SEU-Jolly Team Description Paper

Haidan Gao, Chuan Liu, Wei Cheng, An Wang, Bowei Feng, Ye Xue, Xinhang Lu, Rui Jin, Xinyu Gong, Yanran Wang, Zheng Chen, Zesen Chen, Weijia Zhuang, Qi Zhang, and Xiqing Guo

Robocup Research Group, Southeast University, Nanjing, China
haidangao@163.com

Abstract. This document describes the system of kid size humanoid robot designed by our team, SEU-Jolly, for the RoboCup 2016 competition in Leipzig, Germany. We started at the Robocup several years ago, but it's the first time that we have participated in Humanoid League. Yet we still look forward to getting a handsome result.

1 Introduction

In this paper, we describe the robot system for the autonomous competition in the kid size group of Humanoid League. We have made lots of improvements both in hardware and software system for the RoboCup 2016 competition, hoping for good grades at the first show in Humanoid League. Unfortunately, the rules have changed a lot last year which became a big challenge for us new participants.

Three brand new robots from SEU-Jolly named Jolly1, Jolly2, Jolly3 are fully autonomous robots,who are fixed to the limitation in sizes and other aspects. More details will be specialized in following parts.

2 Overview of the System

A photograph of our robots is showed in Fig.1. And the Table.1 just tells the specification of our robots which are consist of a USB camera, a computer board, an Inertial Measurement Unit (IMU), 20 servo motors, a battery and several user interfaces such as buttons. In Fig.2 we demonstrate the software system. The robot continuously locate where he is by processing the information from camera and IMU. At the same time, he takes actions to searching for the soccer at intervals. Once he finds the soccer, relative commands given by the strategy part will be sent to the body control process for next behaviour. Then, servo motors decode and execute these commands quickly. By combining various simple commands, the robots can realize corresponding actions required for soccer competition.

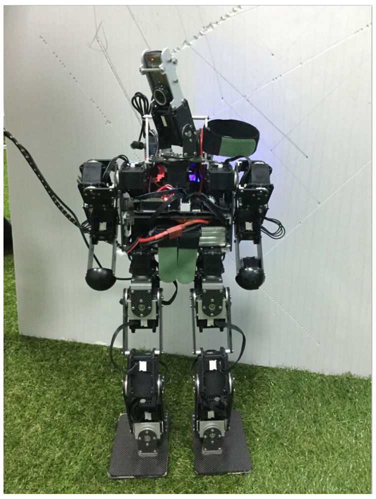

Fig. 1. Robot Jolly1

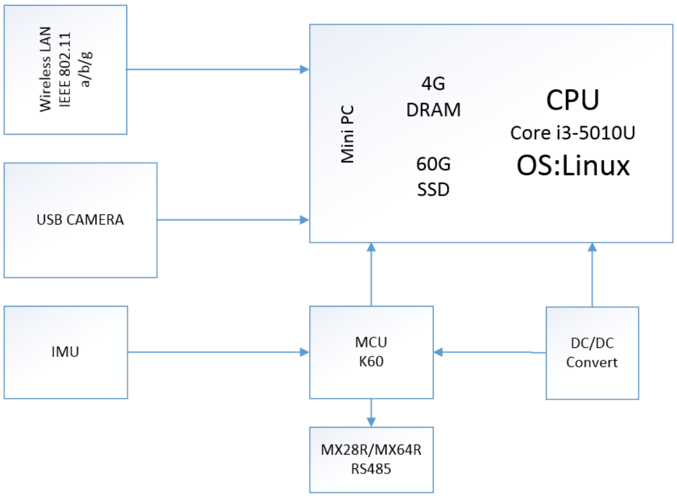

Fig. 2. hardware system

Weight	3960 g(Including Batteries)
Height	530 mm
Velocity(Forward)	0.2 m/s
Walking Directions	All Directions
CPU Board	GB-BX(Core i3H-5010, 1.7 GHz)
OS	Linux(Ubuntu 14.04)
Interface	Ethernet x 1, USB x 2, Push button x 2
Servo motor	MX-28R x 10, MX-64R x 10
Battery	ACE(14.8v 3300 mAh)

Table 1. Hardware System

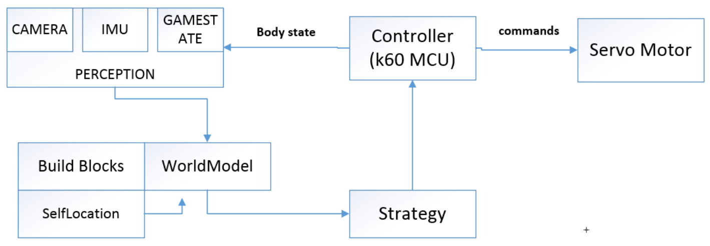

Fig. 3. architecture of software system

3 Computer System

3.1 Hardware

Our computer board(GB-BX) with Core i3-5010 CPU is typical of the high computational capability which is very significant for the robots. And a 60GB SSD is attached to the board for higher I/O speed. All of these are set to support the robot to react quickly in the fierce competitions. With this in mind, all software modules we develop including perception and control are executed on it.

3.2 Development Environment

We use Ubuntu 14.04 32 bit as our development environment both on the robots and our computers. We make use of the router to connect with the robots, which supports for both LAN and WLAN connection. So we just edit and compile source code in our computers, debugging too of course. The WLAN provide us with an easy solution to send files to robots so that we can code everywhere we like without any limitations. However, a direct connection between our computers and robots is also accessible with the Ethernet port when necessary. Considering these all, every teammate has a good chance to try their own idea over and over again.

4 Software System

4.1 architecture

Figure 3 shows the architecture of the software system. The software system are mainly composed of perception, strategy and control three parts. The perception part gets information from sensors and game controller. Images are captured by a USB camera, which are processed in the computer board. Useful information would be sent to the World Model as the sharing parts between processes. Based on these information, blocks like soccer and goal would be built up for self-location and strategies. Since external and internal information both prepared, the strategy part would generate relevant actions for the competition.
The body control tasks are operated in the dedicated control part. The control part controls the body according to commands sent from the strategy part such as walk and kick. The status of the robot (e.g. posture) is periodically sent to the World Model. The control process operates the servo motors by sending commands to them. An IMU is used for gyro feedback and posture estimation. All of these are written in C++.

4.2 Body Control

Two different kinds of servo motors are applied on our robots, MX-64R for the legs and MX-28R for other parts. The maximum torque of MX-64R is 64 kg-cm, which is sufficient to support robots' actions very well. Under the ZMP planning for motion, the robots get fast and stable mobility.

4.3 Image Processing

This process is divided into two parts, one for soccer and another for goal, both very essential. Obviously, this process turns out to be the most important one for this year's rule just changed the yield standards. As past colorful marks are removed, color-based algorithms are longer valid. We use the integral histogram accompanied with some pre-processing, followed by recognition with some characteristics of the goal. This method costs little during the computational process. It's the points of intersection between goal and the ground that are taken as the unique points to be detected. Based on the Harr-like feature, we use the Adaboost algorithm to learn what the soccer is like. To improve our algorithm, we apply the particle filter to minimize the area which we need to search for soccer. In Fig.4 we give an example about the capture of soccer. The resolution of the images can be selected from either 640x480 or 320x240. Our algorithm runs at 20 fps with onboard computer.

4.4 Self-Location

Since the border line is so difficult to distinguish, we take the goal as the feature for robots' self-location. Based on the image information processed by the processing part, we put the Monte Carlo algorithm into practice, which is effective in location.

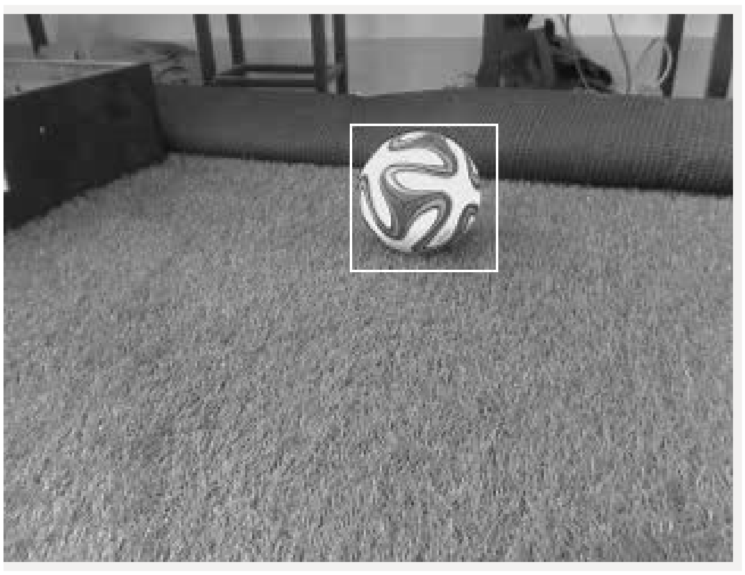

Fig. 4. Capture of Soccer

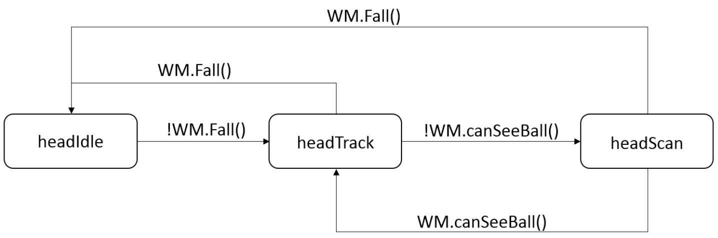

Fig. 5. Head State Machine

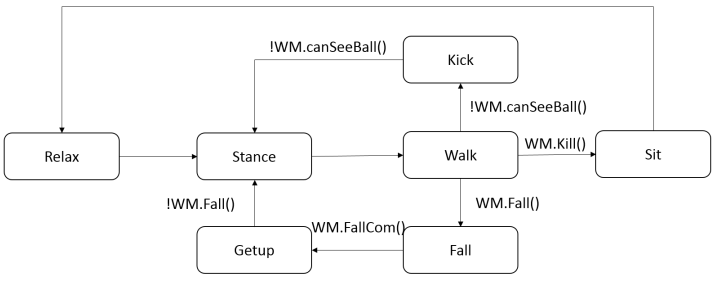

Fig. 6. Body State Machine

4.5 Strategy Part

We use finite state machine to handle low level movements. Head FSM includes looking around to find the ball and tracking the ball if it is found. Motion FSM includes some basic motions such as: standing up, walking, kicking the ball, detecting fall, automatic standing up. Fig.5 and Fig.6 show the state machine we apply in head and body.

4.6 Tools For Software Development

We develop a user-friendly GUI simulator base on Qt as the development environment(Fig.7). In this way, we can get pictures from the distant camera, design the robot action offline and simulate on it.

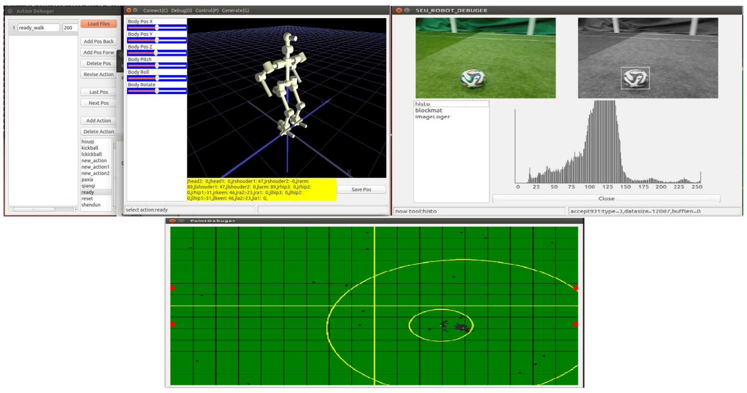

Fig. 7. Simulator

5 Conclusions

In this page, we give a brief introduction to our robots which are newly born for the RoboCup 2016 competition in Leipzig, Germany. Although this competition put much pressure on us, we still believe that our robots would pay back what we have devoted!