Middle Sized Soccer Robots: ARVAND

M. Jamzad[1], A. Foroughnassiraei[2], E. Chiniforooshan[1], R. Ghorbani[2], M. Kazemi[1], H. Chitsaz[1], F. Mobasser[1], and S.B. Sadjad[1]

[1] Computer Engineering Deptartment
Sharif University of Technology
Tehran, Iran.
jamzad@sina.sharif.ac.ir, {chinif, chitsaz, sajjad}@linux.ce.sharif.ac.ir, {mobasser, kazemi}@ce.sharif.ac.ir
http://www.sharif.ac.ir/~ceinfo
[2] Mechanical Engineering Deptartment
Sharif University of Technology
Tehran, Iran.
{ghorbani, forough}@linux.ce.sharif.ac.ir
http://www.sharif.ac.ir/~mechinfo

Abstract. **Arvand** is the name of robots specially designed and constructed by sharif CE team for playing soccer according to RoboCup rules and regulations for the middle size robots. Two different types of robots are made, players and the goal keeper. A player robot consists of three main parts: mechanics (motion mechanism and kicker), hardware (image acquisition, processing unit and control unit) and software (image processing, wireless communication, motion control and decision making). The motion mechanism is based on two drive unit, two steer units and a castor wheel. We designed a special control board which uses two microcontrollers to carry out the software system decisions and transfers them to the robot mechanical parts. The software system written in C++ performs real time image processing and object recognition. Playing algorithms are based on deterministic methods. The goal keeper has a different moving mechanism, a kicker like that of player robots and a fast moving arm. Its other parts are basically the same as player robots. We have constructed 3 player robots and one goal keeper. These robots showed a high performance in Robocup-99: became champion.

1 Introduction

In order to prepare a suitable level for research in many different aspects involved in autonomous robots, we designed and constructed all parts of the robots by our group members in different laboratories of our university. These robots which are the 2nd generation which we made in the last two years, have a controllable speed of maximum 0.53 m/sec. In addition to the basic movements of a robot, special mechanical design of the player robot, enables it to rotate around any point in the field. In practice, the distance between ball center and robot geometrical center is calculated and the robot can be commanded to rotate around the ball center until seeing the opponent team goal. This unique mechanics, to a good extent, simplified and accelerated our playing algorithms.

The machine vision system of player robots uses a widely available home use video camera and a frame grabber. But for goal keeper we used one CCD camera in front and two small video conferencing digital cameras in sides rear. Our fast image processing algorithm can process up to 16 frames per second and recognize objects in this speed. For any recognized object, its color, size, distance and angle from robot is determined. The wireless communications between robots made it possible to test the cooperative behavior in a multi-agent system in a real-time changing environment. TCP/IP protocol was used for communication.

The software is based on deterministic algorithms, designed in object-oriented method and implemented in C++ using DJGPP compiler in MS/DOS. The reason for using MS/DOS was mainly due to the fact that we had to use a floppy disk drive for booting the system because of its reliability in a moving robot in RoboCup environment and also its low price compared to hard disk.

In the following we describe the mechanics, hardware and software systems used for goal keeper and player robots.

2 Mechanical Architecture

According to the motion complexity of a soccer player robot, proper design of its mechanics can play a unique role in simplifying its motion and as a result the playing algorithms. In this regard, different specific mechanisms were designed and implemented for player and goal keeper, that together with the motors current feedback measurement, to a good extent, guided us to the current mechanism which showed a better performance in Robocup-99.

2.1 Player Robot Motion Mechanism

Arvand consists of two motion units in front of the robot and one castor wheel in the rear. Each motion unit has a drive unit and a steer unit. A drive unit is responsible for rotating its wheel in forward and backward directions and also, a steer unit is responsible for rotating its respective drive unit around the vertical axis of the drive unit wheel. The combination of drive unit movement and proper settings of steer units angles with respect to robot front, provides the robot with a continuous rotational move around any point (this point can be selected to be inside or outside robot body) in the field in clockwise or counter-clockwise direction.

Drive unit consists of a wheel which is moved by a DC motor and a gearbox of 1:15 ratio [1]. The steer unit uses a DC motor and a gearbox of 1:80 ratio. For controlling a steer unit, an optical encoder is mounted on the respective motor shaft and its resolution is such that one pulse represents 0.14 degrees of drive units rotation. Figure 1 is from the robot top view and shows the position of drive units for rotating around point A in the field. The coordinates are as shown in the figure 2.

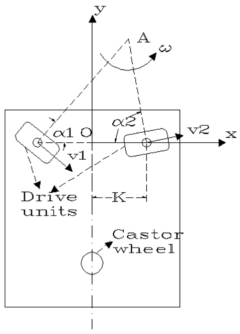

Fig. 1. The position of drive units to make the robot move around point A in the field.

The velocity vectors and angles made by steer units are calculated according to following formulas [2]:

$$v_1 = w \times \sqrt{{y_1}^2 + (k + x_1)^2} \tag{1}$$

$$v_2 = w \times \sqrt{{y_1}^2 + (k - x_1)^2} \tag{2}$$

$$\alpha_1 = Arctg(\frac{y_1}{k + x_1}) \tag{3}$$

$$\alpha_2 = Arctg(\frac{y_1}{k - x_1}) \tag{4}$$

In the above equations, x_1 and y_1 are the coordinates of point A (i.e. the rotation center); k is the distance between y axis and the drive unit rotation center; w is the angular speed of robot around point A; α_1 and α_2 are the rotation angles of left and right drive units with respect to x axis; v_1 and v_2 are the speeds of left and right drive motors, respectively. In special case, if the rotation center A is located on the y axis, equations 1 to 4 summarize to the following equation:

$$v_1 = v_2 = w \times \sqrt{{y_1}^2 + k^2} \tag{5}$$

$$\alpha_1 = \alpha_2 = Arctg(\frac{y_1}{k}) \tag{6}$$

This means that, to rotate the robot around a point (0,y_1), both drive units should be set by the same angle α_1 and then , they should move by the same velocity v_1. As a result the robot will rotate with angular speed of w around the point (0,y_1).

In summary, this mechanism has the following capabilities:

1. Rotating around any point in the field. Appropriate rotation of steer units can bring the drive units in desired angular positions α_1 and α_2. After these angles setting, if the ratio between the angular speed of two drive units is set according to equation 7 (extracted from equations 1 and 2), as a result the robot will rotate around point A.
$$\frac{w_1}{w_2} = \frac{\sqrt{{y_1}^2 + (k + x_1)^2}}{\sqrt{{y_1}^2 + (k - x_1)^2}} \tag{7}$$
In the above formula, w_1 and w_2 are the angular speed of left and right drive units. By setting one of w_1 or w_2, the other is calculated according to above equation.
2. In our software system we can set the drive units to be parallel to each other while having a specific angle related to robot front. This mechanism is useful for taking out the ball when stuck in a wall corner and also dribbling other robots.
3. A kicker arm is installed in front of robot. A solenoid is used to supply it with kicking power. A simple crowbar connects the solenoid to the kicking arm. The power of kicking is controlled by duration of 24 DC voltage applied to the soleniod.

2.2 Goal Keeper Motion Mechanism

We think the goal keeper should have a complete different mechanism from player robot. Because it keeps the goal, it seems that more horizontal speed in front of goal area and deviation-less movement is a great advantage for the goal keeper. Thus, in order to guarantee a nearly perfect horizontal movement for the goal keeper, 4 drive units are installed in the robot (the castor wheel has been eliminated because it causes deviation in the robot movements). However, in practice the robot will be displaced after some movements, therefore it should have the ability to adjust itself when displaced. Horizontal movements and self adjustment can be done by a combination of the following three basic movements:

1. Move forward and backward (Fig. 2-a).
2. Rotate around its geometrical center (Fig. 2-b).
3. Move straight towards left and right (Fig. 2-c).

In order for the robot to perform these movements, 4 drive units and two steer units are installed in the robot. One steer unit rotates two front drive units round their vertical axes simultaneously in opposite directions and the other steer unit does the same on two rear drive units. The drive units wheel has a diameter of 8 Cm and a gearbox of 1/15 ratio. Measurement of the rotation angle for drive units is done by encoders installed on steer unit motor shafts.

To minimize the adjustment movements and also increase goal keeper performance, we installed a fast moving sliding arm on it, such that this arm can slide in left or right direction before the robot body itself moves in these directions. This arm can slide to its leftmost or rightmost position within less than 0.1 of

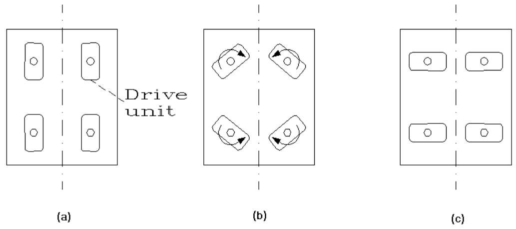

Fig. 2. Goal keeper drive units

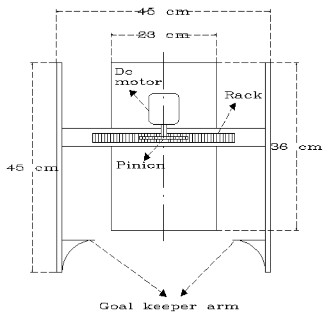

Fig. 3. Sliding arm movement

a second. Considering the front body size of goal keeper which is 23 Cm, and the arm size which is 45 Cm, the robot can cover 68 Cm which is approximately 1/3 of the goal area, within less than 0.1 of a second. Compared to goal keeper maximum speed which is 75 Cm/sec, this arm gives better protection of goal area from very fast moving balls.

Sliding arm movement is carried out by a rack and pinion mechanism, as seen in Fig. 3. To control the amount of arm sliding, an encoder is mounted on the shaft of pinion motor. It is necessary to fix the arm when goal keeper is in a stuck situation with other robots. This is done by using a solenoid which can lock the arm in its present position.

3 Hardware Architecture

The goal of our hardware architecture is to provide a control unit independent of software system as much as possible and also reduce the robots mechanical errors.

Arvand hardware system consists of three main parts: Image acquisition unit, processing unit and control unit.

The image acquisition system of goal keeper consists of a Topica PAL color CCD camera with 4.5 mm lens in front and two digital Connectix Color Quick-Cam2 for the sides rear view. For other robots we used a widely available home use video camera in front which could record the scene viewed by robot too. All robots including the goal keeper used a PixelView CL-GD544XP+ capture card which has an image resolution of 704x510 with the frame rate of 25 frames per second.

The processing unit consists of an Intel Pentium 233 MMX together with a main board and 32MB RAM. Two onboard serial ports are used as communication means with the control unit. A floppy disk drive is installed on the robot from which the system boots and runs the programs.

The control unit senses the robot and informs the processing unit of its status. It also fulfills the processing unit commands. Communication between the control unit and the processing unit is done via two serial ports with RS-232 standard[3]. Two microcontrollers 89C52 and 89C51 [4] are used in control unit. They control the drive units, steer units and kicker. Two limit switches are mounted on each steer unit. Microcontroller counts the number of pulses generated by the encoders mounted on the motor shafts to control the drive unit rotation. Each pulse represents 0.14 degrees of the drive unit rotation. The motor speeds are controlled by PWM pulses with frequency of about 70kHz. Fig. 4 shows the block diagram of the control unit. It allows distributed processing among the main board processor and two processors on this board.

This board performs the following main tasks:

- PWM generation on two drive units, two steer units and also the kicker. PWM control of drive units is done by MOSFET. A relay is used for changing motor rotation direction. In order to control the steer units not to rotate

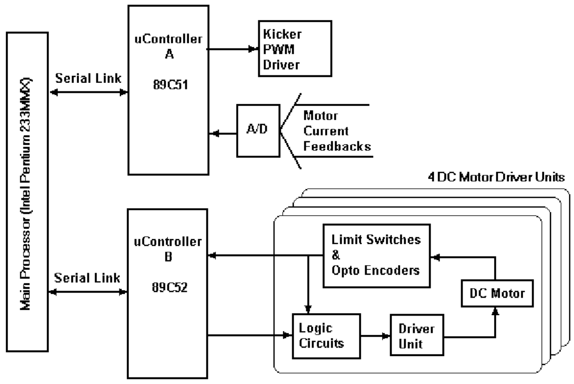

Fig. 4. Player robot hardware components interconnection diagram

beyond their angular limits, two limit switches are installed. If a steer unit touches a limit switch, no more movement in that direction will be possible.
- Measurement the current feed back of drive units. Motor current is measured by an A/D and processed by software in the main processor.
- Measurement of the batteries voltages in order to find their charging level.
- Control of goal keeper sliding arm and its relating lock.
- A pause key on control unit board allows us to stop all movements of robot and also send a signal to the main processor to bring it to suspend mode, which will save batteries.

The control unit board is designed to be robust, easy to test, fast failure finding, easy maintenance and modification, and reliable performance. In addition, it has the capability to handle extra signals which could come from new sensors on the robot, in the future.

4 Software Architecture

Software architecture of **Arvand** consists of four main parts: Real time object recognition, Motion control, Communication and Decision making module. Software which is written in C++ has an object oriented design with 5 classes as follows: Camera class (all related functions for working with frame grabber), Image class (machine vision functions), Motion class (motion functions which is the interface between software and hardware), Communication class (all TCP/IP related networking) and Decision class (all robot playing methods and algorithms).

4.1 Real Time Object Recognition

Objects are detected according to their color. We used HSI[5] color model for recognizing colors, because of its advantage in representing approximately each color in a cube in the HSI space. The color output of our frame grabber board is in RGB. To reach a near real-time speed in color processing, HSI color space is constructed from RGB in off-line. For each color to be recognized, its HSI range is determined in off-line as well (i.e. this range can be set for all colors according to the lighting condition). A one dimensional array of size 65536 that shows all possible RGB input values in our system is filled with a color name according to its HSI range, in off-line. Therefore, in real time, the color of a pixel is determined by two single array access (once for finding the RGB value of a pixel from frame grabber memory and the second time for finding the color name from the above mentioned array). Due to RoboCup regulation, each main object such as ball, ground, wall and a goal is assumed to have a single predefined color. This routine generates a segmented image matrix such that all pixels belonging to an object are assigned the same color name.

To find all objects in a scene, the image matrix is processed from top to bottom only once. To speed up this routine, instead of examining each single pixel in the image matrix, only one pixel from each subwindow of size $w_i \times h_i$ is selected and tested (i.e. w_i and h_i are the minimum width and minimum height of an object which can exist in a scene). If this pixel has the desired color, then we move upward in one pixel step until hitting a border point. At this point a contour tracing algorithm is performed and the contour points of the object are marked.

To find the next object, the search is continued from a subwindow located to the right of the subwindow in which the start point of the previous object was found. In searching for the next object, the marked points are not checked again. At the end of this routine, all objects are determined.

To overcome the possible color error of image acquisition system, during moving on the object contour, if it reaches a pixel with a color different from that of the object, but 3 of its 4 neighbors have the object color, then that pixel color is changed to the color of the object and it is considered to be a contour point. In addition, during contour tracing algorithm, the minimum and maximum x, y coordinates of contour points are calculated. The extracted object fits in a rectangle which upper left and lower right corner have the (min_x, min_y) and (max_x, max_y) coordinates. The size of this rectangle is estimated to be proportional to the real object size. If the object size is smaller than a predefined size, it is taken as a noise and eliminated. However, if because of lighting condition, one or more objects are found inside a larger object with the same color, the smaller objects are considered as noise and deleted. For any object found, its size, color, angle and distance from robot camera are passed to the decision making routine.

4.2 Object Distance Calculation

The object distance from camera can be determined in two methods. In the first method, since in RoboCup, the real size of objects are known (except that of opponent robot which can be estimated before the game), the object distance is calculated as a ratio of its real size and the size calculated in object detection routine. However, this method works only if the robot sees an object completely (there are many situations where only part of an object is visible).

In the second method, the distance is calculated from the object position in the image matrix. This method is independent from detected object size and therefore has less error. We calculated the object distance D according to the following formulas. In these formulas, X_0 is the number of pixels between the image matrix bottom position to the point that has lowest y value in the object selected from image matrix. $YSIZE$ is the image height in number of pixels. The constant parameters H, A and B are calculated off-line. Where, H is distance from camera focal lens center to ground. A is the distance such that if the object is located there, then the object bottom is seen in the lowest part of the image. B is the distance such that if the object is located at that position, then its bottom is seen in the image center. Fig. 5 shows the relation between these parameters.

$$b = Arctg(\frac{B}{H})$$
$$L = (B - A) \times Sin(\frac{\pi}{2} - b)$$
$$K = \sqrt{H^2 + A^2 - L^2}$$
$$X = L(1 - \frac{2X_0}{YSIZE})$$
$$a = Arctg(\frac{X}{k})$$
$$D = H \times Tan(a + b)$$

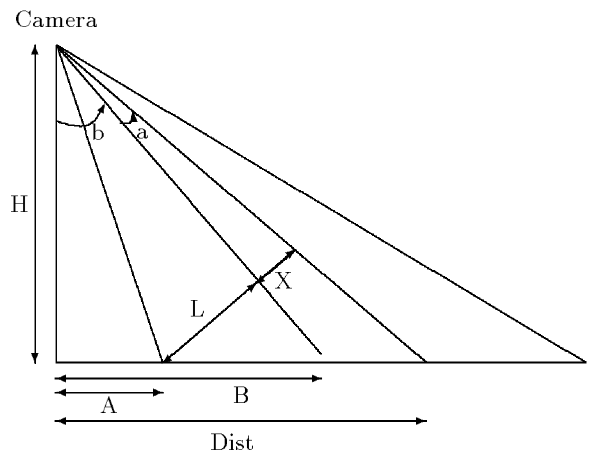

Fig. 5. Geometrical relations for finding object distance

4.3 Motion Control

This module is responsible for receiving the motion commands from the "Decision Making Module" and make the robot move. As it is mentioned in the hardware architecture section, the communication between the processing unit and the control unit is via two on-board PC serial ports using RS-232. So, just some basic computations are done in this module and commands are sent via serial ports to microcontrollers where they are executed.

For example, some commands for player robot movement are go(forward), go(backward), rotate(left), rotate(right), rotate_round(left, 10) (i.e. this stands for rotation around a point 10 centimeters straight from the robot geometrical center), kick (i.e. kicks the ball) and etc.

4.4 Communication

Communication between robots is done by wireless LAN under TCP/IP protocol. We used WATTCP whose main kernel can be downloaded from [6]. Each robot has a wireless network adapter, and there is a computer, we named it message server, outside the field which processes messages of robots and coordinates them. The server provides a useful user interface to command robots manually. Server's main responsibility is to receive the robots messages and inform them of each robot status. For example, in our multi-agent system, if one robot takes control of the ball, it will inform all others via server, and then other robots will not go for the ball.

4.5 Decision making

Principally, the decision making module is that part of **Arvand** software that processes the results of real time object recognition, decides accordingly and finally commands the motion control software. We have taken deterministic approach in these routines. This module is a finite state machine (FSM) whose inputs for changing state are machine vision results, motion control hardware feedbacks and server messages. Each robot playing algorithm kernel finds the ball, catches it, finds the opponent goal and finally carries the ball towards the goal and kicks. But there are a large number of parameters that affect this main kernel and cause interrupts in its sequence. For example, the main method for finding the ball is rotating. When our robot is moving in the field it tries not to collide with other robots. Collision avoidance is done by calculating the distance and angle of other robots and changing the speed of its motors. This capability showed its good performance in dribbling other robots.

In addition, robot ability to measure the motors current, enables it to determine stuck situations and thus making appropriate move to come out of that state.

5 Discussion

Considering the dynamic situation in a soccer game, the elementary generation of robots which are able to show the clarity of a future when we or our children will see humanoid robots playing soccer with human soccer players in a real field, should have a special mechanics, control and vision system which can simplify robots move in such environment. In this regard, it is obvious that soccer robots should be divided into two categories, players and goal keeper, each designed with different concept. The responsibility of a goal keeper and the type of its movements is basically different from other players.

Considering these facts, at the present time, we can not purchase robots which could fulfill these works in the way that we think is appropriate. Therefore, we decided to go through all difficulties of designing and constructing all parts of our robots, including mechanics, control hardware and software by our group.

In RoboCup-99 this idea showed its superiority compared to robots purchased from certain manufactures which had a general purpose design. The possibility to make changes in their mechanics and movement capability were limited, that is why the users were bounded to a certain extent to the parameters put through by the manufacturer.

We believe that one of the keys to the success in this field is designing the mechanics and control hardware which can best fit our idea of soccer player robot. Also we should concentrate on a fast and reliable vision system. It is unbelievable for a human soccer player to mistake a ball in real soccer field. Therefore our robots should be able to be improved to that ability. High resolution of CCD camera and fast and reliable frame grabbers are among the essential tools needed for the vision system.

The sliding arm of our goal keeper moves much faster than robot body to the left and right. Like a human goal keeper, when he tries to catch a ball from sides, his hands move faster and before his body. This design not only enables the robot to catch fast moving balls going from sides, but also reduces the risk of horizontal displacement of robot in fast left and right moves. Because we use two PC main boards with a frame grabber installed on one of them and other hardware boards, the height of our goal keeper is too much (i.e. 60 cm). There is not a proper balance between its width, height and weight, that is why in high accelerations the robot itself become unstable. To overcome this problem, we suggest the replacement of large size mother boards and frame grabber to some small size, so we could fit all hardware equipment in a smaller space. If this problem is solved, it is suggested to use two CCD cameras for sides view instead of digital video conferencing cameras.

At present our player robots can communicate with each other using wireless network by TCP/IP protocol. In practice we sometimes encountered communication stall. We think this is due to TCP/IP protocol when waiting for acknowledgment anytime a packet is sent, and this wait lasts because of electromagnetic noise in the environment. We think it will be more appropriate to use UDP protocol, because it is not a connection oriented protocol and does not wait for acknowledgments. A reliable real-time communication in a multi-agent system is the base for a successful team play.

6 Conclusion

Arvand is the 2nd generation of robots constructed by our team. One advantage of **Arvand** is its unique mechanics design which enables it to rotate around any point in the plane. Therefore, the robot can rotate around the ball center while simultaneously finding the goal position. Object distance measurement and motors speed control, enabled the robots to implement special individual playing techniques in dribbling, releasing themselves when stuck and taking out the ball from a wall corner.

Another advantage of our robots is the use of MS/DOS operating system, because it can be executed on a floppy disk which is a more reliable device compared to hard disk, on mobile robot. Our robots showed a good performance in real games and we are going to improve our software algorithms based on individual techniques and also team play. The wireless LAN system used in our robots provided the communication between robots resulting a cooperative behavior, specially when a robot has the control of ball. A well defined cooperative behavior in a multi-agent system, is the key to success of team play algorithms and also individual techniques. Sliding arm of goal keeper enables it to take fast moving balls from sides. The four drive units moving mechanism reduces the horizontal displacement of goal keeper during movements.

7 Acknowledgments

We like to give our best gratitude to the president of Sharif University of Technology, Dr. S. Sohrabpoor for his special support and caring, also to our university adminisitrative deputy Mr. A. Bayati and research deputy Dr. M. Kermanshah for their full support of this research work. Many thanks to Dr. E. Shadravan from department of mechanical engineering and his technicians in mechanics laboratories, and also all university staff who played a vital role at the right time to help this project.

We like to thank a lot our financial supporters in Iran: Mega Motor, SAIPA, Ministry of Culture and Higher Education, Planning and Budget Organization, Iran Industrial Expansion and Renewal Organization, Fajar Petrochemical Industries, Sanier Co., Iran Telecommunication Co. (Data group) and Advanced Industrial Systems Research Center.

Many thanks to Mr. A. Rajaiyan, Mr. A. Rajabzade and Mr. M. Seyed Razavi the students in computer engineering department, whose help and cooperation was very valuable for us in difficult and desperate situations.

References

1. Shigley, J.E., *Mechanical Engineering Design*, McGraw-Hill, 1986.

2. Meriam, J.L., *Dynamics*, John Wiley, 1993.
3. Mazidi, M.A., and Mazidi, J.G., *The 80x86 IBM PC and Compatible Computers, Volume II*, Prentice Hall, 1993.
4. MacKenzie, I.S., *The 8051 Microcontroller*, Prentice Hall, 1995.
5. Gonzalez, R.C., and Woods, R.E., *Digital Image Processing*, Addison-Wesley, 1993.
6. WATTCP http://www.geocities.com/SiliconValley/Vista/6552/l6.html