PARSIAN
Team Description for RoboCup 2015

Alireza Zolanvari, Mohammad Mahdi Shirazi, Seyede Parisa Dajkhosh, Amir Mohammad Naderi , Maziar Arfaee, Mohammad Behbooei, Hamidreza Kazemi Khoshkijari, Erfan Tazimi, Mohammad Mahdi Rahimi and Alireza Saeidi Shahrivar.

Electrical Engineering Department
Amirkabir Univ. Of Technology (Tehran Polytechnic)
424 Hafez Ave. Tehran, Iran
MhmmdShirazi@me.com

Abstract. This is the team description paper of the Small Size Soccer Robot team "AUT-PARSIAN" for entering the RoboCup 2015 competitions in China. In this paper we will represent our robots' current design in mechanical, electrical, control and software parts. Improvements and developments like new mechanical design, communication system, motor drivers, processor, control system and enhancements in predefined plays, a high speed positioning evaluator will be discussed in detail.

1 Introduction

"PARSIAN" small size soccer robots team, founded in 2005, is organized by electrical engineering Department of Amirkabir University of Technology. The purpose of this team is to design and build small size soccer robots team compatible with International RoboCup competition rules as a student based project.
"PARSIAN" team consists of ten active members from electrical, mechanical and computer science backgrounds. We have been qualified for nine consequent years for RoboCup SSL. We participated in 2008, 2009, 2010, 2011, 2012, 2013, and 2014 RoboCup competitions. Our most notable achievements was AUT-PARSIAN's first place in RoboCup 2012 SSL's Passing and shooting technical challenges and RoboCup 2013 SSL's Navigation challenge.
In this paper we first introduce our robots' Mechanical design (section 2). Our new electrical design will be discussed in section 3 and control system and software will be covered in section 4 and 5.

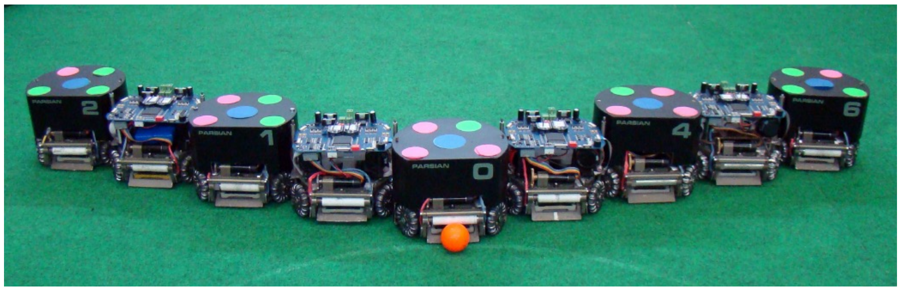


Fig. 1. Our Robots

2 Mechanical Design

In this part an introduction is going to be rendered on the mechanical structure of the new robot designed for RoboCup 2015. In this design the focus has been on three parts: the dribbler part, the center of mass, and easy access to the different parts of the robot which will be explained in detail in the following sections.

2.1. Dribbler structure

The dribbler of the previous robot had some problems with spinning, controlling, and detecting the ball, which created some obstacles to achieve an efficient shoot. To remove these obstacles and detect the ball in all situations we've decided to keep the sensor fixed and put the axis of the dribbler motion in a lower state in comparison to the preceding case.

2.2. Center of mass

To have a high performance in deceleration state in the small size robots, it's far more efficient to keep the center of mass close to the back side of the robot. In order to exert this modification in our new robot we've changed the position of the capacitor and batteries, which are heavier than the other parts, and located them at the back of the robot, as indicated in Fig.2.

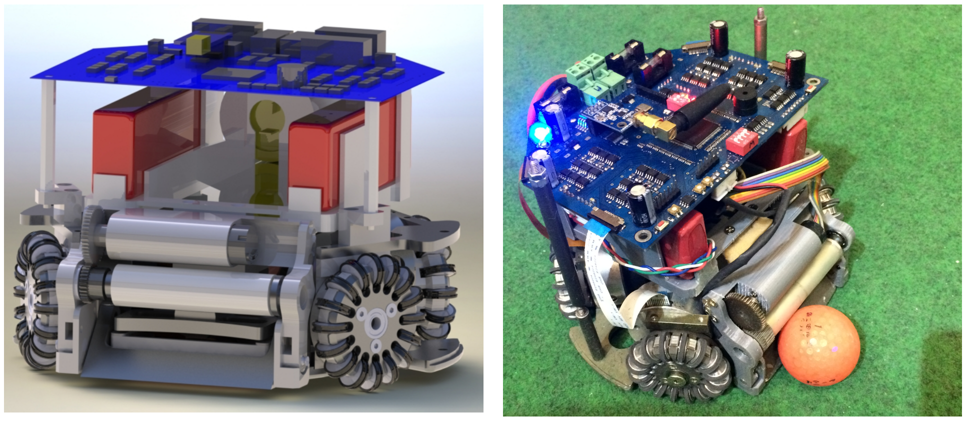


Fig.2. New mechanical design (SolidWorks design Left, Real robot Right)

2.3. Main Structure and Driving System

In this section some information will be given on the mechanical structure including plates, columns, fasteners, motors, and wheels.

7075 Aluminum alloy is the most practical material of the structure. In some cases steel, polyamide, ABS and etc have been used. The robots have got four Omni-directional wheels and a Maxon EC-45 30w brushless motor drives each wheel. The power transmission system of the robots has been reduced to a pair of internal and external gears, which connect motors and wheels.

3 Electrical Design

3.1. Test Bed

A test bed platform was designed to satisfy the need for observing the effects of a single parameter on the whole system, comparison of different modules, and test, evaluation and optimization of every part of robots' electronics.

The test bed board is consisted of a stabilized power source that generates 5v, 3.3v, 2.5v and 1.2v, I/O ports (UART, SPI, USB and J-TAG) and three different interfaces for the processor, motor drivers and wireless communications modules to be able to change each of these modules separately.

3.2 New Main Board

After testing the test bed we found out optimal design for out main board (fig 3) that will be discussed in detail.

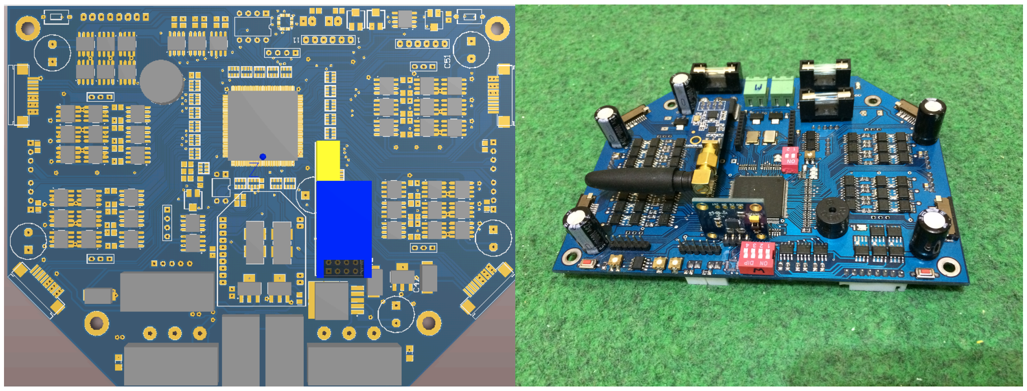


Fig. 3. (a) 3D design (b) Real hardware

3.2. Main Processor

The processor is a Xilinx Spartan 3 (XC3S400) FPGA with a 65.520MHz oscillator for sequential processes and a soft-core CPU (TSK3000A) generated by Altium designer.

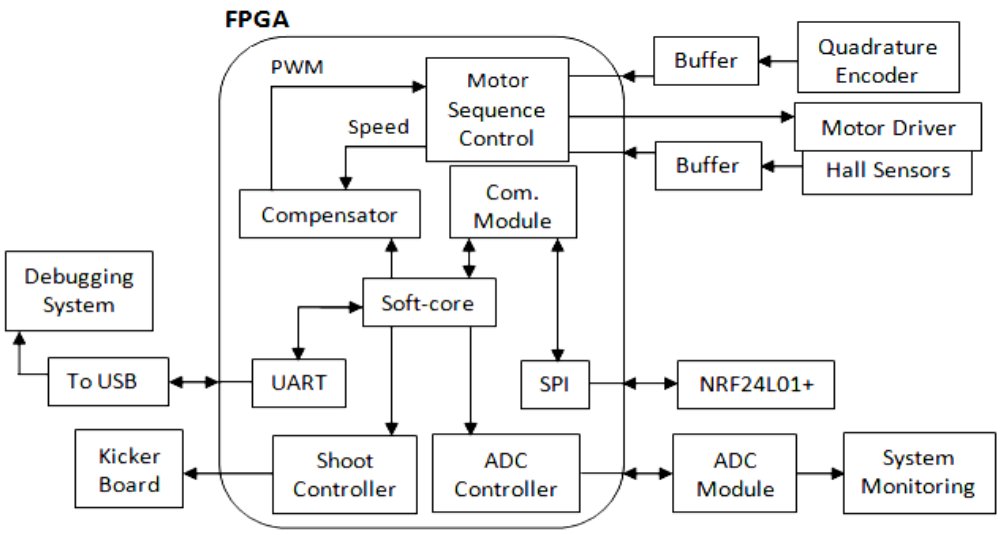


Fig. 4. Electrical system's Block Diagram

3.3. Communications System

We have changed our communications system from XBEE to an NRF24L01+ based module. The reasons for this change to take place are as follows:

1. Lower price
2. Higher air data rate and faster communications
3. Fast switching capability between the RX and the TX modes and the possibility of two-way communications using one module
4. Automatic CRC checking, packet management and data retransmit capabilities
5. More frequency range (More channels)

Currently our preferred settings and output results are:

- Channel 0x65 (frequency 2501 MHz)
- Packet size: 9Bytes Data, 5 Bytes address, 1 Bytes CRC
- Automatic retransmit: Max 15 times, 250us delay.
- Error Rate: 0%
- Data loss<1%

As communications with NRF24L01+ are based on SPI protocol, a microcontroller is used as intermediary between the central computer and the NRF Module, receiving commands through UART and sending them via SPI. To compensate for data loss, each packet is transmitted more than once.

3.4. Kicker Board

A few modifications were made to our last year's kicker board design in order to optimize it and ensure its reliability. The principles however, remain the same. The board, basically a DC to DC boost converter, charges a bank of two 2200uF capacitors up to 200 Volts and discharges them into the solenoids using two IGBTs.

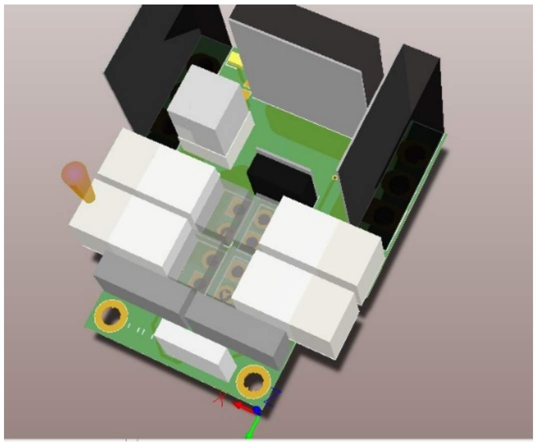


Fig. 5. Kicker board

3.5. Motor Driver

Each motor's BLDC driver unit consists of two modules; An FPGA based digital circuit as main controller and a power driver circuit. The controller receives Hall Effect sensors data, and then generates proper control signals for each motor.

3.6. The encoder

The motor's encoders just as other parts, damage and need to be replaced. But since we cannot afford the price, we decided to design and build our own ones. We use an AS5045 IC that is a 12 Bit programmable magnetic rotary encoder. This IC uses an array of hall sensors. cylindrical magnet as depicted, locates above the IC's surface in an appropriate distance, connected to the rotating part of the motor.

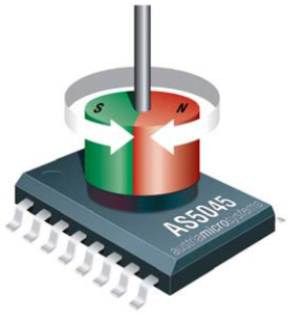


Fig.6. Sensor's IC

We design our PCB just like the original Maxon encoders, to avoid the unnecessary changes to the current housings.

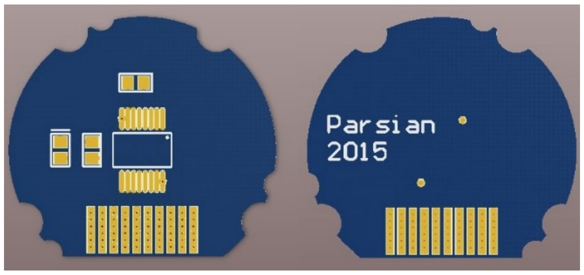


Fig.7. Sensor's board

3.7. Kicker calculations

Since we reviewed our mechanical design for this year, we decided to improve the kicker performance by using a mathematical model. So we use the electromagnetic and mechanic laws as described below:

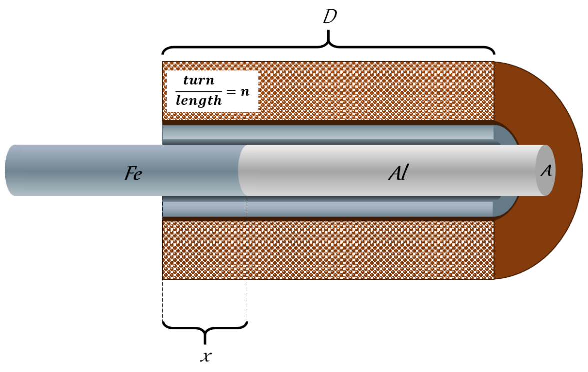


Fig.8. kicker's material

Our kicker solenoid, as depicted, has a 2-parted shaft. So we model it as to two solenoid in series.

$$L = \mu_{Fe} n x A + \mu_{Al} n(D - x)A$$
$$W = \frac{1}{2} L i^2$$
$$F = \frac{\partial W}{\partial x} = \frac{nA}{2}(\mu_{Fe} - \mu_{Al}) i^2$$

Where i is the current that the capacitors discharge through solenoid with. To simplify the math, we assume that $i(t)$ is linear between 0 and $\frac{V_f}{R_{solenoid}}$. Where V_f is the measured voltage of capacitors after kick.

So it yields:

$$v(T) = \int_0^T \frac{F}{m_p} dt = \frac{nA}{2m_p}(\mu_{Fe} - \mu_{Al}) \int_0^T \frac{V_f}{T R_{solenoid}} t.dt = \frac{nA}{2m_p}(\mu_{Fe} - \mu_{Al}) \frac{T.V_f}{2.R_{solenoid}}$$
$$v(T) = \frac{\mu_{Fe} n A V_f}{4 R_{solenoid}. m_p} T$$

Where m_p is the plunger's mass. So the ball's speed after the kick is yield from the conservation law of momentum:

$$m_p V_p = m_b V_b \Rightarrow V_b = \frac{m_p}{m_b} V_p = \frac{\mu_{Fe} n A V_f}{4 R_{solenoid}. m_b} T$$

4 Control System

4.1. Low Level

The desired speed of each motor is specified, the speed control of each motor is considered. For this we have applied a Fuzzy-PID controller. In fact, the proportional, integral and derivate (KP, KI, KD) gains of the PID controller are adjusted according to FUZZY LOGIC. By applying this method one can benefit the precise characters of PID and the flexibility of fuzzy controllers. Despite the more sophisticated control techniques that have been devised in the past decade, PID controller still operates the majority of the control systems in the world due to its simplicity, clear functionality, applicability and ease of use. If the PID parameters are tuned properly, the controller provides robust and reliable performance for one's system. The major problem of applying the conventional PID to DC motors is the non-linear characteristics of the motor which are generally difficult to be modeled precisely. So, with tuned but static PID gains one cannot achieve the desired control specifications under different motor speeds and load disturbance. But by employing the self-tuned Fuzzy-PID, dynamic PID gains are achieved, and one can get the desired control specifications under all the conditions.

The inputs of the Fuzzy-PID controller which the Fuzzification step is performed on them are the error of the speed (the difference between the current speed and the desired speed), the derivative of the speed error and the current speed of the motor. The reason for choosing these inputs is that the size of the error indicates how fast the controller should reduce the error, the derivative of the error is an indication of how the error is going to change, and the current speed is also chosen because the nonlinear characteristics of the motor is not the same at different speeds. Each input's universe of discourse is divided into five overlapping fuzzy sets {Negative Large, Negative Small, Zero, Positive Small, Positive Large} with triangular membership functions. The structure of the controller is shown in Fig.9. [19]

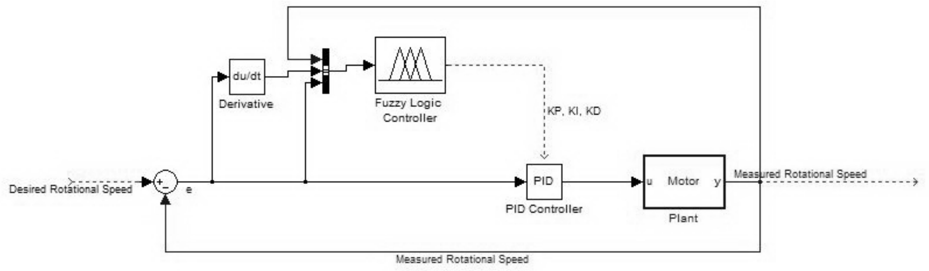


Fig.9. Low-Level Motor Controller Structure

4.2. High Level

One of the most common and big problems in control of the robots is existence of some errors in sensors data which sometimes lead to big mistakes in localization of the robots. In order to reduce these errors, we fuse the information from the sensors and SSL vision. Through this action, we can read and use more reliable data from the sensors in control loop. [18]

The Kalman filter is an optimal and efficient sensor fusion technique. Application of the Kalman filter to localization requires posing the robot localization problem as a sensor fusion problem. The basic probabilistic update of robot belief state can be segmented into two phases, *perception update* and *action update.* [18]

The first step is action update or *position prediction*, the straightforward application of a Gaussian error motion model to the robot's measured encoder and Gyro travel. The robot then collects actual sensor data and extracts appropriate features in the *observation* step. At the same time, based on its predicted position in the map, the robot generates a *measurement prediction* which identifies the features that the robot expects to find and the positions of those features. In *matching* the robot identifies the best pairings between the features actually extracted during observation and the expected features due to measurement prediction. Finally, the Kalman filter can fuse the information provided by all of these matches to update the robot belief state in *estimation*. Fig.10. depicts the particular schematic for Kalman filter localization. [18]

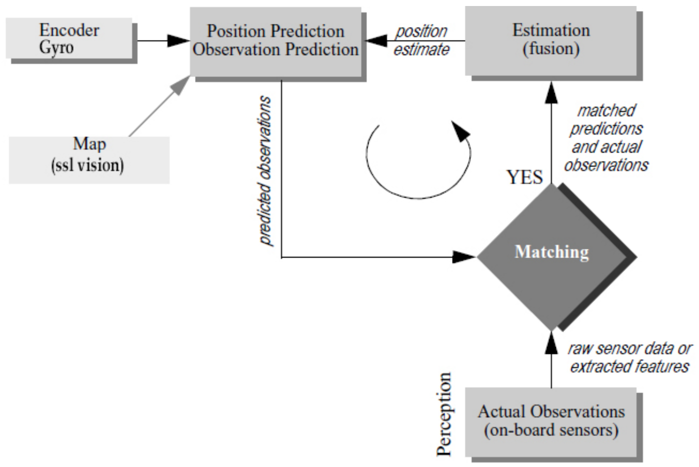


Fig.10. Schematic for Kalman filter robot localization

In future, we want to apply a Fuzzy-extended Kalman filter (Fuzzy-EKF) method to avoid the robot using the large error data to update the position continuously in order to improve the accuracy of localization. A weight scalar will be designed to change the noise covariance by inputting the robot rotation angle, innovation and the measurement data variation into the fuzzy system. Therefore, the proportion of the system and measurement value changed, which decreases the robot state errors indirectly. [20]

5 software

5.1. Architecture

The Coach layer is the first step in the high level planning (decision making) loop. Choosing a formation for the team is done prior to any other decisions. According to policies, that are a mixture of manual configurations (and game- state dependent updated values, each cycle the coach layer decides the team's formation. Therefore, each agent takes part in one of the main plans: defense and offense. Defense plan consists of agents which are near the friendly penalty area, including goalie and some blocker agents. This year, we changed our plans about Marking Plans and it has been merged with Defense Plan. Offense Plan includes the remaining agents that are going to create attacking chances to score. Our major difference from last year is our attacking and offensive plan; we decided to remain the Play-Off Plans, but Play-On plan has completely changed (Will be explained in next chapter).

Coach decides how many agents are needed for Defense Plan according to agents' positions and dangerous conditions; rest of agents will be assigned to Offense Plan.

In Play-Off mode, ball doesn't move and it starts the plan as soon as referee send the commands, Play-On mode is completely different, coach uses a Visual Planner for semi static plans that will be explained in next chapter.

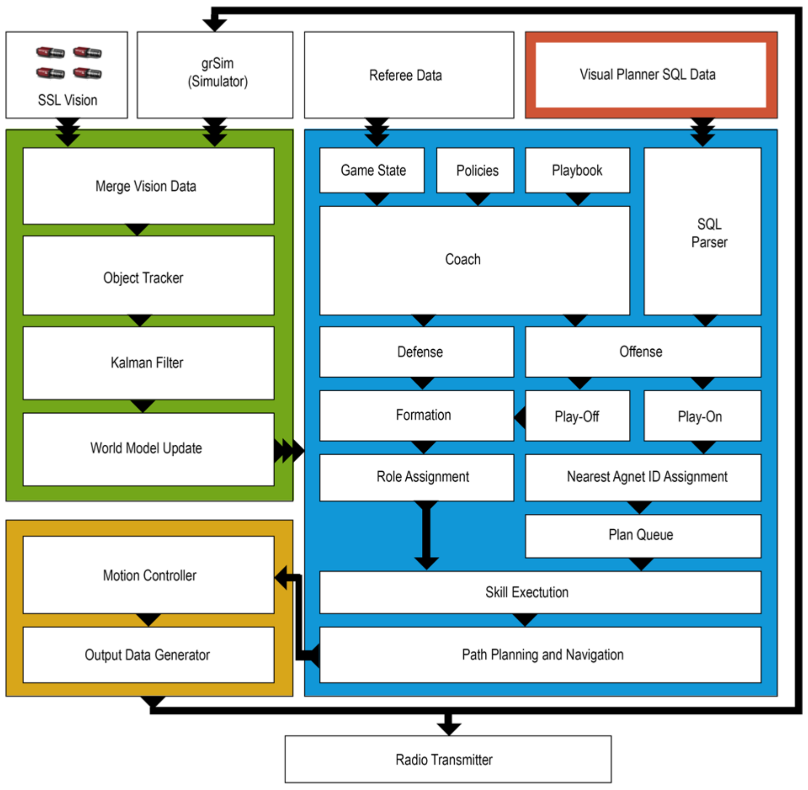


Fig.11. Software chart

5.2. Visual Planner

Since RoboCup SSL competition will be held in double sized field, it is very important to have offensive strategy to score more points and play in multi states. In normal fields it was very difficult to perform a plan completely due to small area of movement and speeds of new robots, but new fields are way easier to create plan for that actually works.

For reasons mentioned, we developed a semi static visual planner (Visual Planner). It's high level software for arranging agents according to ball position and agents conditions.

In each plan first of all we set a ball position then according to that we decide each robot task.

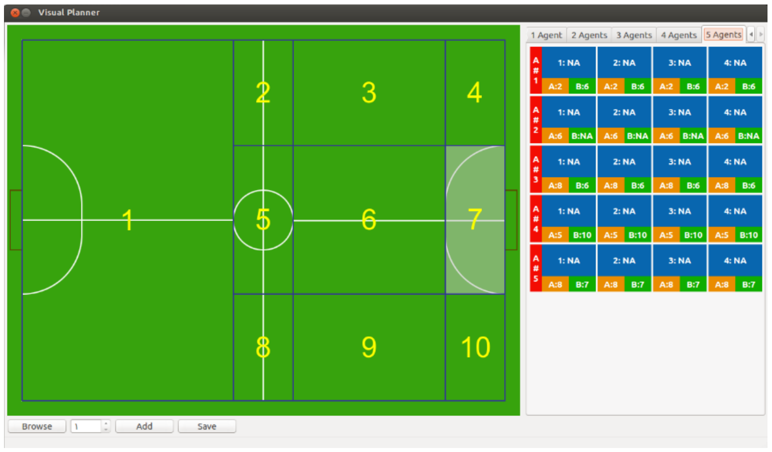
Tasks are consists of Pass, Chip, Kick, Move, and One-Touch skills and each has two offensive and defensive mode. Each agent can have up to 4 tasks and these tasks will be executed upon possibility.

The most challenging problem was to assigning ID to each robot, if we used an exact ID for each agent it would be very difficult to arrange plans for them. Best solution is to use a DFS Tree [17] to find the nearest agent to the current plan dynamically. For example assume there are two robots in area 5 and 9, and plan is to move an agent from 6 to 10, program assigns nearest robot to it and it would be the former robot.

Filed is divided to 6 sections for ball position and 10 sections for agents in order to minimal the plans.

So far it was static plan in dynamic plan, it will decide what points are best positions for starting and ending a plan loop and it will be processed in several threads in order to make it faster and more real time.

And the final step is to end the plan, each plan can be ended by two condition, time and desired conditions (for example when an exact agent receives a pass it immediately goes to next plan, and doesn't wait for all robots to complete their tasks). Advantage of this method is single state plans and it does not need to create multi-level plans that could be disturbed by opponent agents.



A#1	1: NA A:2 B:6	2: NA A:2 B:6	3: NA A:2 B:6	4: NA A:2 B:6
A#2	1: NA A:6 B:NA	2: NA A:6 B:NA	3: NA A:6 B:NA	4: NA A:6 B:NA
A#3	1: NA A:8 B:6	2: NA A:8 B:6	3: NA A:8 B:6	4: NA A:8 B:6
A#4	1: NA A:5 B:10	2: NA A:5 B:10	3: NA A:5 B:10	4: NA A:5 B:10
A#5	1: NA A:8 B:7	2: NA A:8 B:7	3: NA A:8 B:7	4: NA A:8 B:7

Fig.12. Visual planner

5.3.1. Maximum Weighted Bipartite

Matching Graph G = (V, E) in which V is the set of vertices and E is the set of edges is called Bipartite if the set V can be divided into two parts A and B such that, $A \cap B = \emptyset$ (1) $A \cup B = V$ (2) , and also there does not exist any edge in E that connects two vertices in the same set [6]. A subset M of the set E is a matching (collection of edges) when each vertex of V is at most incident to one edge of M. Without loss of generality we can assume our graph complete by adding dummy vertices and edges of weight zero [6, 7]. If each edge of the graph is assigned a weight we have a weighted bipartite graph [8]. If the sum of the weights of the edges in a matching (Mi) is called the weight of that match W(Mi),

$$W(M_i) = \sum_{e \in M_i} w(e) \quad (3)$$

A maximum weighted matching M is a match in such a way that every other matching has lower weight than the weight of M [9]. In the proposed algorithm we employ Hungarian method to solve our assignment problem. This Maximum Weighted Bipartite Matching has been used in defensive skills in [8] and [9] as well.

5.4.2. Applying “MWBM” in our marking problem

At first we make our graph consisting of our defenders at one side and the attackers on the other side as the nodes. Be mentioned that we omit some of opponent’s attackers by some initial checking. We don’t use our blocker in our nodes either. Then we give a weight $\mathbf{W_{(i,j)}}$ to each edge respecting some features. The weight in fact indicates the importance of marking the attacker j by our defender i. In fact if $\mathbf{F_k}$ is the value of feature k and $\mathbf{R_k}$ is the related coefficient and **N** is the number of the features, the weight of each edge is the sum of the features value multiplied by their related coefficients as shown in formula (4).

$$W_{(i,j)} = \sum_{k=1}^{N} R_k * F_k(i,j) \quad (4)$$

The coefficients are now set by hand, but we look forward to applying training procedures and high level algorithms for optimizing them.

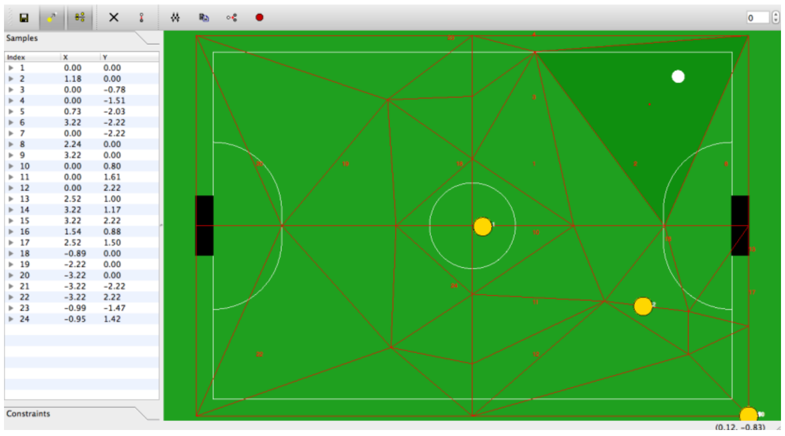
5.4. Dynamic Path Planner

As it's been described in our previous RoboCup ETDP, we create a Dynamic Environment Path Planning algorithm and since the RRT algorithm is the best choice for path planning in non-discrete environments [1][2], we decided to implement it based on RRT instead of dividing the playground to discrete grid, so we re-implement our ERRT path planner and add some new features to that, and it became a new path planner consisting these new features.

First of all we consider the field a little larger because of some circumstances in which path planning inside the field is impossible like when the ball is in corner area. Since the primary path retrieved from ERRT has fractures on account of its random nature, straightening the fracture containing path is done by the basis that says "go forward as much as possible". In order to lessen computation time algorithm starts to generating path on both robot-to-destination and destination-to-robot. One thing that should be considered in using the obtained path is preventing robot from exiting path while it traverses its immediate turns. We handle this matter by limiting the maximum velocity of the robot by the angle of fracture and the distance to reach that. Applying the proposed method in 2013 which resulted in achieving the first place of RoboCup 2013 navigation technical challenge, confirms its efficiency and effectiveness.

5.5. Offensive Positioning

Positioning involves finding the best target position for agents who do not possess the ball regarding the field situation and team strategy [1]. In offensive positioning which is the case when one agent of us owns the ball, all the other agents should modify their position in field in order to increase the chance of receiving pass from the ball owner and increase the scoring chance. Therefore, considering the minimum effective movement in positioning is a crucial task. Here we focus our attention to a widely used algorithm of positioning in RoboCup Soccer Simulation League, Delaunay Triangulation based positioning depending on the ball's position in the field, and we propose a modification to implement it in the offensive situation positioning of our small size soccer robots. In 2008 Hidehisa Akiyama and Itsuki Noda used Delaunay Triangulation of the field depending on the ball position to determine the agents' positions [2-4]. In this method a representative set of potential ball positions are the vertices of these triangles and in each potential ball position we can set the positions of all of our agents. During the match the positions of the agents are determined by an interpolation method between the vertices of the triangle in which the current position of the ball lies. This method has shown a great performance and was adopted by nearly all the teams participating in Soccer Simulation League of Robocup Competitions in recent years. One of the benefits of this method is its simplicity in implementation and initialization by the developer, since it can be done by a visual software namely fEdit provided by Hidehisa Akiyama [5] and all the considerations of smooth movement of players between positions can be handled by the human intuition operation in a visual manner. In Fig.13 the Delaunay Triangulation of the field based on the representative set of ball positions in the GUI of fEdit is shown. In each numbered potential position of the ball, the human developer determines the positions of all the agents. We modified some parameters in this software like the field dimensions and number of players to comply with SSL.



Index	X	Y
1	0.00	0.00
2	1.18	0.00
3	0.00	-0.78
4	0.00	-1.51
5	0.73	-2.03
6	3.22	-2.22
7	0.00	-2.22
8	2.24	0.00
9	3.22	0.00
10	0.00	0.80
11	0.00	1.61
12	0.00	2.22
13	2.52	1.00
14	3.22	1.17
15	3.22	2.22
16	1.54	0.88
17	2.52	1.50
18	-0.89	0.00
19	-2.22	0.00
20	-3.22	0.00
21	-3.22	-2.22
22	-3.22	2.22
23	-0.99	-1.47
24	-0.95	1.42

Fig.13. Delaunay Triangulation of the field

References

[1] Hesam T. Dashti, Shahin Kamali and Nima Aghaeepour (2007). Positioning in Robots Soccer, Robotic Soccer, Pedro Lima (Ed.), ISBN: 978-3-902613-21-9, InTech, Available from: http://www.intechopen.com/books/robotic_soccer/positioning_in_robots_soccer

[2] Akiyama, H., Noda, I.: Multi-agent positioning mechanism in the dynamic environment. In: RoboCup 2007: Robot Soccer World Cup XI. (2008)

[3] H.Akiyama, I. Noda , H.Shimora, Helios 2008 Team Description Paper, RoboCup 2008.

[4] H.Akiyama, Helios 2007 Team Description Paper, RoboCup 2007.

[5] Part of Helios team released materials, available at: http://sourceforge.jp/projects/rctools/

[6] Lecture notes from Michael Goemans class on Combinatorial Optimization. http://math.mit.edu/~goemans/18433S09/matching-notes.pdf, 2009.

[7] M. Paul. Algorithmen für das Maximum Weight Matching Problem in bipartiten Graphen. Master's thesis, Fachbereich Informatik, Universität des Saarlandes, Saarbrücken, 1989.

[8] M.Norouzitallab, A.Javari, A.Noroozi, S.M.A. Salehizadeh, K.Meshgi, Nemesis Team Description Paper 2010, RoboCup 2010.

[9] M.Malmir, M.Simchi, S.Boluki, AUT Team Description Paper 2012, Robocup 2012.

[10] - the industry standard for high performance graphics (2011), http:// www.opengl.org/, [accessed February, 2011]

[11] OpenGL 2. Browning, B., Bruce, J. Bowling, and M. Veloso.: STP: Skills, tactics, and plays for multi-robot control in adversarial environments. Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering 219(1), 33{52 (2005)

[12] Bruce, J., Veloso, M.: Real-time randomized path planning for robot navigation. Lecture Notes in Computer Science pp. 288{295 (2003)

[13] Bruce, J., Veloso, M.: Safe multi robot navigation within dynamics constraints. Proceedings-IEEE 94(7), 1398 (2006)

[14] S.Mehdi Mohaimanian Pour, Vahid Mehrabi, Alireza Saeidi, Erfan Sheikhi, Masoud Kazemi, Ali Pahlavani, Mohammad Behbooei, and Parnian Ghanbari.: PARSIAN extended team description for RoboCup 2013

[15] Inc.: Qt - A cross-platform application and UI framework (2011), http:// qt.nokia.com/, [accessed February, 2011]

[16] Nokia 7. Smith, R.: ODE - Open Dynamics Engine (2011), http://www.ode.org/, [accessed February, 2011]

[17] http://en.wikipedia.org/wiki/Depth-first_search

[18] Roland Siegwart and Illah R. Nourbakhsh (2004), Introduction to Autonomous Mobile Robots, A Bradford Book, The MIT Press, Cambridge, Massachusetts, London, England, ISBN: 0-262-19502-X

[19] S.Mehdi Mohaimanian Pour, Vahid Mehrabi, Alireza Saeidi, Erfan Sheikhi4, Masoud Kazemi, Ali Pahlavani, Mohammad Behbooei, and Parnian Ghanbari.: PARSIAN team description for RoboCup 2014

[20] Hai-Yun Wang, Jong-Hun Park, and Uk-You I Huh, Fuzzy-EKF for the Mobile Robot Localization Using Ultrasonic Satellite, ICCAS 2014