ZJUNlict
Team Description Paper for RoboCup 2013

Yonghai Wu, Penghui Yin, Yue Zhao, Yifan Shen,
Hangjun Tong and Rong Xiong

National Laboratory of Industrial Control Technology
Zhejiang University
Zheda Road No.38, Hangzhou
Zhejiang Province, P.R.China
zhyaic@gmail.com
http://www.nlict.zju.edu.cn/ssl/WelcomePage.html

Abstract. This paper describes the improvements of ZJUNlict throughout last year. We made some optimization in the AI system and communication. In this paper we mainly focus on the work on AI software system. We supply three ways to config our matching strategies, with the previous job of script configuration in both play-level and agent-level, we really make it a flexible method to write our tactic. Moreover, we are now developing our Script system with Lua, which will be used in the RoboCup 2013.

1 Introduction

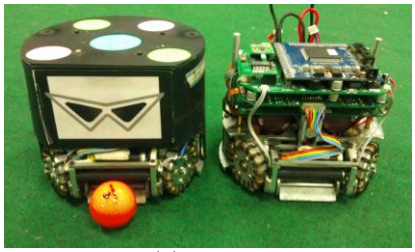
Our team is an open project supported by the National Lab. of Industrial Control Technology in Zhejiang University, China. We have started since 2003 and participated in RoboCup 2004-2009 and 2011. The competition and communication in RoboCup games benefit us a lot. In 2007-2008 Robocup, we were one of the top four teams of the world. And in RoboCup 2012, we even won the second prize in Mexico. We also won the first place in Robocup China Open from 2006 to 2008 and 2011. Our team members are from different majors so that each member can give full play to expertise and make the whole process more efficient.

2 Robot

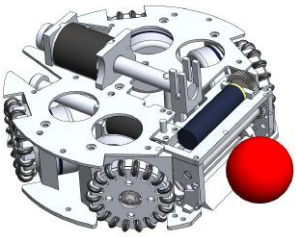
2.1 Mechanical Design

Our new robots (Fig. 1(a))
mechanical information:

- Height: 145mm
- Diameter: 178mm
- Percentage of ball coverage: 18%



(a) Robot



(b) Model

Fig. 1. Our Robots

Our robots are equipped with 4 omni-directional wheels. Each is driven by a 30 watt Maxon brushless motors which helps our robot run at a speed of $2.5m/s$ and $5.0m/s^2$. The reduction ratio of the gearbox with internal spur gear is 4:1. Besides, there are three major machinery devices: a dribbling device, a shooting device and a chipping device.

We redisigned the omni-wheels to reduce the friction between the small passive wheels and the driving wheel so that our robot's movement is smoother. We spend a lot of time testing the dribber materail in order to choose a satisfactory one. The 3d view of robots mechanical system is shown in Figure 1(b).

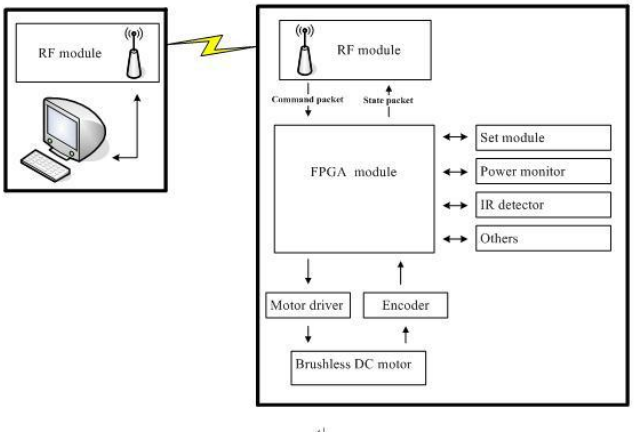


Fig. 2. Schematic diagram

2.2 ElectricalSystem

Our circuit architecture uses FPGA-based all-in-one solution as the central processor module. Our motor driving part is a stable module based on MC33035,

which has been developed to completion since 2007 in Atlanta. There is an encoder module to form a local feedback control loop. A commercial wireless module based on nRF2401 is used on our robot. We choose two smaller capacitors with higher voltage, to achieve a better result. They will help us save more power and permit several shots in short intervals. In addition, we have set module, power monitor module, IR detector module, and so on, in order to complete the functions of our robots show in Fig. 2.

3 AI System

3.1 Strategy Hierarchical Architecture

The AI module for our off-board control system is shown in Fig. 3. It is the brain of planning strategy and coordination among robots in both attack and defense mode. The whole system is composed of world model, decision module and control module.

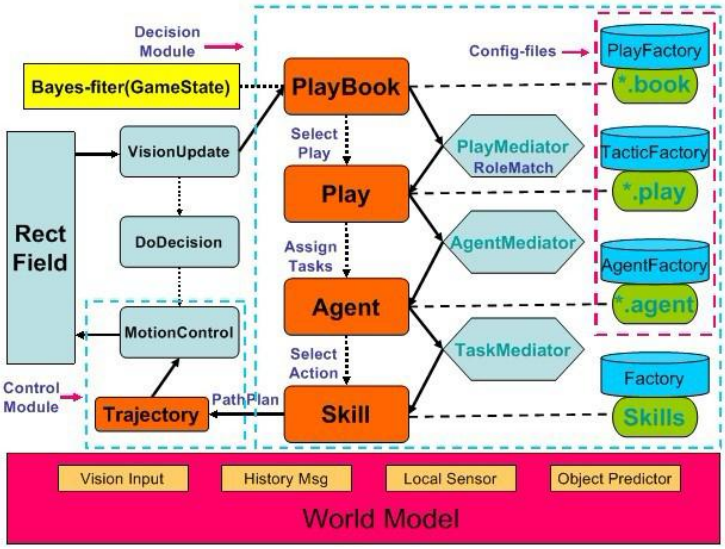


Fig. 3. Software Architecture

With the bayes-based filter evaluating the game status, the decision module selects appropriate play during the match in state of continuity well. Besides, the decision module is rebuilt in a hierarchical style. The plays focus on coordination between teammates, while the agents emphasize on planning skills for the assigned single task from the corresponding plays. The skills are vital for good ones, contributing to executing tasks with high efficiency. Both play-level and

agent-level are configured with Config-files, and will be detailed in next section. The control module is composed of path planning and trajectory generation. It takes RRT algorithm to find a feasible path and Bangbang-based algorithm to solve two-boundary trajectory planning. The world model provides all the information in the match while the decision module will feedback to the predictor in world model. Thus, the close loop system adjusts all agents behavior according to the change of the environment in real time.

3.2 Play

Our AI module is implemented using a play-based approach. Each play represents a fixed team plan, in which each team group has a collaborating mission to perform and that group may have variable agents to execute. We can also consider the play a coach in the soccer game. The plays can transfer to each other, and the group for each agent can change too.

As mentioned above, plays are designed and implemented with the unified FSM module. Plays are composed of many parts, such as applicable condition, evaluating score, roles, finite state machine and role tasks similar to CMU's STP [1]. These parts can all be configured by external scripts. Here's an example of our play-level script in Fig. 4.

3.3 Agent

Agent is performed as robot behavior which is assigned by play to control robot to perform a specific action such as manipulating ball to zone, passing the ball to a teammate, scrambling ball from opponent, getting ball. The agent first select a best proper team member according to the assigned zone and task which receive form play, then selects a proper skill for the team member performing the assigned behavior in every execution cycle and finally generates the best target for robot action. For example, in the shooting task it will be the best shoot point on opponent goal line. Here's an example of our agent-level script in Fig. 5.

3.4 Skill

Skill is a set of basic knowledge for every agent, such as how to move to a point, how to get the ball and kick. Some skill generates a next target point of the specific path which will be passed to the navigation module for finally generating a avoiding collision path. Some skill will direct generate the speed trajectory for some special behavior such as pulling the ball from a opponent front. Each of skills has different main idea of generating path for robot, intercept the ball skill is different from move to point skill in many ways. We can test each skill independently for the best performance for each of skills.

4 New Functions

In this year, we develop some useful functions for our system, which will help us debug our robots and play the game.

```
# Description of play header
SPlay_Attribute attack

SPlay_Applicable
a&b      2
a        validNumMatched
b        canPass(Leader,Special)

SPlay_Timeout    1000

SPlay_Tactic     Show_1P1S

SPlay_MinScore  80  SPlay_MaxScore  89  SPlay_IniScore  85

SPlay_RoleNum    2
SPlay_RoleName   Leader Special

SPlay_StateNum   3
SPlay_StateName state1  state2  finish

# Description of state switch flow
SPlay_StateSwitch
state1  1
  state2      normal      1       a       -1
    a         isBallPassed(Leader,Special)
state2  1
  finish      normal      1       a       0
    a         isBallKicked(Special)
finish  1
  state1      fast        1       !a      1
    a         true

# Description of task for each role
SPlay_StateAction
state1        2
  Special     runBestShootPos   2
    pos       Pos_BestReceive   Leader
    angle     Dir_RoleToRole    Leader>Special
  Leader      pass              2
    pos       Pos_Ajust         Special
    angle     Dir_Robot_Pass    Leader
state2        2
  Leader      runBestAssistPos  2
    angle     Dir_Robot_Ball    Leader
    sender    Rec_Defend_Robot  Special
  Special     shoot             2
    pos       Pos_BestShoot     Special
    angle     Dir_Robot_Shoot   Special
finish        0
```

Fig. 4. Sample Script for Play-level

```
# Description: sample script for an agent

# Define num of inner states
num_of_states 3

# Give name to each state
names_of_states
state1  state2  state3

# Switch of inner states
switch_of_states
state1  2
        state2  normal  2       !a & b
                a       dist2ball<50
                b       canshoot
        state3  normal  2       !a & b
                a       dist2ball<50
                b       canpass
state2  2
        state3  fast    2       !a & b
                a       canshoot
                b       canpass
        state1  normal  2       !a & !b
                a       canshoot
                b       canpass
state3  2
        state2  fast    1       a
                a       canshoot
        state1  slow    2       !a & !b
                a       canshoot
                b       canpass

# Select corresponding skill
skills_of_states
beginning       startskill
state1          getball
state2          kickball
state3          passball
finished        finishedskill
```

Fig. 5. Sample Script for Agent-level

4.1 Uploading When Stop

This season, we add uploading function when the referee core is clicked to have a pause during the normal match. When the referee core is clicked, the upper computer will set a flag bit in the special position in the data packet sent to the robot, which is called upper computer asking. And if the robot find the flag bit match its own car number, the robot will check some of its status, including whether the infra is good or bad, the value of battery voltage and the value of capacitance voltage. After finishing checking, these data will be sent to upper computer to let us know whether this robot is OK. This procedure is called robot answer.

Meanwhile, upper computer will set that flag bit in different special position one by one in terms of robot number to make sure that all six robots will be asked. The cycle of upper computer asking is 16ms, and upper computer will ask each robot at most 10 times. After asking one robot for 10 times without answer, upper computer will give up and ask for the next robot. After lots of tests, we find that the whole procedure can be finished in 0.5s, and the percentage of getting all six robots' information is over 90

Using this function, when the pause of the match, we don't need to go to the playing field to see whether our robots are good or not. It's much more convenient.

4.2 LabVIEW real-time detection

As we know, a good motion control is based on the precise and rapid data feedback. This LabVIEW mode is set up for detecting the real time situation about robot in motion, involved those parameters of each wheel, the measurement result from gyroscope, etc. Further more, we would find the problems and indiscoverable errors of our robot in this way. That is really significance for maintenance and improvement the robot.

4.3 Movement PI Control Depending On Gyroscope

We optimize the movement performance depending on SD-788 gyroscope . We collect the uploaded data delivered by gyroscope. Then we filter the data in corresponding verilog module. The filtered data is used by the movement PI control module. This module will read the rate on Z axis and turn itself into the fit arithmetic-branch which is designed for different rate stages. To take the gyroscope's temperature drift into consideration, we revise the temperature coefficient using LabVIEW and put the coefficient into the measure-decode function.

5 Conclusion

Owing to all team members' hard work, we've made some improvements in our system, both in hardware and software. If the above information is useful to

some new participating teams, or can contribute to the small size league community, we will be very honor. We are also looking forward to share experiences with other great teams around the world.

References

1. Brett Browning, James Bruce, Michael Bowling and Manuela Veloso, STP: Skills, tactics and plays for multi-robot control in adversarial environments
2. Yonghai Wu, Penghui Yin, Yue Zhao, Fan Zhang, Yichao Mao and Rong Xiong, ZJUNlict Team Description Paper for RoboCup2012
3. Yonghai Wu, Penghui Yin and Rong Xiong, ZJUNlict Team Description Paper for RoboCup2011
4. Sebastian Thrun, Wolfram Burgard, Dieter Fox, Probabilistic Robotics, The MIT Press