Development an Omni Directional Walking Based on Neural Network for Biped Robot

S.Hamidreza Mohades Kasaei, S.Mohammadreza Mohades Kasaei and S.Alireza Mohades Kasaei

***Abstract*—This paper presents an overview of our 3D soccer simulation robot . This paper describes the software design of the 3D soccer simulation robot systems of the ADRO Team in 2010. We have tried to focus on areas such as software architected, robot perception unit, robot controller, robot AI and behavior learning. This year, our developments for the 3D soccer simulation include: (1) the design and construction of our new Robot Artificial Intelligent (AI) Architected (2) the design and construction of a new software controller to be used in our robots. The project is described in two main parts: Software and Robot AI. The software is developed a robot application which consists walking controller, autonomous motion robot, self localization base on vision and AI part consist of Local AI and Global AI, Trajectory Planning, Motion Controller. The. Each agent is able to walk, fast walk, pass, kick and dribble when it catches the ball. The project is still in progress and some new interesting methods are described in the current report.**

***Index Terms*—3D Soccer Simulation, Recurrent Neural Networks, Control systems, Robocup.**

I. Introduction

The RoboCup scenario of soccer playing legged robots represents an extraordinary challenge for the design, control and stability of autonomous bipedal and quadrupedal robots.

Naturally robotic soccer is an interactive and complex procedure. It might be so idealistic, but some consider a challenge with a real human football team in 2050, as the final goal of robotic soccer. The robots in 3D soccer simulation league can only use local sensors and local vision. Each team can have a maximum number of six robots. They can communicate with each other via a virtual link. The rules in the competition are the same as the international soccer rules as far as they are practical for robots [1]. So for a 3D soccer simulation team, many technology issues and scientific areas must be integrated, such as control and computer science, Besides, the research technologies of humanoid walking control, autonomous motion, vision and AI system, kicking and shooting ball will be applied [1-10].

The ADRO 3D soccer simulation team was formed in 2007. In 2008 we ranked 1st place Humanoid Kid Size Soccer Robot League in 1st national Khwarizmi Robotic Competitions the Khwarizmi Robotic is one of the, Iranian major Robotic event. In 2009 we achieved the 4th in the 4th International Iran-Open Competitions in the 3-3 games out of 15 teams. The basis for our success was the robust and reliable hardware design, well-structured software architecture and efficient algorithms for sensor fusion and behavior generation. Our main research interest is both, the development of learning robots and the development of improved sensor fusion and sensor integration techniques. In order to let the robot can autonomous play a soccer game, three basic skills are designed and implemented on 3D soccer simulation: image understanding for environment perception, move ability, and artificial intelligence. In order to let the agent have a high ability of environmental detection, a camera and array of sensor are equipped on the body of the implemented robot to obtain the information of the environment to decide an appropriate action. Many functions are implemented on this system so that it can receive the vision signal obtained by Soccer Simulation Server and process the data obtained by gyroscopes. It also can process the high level artificial intelligence, such as the navigation. The 3D agent is designed as a soccer player so that the implemented robot can walk, turn, stand up, shift left and right, backward walk and shoot the ball.

In this paper, we will at first describe the general software design of the ADRO Team, (section 2) and after that focus on our scientific approaches in sensor fusion and robot neural network controller, (section 3). In section4 we will describe the Robot AI and Finally, section 5 concludes this paper. This document describes the current state of the project as well as the intended development for the RoboCup 2010 competition.

Fig. 1. ADRO3D Socer Simulation Team

II. Robot Software

In soccer game, for example, the robot searches a ball and two goals, and moves to its desired location with avoiding many obstacles, therefore we developed our humanoid Agent software in Visual C++ and robot software is divided into four sections: Image perception, artificial intelligent, behavior engine, and motion control (actuation).

Manuscript received February 28, 2011; revised September 28, 2011.

The authors are with Young Researchers Club, Isfahan Branch, Islamic Azad University, Isfahan, Iran.

Figure 2 shows the block diagram of the software which runs in the robot's main processor. The program consists of 4 main blocks:

Image perception: Contains image processing algorithms such as recognition of landmarks and other object. Self localization is done using particle filtering. Particles are scored by comparing a simulated image from each particle with the current frame captured by camera. Using "Sampling-Importance Resampling" method, a new distribution of the particles is created after each step. Particles are also updated using a motion model. Final distribution of the particles converges to the real pose of the robot.

Planning: Planning system of the robot is based on a multi layer, and multi thread structure. The layers are named Strategy, Role, Behavior and Motion. Each layer contains a Scenario which runs in parallel with the scenarios in the other layers. A scenario in a higher level can terminate and change the scenario running in the lower level; however it is usually done in synchronization with the lower level scenario to avoid conflicts and instabilities. (Such as stopping the walking motion while one of the feet is still in the air).

Motion Control: manages all the actuators of the robot, and controls locomotion or any other action of the robot according to the requests from Cognition.

Sensor Control: manages other sensors, and interacts with the Sub-Controller.

III. Motion Control Via Artificial Neural Network

The neural network technique is a very effective tool for controlling complex non-linear systems when we have no complete model information, or even when considering a controlled plant as a black box. The use of a proposed recurrent hybrid neural network to control of walking robot is investigated in this section. The reason to use hybrid layer is that robot's dynamics consists of linear and non-linear parts. ADRO soccer simulation team has two kinds of locomotion pattern: Omni directional walking and special actions such as kicking and getup. Special actions are described using key frames, which can be edited very fast by our software tool ***ADRO Control***. Omni directional walking means the robot can walk in every direction with variable step length. The behavior module determines the target position and orientation according to the results of localization and the sensor measurements, and then constructs an action series which consists of the elementary gaits to realize Omni directional walking. A neural networks based control system is utilized to the control of walking robot. The control system consists of two proposed neural controllers, two standard PD controllers and two legged planar. The proposed neural network (NN) is employed as an inverse controller of the robot. The NN has three layers, which are input, hidden and output layers. In addition to feed forward connections from the input layer to the hidden layer and from the hidden layer to the output layer, there is also feedback connection from the output layer to the hidden layer and from the hidden layer to itself.

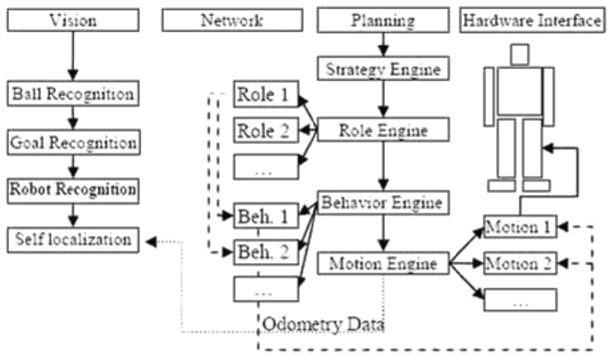

Fig. 2. Structure of the control software

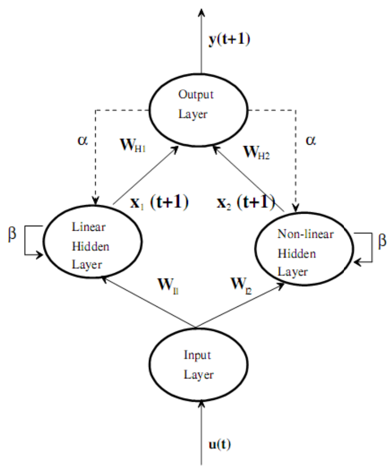

Fig. 3. Schematic representation of the proposed three layered recurrent hybrid network

At a given discrete time t, let $U(t)$ be the input to a recurrent hybrid network, $Y(t)$, the output of the network, $X_1(t)$ the output of the linear part of the hidden layer and $X_2(t)$ the output of the non-linear part of the hidden layer. The equation of the feed forward hybrid network is as follows:

$$X_1(t+1) = W_I * U(t) + \beta X_1(t) + \alpha JY(t)$$
$$X_2(t+1) = F\{W_I * U(t) + \beta X_2(t) + \alpha JY(t)\}$$
$$Y(t+1) = W_{H1} * X_1(t+1) + W_{H2} X_2(t+1)$$

If only linear activation function is adopted for the hidden neurons, the above equations simplify to:

$$X(t+1) = X_1(t+1) = X_2(t+1)$$
$$Y(t+1) = W_H * X(t+1) \Rightarrow Y(t) = W_H * X(t)$$
$$X(t+1) = W_I * U(t) + S_1 * X(t) + S_2 Y(t)$$

Replacing $Y(t)$ by $W_H * X(t)$, gives:

$$X(t+1) = (S_1 + S_2 * W_H) X(t) + W_I * U(t)$$

where W_I is the matrix of weights of connections between the input layer and the hidden layer, W_{H1} is the matrix of weights of connections between the linear hidden layer and the output layer, W_{H2} is the matrix of weights of connections between the non-linear hidden layer and the output layer, $F\{\ \}$ is the activation function of neurons in the non-linear hidden layer. J is, $n_H * n_O$ matrices with all elements equal to1, where n_H is the numbers of hidden neurons, and n_O, the number of output neurons. The weights in S_1 and S_2 are related to the feedback connections and are fixed. The weights of the connections from the output layer to the hidden layer have the same value α and those of the connections from the hidden layer to itself have the same value β. S_1 and S_2 are then given by $S_1 = \beta I$ and $S_2 = \alpha J$, Where I is the $n_H * n_H$ identity matrix.

$$\Rightarrow X(t+1) = (\beta I + \alpha J W_H) * X(t) + W_I * U(t)$$

General form of the equation can be written as follows:

$$X(t+1) = AX(t) + BU(t)$$

where A is a $n_H * n_H$ matrix and B is a $n_H * n_O$ matrix.

The back propagation (BP) algorithm is used to update weights of the proposed neural network. The BP algorithm is a method of supervised neural network learning. During training, the network is presented with a large number of input patterns. The experimental outputs are then compared to the neural network output nodes. The error between the experimental and neural network response is used to update the weights of the network inter-connections. This update is performed after each pattern presentation. One run through the entire pattern set is termed an epoch. The training process continues for multiple epochs, until a satisfactorily small error is produced. The BP algorithm is the most commonly used to update the weights of the neural networks. Beginning with an initial (possibly random) weight assignment for three-layer feedforward network, proceed as flow:

Step1: Present i^p *, and form outputs,* o_i *, of all unit in network.*

Step2: Use the below equation to update Δw_{ji} *between the hidden layer and the output layer.*

$$\Delta w_{ji} = -\eta * \frac{\partial E_{pe}(t)}{\partial w_{ji}(t)} + \alpha \Delta w_{ji}(t-1)$$

Step3: Use the below equation to update weights between input layer and the hidden layer.

$$\Delta w_{ji} = -\eta * \frac{\partial E_d(t)}{\partial w_{ji}(t)} + \alpha \Delta w_{ji}(t-1)$$

Step4: Stop if updates are insignificant or the error is below a preselect threshold, otherwise return to Step1.

where η is the learning rate, and α is the momentum term. $E_{pe}(t)$ is the propagation error between hidden layer and output layer. $E_d(t)$ is the error between desired and neural network output signals. Also as depicted in Fig.3, the first layer received reference inputs from the trajectory generator and also the control errors. Again, the second layer consisted of two parts, a linear part and a non-linear part. Finally, the third layer of the network produced outputs to drive the robot joints.

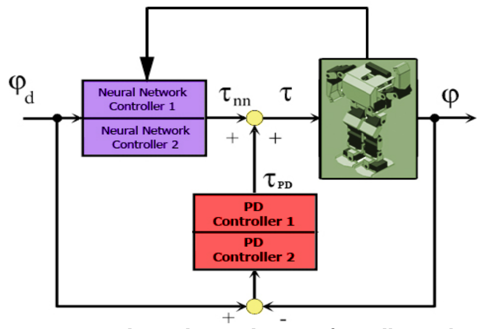

Fig.4. Proposed neural control system for walking robot.

The image sequences of forward walking shown in Figure 7 respectively.

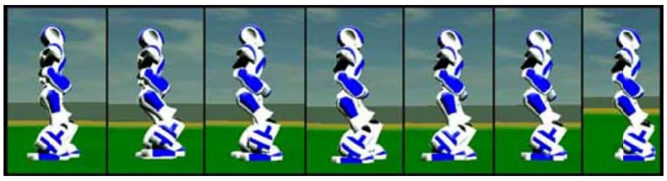

Fig.5. Forward walking image sequence

IV. Image Perception

One of the primary tasks for the Image understanding is to locate a particular goal and calculate the robot's position in relation to it. Utilizing a sufficient algorithm, each time our program performs the processing of the current frame and calculates the position and direction of the robot. It also determines the position and position of the opponent robots as well as the position of the ball. The image-processing algorithm extracts the necessary information like: self-localization, ball, and goal and opponent robot position by looking in the frame.

V. Artificial Intelligent

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.[3-4]

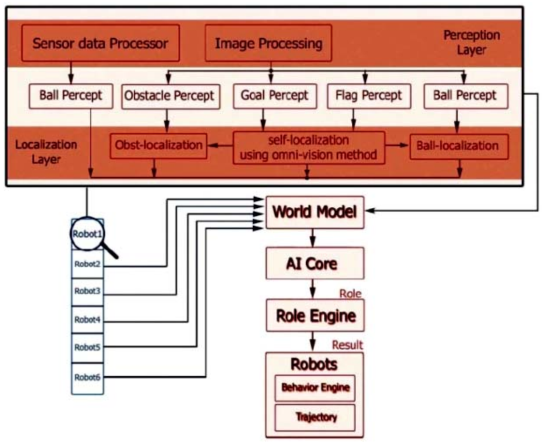

Fig. 6. World model construction and artificial intelligent structure running on the PDA

VI. Role Engine

Role engine module receives information from AI core; process them, and then selected a role. This module is the main section of robot software. The proposed rules for role engine have been turned by various experiences and they are independent of game field conditions. The output of this module is a set of high level commands that send to Behavior engine. Some of high level commands which are produced by role engine module are:

- Go to position
- Go to ball
- Targeting
- Shooting ball
- …

VII. Behaviors Engine

This module receives information from Artificial Intelligent unit. Total functions about Robot Behavior such as stability motors actions, robot path planning, turn camera, walking, shooting, dribbling; motion and etc are controlled in this section.

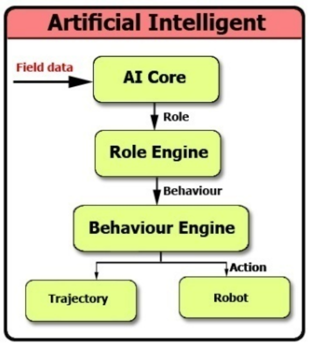

Fig. 7. Artificial Intelligence Algorithm

VIII. 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)[9][12]. In this method different electrical charges are assigned to opponents, goal 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. In our path planning algorithm, an artificial potential field is set up in the space; that is, each point in the space is assigned a scalar value. The value at the goal point is set to be 0 and the value of the potential at all other points is positive. The potential at each point has two contributions: a goal force that causes the potential to increase with path distance from the goal, and an obstacle force that increases in inverse proportion to the distance to the nearest obstacle boundary. In other words, the potential is lowest at the goal, large at points far from the goal, and large at points next to obstacles.

$$U(q) = U_{Goal}(q) + U_{Obstacle}(q)$$

If the potential is suitably defined, then if a robot starts at any point in the space and always moves in the direction of the steepest negative potential slope, then the robot will move towards the goal while avoiding obstacles. The numerical potential field path planner is guaranteed to produce a path even if the start or goal is placed in an obstacle.

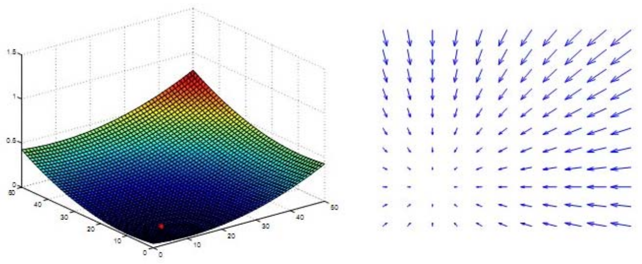
Fig. 8. Goal force (Attractive potential to the goal)

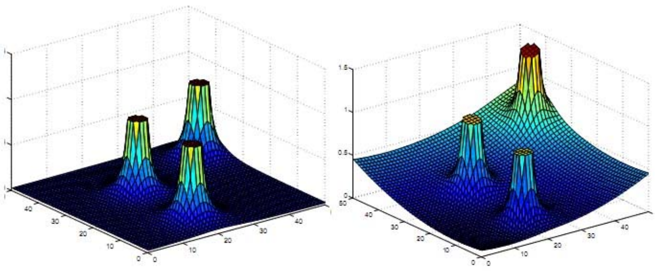
Fig. 9. Obstacle force (Repulsive potential) and Goal Force+ Obstacle force

If there is no possible way to get from the start to the goal without passing through an obstacle then the path planner will generate a path through the obstacle, although if there is any alternative then the path will do that instead. For this reason it is important to make sure that there is some possible path, although there are ways around this restriction such as returning an error if the potential at the start point is too high. The path is found by moving to the neigh boring square with the lowest potential, starting at any point in the space and stopping when the goal is reached.

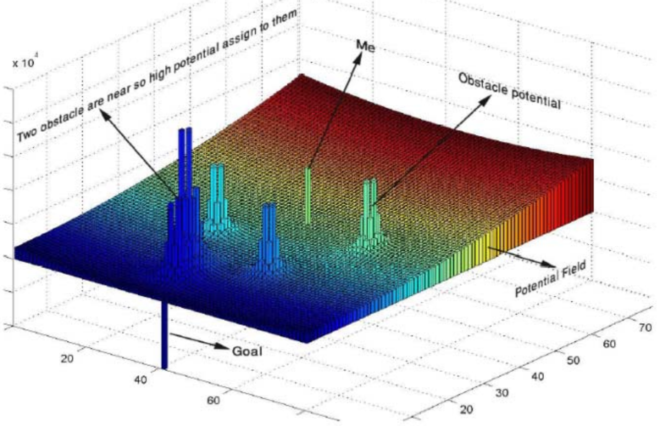

Fig. 10 Potential at every point; it is highest in the obstacles and lowest at the goal. Elsewhere it is generally higher farther from the goal and near obstacles.

IX. Conclusion and Future Work

In this paper, we present the details of design and implemented method of our 3D soccer simulation team. In our robot, Omni directional walking, image understanding and a novel motion controller based on artificial neural network and PD controller have been combined to create a comprehensive Humanoid robot. Our robots can get up from a fall, walk forward and backward, turn right and left, and kick the ball. Our agents can obtain the environmental information to decide the action behavior. We try to making the robot more stable and reliable as the result of our researches. Future plans are to develop and implement autonomous team actions towards participation in the six-by-six games similar to other leagues in RoboCup. Further information's and video about our works are presented on our website *http://www.IranAdro.com.*

REFERENCES

[1] Wong, C.C., Cheng, C.T., Wang, H.Y., Li, S.A., Huang, K.H., Wan, S.C., Yang, Y.T., Hsu, C.L., Wang, Y.T., Jhou, S.D., Chan, H.M., Huang, J.C., Hu, Y.Y.: Description of TKUPaPaGo Team for Humanoid League of RoboCup 2005. RoboCup InternationalSymposium 2005 (2005).

[2] Zhou, C., Jagannathan, K.: Adaptive Network Based Fuzzy of a Dynamic Biped Walking Control Robot. IEEE Int. Conf. on Robotics and Automation. 4 (2000) 3829-3834.

[3] Pauk, J. H., Chung, H.: ZMP Compensation by On-Line Trajectory Generation for Biped Robots. IEEE International Conference on Systems, Man, and Cybernetics. 4 (1999) 960-965.

[4] Huang, Q., Li, K., Nakamura, Y.: Humanoid Walk Control with Feedforward Dynamic Pattern and Feedback Sensory Reflection. IEEE International Symposium on Computational intelligence in Robotics and Automation. (2001) 29-34.

[5] Sugihara, T., Nakamura, Y., Inoue, H.: Real time Humanoid Motion Generation through ZMP Manipulation based on Inverted Pendulum Control. IEEE International Conference on Robotics and Automation. 2 (2002) 1404-1409.

[6] Furuta, T.: Design and Construction of a Series of Compact Humanoid Robots and Development of Biped Walk Control Strategies. IEEE International Conference on Robotics and Autonomous Systems. (2001) 65-88.

[7] Vukobratovic, M., Frank, A. A., Juricic, D.: On the Stability of Biped Locomotion. IEEE Trans. Bio-Med. Eng. 17 (1970) 25-36.

[8] Golliday, C. L., Jr., Hemami, H.: An Approach Analyzing Biped Locomotion Dynamics and Designing Robot Locomotion Control. IEEE Trans. Aut. Contr. (1977) 963-972.

[9] Miyazaki, F., Arimoto, S.: A Control Theoretic Study on Dynamical Biped Locomotion.ASME J. Dyna., Syst., Meas., Contr. 102 (1980) 233-239.

[10] Furusho, J., Masubuchi, M.: A Theoretically Motivated Reduced Order Model for the Control of Dynamic Biped Locomotion. ASME J. Dyna., Syst., Meas., Contr. 109 (1987) 155-163.*D. Equations*

[11] Khatib, O. (1985). Real-time obstacle avoidance for manipulators and mobile robots. In Proc. of the IEEE Intl. Conf. on Robotics and Automation, pages 500–505, St. Louis, Missouri.

[12] Lee, & Park, "Trajectory Generation and Motion Tracking for the Robot Soccer Game," Proceedings of the 1999 IEEE International Conference on Intelligent Robots and Systems, pg 1149-1154, 1999.