Touzet, C. F. / Distributed Lazy Q-learning for Cooperative Mobile Robots, pp. 5-13, Inernational Journal of Advanced Robotic Systems, Volume 1 Number 1 (2004), ISSN 1729-8806

Distributed Lazy Q-learning for Cooperative Mobile Robots

Claude F. Touzet
University of Provence, Integrative and Adaptative Neurobiology Laboratory (UMR 6149)
Case 362, 52, St Jerome Faculty, F-13397 Marseille Cedex 20, France
Claude.Touzet@up.univ-mrs.fr

Abstract: *Compared to single robot learning, cooperative learning adds the challenge of a much larger search space (combined individual search spaces), awareness of other team members, and also the synthesis of the individual behaviors with respect to the task given to the group. Over the years, reinforcement learning has emerged as the main learning approach in autonomous robotics, and lazy learning has become the leading bias, allowing the reduction of the time required by an experiment to the time needed to test the learned behavior performance. These two approaches have been combined together in what is now called lazy Q-learning, a very efficient single robot learning paradigm. We propose a derivation of this learning to team of robots : the «pessimistic» algorithm able to compute for each team member a lower bound of the utility of executing an action in a given situation. We use the cooperative multi-robot observation of multiple moving targets (CMOMMT) application as an illustrative example, and study the efficiency of the Pessimistic Algorithm in its task of inducing learning of cooperation.*
Keywords: *Cooperative robotics, cooperative learning, CMOMMT, Q-learning, lazy learning, reinforcement learning, heterogeneous robots*

1. Introduction

Learning in cooperative robotics is a recent research area. Its promises are beguiling: a way to program a set of robots without having to explicitly model their interactions with the world - including the other team members - to achieve cooperation. Despite attracting lot of interest from researchers involved in cooperative robotics, we must recognize that, today, results are scarce (Cao Y. *et al*, 1997). In particular, it is our opinion that learning techniques should be as automatic as possible and therefore avoid as much as possible the involvement of the human operator, e.g., the description of a model of the interactions between the robots and the world. To achieve this goal, a sub-symbolic approach is mandatory. Among the possible sub-symbolic paradigms, reinforcement learning (Watkins C., 1989) (Sutton R. & Barto A., 1998) is certainly the most used in autonomous robotics today. It offers the automatic building of the representative learning base of examples using a measure of the performance of the desired behavior (Kaelbling L. *et al*., 1996), (Dorigo M., 1996), an important improvement compared to the supervised learning approach (see for example Heemskerk J. *et al.*, 1996). Reinforcement learning will be our paradigm of choice. In the cooperative robotics context, the reinforcement function measures the performance of the whole team of robots.

Cooperative robot learning raises, at least, all the issues of robot learning, plus a number of new ones: (1) usually a much larger search space (combined individual search space), (2) the need for communication or at least awareness of other team members, and also (3) the synthesis of the individual behaviors with respect to the task given to the group.

(1) Robots are real artifacts using sensors and actuators to deal with the real world. The sensor and actuator specifications allow to approximate a computational measure of the search space size. If d is the number of sensors, p the discreteness of a sensor measure, and we assume that all sensors share the same p, then the search space size is equal to $p^d * \#actions$.

(2) Cooperation usually implies that the robots must be aware of other members of the team. Communication among the team members is, by far, the most commonly assumed mechanism and involves an emitter, a receiver, a message, etc. On the other hand, awareness of other robots in the team (Touzet C., 2000) is a simpler, but sufficient,

mechanism to allow cooperation in cooperative multi-robot observation of multiple moving targets (CMOMMT) applications.

(3) In a multi-robot system, the individual behavior of a robot can interact in complex and intricate ways with the task assigned to the team. In such systems, it is generally assumed that a robot must take into account the behaviors of the other team members to demonstrate an effective action selection. Since we assume a very simple cooperation mechanism (i.e., awareness) and we want to explicitly avoid modeling, our solution to guarantee team coordination is by distributing the only available group information - the reinforcement function value - to each individual robot.

The CMOMMT domain is representative of an interesting class of applications - even if seldom tackled. Because of the limited field of view of each robot, and the poverty of the information offered by the awareness process (instead , null and negative rewards. The Pessimistic Algorithm presented in section 5 allows the distribution of the global reward to each of the team members. Beginning in section 6, we use CMOMMT as an illustrative application to report on the performance of the Pessimistic Algorithm in cooperative robot learning with a homogeneous team of robots. We review related work in section 7. Finally, we summarize and offer concluding remarks.

2. Lazy learning and Q-learning

Cooperative robot learning can be defined as «the automatic modification of the team robot behaviors to improve their performance in their environment.» Cooperative learning presents all the constraints associated with individual robot learning, plus several specific ones. In particular, the search space size is increased by the necessity to involve the other team members in the situation. On the other hand, the number of situations that can be explored during a given time does not change at all (compared to a single robot) due to the mechanical nature of the actuators. Therefore, the ratio of number of explored situations versus the size of the search space is even smaller and the time needed to complete the exploration phase in reinforcement learning becomes impractical. Generalization - already a necessary component of robot learning (Touzet C., 1997) - is not able to account for such drastic ratio variations in the number of samples over the search space size: the number of samples needed to estimate a function of several variables to a given level of accuracy grows exponentially (in the worst case) with the number of variables. Incorporation of initial knowledge may help reduce the exploration time required by pointing to specific search space regions, but it constitutes an *ad-hoc* solution.

Lazy learning (Aha D., 1997), also called instance-based learning, promotes the principle of delaying the use of the gathered information until the necessity arises (Fig. 1). of communication), it is impossible to predict the acquisition of new (unseen) targets. If we define the reinforcement function as positive if one or several targets have been acquired by the group, and negative if one or several targets have been lost, then an increase in system performance can only be sought by reducing negative rewards. To this end, we propose a «pessimistic» algorithm able to compute for each team member a lower bound of the utility of executing an action in a given situation. Two learning paradigms have been blended together to this end: reinforcement learning and lazy learning.

In the following section (section 2), we review the specifics of cooperative robot learning and the related advantages and limitations of reinforcement learning and lazy learning. In section 3, we describe the CMOMMT application, and in section 4 we report analytical and experimental results on the possibilities of predicting positive

The same pool of information (*i.e.*, memory) is used for different behavior synthesis. The lazy memory could be considered as a way to reduce the duration of any robotic learning application. In the context of reinforcement learning, lazy learning provides instantaneously (after the initial and unique sampling phase) a set of situation-action pairs that can be considered as the result of a random exploration phase. Lazy learning samples the
situation-action space, storing the succession of events in memory (1) and, when needed, probes/searches the memory for the best move (2).

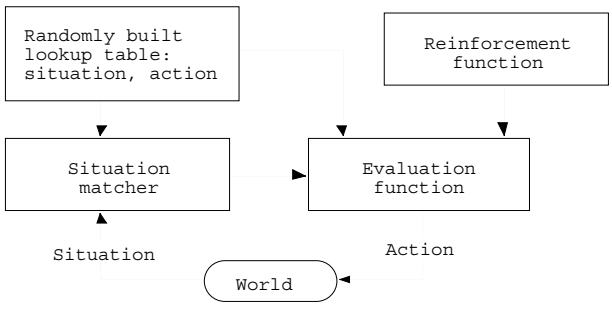

Fig. 1. Lazy learning: Randomly sampled situation-action pairs in the lookup table are used by the situation matcher to select the action to execute in the current situation. The reinforcement function qualifies the actions proposed, helping to select the best one.

(1) The sampling process stores the successive situation-action pairs generated by a random action selection policy. The exploration phase is done only once, stored and used later by all future experiments. By storing situation-action pairs, a lazy memory builds a model of the situation transition function. Two questions immediately arise about the legitimacy of considering lazy learning as a model, and if so, about the quality of the model:

- Lazy learning assumes that the environment is not changing. Any change in the environment diminishes the quality of the lazy memory. However, basic features, such as the effects of moving forward or backward in front of an obstacle, tend to persist despite environment variations and are, in any case, important usable knowledge.

- It has been demonstrated (Whitehead S. *et al.*, 1993) that random exploration might be dangerous and in some environments is an immensely ineffective method of gathering data, requiring exponentially more data than a system that interleaves experience gathering with policy-building more tightly. However, as the author remarks these results only apply to the «go to a particular location» type of applications. They do not generalize to more «reactive» behaviors like obstacle avoidance or target observation.

(2) In order to express a behavior, the memory must be searched. (Sheppard J. & Salzberg S., 1997) propose to search the memory with the reinforcement function. Their objective is to provide a method for predicting the rewards for some state-action pairs without explicitly experiencing them. They call their algorithm lazy Q-learning. For the current real world situation, a situation matcher locates all the states in the memory that are within a given distance. If the situation matcher has failed to find any nearby situations, the action comparator selects an action at random. Otherwise, the action comparator examines the expected rewards associated with each of these situations and selects the action with the highest expected reward. This action is then executed, resulting in a new situation. There is a fixed probability (0.3) of generating a random action regardless of the outcome of the situation matcher. New situation-action pairs are added to the memory, along with a Q-value computed in the classical way. Among similar situation-action pairs in the memory, an update of the stored Q-values is made.

There is a limit to the usefulness of this lazy memory because the Q-values associated with the situation-action pairs only apply for a particular behavior. With the desire of reducing as much as possible the learning time and also of preserving the usefulness of the lazy memory, we modified the algorithm in the following way: the situation matcher always proposes a nearest situations set and there is no random selection of actions by the action comparator. Also, the Q-values are not stored with the situation-action pairs, but are computed dynamically as the need arises.
In a lazy Q-learning experiment, the exploration phase is only done once, stored and used later by all future experiments. Therefore, an experiment only requires a test phase: a measure of the performance of the learning. The learning phase corresponds to the probing of the memory and involves lots of computations. However, the computation time requirements are negligible compared to the robot mechanical time requirements: an experiment with lazy learning and a group of robots can be shorter (effective time) than an experiment involving just one robot and eager learning.

3. The CMOMMT Application

We choose to illustrate our work with the cooperative multi-robot observation of multiple moving targets (or CMOMMT for short) problem (Parker L. E. & Touzet C., 2000) (Parker L. E. *et al.*, 2002). In a bounded arena (Fig. 2), a team of robots with 360° field of view of limited range has to maximize the observation time of a set of targets moving randomly (5% probability of change of direction, maximum speed less than the maximum robot speed).

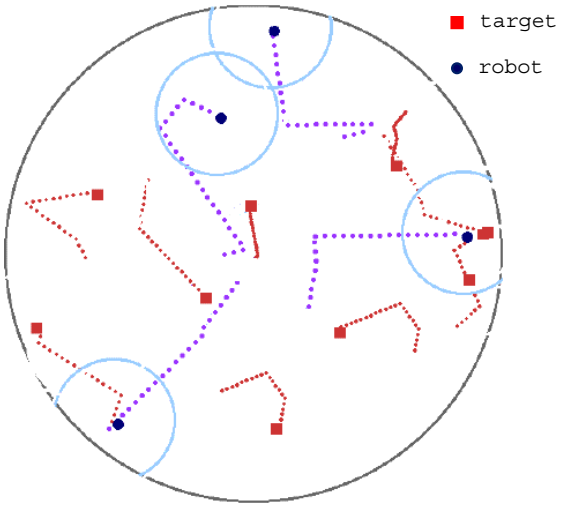

Fig. 2. Bounded arena, with 4 robots moving randomly (circle). The radius of the arena is 5, the radius of the sensory perception range of the robots is 1. There are 9 targets (square), also randomly moving. The dotted lines indicate the paths followed by the robots and the targets. The targets have different speeds: the closer the dots, the lower the speed.

We say that a robot is monitoring a target when the target is within the robot's sensory field of view. The objective is to maximize the collective time during which targets are being monitored by at least one robot. The radius of the sensory robot range is less than the size of the arena, implying that robots have to move to maintain observational contact.
Each robot situation is a vector of two times 16 components. The dimensionality of the search space is thus 32. The first 16 components code the position and orientation of the targets. It simulates a ring of 16 sensors uniformly distributed around the robot body. Each sensor measures the distance to the nearest target. The sensor position around the body gives the orientation. The second ring of 16 components codes in the same manner the position and orientation of neighbor robots. The maximum range allowing a target or a robot to be seen is 1/10 of the diameter of the arena.

The robot actions are rotation and forward move distance.

4. Predictions in CMOMMT

The lazy memory is built using a random action selection policy for the robots, and recording at each time step the total number of targets under observation by the team. It is important to be able to ascertain the quality of the lazy memory: in fact the quality of the non-explicit model that has been built. Certainly, the larger the number of samples in the memory, the better the performance we can expect from the following Q-learning phase. However, we need to know before the Q-learning phase starts, that the memory will prove useful. The coherence of the memory can be measured and compared to an incoherent memory. The coherence of the memory is demonstrated by the consistency with which positive rewards lead to positive reward in similar situations, null rewards lead to null rewards in similar situations and negative rewards lead to negative rewards in similar situations. The larger the number of similar situations the better the quality of the demonstration. It is important to note that the maximum number of situations that can be considered as similar is directly proportional to the size of the memory (the larger the better).

The graphs shown in Fig. 3 report the coherence of the non-explicit model. The X-axis represents the number of similar situations considered (from 1 to 10); the Y-axis represents the percentages of positive, null and negative rewards. Since we are using every situation of the memory as input, if the size of the unit is one, then the set is reduced to the current situation. Note that if one of the situations in the set is negative then the whole set is considered as being of negative reward value; if there is no negative reward and at least one null reward, then the whole set of similar situations is considered of null reward value; otherwise there are only positive rewards associated with the situation set and the corresponding value is positive. The memory has been built using 10,000 situation-action pairs. Then, each point is computed using 10,000 sets of «similar» situations. An incoherent memory can be obtained through a random selection of the situations belonging to the sets of «similar» situations. The results obtained in this case (dotted lines in the Fig. 3) are in complete accordance with the values analytically derived. The computation of the different probabilities is presented here (using the following pseudo-code):

```
r+(1) = 0.210; r0(1) = 0.578; r-(1) = 0.212;
for (i = 2; i<=10; i++)
{
r+ (i) = r+(i-1) * r+(i-1);
r0(i) = 2*( r+(i-1) * r0(i-1)) + r0(i-1) * r0(i-1);
r- (i) = 2*( r+(i-1) * r-(i-1)) + 2*( r0(i-1)* r-(i-1))   +
        r-(i-1) *  r-(i-1);
}
```

Where $r^+(1)$, $r^0(1)$, and $r^-(1)$ are the numbers of positive, null and negative rewarding situations in the entire memory. $r(1)$ is the percentage of rewards in the memory when considering i similar situations.

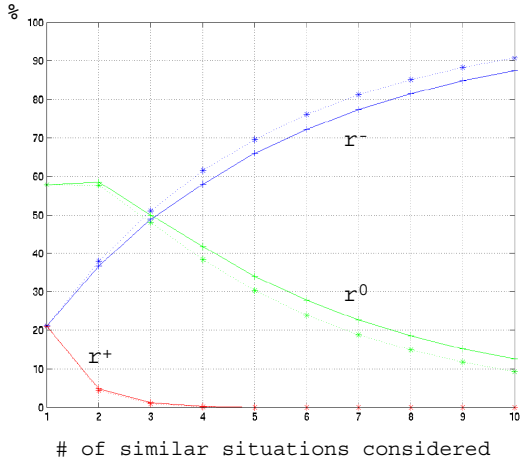

Fig. 3. Coherence of the non-explicit model built by the memory. In dotted lines, the incoherent memory; in plain lines the actual measures. The differences between incoherent memory and the actual one suggest that negative situation-action regions are continuous as are null-reward regions. There is no coherence for the positive rewards, which means there is no (represented) region in the memory that is rewarding. Even a small difference (coherent-incoherent) can have a tremendous impact on the robot performance behaviors: a little gain for each selection of an action can be overwhelming.

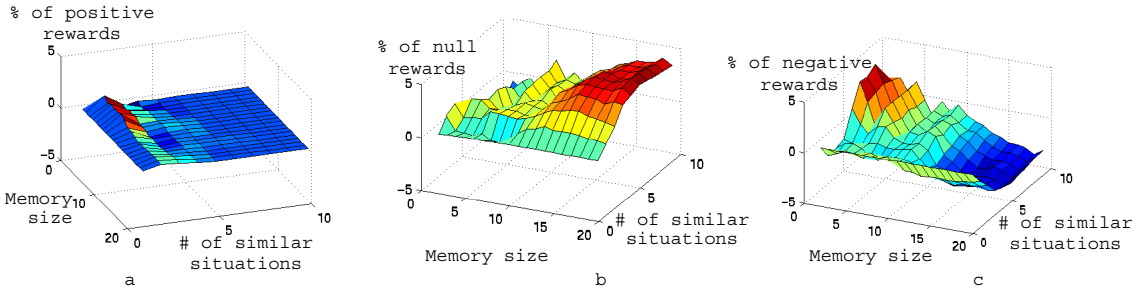

Fig. 4. Differences between the actual memory and its incoherent version. (a) Positive rewards are very similar-impossible to predict. (b) Null rewards are more numerous, and (c) Negative rewards are less numerous: there are specific regions of the search space that encode null rewards and others that encode negative rewards. The memory size has to be multiplied by 100 ([100 .. 2000]).

As we can see, the measured values (plain lines in the Fig. 3) are identical to the incoherent memory ones for the positive rewards. This suggests that it is impossible to predict (using the information in the memory) how to acquire new targets. There is no particular rewarding location (*i.e.*, region) in the search space. On the other hand, there is a significant difference with the incoherent memory for null and negative rewards. This means that negative rewards tend to be associated with similar situations. It shows that negative situations belong to the same regions of the situation space and therefore a learning technique that will be able to avoid these regions will exhibit learning (over a purely random action selection policy). Because it is a global and static measure, the difference between a coherent and an incoherent memory gives no indication of how good the final performance of a given learned behavior will be.
Fig. 4 presents the differences (Z-axis) in percentages of positive, null and negative rewards (measured values - incoherent memory), with respect to the size of the memory and the size of the «similar» situation sets. As we can see, gain is directly proportional to the size of the memory and to the size of the set of similar situations. Also, a large memory tends to smooth the surface (by reducing the standard deviation).

5. The Pessimistic Algorithm

The principle used to select the best action to execute for a given robot in its current local situation is the following: «Find the lower bounds of the utility value associated with the various potential actions that may be conducted in the current situation; then, choose the action with the greatest utility». A lower bound is defined as the lowest utility value associated with a set of similar situations.
The idea behind the Pessimistic Algorithm is that the local situation (to a given robot) is an incomplete observation of the true state of the system, and the fact that - instead of trying to solve the observation problem by completing the observation - we are only interested in ranking the utility of the actions, *i.e.*, the utility of the achieved situations. If we use a unique instance of the robot memory to obtain the utility of the situation, then chances are that the utility attributed to this local situation is due in fact to other robot's actions. This probability decreases proportionally with the number of similar situations that are taken into account. If a large number of situations are considered, then there must be at least one for which the reward directly depends on the local situation. By taking the minimum utility value of the set of similar situations, we are guaranteed that, if the value is null, then this situation achieved is not related to the loose of one target, or more.
The utility U associated with a given situation can be computed in many ways. It can be the exact value of the reinforcement function for this particular situation-action pair, or it can be a more elaborate variable. For example, in our experiments we store the situation-action pairs, plus the number of targets under observation in the lookup table (M). However, the value that is used as utility is +1 if one (or more) targets have been acquired compared to the previous situation, -1 if one (or more) targets have been lost, or 0 otherwise. An exact Q could be used, but it would require to run the Q-learning algorithm with all the samples stored in the memory, several times (in a way very similar to Dyna-Q (Sutton R., 1991)).
The key to successful application of the lazy Q-learning algorithm is the identification of similar situations. We use a measure of similarity of the following form:

$$similarity(a,b) = \sum_i^p |s_a(i) - s_b(i)|$$

Where s_a and s_b are two situations and p is the number of components of the situation. The smaller the value measured, the greater is the similarity. Depending on the size of the memory, similarity between situations (fig. 5) can be computed using brute force (small memory) or in a more elaborate way using a clustering method to pre-select the situations.

6. Experiences

Our first experiment is dedicated to verifying that in the CMOMMT context there is a positive effect on the performance due to cooperation. The existence of such cooperation leverage justifies the need for several robots cooperating together. In a second experiment, the impact of the Pessimistic Algorithm will be demonstrated by comparing the performance of two groups of robots. One uses a memory built by several robots (and targets) and therefore requires a «distribution» mechanism of the rewards, and the second uses a memory built with a single robot (and several targets): no uncertainty in the situation observation - but no possible cooperation.

6.1. Cooperation effect

A-CMOMMT (Parker L.E. & Touzet C., 2000) is today the most effective human-designed robot policy for CMOMMT applications. It combines low and high level control algorithms. Local control of a robot team member is based upon a summation of force vectors, which are attractive for nearby targets and repulsive for nearby robots. High-level reasoning control involves the computation of a probability that no other
robot is already monitoring the target and a probability that a target exists, modeled as a decay function based upon when the target was most recently seen, and by whom. The number of available targets has no effect on the results since the built-in awareness in A-CMOMMT allows the team to have only the nearest robot following a given target.
In figure 6, we plot the performance of a non-cooperative team (i.e., no awareness) vs. the size of the arena vs. the number of robots. Using A-CMOMMT policy for the robots (i.e., adding cooperation through awareness), we obtain the results displayed in figure 7 (same conditions as Fig. 6). Figure 8 reports the gains in performance due to cooperation (results of Fig. 7 - Fig. 6). Each performance graph has been obtained under the same

conditions: 15 randomly moving targets (5% probability of changing direction, maximum speed

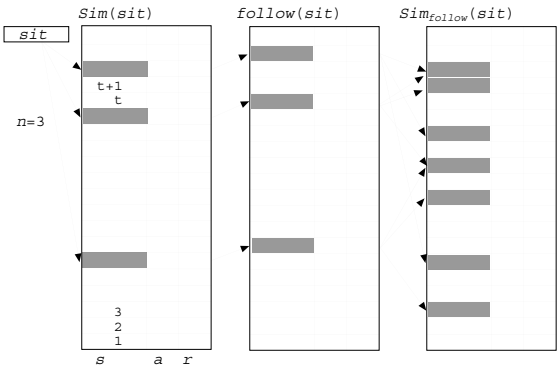

Fig. 5. Illustration of the Pessimistic Algorithm: Building the sets of similar situations.

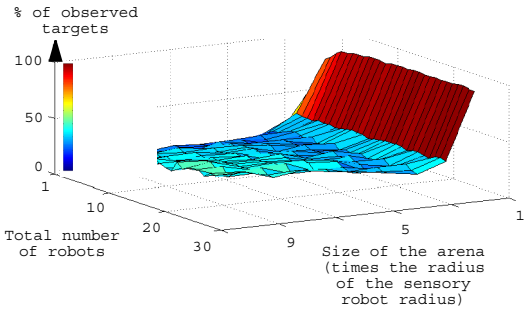

Fig. 6. Performance of a «no cooperation» group of robots. Each robot is equipped with a behavior that places it at the geographical center of the sensed targets, but it does not take into account the other robot positions.

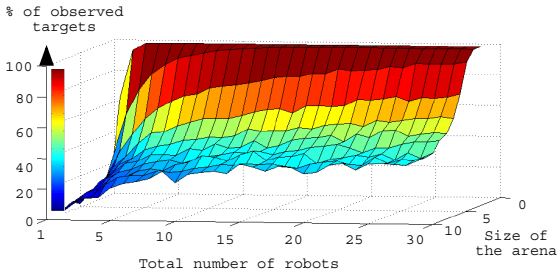

Fig. 7. Performance of a cooperating team. Each robot uses the A-CMOMMT policy, and therefore is aware of the position of the other robots in its sensor field of range.

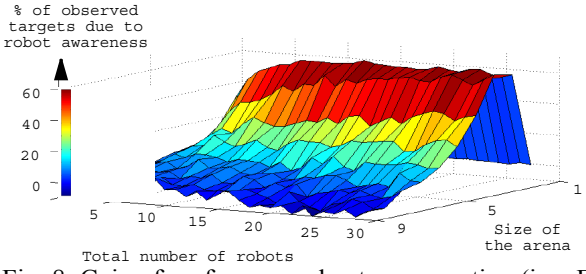

Fig. 8. Gain of performance due to cooperation (i.e., Fig. 7 - Fig. 6): taking into account the other robots improves the overall performance of the team.

less than the maximum robot speed), sensory perception radius for each robot set to 1. The results displayed are the mean of 5 different experiments (each one of 600 iterations).

6.2. Distributed lazy Q-learning

An increase in the performance due to awareness of other team members has been demonstrated in the previous section. This justifies the building of a lazy memory using a team of robots, and the necessary distribution of the global reward information to the individual robots. We verify the efficiency of the Pessimistic Algorithm by comparing the performance of a team of robots using a memory built by one robot (no cooperation case) and a team of robots using a memory built by a group of robots.

Our first experiment, however, is intended to verify that the lazy Q-learned behavior is consistent, at least better than a random action selection policy. Let us remember that each robot situation is a vector of two times 16 components. The first 16 components code the position and orientation of the targets. It simulates a ring of 16 sensors uniformly distributed around the robot body. Each sensor measures the distance to the nearest target. The sensor position around the body gives the orientation. The second ring of 16 components code in the same manner the position and orientation of neighboring robots. The maximum range for a target or a robot to be seen is 1 (arena radius is 5). The actions of each robot are rotation and forward movement. Fig. 9 shows the performance of a lazy (Q-)learning policy, a purely random action selection policy, a user-defined non cooperative policy and A-CMOMMT versus the size of the lazy memory (from 100 to 900 situation-action pairs).

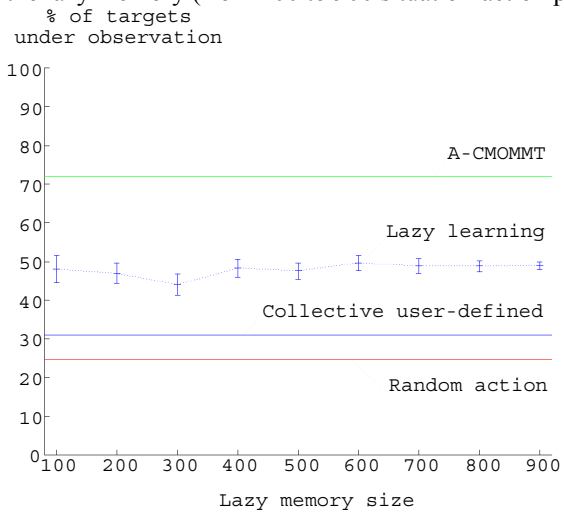

Fig. 9. Performances of the cooperative lazy (Q-)learning compared to a random action selection policy, a user-defined non cooperative policy and A-CMOMMT. The size of the lazy memory varies between 100 to 900 situation-action pairs. There are 10 robots and 10 randomly moving targets. The results are the mean of 10 different experiments per point for lazy learning policy, and 100 experiments for the other 3 policies. Each experiment duration is 1000 iterations.

The measure of performance is an «objective» measure: the mean observation time of all targets. Each point is the average of 10 experiments. The standard deviation is also plotted on the graph. The lazy memories are obtained through an initial exploration involving from 15 to 25 targets and a single robot.
During the sampling, the targets are fixed and the robot's policy is random action selection (5% chances of direction and orientation changes). The reinforcement function returns +1 if the total number of targets under observation increases, -1 if this number decreases, or 0 otherwise.
As we see there is an important performance gain associated with the lazy Q-learning over a purely random selection policy. This clearly demonstrates the importance of lazy Q-learning as a learning technique. Even more interestingly, lazy Q-learning performs better than the user-defined non-cooperative policy. It is important to note that neither policy is aware of the existence of the other robots. Both policies use the same sensory information, i.e., the distance and orientation of nearby targets. It is our opinion that the variation of performance is due to the fact that the lazy Q-learned behavior is somewhat less rigid than the user-defined policy. A lazy Q-learning guided robot will follow a target not as close as it could be, and doing so, will exhibit an erratic path, moving from one side of the target to another, back and forth without losing the target (see Fig. 10). In doing so, the surface under observation per unit of time is larger than the covered surface by the more rigid center-of-gravity-oriented robot. On the other hand, because it does not take into account the neighboring robots, it is easy to understand why the lazy Q-learned behavior performance cannot reach the level of the A-CMOMMT performance.

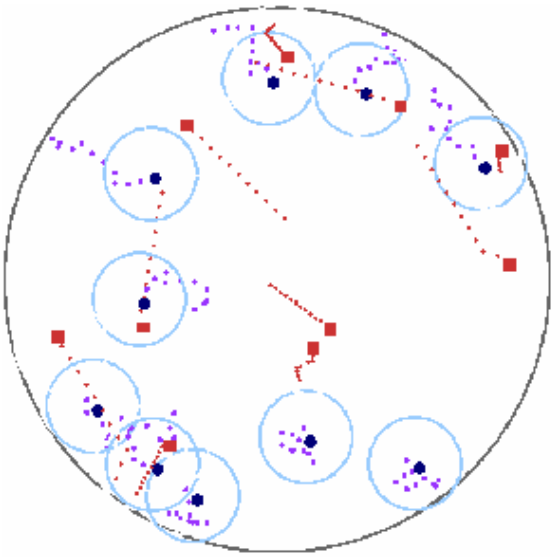

Fig. 10. Bounded arena, with 10 robots using the lazy Q-learned behavior (no awareness involved). The radius of the arena is 6, the radius of the sensory perception range of the robots is 1. There are 10 randomly moving targets. The dotted lines indicate the paths followed by the robots and the targets. The targets have different speeds: the closer the dots, the slower the speed.
Relative to the lazy Q-learned behavior, it is logical that a larger size of the lazy memory implies a smaller standard deviation. This is related to the fact that the larger the number of samples, the greater the probability that the sampling will effectively be representative. As to decide what memory size fits better for a given application, we are only able to point out that the larger the better.
Due to the huge number of possible situation-action pairs, there is no hope of achieving an exhaustive sampling, therefore the performance should always be increasing (even by extremely small values). However, as shown by the discrepancy between collective user-defined and lazy learning policies, even a small memory can bring an important increase in performance.
Figure 11 shows the increase in performance associated with robot awareness vs. a purely collective behavior. Each robot behavior (cooperative or collective) is learned through lazy Q-learning, but only the cooperative team uses the distributed version (i.e., the Pessimistic Algorithm). The dimensionality of the search space is 32 (targets + robots) for cooperative behavior, and only 16 (targets) for collective behavior. The lazy memory is obtained through an initial exploration (length 120 iterations) involving 10 targets and 5 robots. The policies for targets and robots were random action selection. Each iteration the probabilities of direction change for a target was 5%, where it was 100% for a robot. The total number of situation-action pairs in the associative memory is 600 (= $120 * 5$). The reinforcement function is provided to the group as a whole: it returns +1 if the total number of targets under observation increases, -1 if this number decreases, 0 otherwise.
Compared to Fig. 8, we see that the impact of cooperation is less noticeable in our learning experiment: a maximal improvement of 25% (compared to 60%). However the shapes of both surfaces are similar: the preferred arena size is between 2 and 6, and the greater the number of robots, the more important the impact on the performance. The counter-effect of very large arenas is easily spotted, in particular for small numbers of robots.

7. Related work

(Schmidhuber J. & Zhao J., 1997) study systems of multiple reinforcement learners. They use a simple backtracking method called the «success-story algorithm» (SSA) to evaluate the learning modifications that have occured since the last evaluation at certain times, and undo all those previous modifications that were not empirically observed to improve the performance. The SSA is a principle, like the Pessimistic Algorithm, that can be plugged into a wide variety of learning algorithms. The main difference is that SSA uses a criterion related to the increase of positive rewards, where our algorithm is using lower utility bounds. The SSA does not apply to a CMOMMT

application because, as we have shown, positive rewards are impossible to predict and will elude any policy that would attempt to maximize the rewards.
The problem of distributing the global reward to the individual robots is related to the fact that the situation observed by any robot is local. It is an observation that is incomplete. It can be considered as a Partial Observable Markov Decision Problem (POMDP). Apart from the naive strategy that consists of ignoring the problem and treating observations as if they were the states of the environment, a number of algorithms have been developed by the POMDP community (Littman M., 1996) (Singh S. *et al.*, 1994). However, all these algorithms (a representative example is provided by the Witness Algorithm (Cassandra A., 1994)) aim to reduce the uncertainty on the observation, principally by using contextual information (like previous sequences of observations). We have shown that a CMOMMT application does not allow to predict an action that will allow the acquisition of a new target. Therefore, there is no way to generate positive rewards and this is not due to an absence of precision on the situation. Relative to negative and null rewards, the pessimistic algorithm uses the whole memory to find similar situations, where the POMDP algorithms usually use the last few encountered situations.

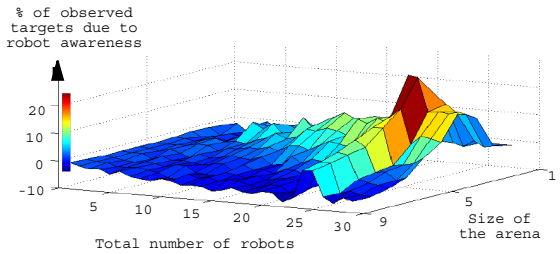

Fig. 11. Percentage of observed targets due to the distributed (i.e., Pessimistic Algorithm) version of the lazy Q-learning (cooperative minus non cooperative) vs. the number of robots (N = (1 , 30)) and vs. the radius of the bounded arena (1, 9). There are 15 randomly moving targets. The sensory perception radius for each robot is 1. This graph is obtained in similar conditions as those of Fig. 8, except for the behaviors, which are all lazy Q-learned here.

(Darrell T., 1998) use hidden-state reinforcement learning techniques to solve decision tasks in POMDP. They apply their learning technique to face recognition, a problem that involves taking actions (adjusting the perceptual apparatus or other effectors) depending on the situation (image). To avoid perceptual aliasing while learning, all similar experiences are combined when computing the utility of a possible action, including experiences with both target and distractor patterns. Using McCallum's Nearest Sequence Memory (McCallum R., 1995), they find the *K* nearest neighbors in the memory for a given action, and compute their average *Q* value. In a CMOMMT application, where if *n* is number of robots in the team, there is an average chance of *1/n* that the reward belongs to this individual, averaging rewards will not solve the problem. That is why the Pessimistic Algorithm uses a lower bound of the utility (*i.e.*, the worst hypothesis).

(Michaud F. & Mataric M., 1998) use a set of initially given «behavior producing» modules to choose from, and a memory-based approach to dynamically adapt the selection of these behaviors according to their history of use. The experimental set-up is a multirobot foraging task. Their use of behaviors-instead of actions-allows reducing the search space. There are very few behaviors compared to actions. Also, a comparison of performance with approaches synthesizing behaviors at the action level is difficult. What they learned is a controller policy of the type: «which behavior to activate». The selection of the behavior is based on the sum of the expected rewards, multiplied by a frequency variable (their objective is to learn in non-stationary conditions). It must be noted that the other robots were to provide non-stationary conditions in the environment and cooperation was not sought.

(Mataric M., 1997) points out that learning social rules appears to require a non-greedy approach. She presents results which tend to confirm that vicarious (global) rewards are necessary to achieve learning with a group a mobile robots learning to yield and share information in a foraging task. As with the previously cited work, it is not the actions, but instead the behaviors that are to be selected. There are only 5 behaviors and 6 situations to consider.

8. Conclusion

Effective cooperation requires that the global problem-solving state influence the local control decisions made by a robot. A robot with a purely local view of the problem cannot learn effective cooperative control decisions that may have global implications, due to the uncertainty about the overall state of the system. In a reinforcement learning paradigm, the only available global information is the reinforcement function that measures the performance of the group during a learning phase. CMOMMT application does not provide global information in the test phase. So, learning must be accomplished using the information gathered during the initial stage. A lazy memory is then a logical choice. However, to use that memory, one has to distribute the reward associated with the team behavior to each individual robot. We have proposed a distribution mechanism for the lazy Q-learning: the Pessimistic Algorithm, which is able to compute for each team member a lower bound of the utility of executing an action in a given situation. The results show that this algorithm accounts for increased performance of the group of robots vs. a non-cooperative group.
To our knowledge, this is the first successful attempt to apply «true» sub-symbolic learning with a team of robots. The related work either assumes the existence of a model and tries to eliminate the uncertainty associated

with a situation (POMDP); uses local indicators that help individual robots to perform their tasks; or deals with a symbolic level of representation as in the multi-agent domain.

Apart from the distribution of the rewards, there are several issues of importance for lazy Q-learning. The first is the quality of the reinforcement function (Santos J. M. & Touzet C., 1999), the second is the quality of the sampling (size, representativity), and the third is the quality of the probing process (generalization). The state-of-the-art in these domains only allows us to assume that non-optimal values and criterion were used in the experiments reported here. However, it is our opinion that this does not alter the illustrative quality of the results presented here (see Fig. 11). We will nevertheless continue our research in each of these three issues, as we will also validate the Pessimistic Algorithm in more challenging conditions.

9. References

Aha, D. (ed.) (1997), *Lazy Learning*, Kluwer Academic Publishers, .

Cao, Y. U., A. Fukuaga and A. Kahng (1997), Cooperative Mobile Robotics: Antecedent and Directions, *Autonomous Robots* 4, pp. 7-27.

Cassandra, A. R., L. P. Kaelbling and M. Littman (1994), Acting Optimally in Partially Observable Stochastic Domains, Proceedings of the Twelfth National Conf. on Artificial Intelligence (AAAI-94), Seattle, WA.

Darrell, T. (1998), Reinforcement Learning of Active Recognition Behaviors, Interval Research Technical Report 1997-045. Available from: http://www.ai.mit.edu/people/trevor/papers /1997-045/ , Accessed 2004-01-15 Portions of this paper previously appeared in Advances in *Neural Information Processing Systems* 8, (NIPS '95), pp. 858-864, MIT Press, and *Intelligent Robotic Systems*, M. Vidyasagar ed., pp. 73-80, Tata Press, 1998.

Dorigo, M. (1996), Introduction to the Special Issue on Learning Autonomous Robots, M. Dorigo (Guest Ed.), *IEEE Trans. on Systems, Man and Cybernetics*-part B, Vol. 26, No. 3, pp. 361-364.

Heemskerk, J. and N. Sharkey (1996), Learning Subsumptions for an Autonomous Robot, IEE seminar on self-learning robot, Digest No: 96/026, Savoy Place London, 12, England.

Kaelbling, L., Littman, M., and A. Moore (1996), Reinforcement Learning: A Survey, *Journal of Artificial Intelligence Research* 4, pp. 237-285.

Littman, M. L. (1996), Memoryless policies: theoretical limitations and practical results, From Animals to Animats 3: Proceedings of the Third Int. Conf. on Simulation of Adaptive Behavior, Cliff, d., Husbands, P. Meyer, J-A. and Wilson, S. W. (Eds.), Cambridge, MA, MIT Press.

McCallum, R. A. (1995), Instance-based State Identification for Reinforcement Learning, in *Advances In Neural Information Processing Systems* 7, MIT Press.

Mataric, M. J. (1997), Learning Social Behavior, *Robotics and Autonomous Systems* 20, pp. 191-204.

Michaud, F. and M. J. Mataric (1998), Learning from History for Behavior-Based Mobile Robots in Non-Stationary Conditions, Special joint issue *Machine Learning* and *Autonomous Robots* Journals, H. Hexmoor and M. Mataric (Guests Eds.).

Parker L. E., Touzet C. & Fernandez F. (2002), Techniques for Learning in Multi-Robot Teams, *Robot Teams: From Diversity to Polymorphism,* Tucker Balch and Lynne E. Parker (eds.), Natick, MA.

Parker L. E. & Touzet C. (2000), Multi-Robot Learning in a Cooperative Observation Task, *Distributed Autonomous Robotic Systems 4,* Lynne E. Parker, George Bekey, and Jacob Barhen (Eds.), Springer, pp. 391-401.

Santos, J. M. and C. Touzet (1999), Exploration Tuned Reinforcement Function, *Neurocomputing*, Vol. 28 , No. 1-3, pp. 93-105.

Sheppard, J. W. and S. L. Salzberg (1997), A Teaching Strategy for Memory-Based Control, *Lazy Learning*, D. Aha (Ed.), Kluwer Academic Publishers, pp. 343-370.

Schmidhuber, J. and J. Zhao (1997), Multi-Agent Learning with the Success-Story Algorithm, *Distributed Artificial Intelligence Meets Machine Learning-Learning in Multi-Agent Environments*, G. Weiss (Ed.), Lecture Notes in Artificial Intelligence, Vol. 1221, Springer-Verlag, pp. 82-93.

Singh, S. P., T. Jaakkola and M. Jordan (1994), Learning Without State-Estimation in Partially Observable Markovian Decision Processes, Proceedings of the Eleventh Int. Machine Learning Conf..

Sutton, R. S. (1991), Reinforcement Learning Architectures for Animats, Proceedings of the First Int. Conf. on Simulation of Adaptive Behavior, *From Animals to Animats*, J-A Meyer and S. W. Wilson (Eds.), MIT Press, pp. 288-296.

Sutton, R. and A. Barto (1998), *Reinforcement Learning*, MIT Press Bradford Book.

Touzet C. (2000), Robot Awareness in Cooperative Mobile Robotics, *Autonomous Robots*, Vol. 8, No. 1, pp. 87-97

Touzet, C. (1997), Neural Reinforcement Learning for Behaviour Synthesis, Special issue on Learning Robot: the New Wave, N. Sharkey (Guest Ed.), *Robotics and Autonomous Systems* 22, No 3-4, pp. 251-281.

Watkins, C. J. C. H. (1989), *Learning from Delayed Rewards*, Ph.D. thesis, King's College, Cambridge, England.

Whitehead S., J. Karlsson and J. Tenenberg (1993), Learning Multiple Goal Behavior via Task Decomposition and Dynamic Policy Merging, *Robot Learning*, J. Connell and S. Mahadevan (Eds.), Kluwer Academic Publishers.

This research is funded in part by the Engineering Research Program of the Office of Basic Energy Sciences, U.S. Department of Energy, under contract No. DE-AC05-96OR22464 with Lockheed Martin Energy Research Corporation.