J. Basic. Appl. Sci. Res., 2(9)9705-9709, 2012
© 2012, TextRoad Publication

ISSN 2090-4304
Journal of Basic and Applied Scientific Research
www.textroad.com

Using Cooperation without Communication in a Multi-Agent Unpredictable Dynamic Real-Time Environment

Ali Amini

Computer Engineering Department, Shiraz University, Iran

ABSTRACT

Developing a program for robot teams can be a difficult task. This article includes a brief description of general ideas of the Persian Gulf team. One of the most general goals of the Persian Gulf team is to carry out the idea of cooperation without communication between the Agents. This article describes how this idea will be carried out. We use simulation environment for experiment this research.2D soccer simulation server is free software for simulate soccer game.

KEYWORDS: Multi-agent systems, Communication, Robocop, Software engineering

INTRODUCTION

The aim of this article is testing the results of non communicational collaboration in smart multi-agent systems by sketching a scenario. The purpose of choosing the Soccer Server 2D environment as a test for the results of this article is that in this environment there is a basic Agent which is the result of the UVA [1] team attempts. This Agent is capable of doing all kinds of low level skills. Besides, carrying out this idea according to base code of UVA team seemed to be accessible. Regarding this, the members of the team started to test the results of the article with a small change in the architecture of UVA 2003 team. In the first part, we will briefly describe the reasons for Non communicational collaboration in smart multi-agent systems. In the second part, the soccer simulation server will be described. In the third part Persian Gulf Agent architecture will be described. At the end is a brief description of the scenario and how it has been chosen.

The reasons for using cooperation without communication[2]

Although communication is a powerful instrument for accommodating interaction (and has been examined in previous work [6]), in our analysis here we consider only situations in which communication between the agents is impossible. While this might seem overly restrictive, such situations do occur, e.g., as a result of communications equipment failure or in interactions between agents without a common communications protocol.

The goal of multi-agent systems in most cases is solving the complex problems by distribution it through different dimensions. In such systems a team of voluntary agents is able to achieve specified goals by collaboration. In most of such systems the agents are designed as existences with limited information and abilities. The communication between two agents is necessary due to two following reasons: Firstly, to facilitate the dispatching information to the agents that the necessary information is not available for them and secondly the initiation of a collaboration strategy by diversified logarithm and procedures. Unfortunately, one of difficulties of artificial multi-agent systems is lack of stability and secure communication which also are visible in the real world. It is evident that lack of communication or unsecured communications in a team of agents with common goals can rebuff achieving final goals [3].Some of the reasons which cause the real or resembling multi-agent systems to repeat the unsecured communications or cut of communications can be numerated as following:

Disorder and cut of communicational systems: this reason is the first and utmost reason of lack of secure communication .Each agent in real or software environment may use different facilities to communicate. For example, cut of telecommunication can be happened in relief and rescue operations while a team of several policemen and fire fighters are in a mission. In a situation, the cut of battery power of one agent or wave interference and noise in frequencies can result in cut of communication and results in disharmony.

Decrease in quality of communication due to miscommunications: some duties are so rapid that there is not enough time to communicate and process the information, especially if miscommunication also interferes, it can cause the work to finish or the quality to decree dramatically. This problem occurs in systems with critical function such as football playing in taking offside pass by defense players of opposite team in which the necessary time for communication may result in missing the fatal time and inefficiency.

***Corresponding Author:** Ali Amini, Computer Engineering Department, Shiraz University, Iran.

High traffic of communication channel: High traffic in excessive messages cause that the necessary messages would delay or not reach its destination. For example, if in a football game all players want to communicate to a co player , the situation of apposite players who are persuading him from behind [1] , this players may not able to hear more important messages such as pass to a specified point or would not receive all messages correctly .

Decrease of security of collaboration and risk causing: This situation can occur in multi-agent systems which the agents are enemy to each other. For example, request to a co-player by his colleagues to pass to a specified point may be misused by players of opposite team.

Lack of communicational infrastructure: In some of multi-agent systems for different reasons there is not any infrastructure for communication between agents .So, in some systems the agents oblige to employ the methods for collaboration to decrease the excessive communication to a minimum level so they can achieve their goals with limited and secured communication.

Considering the above mention reason is seems that using cooperation without communication in multi-agent systems causes the agents reach the team's goal faster. For this purpose we used a specific method in 2D soccer simulation. In this method an activity area is defined for each player and in each area the player performs a series of tasks which make the team reach its aim. For example in the following figure a football field in 2D soccer simulation has been divided in to 11 part that in each part each player has a specific task.

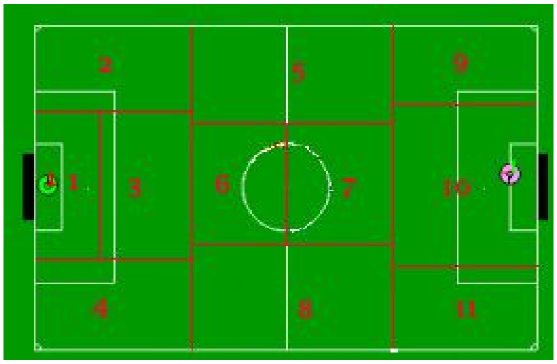

Figure 1
A football field in 2D soccer simulation has been divided in to 11 part

For instance, it is defined for the right defense, to whom it shoot pass the ball, how much distance it should carry the ball, where to attack the opponent player. For this purpose we used different scenarios. These scenarios can change during the game rapidly. Each scenario determines the role of player in specific formation. We will describe this issue in Scenario and its role in cooperation with out communication section.

Soccer simulation system

The RoboCup Soccer Server is a soccer simulation system which enables teams of autonomous agents to play a match of soccer against each other. The system was originally developed in 1993 by Dr. Itsuki Noda (ETL, Japan). The soccer server provides a realistic domain in the sense that it contains many real world complexities such as sensor and actuator noise and limited perception and stamina for each agent. One of its purposes is the evaluation of multi-agent systems, in which the communication between agents is restricted. The information presented is largely based on [7] and partly the result of several experiments that we have performed when studying the behavior of the soccer server. The RoboCup simulator consists of three main components:

- the soccer server
- the soccer monitor
- the logplayer

A simulation soccer match is carried out in client-server style. The soccer server provides a domain , simulates all the movements of objects in this domain and controls a soccer game

according to several rules. A simulation soccer match is carried out in client-server style.

All communication between the server and the clients is done via UDP/IP sockets. Using these sockets, client programs send requests to the server to perform a desired action (e.g. `kick'). When the server receives such a message it handles the request and updates the environment accordingly. After fixed intervals the server also sends sensory information about the state of the world to each player. Although direct communication between the clients is not permitted, it is allowed for clients to communicate with each other indirectly via the server using say and hear protocols which restrict the communication. When a match is to be played, two teams each consisting of 11 separate clients make a connection with the server. The objective of each team is to direct the ball into the opponent goal, while preventing the ball from entering their own goal.

It is important to realize that the server is a real-time system working with discrete time intervals (cycles). Each cycle has a specified duration defined by the server parameter simulator step1 which in the current server version has a value of 100ms. During this period clients can send requests for player actions to the server and the server then collects these requests. It is only at the end of a cycle however, that the server executes the actions and updates the state of the world. The server thus uses a discrete action model. When a client sends multiple action requests to the server during a single cycle, the server randomly chooses one for execution and discards the others. It is thus important that each client sends at most one action request during a cycle. On the other hand, sending no request during a given cycle will mean that the agent misses an opportunity to act and remains idle. This is undesirable since in real-time adversarial domains this may lead to the opponents gaining an advantage. Therefore, slow decision making leading to missing action opportunities has a major impact on the performance of the team.

The architecture of agent

From a software engineering perspective, architecture alludes to "the overall structure of the software and the ways in which that structure provides conceptual integrity for a system" [4]. In its simplest form, an architecture describes the structure of the system's components (modules), the way in which the components interact and the structure of the data that are used by these components. As such, the architecture can serve as a framework from which abstractions or more detailed descriptions of the system can be developed. An architectural design can be represented by a number of different models [5]. The architecture of the Persian Gulf agent is made up of 3 layers: Fixed rule, low level and high level layers. Each layer is separated from other layers according to the figure below. In this architecture the fixed rule is added for the cooperation without communication can be considered as the primary knowledge of the agent.

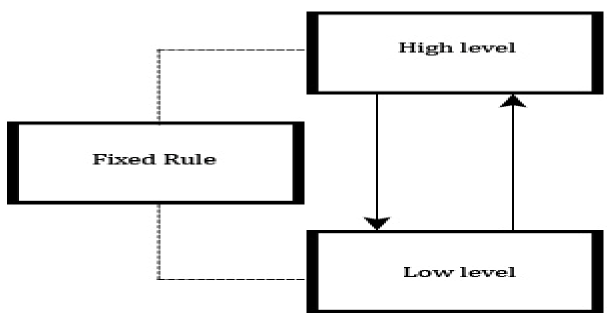

Figure 2

Low level layer gets information from environment and sends it to the high level layer that makes a high level decision and sends it to the low level layer. Low level layer then do an act in environment. When a miss-communication occurred, fixed rule helps the agent be updated. The architecture for the agents is therefore built in such a way that each agent can work independently. Team behavior then results from the combination of behaviors of individual agents. Three levels of abstraction can be distinguished in the description of an autonomous system which are each represented by their respective architecture:

- Functional Architecture. This architecture concerns itself with the functional behavior that the system should exhibit. It describes what the system should be capable of doing, independent of hardware constraints and environmental conditions.
- Operational Architecture. This architecture describes the way in which the desired behavior can be realized, taking into account the constraints on the system as well as the environment in which it has to operate. It assigns different pieces of the overall problem to different logical modules, which have the responsibility to solve their part within a certain time.
- Implementation Architecture. This architecture maps the operational aspects onto hardware and software modules and as such it gives a detailed description of the realization of the system.

Scenario and its role in cooperation with out communication

Scenario: It defines that which part of football ground each player can play, to whom he will pass the ball and the circle of his permission. Scenario is presented to the agent in the form of a file in each moment of game and the agents act according to it. Scenario specified each player's special rule in game. A player will know which action must be done in miss-communications. The scenario displayed in table 1. For example, the offensive scenario in right hand of arrangement is 3-5-2. Table 1 shows a scenario in which we have the following factors:

- Pass to: Whenever current player owns the ball and passes it to the other agent which its number.
- Can dribble: If the current agent can move with Ball.
- Dribble Range: The minimum distance with competitor when the agent is able to dribble.
- Max Y& Min Y, Max X & Min X: If agent who owns the ball should pass the current agent, he should move to the rectangle which its number is specified.

Attack Right 3-5-2	Agent number										
	1	2	3	4	5	6	7	8	9	10	11
pass to	2	5	4	2	6	11	9	10	0	11	0
Can Dribble	0	1	0	0	1	1	0	1	1	1	1
Dribble Range	0.0	3.0	0.0	0.0	2.0	2.0	0.0	1.0	0.5	0.5	0.5
Max X	0	32	-32	-32	-32	-32	15	25	-15	-15	-15
Max Y	0	32	-32	32	-32	-32	-15	25	-45	-25	15
Min X	-2	3	-3	1	-2	0	-2	-2	-2	0	1
MinY	-1	1	-3	2	-1	2	-4	3	2	3	1

Table 1

Table 1-offensive scenario in right hand of arrangement is 3-5-2 **State:** The first above left house of table indicates the scenario variety which is mentioned in the scenario "3-5-2 (Fast) Attack Right" which means fast anti attack from right side of 3-5-2 arrangement.

RESULTS AND DISCUSSIONS

In some cases breaking off the communication can pose some problems. For example, the possessor of the ball doesn't see behind his back. In such conditions one player is responsible for reporting the situation behind this player. We just added conditions for the team and results were unbelievably better than before. Agents use communication to increase the reliability of their world model and cooperation without communication helps them in miss-communication conditions. Intelligent agents will inevitably need to interact flexibly with other entities. The existence of conflicting goals will need to be handled by these automated agents, just as it is routinely handled by humans. The results in this paper and their extensions should be of use in the design of intelligent agents able to function successfully in the face of such conflict. The results of this paper is shown in table2. Accuracy means that the skills of a few percent of the direct pass is safe.

Skills	Without cooperation accuracy	With cooperation accuracy
Direct Pass	48.0%	59.9%
Shoot To Goal	67.0%	79.6%
Defense	45.7%	67.9%
Goal Keeper Skill	23.0%	54.0%
Attack	43.9%	78.0%
Robot Vision	29.6%	67.7%

Table 2

Results show that accuracy skills in the use of cooperation has increased as well. The results of running the code several times on the simulator is obtained. In order to record the results of the simulation we used the system log server. For example, skill level of the defense of our goals as a registered feature.

Future works

Results indicate that the developments. Our future work will focus on new methods. Our future work will focus on new methods. Although we are working to improve this method in the soccer simulator. The new method we will use the new features for cooperation between the agents. For example, the shared task can be a feature.

REFERENCES

1. Remco de Boer & Jelle Kok. The Incremental Development of a Synthetic Multi-Agent System: The UvA Trilearn 2001 Robotic Soccer Simulation Team. Master's thesis, university of Amsterdam, The Netherlands, Feb.2002.

2. 1. Michael R. Genesereth, Matthew L. Ginsberg, and Jeffrey S. Rosenschein.Cooperation without communication. Stanford University,1986.

3. P. Stone and M. Veloso. Communication in Domains with Unreliable, Single-Channel, Low- Bandwidth Communication. In Collective Robotics, pages 85-97, Berlin, July 1998. Springer-Verlag.

4. M .Shaw and D .Garlan. Formulations and Formalisms in Software Architecture. Lecture Notes in Computer Science, volume 1000, 1995.

5. D. Garlan and M. Shaw. An Introduction to Software Architecture. Advances in Software Engineering and Knowledge Engineering, volume 1, 1995.

6. Jeffrey S. R osenschein and Michael R. Genesereth, Deals among ration agents. IJCAI-85, pp. 91-99.

7. E. Foroughi, F. Heintz, S. Kapetanakis, K. Kostiadis, J. Kummeneje, I. Noda, O. Obst, P. Riley,and T. Ste®ens. RoboCup Soccer Server User Manual: for Soccer Server version 7.06 and later, 2001. At http://sourceforge.net/projects/sserver.