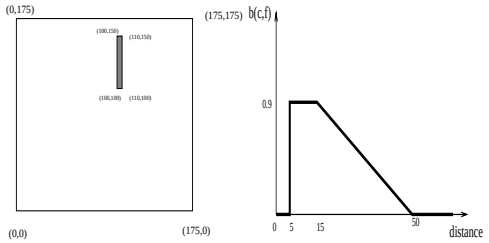

Fig. 2. *(a) The 2D environment with size* 175×175*. There is an obstacle bounded by* $100 \leq x \leq 110$ *and* $100 \leq y \leq 150$ *within the environment. (b) The value of the detection function.*

The first set of experiments tests the influence of the action execution time on the performance of the team when (i) each agent considers all the activities of other agents and (ii) when each agent ignores any activities of other agents. Figure 3(a) gives the scenario of the experiments. The initial outside probability $\mathbf{p}(c_{out})$ is 0.05; the initial probability $\mathbf{p}(c)$ for any element c within the shaded area (bounded by $10 \leq x \leq 165$ and $45 \leq y \leq 55$) is 0.000179; the initial probability $\mathbf{p}(c)$ for any element c other than the shaded area is 0.000026. There are two agents within the environment. Their positions are $(90, 15)$ and $(91, 15)$ respectively. From Figure 3(b), we can see that for action execution times of 1" and 1.5", the strategy of paying attention to the other agent's activities does not perform as well as the strategy of ignoring the activities of the other agent. This situation changes gradually as the actions become more and more expensive, or in other words, as the action execution time becomes longer and longer compared to the action selection and the environment updating time. This illustrates that an agent should consider paying attention to the activities of other agents only when its action execution cost is high.

The second set of experiments test the criteria proposed in Section 3.2. The search environment and the initial target distribution are the same as before. But

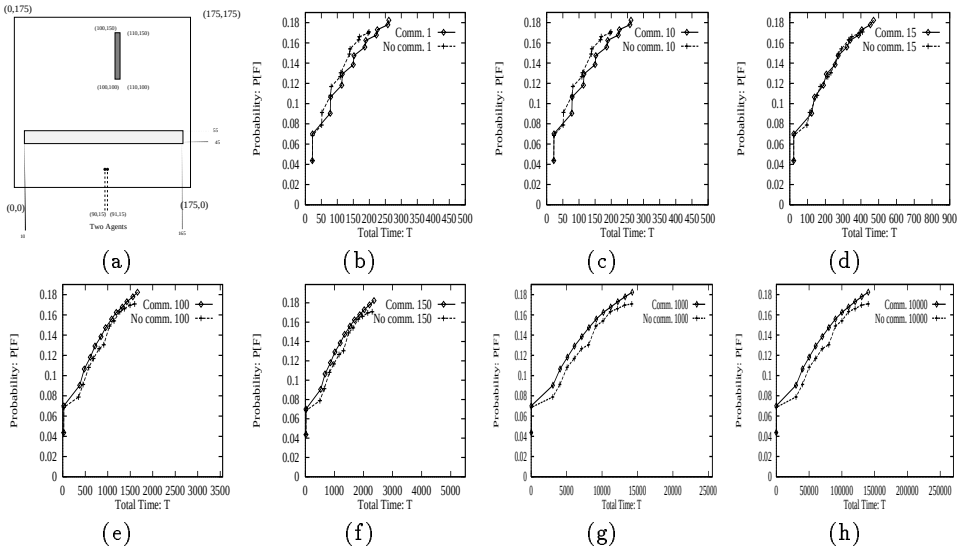

Fig. 3. *(a) The experimental scenario. (b)(c)(d)(e)(f)(g)(h) The probability of detecting the target* $P[\mathbf{F}]$ *versus the time constraint* T *for different execution times of the two agents, where execution times are: (b)* 1", 1.5"*; (c)* 10", 10.5"*; (d)* 15", 15.5"*; (e)* 100", 100.5"*; (f)* 150", 150.5"*; (g)* 1000", 1000.5"*; (h)* 10000", 10000.5"*. In the figure, "Comm." refers to the strategy where each agent attends to the other agent's activities and "No Comm." refers to the strategy where each agent ignores the activities of other agents.*

the positions of the two agents are changed to $(35, 15)$ and $(135, 15)$, respectively. The action execution time is 15" for the agent at $(35, 15)$, and 15.5" for the agent at $(135, 15)$. Since the distance between the two agents is equal to the sum of the influence radius of the two agents, the conditions in Property A hold. Thus, there is no benefit for one agent to keep track of what the other is learning. In this set of experiments, the actions and their sequence is exactly the same for the two strategies (communication and no communication). However, the start execution time for the actions with the same index in the two sequences are different, because in the communication strategy more time must be spent to update the environment. Figure 4(b) gives the delay in the execution starting time as a function of the action index of the team for the strategy of attending to the communications. Figure 4(c) compares the performance of the two strategies. The result is the same as one would predict from the theory in Section 3.2: in this case ignoring the communication between agents is better than attending to it.

The third set of experiments test the benefits of selectively controlling the content of the activity window based on the results in Section 3.2. There are 5 agents in the environment (Figure 5(a)). Two of them are on the left with positions $(45, 15)$ and $(46, 15)$ and action execution times 7.1" and 7.4", respectively.

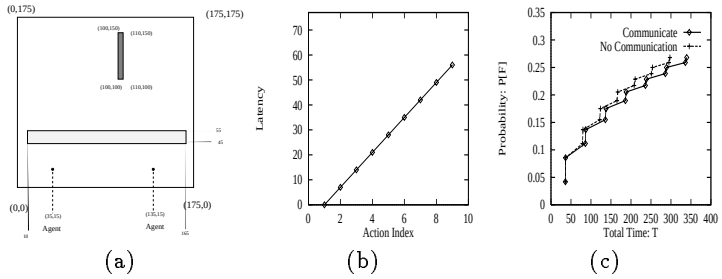

Fig. 4. *Testing the activity window criteria. (a) The environment. (b) The delay in action execution time when each agent attends to the activity of the other agent. (c) Comparison of the performance when each agent attends to the other agent's activities and the performance when they ignore each other.*

Three agents are on the right with positions $(155, 15)$, $(156, 15)$, and $(157, 15)$, and action execution times 7.5", 7.6", and 7.7", respectively. Based on Section 3.2, it is beneficial for agents on the left to attend to each other and for agents on the right to attend to each other, but it is not useful for any agent on the left (right) to listen to any agent on the right (left). The experimental results are shown in Figure 5(b). We can see that the strategy of selectively controlling the content of the activity window based on Section 3.2 is much better than the strategy in which each agent keeps track of all the activities of the team.

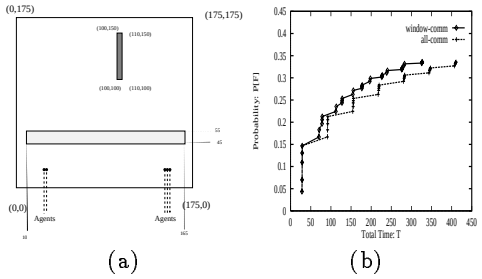

Fig. 5. *(a) The search environment. (b) Comparison of performance when the content of the activity window is selectively controlled based on Section 3.2 (window-comm) and the performance when each agent attends to all the activities of the team (all-comm).*

5 Conclusion

This paper addresses the issue of attending to knowledge acquired by the activities of fellow agents in a multi-agent domain — how much and to what extent should an agent care about what teammates are doing. The concept of activity

window is proposed to represent the view of an agent on the activities and effects of its teammates. We study several factors that may influence performance when selecting the content of an agent's activity window in the context of an object search team, and we propose several criteria in this regard. Experimental results are presented which support our criteria and show the influence of the content of an agent's activity window on the team performance. We believe that some of the analysis presented in this paper can be applied to other multi-agent domains.

References

1. B. Hayes-Roth, L. Brownston, and R. Gen. Multiagent collabration in directed improvisation. In *Proceedings of the International Conference on Multi-Agent Systems (ICMAS-95)*, 1995.
2. T. Hogg and B. Huberman. Controlling chaos in distributed systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6), 1991.
3. J. Kephart, T. Hogg, and B. Huberman. *Dynamics of computational ecosystems: implications for DAI.* Distributed Artificial Intelligence, Volume 2, Research Notes in Artificial Intelligence, Pitman, 1989.