EVO Team TDP

Team members: Mahdi Salmani, Ali Hosein Mazaheri Tehrani

Mentors: Mohammad Baghersad, Mohammad Khosha, Mohammad Aref Nasiri,
Soheil Arjmandkia, Mohammad Mahdi Saket, Mahdi Ghandehari

Reza Vaseghi secondary high school, Khazaie Street, Tamadon interchange, Esfahan, Iran

robotmb@gmail.com

Abstract: In this team description paper, we are more than happy to share our information and experiences that we have learned and gained from 2014 till now with you.
Experiences in the field of producing mirror, electronical kicker circuit, dribbler power transmission, robot mass center, attack algorithm, goalkeeper algorithm, etc. One of our robot advantages is using parallel computing with 12 microcontrollers. The most important part of this paper is boundary detection algorithm using phototransistors (Line Meter) and emergency using of spare SRF sensors. By exact focusing on each of the mentioned sections, the best result has been implemented on the robot so far.

Keywords: Robocup, Line meter, Dribbler.

1 Introduction

1.1 Team background

Sadra Robot team participated in the rescue line league in 2014 and 2015, soccer light wight league in 2016, and finally in the soccer open weight league in 2017 to 2021 as Sadra, VCC and EVO teams. In 2017 and 2018, team won the first place in domestic competitions of IranOpen and has the experience of participating in the RoboCup competitions of Japan and Canada. The members of the team are Mahdi Salmani in the role of software and electronics and Ali Hossein Mazaheri Tehrani was responsible for robot hardware.

1.2 Team photo

1.3 Current year's highlights

The robots made this year have important highlights that include the following. Accurate detection and decisions of how to deal with the boundaries around the field. Dribble the opposing robot when attacker robot approaches the opposite goal area. Check and identify the empty part of the opponent's goal to find the best shooting position. change the defending robot role to the attacking robot after 5 seconds, when the ball stays constant on the field, which is a sign that the attacker robot is outside of field.

2 Robots and Results

2.1 Hardware

The robot was designed by SolidWorks software, which is a CAD-based engineering software. The robot is designed to have a low center of gravity by placing large heavy pieces at the bottom of the robot to have a stable and uniform movement without any wobble. As a result, the robot will not detach from the ground during fast accelerations.

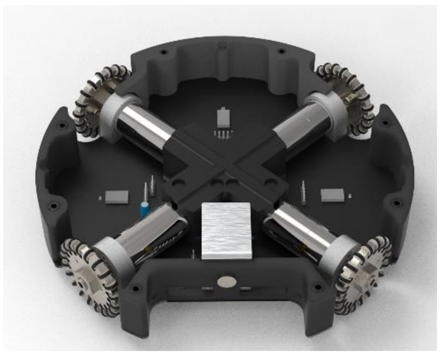
Mirror:

Different models of mirrors were designed in the software and were made by CNC milling with Teflon and ABS materials. Finally, by performing repeated experiments, the current shape of the mirror was reached, which covers the entire playground (also SuperTeam field) with a suitable 360 degree viewing angle. The material of choice was Teflon because it led to a smooth surface in the chemical polishing process, without corrosion and easy changing the original shape after final polishing. Finally, with chrome plating and a few steps of polishing, the current mirror surface was obtained.

Motion system and motors:

The motion system uses four MAXON DCX22 motors with GPX22 gearbox; Maxon Graphite Brush Motor is capable of producing 0.4 Nm of torque and 11700 RPM without gearbox, which with the help of a ceramic gearbox with a conversion ratio of 16:1 has an output of 6.4 Nm and 730 RPM at the output to the wheels.

The motors are mounted cross-shaped on PCB chassis. The angle of the motors is 85 degrees from the sides and 95 degrees from the front and back, which allows to have more space for the robot's ball capture zone in addition to having high maneuverability.



Motor stands:

According to the size of the motors and the amount of pressure that is applied to them, in the strongest and also lightest state, the motor stands were made of aluminum material.

Surrounding protective frame:

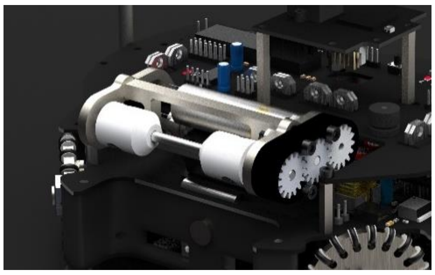
The frame around the robot is made by 3D printer with PLA material, which is light, affordable and has a high impact resistance. INFILL rate (55%) with hexagonal pattern gives us the best strength and lightness.

Dribbler (Spin back):

Dribbler was completely designed in Solidworks and after making the prototypes with wood and laser cutting, the final dribbler was made with aluminum material with wire cut and CNC milling machines. Different rollers designed in Solidworks according to different ball entry conditions to the robot's ball capture zone and after production and testing, the optimized sample was used in the robot. The frame of dribbler is made with 3D printer and the roller itself is made of CV2 silicone by injection molding method.

The motor which is used in dribbler mechanism is MAXON DCX16 with GPX16 ceramic gearbox with 16:1 conversion ratio. output shaft RPM is 400, which transmits power with three gears.

A spring has been used to remove the vibration of mechanism.



2.2 Electronics

power supply circuit:

The power supply circuit consists of two parts. Due to the need of minimal electronic noise, a linear power supply circuit should be used, but because linear regulators generate excessive heat, the battery voltage must first enter a switching regulator (that LM2596 was used in robots) and the circuit voltage will reduce from 14.6 to 7 volts, then it should enter a linear regulator (like MIC29302 that was used in robots) and reduce voltage from 7 to 5 linear volts also with less noise.

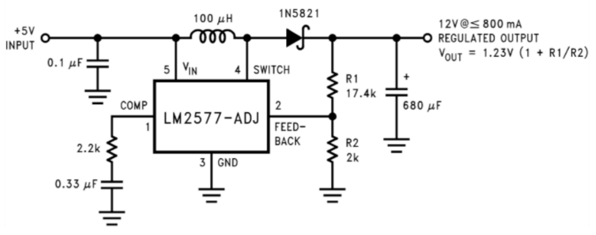
Using the linear regulator of MIC29302 alone (not using the LM2596 switching regulator) causes high temperature, high energy loss and damage to the circuits.

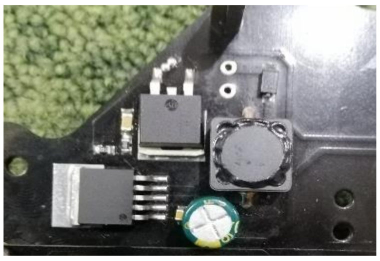
Dribbler motor driver circuit:

This circuit includes an IRFZ44 MOSFET and a TC4427 MOSFET driver. The reason for using a single MOSFET is that the dribbler motor rotates in only one direction and at a constant speed so there is no need to design a sophisticated and advanced motor driver.

kicker circuit:

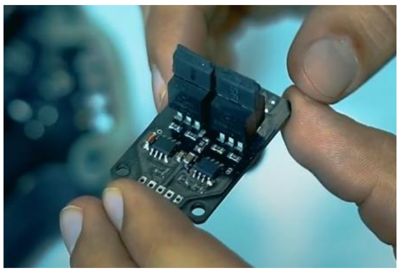
For the kick system, high voltage is needed. So, LM2577 IC is used to convert 12V to 30V. A capacitor is charged at 30 volts and discharged onto the solenoid by a mosfet with microcontroller command.





Motor drivers:

Motor drivers may be damaged due to instantaneous current flow. So, we decided to modulate them. In this case, the motor drivers can be easily detached from the main board and can be replaced in a few seconds.



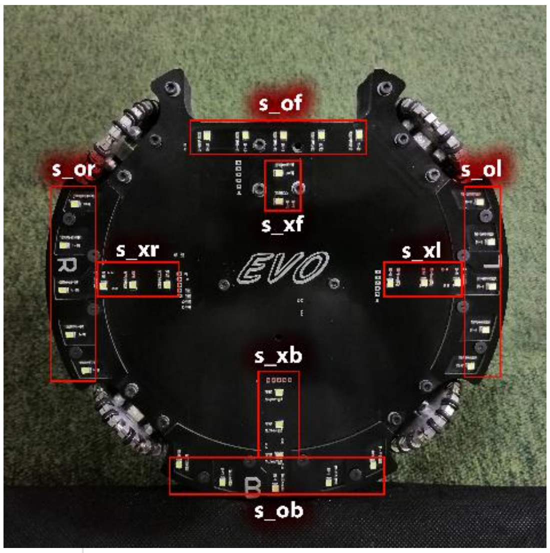
2.3 Software

Our robot has 12 microcontrollers, a camera microcontroller, four line-sensor microcontrollers, a gyroscope microcontroller, four motor driver microcontrollers, a microcontroller for rangefinders and a main microcontroller. Information processing in our robot is done simultaneously. So, 11 microcontrollers are processing data at the same time with the main microcontroller and only transfer prepared data with the main microcontroller.

Line detection:

Our robot has (31) phototransistor sensors on the bottom of the robot. To increase the data processing speed of these sensors, we used 4 small microcontrollers instead of one large microcontroller.

To do that, each 8 sensors on each 4 sides of the robot (except for the front, which has 7 sensors) are connected to a microcontroller and its output voltage, which is between 0 and 5 volts, is measured by the microcontroller's ADC, and based on the calibration, it is decided whether the sensor is on the white line or on the green area in the field.



Finally, line-detection microcontrollers send a value of 0 or 1 to the main microcontroller for each sensor. Now 31 values for 31 line-sensors are available in the main microcontroller based on which, the robot decides the reaction to the boundaries.

The line detection algorithm starts when only one of these 31 sensors returns a value of 1 (white line is detected). Now the first thing to consider is to detect the direction of the robot when it gets close to the boundaries (From which side of the field is the robot exiting?). Depending on the location of the sensors on the line, it can be determined from which side of the field, robot is exiting. Which has **4** conditions (**front**, **back**, **left** and **right**)

```
if(s_all>0)
{

   if        ((s_ol>2 || s_xl>1 || ((s[5] || s[4]) && (s[11] || s[12])))  && out_r==0  )                out_l=1;
   else if   ((s_or>2 || s_xr>1 || ((s[1] || s[2]) && (s[14] || s[15])))  && out_l==0  )                out_r=1;


   if        (s_f>4 || ((s[6]  || s[7])  && (s[20]|| s[19])) || ( s[22] && (s[20] ||  s[6])) && out_b==0 )  out_f=1;
   else if   (s_b>4 || ((s[10] || s[9])  && (s[16]|| s[17])) || ( s[28] && (s[16] || s[10])) && out_f==0 )  out_b=1;

   //s:sensor  r:right   l:left  f:front  b:back  O:circular  X:cross

}
```

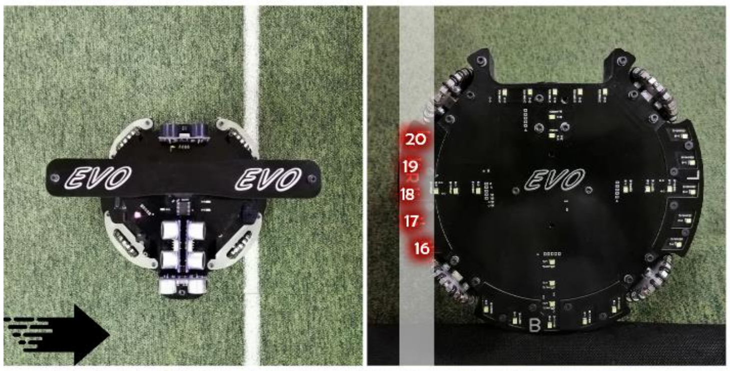
The next step is to diagnose the severity of the danger.

As you know, a robot can reach the boundary lines with different speeds, so its reaction should be different as well.

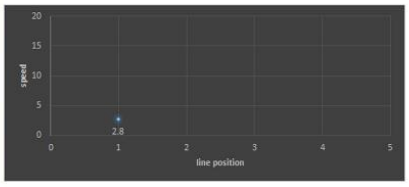
This is where our new and innovative algorithm called line-meter (LM) is implemented and can calculate how much of the robot has existed from boundaries. The higher this amount (LM), the faster the robot tries to return to the field.

In the following formula, all the sensors are measured and the line-meter number (LM) is calculated according to the coefficients. Then by using a few experimentally and mathematical obtained formula, the speed of the robot (SP) is calculated.

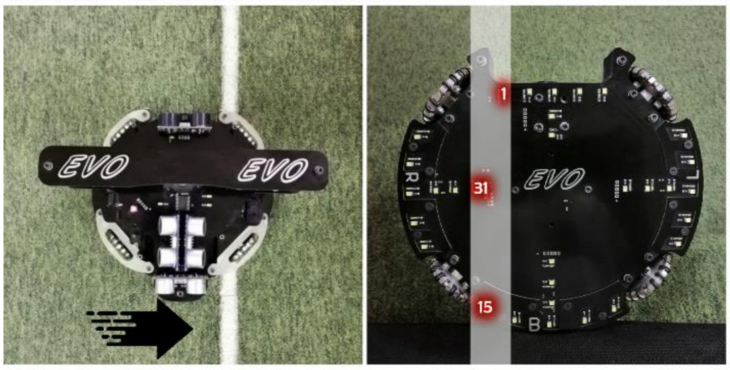
```
else if(out_R) {
   LM=(((s[6]+s[7]+s[8]+s[9]+s[10])*30) + (s[23]*120) + (s[24]*110)+ (s[25]*100) + ((s[5] || s[11])*90)
   + ((s[4] || s[12])*80) + (10*(s[3]+s[21]+s[22]+s[28]+s[27]+s[26]+s[13])) + ((s[14] || s[2])*40)
   + ((s[1] || s[15])*30) + (s[31]*20) + (s[30]*15) + (s[29]*10) +  (s[16]+s[17]+s[18]+s[19]+s[20]));
   sp=floor(sqrt(LM))*1.40;
   if(sp>15) sp=15;}
```



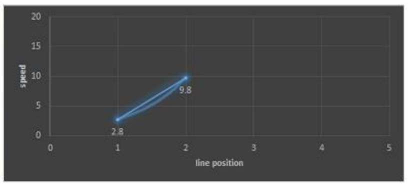
```
LM=(s[16]+s[17]+s[18]+s[19]+s[20]);
sp=floor(sqrt(LM))*1.40;
(sp=2.8)
```



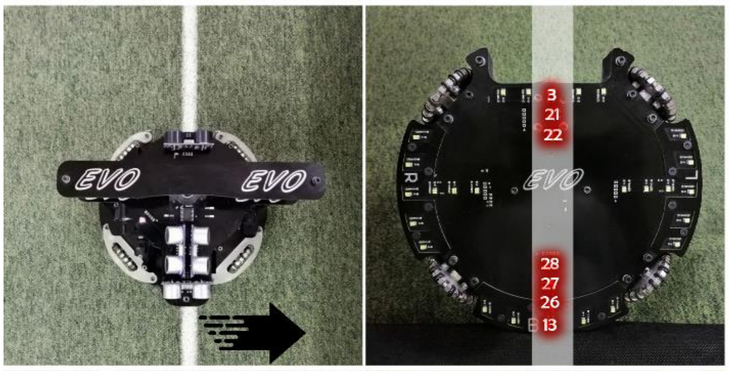
1



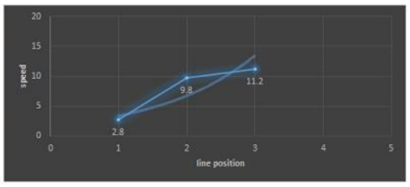
```
LM=(((s[1] || s[15])*30) + (s[31]*20));
sp=floor(sqrt(LM))*1.40;
(sp=9.8)
```



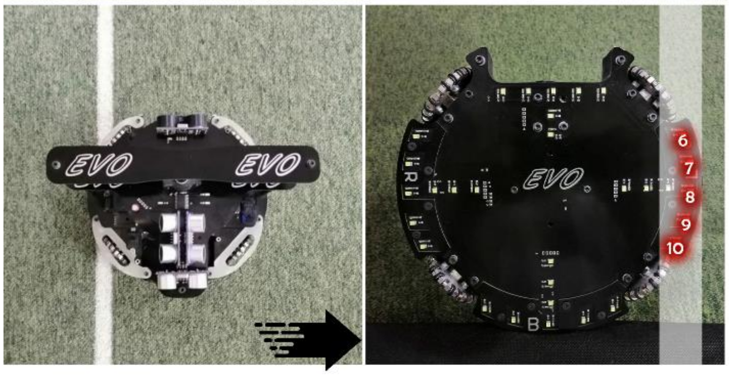
2



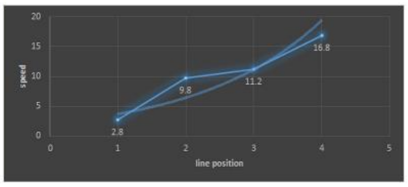
```
LM=(10*(s[3]+s[21]+s[22]+s[28]
            +s[27]+s[26]+s[13])
sp=floor(sqrt(LM))*1.40;
(sp=11.2)
```



3



```
LM=((s[6]+s[7]+s[8]+s[9]+s[10])*30);
sp=floor(sqrt(LM))*1.40;
(sp=16.8)
(sp>15 so sp=15)
```



4

In the above cases, respectively, in the first stage, the robot has just reached the white line, so it returns to the field with a slight speed (sp=2.8). The further the robot goes to the boundaries (steps 2, 3 and 4), the faster it returns to the filed (higher sp) and thus the exiting percentage of the robot is close to zero. Another feature of this algorithm is that if the robot is pushed by the opponent robot, it will return to the field.

Attack algorithm:

First, the robot goes to the ball according to its coordinates and tries to catch the ball in its ball capture zone. This is confirmed by the Omron sensor inside the ball capture zone as well as the position of the ball in the mirror image. Now it is warrantied that the ball is inside the ball capture zone of our robot.

Now the robot attacks the empty part of the opponent's goal that has been processed by the camera. If the robot fails to score goal, the SRF sensor on the front is checked, and if it turns a short distance, it realizes that it is involved with the opponent's robot.

Then it must be checked whether the robot dribbler is active or not (since the dribbler is a mechanical part and different conditions in the match have high impacts on its activity, it is possible that the dribbler become damaged during the match and can no longer catch the ball). So, a dip switch was considered that can inform the microcontroller if the dribbler is not available, without the need of reprograming. we can change the position of dip switch when the robot is damaged or when the game restart.

2.4 Results

Keeping the defender in penalty area:

In previous years, it was the duty of SRF sensors to detect whether the robot was inside the goal, which measured the distance from the walls and located the robot relative to the goal. But there was a problem, when the opponent's robot was next to the goalkeeper, the SRF returned the wrong value, so it was decided to use Omron sensors on the sides and backwards of the goalkeeper. These sensors help to detect goals and goalposts, so the robot never goes out of penalty area.

Changing goalkeeper to attacker robot:

If the goalkeeper robot detects that the ball has not moved for 5 seconds, it will notice that something has happened on the field and that the attacker robot may have been damaged. So, it changes its role to attacker robot, then returns to its goal after a short time

Spare sensors for boundary detection:

The robots use rangefinder sensors, which in case the bottom line-sensors (phototransistors) are damaged, these SRF sensors helps the boundary detection algorithm and prevent the robot from exiting the field.

Removing damaged line-sensors:

This robot also has the ability to remove the damaged phototransistor sensor from the program. It is very likely that one or more of the bottom line-sensors will be damaged due to the receiving shocks of robots during the matches and send the wrong information to the main microcontroller. Robots have the ability to detect a faulty sensor and remove it from the code. First, turning on the robot and placing it on the green area of the field. Then it is expected that all sensors give a value of 0. Any sensor that returns a value of 1, indicates that it is faulty, then it is removed from the code and its value is permanently 0 regardless of the value sent.

3 Conclusions and Future Work

3.1 In the process of building robots, the most important lesson we learned was **teamwork**. We were able to update our skills and add to our information in these years. Participating in Japan and Canada Robocup competitions was a very enjoyable and informative experience for us that expanded our global connections with people in all over the world.

3.2 Our goal in the near future is teaching the acquired skills to new students, as well as increasing and updating our scientific information in the field of robotics. The main goal in the future will be integrating robotics in everyday life and use robotics in various industries to make life easier.

References

1. Microchip.com
2. https://www.alldatasheet.com/view.jsp?Searchword=DSPIC30F4013
3. Github.com
4. https://www.maxongroup.com/maxon/view/service_search?query=dcx22
5. http://www.ccsinfo.com/newsdesk_info.php?newsdesk_id=231
6. https://www.youtube.com/watch?v=391dXDjqzXA