Proceedings of the International Conference on Man-Machine Systems (ICoMMS)
11 – 13 October 2009, Batu Ferringhi, Penang, MALAYSIA

How Does "Modular Hierarchical Architecture Concept" Help Us to Enhance Robot's Function?

M.Taheri [1], S.H.Mohades Kasaei[2], S.M.Mohades Kasaei

Institute of Robotics and Intelligent Systems,
Islamic Azad University, Khorasgan Branch,
Club, Isfahan 81595-158, Iran
M.Taheri@khuisf.ac.ir

***Abstract*- As the study of autonomous mobile robots grows in popularity in recent years, the demand for designing simple, suitable, and reliable robot has also increased. However, a survey through literature reveals little about the software design process or unified methodology to actually design proper mobile robots. The software is developed in two main parts, one is the server application which consists Network, World Modeling, Global AI, Condition Monitoring and the second is a player application which consists Image processing, Network, local AI, Trajectory Planning and a MIMO Motion Controller. This paper proposes formalism for the description and automated implementation of the functional modules of an autonomous mobile robot in the framework of multi-layered control architectures. The formalization of the description of a module, including its behavior and its interfaces with other modules makes the implementation easier by providing higher level tools and automatic consistence checking tools in the module compiler. The software architecture allows high level communication between modules on different abstraction levels of the control architecture within one robot system as well as communication between different and heterogeneous robots and computers using wireless network. Very different behavior control paradigms may be realized on the basis of the developed architecture. Our experience of developing control architectures for autonomous mobile robots has led us to more modular systems than existing ones. In the architecture, perceiving processes are separated from the robot's behavior hierarchy. Behaviors can be executed in parallel, and there is a behavior selector in every layer to cope with conflictions among behaviors. Our architecture preserves and extends the key properties of behavior-based systems.**

***Keywords*- Autonomous robot, Behavior decisions, Ccooperative multi-robot systems, Image processing, Mobile robot , Omni-directional system.**

I. INTRODUCTION

Cooperating systems of multiple autonomous robots are currently investigated in several scenarios, e.g. cooperative transport of a load by flying or driving robots, cooperative monitoring and surveillance in disaster areas or in the highly dynamic environment of autonomous robot soccer teams. Our project provides six robots. For cooperative scenarios the exchange of sensor data, world model information and behavior decisions is imperative for an effective robot team performance. One challenge in the development of mobile autonomous robots reacting to fast changing environments is the execution of rapid, goal oriented and situation aware motions considering motion stability and real time constraints. For the solution of the herein presented tasks wheeled robots are investigated. It is expected that robots with different degrees of autonomy and mobility will play an increasingly important role in all aspects of human life. To do this, robots must become much more complex than they are today in their hardware, software and mechanical structures. The drawbacks of increasing robot complexity may lead to low reliability and increasing size, weight, cost, power consumption, and motion limitation.

Our aim is to have a cooperative team behavior while attaining a high individual robot performance in image processing, decision making and motion control. Some studied subjects in this work are: "Artificial Intelligence includes Fuzzy Decision-Making and Neural Behaviors Learning based on Strategy concept, Image processing and high performance motor driver modules". Our robots are equipped with three-wheeled full omni directional navigation system and omni-vision system. The hardware of the robots will be introduced first and a scheme of the module-based software diagrams and an introduction to its different units are considered afterwards. Focusing on the most significant research issues, an overall description of the team is presented here as a soccer robots team. . A special scenario for soccer robots in such an environment is given in robot soccer in the RoboCup, an annually held world wide competition for different robot types. To meet these challenges suitable software architecture is required.

II. HARDWARE ARCHITECTURE

The robots' mechanics consist of a three-wheeled movement system (Figure 1). The wheeled movement system has three actuated wheels and each is driven by one DC motor. Having three independently controllable wheels, the robot can rotate or move in either direct or curved path. Using this system, robot can reach a maximum speed of about 2 meters per second. Each robot has three DC motors. The main board uses AVR microcontrollers as a digital processing unit that enables the robot to have some onboard processing. The program for the microcontrollers is written in AVR C language.

The omni-directional vision system consists of a hyperbolic mirror, a Color Digital FireWire Camera (Basler Digital Camera) and on each robot we have a standard laptop with 2GHz cpu running Windows. On this laptop the autonomous behavior of the robot is computed ,based on the data gathered from the omnidirectional camera.

Figure 1: A comprehensive omni directional robot having omni directional vision, and motion.

III. ROBOT SOFTWARE

A. *Image processing algorithm*

Utilizing the camera and the omni-vision system, each time the computer on each robot performs the processing of the current frame and calculates the position, direction and velocity of the robot. It also determines the position and velocity of the opponent robots as well as the position and velocity of the ball by pixels moving amount and using a calibrated distance map (Figure 2). The image-processing algorithm first filters the image by using a table for labeling the colors then recognizes the contiguous regions through either a BFS or a DFS search algorithm and finally extracts the positions by looking in the Image to ground map table. The implemented color extraction algorithm is based on lookup tables and the object detection in a radial model.The pixel classification is carried out using its color as an index into the table. The server gives the necessary commands to the image processing computers. The algorithm used to find objects is optimized to process the maximum number of frames. First it searches the pixels by swiping them with certain steps, when it finds a desired one and detects that object, saves its coordinates so the next time it can start back with the same point about. Sometimes for better image manipulation the RGB color space is converted to HLS (Hue, Saturation and Luminance). To recognize a certain color, a combination of conditions on Hue, Saturation and RGB is used. This procedure makes the color recognition independent from the change of brightness and other unpredicted conditions. We are trying to evaluate new methods to find some kinds of objects based on pattern recognition to reduce the effect of changing the colors on algorithm. The image processor receives its data through fire wire port connected to a Basler digital video camera with the speed of 20 to 30 frames per second [1].

Figure 2: Vision system in the running mode

B. *Network*

The network physical layer uses the shunted ring topology [7]. The UDP (User Datagram Protocol) network protocol is used for the software communication layer. The data flow of the network is as follows: A half field data (the data representing the position and status of the robots, opponents, goals and the ball) is transmitted to the server from each client computer of robots, the server combines them, constructs the complete global localization field then sends the appropriate data and commands (indicating which objects each robot should search for) back to the clients. When the data is completed it is passed to the AI unit for further processing and to decide the next behavior of the robots.

C. *Artificial Intelligence*

In this section the AI part of the software is briefly introduced. There are three distinct layers: AI Core, Role Engine and Behavior Engine. AI Core receives the computed field data from World Map Modeling unit and determines the play state according to the ball, opponents and our robots positions. Considering the current game strategy,

determination of the play state is done by fuzzy decision-making to avoid undesirable and sudden changes of roles or behaviors. Then AI Core sends a set of roles to Role Engine to be assigned to the robots. Because there are instances in which the image-processing unit cannot see the ball, a memory is implemented in the AI Core for the position of ball that specifies which robot owns the ball. Since there is a relationship between new roles and old roles, roles are changed in a manner that robots never experiment sudden changes in roles (for example the role never changes from defense to attack in next cycle). Role Engine receives a set of roles from AI Core and provides the Behavior Engine with a set of behaviors for robots. Twin or triple roles are implemented so that the robots really cooperate with each other to do their roles. Behavior Engine receives an array of behaviors from Role Engine and then according to the allocated behavior each robot must execute commands, with the help of Trajectory Unit, it sets the control inputs for the controller. It should be noted that the behaviors are learned offline so that they can provide a wide range of flexibility in a certain behavior (Figure 3) [2], [3], [4].

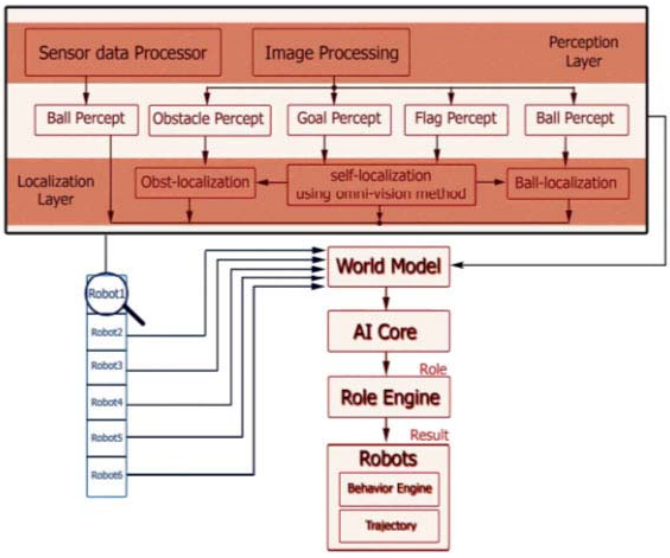

Figure 3: Artificial Intelligent Structure

D. *Trajectory*

Since the motion trajectory of each robot is divided into several median points that the robot should reach them one by one in a sequence the output obtained after the execution of AI will be a set of position and velocity vectors. So the task of the trajectory will be to guide the robots through the opponents to reach the destination. The routine used for this purpose is the potential field method (also an alternative new method is in progress which models the robot motion through opponents same as the flowing of a bulk of water through obstacles). In this method different electrical charges are assigned to our robots, opponents and the ball. Then by calculating the potential field of this system of charges a path will be suggested for the robot. At a higher level, predictions can be used to anticipate the position of the opponents and make better decisions in order to reach the desired vector (Figure 3). In order to there are many paths from the goal to any point, we define the potential to be the smallest possible potential, meaning the potential corresponding to the path with the least work (Figure 4) [4], [5].

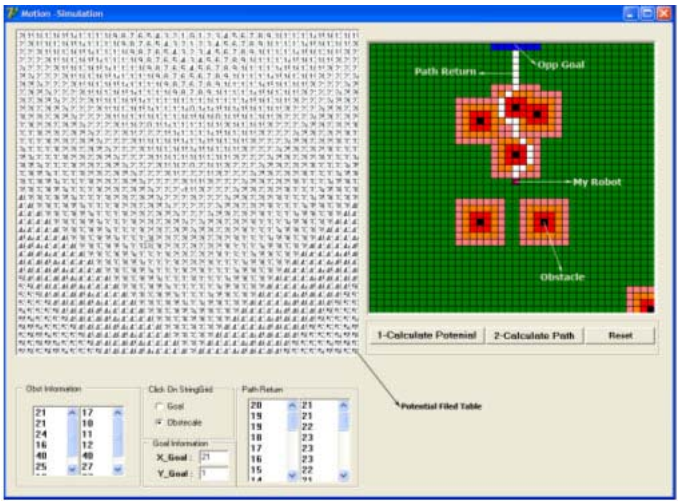

Figure 4: Trajectory Algorithm

E. *Controller*

So far we have reached a point where the trajectory has calculated a path in conjunction with the destination velocity vectors. If the robot moves on this path, it will reach the destination with no contact. Due to the non-symmetric nature of the wheeled movement system and different frictional forces in the wheels, we can't rely on exact operation of this open-loop system. Therefore, it's required to use a controller with a proper feedback to produce the correct control signals for the wheels of the robot in order to move accurately through the path. The movement system has three parameters that are the motor power (which is represented here by PWM) of the three wheels. Controlling the position of robot is done by using a PC-Based PI controller on velocity that obtains its feedback from three encoders located on the bisector lines of the motor line angles. The robot position is calculated via inverse kinematics method then the controller calculates necessary virtual forces (Fx, Fy) and moment (M) to reach the desired velocity vector and orientation. Finally in a forward dynamic algorithm the appropriate PWMs are obtained and applied to the motors. The algorithm was described here called Jacobean based MIMO controller which changes the controlled system variables (angular velocity of motors) into more visible ones (linear and angular velocity of the robot).

Then by utilizing a PI SISO controller and calculating the required values, it changes the outputs in a way to be applicable to the system .The controller always modifies the orientation and velocity of each robot to the value given by trajectory unit [6].

F. *Self-Localization*

We have developed and implemented three separate omni directional wheels coupled with shaft encoders placed 60 ° apart of the main driving wheels. Free shaft rotation and the flexible connection to the structure ensures minimum slippage and firm contact of these wheels to the ground, all these result in a great improvement in output precision. Having the shaft-encoder values, one can extract the robot position and orientation from Direct Method. This method eliminates the differentiation and integration operations which are necessary in Differential Method. Sudden faulty reports of the vision self-localization algorithm can be filtered out having the position and orientation report of the odometry system. The field lines are detected by applying thresholds to the image. To avoid detecting every white object, a line color (white) is detected only in the region of a field color (green). Using a calibrated distance map, the robot recognizes the distance to a field line in a real world. Self localization system is based on these distance data. In order to avoid the remaining cumulative error, odometry system parameters can be initialized every time the vision could calculate the position reliably (Figure 5) [8].

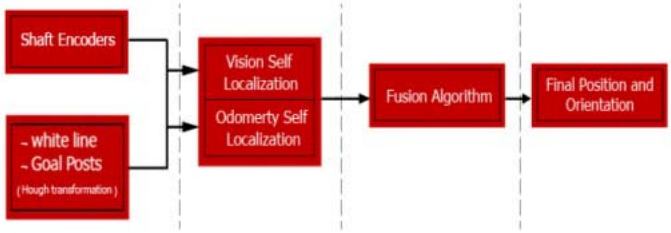

Figure 5: The diagram above shows the overall mechanism of the self-localization system.

IV. Conclusion

In this paper cooperative behavior, Condition Monitoring, world modeling for an autonomous omni-directional mobile robot is presented. The main trust of our research has been put into developing reliable and fast soccer robots for environmental dynamic monitoring. Combination of odometry and vision led to a more accurate and reliable self-localization algorithm. The performance of our robots team in Robocup competitions showed that the combination of methods and techniques described in this paper are led to a successful autonomous robot and soccer player team (Figure 6). The basis for our researches was the robust and reliable hardware design, the well-structured software architecture and efficient algorithms for sensor fusion and behavior generation.

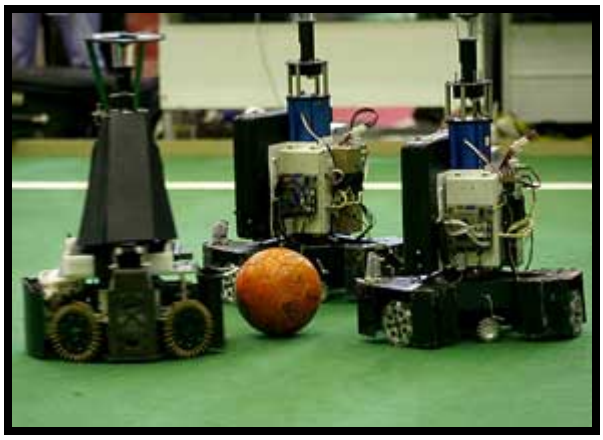

Figure 6: Our autonomous mobile robots in robocup competition.

References

[1] Ripley, B.D. Pattern Recognition And Neural Networks. Cambridge University Press, 1996.
[2] S.J.Russell, P.Norvig. Artificial Intelligence: A Modern Approach. Prentice Hall, 2nd Ed 2002.
[3] R.R.Murphy. An Introduction To Ai Robotics. Mit Press, 2000.
[4] C H.J.Zimmermann, H.J.Zimmerman. Fuzzy Set Theory And Its Applications. Kluwer Academic Publishers, 4th Ed. 2001.
[5] G.Weiss. Multiagent Systems. A Modern Approach To Distributed Artificial Intelligence. Mit Press, 2000.Keigo Watnable, "Control Of An Omnidirectional Mobile Robot", Second International Conference Acknowledge-Based Intelligent Electronic Systems,21-23 April
[6] Groth, David; Toby Skandier (2005). Network+ Study Guide, Fourth Edition'. Sybex, Inc.. Isbn 0-7821-4406-3.
[7] Fredric Chenavier,James L. Crowley: "Position Estimation For A Mobile Robot Using Vision And Odometry", Proceeding Of Ieee International Conference On Robotics And Automation.