RoboDragons 2012 Team Description

Kotaro Yasui, Taro Inagaki, Hajime Sawaguchi, Yuji Nunome, Hiroaki Sasai, Yuki Tsunoda, Shinya Matsuoka, Naoto Kawajiri, Togo Sato, Kazuhito Murakami and Tadashi Naruse

Aichi Prefectural University, Nagakute city, Aichi, 480-1198 JAPAN

Abstract. This paper describes a system configuration of RoboDragons, which is a team of Aichi-Prefectural University in Japan. This year, we concentrate on the software development. Main technical improvements include the algorithms of a mark defense strategy and a way how robot follow the dynamic ball in the offense strategy.

1 Introduction

In the team description paper (TDP) of RoboDragons2012, we briefly state the overview of hardware and software architecture of our team and then describe in detail about the new software development this year.

For the hardware, our current robots are the fifth generation ones in our Labs. Our teams have six robots at total. Each robot mainly consists of control unit, voltage booster, motors, wheels, dribbling device, IR sensors, kick devices, and communication device.

For the software, the base of our system is originated from the one at the time when we made a joint team with CMU in 2004 and 2005. Since 2005, many functions are added and improvements are accomplished. Then, this year, we present the algorithms of marking a opponent robot in the defense strategy and following the dynamic ball in the offense strategy.

In the following sections, we describe more details about them.

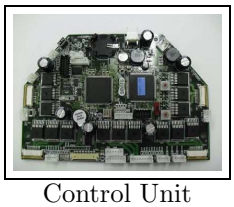
Fig. 1. Current Robot
(Left: with cover, Right: without cover)

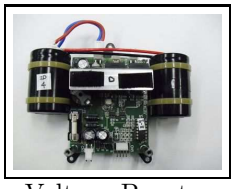
2 Hardware Architecture

In this section, we show the features of our robot. All devices attached to the robot are described as below.

- a cylinder with dimensions of 145 mm height and 178 mm diameter
- the maximum percentage of ball coverage : about 18%

Device	Description
Control Unit	CPU: Hitachi's SH2A processor with FPGA
Voltage Booster	convert from 15V DC to 150V $\sim$ 200V DC condenser has a capacity of 4500 μ charging time is about 2 sec (output voltage: 200V)
Motor	"EC 45 flat 30 W" by Maxon gear reduction ratio between motor and omni-wheel is 21:64
Wheel	4 omni-wheels, each has 15 small tires in circumference diameter: omni-wheel(56mm), small tire(13mm)
Dribble Device (dribble roller and motor)	dribble roller: 20mm in diameter and 73mm in length made of alminum shaft with silicon rubber motor: "EC 16 15W" by Maxon gear reduction ratio between motor and roller is 1:5.4
IR Sensor	3 pairs with infra-red light emission diodes and photo diodes irradiation angle of the light is about 15 degree
Kick Bar	made of a solenoid and 7075 alminum alloy solenoid: a coil winding a 0.6 mm$^{\phi}$ enameled wire (Straight) propel a ball with 11.2m/sec at maximum (Chip) fly a ball with 2m distance and 1m height at maximum
Communication Device	modem : "FRH-SD07T" by Futaba (2.4GHz Spread Spectrum)

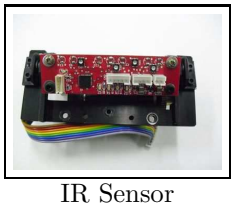

Control Unit

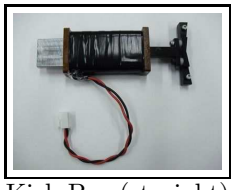

Voltage Booster


Motor


Wheel


Dribble Device


IR Sensor


Kick Bar (straight)


Communication Device

This is a data/signal flow of a control program on the hardware. The description is written in RoboDragons 2010 TDP.[2]

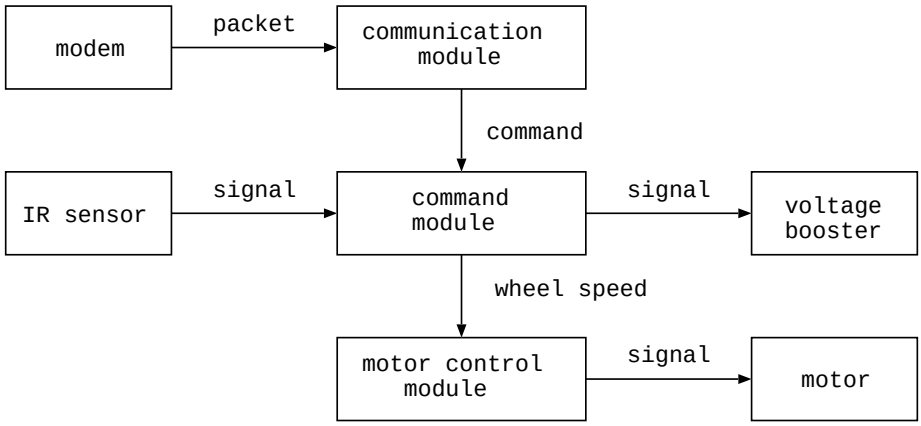


Fig. 2. Data/signal flow

3 Software Architecture

In this section, we show how our software architecture is composed and relates to the information from real world. The overview of our software system is shown as a diagram below.

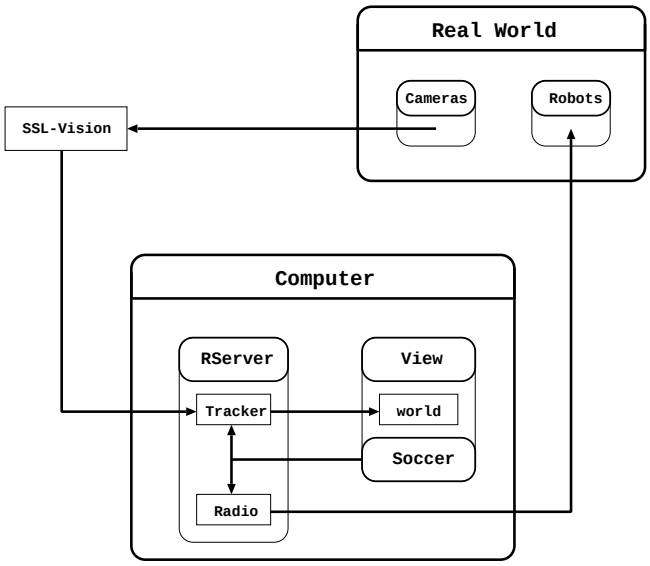


Fig. 3. Overview of Software System

The features of our host computer shown in Fig.3 are

- Intel Core 2 Duo P8400 with 2GB memory and Ubuntu 11.04/Linux OS.
- used to run 3 main modules (RServer, View, and Soccer).

(1) The *Rserver* module receives SSL-Vision data and uses tracker sub-module to predict the ball and robot states by Kalman Filter. These information are preserved to world storage, which is shared by other modules. To send a command to each robot, a radio sub-module is used.

(2) The *View* module is used to see the simulated image of real world so that users are easy to understand the situation. To do so, users set the numbers of robots and our team color.

(3) The *Soccer* module is used to make an action command for each robot. By using the information of real world, this module chooses the best strategy, gives a role to each robot, and decides a route for each robot.

4 Mark Defense Strategy

A mark defense strategy is used to defense a goal from a shoot robot in a set play. We have two kinds of tactics to exploit its strategy; one(ShootCut) is to prevent the goal by locating a mark defense robot on the shoot path, and the other(PassCut) is to intercept the ball on the pass.

4.1 Algorithm of ShootCut

1. allocate a defense robot to each shooting robot
2. process to mark a shooting robot R_{S} by the corresponding defense robot R_{D}

   S**:** a coordinate on the position kicked by R_{S}
   G**:** a coordinate on the mid point of our goal
   D**:** a coordinate on the center position of R_{D}

   (a) calculate the equation (1)

$$T = \frac{\boldsymbol{p} \cdot \boldsymbol{q}}{|\boldsymbol{p}|^2} \cdot \boldsymbol{p} + G \tag{1}$$

   (b) if $|\overrightarrow{GS}| < |\overrightarrow{GT}|$, then $T \leftarrow S$

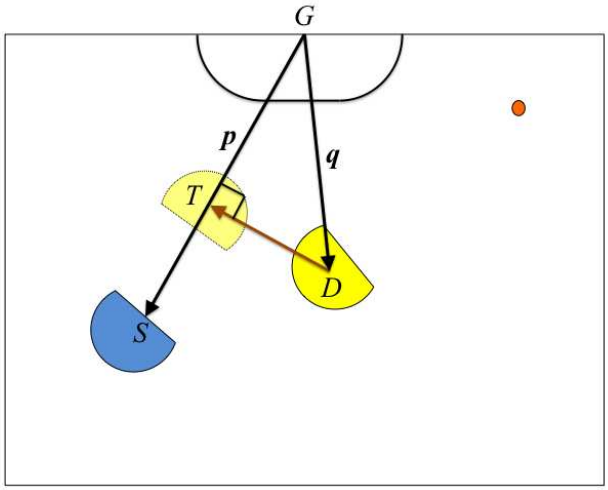


Fig. 4. A way to prevent the goal by ShootCut

In the Fig.4, $\boldsymbol{p} : \overrightarrow{GS}$ and $\boldsymbol{q} : \overrightarrow{GD}$

(c) If the distance between T and D < threshold distance t_1, go to (d). If not, go to (e).
(d) calculate T by a equation (2) as shown below.

$$T \leftarrow d_1 \frac{\boldsymbol{p}}{|\boldsymbol{p}|} + T \tag{2}$$

d_1 is a distance approaching to R_{S} and the given constant value
(e) target location of R_{D} is calculated as T

4.2 Algorithm of PassCut

The algorithm of PassCut is almost as same as ShootCut, but change G to B, where B is a center coordinate of ball location. After the calculated equation (1) is shown as a diagram below.

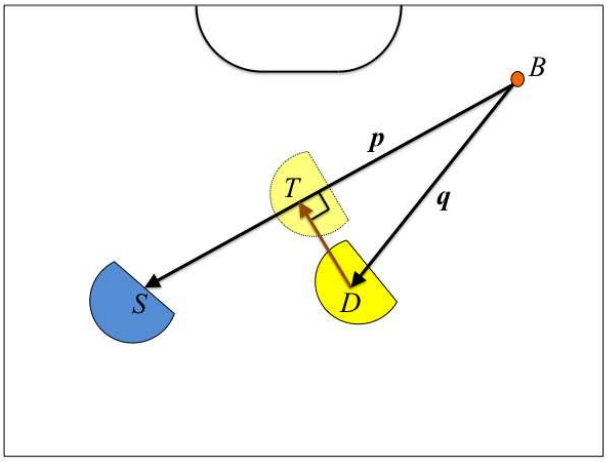


Fig. 5. A way to prevent the goal by PassCut

In the Fig.5, $\boldsymbol{p}$: $\overrightarrow{BS}$ and $\boldsymbol{q}$: $\overrightarrow{BD}$

5 The algorithm of a robot following a ball in dynamic

1. **Definition of the variables shown as below (refer to the figure 6).**

B**:** a coordinate on the present ball position
R**:** a coordinate on the present robot position
$\boldsymbol{V_B}$**:** a vector of the present ball velocity
$\boldsymbol{V_R}$**:** a vector of the present robot velocity
G**:** a coordinate on the shooting target position
L**:** an offset length which is not to hit the ball
(the less distance between B and R, the less length of L)
D_{B}**:** a constant ball decceleration, which takes account of a friction
$V_{\max}$**:** a max velocity of the robot
A_{R}**:** a max acceleration of the robot
D_{R}**:** a max decceleration of the robot

Let make an assumption that a robot can reach the ball in T second.

$\boldsymbol{V_B'}$**:** a vector of the ball velocity after T second
$$\boldsymbol{V_B'} = \boldsymbol{V_B} - D_{\mathrm{B}} T \frac{\boldsymbol{V_B}}{|\boldsymbol{V_B}|}$$
$\boldsymbol{V_R'}$**:** a vector of the robot velocity after T second
B'**:** a coordinate on the ball position after T second
$$B' = \frac{1}{2}(\boldsymbol{V_B} + \boldsymbol{V_B'})T + B$$
R'**:** a coordinate on the robot position after T second
$$R' = \frac{\overrightarrow{GB'}}{\left|\overrightarrow{GB'}\right|} \cdot L + B'$$

$\boldsymbol{p}$**:** a unit vector of $\boldsymbol{V_B}$ $(\frac{\boldsymbol{V_B}}{|\boldsymbol{V_B}|})$
$\boldsymbol{q}$**:** a vector of rotating $\boldsymbol{p}$ by $\pi/2$(rad) on counter-clockwise
when $\boldsymbol{p} = (p_1, p_2)$, $\boldsymbol{q} = (-p_2, p_1)$
$\boldsymbol{d_R}$**:** a vector from R to R'

The subscript $(\boldsymbol{p})$, $(\boldsymbol{q})$ on the right top of $\boldsymbol{V_R}$, $\boldsymbol{V_R'}$, $\boldsymbol{d_R}$ is a component of the vector $\boldsymbol{p}$ and vector $\boldsymbol{q}$ of $\boldsymbol{V_R}$, $\boldsymbol{V_R'}$, $\boldsymbol{d_R}$ respectively.

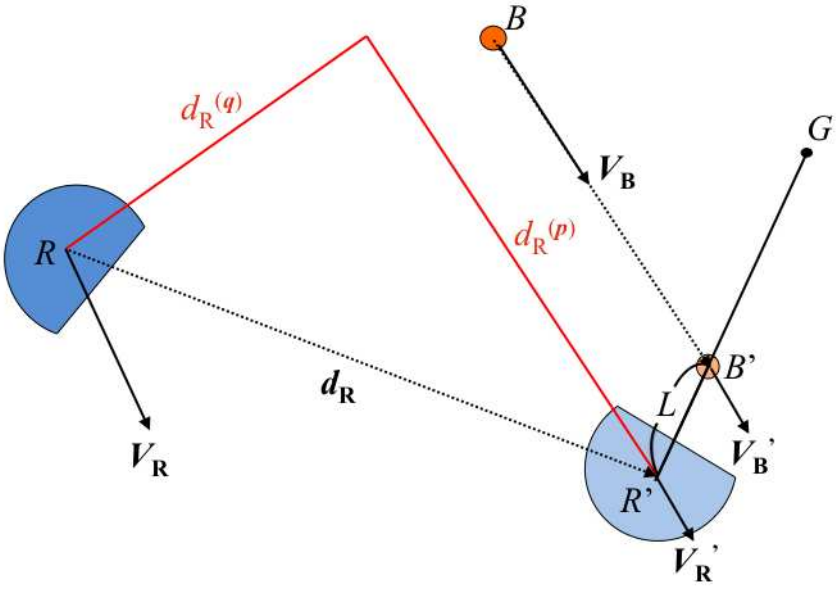


Fig. 6. Relation to the variables used in the algorithm

2. **Explanation of the algorithm**

In order to move the robot toward $R^{'}$, we desolve the robot velocity vector into components of a vector $\boldsymbol{p}$ and $\boldsymbol{q}$ directions.

the component of $\boldsymbol{p}$: control the robot velocity by making a motion profile as $V_R^{'(\boldsymbol{p})}$ to be $|\boldsymbol{V_B'}|$ when the robot moved $d_R^{(\boldsymbol{p})}$. [1] (step(3))
the component of $\boldsymbol{q}$: control the robot velocity by making a motion profile as $V_R^{'(\boldsymbol{q})}$ to be 0 when the robot moved $d_R^{(\boldsymbol{q})}$. (step(4))

From the information so far, if T is given, we can make a motion profile of the robot for $\boldsymbol{p}$ and $\boldsymbol{q}$ directions in turn, because we can calculate $B^{'}$, $R^{'}$, $\boldsymbol{d_R}$, $d_R^{(\boldsymbol{p})}$, and $d_R^{(\boldsymbol{q})}$ one after another. On the contrary, if the motion profile of the robot for a direction $\boldsymbol{p}$ is made, we can calculate T. Therefore, we suggest the method, which calculates T and makes a motion profile for the direction $\boldsymbol{p}$ at the same time.

[1] The reason why we are doing this is to make more oppotunity of kicking the ball and accomplish the kick safety.

3. **A method of making the motion profile for the direction $\boldsymbol{p}$**

Due to limitations of spaces, we assume the precondition below.

$$\{d_{\mathrm{R}}^{(\boldsymbol{p})} \geq d_{\mathrm{R}}^{(\boldsymbol{q})}\} \wedge \{V_{\mathrm{R}}^{(\boldsymbol{p})} \geq |\boldsymbol{V_{\mathrm{B}}}|\}$$

Make a motion profile shown as a figure below.

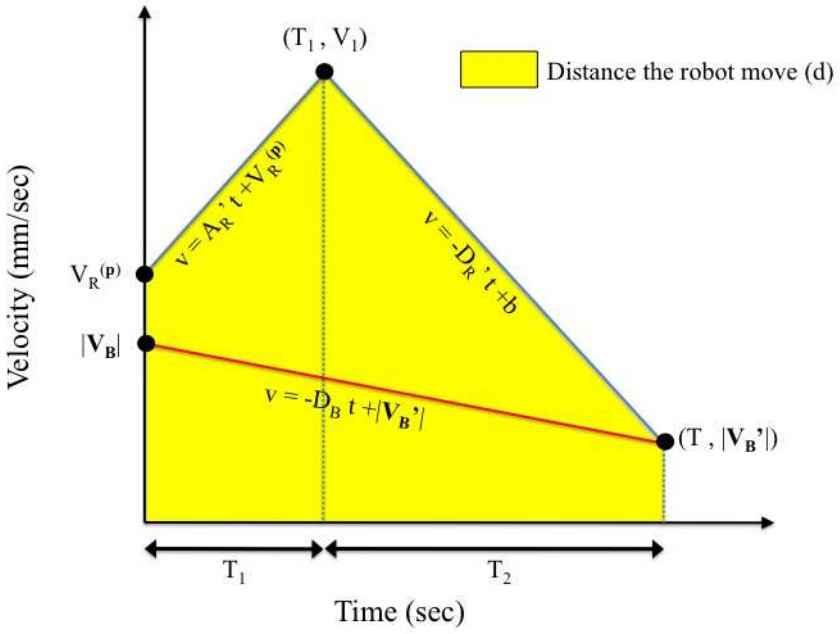


Fig. 7. Motion Profile

At this point, $A_{\mathrm{R}}^{'} = \frac{1}{\sqrt{2}} A_{\mathrm{R}}$, $D_{\mathrm{R}}^{'} = \frac{1}{\sqrt{2}} D_{\mathrm{R}}$. In the figure 7, unknown variables ($|\boldsymbol{V_{\mathrm{B}}^{'}}|$, b, T_1, T_2, V_1, d) are calculated in succession if T is provided.
Then, if $d = d_{\mathrm{R}}^{(\boldsymbol{p})}$, we can succeed to make $V_{\mathrm{R}}^{'(\boldsymbol{p})}$ to be $|\boldsymbol{V_{\mathrm{B}}^{'}}|$ when the robot moved the distance $d_{\mathrm{R}}^{(\boldsymbol{p})}$. Therefore, it is important to give T and T is calculated by bisection method, Newton's method[1], and so on. The initial value of T in its method is, for instance, the time when the ball is just stopped by a constant ball decceleration D_{B}. If $V_1 > V_{\mathrm{max}}^{'}$ $(= \frac{1}{\sqrt{2}} V_{\mathrm{max}})$, then make the motion profile including a time when the robot velocity is a constant $V_{\mathrm{max}}^{'}$.

4. **A method of making the motion profile for the direction $\boldsymbol{q}$**

   From step(3), the time when the robot can reach the ball (T) is calculated, so a robot position willing to move after T second ($R^{'}$) can be calculated by step(1). Then from $R^{'}$, a distance between R and $R^{'}$ for a $\boldsymbol{q}$ direction ($d_{\mathrm{R}}^{(\boldsymbol{q})}$) can be calculated. Therefore, we can make a motion profile as $V_{\mathrm{R}}^{'(\boldsymbol{q})}$ to be 0 when the robot moved $d_{\mathrm{R}}^{(\boldsymbol{q})}$. [2] To make the motion profile, we use the maximum acceleration of the robot as $A_{\mathrm{R}}^{'}$, the maximum decceleration of the robot as $D_{\mathrm{R}}^{'}$, the maximum velocity of the robot as $V_{\mathrm{max}}^{'}$.

5. **A robot velocity after sending frame period from step(3) and (4)**

   From step(3) and (4), a motion profile for each direction of $\boldsymbol{p}$ and $\boldsymbol{q}$ is provided. Then what we want to do is to calculate a robot velocity after sending frame period ($\boldsymbol{V_{RS}}$) in order to move the robot toward $R^{'}$. To do so, look for the corresponding robot velocity for each direction where $t =$ sending frame period from the motion profile. We call the robot velocity searched for $\boldsymbol{p}$ direction as $V_{\mathrm{RS}}^{(\boldsymbol{p})}$, $\boldsymbol{q}$ direction as $V_{\mathrm{RS}}^{(\boldsymbol{q})}$. Then, send to the robot for a vector $\boldsymbol{V_{RS}} = V_{\mathrm{RS}}^{(\boldsymbol{p})} \cdot \boldsymbol{p} + V_{\mathrm{RS}}^{(\boldsymbol{q})} \cdot \boldsymbol{q}$.

[2] This is a most commonly used method for making a motion profile of the robot velocity.

6 Conclusion

In this paper, we described our hardware and software architecture. This year, especially, we try to improve the software strategy: the algorithms of marking the opponent robot in defense and following the dynamic ball in offense. From this algorithms, we think the marking defense in set play and the offense without set play will improve in the game.

7 acknowledgement

This work was supported by the cheif director's special study fund of Aichi Prefectural University and the president's special study fund of Aichi Prefectural University.

References

1. "Newton's Method" <http://www.math.montana.edu/frankw/ccp/calculus/numerical/newton/learn.htm>
2. Akeru Ishikawa, Takashi Sakai, Jousuke Nagai, Taro Inagaki, Hajime Sawaguchi, Yuji Nunome, Kazuhito Murakami and Tadashi Naruse "RoboDragons 2010 Team Description", 2010
3. Taro Inagaki, Hajime Sawaguchi, Akeru Ishikawa, Kotaro Yasui, Tomomi Yasui, Yuji Nunome, Kazuhito Murakami and Tadashi Naruse "RoboDragons 2011 Team Description", 2011