Abstract— This paper is the team description paper of "PersianGulf" team in the Junior Soccer league, participating in RoboCup Canada 2018. It details our research and development process in building robots for the RoboCup Junior Soccer Open Category. The paper covers all structural, electronic, and programming aspects, highlighting our most significant challenges and the strategies employed to overcome them. Additionally, it explains our approach to enhancing the robots' performance in the competition. Our school achieved second place and the Best Superteam Integration award in 2014 and 2015, second place in the Superteam category in 2013, and fourth place with the best presentation in 2012, among other accomplishments.

1 Introduction

Farzanegan1 High School began its involvement in Junior Soccer in 2005. In the 2006 RoboCup, the team qualified for the quarter-finals and achieved third place. Farzanegan High School has made numerous achievements since 2005. For upcoming competitions, the team's goals are twofold: first, to prepare for detecting a passive ball, and second, to develop more dynamic and intelligent behavior. In the 2017 competitions, the primary structure of the robots remained the same as the previous year. Figure 1 shows the "PersianGulf" team robots. Some requirements for reaching this target have been met by redesigning the electrical mechanisms. Additionally, a camera and a processor are needed instead of an IR sensor to detect the ball. This paper is organized as follows: First, it describes the entire hardware used in the robots. The software architecture is outlined in Section 2.

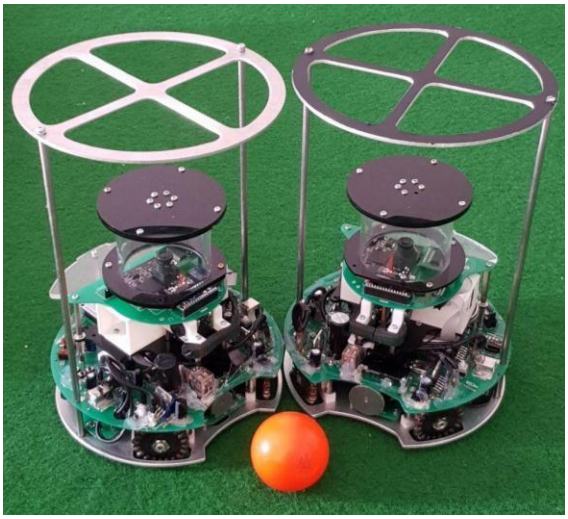

Figure 1 - The Robots

2 Hardware

2.1 Ball detection

After switching to a passive orange ball, we decided to explore image processing. Following extensive research, we found that ball detection requires the use of a mini-computer (such as a Raspberry Pi or Odroid) due to its high frequency and Linux-based OS. However, this approach has both advantages and disadvantages: On the plus side, there are powerful tools for coding and flexible programming options. On the downside, the approach is very complex for us, and the OS is quite heavy (with a 7-second boot time). Ultimately, we decided to use a CMUCam. It offers a good frame rate (50 frames per second) and is easy to use. Additionally, a significant advantage is its Arduino compatibility.

2.2 Central processor

The best platform for using the CMUCam5 Pixy is Arduino, but which Arduino model is better for junior soccer robots? Most Arduinos use an 8-bit AVR microcontroller, except for the DUE, MKR1000, 101, Zero, and MKRZERO. Most AVR microcontrollers operate at a 16MHz frequency. This rate is not suitable for us because many devices and modules must be monitored quickly. Therefore, we chose the Arduino Due for its higher frequency (84MHz) and reliability, thanks to the 32-bit ARM microcontroller.

2.3 Location Detection

Finding the location of each robot is one of the important aspects of our work. During our research, we discovered a variety of sensors for measuring distance. We can categorize distance measurement sensors into two types: Infrared sensors (like the old GP2Y0A21YK) and Ultrasonic sensors. Infrared sensors have a significant advantage: a very narrow beam and high speed (thanks to the nature of IR radiation). However, they have a major disadvantage: they are not effective when the sensor is not perpendicular to the wall. Ultrasonic sensors have an advantage in that they are not sensitive to angles due to their wider beam. Their weakness is a lower speed (because of the nature of the mechanical wave). Therefore, we chose the MB1020 LV-MaxSonar-EZ2, a popular low-cost ultrasonic sensor, which we use with an ADC. After receiving data from all four sensors, we can approximately determine our location on the field.

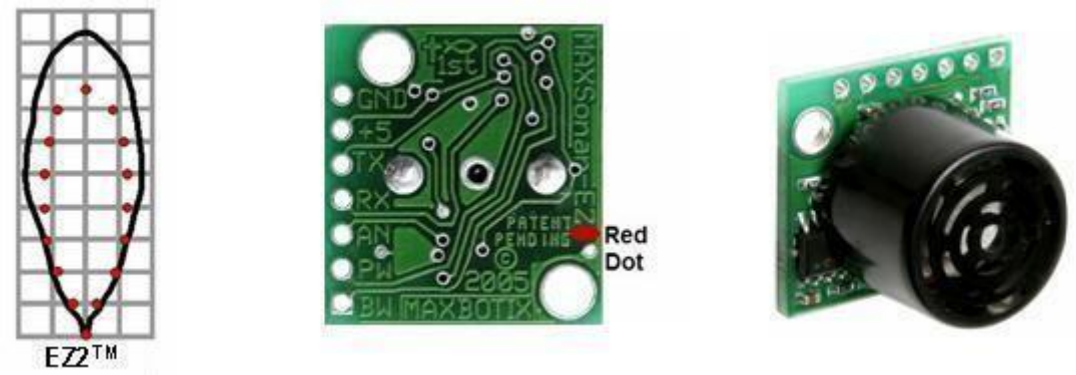

Figure 2 - MB1020 with beam pattern

2.4 Avoid out-of-bounds

We found a reputed sensor, the TCRT5000, which is very popular for line detecting and line following, and is also used in copy machines. However, a negative point about the TCRT5000 is that it is susceptible to magnetic fields, such as those from DC motors. Additionally, the sensor must be placed next to the robot's motors. Therefore, we cannot use the TCRT5000. Instead, we use a photoresistor (or light-dependent resistor, LDR) in combination with an LED. This component is a light-controlled variable resistor, meaning the resistance of a photoresistor decreases with increasing incident light intensity; in other words, it exhibits photoconductivity. If you use an LDR as part of a voltage divider circuit, you can obtain a voltage output. So, when the sensor is close to the white line, the voltage will change, allowing you to detect the line.

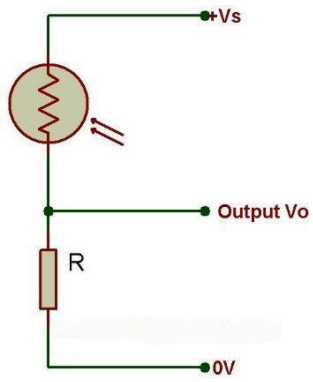

Figure 3 - LDR in voltage divider circuit

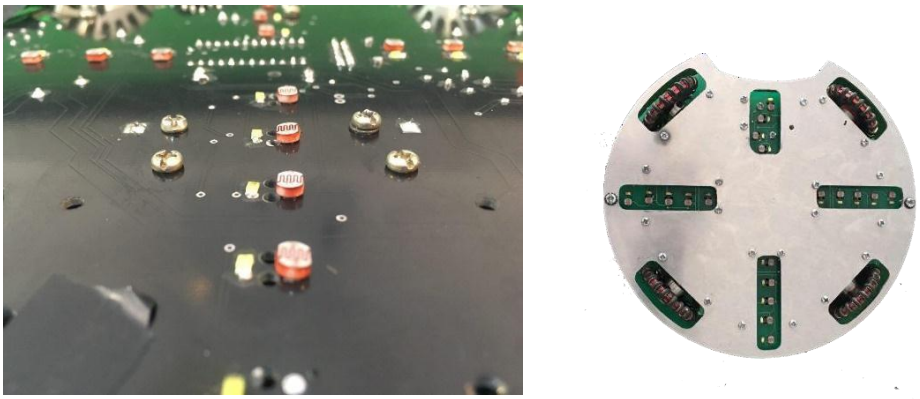

Figure 4 - LDR mounted on PCB

2.5 Compass

The robot always has to face toward the opponent's goal, so we need a compass sensor to aid in navigation and to detect our angle on the field. The Devantech CMPS03 would be a good choice; it produces a unique number to represent the direction the robot is facing. When the robot is turned on, we save the compass reading and set it as the origin of the coordinate system.

Figure 5 - CMPS03

2.6 Main board

One of the best applications for designing electronic parts and printed circuit boards (PCB) is the Eagle; it is open-source, famous, and easy to get. But it is a very complex tool to work with. So, we decided to work with Altium Designer which is an expensive software, but on the other hand, it's very helpful, and it's used more in our country.

We tested and designed many boards and this is the last and the best format.

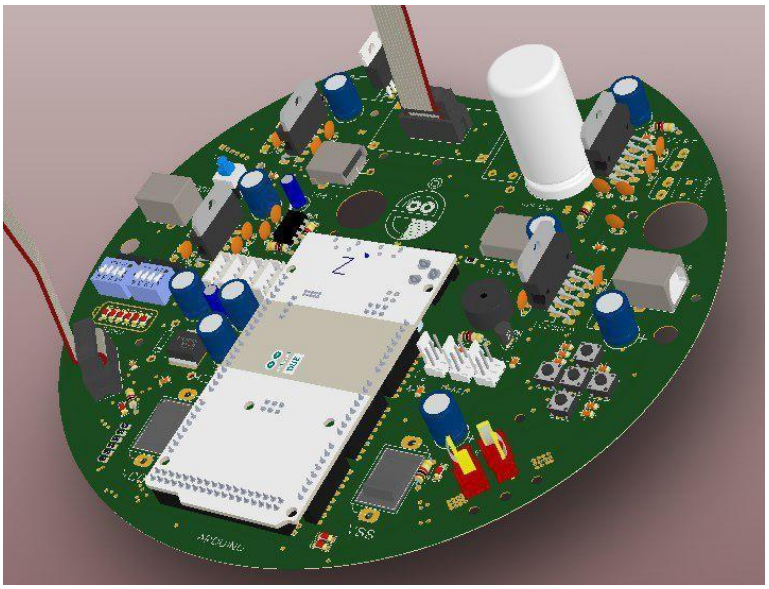

Figure 6 - Designed PCB in Altium Designer

2.7 DC Motor Driver

For driving motors, we need a driver. Therefore, we chose the L6203. This IC is a full-bridge driver for motor control applications, realized in Multipower-BCD technology, which combines isolated DMOS power transistors with CMOS and bipolar circuits on the same chip. The L6203 is an IC driver for DC motors with an output voltage of 48 V and a current capacity of up to 4A. Using PWM, we can vary the motor speed.

2.8 Kick System

To increase the voltage to the level needed for kicking the ball, we used an XL6009 module. This module boosts the voltage and then charges the kick capacitor, which is located at the front of the robot. Ultimately, by activating the relay, this voltage is connected to the solenoid, completing the kicking process.

2.9 Communication

The communication between our robots is facilitated through the HC-05 Bluetooth module, which is set up using the USART protocol. It is an easy-to-use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. The algorithm and usage details are explained in the software section.

2.10 Convex Mirror

Because of the competition rules, we are only permitted to use one camera for each robot. So, we have two ways to provide 360 degrees view:

A) use CMUCAM5 Pixy on a servo motor

B) using convex Mirror (A convex mirror, diverging mirror, or fish eye mirror is a curved mirror in which the reflective surface bulges toward the light source.)

We decided to implement Plan B because Plan A had limitations in rotation, which we found undesirable. Subsequently, this led to a problem: what is the equation for a mirror? To address this, we conducted simulations with various equations. We consulted with physics and geometry teachers, and after extensive research and simulations, we found this hyperbola equation:

$$\frac{y^2}{61} - \frac{x^2}{45} = 1$$

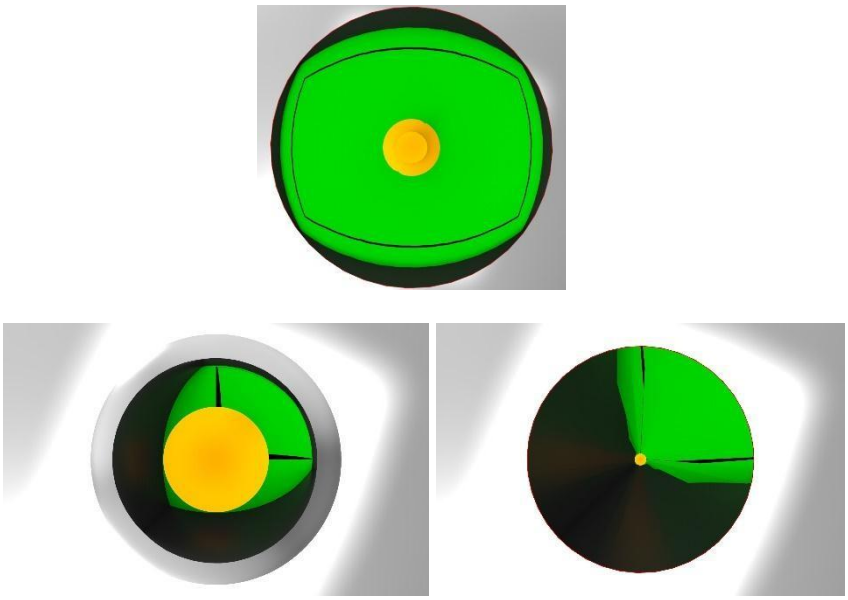

Figure 7 - Convex mirror in simulation

3. Software

As we mentioned before, the main processor of our robot is the Arduino Due from the Arduino family, so we have to compile it with the Arduino IDE. The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. The base part of the code contains functions that are only called once (setup). We add functions to shorten the program and the code that runs continuously (loop).

3.1 Ball following

In the first step, we need to code the robot to move towards the ball. Initially, we should determine the robot's angle using image processing. As previously mentioned, the CMUCam sends data that includes the x and y values of the ball (relative to the origin coordinate) via SPI protocol to the Arduino. Then, by calculating ArcTan(y/x), we can determine the angle between the center of the robot and the ball. Now, there are two methods to move the robot towards the ball using this angle. The first method involves writing some static moves that the robot can follow based on the angle of the ball. The second method is to code a function that generates movements for each angle.

In the second step of the ball following, the robot should move behind the ball. So, the robot, when it is close to the ball must go in the direction of the angle of the ball with a constant angle shift (for example 40 degrees). Using the code below:

```
BALLx = pixy.blocks[0].x - XCenter;
BALLy = pixy.blocks[0].y - YCenter;
BALLxCenter = abs (BALLx) + (pixy.blocks[0]. width / 2);
BALLyCenter = abs (BALLy) + (pixy.blocks[0]. height / 2);


at = atan2(BALLy, BALLx) * 180 / PI;
if (at < 0) at = at + 360;
Distance = sqrt(BALLxCenter * BALLxCenter + BALLyCenter * BALLyCenter) – 50;
      // The 50 is robot. Removing the offset
if (Distance < 80)
{
  if (at < 10 || at > 350) angtoM(0); // The function move Robot's in right angle
  else if (at > 180 && at <= 350) angtoM(at - 40);
  else if (at <= 180 && at >= 10) angtoM(at + 40);
}
else
{
  angtoM(at);
}
```

3.2 Goal Detection

The robot should always move towards the opponent's goal, so, as we mentioned earlier, we need a compass to determine our deviation from geographical north and navigate accordingly. The CMPS03 module is a good choice; using this module, the robot can identify both its own and the opponent's goals. This is achieved by measuring the angle relative to the original axis, with numbers ranging from 0 to 128. We then write a function to generate a new number by adding the output data from the module with an appropriate coefficient. This final number is then combined with the motor speed, influencing the robot’s movements to keep it oriented towards the opponent's goal. Furthermore, to detect the goals, we can use their colors. According to the rules, one goal is blue and the other yellow, so using the Pixy, we can identify our own goal and the opponent's, thus maintaining the correct direction.

The first step in writing the goalkeeper's program is to identify our own goal. For this, we use PixyMon; with Pixy’s camera, we can detect up to seven objects simultaneously. This way, we can identify the goals and even estimate their distance from the robot. PixyMon provides the length and width of the detected objects, allowing us to pinpoint the center of the gate as a reference point and calculate the robot’s distance from it.

3.3 Communication algorithm

Each robot that has the ball in front assumes the role of Forward, while another robot plays as the Goalie. However, if Bluetooth is unavailable or the robots cannot connect to each other, a default function will be applied, with one robot acting as Forward and the other as Goalie.

4 Structure

According to the rules, the robot should have a circular shape of 22 cm in diameter, where the ball can enter no more than 2.5 cm from the edge.

The robot's mechanic is composed of several important parts that are listed below:

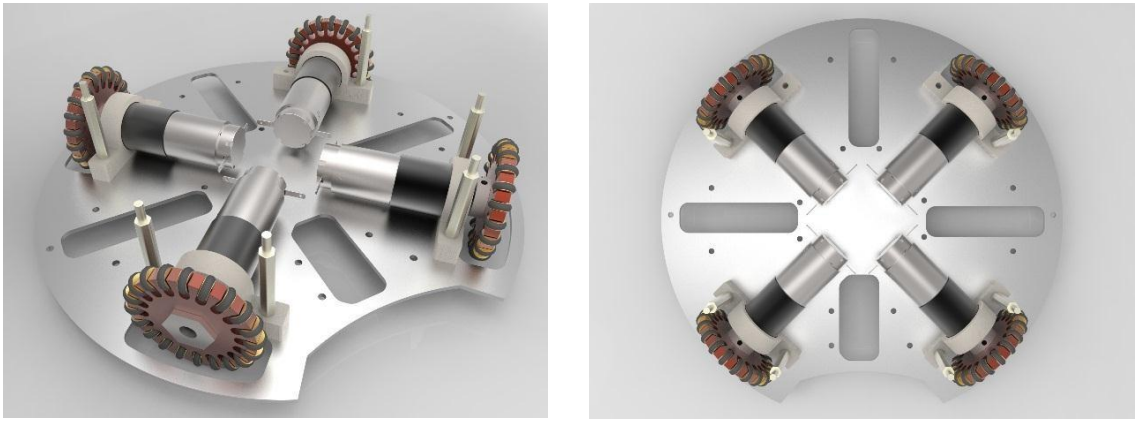

Figure 8 – Robot structural design

4.1 Motors

The Motors used in the robots are MAXON DC-Brushed 10200 RPM 15V. We use a 1:14 gearbox and have about 800 RPM as output. According to robots speed and Torque are the factors which are important.

4.2 Wheels

Based on the robots from previous competitions and our research, we observed that omnidirectional wheels are the best choice for our robot. Therefore, we chose an omnidirectional wheel. It has several small, rubber-coated wheels placed within the main wheel, creating friction with the ground. This wheel can also function as a freewheeling wheel. Another advantage of the omnidirectional wheel is that it doesn't lock during rotation and it maintains the robot's direction consistently.

For practical purposes, we opted to use a robot with four omnidirectional wheels, which is much faster. Additionally, its programming is simpler and more efficient compared to robots with three wheels.

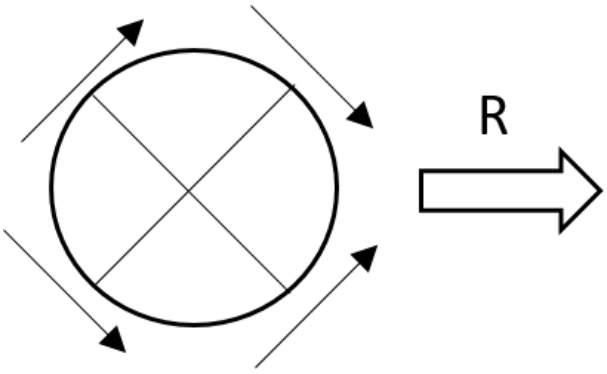

Figure 9 - Robots Motor placement

5 improvements in RoboCup 2018

We plan to add a dribbler that uses spin to control the ball. Another feature we aim to introduce pertains to the goalie. Playing in the opponent's field and removing the ball from the goalie's area is highly effective and important. Therefore, we want to add a fan to create airflow, allowing us to change the ball's direction and remove it from our side. We hope these improvements will significantly enhance our performance in the competition.

6 Reference

1. Tech article: how to have a very fast boot time with Raspberry Pi:

http://www.samplerbox.org/article/fastbootrpi

2. CMUcam5 Pixy

http://www.cmucam.org/projects/cmucam5

3. Arduino Compare board specs

https://www.arduino.cc/en/Products/Compare

4. Wikipedia, Photoresistor

https://en.wikipedia.org/wiki/Photoresistor

5. St website. L6203 motor Driver

http://www.st.com/en/motor-drivers/l6201.html

6. SparkFun, HC05

https://www.sparkfun.com/products/retired/9977