Unsupervised Multi-Agent Exploration of Structured Environments

Dario Maio, Stefano Rizzi

DEIS - Università di Bologna
Viale Risorgimento, 2
40136, Bologna, Italy
dario|stefano@deis64.cineca.it

Abstract

Exploration is a central issue for autonomous agents which must carry out navigation tasks in environments whose description is not known *a priori*. In our approach the environment is described, from a symbolic point of view, by means of a graph; clustering techniques allow for further levels of abstraction to be defined, leading to a multi-layered representation. In this work we propose an unsupervised exploration algorithm in which several agents cooperate to acquire knowledge of the environment at the different abstraction levels; a broadcast model is adopted for inter-agent communication. All agents are structurally equal and pursue the same local exploration strategy; nevertheless, the existence of multiple levels of abstraction in the environment representation allows for the agents' behaviours to differentiate. Agents carry out exploration at different abstraction levels, aimed at reproducing an ideal exploration profile; each agent selects dynamically its exploration level, based on the current demand.

I. Introduction

Autonomous agents are mobile versatile machines capable of interacting coherently with an environment and executing a variety of tasks in unpredictable conditions (Covrigaru & Lindsay 1991) (Nitzan 1995). Autonomy means capability of navigating the environment; navigation, in turn, necessarily relies on a topological and metric description of the environment.

In our work we consider the case where the agent is given no a priori knowledge, so that it must learn the description of the environment on-line by exploring it and interpreting sensor data. However, we assume that some general information concerning the type of environment to be explored is available (*meta-knowledge*). Firstly, we assume that the descriptions of typical sensory patterns present in the environment are given; the selection of patterns corresponding to distinctive or significant categories of objects and places enables recognition of *landmarks* through a sensor-based classification algorithm (Christensen et al. 1994) (Kuipers & Byun 1988) (Phillips et al. 1988). Examples of landmarks are computers and telephones in office environments, medical equipment and receptions in hospital environments. Secondly, we assume that characterization of semantically significant *clusters* of objects or places is possible; agents recognize cluster borders by sensing the passageways between adjacent clusters (Maio & Rizzi 1993a). Example of clusters in office environments are rooms and floors, identified by recognizing doors and stairs respectively.

Based on these assumptions, we have proposed in (Maio & Rizzi 1993b) a multi-layered architecture for representing the environmental knowledge to be used by an autonomous agent for navigation. The concepts and formalisms necessary in the context of this paper are outlined in section II.

The lack of a priori knowledge of the environment gives relevance to the problem of on-line exploration. In order to be ready to carry out navigation tasks as soon as possible, the agents should acquire rapidly a topological and metric description of the whole environment; an agent which knows in deep detail the description of a single room will be less useful, for most tasks, than an agent which knows less about each room but owns a general picture of the disposition of the rooms into departments and floors. Aimed at giving a formal evaluation of an exploration strategy from this point of view, in section III we define an optimal exploration profile.

In (Maio & Rizzi 1993c) we have proposed an algorithm for supervised multi-agent exploration, where the supervisor dynamically assigns each agent the task of exploring the environment at a specific abstraction level, and coordinates the agents assigned to the same level. Since the supervisor owns, at any time, an exact picture of the exploration progress, the supervised architecture allows for an accurate scheduling of the resources. On the other hand, the supervised architecture is based on a point-to-point communication model which is not always feasible and leads however to high communication costs. Besides, since knowledge of the environment is stored in the supervisor, the existence of several agents is not fully exploited to achieve fault tolerance.

In the unsupervised architecture we describe in this work, we adopt a broadcast communication model; fault tolerance is improved by having each piece of knowledge shared by several agents. The exploration script is

presented in section IV, and its performance is discussed in section V.

II. Clusters and layers

Our approach to autonomous agents is based, in the first place, on the distinction between reactive motion control and high-level path planning, which are carried out by relying on sub-symbolic and symbolic knowledge, respectively (Ciaccia, Maio & Rizzi 1991) (Gat 1993). The symbolic representation of the environment is distributed on different abstraction levels (*layers*) in order to supply a richer description of the environment and, at the same time, carry out navigation tasks more efficiently (Maio & Rizzi 1994). In this section we briefly introduce the knowledge representation formalism used in the rest of the paper.

Let $L^{(0)}$ and $R^{(0)}$ be, respectively, the sets of landmarks and feasible inter-landmark paths (*routes*) experienced at a given time. We define as *symbolic layer* the (weakly connected) directed graph $\mathcal{L}^{(0)} = (L^{(0)},R^{(0)})$; each arc is labelled with the *cost* paid when covering the corresponding route.

Given a graph $\mathcal{G}$=(V,A), with V a set of vertices and A a set of arcs, we denominate with *clustering* a partition of the vertices and arcs of $\mathcal{G}$ into a set of clusters and a set of bridges. A *cluster* is a connected sub-graph of $\mathcal{G}$. The *bridge* between two clusters c_i and c_j is the set of the arcs of $\mathcal{G}$ which connect a vertex of c_i to a vertex of c_j. All clusters and bridges are disjointed.

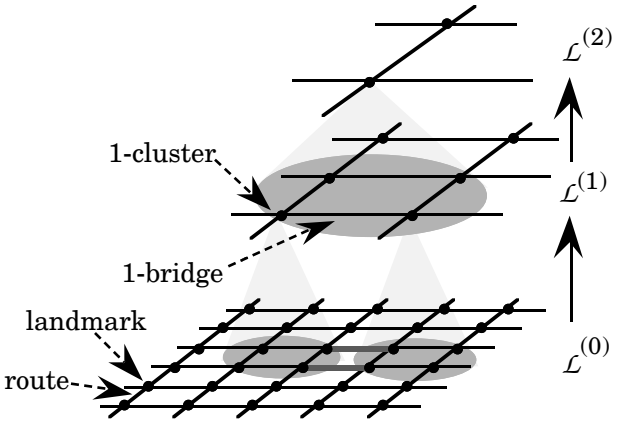

Figure 1. Definition of clustered layers. The two grey circles at the bottom are subgraphs of the symbolic layer, represented in the 1-clustered layer as vertices (1-clusters). The 1-bridge connecting the two 1-clusters corresponds, in $\mathcal{L}^{(0)}$, to the two dark-grey routes.

We call *1-clusters* and *1-bridges* the clusters and bridges determined by clustering the symbolic layer. The directed graph whose vertices and arcs correspond, respectively, to the 1-clusters and the 1-bridges is called the *1-clustered layer* (see Figure 1). A clustered layer may in turn be clustered; in general, we name *k-clustered layer* and denote with $\mathcal{L}^{(k)}$ (k=1,..n, where n is the maximum abstraction level) the graph obtained by applying clustering k times, starting from the symbolic layer. The clusters and bridges of a k-clustered layer are called *k-clusters* and *k-bridges*, respectively. The symbol $c^{(k)}$ denotes a k-cluster; we call *cardinality* of $c^{(k)}$ the number of (k-1)-clusters it contains. We will assume that the n-clustered layer contains exactly one n-cluster $c^{(n)}$.

Figure 2 shows an example of how clustering can be applied to an office environment.

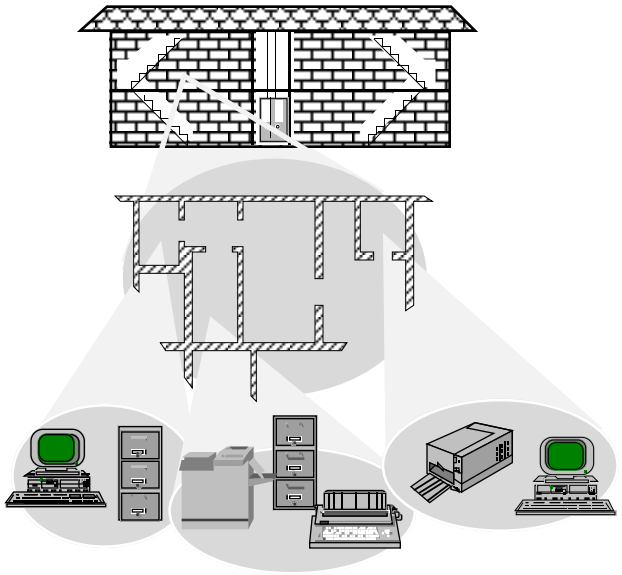

Figure 2. Clustering of an office environment. Landmarks in the symbolic layer correspond to useful objects, and are grouped in 1-clusters corresponding to rooms. Rooms in the 1-clustered layer, in turn, are grouped into floors which form the 2-clustered layer. The 3-clustered layer describes the whole building as a set of floors.

III. An optimal profile for exploration

We call *w-optimal* an exploration strategy whose primary goal is to acquire knowledge of the w-clustered layer $\mathcal{L}^{(w)}$. A w-optimal strategy aims at exploring exhaustively the whole w-clustered layer, without considering the other abstraction levels. The w-clustered layer is completely explored when all the w-bridges have been experienced in both directions, i.e., at least two opposite routes belonging to each w-bridge have been covered.

Let r be the number of routes covered at a given time t. It is possible to estimate the number $r_w^{(k)}$ of k-bridges which should have been experienced at time t when a w-optimal strategy is followed (of course, when an agent takes a route, it also experiences the higher-level bridges which include that route); $r_w^{(k)}$ is proportional to r, and is function of k and w. It is thus possible to define a w-

optimal *exploration profile* as a vector:

$$\mathbf{p}_W = [p_W^{(1)},...p_W^{(n-1)}]$$

where

$$p_W^{(k)} = \frac{r_W^{(k)}}{r}$$

If landmarks and routes are uniformly distributed within the map and clusters are regularly-shaped, the approximate expression of $p_W^{(k)}$ turns out to be:

$$p_W^{(k)} = \begin{cases} \frac{1}{\sqrt{c^k}} & \text{, for k=1,...w} \\ \frac{1}{c^{k-w}\sqrt{c^w}} & \text{, for k=w+1,...n-1} \end{cases} \quad (\text{w=0,...n-1})$$

where c is the average cardinality of clusters. A strategy is w-optimal if, at any time during exploration, the ratio between the number of k-bridges and the number of routes in the knowledge base is equal to $p_W^{(k)}$, for k=1,...n-1.

The derivation of the formula above is omitted here for briefness; it is based on the assumption that the agent's path, seen on the different clustered layers, has very different characteristics. Consider for instance an agent following a 1-optimal strategy in an office: its aim is to discover as quickly as possible new 1-clusters, i.e. rooms. Seen on the symbolic layer, the agent moves in long straight-line paths which traverse the rooms without visiting all their landmarks; seen on the 1-clustered layer, the agent follows a path which, floor by floor, exhaustively visits all the rooms. Hence, if c is the average cardinality of clusters, the agent discovers a new room approximately for each $\sqrt{c}$ landmarks visited (cluster diameter), whereas it discovers a new floor for each c rooms visited.

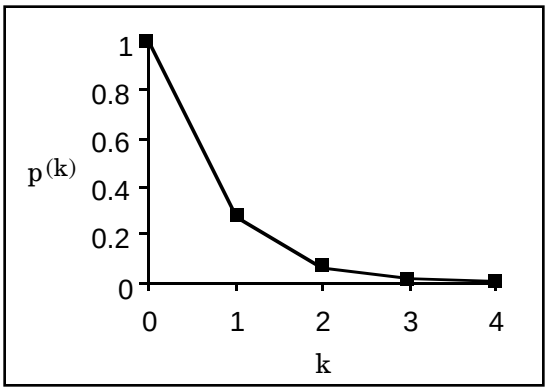

Figure 3. Optimal exploration profile as a function of k (n=5, c=10).

The exploration profile we pursue in this work is the average of n profiles, each aimed at a different abstraction level (see Figure 3):

$$\mathbf{p} = [p^{(1)},...p^{(n-1)}]$$

where

$$p^{(k)} = \frac{1}{n}\sum_{w=0}^{n-1} p_W^{(k)} = \frac{c^{-k/2}}{n}\left(\frac{1-c^{-k/2}}{c^{1/2}-1} + n - k\right) \quad (\text{k=1,...n-1})$$

If f is the average number of k-bridges entering a k-cluster, and c^{n-k} is the number of k-clusters, $c^{n-k}\cdot f$ is an estimate of the total number of k-bridges in the k-clustered layer. As exploration proceeds and the number r of routes experienced increases, the expected number of experienced k-bridges, $p^{(k)}\cdot r$, exceeds $c^{n-k}\cdot f$, meaning that the exploration of the k-clustered layer has been completed. Following the optimal profile this happens first for k=n-1 and then, progressively, for decreasing values of k down to 0.

Let r be the number of routes covered at a given time t; we define *optimal* a strategy if the number $r^{(k)}$ of k-bridges experienced at time t is:

$$r^{(k)} = \min\{ p^{(k)}r, c^{n-k}\cdot f \}$$

IV. The exploration script

In our approach, exploration is carried out at a symbolic level. The link between symbolic exploration and the sensor level is established by assuming that:

- When an agent is located in a landmark, it can determine the directions of the routes departing from that landmark. If a restricted number of paths are physically possible in the environment (for instance, the streets in a city), these can be directly recognized by sensors (for instance, a sonar array); otherwise, assuming that landmarks can be sensed only within a given distance range, the agent will consider as routes all the paths leading to the neighbouring landmarks.
- When an agent is located in a landmark, it can choose to explore one of the routes sensed.
- The agents know the number of levels of clustering at which the environment is to be represented; they can recognize the routes belonging to a k-bridge for any k (for instance, a route traversing a threshold).

All the agents are structurally equal; on the other hand, the hierarchical clustering defined on the symbolic layer enables the agents to diversify their behaviours by assigning themselves to increasing abstraction levels. We will call *k-agent* (k=0,..n-1) one whose task consists in exploring the graph representing the k-clustered layer, i.e., one following a k-strategy.

A. Graph exploration

The k-agents tend to carry out exploration locally, that is, within the (k+1)-cluster they are currently in (*scope*). The graph-exploration algorithm that agents adopt to explore their scope is a variant of Tremaux's algorithm (Rosentiehl 1971).

The classical Tremaux's algorithm carries out exhaustive

exploration of a directed graph by considering local knowledge only; it requires that all arcs are bi-directional and it is optimal, meaning that each arc is visited exactly once.

In our approach, several agents may have the same scope and thus interfere in each other's exploration schedule; moreover, we assume that mono-directional routes may exist in the environment (one-way streets, doors which can be opened one way only, etc.). Hence, an agent following Tremaux's algorithm may occasionally "get lost", i.e., it may reach a vertex and know not which arc to choose next.

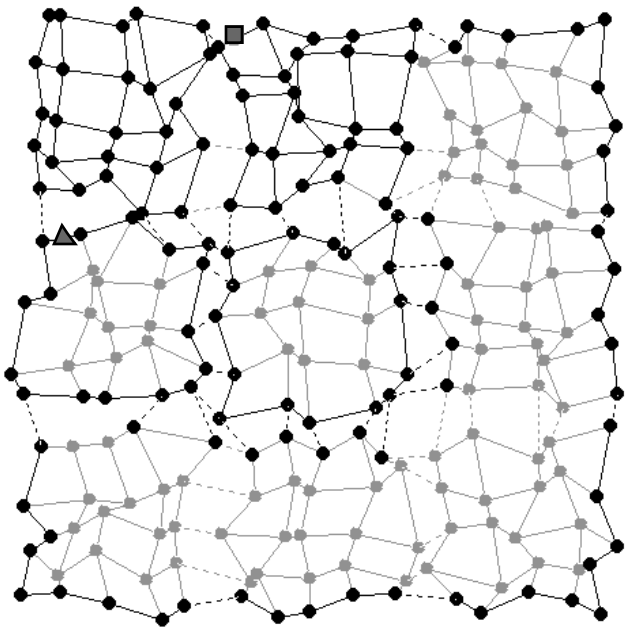

Figure 4. Different exploration paths followed by a 0-agent and a 1-agent. Dashed routes are those belonging to 1-bridges; black routes are those already covered. The square and the triangle show, respectively, the current positions of the 0- and the 1-agent.

From a conceptual point of view, the algorithm adopted by all agents to explore the levels they are assigned to is the same. Nevertheless, while the 0-agents apply the algorithm to landmarks and routes, which are physical entities in the environment, the other agents apply it to clusters and bridges, which are only useful abstractions. In particular, while visibility of the routes departing from a landmark is guaranteed by the sensor level, the same is not true for k-clusters (k>0): for instance, knowing which 1-bridges depart from a 1-cluster entails following the whole edge of the 1-cluster. Figure 4 shows the different exploration paths followed by a 0-agent and a 1-agent on the same map: the 0-agent carries out exhaustive exploration inside 1-clusters; the 1-agent, instead, follows the edges of the 1-clusters and takes the routes contained in the 1-bridges. From a behavioural point of view we might say that, though all agents are equally "curious" (due to their standard exploration strategy), those working on low layers are "meticulous", while those working on high layers are more "superficial".

Our graph-exploration algorithm can be sketched as follows:

Tremaux

```
{  /* the agent has reached vertex v through arc from_a */
   if v has already been visited
   {  find the set of arcs departing from v, A;
      /* at level k>0, this entails following the edge of v */
      if ∃ to_a∈ A: to_a is the opposite of from_a
           and to_a is unknown
         return to_a;
         /* a cycle has been closed: if possible, go back */
      else
         return null;
         /* lost: from_a is one-way, or some other agent has
         experienced its opposite */
   }
   else
      if ∃ to_a∈ A: to_a and its opposite are unknown
         return to_a;    /* if possible, go ahead */
      else
         if ∃ to_a∈ A: to_a is unknown
            return to_a;    /* cul-de-sac: turn back */
         else
            return null;    /* lost */
}
```

B. The agenda

In order to be able to continue exploration when it gets lost, each agent owns a personal *agenda* which is structured in layers corresponding to the different clustered layers: the k-th layer of the agenda reports, for each (k+1)-cluster visited, all the routes belonging to k-bridges that have not been explored yet. The agenda is updated every time the agent reaches an unknown landmark by adding the departing routes; routes are removed from the agenda as they are explored. When a k-agent gets lost, it first consults its agenda *locally*, that is, it looks for an unexplored k-bridge contained in the scope. If no such routes are found, the agenda is consulted *globally*, that is, the agent accepts to change its scope to a different (k+1)-cluster.

C. Communication

The agents communicate by broadcasting messages; a message sent from an agent is received only by the agents who are currently placed within a circular area centred in the sender. We assume that message reception is error-free. Messages are aimed at:

1. Cooperation. Agents situated in the same area should coordinate with each other in order to avoid repeated exploration of clusters and bridges.
2. Knowledge sharing. During exploration, and especially when exploration is over, agents should know as much as possible about the whole environment.

Cooperation is accomplished by encouraging agenda sharing between near agents. Every time an agent discovers a new landmark, it transmits the set of departing routes (message *landmkMSG*); each agent receiving the message puts these routes in its agenda. Every time an agent explores a route, it transmits a message so that the near agents can remove that route from their agenda (message *delMSG*).

When an agent receives any message from another agent, if he has not received other messages from the same agent recently, it broadcasts its whole knowledge base and its agenda (message *knowledgeShareMSG*). Thus, two agents who have not met for some time are enabled to share their knowledge.

D. Level assignment

A k-agent explores the environment at abstraction level k. Each agent dynamically selects its exploration level, aimed at reproducing faithfully the optimal profile. Consider a k-agent who, after getting lost, consults its agenda locally and finds no routes to be explored within its scope. Before consulting its agenda globally, this agent will try to discover if, according to the demand of agents on the different levels, "moving" from level k to another level k' is convenient. This is done by comparing the current profile of exploration,

$$\mathbf{q} = \left[\frac{r^{(1)}}{r},\ldots\frac{r^{(n-1)}}{r}\right]$$

where r and $r^{(k)}$ (k=1,...n-1) are, respectively, the number of routes and of k-bridges (k=1,...n-1) experienced, with the optimal profile, **p**. The level k' to which the agent should be moved is the one whose exploration is behind schedule to the greatest extent.

E. The script

In this section the agent's exploration script is sketched. Variable k is used to store the current abstraction level of the agent. Variable *mode* has value "explore" if the agent is following Tremaux's algorithm, value "goTo" if the agent has successfully consulted its agenda and is following the shortest path to an unknown route.

```
explore()
{  choose my initial assignment level, k;
   set mode to "explore";
   set scope to (k+1)-cluster where I am;
   let v be the landmark where I am;
   put v in knowledgeBase;
   let R be the set of the routes departing from v;
   put the routes in R in agenda;
   while agenda is not completely empty do
      handle event landmark(v, from_r)
      and message reception;
}
```

Event handling:

```
when landmark(v, from_r) do
/* reached landmark v through route from_r */
{  if mode=="goTo"
      if the goal has been reached
      {  set mode to "explore";
         set scope to (k+1)-cluster where I am;
      }
      else
         take the next route in the planned path;
   if mode=="explore"
   {  let R be the set of the routes departing from v;
      /* since mode is "explore", from_r is certainly
        unknown */
      put from_r in knowledgeBase;
      if v is unknown
      {  put v in knowledgeBase;
         put the routes in R in agenda;
         broadcast: landmkMSG(v,R);
      }
      /* determine a route to_r for leaving v */
      try to apply Tremaux's algorithm;
      if lost
      {  consult agenda locally;
         if agenda is locally empty
            if agenda is not completely empty
            {  find the level where agents are most needed, k';
               if k'==k
               /* changing level is not convenient */
                  consult agenda globally;
               else
               /* changing level is convenient */
               {  k=k';   /* become a k'-agent */
                  set scope to (k+1)-cluster where I am;
                  at the new level,
                  try to apply Tremaux's algorithm;
                  if lost
                  {  consult agenda locally;
                     if agenda is locally empty
                        consult agenda globally;
                  }
               }
            }
      }
      if route to_r has been determined
      /* leave v through route to_r */
      {  delete to_r from agenda;
         broadcast: delMSG(to_r,k);
         take route to_r;
      }
   }
}
```

By denoting with agenda[k] the k-th level of the agenda, the procedures for local and global consultation of the agenda can be summarized, respectively, as follows:

localConsult
{ /* *within* scope, *find the nearest k-cluster whose k-bridges have not been completely explored* */
goal=nearest{c: (∃r∈ agenda[k]: c is the starting k-cluster for r and r is within scope)};
plan the shortest path to goal;
set mode to "goTo";
return first route in the planned path;
}

globalConsult
{ /* *find the nearest k-cluster whose k-bridges have not been completely explored* */
goal=nearest{c: (∃r∈ agenda[k]: c is the starting k-cluster for r)};
plan the shortest path to goal;
set mode to "goTo";
return first route in the planned path;
}

V. Performance evaluation

In this section we discuss the performance of our algorithm from three points of view: adherence to the optimal exploration profile, efficiency, fault tolerance.

The deviation of the exploration profile from the optimal one can be evaluated at time t, when r route have been covered, as

$$\text{profileDeviation}(t) = \frac{1}{n-1} \sum_{k=1}^{n-1} \frac{|r'^{(k)}(t) - r^{(k)}|}{r^{(k)}}$$

where $r'^{(k)}(t)$ is the number of k-bridges actually experienced at time t and $r^{(k)}$ is the optimal number of k-bridges calculated in function of r as shown in section III.

Efficiency is evaluated by comparing the cost of exploration per agent with the ideal cost which would be paid from each agent if no route had been taken more than once:

$$\text{costPerAgent} = \frac{\text{totalCost}}{g}, \quad \text{idealCostPerAgent} = \frac{\text{idealCost}}{g}$$

where g is the number of agents and idealCost is the sum of the costs of all the routes in the environment.

Fault tolerance is calculated as the average percentage degree of knowledge sharing:

$$\text{faultTolerance}(t) = \frac{1}{g} \cdot \frac{1}{r} \sum_{i=1}^{g} i \cdot h(i) \le 1$$

where r is the total number of known routes and h(i) is the number of routes known to i agents at time t. If faultTolerance(t) is 1, then all agents share all the knowledge.

Figure 5 shows, for a sample map, how the average profile deviation, the efficiency and the fault tolerance depend on the number of agents. It appears that profile deviation is always contained within 15%, and fault tolerance is more than 90%. Efficiency is lower when several agents are employed, since they tend to interfere with each other.

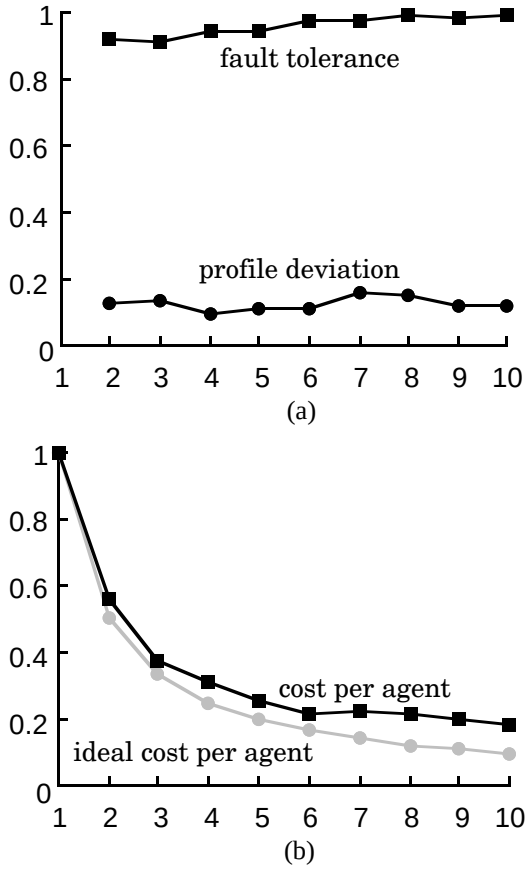

Figure 5. (a) Profile deviation and fault tolerance, both averaged on the whole exploration, and (b) efficiency in function of the number of agents. The sample map employed has more than 500 landmarks and 1400 routes, and has 4 clustering levels. The transmission range is 10% of the total map diameter.

VI. Conclusion

In this paper we have presented an algorithm for unsupervised multi-agent exploration of unknown environments. According to the current necessity, each agent dynamically selects a specific abstraction level for exploring the environment; coordination with the other agents and knowledge sharing are accomplished by message broadcasting.

References

Ciaccia, P.; Maio, D.; and Rizzi, S. 1991. Integrating knowledge-based systems and neural networks for navigational tasks. In Proc. IEEE COMPEURO, 652-656.

Bologna, Italy.

Christensen, H.I.; Kirkeby, N.O.; Kristensen, S.; Knudsen, L.; and Granum, E. 1994. Model-driven vision for in-door navigation. *Robotics and Autonomous Systems* 12:199-207.

Covrigaru, A.A.; and Lindsay, R.K. 1991. Deterministic autonomous systems. *AI Magazine* 12(3):110-117.

Gat, E. 1993. On the role of stored internal state in the control of autonomous mobile robots. *AI Magazine* 14(1):64-73.

Kuipers, B.J.; and Byun, Y.T. 1988. A robust, qualitative method for robot spatial learning. In Proc. AAAI88 2:774-779. Saint Paul, Minnesota.

Maio, D.; and Rizzi, S. 1993a. Map Learning and Clustering in Autonomous Systems. *IEEE Trans. Patt. Anal. Machine Intell.* 15(12):1286-1297.

Maio, D.; and Rizzi, S. 1993b. Knowledge architecture for environment representation in autonomous agents. In Proc. Proc. Eighth Int. Symposium on Computer and Information Sciences:4-11. Istanbul, Turkey.

Maio, D.; and Rizzi, S. 1993c. Supervised multi-agent exploration of unknown environments. In Proc. 3rd Workshop D-AI*IA 93:90-99. Rome, Italy.

Maio, D.; and Rizzi, S. 1994. A hybrid approach to path planning in autonomous agents. In Proc. 2nd Int. Conf. on Expert Systems for Development:222-227. Bangkok, Thailand.

Nitzan, D. 1985. Development of intelligent robots: achievements and issues. *IEEE Journal of Robotics and Automation* RA-1(1):3-13.

Phillips, W.A.; Hancock, P.J.B.; Willson, N.J.; and Smith, L.S. 1988. On the acquisition of object concepts from sensory data. In Eckmiller, R.; and Malsburg, Ch.v.d. eds. *Neural Computers* F41:159-168. NATO ASI Series:Springer-Verlag.

Rosentiehl, P. 1971. Labyrinthologie mathématique. *Mathématiques et Sciences humaines* 33:5-32.