ZJUBase 3D - Team Description 2006

JIANG Hao, DU Xinfeng, LUO Dijun,
ZHANG Yifeng and ZHOU Yang

National Laboratory of Industrial Control Technology
Zhejiang University, China

jianghao@iipc.zju.edu.cn

February 12, 2006

Abstract

We have accumulated rich experiences in over a hundred years of human soccer development, which are useful and valuable for robotic soccer research. Our recent goal is to analyze the knowledge of human soccer and convert it into robotic techniques. These techniques should never rely on any concrete platform but fit the most environments, even in the other league of RoboCup. In this paper, we introduce our progress in this kind of agent soccer techniques, mainly about 1) a pass evaluation method and 2) a dynamic positioning strategy. These two problems correspond the two parts of agent decision, decision with and/or without the ball. In addition, a mathematical method of Bayesian estimation for agent self-localization is described. At last, we take a short introduction to our off-line debugging system, which can be visualized graphically, based on a server/client architecture.

1 Introduction and Previous Work

In the year 2004, our research focused on the world model maintenance and some basic skills of a soccer agent. In our last TDP paper [1], we have taken an introduction to our progress on 1) filtering the vision signals and estimating the velocity with a least square method and 2) a ball-interception skill based on a nonlinear optimization model.

Last year, we proceeded our research on these issues and achieved some more goals. An approximately linear optimization model and a neural network are imported to improve the interception performance. Furthermore, a Kalman Filtering algorithm [2] is employed for the world model maintenance, combined with the least square method. However, we no longer discuss them here but focus on the high-level agent decision.

In chapter 2 and 3, we describe the agent decision problem on two aspects, with and/or without the ball, i.e. pass evaluation and dynamic positioning respectively; and also our solutions, which are both studied from human knowledge/experience on soccer. In addition, a self-localization method and a GUI off-line debugging system are introduced in chapter 4. At last, in chapter 5 we take a short discussion and a glance at our future work.

2 Decision with the Ball: Pass Evaluation

In this chapter we discuss the decision problem while the ball is in the agent's control (i.e. the agent intercepts the ball faster than any other agent does). In our experience of human soccer game, a player who controls the ball usually has three choices: 1) shooting on the goal, 2) dribbling to a destination and 3) passing the ball to a teammate. In our opinion, passing the ball is the essential aspect of soccer. In fact, the other two behaviors can be also reduced into the passing: 1) shooting can be thought as passing the ball to an infinite destination behind the goal; and 2) dribbling can be disposed as passing to itself.

Even if the agent has already decided to pass the ball, two problems are required to solve: 1) where to pass and 2) how to pass the expected destination. The latter is an elemental skill of the agent which should not be classified as a decision. Here we only discuss the former one. Yet the pass evaluation is the most essential thing to decide where to pass.

The pass evaluation is such a problem: given a number of positions in field as candidates, to calculate a valuation for each position. After this, immediately the agent could know what to do next. To lead the description more explicitly here, we assume all the agents in fields are stationary, i.e. with a zero velocity. In our common knowledge about human soccer, the major influence on a pass point comes from two aspects, 1) the relative position to the two goals and 2) the nearest players in both team. According to this, we propose such a pass evaluation formula,

$$V(P) = \kappa_1 e^{-\alpha_1 d_1} - \kappa_2 e^{-\alpha_2 d_2} + \frac{\kappa_3}{|P - P_{goal}| + 1} \tag{1}$$

The $V(P)$ stands for the valuation of the pass position P. d_1 and d_2 are the distances between the ball and the nearest players in our team and in the opponent team, respectively. P_{goal} represents the position of opponents' goal. κ_1, κ_2 and κ_3 are parameters, which can be test and set manually or automatically by machine learning.

Here we choose an e^x function, since we notice that if the player stays too far away from the ball, his influence must be ignored, and the e^x function fits this characteristic very well. However, the formula should be considered as only an outline of the pass evaluation. We expect to point out these two factors through it. A cutline of (1)'s effect is shown in Figure 1.

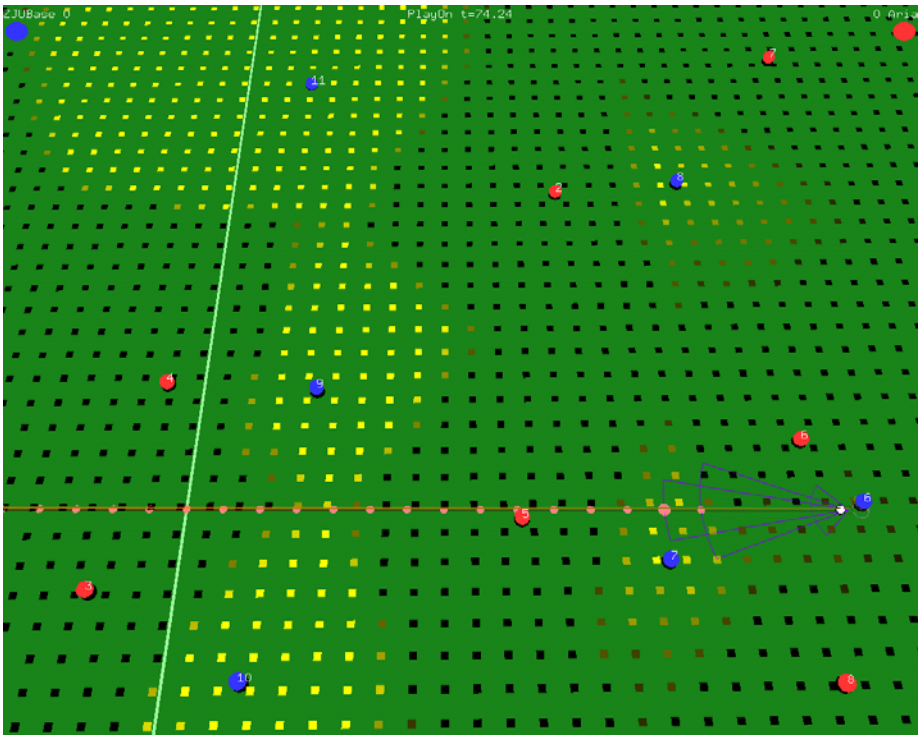

Figure 1: The effect of the pass evaluation formula. The colored small points on the green field illustrate the pass points as candidates. The color of a pass point represents its numerical valuation. A brighter color denotes a higher valuation. Own side is the blue team and the opponents are in red.

3 Decision without the Ball: Dynamic Positioning

In most time during a soccer game, a player doesn't keep the ball. At these moments, the player needs to stay(or run to) at a good position to wait for pass or defend. At a certain moment, the positioning destination of an agent is called its strategic position. Which strategic position to choose and how to drive there is the problem we need to discuss. Generally, the valuation of a strategic position relies on the ball position currently and the distribution of other players in field. Here we only discuss our methodology depending on the ball position.

We use an approach named NURB curves to describe the strategic movement of players at a certain moment. In our strategy, we define a parametric function using the ball position to calculate the player's position. We can express the function mathematically like this,

$$Q(u) = \frac{\sum B_i \omega_i N_{i,k}(u)}{\sum \omega_i N_{i,k}(u)} \tag{2}$$

The B_i are the projections of the four-dimensional control points and the ω_i are their weights. The $N_{i,k}(u)$ is defined like this,

$$N_{i,0}(u) = \begin{cases} 1 & t_i \leqslant u < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$N_{i,k}(u) = \frac{(u - t_i)N_{i,k-1}(u)}{t_{i+k} - t_i} + \frac{(t_{i+k+1} - u)N_{i+1,k-1}(u)}{t_{i+k+1} - t_{i+1}} \tag{4}$$

Here t_i is the conventional notation for the i_{th} knot in the knots vector and k is the order of the curve. In our functions these parameters is determinate. The three equations are based on a NURB curve. To learn more about the NURB curves, please refer to some related books or papers, such as [3].

Here an important motivation for employing the NURB curves is the ability to control smoothness and the convenience. For example, we place some spots B_i arbitrarily as control points, and draw the curve with a NURB function, as shown in Figure 2(a). Then in Figure 2(b), we move the point B_7 and the curvature changes. So in order to get the required curve we only need to place and adjust the control points. This feature enables us to build a graphical editor and get much easier to adopt the positioning strategy.

4 Other Aspects

4.1 Self-Localization

At the very beginning of world model maintenance, the agent needs to locate itself. Particularly in the 3D simulation environment, the agent is able to do

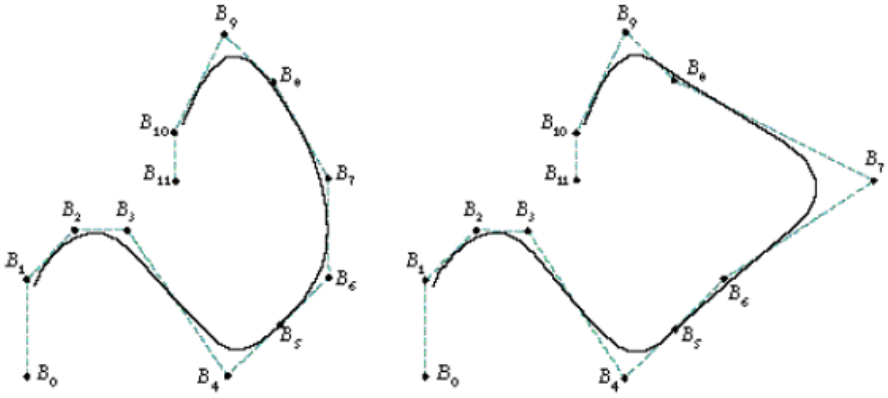

Figure 2: (a) A NURB curve. (b) A NURB curve with a control point B_7 moved.

it by its vision sensor, which apperceives the relative positions of the field corners and goals. It's easy to get the self-position if we calculate directly from the relative flags and use their average. However, we expect the noise distribution parameters (which are already known) can be fully utilized, accordingly to achieve a better localization. Therefore, we employ a Bayesian estimation.

Bayesian principle is a basic knowledge even for every undergraduate, so we no longer repeat it here. Our idea comes from [4], which introduces a universal methodology for fitting parametric and nonparametric models to noisy data. Obviously, the noise of our 3D simulation platform is a parametric model and all the parameters are known (from the user manual or the source code). According to Bayesian principle, we can get a formula as below,

$$Pr(M|D) = \prod Pr(p_i|M) \tag{5}$$

where M stand for the real model, and D is the noisy data set. p_i is a single data of agent position in the set D and $Pr(x|y)$ represents the probability of event x under the condition y.

In (5), $Pr(p_i|M)$ is an expression of variable p^*, where p^* is the estimated position. In order to improve the accuracy of p^*, we need to make $Pr(p_i|M)$ as large as possible. Again the maximal value of $Pr(M|D)$ can be solved by a nonlinear optimization method. The algorithm is already referred in our last team description paper [1].

4.2 Off-line Debugging System

Off-line debugging system is an important part for realtime robotic research and its development is a meaningful job. As the simulator requires a monitor to visualize the simulation status, the debugging system also calls for the visu-

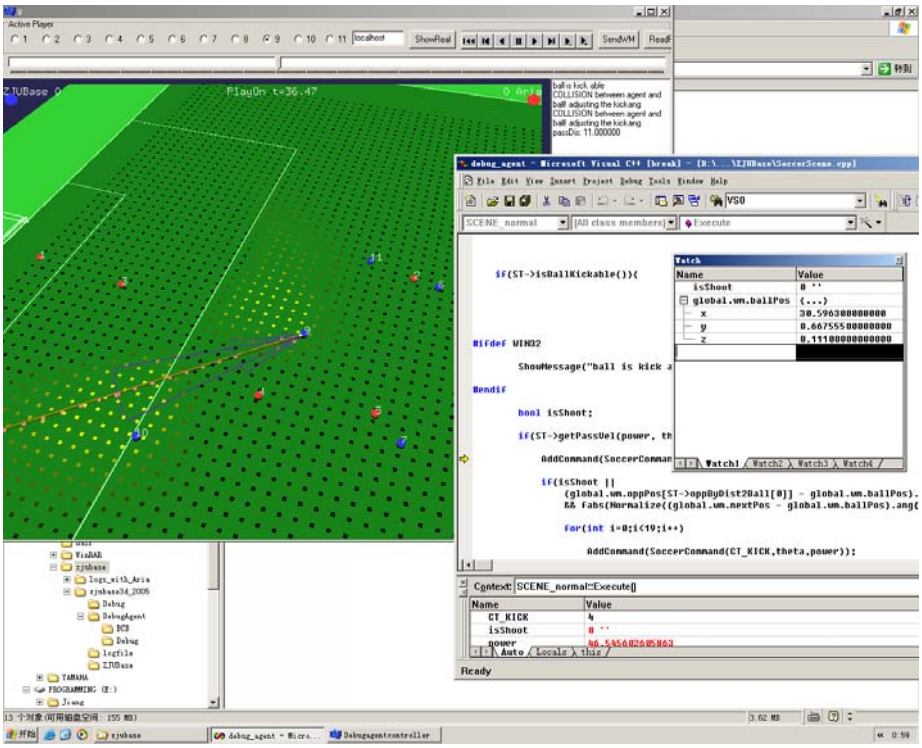

Figure 3: A screen shot of our off-line debugger

alization of its data. A graphical visualization can greatly help us understand the data, particularly in our 3D simulation.

In order to make the debugging data visualized all the time, the debugging system is divided into two parts. They communicate through a socket connection. The server side loads the online log-file and broadcasts the vision message to all the clients. A debugging client contains a copy of the online agent, which repeats its online decision after it receives the message from the debugging server. A graphical monitor is integrated into the server side and display the current status.

Furthermore, the client can send a drawing request to the server and the monitor will drawing the requested things on screen. This function is very helpful for debuggers to analyze the current situation. The debugging monitor is also able to employ as a normal monitor for connecting the simulator, including sending a specific world model to the simulator for a particular purpose.

5 Conclusion and Future Work

We have briefly discussed the agent decision problem in the former chapters. In our recent experiments, we found that the pass-evaluation formula cannot work properly in some cases. It reminds us more factors should be imported from human knowledge and the teamwork should be considered. For the strategic positioning, we are still working on how to build another solution to take other agents' positions into consideration. In RoboCup Symposium last year, there is a paper [5] which introduced a dynamic positioning based on Voronoi cells, which may be helpful to us on this problem.

The off-line debugger greatly helps us in development. We introduce it more thoroughly in our 3D development team description paper. Since the new legged agent model has come out, we plan to build a stronger version that supports customized agent models in future.

6 References

[1] LUO Dijun, JIANG Hao, GUAN Jun, ZHANG Yifeng, ZHOU Yang. **ZJUBase 3D 2005 Team Description**. In *Proceedings CD RoboCup 2005*, Osaka, Japan, July 2005.

[2] R. E. Kalman. **A New Approach to Linear Filtering and Prediction Problems**. In *Transactions of the ASME - Journal of Basic Engineering*, 82 (D): 35-45, 1960.

[3] Philip J. Schneider. **NURB Curves: A Guide for the Uninitiated**. In *Develop, the Apple Technical Journal*, Issue 25, June 1996.

[4] Michael Werman and Daniel Keren. **A Bayesian Method for Fitting Parametric and Nonparametric Models to Noisy Data**. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, Vol. 23, No. 5, May 2001.

[5] H. Dashti, N. Aghaeepour, S. Asadi, M. Bastani, Z. Delafkar, F. Disfani, S. Ghaderi, S. Kamali, S. Pashami and A. Siahpirani. **Dynamic Positioning based on Voronoi Cells (DPVC)**. In *The 9th RoboCup International Symposium*, Osaka, Japan, July 2005.