Robot Hardware, Software, and Technologies behind the SKUBA Robot Team

Kanjanapan Sukvichai, Teeratath Ariyachartphadungkit, and Krit Chaiso

Faculty of Engineering, Kasetsart University, 50 Phaholyothin Road, Thailand
fengkpsc@ku.ac.th,
{teeratath,nuopolok}@hotmail.com

Abstract. SKUBA is a winner from RoboCup 2011, Turkey. The aim of this paper is to explain the basic concepts of the design of the robot hardware and the AI system. The robot mechanics are explained in detail along with the electronic boards. Protection circuits are added to make the robot more robust. A torque controller is implemented as low-level controller to reduce the effect of surfaces. The high level AI system is separated into three major modules: predictor/tracker, strategy, and control module. The tracking algorithm and the Kalman Observer are implemented in the predictor/tracker module. The strategy module takes care of playing and evaluates the probability of each play against the opponent. Finally, the control module is explained. Path planning algorithms and a modified kinematics equation are implemented in this module in order to make the robot move along the desired trajectory with less velocity error.

Keywords: Small Size League, mobile robot, torque controller, modified kinematics.

1 Introduction

The Small Size League is a part of the RoboCup robot competitions, which take place every year. The aim of RoboCup is to promote robotics and AI research by offering a publicly appealing, but formidable challenge. The Small Size League is a robot soccer competition that is designed based on the FIFA rules. The main interesting and strong characteristic of this league are the complex AI algorithms, fast speed game play, and multi-agents cooperation. Each team has five robots on the field and plays against the opponent. The overhead wide-angle cameras are mounted over each side of the field and send their images to the SSL-Vision system [1]. The SSL-Vision will broadcast the positions of the ball and all robots to each team via a network cable.

Skuba is a Small Size League soccer robot team from Kasetsart University, which has participated in the RoboCup competition since 2006. We won the championships in the last three years, i.e. the RoboCup 2009 in Graz, Austria and 2010 in Singapore and RoboCup 2011 in Istanbul, Turkey. We also won the RoboCup Iran Open in Tehran in April 2011. In this paper, the SKUBA robot structure and SKUBA software are revealed and explained.

T. Röfer et al. (Eds.): RoboCup 2011, LNCS 7416, pp. 13–24, 2012.

2 SKUBA Robot

This section describes the robot electronics system that is used in the driving system including its designs and components. Details about operations and algorithms are presented in the firmware section. The robot consists of two electronics boards: the main board and the kicker/chipper board. The main board handles all of the robot tasks except kicking. The kicker board controls the entire kicker system. The robot mechanic is described in the last section of this topic.

2.1 Robot Electrical Circuit

The board consists of a Xilinx Spartan-3 XC3S400 FPGA, motor driver, user interface, some add-on modules and a debugging port. The microprocessor core (RISC-32 architecture) and interfacing logic for external peripherals are implemented using FPGA in order to handle the low-level control of the brushless motors such as velocity and position control. The main electronics board receives commands from the main software running on an off-board computer. The board integrates the processing components together with the power components to keep the board compact and to minimize wiring. With limited space, almost all components are implemented as small SMD packages. There are two types of brushless motors in the robot: the driving motor and the dribbling motor. Each driving motor is a 30 W Maxon EC45 flat motor with a custom back-extended shaft for attaching an encoder wheel. The motor itself can produce a feedback signal from Hall sensors for measuring wheel velocity. However, this multi-pole motor sends only roughly 48 pulses per revolution; therefore, this motor is equipped with a US Digital E4P encoder, which has a higher resolution due to its 1440 pulses per revolution. The dribbling motor is a high speed 15 W Maxon EC16 motor connected with the encoder in the same fashion as the EC45. Figure 1 shows the SKUBA main PCB board.

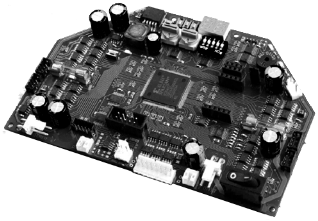
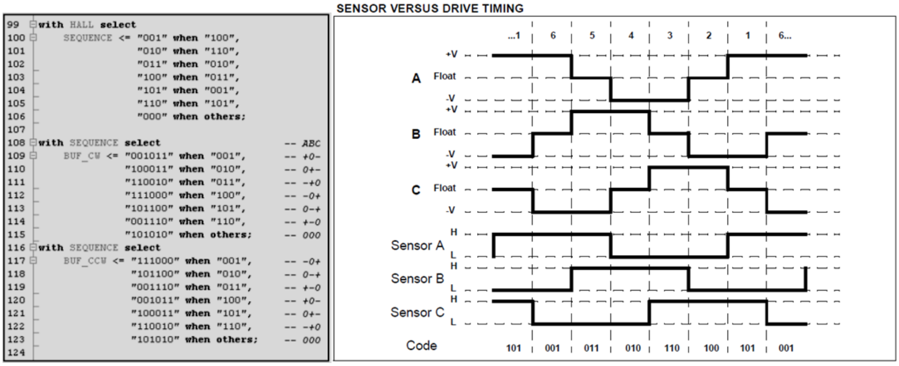


Fig. 1. SKUBA main PCB board

In order to drive the brushless motor, the driven PWM signal must be generated in particular sequences through several groups of MosFETs. The digital sequencer component is implemented in an FPGA via the VHDL language in order to control the driving sequence perfectly. There are two critical issues of the driving system that can damage the robot: motor dead-time and over-current. The dead time of the motor is one of the most critical issues for the robot, because it can damage the driver

MosFET, the battery, and the power supply of the robot; hence the dead-time protection is also implemented in the FPGA. The dead-time value can be found in the motor datasheet. The motor sequence diagram and the VHDL code are shown in Fig. 2.



```vhdl
with HALL select
    SEQUENCE <= "001" when "100",
                "010" when "110",
                "011" when "010",
                "100" when "011",
                "101" when "001",
                "110" when "101",
                "000" when others;

with SEQUENCE select                             -- ABC
    BUF_CW <= "001011" when "001",               -- +0-
              "100011" when "010",               -- 0+-
              "110010" when "011",               -- -+0
              "111000" when "100",               -- -0+
              "101100" when "101",               -- 0-+
              "001110" when "110",               -- +-0
              "101010" when others;              -- 000
with SEQUENCE select
    BUF_CCW <= "111000" when "001",              -- -0+
               "101100" when "010",              -- 0-+
               "001110" when "011",              -- +-0
               "001011" when "100",              -- +0-
               "100011" when "101",              -- 0+-
               "110010" when "110",              -- -+0
               "101010" when others;             -- 000
```

Fig. 2. The VHDL code of the motor sequencer (left) and the motor trimming diagram (right)

The second problem is the over-current problem. This problem occurs when the load of the motor is larger than during normal operation such as when two robots collide, or push against each other. The motor driving current is measured by an ACS712 and the measured data is used in the over-current protection circuit inside the FPGA. If the current is above the set boundary, the firmware in the microprocessor will reduce the PWM duty cycle to keep the current always below the motor limit.

Other circuits, which are needed in the small size robot, are the kicker, the chipper, the ball IR sensor and the wireless circuit. Kicker and chipper circuits are similar since they are the boost convertors, which pump the voltage from 18.5 volts to 200 volts and store it in two capacitors. The IGBT will discharge the stored energy to the kicker and chipper solenoids when needed. The ball infrared sensor is placed at the front of the robot to check for the presence of a ball. Checking for the ball is important, because if the robot kicks or chips without a ball, the kicker and chipper mechanisms can be damaged by the reaction force. A Nordic wireless module (nRF24L01+), which operates at 2.4 GHz, is used as the robot's wireless communication module. In each SKUBA robot, two wireless modules are used for bi-directional communication since the robot's current information is important for the AI, for example the battery level, whether a ball is present, some error codes, etc.

2.2 Robot Firmware

The robot firmware is the low-level motion control software. The robot controller is the most important part of the SKUBA system, because high-level programs such as the AI and the strategy module cannot work properly if the robot cannot respond to the command as it should do. The command sent by the AI consists only of the desired translational and angular velocities of the robot, not of the wheels. This command must be converted into wheel angular velocities before it can be used. The challenge of the low-level controller design is to make the robot move with good accuracy. The basic control scheme is a wheel angular velocity PD or PI controller. This controller is easy to implement, but it does not guarantee consistency, when the floor friction is changed. A torque controller is selected and implemented in the SKUBA firmware, since the input of the robot dynamic equation is torque. The torque convertor/duty cycle convertor block is added to the regular wheel angular velocity PI controller as shown in Fig 3.

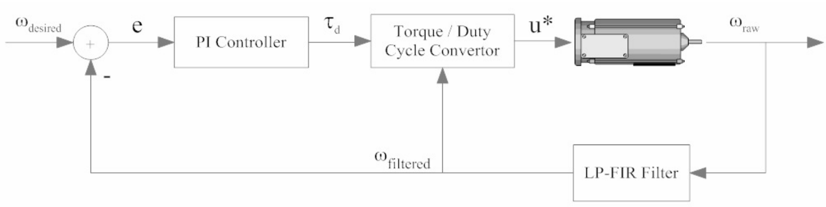


Fig. 3. The torque controller block diagram

The torque convertor equation and its parameters can be found by using the datasheet or experiments. In this paper, the Maxon brushless motor datasheet is used to derive the torque convertor equation as shown in Fig 4. The conservation of energy shown in equation (1) and the motor torque-angular velocity conversion parameters are used to solve the torque convertor equation. The motor torque is the function of the applied voltage and current angular velocity as shown in equation (2).

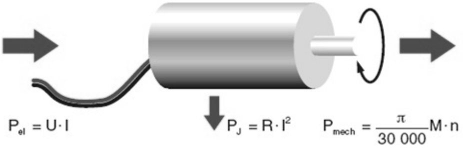


Fig. 4. Model of the Maxon brushless motor

$$U \cdot I = \frac{\pi}{30{,}000} \cdot \dot{\phi} \cdot \tau_m + R \cdot I^2 \tag{1}$$

$$\tau_m = \left(\frac{k_m}{R} \right) \cdot U - \left(\frac{k_m}{k_n \cdot R} \right) \cdot \dot{\phi} \tag{2}$$

where

τ_m is the output torque of the motor

I, U, R are input current, voltage, and armature resistance, respectively

k_m, k_m are the electro-mechanic and angular velocity constant, respectively

$\dot{\phi}$ is the current angular velocity of the motor

Equation (2) cannot be used directly, since the input of the motor is a PWM signal and not a constant voltage. Thus the ratio between the maximum output and the desired torque at a particular speed is used to construct the PWM duty cycle equation. The desired torque is found by a PI controller with the error between the filtered desired velocity and the actual angular velocity of the motor used as input. The complete controller rule can be written in discrete form as equation (3):

$$DutyCycle = u^*[n] = \frac{k_p e[n] + k_i \sum_1^n e[i]}{\left(\frac{k_m}{R}\right) \cdot V\max - \left(\frac{k_m}{k_n \cdot R}\right) \cdot \omega_{filtered}} \tag{3}$$

where

$\omega_{filtered}$ is the currently measured angular velocity after filtering
e is the error between filtered desired velocity and actual angular velocity
$V\max$ is the power supply voltage that is applied to the motor driver
u^* is the duty cycle of the PWM signal
k_p, k_i are the proportional and integral PI controller gains, respectively

The optimal PI gains are found by manually tuning the robot on three different surfaces. The benefit of this controller is the robustness of the robot motion when the robot moves on surfaces with different frictions [2] such as the competition carpets. Fig. 5 shows the robot tuning on different surfaces. Fig 6 shows the trajectories of the robot, comparing the PI controller and the torque controller on three surfaces.

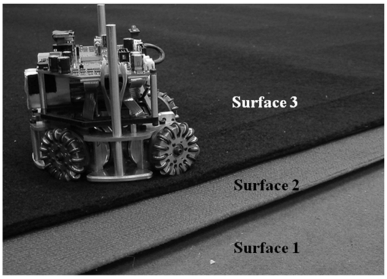


Fig. 5. Robot tuning on three different surfaces

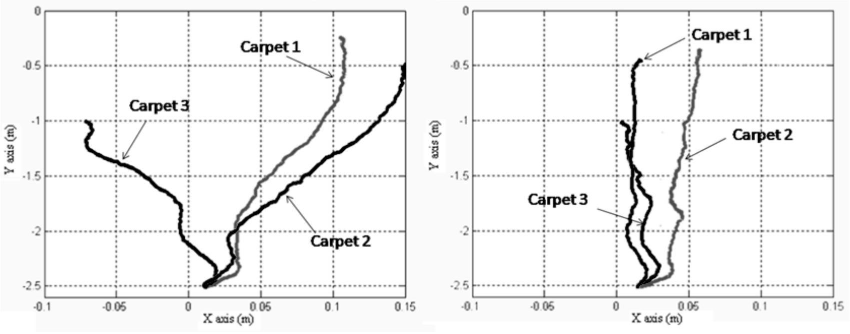


Fig. 6. Robot trajectories on 3 different carpets

2.3 Robot Hardware

The SKUBA robot consists of 4 omni-direction wheels, a kicker, a chipper, and a dribbler mechanism. Most parts of the robot are made from aluminum number 7075 except for the kicker and chipper solenoid, which are made from hard plastic (the previous version used glass epoxy). The wheels are located at 33, 147, 225, and 315 degrees. The idea of the symmetry design of the robot wheels is that the robot usually moves forward and backward more often than left and right, thus this design allows higher speeds for moving forward and backward than it does for moving left and right. The robot chassis and wheel location are shown in Fig. 7.

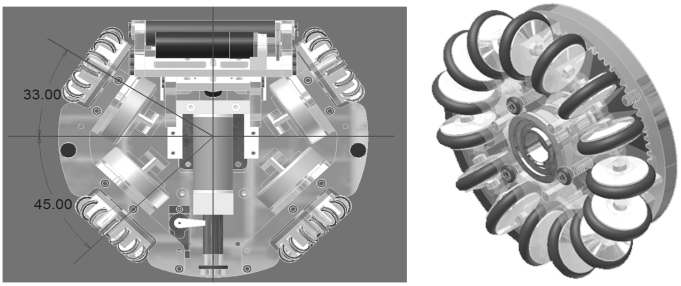


Fig. 7. Robot chassis (left) and robot wheel (right)

The kicker mechanism consists of a moving part and a stationary part. The solenoid, which is fixed to the robot chassis, is made from multiple layers of copper wire number 22 wrapped around a cylindrical and rectangular parallelepiped shape of hard plastic for the kicker and the chipper, respectively. The kicker plunger is made from iron and aluminum while the chipper plunger is only made from iron. An iron rod is in the middle between two aluminum rods in the kicker plunger. Fig. 8 shows the kicker and chipper mechanisms.

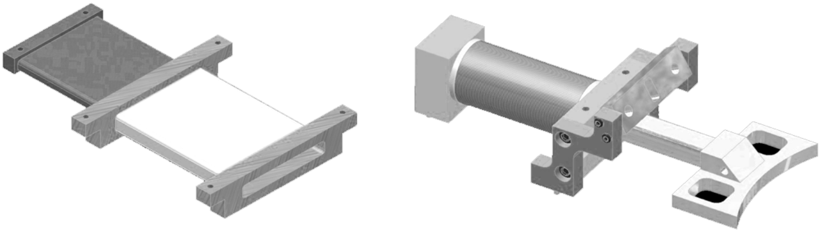


Fig. 8. Chipper (left) and kicker (right) mechanism

The dribbler is made from aluminum rod, which is covered by rubber. This part is also used to catch a moving ball. Thus the soft sponge is placed at the back of the dribbler in order to reduce the energy of the ball. The dribbler part is shown in Fig 9.

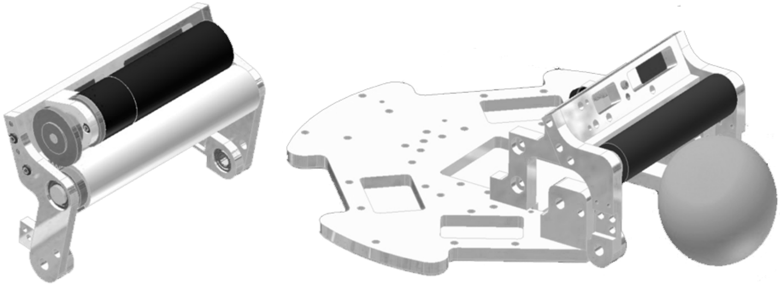


Fig. 9. Dribbler mechanism (left) and assembled to chassis (right)

3 SKUBA Software

In this topic, the software modules of the SKUBA team are explained. The SKUBA software system can be separated into three major parts, which are predictor/tracker, strategy planner, and control module.

Although the information about the robots and the ball are sent by the SSL-vision system frequently, they are not fast enough for a very fast ball or game. Moreover, the latency of the communication between the SSL-vision server and the team AI, the processing time, and the communication from the team computer to the robots can make the system a ‘non-real-time system’. The predictor/tracker module is used to correct this information and to predict the future states of the game before sending it to the next modules. Basically, the predictor can be implemented by using a Kalman filter (observer). The ball and the robots can be modeled as particles running in the free space (field) in a continuous time domain without the control input. There is no control input in this case because it is not possible to know the commands that are sent to the other robots. The measurement noise and state error will be added to the Kalman filter. The optimal state transition matrix for the observer can be found by the algebraic Riccati equation. Finally, the continuous time result will be converted by using bilateral conversion that converts the continuous time system to a discrete time system. In the SKUBA system, the ball predictor and the robots predictor are separated, because some constraints are not the same. The tracker is also important, since the SSL-vision can send out ‘more than one’ ball and robots. The tracker is a state machine program that considers the current position of the ball/robots state compared with the previous state and makes the decision by probability. The maximum speed of the ball and the robots are accounted for in the equation of possibility.

The tracking of the chipped ball is one of the most challenging topics. The ball is captured from the real world to the image frame axis by the SSL-vision and sent to the AI software. The three-dimensional information is lost. In order to reconstruct the ball height, a coordinate transformation is applied. This process can be considered as the reverse process of the SSL-vision. The ball position from the SSL-vision is represented as {x’, y’, 0} while the real position should be {x, y, z} as shown in Fig. 10.

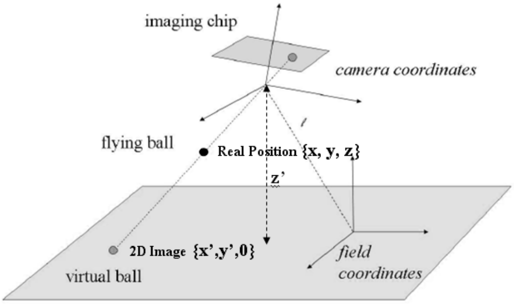


Fig. 10. Three-dimensional ball trajectory

The equation of the motion of a ball in three dimensions is

$$\{x, y, z\} = \left\{x_0 + v_x t + \tfrac{1}{2} g_x t^2, y_0 + v_y t + \tfrac{1}{2} g_y t^2, z_0 + v_z t + \tfrac{1}{2} g_z t^2\right\} \tag{4}$$

The values of $\{v_x, v_y, v_z, x_0, y_0, z_0\}$ are needed to predict the trajectory of the ball. If the camera is stationary above the field, the height z' is a constant. By using the ratio of two similar triangles, the ratio of frame *i* can be expressed as:

$$\left.\begin{Bmatrix}\alpha \\ \beta\end{Bmatrix}\right|_{Frame\ i} = \left.\left\{\frac{x'}{z'}, \frac{y'}{z'}\right\}\right|_{Frame\ i} = \left.\left\{\frac{x_0 + v_x t + \frac{1}{2} g_x t^2}{z_0 + v_z t + \frac{1}{2} g_z t^2}, \frac{y_0 + v_y t + \frac{1}{2} g_y t^2}{z_0 + v_z t + \frac{1}{2} g_z t^2}\right\}\right|_{Frame\ i} \tag{5}$$

By using the rotation matrix that converts from earth axes to image axes and considering the trajectory as a parabolic curve, the final equation can be written in the following form when $\{g_x, g_y, g_z\} = \{0, 0, -9.81\}$

$$\begin{bmatrix} z_0\alpha_i + v_z\alpha_i \cdot t_i - x_0 - v_x t_i \\ z_0\beta_i + v_z\beta_i \cdot t_i - y_0 - v_y t_i \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} g_z \alpha_i t_i^2 \\ -\frac{1}{2} g_z \beta_i t_i^2 \end{bmatrix} \tag{6}$$

There are 3 unknown variables, therefore it needs at least 3 frames to compute the real three-dimensional trajectory. The result of the reconstruction is shown in Fig. 11. Fig. 12 shows the trajectory of a curveball in SKUBA system.

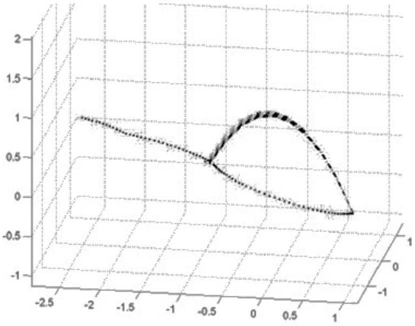


Fig. 11. Three-dimensional ball trajectory reconstruction

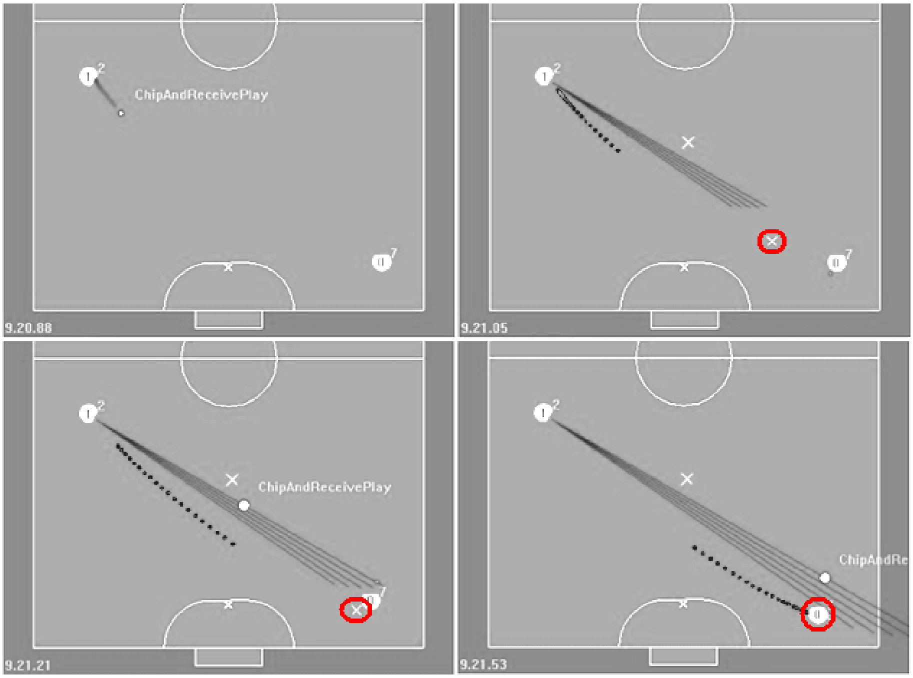


Fig. 12. A curveball in the SKUBA system. The circle shows the predicted drop point.

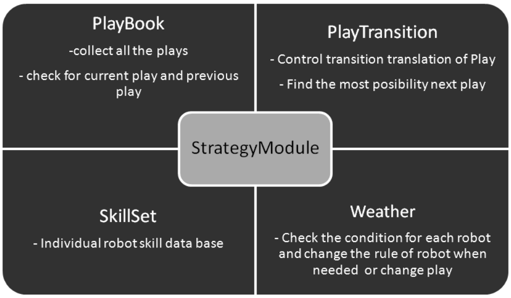


Fig. 13. Strategy module and its sub systems

After the position of the robots and the ball is corrected by the predictor/tracker module, this data will be sent to the strategy module shown in Fig. 13. The strategy module consists of four subsystems, which are PlayBook, PlayTransition, SkillSet, and Weather. SkillSet is a collection of robot individual skills (basic operation or behavior that each robot can do) such as move forward, open dribbler, and a kick skill. The SkillSet is the lowest level of the strategy module. PlayBook is the collection of the play. Play is the sequence of skills that robots have to perform such as free kick etc. In this subsystem, the history of play is recorded, because this information will be used to evaluate the quality of that play and if the quality of the play is high, that play will be used more in this particular game. PlayTransition acts like the conductor that controls the skill sequence rhythm to make a good play. PlayTransition is also used with the history of play to find the next possible play. Finally, the Weather module is used to check the condition of the game, for example whether the attacker robot lost a ball or there is no goalie in the game. The weather module will send a command to the strategy module to change the duty of the robots. The output of this module are the positions of the robots and their duties.

The final module in the SKUBA software is the control module. This module is the most complex one of the system. It controls the high-level motions of all robots, computes the "reasonable points" [3] and the trajectories. It then sends the commands to the robots. The reasonable point for each robot is calculated using the Passing Grid Evaluation proposed by CMDragon in 2009 [3] as shown in Fig 14.

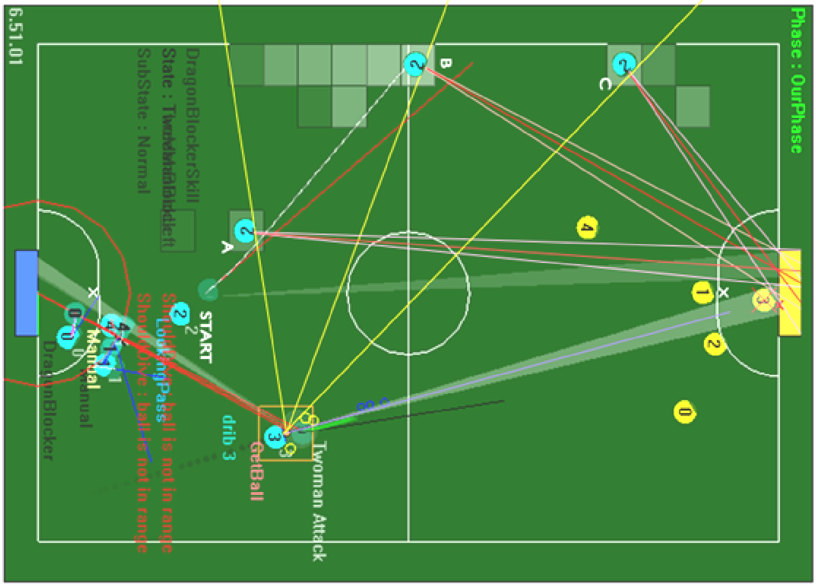


Fig. 14. Grid Evaluation in the SKUBA system

The path planning generation techniques that are used in the SKUBA system are Rapidly-exploring Random Tree (RRT), Sub Goal, and Open loop motion. The RRT is used to generate the trajectory at the regular motion. Sub Goal is used when there are many obstacles along the robot path. Finally, the Open loop motion is the special motion that is used when the robot needs to play a special skill, for example, the robot runs and hits the ball while the ball is still moving in the free space etc. The Open loop path planning uses a geometry calculation and a physical speed calculation to achieve such a movement.

The control algorithm in the SKUBA system that differs most from other teams is the robot velocity control, instead of a robot position control [4]. In the SKUBA system, we definitely control the velocity along the trajectory that is generated by

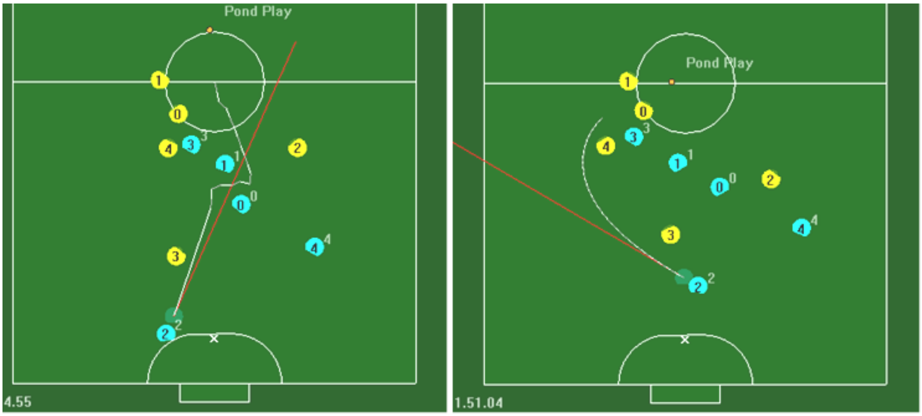


Fig. 15. RRT path (left) and Sub Goal path (right)

modified trapezoid path profiles and not the position. Velocity is more important, because the robot must arrive at a point at a deadline. This enables the robot to play in a very fast game.

The position is cared less about in this algorithm, therefore its overshoot response can lead to a major problem, which is the 'two defenders' penalty. But on the other hand, this approach has advantages when attacking, because position overshoots will help the robot to prevent opponent robots from getting a ball. In order to make a precise velocity control, the velocity PID control loop is added to the control module along with modified kinematics.

The modified kinematics is fully provided in the SKUBA ETDP 2011 [5]. The idea is to merge the friction effect with the regular mobile robot kinematics. The friction matrix is found in experiments. This matrix can be time-varying or constant, in SKUBA's case, this matrix is constant. Because of the complexity, the modified kinematics is implemented in the AI and not in the low-level robot controller. Thus the final equation will be reform to the function of the old command that generates the friction effect matrix as shown below from the trapezoid path profiles generator pulse:

$$\zeta_{send} = \zeta_{send_{old}} + \vartheta \tag{7}$$

where,

$\zeta_{send} = \begin{bmatrix} v_x & v_y & \omega \end{bmatrix}^T$, and ϑ is the friction effect matrix

There are so many things needed to be tuned and recorded during the competition. There is an auto calibration system implemented in the SKUBA system. This system helps to reduce the large amounts of time when manually tuning, and it allows team members to focusing more on the human strategic planning and fixing the robots. The friction effect matrix and the ball/robots predictor are automatically calibrated on the competition field. The Open Loop control also needs to be tuned.

4 Conclusion

Our system has been continuously improved since the beginning. This year, we introduced some improvements in the high-level motion controller and the necessity of the velocity controller to achieve a higher quality of game play. The automatic calibration software is fully used since RoboCup 2010 and it greatly reduces the amount of team setup time, which allowed us to focus more on the strategic planning. The software, which runs the robot team, was built in 2006 and has been improved each year. It has given us very successful competition results for the last several years; the results are summarized in Table 1.

Table 1. Competition results for SKUBA SSL RoboCup team

Competition	Result
RoboCup 2006	Round Robin
RoboCup Thailand Championship 2007	3rd Place
RoboCup Thailand Championship 2008	2nd Place
RoboCup 2008	3rd Place
RoboCup 2009	1st Place
RoboCup China Open 2009	1st Place
RoboCup 2010	1st Place
RoboCup Iran Open 2011	1st Place
RoboCup 2011	1st Place

There are two main focuses for RoboCup 2012, the flux residue in the chip solenoid and game play data mining. This flux will generate the unpredictable chipping behavior. This information will be also be used to improve the trajectory model of the flying ball using an energy equation. The game play data mining is already implemented this year and the result can be used for a defender, but next year we will develop a new algorithm that can mine the useful information for an attacker. We hope that our robot team will perform better in this year and we are looking forward to sharing experiences with other great teams around the world.

References

1. Zickler, S., Laue, T., Birbach, O., Wongphati, M., Veloso, M.: SSL-Vision: The Shared Vision System for the RoboCup Small Size League. In: Baltes, J., Lagoudakis, M.G., Naruse, T., Ghidary, S.S. (eds.) RoboCup 2009. LNCS (LNAI), vol. 5949, pp. 425–436. Springer, Heidelberg (2010)
2. Sukvichai, K., Wasuntapichaikul, P., Tipsuwan, Y.: Implementation of Torque controller for brushless motor on the omni-directional wheeled mobile robot. In: ITC-CSCC 2010, Pattaya, Thailand, pp. 19–22 (2010)
3. Zickler, S., James Bruce, J., Biswas, J., Licitra, M., Veloso, M.: CMDragon Extended Team Description Paper. In: RoboCup 2009 (2009)
4. Sukvichai, K., Wechsuwanmanee, P.: Development of the modified kinematics for a wheeled mobile robot. In: ITC-CSCC 2010, Pattaya, Thailand, pp. 88–90 (2010)
5. Chaiso, K., Sukvichai, K.: SKUBA Extended Team Description Paper. In: RoboCup 2011 (2011)