Basic Requirements for a Teamwork in Middle Size RoboCup

M. Jamzad[1], H. Chitsaz[1], A. Foroughnassirai[2], R. Ghorbani[2], M. Kazemi[1], V. S. Mirrokni[1], and B. S. Sadjad[1]

[1] Computer Engineering Department
Sharif University of Technology, Tehran, Iran
jamzad@sina.sharif.ac.ir/~ceinfo
{chitsaz,nassirae,ghorbani,kazemi,mirrokni,sadjad}@linux.ce.sharif.ac.ir
[2] Mechanical Engineering Department
Sharif University of Technology, Tehran,Iran.
http://www.sharif.ac.ir/~mechinfo

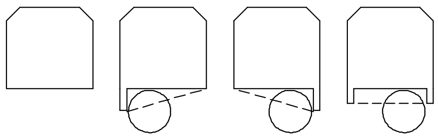
Abstract. In this paper we describe some mechanical, hardware and software aspects of our robots specially used in teamwork. The pneumatic design of the gripers and kicker enables the robot to make a good control of ball when dribbling and also passing the ball in short and long distances. The teamwork software enables our robots to perform a cooperative behavior by dividing the field to three major regions and assigning each robot a role such as defender, forward and middle in these three regions. The server can order the robots to change their role when needed and make them return to their original state after a short time.

1 Introduction

In a successful soccer robot play, all robots should behave in a cooperative manner [3]. This can be done if each robot has some minimum performance capabilities. Which is, they must have the mechanical capability to pass the ball in short and long distance, good dribbling ability and some sensing devices to inform the robot of the special situation going on around it. In addition, the software should be able to implement a cooperative behavior among robots with a certain strategy. In this paper we describe some aspects about mechanical design of our robots, sensing devices such as touch sensors and infrared sensors installed around robot body and finally we give a brief description of our software for teamwork.

2 Robot Pneumatic System

The mechanical design, and controlling software of our robot, enables it to dribble the ball in various situations [1]. In practice, there are some cases that fixed position griper will be a limit to fast dribbling. Therefore we designed variable length gripers such that each griper can possess 3 lengths: $0, 6.6$ or $9.5cm$ as shown in figure 1. During the game, when the robot does not have the ball, the gripers are in position (a), have length zero. The reason for this is because the robot size will become smaller and it can maneuver around safer. Gripers are set to have length $6.6cm$ in position (d), when the robot is moving with the ball in its gripers.



(a) (b) (c) (d)

Fig. 1. Ball position in robots' gripers in four situations

Our robots can rotate left or right round any point on the field or on the robot body, while holding the ball within its gripers [1]. However, if both gripers are fixed in $6.6cm$ in fast rotations the robot loses the ball. In these situations we make the gripers to stay in position b or c where gripers have length 0 and $9.5cm$. The displacement of each griper is done by a pneumatic actuator with a maximum course of $10cm$.

To enable the robot to perform a reliable teamwork by means of short or long and quick passes, the kicking power should be controlled. We used a pneumatic actuator with a course of $3cm$, where, with a special mechanical design, the actuator linear displacement is transformed to an arm movement (the kicker). The maximum speed of kicking is 3 to 4 $\frac{m}{sec.}$. By this mechanism, we can make the robot pass the ball with desired speed. This is done by changing the settings of time interval in turning on and off the solenoid valve of pneumatic actuator.

3 Robots' Sensing Devices

Infra-red (IR) Sensors, Touch Sensors, and a Video Camera are used in our robots. There are 12 IR sensors around the robot, two of which are on the two gripers. Each IR sensor has a transmitter and a receiver. Each sensor has a limited and straight view sight and outputs 1 if there is anything within its 10cm distance in its sight of view and outputs 0 otherwise. Also, we use 24 touch sensors around the robot two of which are in front of the two gripers. Finally, the main sensor in our robots is the Sony Handycam. We use a low-price frame-grabber based on BT878 chip in our robots to capture the image.

4 Some of Robots' Algorithms

4.1 Algorithms for going to a point with dribbling

One of the aspects, that is important for robot movement, is its ability to go to a certain destination while doing obstacle avoidance. In the following, we describe how our robots can perform this task.

– *Finding wheels setting for different velocities:*
  In our software we defined an instruction named $go\_angle\_distance(\alpha, d)$ for setting the velocity of wheels. By changing the parameters of this instruction,

we can force the robot to make a curved shape move toward a point which is in distance d from the current position of the robot. This move is done in such a way that when the robot has reached the desired point, its front side would have an angle α with respect to the robot front in its initial stage, as it is shown in figure 2. We found the PWM settings for the left and right wheel motors to make the robot move certain distances with fixed angles. The PWM setting for all other distances and angles were calculated by linear interpolation of the above settings.

– *Tracing an object and go toward it:*
  The following loop shows how the robot can go toward an object o:

```
while (distance(o) > near_distance){
  if (distance(o) < near_distance + 10(cm))
    go_angle_distance(angle(o), distance(o));
  else
    go_angle_distance(angle(o), near_distance + 10(cm));
}
```

  Where distance(o) and angle(o) are the distance and angle from an object o and near_distance is a constant used for close distances.

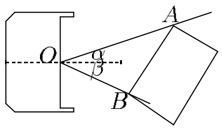


Fig. 2. Left and right angles of our robot with respect to an opponent robot.

– *Dribbling other robots*
  A robot should be able to dribble other robots which are within a close distance to it. In our dribbling procedure, we assume a bounding box around the opponent robot that should be dribbled [2]. As it is seen in figure 2, α and β are the left and right angles of our robot with respect to the left and right side of the opponent robot. The dribbling procedure is described in the following general code:

```
while (distance(o) > near_distance){
  // near_robot_dist means the close distance within which we dribble.
  if((distance(robot)<near_robot_dist)&&(distance(robot)<distance(o))){
    if (fabs(left_angle(robot)) < fabs(right_angle(robot)))
      go_angle_distance(left_angle(robot) - 15, distance(robot));
      // It is easier to dribble from left side angle
    else  go_angle_distance(right_angle(robot) + 15, distance(robot))
  }
  else // The same code as in "Tracing an object and ..." section.
}
```

Fig. 3. A picture of our player robots.

4.2 Teamwork

We have virtually divided the field into regions where each region can be the territory of one or more robots. Our robots act as a forward, a middle and a backward in pre-defined territories. The server implements dynamic task assignment to all robots to make sure each robot remains in its own territory. In practice depending on the game situation, robots can change their role, which is, a defender can become a forward if needed and then return to become defender. The most efficient method for task assignment is dynamic task assignment. In this method, server assigns the tasks to the robots so that each robot will be closer to its own territory. In our teamwork strategy we consider the following main situations: (a) None of our robots can see the ball. (b) At least one of our robots can see the ball, and the ball is not in their control. (c) One of our robots is carrying the ball.

In situation (a), all robots will go to their original region after some time and start searching for the ball in those regions. In case (b), the server selects the robot which is closer to the ball and order it to go for the ball. Other robots which are seeing the ball, will go toward the ball, but stay away from it within a certain distance. This behavior will prevent our robots to fight for the ball and make enough room for only one robot to control the ball. In situation (c), when one robot is carrying the ball, at first, the other two robots will trace the ball and then after some time, they will go to the backward and middle regions. In situations when no robot can see the ball, the server can inform the last position of the ball to the robot which is closer to it and order it to go for the ball in that region. We use a LANEscape WL2420 wireless LAN card working in 2.4GHz frequency as the communication mean and an Ad-Hoc architecture. However, we think a good team work requires a perfect localization method. Figure 3 shows a picture of our player robots.

References

1. M. Jamzad, A.Foroughnassirai, et al, *Middle sized Robots: ARVAND*, RoboCup-99: Robot Soccer world Cup II, Springer (2000), pp 74-84 .
2. M. Jamzad, et al, *a fast vision system for middle size robots in RoboCup*, submitted to RobCup Symposium, Seattle-2001.
3. S. Coradeschi, et al, *Overview of RoboCup-99*, AI magazine, Vol. 21, No. 3 (2000).