Team TH-MOS

Pei Ben, Zhang Jiwen, Shi Xunlei,
Department of Mechanical Engineering, Tsinghua University, Beijing, China

Abstract. This paper describes the design of the robot "MOS series" and the improvements of robot MOS2018, which is based on MOS2017. MOS2018 robots are used as a vehicle for humanoid robotics research on multiple areas such as stability and control of dynamic walking, external sensing abilities and behavior control strategies. Compared with the MOS2017, the robot has some changes to run more steady and more efficient, MOS2018 will be used in RoboCup 2018 competitions.

Keywords. RoboCup, Humanoid, omnidirectional walking, vision, self-localization

1 Introduction

TH-MOS has participated in RoboCup Humanoid League competition since 2006. The MOS robot works on DARwIn-op(Dynamic Anthropomorphic Robot with Intelligence, open platform) software platform with the platform developed earlier by the team. Our team members are devoting ourselves to improve the robots' performances in the field. To make the robot know where it is in the field and become more smart is our target this year.

2 Hardware and Electronics

A photograph of MOS2018 is shown in Fig 1.

The whole structure does not differ a lot from MOS2017. We rebuild the chest of the robot, getting it easy to disassemble the board. Also, we change the position of the ports in the board, so that we can debug conveniently and don't need to disassemble the board. In order to make the robot get up faster, arms of the robot are designed longer.

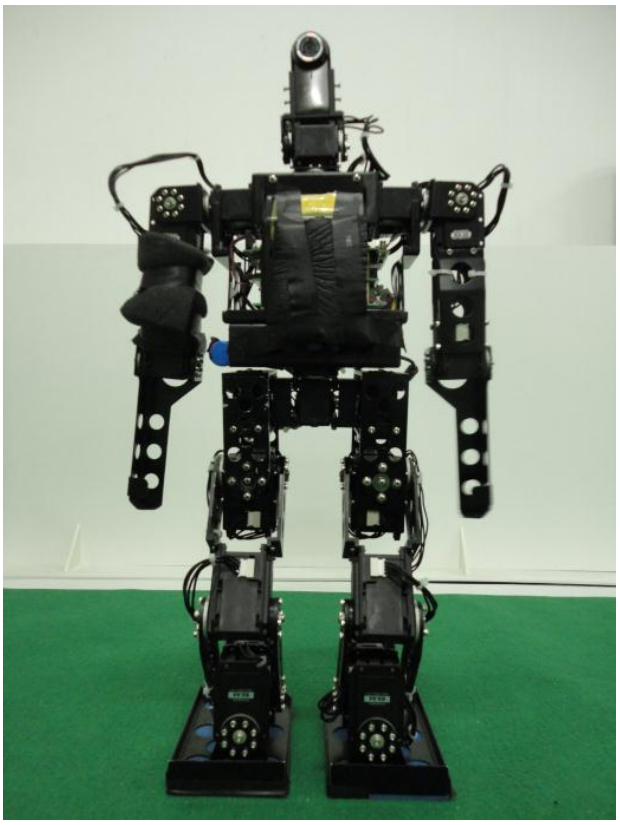

Fig. 1. Humanoid robot MOS2018

Our robot has twenty-one degrees of freedom (DOF): six in each leg, three in each arm, two in the neck and one in the waist. The DOF in the waist provides a better control performance of the robot when kicking the ball and getting up from a fall. Besides DOF configuration, the parameters of different parts such as leg length and ankle height are determined by simulation with gait generating algorithms to ensure better walking stability.

The electronic system of the robot provides power distribution, communication buses, computing platform and sensing schemes for the robot. For having a human-like sensation, we use the camera for its vision perception, a 6-axis sensor (connect 2-axis accelerometer, 2-axis gyro and 2-axis digital compass) for dynamic balanced control and servo motors. Because the legs carry more weight than the arms, the voltage of serve motors in leg is 20V, while the voltage of serve motors in arm is 15V. The architecture of electronic system is shown in figure 2.

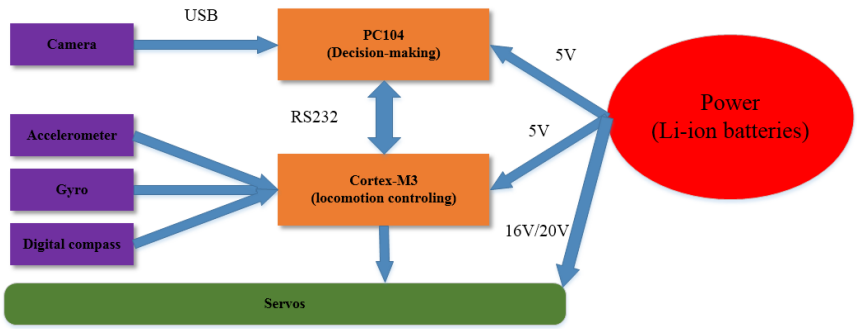

Fig. 2. Electronics architecture for the robot

3 Software and Algorithms

The team transplanted DARwIn-op's algorithms into former software platform since MOS 2014. The port was modified between software and hardware, and used specific algorithms of DARwIn-op. The software architecture is composed of two layers. The first layer receives and processes messages from WLAN (used for team communication), digital compass, camera and joint position sensors. The second layer determines the behavior of the robot using results computed in the first layer and the directions from the controller box.

In the algorithm, we use a finite state machine named Motion FSM to handle all low level body movements such as: standing up, walking, key frame motion, detecting fall, automatic stand up. The FSM guild the robot what to do in different situation, so that the robot can do its job in the field.

The basic motion FSM is shown in figure 3.

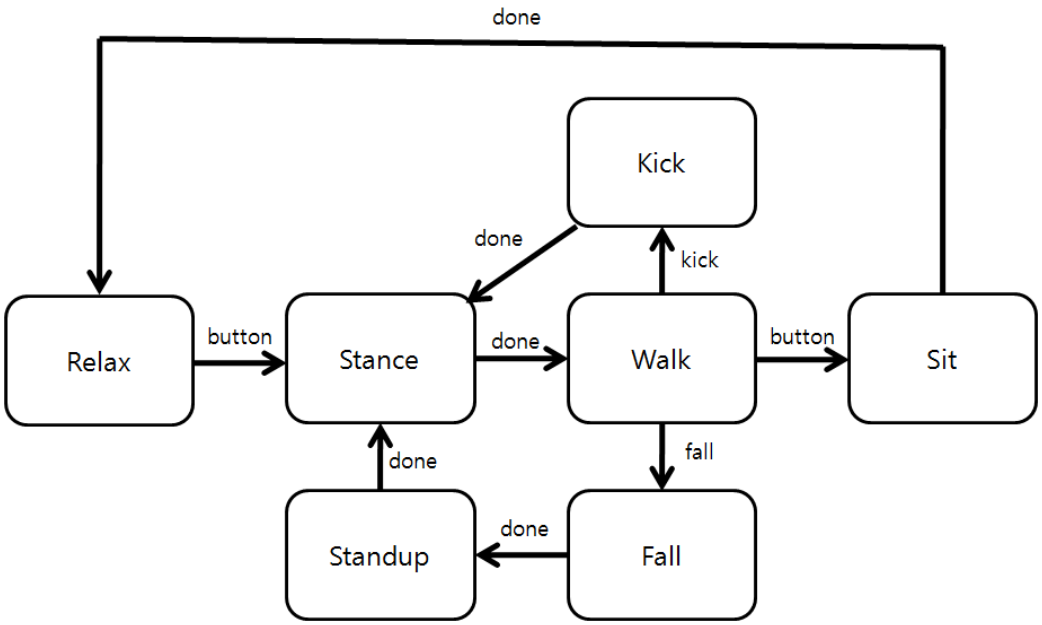

Fig.3 Basic Motion FSM

For the vision of the robot, the eye of the robot should be towards the ball, the field or the goal in the right time. So we have the Head FSM control the head moving. Body FSM controls the servo motors in the body, so the robot can do what it should do and tells what it should do.

Figure 4 is body FSM and head FSM.

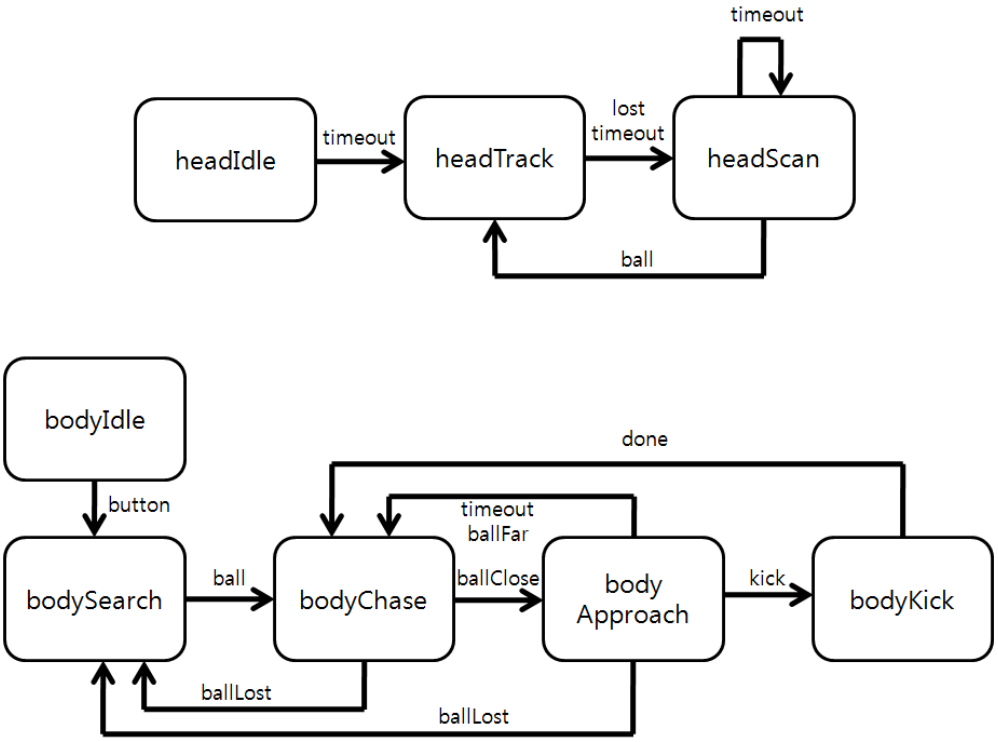

Fig.4 Body FSM and Head FSM

3.1 Omnidirectional Walking

The gait of most bipedal robots is controlled by precomputed trajectories, however, in robot soccer, a dynamic environment forces the robot to adapt their walking direction, speed and rotation to the changes [1]. A robot should be able to arrive at any point in the field and in its sight. And it should rotate itself towards the aim. Based on predefined walking styles, complex path planning algorithm is needed. The generated series of gait can be eliminated when surrounding varies to some extent.

Our goal is capsulate the biped robot into an omnidirectional moving platform in the view of the mounted camera on head, and making gait parameterized with 3 pa-rameters: offset in forward and sidle direction and another rotation direction around Z axis.

Several walking strategies have been developed, most of which are based on the Three-Dimensional Linear Inverted Pendulum [2]. Firstly, foot trajectory is directly deduced from the foot planner from the gait command. Second, the center of pressure trajectory is defined based on ZMP discipline. COM trajectory is simply related to that of COP assuming the robot as a three-dimensional linear inverted pendulum [3][4][5]. Third, inverse kinematics generates joint trajectories based on the former foot and COM trajectories. An analysis resolution of inverse kinematics can be derived from the specific hip configuration of MOS 2013, which ensured the 3 joints intersected on a single point. [6] had issued the details of this method.

In our research, multiple formulas describing the trajectories are sampled, normalized, and saved in motion control board, and thus both of trajectory type and gain can be adjusted offline, and leaves joint trajectories generated online. An accelerating and decelerating algorithm is also developed to cope with a sudden change of walking speed command from behavior.

3.2 Vision Processing

The field lines can provide robots with a lot of useful information for self-localization and control strategies. In real-time competitions, the detection must be efficient. Below is a method we find lines.

First, we scan the image with a large scan-step roughly. Then we make a local-precise scanning in regions where may be edge points. In this process, we use Sobel Operator to detect the edge points considering the limited computing ability of the CPU carried by the robot. We also add the limitation of the environment and the robot's camera to reduce the scanning time, and use an adaptive threshold of gray value, based on the change of gray value along the edge between white and green, to improve the accuracy. Fig. 5 shows the result of point extraction.

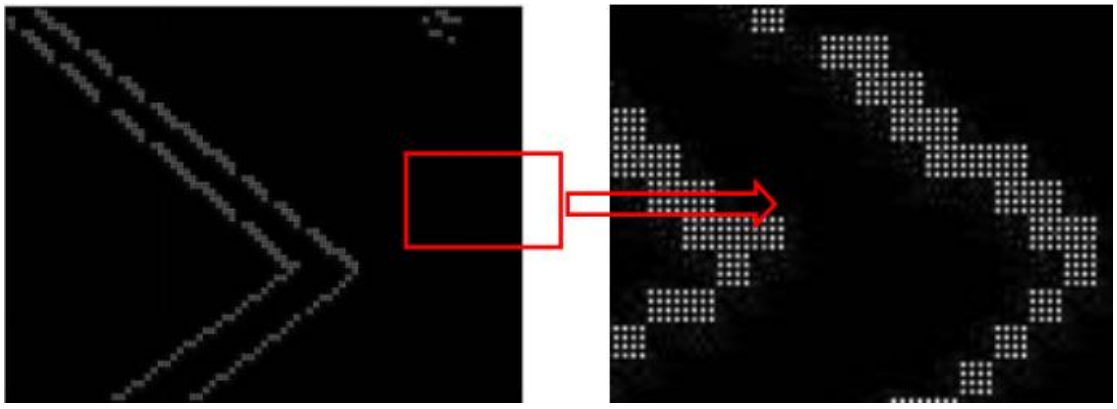

Fig. 5. Precise scanning and edge points extracted

After all edge points are picked out, a modified Hough Transform [7] combining with the gradient direction of these points is used to extract lines. Thanks to the limi-tation of the gradient direction, many invalid calculations are avoided and the calcu-lating time is reduced greatly. In addition, to improve the accuracy, we also recalcu-late some important parameters. Fig. 6 shows the detection result of straight lines in different images.

Fig. 6. The detection result

For efficiency, our robots tell the line by the color now. First, we get the image of the field. Then we sign the color in the image through our MATLAB program. We sign the line with white, sign the grass with green. When our robot is finding the lines, it scans the image through rows and cols. If it finds the color is changing as “green-white-green”, it judges the white area as a line. Then it computes if this area is too large or too small, if it is too high or too far. If this area is unqualified, our robot judges this area is not a line. Else, our robot confirms it is a line. In the same way, we can get the goal from the image.

But we can’t find the ball through this way. The ball used in RoboCup has been too colorful since 2016. We can’t get the ball just by color. We find the ball by its shape. The ball is a sphere, so it is a circle in the image. We use the classical Sobel detector to process the original the image from the camera to get an image for RHT. Then we use the randomized Hough transform (RHT) to find the center circle and compute its position.

3.3 Self-localization

Self-localization is a state estimation problem. The robot needs to estimate its position and orientation from the data of its sensors, mostly camera. We choose the widely used particle filter algorithm to solve this problem. The theoretical foundation is from ‘Probabilistic Robotics’, [8] and some ideas are from GT2005. [9]

To know where it is in the field, the robot has to get information from its camera. The camera gets the photos of the environment, the robot detects and gets the features in the photos by its algorithm. Then the robot compute the differences between features in the photos and features predicted. Through the differences, the robot gets its position and gesture. Because our robot has only one camera as the eye, it can’t get the distance information from the photos. We make borders of the field be landmarks, our robot can get distance information by matching landmarks.

There are too many interferences in the environment, so the information robots got would have much noise. To eliminate the noise, we use particle filter algorithm.

The prediction, or control update, incorporates the states of particles with data from the odometer and IMU, and then some Gaussian noise is added. In the measurement update, we first incorporate the data from the camera and the compass, so we can distinguish similar landmarks and know their directions relative to the robot, and then we can update the states with this information. After that, we resample the particles. In this step, we’re trying to keep as ‘many low-probability particles’ as possible. In the fourth step, we draw a final estimation from the particles, which can be used to make decisions in behavior algorithms. The state space is divided into 10x10x10 cellsand we find the 2x2x2 cube which has the most particles. The weighted average of particles in this cube is the final estimation.

The algorithm of initialization of particles is also important. We design different algorithms for different situation, such as initialization for just stand up, and initialization at the beginning of the match.

If the center circle is found, our robots will locate based on the information the center circle. It is not difficult to compute the distance between the robot and the center the field by using some optical knowledge and geometry skills. Then combined with the magnetic location which can get the information of the direction of robots, we can know the exact position of our robots.

3.4 Behavior

The architecture of the algorithms of robot behavior is based on a hierarchical state machine implemented in XABSL [10]. The architecture is composed of a series of options. A simplified option graph of robot behavior is shown in Fig. 7. The main task, which enables the robot to play soccer autonomously, is defined as the root option. The root option is separated into subordinated options until they become basic options, which can be executed by the robot. Those basic options include getting up from a fall, finding the ball, walking and kicking the ball.

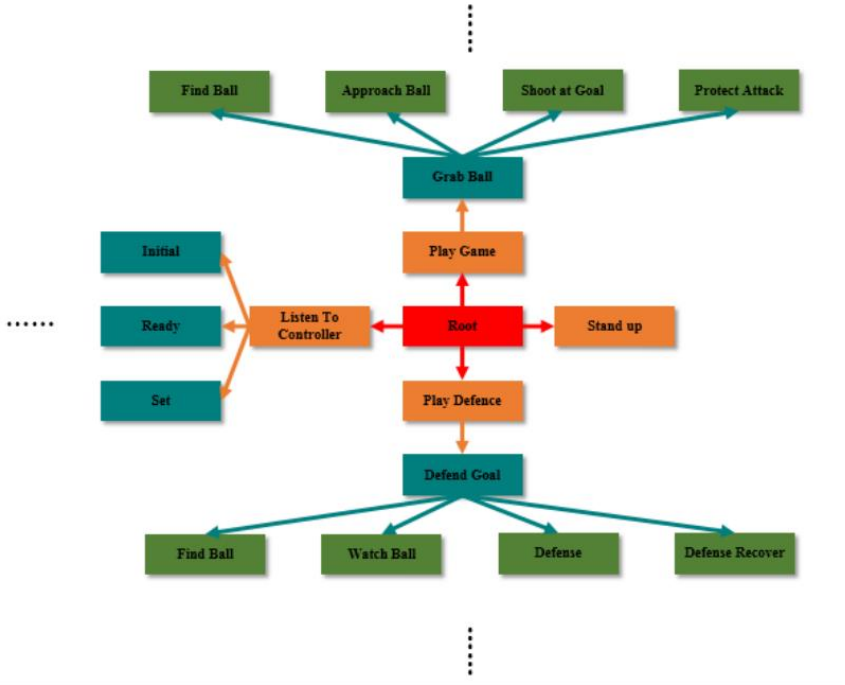

Fig. 7. A simplified option graph of robot behavior

Cooperation between robots is implemented on our robots. Our robots can share information of their position, where the ball and the goal is through WLAN , so that they can cooperate to get the ball into the goal. Also, our robots share what they are doing now, if they are doing the same thing, one robot will stop doing this for not obstructing the other. We are trying to make our robots can pass the ball between each other, so we can make the ball in our control.

4 Prior Performance in RoboCup

Team TH-MOS has participated in RoboCup Humanoid League competition since 2006. We are in the top-16 ranking list for 2012/13, and ranking second in technical challenge in 2013. This year , we believe MOS2018 will have a better performance with the efforts of generations.

5 Conclusion

This paper mainly introduce the details of MOS2018,including its hardware configuration, electronics architecture, software architecture, and what our team is doing now..

References

1. S. Behnke. Online Trajectory Generation for Omnidirectional Biped Walking. Proceedings of the 2006 IEEE International Conference on Robotics and Automation. Florida, 2006, pp. 1597-1603.
2. S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, H Hirukawa. Biped Walking Pattern Generation by Using Preview Control of Zero-Moment Point. IEEE International Conference on Robotics and Automation. pp. 1620-1626, 2003.
3. D. Gouaillier, C. Collete, C. Kilner. Omni-directional Closed-loop Walk for NAO, 2010 IEEE-RAS International Conference on Humanoid Robots. Nashville, TN, USA, 2010, pp. 448-454.
4. J. Alcaraz-Jimenez, D. Herrero-Perez, H. Martinez-Barbera. Motion Planning for Omnidirectional Dynamic Gait in Humanoid Soccer Robots. Journal of Physical Agents, 5(1): 25-34. 2011.
5. J. Strom, G. Slavov, E. Shown. Omnidirectional Walking using ZMP and Preview Control for the NAO Humanoid Robot. RoboCup 2009: Robot Soccer World Cup XIII. Springer, 2009, pp. 378-389.
6. C. Graf, A. Hartl, T. Rofer, T. Laue. A Robust Closed-Loop Gait for the Standard Platform League Humanoid. Proceedings of the 4th Workshop on Humanoid Soccer Robots in conjunction with the 2009 IEEE-RAS International Conference on Humanoid Robots. Paris, France, 2009, pp. 30-37.
7. O. Chutatape, Linfeng Guo. A modified Hough transform for line detection and its performance. Pattern Recognition, 32(2): 181-192, 1999.
8. S. Thrun, W. Burgard, D. Fox. Probabilistic Robotics. MIT press, 2005
9. T. Rofer, T. Laue, M. Weber, et al. German Team RoboCup 2005. [Online]. Available: http://www.germanteam.org/GT2005.pdf
10. M. Lotzsch, M. Risler, and M. Jungel. XABSL - a pragmatic approach to behavior engineering. Proceedings of IEEE/RSJ Intl. Conf. of Intelligent Robots and Systems.