Skuba 2011 Team Description

Krit Chaiso[1],Teeratath Ariyachartphadungkit[2], and Kanjanapan Sukvichai[2]

[1] Department of Computer Engineering
[2] Department of Electrical Engineering
Faculty of Engineering, Kasetsart University
50 Phaholyothin Rd., Ladyao, Jatujak, Bangkok, 10900, Thailand
nuopolok@hotmail.com, fengkpsc@ku.ac.th

Abstract. This paper presents a brief description of Skuba, a Small-Size League RoboCup robot team. The robot system is designed under the RoboCup 2011 rules in order to participate in the RoboCup competition in Turkey. The Skuba system consists of two main components which are explained in the Robot and the Software Architecture section. This year, we do more research on new ball prediction algorithm which used to predict the motion of a ball in our decision making program. This new predictor will be used mainly in dynamics passing plays.

1 Introduction

Skuba is a small-size league soccer robot team from Kasetsart University, which has entered the RoboCup competition since 2006. We got the championship last two years from the RoboCup 2009 in Graz, Austria and RoboCup 2010 in Singapore.

The robot system consists of two main components: the robot hardware and the software. The software makes strategic decisions for the robot team by using information about the object positions from the vision system. The global vision system run by the shared vision software, SSL-Vision, which uses two cameras mounted over field. The software executes plans by calculating the robot actions and then sends the commands to each robot.

This year, the main focus of our development is the ball prediction. When we analyze the recorded log file from last two RoboCup, we found that there is a problem when a robot tries to pass a ball to other robot. In the past, we couldn't predict the position of ball very precise in very far future and we can just only predict the position of ball in present time and near future.

In order to make more interesting dynamically game play such as "touch and shoot" and "blind side run". The calculation of the position and velocity must be very precise. Predict state of ball from the Kalman filter alone isn't enough to predict ball future accurately. Therefore, we construct the new ball prediction system for predict position and also velocity of ball and uses it throughout our system.

2 Robot

Our team has ten identical robots, six of them were built in 2008 and another four were built in 2009 with some minor changes in material and mechanical design. We are not planning to make any major changes to the design. The robot hardware is the same as used in last year. More details about the robot hardware can be found in [1]. This year, Infrared sensor array (IR array) is introduced to Skuba robot. This IR array is used to measure ball position when it locates in front of the robot shown in Fig. 1. Voltage output from each infrared sensor is normalized, combined and calculated the peak position. This peak will show the real position of a ball related to the robot front.

Fig. 1. Infrared sensor array

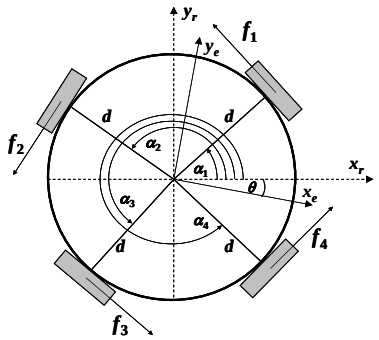

Fig. 2. Robot wheel configuration

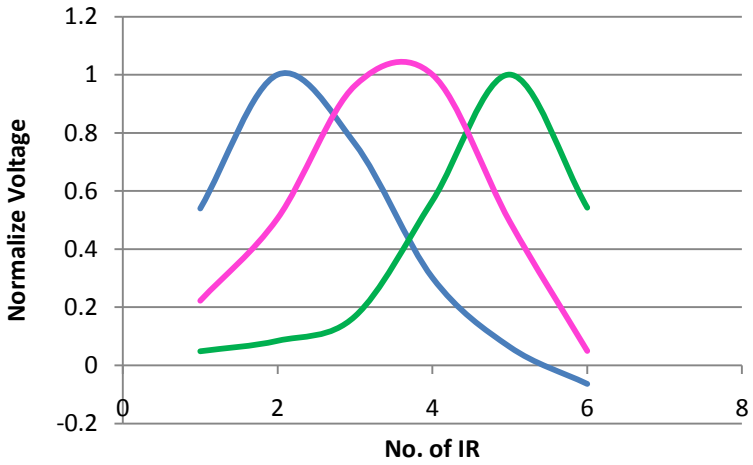

Fig. 3. Relation between normalize voltage from each IR and ball position which ball is center (pink) , left (blue) , and right (green)

Each robot consists of four omni-directional wheels which are driven by 30 watt Maxon flat brushless motors. Each motor is equipped with a 360 CPR optical encoder to provide signals for speed measurement. The robot uses the dribbling device to improve ball handling capability, the dribbler is a round bar covered with a silicone tube and connected to a high speed brushless motor. The bar can spin up to 13000 rpm. The kicker has ability to kick the ball at speeds up to 14 m/s using a solenoid. The chip-kicker is a flat solenoid attached with a 45 degree hinged wedge located on the bottom of the robot which can kick the ball up to 7.5 m before it hits the ground. Both of the solenoids are driven from two 2700µF capacitors charged to 250V. Kicking devices are controlled by a separate board located below the middle plate. The kicking speed is fully variable and limited to 10 m/s according to the rule.

The controller of the robot hardware is done by using a single-chip Spartan-3 FPGA from Xilinx. The FPGA contains a soft 32-bit microprocessor core runs at 30 MIPS and interconnected peripherals. This embedded processor executes the low level motor control loop, communication and debugging. The brushless motor controller, quadrature decoder, kicker board controller, PWM generation and onboard serial interfaces are implemented using FPGA logic gates. The robot receives control commands from the computer and sends back the status for monitoring using a bidirectional 2.4GHz wireless module. A Kicker board is a boost converter circuit using a small inductor. The board is separated from the main electronics for safety.

The robot has a diameter of 176 mm and a height of 147mm.The dribbler covers up to 20% of the ball diameter. The 3D model of the robot and the real robot are shown in Fig. 4 and Fig. 5 respectively.

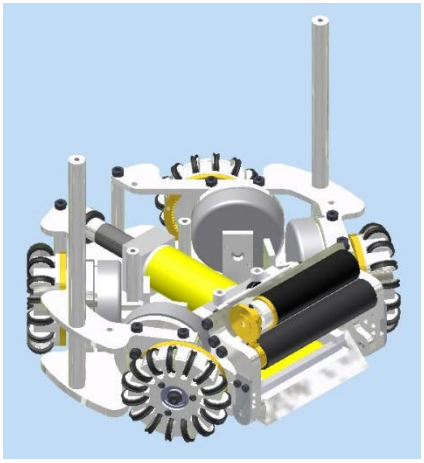

Fig. 4. 3D mechanical model of the robot

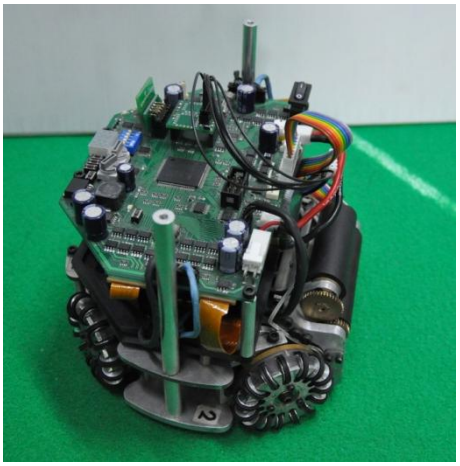

Fig. 5. Real robot

2.1 Modified Robot Kinematics

Normally, when the software sends the velocity command to the robot, it doesn't perform any velocity feedback control and it assumes that the robot's motion controller has already taken care of this. But due to the loss from friction, wheel slippage and other real world problems, the robot cannot move as fast as commanded. The regular robot kinematics describes an ideal situation where there's no system disturbance. In order to control the robot more accurately, the robot kinematics is modified with the some disturbance parameters. The friction force and traction torque vector are defined.
The normal kinematics can be written as:

$$\zeta_r = \psi \cdot \zeta_{Desired} \tag{7}$$

where,

$$\zeta_r = \begin{bmatrix} \dot{\phi}_1 & \dot{\phi}_2 & \dot{\phi}_3 & \dot{\phi}_4 \end{bmatrix}^T$$

$$\zeta_{Desired} = \begin{bmatrix} \dot{x} & \dot{y} & \dot{\theta} \end{bmatrix}^T$$

$$\psi = \begin{bmatrix} \cos\theta \cdot \sin\alpha_1 + \cos\alpha_1 \cdot \sin\theta & \sin\theta \cdot \sin\alpha_1 - \cos\alpha_1 \cdot \cos\theta & -d \\ \cos\theta \cdot \sin\alpha_2 + \cos\alpha_2 \cdot \sin\theta & \sin\theta \cdot \sin\alpha_2 - \cos\alpha_2 \cdot \cos\theta & -d \\ \cos\theta \cdot \sin\alpha_3 + \cos\alpha_3 \cdot \sin\theta & \sin\theta \cdot \sin\alpha_3 - \cos\alpha_3 \cdot \cos\theta & -d \\ \cos\theta \cdot \sin\alpha_4 + \cos\alpha_4 \cdot \sin\theta & \sin\theta \cdot \sin\alpha_4 - \cos\alpha_4 \cdot \cos\theta & -d \end{bmatrix}$$

Desired robot velocity ($\zeta_{Desired}$) is used to generate robot's wheel angular velocity vector (ζ_r). This wheels angular vector is the control signal which is sent from PC to interested mobile robot. The output linear velocity ($\zeta_{Observed}$) is observed by a bird eye view camera. The output velocity contains information about disturbances, therefore by comparing the desired velocity and the output velocity. The output velocity can be defined as (8) when assuming that disturbance is constant for the specific surface. The two disturbances are modeled.

$$\zeta_{Observed} = (\psi^{\dagger} + \varepsilon) \cdot \zeta_r + \Delta \tag{8}$$

where,

$\psi^{\dagger}$ is the pseudo inverse of the kinematic equation

ε is the disturbance gain matrix due to the robot coupling velocity friction

Δ is the disturbance vector due to the surface friction

The disturbance matrices can be found from experiments. From data in last year, the disturbance vector of the surface friction is constant but the coupling velocity friction is a nonlinear function with respect to the robot translation and angular velocity. With these two disturbance parameters, the robot command can be compensated and result in the actual robot output velocity command.

The surface friction is easy to find just by using two observed experimental data while the coupling velocity friction matrix can be estimated using the calibration

software. The software performs the experiment by running the robot at different speeds and observing the output velocity from the robot. Then, the disturbance ε can be estimated by using a second order polynomial least squares fitting method.

By using modified kinematics to generate the control command, the robot can move more accurately. The comparison of the experimental result is shown in Fig. 6.

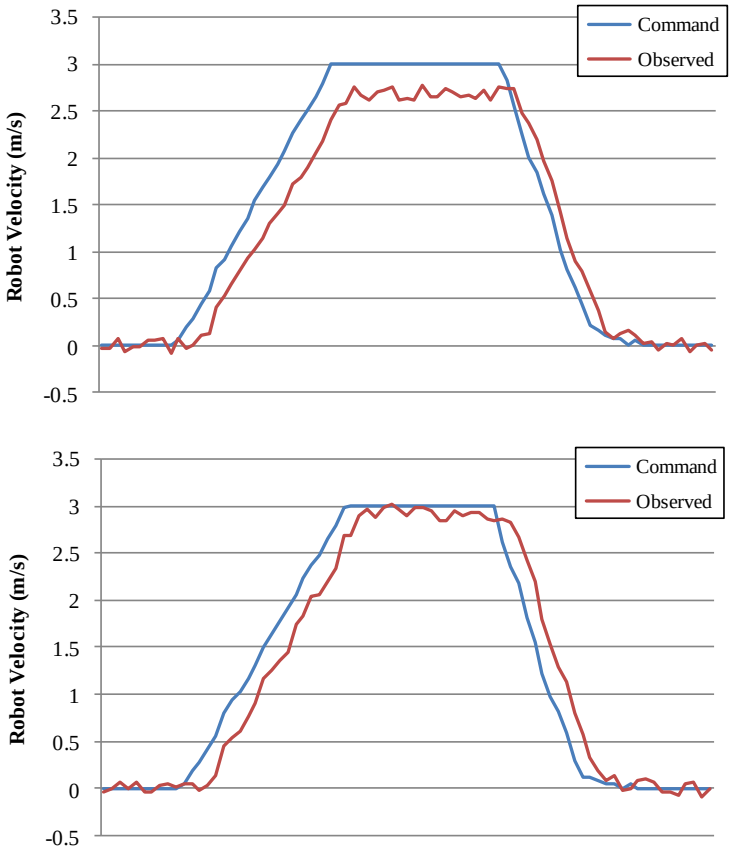

Fig. 6. The robot observed velocity profile using normal kinematics (top) and modified kinematics (bottom)

3 Software Architecture

The overall software architecture is illustrated in Fig. 7. The software consists of several modules organized as a multilayer architecture. This software has been being continuously developed since RoboCup 2006 based on the strategy structure of Cornell Big Red 2002's software. More detailed information of the software can be found in [1].

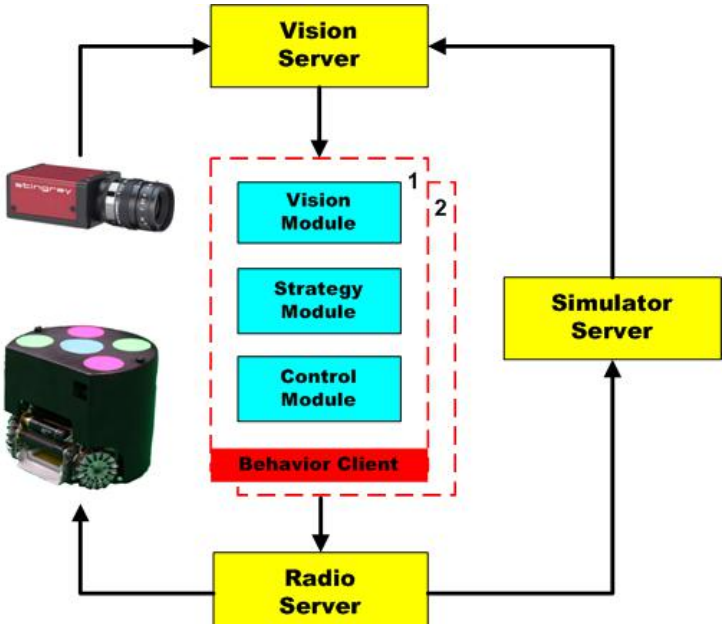

Fig. 7. The software architecture

3.1 SSL-Vision

The use of shared vision system named SSL-Vision is required by the competition rule. This new vision software can be integrated into the system by simply replacing our Vision Server software. With some code changes in the vision protocol, the existing software works with the shared vision system successfully. The SSL-Vision also provides geometric parameters which are very useful for the chip-kicker calibration which is described in the next section.

3.2 Kicker Automatic Calibration

The automatic calibration system was added in last year in order to reduce manpower and time during the team setup process. The calibration system will find the relation between input and output parameters. Similarly to the motion controller calibration method described in section 2.1. the kicker calibration

software is used to estimate the relationship between both the chip-kicking distance or ball speed and the magnitude of the kick command sent to the robot The relationship is then modeled using second order polynomial and can be estimated from the experimental results using least squares polynomial fitting. The example of the kicker calibration result is shown in Fig. 8.

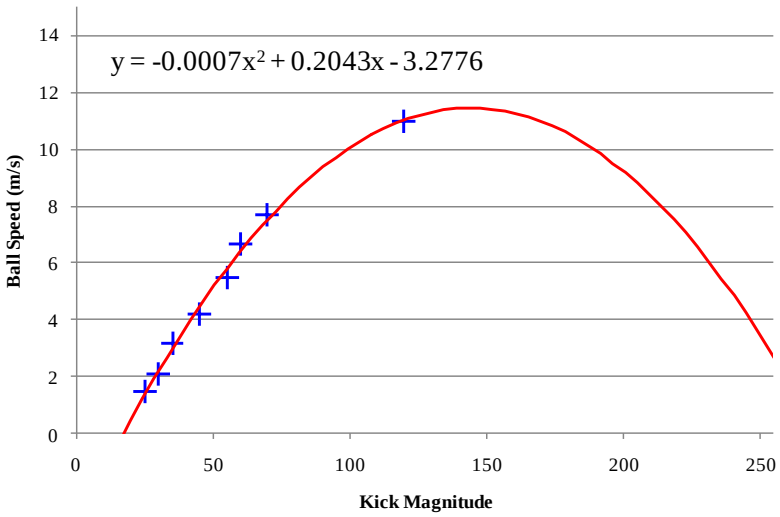

Fig. 8. Relationship between the ball speed and the magnitude of the kick command obtained using second order polynomial least squares fitting

3.3 Ball Prediction System

The estimated states from Kalman filter is not accurate enough to predict the real ball position and velocity because two main issues. One is ball model is not good enough and two is vision system which are lens curvature and an overlap area between two cameras. The precise model of the ball movement is described. There are two state of the motion, first is rolling and second is slipping [2] shown in Fig. 9.
But if the motion of the ball is directly implement, the extended Kalman filter is used. Moreover, all of the parameters in the equation must be found every time when robots are running in different fields. Therefore, in order to make the predictor easier, the experimental approach is selected. The velocity of the ball, which is controlled by robot kicking system, in every frame are collected and fitted to a single equation which is the cubic polynomial expression shown in Fig. 10. Peak of ball velocity is considered and used as a main component in order to fit the cubic curve. The cubic polynomial is now use as the new pre-predicting function and used in every module in our software.

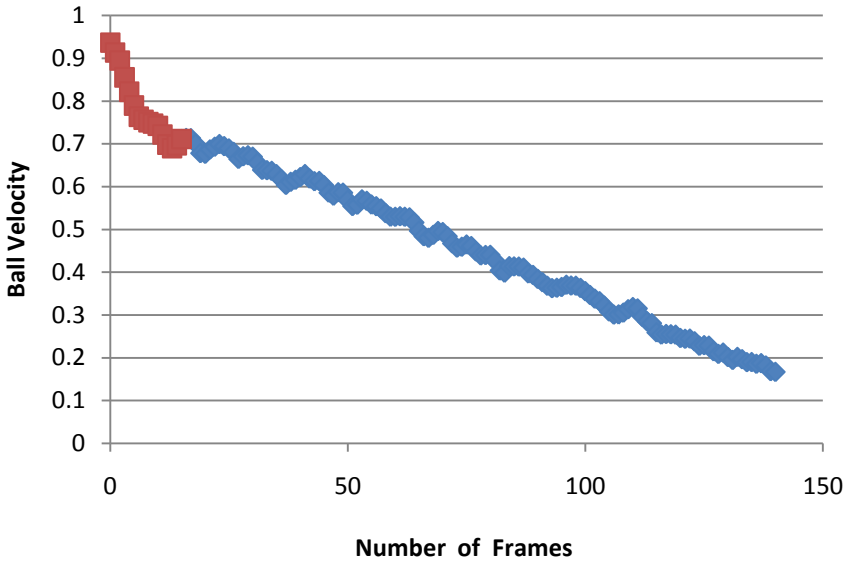

Fig. 9. Measurement ball velocity for slipping (red) and rolling (blue)

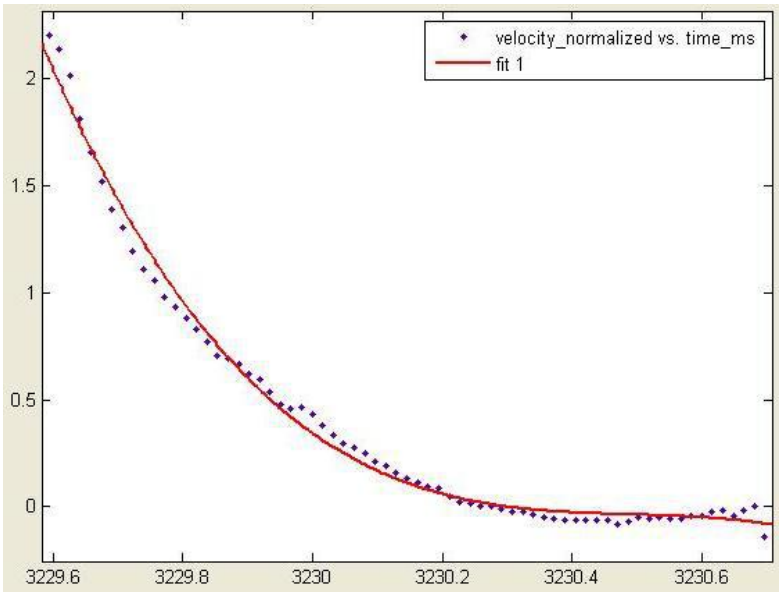

Fig. 10. Cubic Polynomial function between ball velocity and number of frames

3.4 Automated Strategy Planning

This year, we try to implement automatic strategy planning algorithm to our system. The log file from the real and simulation game is used as the input to the strategy predictor system. The output from this system is the prediction of the opponent robots duties. This information is very useful when we try to guest the opponent strategy and generate our strategy plan.

4 Conclusion

Table 1. Competition results for Skuba SSL RoboCup team

Competition	Result
RoboCup Thailand Championship 2005	3rd Place
RoboCup Thailand Championship 2006	Quarter Final
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

Our system has been continuously improving since the beginning. Last year, the auto calibration software is successfully used to reduce manpower tuning robot before every match and, in this year, we focus on an improvement of accurate ball prediction algorithm. The new ball prediction system software is still in experiment period and it will be tested in IranOpen 2011. The precise ball predictor will make the dynamic game plays possible and make currently static game plays smoother than last year. If the result comes out good enough, this predictor will be used in RoboCup 2011, Turkey. The software which runs the robot team was built in 2006 and improved each year. It has given us very successful competition results for the last several years, the results are summarized in table 1. 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. Srisabye, J., Wasuntapichaikul, P., Onman, C., Sukvichai, K., et al.: Skuba 2009 Extended Team Description. In: Proceedings of RoboCup 2009
2. Raúl Rojas,Mark Simon : Like a rolling ball , FU-Fighters , The Book – Robot Playing Soccer