Proceedings of the
2007 IEEE Intelligent Vehicles Symposium
Istanbul, Turkey, June 13-15, 2007

Bondgraph Robot Soccer Simulation for Minimum Time Attacker Maneuvers

M.A Jarrah, and F.M Al-Saleem

***Abstract*- Dynamic model of a differential drive soccer player using bondgraph method is presented. Minimum time maneuvers for the attacker behavior based on the model derived using bondgraph methodology are studied. The paper presents a new cooperative attacking behavior for two agents in a real time robot soccer game. This behavior consists of four hierarchy levels, the intelligent level, path planning level, path following and velocity controller level. This paper reports the Simulation and experimental results of applying those hierarchy levels on the AUS robot soccer team.**

I. Introduction

Robot soccer is an active area of research and it is the place where new intelligent strategies are developed and examined. Examples of technologies utilized are image processing, control theory, artificial intelligence, multi-agent systems, and motion planning and embedded systems [1]. Much has been written about solving the problem of motion planning of a differential drive robot generally and a robot soccer player specifically, but unfortunately, much of these studies try to solve this problem just using the kinematics model of the mobile robot. Limited studies use dynamic models of the mobile robot [2]. Dynamic model derivation of a differential drive robot using the Lagrange formalism was obtained [2]. Although the kinematics model of a differential drive robot captures the nonholonomy property, the problem of neglecting the dynamic effects cannot be justified at high linear velocities [3]. Stable full-state tracking problem is investigated for nonholonomic mobile robot based on its internal dynamics model [4].

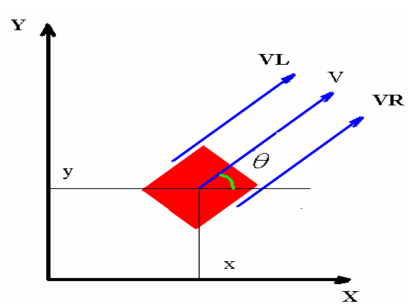

Fig. 1: Robot Kinematics model

A kinematics model of a two wheeled mobile robot with non-slipping wheels is shown below [5].

M. A. Jarrah is with the American University of Sharjah, Sharjah, UAE (corresponding author: 00971-6-5152464; fax: 00971-6-5152979; e-mail: mjarrah@aus.edu).
F.M Al-Saleem., was a graduate Student at the American University of Sharjah, Sharjah, UAE . (e-mail: b00016000@aus.edu).

$$v = \frac{V_L + V_R}{2} \qquad \omega = \frac{V_L - V_R}{L} \tag{1}$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{2}$$

Where V_L and V_R are the left and right wheel velocities correspondingly.

Equation 2 has three variables to be controlled, but only two inputs $[\omega \quad v]$. This explains why, in general, no control is guaranteed to move the robot from a given posture (x, y, θ) to desired posture (x_d, y_d, θ_d) [6].

II. Bondgraph Modeling

Bondgraph modeling originally developed in the late 1950s offers a compact and sound methodology for modeling multi-domain dynamic systems. The figure below shows the symbol of a bondgraph.

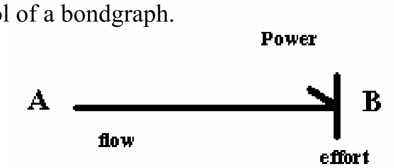

Fig. 2: Bondgraph symbol

This symbol represents the dynamics interaction between two components of a dynamic system. It has three components; the line that joins element A to element B, the half arrow that indicates that power flows either from A to B or from B to A and finally the third component is the causal stroke, a perpendicular small line either at end A or at end B. This stroke indicates which power variable is the input or the output to A or to B. More details can be found in reference [7]. The bondgraph model of a DC motor that the robot has is shown below (Figure 3). In this model R is either the motor resistance or the friction, Se is the controlled motor input voltage, GY is the gyrator converting current to torque, I is either the motor inductance or mechanical inertia.

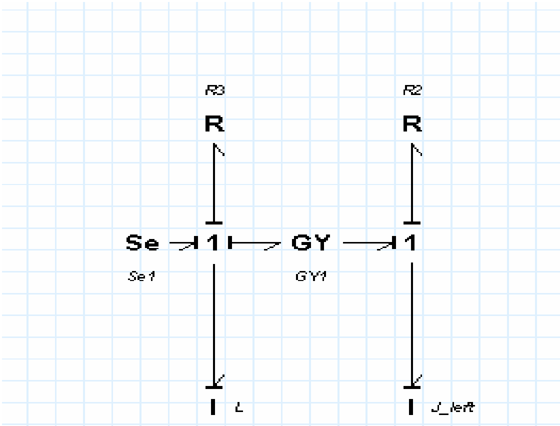

Fig. 3: Dc Motor Bondgraph model

A. *Robot Bondgraph Based Model*

Figure 4 shows the complete bondgraph model of the differential drive robot. The responses of such a model given different input values for V_L and V_R are shown in Figures 5.

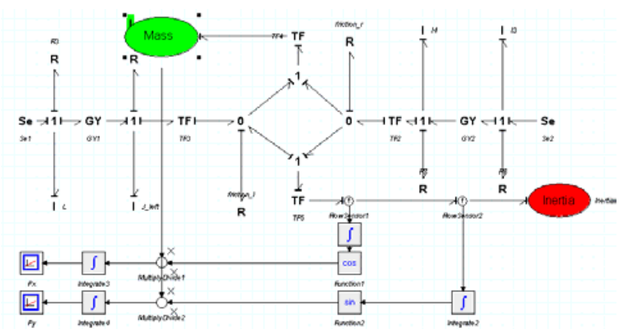

Fig. 4: Differential drive robot Bond Graph Model

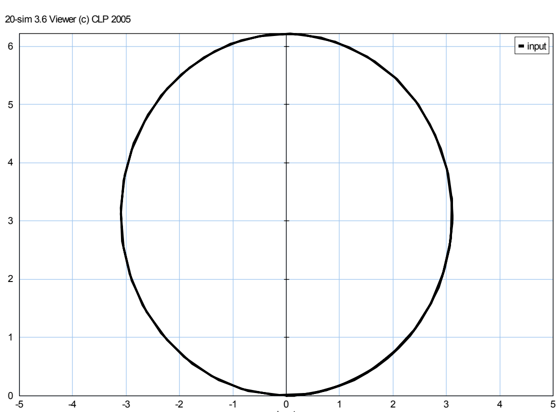

Fig. 5: Mobile robot Bond Graph response at V_L =2V, V_R = 6V

B. *Differential drive robot State space equations*

Starting from the left to the right, each bond of the robot model is assigned its flow and efforts values, (Figure .6)

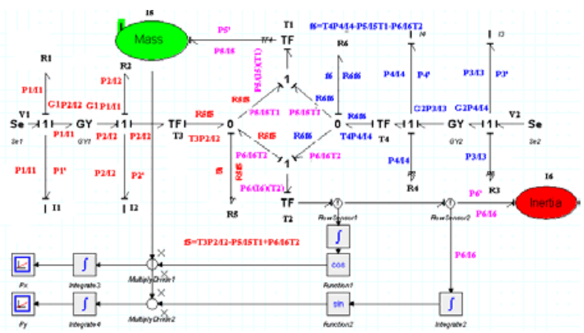

Fig. 6: Bondgraph, state space model derivation and nomenclature.

Now the state equations that govern the behavior of the differential drive robot are:

$$\begin{pmatrix} \dot{P}_1 \\ \dot{P}_2 \\ \dot{P}_3 \\ \dot{P}_4 \\ \dot{P}_5 \\ \dot{P}_6 \end{pmatrix} = \begin{bmatrix} -\frac{R_1}{I_1} & \frac{G_1}{I_2} & 0 & 0 & 0 & 0 \\ \frac{G_1}{I_1} & -\frac{R_2}{I_2}-\frac{R_5}{I_2} & 0 & 0 & \frac{R_5}{I_5T_3T_1} & -\frac{R_5}{I_6T_3T_2} \\ 0 & 0 & -\frac{R_3}{I_3} & -\frac{G_2}{I_4} & 0 & 0 \\ 0 & 0 & \frac{G_2}{I_3} & -\frac{R_4}{I_4}-\frac{R_6}{I_4} & \frac{R_6}{T_4I_5T_1} & \frac{R_6}{T_4I_6T_2} \\ 0 & \frac{R_5}{T_1I_2} & 0 & \frac{R_6}{T_1I_4} & -\frac{R_6}{T_1^2T_4I_5}-\frac{R_5}{T_1^2T_3I_5} & -\frac{R_6}{T_1T_6T_4I_6}+\frac{R_5}{T_1T_2T_3I_6} \\ 0 & -\frac{R_5}{T_2I_2} & 0 & \frac{R_6}{T_2I_4} & -\frac{R_6}{T_1T_2T_4I_5}+\frac{R_5}{T_1T_2T_3I_5} & -\frac{R_6}{T_2^2T_4I_6}-\frac{R_5}{T_2^2T_3I_6} \end{bmatrix} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{pmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$$

Where P_i is the momentum associated with i[th] state variable: current and angular velocity in the left motor, current and angular velocity in the right motor, robot speed, and robot angular velocity correspondingly.

where I_1, I_3 left and right motor inductances, I_2, I_4 are left and right motors inertias, G_1, G_2 are left and right motors gyrator values. I_5, I_6 are robot mass and robot mass moment of inertia about the vertical. T_3, T_4 are left and right motor gear ratios, R_1, R_3 are left and right motor armature resistances. $R_2, R_4, R_5, and\, R_6$ are friction coefficients.

III. Motion Planning Strategies:

Three motion-planning levels for the attacker behavior of the two-robot-soccer players are considered. The top level is where decisions are taken to select the suitable robot to perform the desired behavior. The second level in the control architecture is responsible for deriving the optimum way to perform the selected behavior (in this paper the attacker behavior). This is done by implementing a motion-planning controller to ensure that the robot follows the path suggested by the second level. Moreover, the attacker behavior is divided into two different cases; attacking while the ball is stationary and attacking whiles the ball is moving.

The design of the attacker behavior of the robots is structured as shown in Figure 7. The system is designed to receive an image of the playground every 32 ms, processes it, extract the posture information of the robots and the ball using an image processing code written using the visual C++, and send all these information to the strategy code. In this code the attacker behavior is implemented. To test the developed attacker behavior, two modes of this behavior are presented, one in case of stationary ball, the other is in case of a moving ball. Each of these behaviors consists of three parts, the supervisory controller in which the role of each robot is selected; the other two parts take care of generating a path and following that path from the current positions of the robot to the position of the ball with a desired direction of arrival.

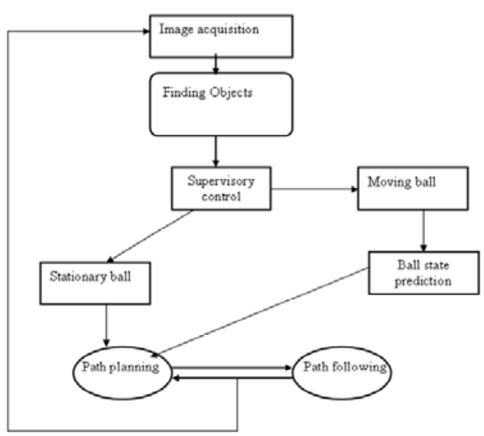

Fig. 7: Attacking behavior stages

Supervisory control

supervisory controller which determine the role of each of the two robots is designed based on Finite State Machine and Neural network method, the inputs to these controllers will be the orientation of the two robots, the distance between them and the ball, the distances between them and any obstacle (if any). In the case of the state machine these inputs will fire the transition between the suggested states, while in case of the neural network; these inputs will pass through the neurons, to produce the desired outputs.

Path planning and path following controller:

A hybrid controller that takes the advantages of the controllers discussed above in the literature is designed.

Low level velocity controller:

The functionality of the low-level velocity controller is to provide the velocity command to each of the right and left wheels. Desired velocity command to the controller is calculated based on the path parameters. The low level velocity controller is implemented using the Motorola 68HC12 microcontroller.

Moving and stationary ball scenarios:

The attacker behavior for a stationary ball will be used to formulate the attacker behavior for the case in which the ball is moving with some modifications. These modifications are required from the fact that it is really impossible to design a path controller for the robot that depend only on the current position of a moving ball in a real time match. To solve this problem the prediction of the future states of the ball (position, velocity) must be used. After the prediction of the future states of the ball, two alternative modifications to the above stages were suggested, see the figure below.

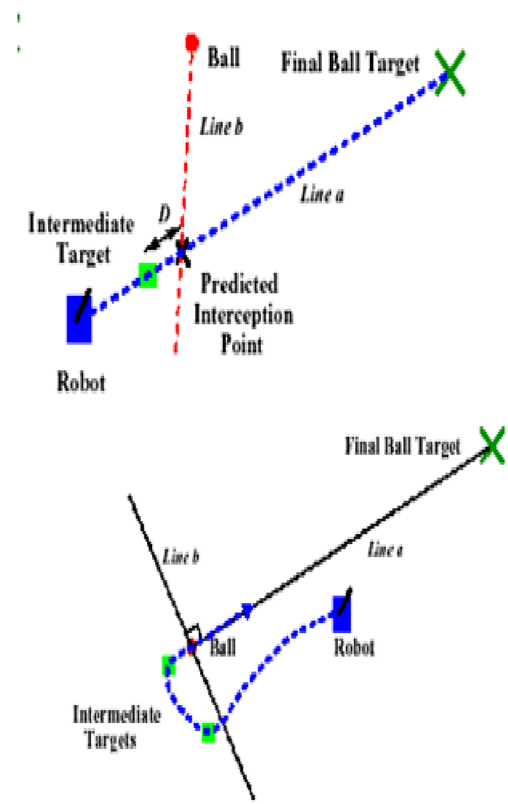

Fig. 8: Two alternatives to shoot a moving ball

IV. PATH PLANNING AND FOLLOWING RESULTS

This part presents the simulation and experimental results to shoot a stationary ball to the goal area, on our AUS robot soccer player. It is important to say that, for the implementation of all methods, the same formulas for the linear and angular velocity were proposed (Figure 6)

$$V = constant$$
$$\omega = K_p(\angle\theta_{des} - \theta_{robot})$$

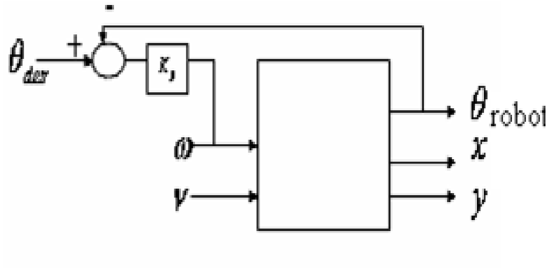

Fig. 9: Block diagram of the Path following implementation

where V is the linear velocity (constant value), ω is the angular velocity, θ_{des} is the desired angle that is calculated in the path following according to the path planning method**,** θ_{robot} is the actual angle of the robot estimated by the vision system, and K_p is the proportional control gain.

Determining the K_p value is not a straight forward because K_p is function of the linear velocity V . The minimum turn radius (r), that the robot can turn without slipping depends on V and it is given by:

$$r = \frac{mV^2}{F_{cen}}$$

where F_{cen} is the centrifugal force which is measured by the lateral accelerometer.

On the other hand, for a given value (V) the K_p gain will work fine until error in the robot orientation $\theta_e = \theta_{des} - \theta_{robot}$ exceeds certain threshold. This is significant when (V) value becomes high. For example Figure 10, shows a step change demand in robot orientation response of the robot for 40° and 80° when v =100 cm/s and K_p=**.5**.

It can be seen that when the reference input is 40° the system is stable, but when the reference input is 80° the system becomes unstable. On the other hand the upper part of Figure 10 shows the step response of the robot angle for different step demands when v =0 cm/s and K_p=**.5**. Extensive studies of the controller analysis are presented [3]. The lower part of figure 10 shows the controller performance using potential field method path planning.

A. Methods drawback

Three different methods to shoot a ball toward a given direction were simulated on Matlab before they were implemented on AUS robot soccer players using visual C++ language programming; it was shown that running all these methods at high speed will cause problems [5]. Unless one considers the dynamic behavior of the robot, on the implementation, increased speed will be a problem. On the other hand, it was clear that the vector field method was the better method in term of optimizing the path between the robot and the ball, and in term of dripping the ball toward the center of the goal.

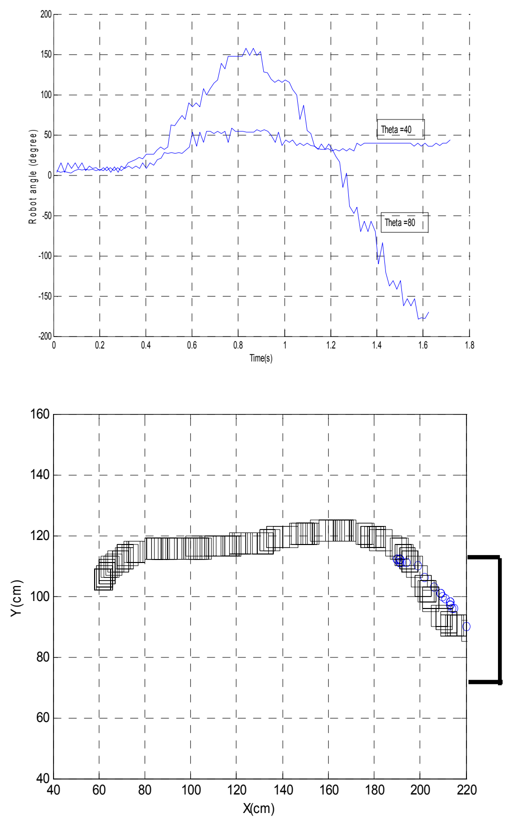

Fig. 10: step response of the robot angle for 40° and 8 when V =1 m/s , K_p=.5.

B. Low Level velocity control results

The dynamic controller development is implemented using the dSpace real time rapid development system. Once the controller is developed using this system, it is transformed to the real soccer player using the 68HC12 microcontroller. .

The dSpace rapid prototyping system was used to implement the controller and verify its operation before implementing on the real robot embedded controller. The blue line is the commanded signal while the red line is the dSpace rapid prototyping system output

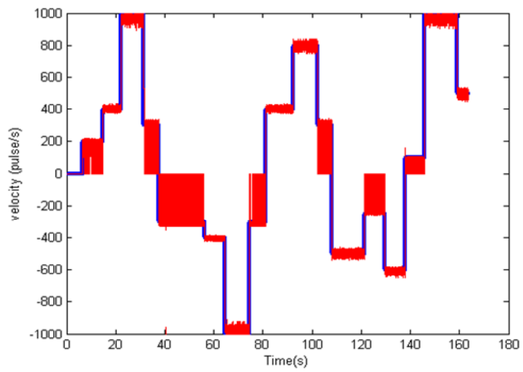

Fig. 11: Low level velocity controller dSpace results

2-Microcontroller

Figure 12 shows the results the of the implanted low level velocity controller using the robot Microcontroller. It shows that the implementation is on the 16 bit system is having similar performance to the dSpace system.

C. Dynamic and Kinematics Model Results

a- Kinematics Model Results

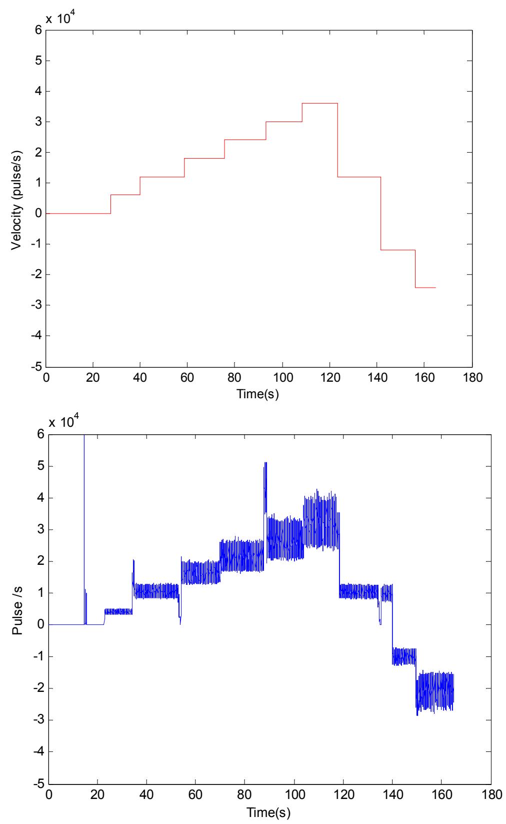

Fig. 12: Low level velocity controller HC12 results

A kinematics' model of the robot player was built in Matlab Simulink. To test the model different inputs where applied.

Figure 13 below show the simulation results of applying v_l=1m/s and v_r=0 m/s.

b- Dynamic Model Results using Bond graph

Kinematics models become invalid when high speed and angular rates are involved for both the robot and the ball models. Centripetal accelerations as well as linear and angular accelerations violate the no slip condition taken for granted in the kinematics models

Kinematics models become invalid when high speed and angular rates are involved for both the robot and the ball models. Centripetal accelerations as well as linear and angular accelerations violate the no slip condition taken for granted in the kinematics models.

To simplify the control task, angular as well as linear accelerations in both x and y directions are measured. This allowed control of the robot at high speed compared with the kinematics controller alone.

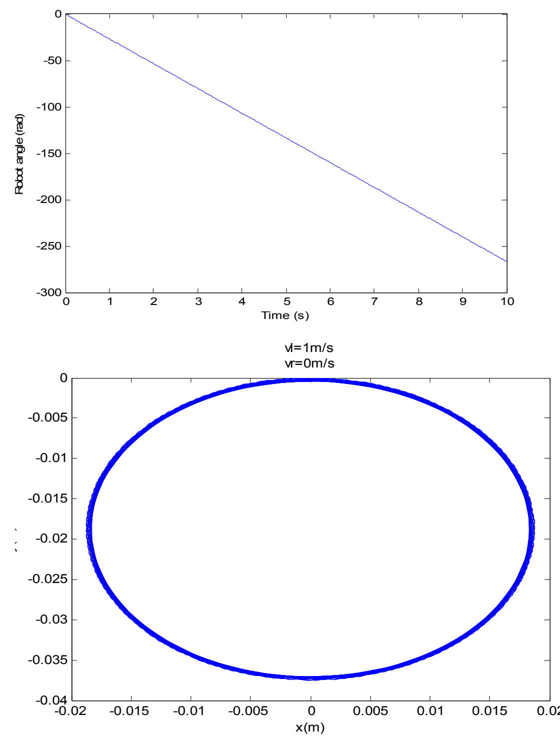

Fig. 13: Kinematics simulation results (v_l=1m/s and v_r=0 m/s)

Two prediction strategies were developed and implemented in the controller to intercept a moving ball. The first is based on the least square method and the second is based on Kalman filter to fuse vision system measurements with the predictions of the dynamic model of the robot ball interception algorithm. Figure 14 shows the prediction ability of the ball position using Kalman filter. The blue line is the true ball position. The lower part of the figure shows when the ball hits the wall at x =0.

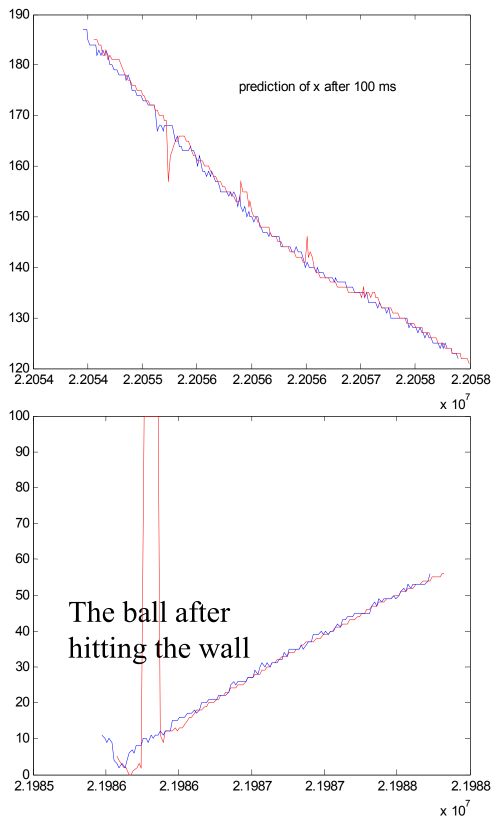

Fig. 14: Predection of a moving ball positin

V. Conclusions

Boundgraphs method is used to derive a full dynamic model of the robot soccer player. Bondgraphs methodology has the ability to represents the internal interactions of the system dynamics of the robot player. Comparison between dynamic and kinematic model based control shows the real advantage of the dynamic model based controller when the speed is increased for the robot soccer players. Prediction ability is enhanced by the controller with the acceleration feedback.

References

[1] Werner. D. Dierssen, "Motion Planning in a Robot Soccer System," Master's thesis, University of Twente, 2003

[2] R. Fierro and F. L. Lewis, "Control of a Nonholonomic Mobile Robot Using Neural Networks," IEEE Transactions on Neural Networks, Vol.9, 1998, Pp 589-600.

[3] F.M Al-Saleem, and M.A Al-Jarrah, "Evaluation and comparison study of shooting ball algorithms," AUS International Symposium on Mechatronics (AUS-ISM05), 2005.

[4] DanweiWang and GuangyanXu, "Full-state tracking and internal dynamics of nonholonomic wheeled mobile robots," Mechatronics, IEEE/ASME, Vol. 8, pp.203- 214, 2003.

[5] Peter Stone and Manuela Veloso, "Towards Collaborative and Adversarial Learning: A Case Study in Robotic Soccer," Pittsburgh, International Journal of Human Computer Studies, 1998.

[6] Kim, J.-H., Kim, D.-H., Kim, Y.-J., and Seow, K.-T, "Soccer Robotics," Springer, 2004.

[7] Karnopp, D. C., Margolis, D. L., and R. C.Rosenberg, "System Dynamics. Modeling and Simulation of Mechatronics Systems," Wiley and Sons pp. 307-324, 2000,