RoboFEI 2015 Team Description Paper

Fernando Rodrigues Jr., Danilo Pucci, Caio Schunk, Caio Braga José, Victor Torres, Thiago Silva, Victor Amaral, André De Oliveira Da Silva, Reinaldo A. C. Bianchi, and Flavio Tonidandel

Robotics and Artificial Intelligence Laboratory
Centro Universitário da FEI, São Bernardo do Campo, Brazil
{flaviot, rbianchi}@fei.edu.br

Abstract. This paper presents the description of the RoboFEI Small Size League team as it stands for RoboCup Small Size League competition 2015, in Hefei, China.
The paper contains descriptions of the mechanical, electrical and software modules, designed to enable the robots to achieve playing soccer capabilities in the dynamic environment of the Small Size League.

1 Introduction

For the RoboCup 2015, RoboFEI team intends to use basically the same electronic project that has been used during the last five years, with minor modifications. The Mechanical design is under maintenance, studies are focused to avoid the chip kick parts deformations.

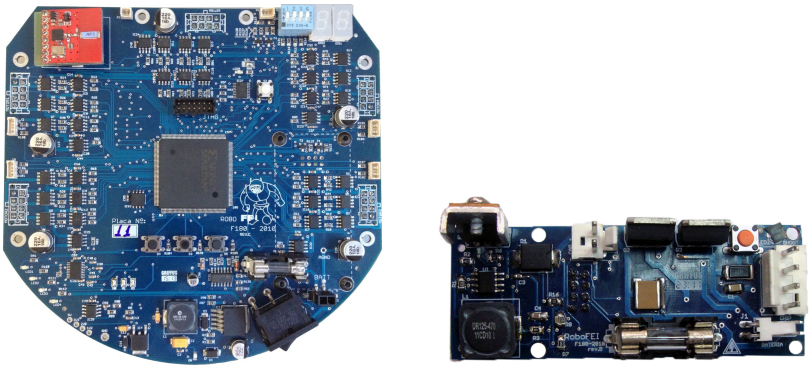
The strategy software has been refactored and turned the code more agile, easy and efficient. The next step is add new features and functionalities to improve the robot's skills. The team is developing a new intercept module approach using an Extended Kalman Filter (EKF) as a tool to estimate the ball and robot positioning along the time.

2 Electronic Design

Last year, RoboFEI's team released it's current electronic design as open source to the community. All the schematics, layouts and firmware are available at http://www.bitbucket.org/robofei, under a Creative Commons license.

The electronic design has been used for the last five years and received few modifications since reached desired performance and stage of development.

RoboFEI's electronic consists of two boards (Fig. 1): the Main Board is responsible for all embedded computation and robot's motion control. The Kicker Board commands the kicking devices and it's associated power electronics. More details about design can be found in [8] and on RoboFEI's open-source repository.



(a) Main Board (b) Kicker Board

Fig. 1. The current RoboFEI's Boards.

2.1 Radio System

The radio unit is based on the Nordic nRF24L01+ transceiver, with two 2.4GHz transceivers connected to the microcontroller, one for transmission and one for reception. However, for this year a substantial modification was made to the radio unit, upgrading the microcontroller from an ARM7 to a Cortex M4 from STM32 family. Due to a better performance and computing power, is possible to improve the algorithms used for real-time robot diagnostics, such as having the robot sending more frequent current and odometry readings from a motor that is presenting an abnormal power consumption problem, which may spot an early mechanical problem or a damaged hall sensor.

Another modification, evaluated on inside-lab tests and experiences in Latin American Robotics Competition 2014, is the improvement of radio's RF amplifiers. During RoboCup 2014, we found that the robot's transceiver (a 0dBM module), did not have enough power to transmit back to the radio due to the longe distances between the robot and radio unit in the new field. To solve this problem, as well as to simplify our radio board design requirements, we are now using a commercially available module, "nRF24L01 + PA + LNA". This module contains both a built-in 20dBm power amplifier and a low noise amplifier on the receiving side, which has enough signal amplification to ensure the robot's transceiver signal be received across the double-sized field.

2.2 Electronic Work in Progress

For this year we intent to make a new Kicker Board design. After some years using the same design, we are studying other topologies' viability such as flyback and boost converters. The actual design charges a Capacitance $5400uF$ with $160V$ in 6 seconds. We consider it a long time, as during games we need a faster charges then it (we intent to get few hundred of milliseconds or a second). Beyond that, we are looking for on-the-fly diagnostics from kicker board like

charge complete event, current measurement, charge enabler to keep the circuit on standby mode. The main requirement of new design is to keep the kicker board independent and isolated from main board avoiding signal circuit to be in contact with high voltage.

3 Mechanical Design

In compliance with the SSL rules, the height of the robot is 148 mm, the maximum percentage of ball coverage is 15% and the maximum projection of the robot on the ground is 146 mm.

Table 1. Robot's Mechanical Specifications.

Height	148 mm
Weight	2,6 kg
Percentage of ball coverage	15%
Main Material	6000 series Aluminium Alloy
Roller bar material	Polyurethane (PU): Hardness of 20, 25 and 30 Shore A
Driving motor	Maxon EC-flat 45 50W
Gear ratio	3:1
Dribbler device motor	Maxon EC-Max 22 25W
Solenoid Plunger material	SAE1020 steel
Solenoid coil	AWG21 wire

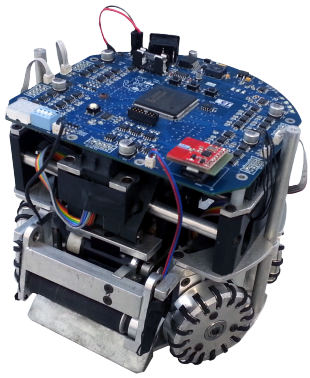


Fig. 2. The RoboFEI robot.

The current robot uses a 6000 series aluminum alloy as main material, the factor hardness/weight has a good relation and less frequent part replacements

are needed. Wheel axes and the small rollers of the omni-directional wheels are exposed to severe stress thus are made of stainless steel instead. Nylon is found in battery's supports due to electrical isolation and its lightweight.

The Robot weighs about $2.6kg$ and the general design could be seen on Fig 2.

3.1 Casing

The robot's casing, formerly made of polystyrene (PS), has been replaced by carbon fiber casing. The fiber was chosen mainly due its high resistance, lightness and longer life span.

The manufacturing of the case was made completely in our laboratory, with an external mold based on a polyurethane (PU) robot model, three carbon fiber layers positioned at forty five degrees alternated angles and a vacuum pump. The materials used were: AR260 epoxy resin and AH260 catalyzer combine with RC200P carbon fiber.

3.2 Motor Bracket and Wheel Shaft

The wheel shaft has been requiring excessive maintenance during the previous competitions, as it becomes loose from the motor bracket. This loosening causes vibration and subjects the motors to axial loads that can damage it. We then began to analyze how to solve the problem of the shaft affix into the motor bracket.

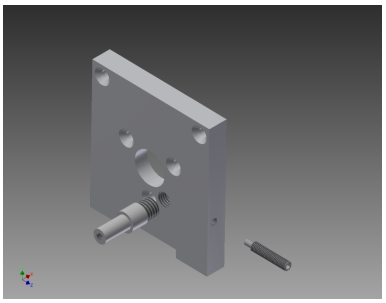
The shaft (made of stainless steel), was threaded into the motor bracket (made of aluminum), and secured by a transversal screw locking, as seen on Fig. 3(a). In this assembly, eventually the wheel shaft affix becomes loose due to the thread's deformation caused by the high tensions generated by the robot's movement during the game. Past attempts to solve the problem by interference adjustment and thread-locking glue, or with high interference and a non-threaded shaft, were not successful.

After analyzing the problem and evaluating the design from Immortals' team, we concluded that a shaft with a built-in flange, attached onto the motor bracket with two screws, as seen in Fig. 3(b), would be more resistant to impact, vibrations and tension variation.

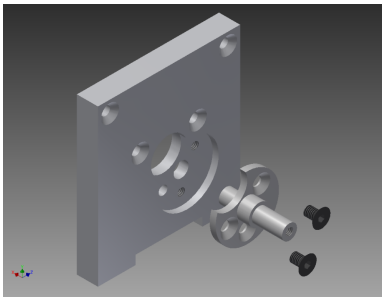
3.3 Encoder Alignment and Positioning

Another problem we faced in the past is the excessive maintenance necessary to keep the encoder aligned and positioned in order to provide accurate odometric information to the main board. The old encoder's support has been fixed only at the motor bracket's base, and its bottom face kept in contact with the inferior plate, which was not always precisely flat, causing a deformation in the support (see Fig. 4(a))

To minimize such problem we extended the support's length, fixing on a joint that links the upper region of the encoder's support with the motor bracket,

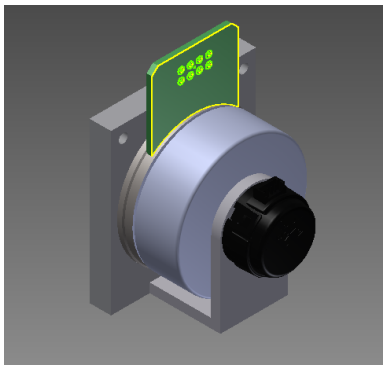


(a) Transversal Screw Locking

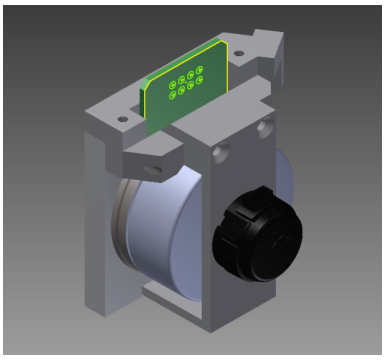


(b) Built-in Flange

Fig. 3. Motor Bracket and Wheel Shaft.



(a) Old assembly.



(b) Current assembly

Fig. 4. Enconder Assemby

providing a second fixation point. Also, to prevent the contact between the inferior plate and the encoder's support, we diminished its thickness, to ensure that small deformities in the inferior plate would not affect the encoder's support (see Fig. 4(b)).

3.4 Mechanical Work in Progress

The robots mechanics are under continuous improvement, the height of chip kick device and the roller ball handling are requiring main attention. The Chip kick system is under strong mechanical stress. After some kicks the parts starts to bend and needs to be replaced, the exceeded dissipation of energy and non-optimized geometry causes important deformations, thus we have been developing a new mechanism with the proposal to increase the lever arm distance to preserve the same torque with less force values.

Aiming at improving the quality of the ball handling, a new roller mechanism has been developed, the device consists of a inverted pendulum where the

damping is provided by a viscoelastic foam. These two new mechanisms are now in an evaluation period (see Fig. 5).

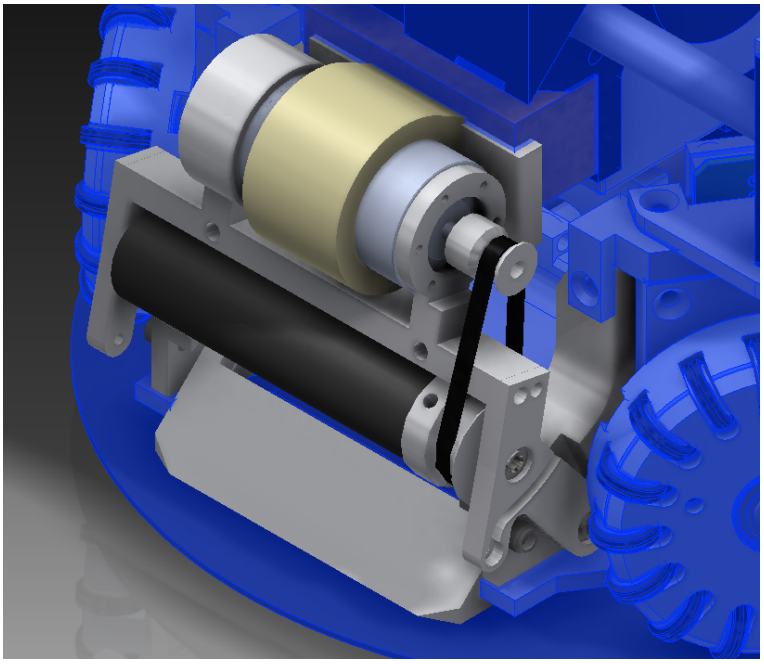


Fig. 5. New roller and chip kick prototype

4 Path Planning and Obstacle avoidance

The path planning and obstacle avoidance algorithm employed is based on the Rapid-Exploring Random Tree (RRT) with KD-Tree data structures, proposed by [1], and on the ERRT algorithm developed by [6], complemented by an algorithm to include preferred path heuristics and set the angle of approach. The algorithm based on RRT was chosen because (i) its capacity to efficiently explore large state spaces using randomization, (ii) the probabilistic completeness offered, (iii) its lookahead feature and (iv) the easiness of the algorithm's extension, when new constraints or heuristics are deemed necessary.

The add-on algorithm has the function to set the angle which the robot approaches the ending point, as commanded by the strategy layer, an item that many path planners do not treat. It is not desirable, for example, that a robot going to the ball on the defensive field accidentally hits the ball in the direction of its own goal, or yet, that an attacking robot arrives at the ball in a position in between the ball and the opponent's goal. To create a path that conforms

to the angle of approach requirement, a circular virtual obstacle centered on the ending point is created, with a 10° width circle segment and vertex at the desired angle removed. This effectively forces the path planner to create a path that reaches the ending point passing through this 10° opening. The radius of this obstacle-like constraint is set to a value close to half the size of a robot.

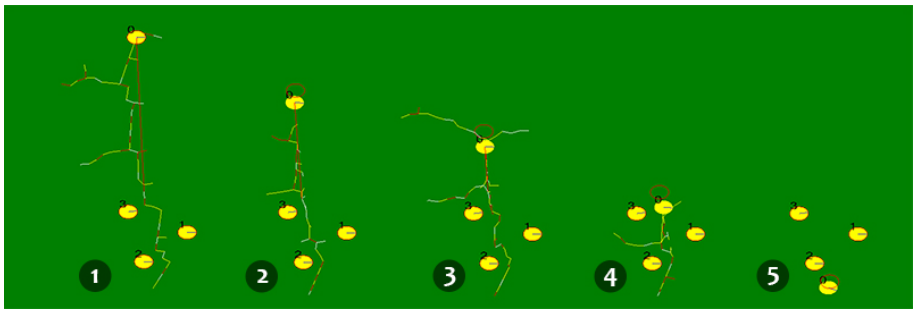


Fig. 6. Path Planning and obstacle avoidance using RRT and KD-Tree. The numbers represents the algorithm state along the time.

5 Strategy System

For this year we will use the same software structure(Fig. 7) used last year that basically consist of some world modeling blocks, logically independent agent modules, visualization and data logging blocks.

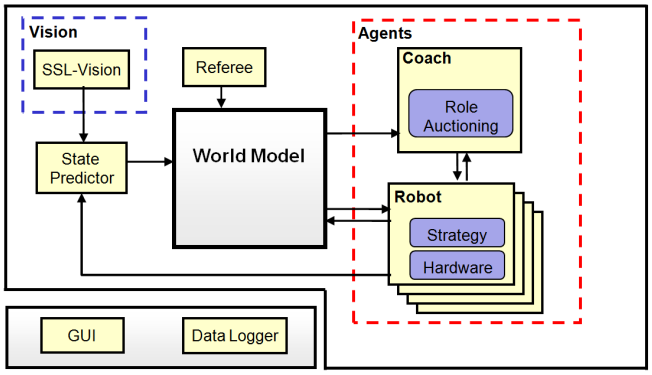


Fig. 7. The Software Diagram

5.1 World Modeling

The world model is updated by the state predictor module. This module receives vision data from the SSL-Vision and motion command data from the agent modules, sent when they command the robots via radio, and performs state predictions. The prediction is to advance the positions sent by the SSL-Vision from their original capture time to the present and then forwarding one strategy cycle in the future, the so called latency of the strategy system. This latency is currently on the order of 80ms.

The prediction algorithms used for ball, robots and adversaries are different. For this year, the prediction algorithms used for ball has been updated due to the camera overlap problem occured in RoboCup 2014 - Brazil.

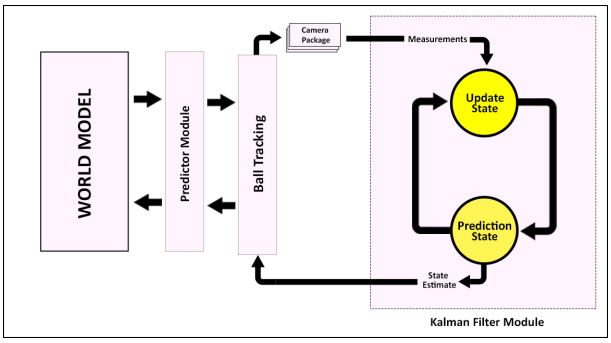


Fig. 8. Extended Kalman Filter Module Diagram

The team had problems filtering the ball along the double size field, the increase of overlap regions on the new field made our vision module unstable, the ball has been missed very often during the game. As a solution, the team has been refactoring the module that receives the information from the SSL-Vision. A new tracker has been implemented aiming to a better compatibility and software modularity. The tracker uses an Extended Kalman Filter (EKF) (see [4]), implemented from scratch using the Eigen library [1](Fig. 8).

To ensure the filter convergence, the ball friction coefficient must be measured and the variances configured according to the presence of robot near the ball. Aiming to evaluate the filter output, the ball velocity has been obtained using different methods (Fig. 9). The Kalman Filter produced a smooth velocity response with a lower noise level.

The robot's prediction is performed similarly to [2], with multi-layer percetion neural networks. These networks are trained off-line to learn the robot's motion model, receiving past frames and motion commands as input and a frame n steps in the future as output. Once trained, the networks are used for on-line estimation of the robot's position and rotation.

[1] http://eigen.tuxfamily.org

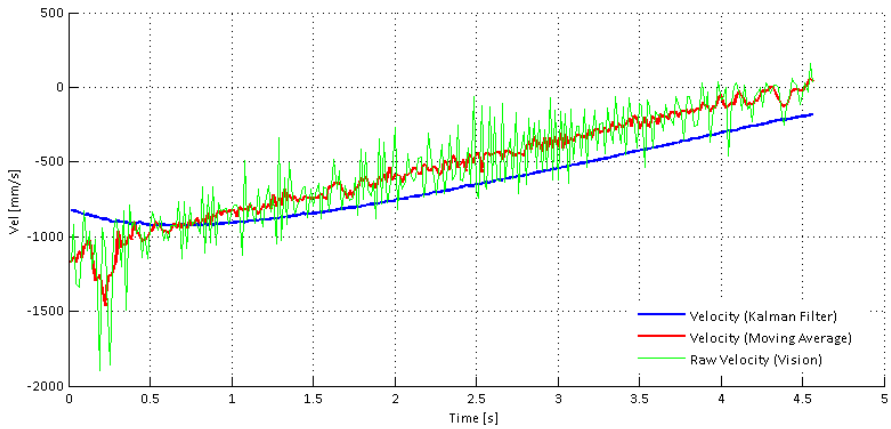


Fig. 9. Ball Velocity estimation using a Kalman Filter

As for opponent estimation, currently it is done with simple extrapolation of the last velocity data and Gaussian functions.

5.2 Agent Modules

Each robot player is an independent module, executing its own instance of one or more strategy submodules and its hardware specific functions (such as motion control and sensing). The current implementation relies basically on a layered strategy architecture and a market based approach for dynamic allocation of functions, both described ahead on this section.

5.3 Strategy module

Building multi-agent systems in a layered architecture with different levels of abstraction is a popular approach (see [7], [3] and [5]) well suited as foundation for machine learning algorithms, one of the research goals. For this reason, the strategy module architecture was divided in three abstraction layers.

The lowest layer has the so called *Primitives*. Primitives are actions that mostly involve directly activating or deactivating a hardware module such as to kick the ball with a given strength, activate the dribbling device, rotate or move to a position.[2]

On top of the primitive layer, is the *Skills* layer. Skills are also short duration actions but involving use of one or more primitives and additional computation, such as speed estimation, forecasting of objects' positions and measurement of primive tasks' completion. This layer has a small set of skill functions, yet that

[2] Actually, moving to a position is a special case of a primitive with underlying complex logic. It calls the path planning system to perform obstacle avoidance.

represent the basic skills required in a robot soccer game, like shooting the ball to the goal (aiming where to shoot), passing the ball to a teammate, dribbling, defending the goal line or tackling the ball (moving toward the ball and kicking it away).

One example of such skill is the Indirect Free Kick skill, which employs a multi-criteria weighted evaluation to determine the best for the robot to pass the ball to. Grids are constructed in different areas of the field, then the weighted multi-criteria evaluation function is employed to decide which of the grids contains the best candidate position. Once the area is chosen, the function recomputes using a finer grid, to determine the exact position. The objectives evaluated are the Euclidean distance of each position, in relation to both the robot and the ball, the width of the angle a robot in that position would have to to kick to the goal and the distance between the current positions of the teammates receiving the pass and the chosen positions in the grid.

The skills are employed by the *Roles* layer, which contains different roles, created using combinations of skills and the logic required to coordinate their execution. This logic uses a FSM (finite state machine) concept with a basic score atribuition to different actions. The decisions are made according to some game variables such as the ball possession, enemy distance and goal possibilities.

There are roles called fullback, defender, midfielder, striker, forward and attacker. No particular robot is tied to a given role (except the goalkeeper), and there is no limitation on how many instances of the same role can exist, what allows dynamic selection mechanisms to create role combinations without restrictions.

5.4 Strategy Work In Progress

About two years ago, we started a code refactoring project, with a new major software version. The results are very positive. With the refactoring, the programming code is once again consistent with its conceptual design, what removed several glitches and deadlocks, and also allowed us to view and fix parts of the program that indeed had conceptual errors. One of the basis of the new code is a continuous code review system, by which at least one reviewer (among the programmers of the team) must sign-off the code before it is released, avoiding problems in code readability, such as lack of clarity and comments, as well as standardization and performance problems. The review system has proven so positive that we intend to try tool for Git repositories called Gerrit [3], that controls the work-flow of peer code reviews and controls them before the actual commits to the main repository.

Eigen [4], an optimized scientific library, has been implemented to provide linear algebra support, with matrices operations, numerical solvers and optimization algorithms. The library implementation is recent and some tests such as performance and stability are currently in progress.

[3] http://code.google.com/p/gerrit/
[4] http://eigen.tuxfamily.org

A new Intercept Module is under development aiming to increase the robot capabilities on catching faster balls and providing passes with efficiency.

Acknowledgments

We would like to thank, in advance, the Small Size League Committee, for the consideration of our material. We would like also to immensely thank the staff of Centro Universitário da FEI, for all the help we always received.

References

1. A. Atramentov and S. M. LaValle. Efficient nearest neighbor searching for motion planning. In *IEEE International Conference on Robotics and Automation*, pages 632–637, 2002.
2. Sven Behnke, Anna Egorova, Alexander Gloye, Raul Rojas, and Mark Simon. Predicting away robot control latency. In *Proceedings of 7th RoboCup International Symposium*. Springer, 2003.
3. Michael Bowling, Brett Browning, and Manuela Veloso. Plays as effective multiagent plans enabling opponent-adaptive play selection. In *Proceedings of International Conference on Automated Planning and Scheduling (ICAPS'04)*, 2004.
4. Brett Browning, Michael H. Bowling, and Manuela M. Veloso. Improbability filtering for rejecting false positives. In *ICRA*, pages 3038–3043. IEEE, 2002.
5. Brett Browning, James Bruce, Michael Bowling, and Manuela Veloso. Stp: Skills, tactics and plays for multi-robot control in adversarial environments. In *IEEE Journal of Control and Systems Engineering*, volume 219, pages 33–52, 2005.
6. James Bruce and Manuela Veloso. Real-time randomized path planning for robot navigation. In *Proceedings of IROS-2002*, 2002.
7. Maja J Mataric. Learning in behavior-based multi-robot systems: Policies, models, and other agents. *Cognitive Systems Research*, pages 81–93, April 2001.
8. Danilo Pucci, Fernando Rodrigues Jr., Caio Schunk, Caio José Braga, Victor Torres, Thiago Silva, Victor Amaral, André De Oliveira Da Silva, José Angelo Gurzoni Jr., Reinaldo A. C. Bianchi, and Flavio Tonidandel. Robofei 2014 team description paper. 2014.