Lotus 2010 Team Description

Mohammad Reza Khojasteh[1] ,Mehdi Ajdari[1], Ali Ajdari [1],
Mehrana Zohari[1], Aida Kazimi[1]

[1]AI & Robotic Research Laboratory, Persian Nice Dream Co. Shiraz – Iran.
info@pndream.com

Abstract. The aim of this paper is to technically describe Lotus 2010 Team's contributions to rescue agent development that is going to participate in RoboCup competitions this year. These papers include Lotus 2010 agent's architecture, a technique called Reactive Planning for produces a set of condition-action rules, and at last we are trying to implement cooperation without communication in such a complex domain.

Keywords: RoboCup, Cooperation, Communication.

1 Introduction

RoboCup is an international, interdisciplinary research project in which physical agents compete in dynamic real-time environments. In fact, the main goal in this domain is minimizing the damage by helping trapped agents, extinguishing fiery collapsed buildings, and rescuing damaged civilians. The RoboCup Rescue Simulation system provides a test bed for research in artificial intelligence, etc. This goal has been achieved by simulating a large urban disaster, and modeling heterogeneous agents' actions in this environment. As a rescue simulation team we have tried to improve available solutions for each agent's problems. Our contributions include new ideas for cooperation with out communication [1].

This Cooperative work by multiple mobile robots is effective for dangerous work at building sites, for repair work in stricken areas, etc [2]. Coordination and cooperation between agents have been a topic of research for many years [3], [4], and is a part of everyday life [5]. In many of multi agent systems, agents are designed as uninformed and limited creatures [6] so for tight cooperation, various kinds of information must be exchanged between robots in an unknown situation [2]. There are two major reasons to communicate between agents in unknown situation, first sending necessary information s of environment to other agents that aren't accessible to it, second implementing team coordination strategies [19]. However, in a number of cases of cooperation between agents, no communication takes place. As usual the main interest behind the Lotus's effort in this domain is developing and using Cooperation without communication methods in such a complex domain. We are trying to solve Cooperation with out communication with Scenario [1] and "Best Corner in State Square" [8] methods in RoboCup Rescue Simulation domain.

In this article we are going to present the main features of Lotus 2010 that include implement cooperation with out communication for our team and Lotus 2010 agent's architecture. We try to create a bed test for testing the results of our article in a multi-agent environment that there is no enemy there (see [1]).

1.cooperation with out communication

Some of Lotus2009 and Persia2006 team members have formed Lotus2010. In Lotus2010, we have integrated two methods (named "Scenario" and "Best Corner in State Square") which could make new a method of cooperation without communication in RoboCup Rescue Simulation environment.

In environments where coordination is essential for goal accomplishment, agents must thus appropriately communicate for cooperative task achievement [9]. However, in a number of cases of cooperation between agents, no communication takes place (see also [10]). For instance, without communication people coordinate their actions so that they do not bump in to each other on the street. Cooperative problem solving without communication is an often-studied field within multi-agent research [11]. Cooperation without communication is also required for more specific tasks, such as playing a soccer game where two defense players coordinate their actions without communication, as the speed of the game prohibits this [11].

1. 1 do we need cooperation without communication?

Any researchers have indicated that communication is not necessary for achieving coordination [10], Stan Franklin in an article [12] points to the out the communication was not involved in many cases in real life scenario where coordination was achieved. He made the following three observations: 1. Coordination with or without communication is a property of multi agent systems. 2. Coordination without communication was common in such systems in real life, e.g. players of a football team coordinating without explicit communication.3.Repeated and frequent sampling of the environment and responding thereto, is the underlying mechanism of such coordination. In addition the three reasons of Franklin, it is possible to say that sometimes communicating is very difficult or even impossible because, unfortunately one of the critical problems in multi agent system (such as RoboCup Rescue Simulation) is unstable and unreliable communication.

1. 2 Scenarios

We implement agents which follow and obey the specific rules under any circumstances [1]. We call these rules a scenario, which includes a complete necessary set of information for agent behavior.

Formal definition is:

$S = (\mu, \alpha, \partial)$

μ = Condition
α = Space that agent has permission to move in map
∂ = Set of action that agent can do them

Every scenario includes every agents' behaviors in particular situation.

1. 3. The “Best Corner in State Square” Technique

The number of states in a simulated robotic rescue domain is very large and therefore, for an agent to consider all of them is impossible. In fact, the most important job in this regard, is to design a proper generalization of the environmental state space for the agent. If we call the set of agent’s actions A, each agent will have |A| possible actions in each of |V| states and therefore, the set to be learned for the agent will have at most $V \times A$ elements. If we choose the sets V and A wisely, our agents can learn effectively in a complex and real-time environment using limited samples. In fact, the sets V and A should have the properties that cover all states and actions as much as possible and they should be good mappings of the sets of all possible states and actions that exist in the domain of agents. For generalization of the environment, we divide the map into equal squares each with side length 20. By doing so, at any moment of the simulation, each agent is in one of these squares. Considering the fact that each agent has a limited view of its environment, it can’t count on the whole map. So, we have focused on 8 squares around the square that has the agent in it. We have considered a numeral quantity for each of 8 squares around the agent with give score to their.

1.4 Implementation

We have merged two ideas which are Best corner and Scenario in order to create an environment for cooperation without communication. In following section we explain this cooperation.

1.4.1 Search Zone

In every scenario what to do where to go in a map has been specified for each agent. All agents divide the map to number of the agents in the same type and after that some zones will be created. Afterward they will be indexed from the north west of the map. The agent which has the minimum ID moves toward a zone with the lowest index. The next agent moves toward the next zone. So, there is only one agent in each zone. This action will be done for our three type agents and lead us to a way which our agents are divided in map.

```
Condition = when no danger is viewed
Function senarioInSafeMode (agent)
Begin
    For I = 1 to object (agent).count then
```

```
        Zone [I]:= createZone (I);
    While Condition do
    Begin
      While not Object (agent).zoneSearchComplete () do
           Object (agent).searchInYourZone (&Zone [I]);
       agent.zone ++
    end;
 End;
```

As it has showed above, once an agent selects a scenario for carrying out its jobs, it must be submissive to the scenario comments until the time that another scenario circumstances come true. SearchInYourZone function first divides the whole zone into 20*20 squares. It is recommended to each square from the zone center and gives a number in a range of 0 to 100. The higher the number is, the more important the squares are for the agent. If an agent faces an action (which is in its territory) in a square, it will start doing that action and will change the value of the square to 100.

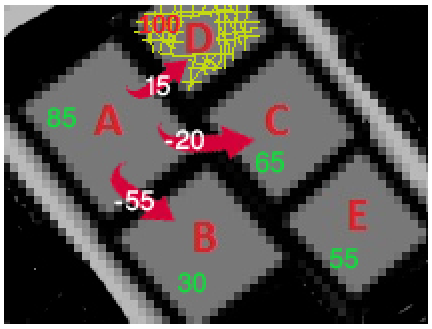
Otherwise, the action will be located in a queue till the time that it faces the related agent.

1.4.2 Scenario for extinguish fire

In this scenario we consider each building as one square. We can calculate the state vector of each square using the following formulation:

Pi = NumOf (Civilian) / Count (Civilian) * 100 + danger

In this formulation, danger is used to represent how much this square would be at the risk of catching fire. This number is calculated based on considering some factors such as the distance to the fire, the distance to refuge, the number of the blocked roads, and the direction of the fire towards the square being considered in the formula. Then, we compare the numbers obtained in each square's vector to its neighbors. Squares with positive vectors are at greater risk for catching fire. Agents who are assigned to extinguish the fire or rescue are better to move to the squares which have negative vectors. Consider the following illustration:



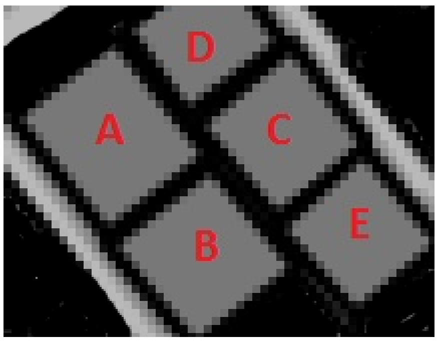
In this picture building D is burned and so is assigned a score of 100, building A has a score of 85 score, building B has a score of 30, building C has a score of 65, and building E has a score of 55. Greater score values mean that their corresponding buildings are in greater dangers. By a positive vector we mean that after our calculation, the direction of that vector would be towards a building which is more dangerous than the building which contains the start point of the vector. Considering the above mentioned facts, Building B would be a safe place for extinguishing fire.

1.4.3 Search Point

There is a scenario for searching in zones which all agents should follow. If each agent wants to search each building, we will lose a lot of time. We know that our agents are able to see or sense civilians based on the locations they are in. According to this fact, we would try to search buildings in the same neighborhood together in order to save time. To do so, we calculate an average of neighbors for each building. This average could be calculated with the following formula:

$X_i = \sum \text{Neighbors (building i)} / \sum \text{Neighbors (Neighbors (building i))}$

After calculating this value for each building, we would try to find the smallest Xi value in each zone to start searching from the point having that value. For instance, consider the following illustration:



In this map, we would have:

$X_A = 3 / (2+4+3+2) = 3/ 11$
$X_B = 3 / (3+4+2) = 3/ 9$
$X_C = 4 / (3+3+2+2) = 3/ 10$
$X_D = 2 / (3+4) = 3/ 7$
$X_E = 2 / (2+4) = 2/ 6$

So, it seems that building A is a better point to start searching for civilians. After searching building A, we don't need to search buildings D, C, B, and E. That is because our agents are able to sense civilians in those buildings while there are searching for civilians in building A.

If an agent has several jobs to do in any instant, it should determine the most important job with reactive planning.

2 Architecture of Lotus 2010 agent

Our layered agent architecture consists of three layers including communication layer, Planning and decision making layer, Strategy and Information layer.

2.1 Planning and decision making layer

This layer contains components that are used to planning and decision making process. Planning is one of most important cases that we implemented in this architecture. In this layer we have used a technique called Reactive Planning. Reactive Planning is described entirely in the following.

2.1.1 Reactive Planning

In theory, a reactive planning system can handle exogenous events as well as uncertain effects and unknown initial conditions. Reactive Planning is a technique that be used in dynamic and real-time environments. A reactive planning system can react to events, which have not been foreseen at planning stage for different reasons. Reactive planning is increasingly becoming an active area in AI research [7].The more general term, planning, is viewed within the AI literature as the process of finding a sequence of actions that brings a system from a given state to a desired goal state [15]. A system is called reactive, if it can react in an acceptable amount of time to any changes that occur in the world while the system is running (Wilkins, 1988). A reactive planning system can react to events, which have not been foreseen at planning stage for different reasons (e.g. because they were not known or because it would have been too expensive to consider them at planning stage). Reactive planning integrates plan generation and execution, which constitutes a suitable platform to deal with real-world problems. In reactive planning, an agent is defined as a combination of a planner plus a reactor [16]. However, we made some changes in this technique and our implementation is somewhat different from that but the main idea not changed.

2.1.2 Reactive Planner

The Reactive Planner is more like a hybrid of both planning and reactive planning similar to what is described by Bjrklund in reference 17. The Reactive Planner is the brain of the robot and all the strategic decisions are made in this component. The Reactive Planner use information from the Strategy and information layer to make real time decisions. Reactive Planner will select best task for agent. In order to select the best task all tasks will be evaluated. The process starts with checking all conditions the task has. If all conditions are fulfilled the Reactive Planner will use information from the Strategy and Information layer to calculate a value how suitable this task is that called priority because most valuable task must have higher priority. After all tasks have been evaluated the task with the highest priority will be selected as the best task and sent to the queue of tasks for execution.

The calculations of how good a task is are based on world state. Priority of a task is a percent that ranges from 0 to 100 and shows how good a task is.

```
For each task in tasks Collection
  If (all conditions are fulfilled) then
        Calculate how good the task is for this state
        Store task and calculated value
  End if
Next

If (there is no suitable task) Then
  Add default task to queue of tasks
Else
  Add highest priority task to queue of tasks
End if
```

2.1.3 Plan Monitor

Plan Monitor manages queue of tasks. It monitors the execution of a plan, to update the current plan (first item of queue of tasks) by removing completed tasks, inserting new tasks (if current task needs to be repaired), removing existing tasks in order to accommodate tasks with a higher priority, (e.g., it changes its mission or the task turns out to be infeasible), or repairing its plan when failure occurs (e.g., by inserting new tasks to re-enable a failed task conditions). Tasks priorities are handled essentially via plan merging by accommodating first tasks with a higher priority. For instance, once the robot has its plan for the mandatory tasks, it should accept volunteer duties depending on time availability.

2.2 Strategy and Information layer

Strategy and Information layer provides any information from world by processing data and calculating all possible requirements of other components. This layer has 2 components: World Model, Strategy management.

4. Conclusion

The results of our simulations show that the “Best Corner in State Square” technique in which the agent’s decision is based on its local space is very effective when one wants to use local cooperation. Also scenario might play an important role in cooperating without communication of agents.

Reference

[1] Ajdari M., Heidari H., Tavakoli A., Massh B. A, Cooperating Without Communication in Multi-Agents Systems by Emulating From Human Society, Proceedings of The *1st* RoboCup IranOpen International Symposium.

[2] T. Fujita, H. KimUri, “Tight Cooperative Working System by Multiple Robots,” Graduate School of Information Systems University of Electro-Communications, Chofu City, Tokyo 182-8585, JAPAN, Proceeding of the 1998 IEEE/RSI Intl. Conference on intelligent Robots and Systems Victoria, B.C, Canada October 1998.

[3] Dignum F., “Agent Communication and Cooperative Information Agents”. In M. Klusch and L. Kerschberg (eds.) Cooperative Information Agents IV - The Future of Information Agents in Cyberspace (LNCS-1860), Springer- Verlag, 2000, pages 191-207.

[4] Genesereth, M.R., Ginsberg, M.L., and Rosenschein, J.S., “Cooperation Without Communication”, The National Conference on Artificial Intelligence, Philadelphia, Pennsylvania, August 1986, pp. 51-57.

[5] Tibor Bosse1, Mark Hoogendoorn1, Catholijn M. Jonker, “The Distributed Weighing Problem A Lesson in Cooperation without Communication”

[6]Jennings N. R., “Cooperation in Industrial Multi-Agent Systems.” World Scientific, Singapore, 1994.

[7] Vassileva J.: Reactive Instructional Planning to Support Interacting Teaching Strategies. Federal Armed Forces University – Munich, Germany

[8]Khojasteh M. R. and Meybodi M. R., The technique “Best Corner in State Square” for generalization of environmental states in a cooperative multi-agent domain, Proceedings of the 8th annual CSI computer conference (CSICC’ 2003), pages 446-455, Mashhad,Iran, Feb. 25-27, 2003.

[9] Doran, J.E., Franklin, S., Jennings, N.R., Norman, T.J., “Cooperation in Multi-Agent Systems”, the Knowledge Engineering Review, 1997 (3), pp. 309-314.

[10] Omar Ahmad, Javed Iqbal Mahmood A.Khan, “coordinate with out communication: finite states automation in behavior base multi Robots system,”

[11] Tibor Bosse1, Mark Hoogendoorn1, Catholijn M. Jonker, “The Distributed Weighing Problem A Lesson in Cooperation without Communication”

[12]Stan Franklin, “Coordination without communication,” unpublished. http://www.cs.memphis.edu/~franklin/coord.html.

[13] Remco de Boer and Jelle Kok, “Synthetic Multi Agent System The UVA Trilearn 2004 Robotic Soccer Simulation.

[14]. Ball D., Wyeth G.: Classifying an Opponent’s Behavior in Robot Soccer. School of Information Technology and Electrical Engineering University of Queensland Australia, 4072

[15] VERIMAG: Analysis and Tools: Scheduling and Planning Algorithms (2005).

[16].Wolverton, M., Washington, R.: Segmenting Reactions to Improve the Behavior of a Planning/Reacting Agent. In Proc. of AIPS. (1996) 245-250.

[17]. Bjrklund A.: Cooperating Soccer Playing Robots, Department of Computing Science Ume University, SE-901 87 Ume , Sweden (2002).

[18]. Narendra K.S. and Thathachar M.A.L., Learning Automata: An Introduction, Prentice Hall, Inc., 1989.

[19] Khojasteh M. R., Faraji F. Kazimi A., Nouri Zadeh M., Ghasemi Nik Z., Jafari A.
Persia 2006 Team Description, 2006.