ZJUNlict: RoboCup 2014 Small Size League Champion

Chuan Li, Rong Xiong(✉), Zeyu Ren, Wenjian Tang, and Yue Zhao

National Laboratory of Industrial Control Technology,
Zhejiang University, Hangzhou, People's Republic of China
rxiong@iipc.zju.edu.cn
http://www.nlict.zju.edu.cn/ssl/WelcomePage.html

Abstract. The Small Size League is one of the important events in RoboCup Soccer. ZJUNlict got the first place in Robocup 2014. In this paper, we introduce the improvement we have made in the past year. We describe the overview of the mechanical design, show the design of the protector for Infrared emission tube as well as the shield for wheels. Simulation is given to show how our design works. Then the lower level firmware architecture is illustrated. The dynamics analysis of the robot is presented to help improving the robots' performance and reducing motion deviation in y axis. Finally we present how we organize defense employing Close-Marking defense along with Zone defense imitating human player.

1 Introduction

The Small Size League is one of the important events in RoboCup Soccer. It is basically a game between two robot teams restricted to rules similar to human soccer game. Each team consists of six robots and competes to goal more than the opponent. The league is devoted to the advancement of mechanical design, artificial intelligence and multi-agent cooperation of mobile robots with the distant expectation that robot will play a game with the FIFA champion in 2050.

ZJUNlict from Zhejiang University has participated in this League for ten years since 2004. We received our first championship in RoboCup 2013, Netherland and won the championship again in RoboCup 2014, Brazil. Although our robot is of one of the best performance among the teams in Small Size League equipped with our flexible and powerful strategies for both attack and defense, many problems and defects still exist for us to deal with after Robocup 2013. Firstly, during the game play, our robot is very likely to be damaged and needs carefully check and fixing after each game, especially for the wheels. Secondly, the Soft Core architecture needs to be reformed as the old design is difficult to maintain and introduces many bugs caused by the nested reference. Thirdly, there is obvious movement deviation in y axis of our robot, which influences movement accuracy. Finally, the defense should be strengthened because the attack organized by other teams becomes more threatening.

R.A.C. Bianchi et al. (Eds.): RoboCup 2014, LNAI 8992, pp. 47–59, 2015.
DOI: 10.1007/978-3-319-18615-3_4

The remainder of this paper is organized as follows. Section 2 introduces the mechanical design of our robot with detailed introduction of the protector and shield we design to make the robot more stable. Section 3 describes the firmware architecture. Section 4 analyzes the dynamics of the robot and addresses the cause of the deviation in y axis direction. Section 5 describes the defense strategy, which consists of Close-Marking defense and Zone defense. Section 6 concludes the paper.

2 Mechanical Design

2.1 Overview

As shown in Fig. 1, The robot is equipped with four omni-wheels, a dribbling device, a shooting device, and a chipping device.

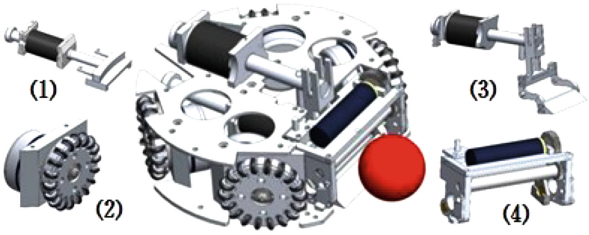


Fig. 1. Mechanical design of the robot: (1) shooting device; (2) omni-directional wheel; (3) chipping device; (4) dribbling device

Each omni-wheel is composed of 16 distributed little wheel with o-ring rubber and an aluminous base wheel. The little wheel rotates freely along the direction perpendicular to the rotation axis of the base wheel so that the robot can carry out omnidirectional movement. Each base wheel is driven by a 50 watt brushless Maxon motors with a gearbox, of which the reduction ratio is 3.18:1.

The shooting device and the chipping device are driven by two solenoid respectively, of which the parameters are calculated accurately in advance. The electromagnet is charged by two big capacitors. The time and the force of kicking the ball can be controlled by specifying the capacitor voltage as well as the discharge time.

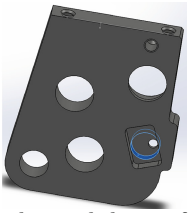
The dribbling device is designed to control the ball and not losing it. The core component is a stick swathed with a special pipe circum, which rotates the ball to keep the ball in the dribbling device. The material of the pipe has significant effects on the robot's ability to control the ball, so we have chosen a special kind of rubber after a series of experiment and long term tests.

2.2 Mechanical Structure Adjustment

Infrared Emission Tube Protector. In RoboCup 2013, we have found out that the infrared emission tube and the rubber rings on Omni-direction wheels of our robots easily got damaged due to mechanical collision with other robots in the game. We had to frequently replace these components, which are both cumbersome and not environmentally friendly. So this year in 2014, we have made some special structure adjustments to solve these problems and obtained remarkable results.

As shown in Fig. 2, the cylindrical inner wall which painted blue is the exactly place to install infrared emission tube. In order to ensure the ball detected once it runs into the dribbling device, the infrared emission tube is mounted in the front of the device within a small box. When crashed, the deformation of the box is likely to squeeze the tube. In our new design, a protector is being installed in front of the infrared emission tube. Impact force will disperse through the protector to the whole mechanical structure rather than concentrate in the front so as to reduce the deformation.

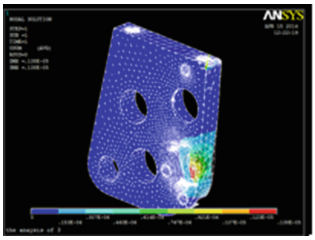
(a) Old mechanical design for infrared



(b) New mechanical design for infrared

Fig. 2. Mechanical parts (Color figure online)

We use the software Ansys to compare the vector displacement of the two mechanical structure [1]. The result is shown in Fig. 3. The vector displacement declines to $7.785 * 10^{-6}$ from $7.738 * 10^{-5}$.



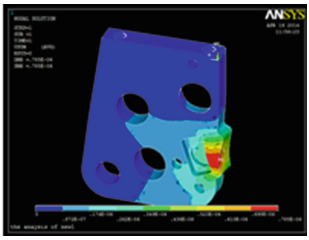
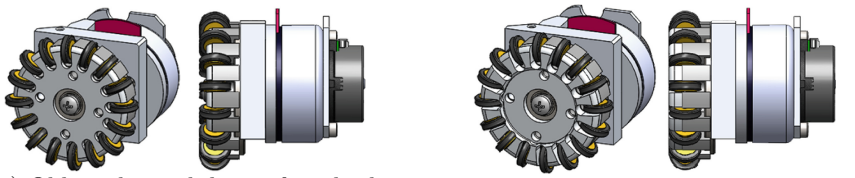


Fig. 3. The result of analysis about the protector

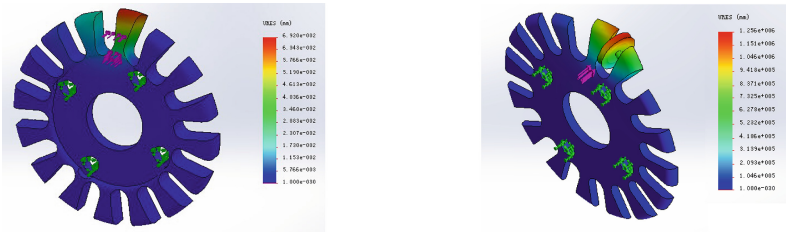
Omni-Wheel Shield. Another adjustment happens on the rubber rings of the omni-wheels. We increase the thickness of the big wheel flaps.The adjustment does not change the basic functions of the omni-wheel. Due to the increased thickness, the wheels will not be subjected to the direct mechanical impact. This adjustment received a remarkable effect in RoboCup 2014. The structures of the old version and the new version are shown in Fig. 4(a) and Fig. 4(b) respectively.



(a) Old mechanical design for wheel (b) New mechanical design for wheel

Fig. 4. Wheel design

We have verified the effect of the proposed improvement using the SimulationExpress of Solidworks. Assume the collision speed between the robots is $3\,m/s$, the mass of the robot is $3\,kg$ and the collision time is $0.05\,s$, and after collision the robot will immediately stop. According to the theorem of momentum, the collision force is $180\,N$, which is the extreme case. Because of the protection of the big wheel flaps, the rubber wheels will not subject to the impacts any more. So we just analyze the situation that the impact happens on the big wheel flaps. Figure 5(a) shows the effect of the force applied in the simulation as well as the deformation of the wheel component. The material of the component is aluminum 7075, with the elastic modulus $7.2 * 10^{10}$ and the poissons ratio 0.33. From the Figure, we can conclude that the maximum deformation is $6.92 * 10^{-2}$ mm, which is so small that it can be neglected. Therefore the new design achieves a great effect in protecting the rubber rings of Omni-direction wheels.



(a) simulation of the new wheel (b) simulation of the old wheel

Fig. 5. Wheel simulation result

We have also made an analysis to the old mechanical structure. The little wheels are subjected to the direct mechanical impact so we add one to analyze.

And we have brought pressure to the rubber rings of omni-wheels and analysis under rated pressure what is the theoretically deformation of the rubber rings. Similarly Fig. 5(b) shows the constraints and force added to the part as well as the deformation of the rubber. The material is normal rubber with elastic modulus 10000 and the poissons ratio 0.45. The deformation of the rubber is $1.256 * 10^6$ mm, which means that the rubber will be definitely broken, and in real situation the deformation of the rubber can not be achieved at all. The theoretically result matches the actual phenomenon.

3 Firmware Design

3.1 Overview

We choose Altera Cyclone III as our central processor unit. The firmware in the chip is divided into Hard Core and Soft Core. The former defines the basic electronic resource(e.g. IOs, Timer, UART) available for the program written in Verilog and with SOPC Builder. The latter runs the main function written in C based on processor defined by Hard Core.

The Hard Core flowchart is shown in Fig. 6. The main part of the Hard Core consists of standard module, including Nios processor for Soft Core, Uart for chip communication, PIO for signal's input and output and so on. Other unique user modules featured with special communication protocols and control method are written in verilog, such as "motorcontrol" for BL motor control, which is of the greatest importance among our modules. The virtual pin of all this module is assigned to the physical pin of the Cyclone III chip.

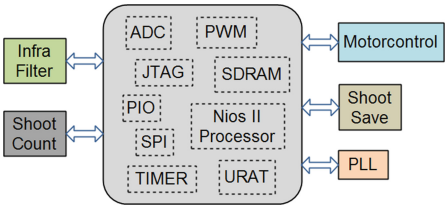


Fig. 6. Hard Core structure

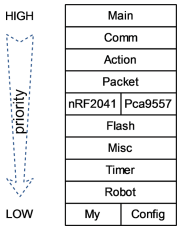


Fig. 7. Soft Core flowchart

The function of the Soft Core is communication and motion execution [4]. Our previous Soft Core is not well designed and leaves with a lot of nested references among the c-files and h-files. As the robot's function extends, the code becomes more complicated and difficult to debug and maintain. This year, we reorganize the overall Soft Core architecture and specifies different priorities for different modules, which is shown as Fig. 7. As illustrated, modules on the same level has the same priority in execution. Each module has its specific function and priority. It is recognizable that Main has the highest priority and My has the lowest priority as well as Config. Functions in a module can only call functions in modules with the same or lower priority level.

4 Dynamics Analysis

Our robot is one with best performance in Robocup Small Size League. It can chip and shoot stably and execute smooth motion according to the visual feedback and the control strategy. However, when without vision feedback, it cannot go straight along y axis while go well along x axis. This is a common problem in Small Size League, so we decide to analyze the dynamics of the four-wheels driven robot to find out possible solution [2].

The robot's coordinate system (X,Y,θ) and the field coordinate system (X_L, Y_L,θ_L) are defined as shown in Fig. 8 and the positive rotation is counter-clockwise. So the coordinate-transformation matrix R_θ is shown as the Eq. (1). For each wheel, there is a similar coordinate system as Fig. 9 and a coordinate-transformation matrix as Eq. (2).

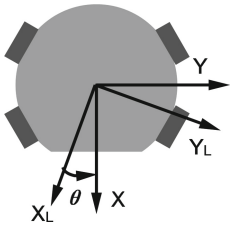


Fig. 8. Robot and field coordinate system

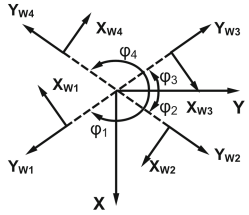


Fig. 9. Moment equilibrium

$$\begin{bmatrix} \dot{y_L} \\ \dot{x_L} \\ \dot{\theta_L} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{y} \\ \dot{x} \\ \dot{\theta} \end{bmatrix} \tag{1}$$

$$\begin{bmatrix} F_{xi} \\ F_{yi} \\ T_i \end{bmatrix} = \begin{bmatrix} \cos\varphi_i & -\sin\varphi_i \\ \sin\varphi_i & \cos\varphi_i \\ R & 0 \end{bmatrix} \begin{bmatrix} F_{wi} \\ f_{wi} \end{bmatrix} \tag{2}$$

where

- F_{xi} is force that No.i wheel provides in the x direction,
- F_{yi} is force that No.i wheel provides in the y direction,
- T_i is torque that No.i wheel provides in the positive rotation,
- F_{wi} is force that No.i basic wheel provides,
- f_{wi} is resistance force of No.i little wheel.
- φ_i is the angle between the wheel and y axis, $\varphi_1 = -145°$, $\varphi_2 = -35°$, $\varphi_3 = 35°$, $\varphi_4 = 145°$
- R is radius of the robot.

The key to the analysis is to solve out F_{wi} and f_{wi}. During rotation, it is very likely that the onmi-wheels will dig into the carpet. In this condition, we assume

that the force the wheel provide to drive the robot is proportional to their support force from the ground and the difference in speed between the wheel and the ground. The reaction force comes from two parts: one is the friction caused by pressure between the ground and the basic wheel, the other is the friction caused by pressure on the small wheels. F_{wi} and f_{wi} can be defined as

$$F_{wi} = KN_i(\omega_i r - \dot{x_{wi}}), \tag{3}$$

$$f_{wi} = -(abs(F_{wi}\mu_2 + N_i\mu_1))sgn((\dot{y_{wi}})). \tag{4}$$

K is the speed-force coefficient, μ_1 and μ_2 are coefficient of F_{wi} and N_i. These coefficient can be measured experimentally. r is the radius of the wheel. ω_i is the angular velocity of the No.i wheel. $\dot{x_{wi}}$ and $\dot{y_{wi}}$ can be derived as

$$\begin{bmatrix} \dot{x_{w1}} \\ \dot{x_{w2}} \\ \dot{x_{w3}} \\ \dot{x_{w4}} \end{bmatrix} = \begin{bmatrix} \cos\varphi_1 & \sin\varphi_1 & -R \\ \cos\varphi_2 & \sin\varphi_2 & -R \\ \cos\varphi_3 & \sin\varphi_3 & -R \\ \cos\varphi_4 & \sin\varphi_4 & -R \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}, \begin{bmatrix} \dot{y_{w1}} \\ \dot{y_{w2}} \\ \dot{y_{w3}} \\ \dot{y_{w4}} \end{bmatrix} = \begin{bmatrix} -\sin\varphi_1 & \cos\varphi_1 & 0 \\ -\sin\varphi_2 & \cos\varphi_2 & 0 \\ -\sin\varphi_3 & \cos\varphi_3 & 0 \\ -\sin\varphi_4 & \cos\varphi_4 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \tag{5}$$

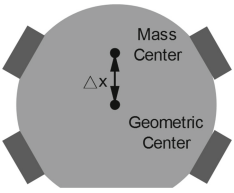


Fig. 10. Mass center deviation

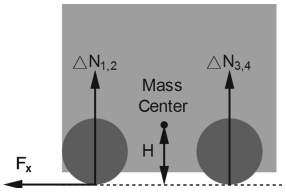


Fig. 11. Moment equilibrium

When the robot is in still, the center of mass of the robot is not in its geometric center, the deviation, denoted by Δx, is shown in Fig. 10. There exist normal force N_{i0} to meet force equilibrium. During acceleration, there exists ΔN_i to meet the moment equilibrium, as shown in Fig. 11. The normal force N_i can be defined as

$$\begin{bmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \end{bmatrix} = \begin{bmatrix} N_{10} \\ N_{20} \\ N_{30} \\ N_{40} \end{bmatrix} + \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \frac{H}{2R\cos\varphi} & 0 \\ 0 & \frac{H}{2R\cos\varphi} \end{bmatrix} \begin{bmatrix} F_x \\ F_y \end{bmatrix} \tag{6}$$

where

$$\begin{bmatrix} F_x \\ F_y \end{bmatrix} = \begin{bmatrix} m & 0 \\ 0 & m \end{bmatrix} R_\theta^{-1} \begin{bmatrix} \ddot{x_L} \\ \ddot{y_L} \end{bmatrix}, \tag{7}$$

$$N_{10} = N_{20} = \frac{mg}{4}(1 - \frac{\Delta x}{R}), N_{30} = N_{40} = \frac{mg}{4}(1 + \frac{\Delta x}{R}) \tag{8}$$

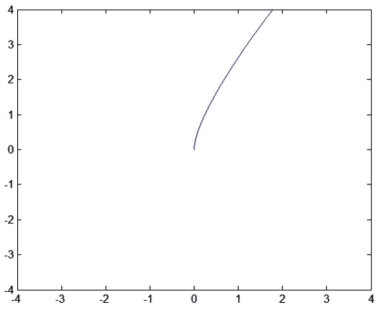
To describe all of the state variable, Eq. (6) can be transformed into the augmented matrix:

$$\begin{bmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \end{bmatrix} = \begin{bmatrix} N_{10} \\ N_{20} \\ N_{30} \\ N_{40} \end{bmatrix} + \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & -\frac{1}{2} & 0 \\ -\frac{1}{2} & -\frac{1}{2} & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} \frac{H}{2R\cos\varphi} & 0 \\ 0 & \frac{H}{2R\cos\varphi} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & J \end{bmatrix} R_\theta^{-1} \begin{bmatrix} \ddot{x_L} \\ \ddot{y_L} \\ \ddot{\theta_L} \end{bmatrix} \tag{9}$$

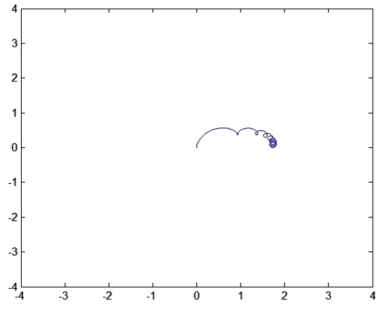
Substitute Eq. (9) into

$$\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & J \end{bmatrix} R_\theta^{-1} \begin{bmatrix} \ddot{x_L} \\ \ddot{y_L} \\ \ddot{\theta_L} \end{bmatrix} = R_\theta \sum_{i=1}^{4} \begin{bmatrix} \cos\varphi_i & -\sin\varphi_i \\ \sin\varphi_i & \cos\varphi_i \\ R & 0 \end{bmatrix} \begin{bmatrix} F_{wi} \\ f_{wi} \end{bmatrix} \tag{10}$$

and we can run simulation using Matlab. The result is shown as Fig. 12.



(a) Robot trajectory(Vy=1.4m/s)



(b) Robot trajectory(Vy=2.4m/s)

Fig. 12. Simulation result

The result simulated by Matlab is consistent with the facts.

When Robot is running in the direction of y axis, the f_{wi} and F_{wi} can be express specifically. It can be deduced from Eq. (1):

$$\begin{bmatrix} \ddot{x_L} \\ \ddot{y_L} \\ \ddot{\theta_L} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta} \end{bmatrix} + \begin{bmatrix} -\sin\theta & -\cos\theta & 0 \\ \cos\theta & -\sin\theta & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}\dot{\theta} \\ \dot{y}\dot{\theta} \\ \dot{\theta} \end{bmatrix} \tag{11}$$

Substitute Eqs. (3), (4), (9), (10) into (11).

$$K \begin{bmatrix} mg\cos 35^\circ(\cos 35^\circ + \sin 35^\circ\mu_2) & 0 & 0 \\ 0 & mg\sin 35^\circ(\sin 35^\circ - \cos 35^\circ\mu_2) & \Delta xmg(-\sin 35^\circ + \cos 35^\circ\mu_2) \\ 0 & \Delta xmg\sin 35^\circ & R^2\, mg \end{bmatrix}$$
$$\left(\begin{bmatrix} \dot{x_s} \\ \dot{y_s} \\ \dot{\theta_s} \end{bmatrix} - \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \right) - \mu_1 \begin{bmatrix} 0 \\ mg\cos 35^\circ \\ 0 \end{bmatrix} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & J \end{bmatrix} \left(\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta} \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}\dot{\theta} \\ \dot{y}\dot{\theta} \\ \dot{\theta} \end{bmatrix} \right) \tag{12}$$

It can be inferred that the rotation is coupled with the motion in the direction of y axis. When the robot is running in the positive direction of y axis, there exists angular velocity because of deviation of the center of mass as well as the deviation of the trajectory. It is possible to compensate the deviation by introducing velocity feedback to angular velocity which requires more sensor to detect the actual speed of the robot, for example, the visual system. In order to simplify the solution, we suggest to solve the issue fundamentally by designing the counterweight for the robot so as to make the mass center overlap with the geometrical center, and thus to reduce the deviation Δx.

It can also be inferred that the rotation is coupled with the acceleration of the robot. The deviation of the center of mass cannot be absolutely eliminated. So when the acceleration of the robot is large, there exists angular velocity leading to the deviation of the trajectory. So we introduce the velocity and acceleration limit into our program to reduce the deviation. For example, when the robot move fast, the velocity of the robot is high. High velocity and high acceleration will cause large deviation of the trajectory according to the analysis. We limit the acceleration in that case because the velocity can not reduce immediately.

5 Defense Strategy in AI System

The Intelligent Control System of ZJUNlict has been fully introduced in the Champion paper last year, which includes strategy selection and trajectory generation based on learning approach [5]. The play script written in Lua assigns tasks to each robot. In this section, we mainly introduce our defense which is a vital part for competition and a superiority of our team. We only lost one point in the RoboCup 2014 tournament, which verifies the effectiveness of our strategy. Our defense is divided into two parts just like real person game: One is Close-Marking Defense, the other is Zone Defense.

5.1 Close-Marking Defense

A flow chart of Close-Marking Defense is shown as Fig. 13. We get the information about all opponent robot and calculate feature value. According to the attribute value we will match the role for every opponent robot, such as leader, passer, etc. Then attack array is set up to describe the robot in order and design defense strategy at last.

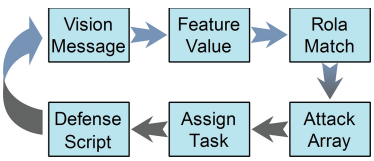


Fig. 13. Defend flow chart

Feature Value. We calculate all feature values for every opponent robot according to vision messages. Feature values, which is used to estimate the threatening level, reflects the state of opponent robots. We have more than a dozen of feature, some are simple value which can be calculated by vision message according to some obvious geometrical relationship, just like the distance between robot and ball, the distance between opponent robot and our goal, shoot angle and so on. The other are some complex feature value which can be calculated by simple feature value, just like Touch Ball value(the ability to receive ball and shoot), Chase Ball value(the ability to chase all and shoot), Pass Ball value(the ability to chase and shoot). For example, Touch Ball value of an opponent robot depends on the shoot angle of the opponent robot, whether other robots block in the pass line, the distance between the robot and our goal. All These features will be used in the next step opponent robots role matching.

Opponent Robots Role Match. Different roles have different priorities. A default role group includes:

- **receiver** is an offensive role which receives the ball and finishes shooting.
- **leader** is an offensive role which controls the ball.
- **attacker** is an offensive role which is always ready for the attack.
- **defender** is a defensive role which takes part in defense strategy.
- **goalie** is the goalie.

The role matched degree is calculated according to the features of robot. For example, when we judge whether the robot is a receiver, we will use three feature value: Touch Value(the ability to receive ball and shoot), Chase Value(the ability to chase all and shoot), Receive Angle(the angle between ball receiving and ball shooting). Matched degree will be calculated for each robot in priority decreasing order.

Attack Array. Attack array is a list of opponent robots. In attack array, We generate the attack array according to the match result. Now every opponent robot has their role and the matched degree for this role. We can compare the role priority and matched degree to set up attack array. But we don't need to mark for certain roles, such as goalie, opponent blocker in kick off area, which should be ignored.

Design Defend Script. We design our defence script in the form of a Finite State Machine [4]. The task is assigned in a state and can receive a parameter which is a number representing the order in attack array. So the robot will defend the corresponding opponent in attack array.

There is an example to demonstrate our defense [10]. Figure 14 shows the task assignment in the defense script. Defend Kick is a task defending the opponent robot which takes the free kick. We can see marking task receiving a parameter which is just an order in the attack array.

A simulation is shown as Fig. 15. Yellow team is the attack side. There are five attackers, four are of attacker role, and one is of leader role. Their defense superiority order is 4-3-1-5-2.

```
["DefendState"] = {
        Leader   = task.defendKick(),
        Special  = task.marking("First"),
        Middle   = task.marking("Second"),
        Defender = task.marking("Third"),
        Assister = task.marking("Fourth"),
        Goalie   = task.goalie(),
}
```

Fig. 14. A part of defense script

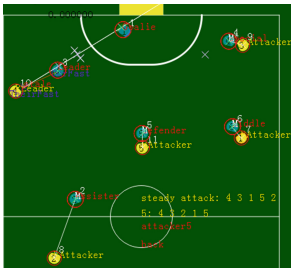
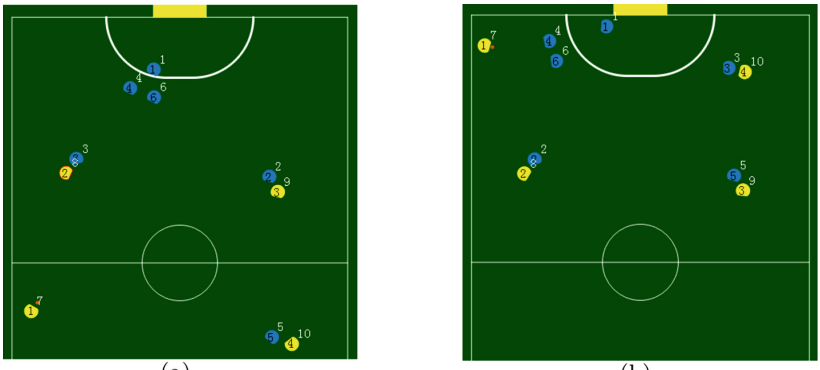


Fig. 15. Defense simulation (Color figure online)



(a) (b)

Fig. 16. Two basic types of Zone Defense: (a)1-2-2; (b)2-3

5.2 Zone Defense

Close-Marking Defense is our main defense strategy. But when competing with some strong teams, they may advance and get rid of our defense robots. We design Zone Defense which mainly takes effect in free kick defense to deal with this kind of situations.

A basic thought is to predict a zone where opponent robots can receive ball and finish shooting according to the angle of passer, the positions of all robots and the max marking range of our robot. Then we can calculate an optimum position according to the opponent robot in the predictable zone. Just like basketball team, we have two basic positioning type, one-two-two defense(one robot in front field, two robots in middle field, two robots in back field) and two-three defense(two robots in middle field, three robots in back field, it's a more conservative formation), which are shown as Fig. 16.

5.3 Summary

We design two kinds of defense to ensure we lose no points to the opponent. With the expansion of game field, Zone Defense will play a more and more important role. In 2015, how to mix Close-Marking Defense and Zone Defense together more flexibly will be a key focus of our team.

6 Conclusion

In this year, we focused on a more stable robot system and a more effective defense strategy. First, the protector for the infrared tube and the shield for the wheel was designed to reduce mechanical wastage, with simulations to analyze and verify the design effectiveness. Second, We reformed the Soft Core architecture to avoid the unreasonable bugs cause by nested references, and made it easier to understand and maintain. Third, we analyzed the robot dynamics to figure out the reason of movement deviation in y direction and come up with a mechanical counterweight design to solve this problem. Finally we design a more effective defense strategy using Close-Marking defense and Zone defense. These efforts make ZJUNlict score 43 goals and only lose 1 goal in RoboCup2014. In next year, we will continue to focus on the defense strategy and try mixing those two methods more flexibly. The counterweight will be designed to compensate the deviation in y direction. We also plan to come up with a mini-driver board for motor driving which can be fixed or replaced easily and flexibly.

References

1. Mu, F., Wu, M., Yang, Y., Yang, G., Yin, D.: Tructure optimization and casting simulation of engine trestle based on CAE technology. In: Computer Supported Cooperative Work in Design (CSCWD), pp. 41–46 (2014)
2. Byun, K.-S., Song, J.-B.: Design and construction of continuous alternate wheels for an omnidirectional mobile robot. J. Robot. Syst. **20**(9), 569–579 (2003)
3. Ren, C., Ma, S.: Dynamic modeling and analysis of an omnidirectional mobile robot. In: Intelligent Robots and Systems (IROS), pp. 4860–4865 (2013)
4. Wu, Y., Zhao, Y., Xiong, R.: ZJUNlict Team Description Paper for RoboCup2013 (2013)

5. Zhao, Y., Xiong, R., Tong, H., Li, C., Fang, L.: ZJUNlict: RoboCup 2013 small size league champion. In: Behnke, S., Veloso, M., Visser, A., Xiong, R. (eds.) RoboCup 2013. LNCS, vol. 8371, pp. 92–103. Springer, Heidelberg (2014)
6. Wu, Y., Yin, P., Zhao, Y., Mao, Y., Xiong, R.: ZJUNlict Team Description Paper for RoboCup 2012 (2012)
7. Browning, B., Bruce, J., Bowling, M., Veloso, M.: STP: skills, tactics and plays for multi-robot control in adversarial environments. J. Syst. Control Eng. **219**(I1), 33–52 (2005)
8. Sheng, Y., Wu, Y.: Motion prediction in a high-speed, dynamic environment. In: Tools with Artificial Intelligence, pp. 703–705 (2005)
9. Thrun, S., Burgard, W., Fox, D.: Probabilistic robotics, vol. 1. MIT press, Cambridge (2005)
10. tolua++ Reference Manual. http://www.codenix.com/tolua/tolua++.html