2020 Warthog Robotics VSS Project Team Description Paper

Rafael Lang, Guilherme Oliveira, Heloisa Barbosa, Adam Pinto, Pedro Pizarro,
Arthur Demarchi, Lucas Camolezi, Leonardo Altemari, Ivan Silva, Roseli Romero

***Abstract*— This paper presents the Warthog Robotics Sub-Zero project, developed since 2013 at the University of São Paulo at São Carlos, and the main improvements made to this RoboCup VSS project since the last competition. The project holds two Latin American championships and is under active development by the Warthog Robotics group. The mechanical structure is a mix of 3D-printed parts and composite materials and accommodates two brushed DC motors for locomotion. The system architecture is based on the GEARSystem library, with a decision tree strategy module, and powered by some filtering algorithms on the vision module. During the last development cycle, most of the development was focused on software and simulations since the current situation, brought up by the corona virus, made any hardware modifications intangible. Therefore, modifications here mentioned will cover a navigation algorithm, vision improvements and a strategy tuning. The team presents full game capability with accurate and fast responses to strategy and referee commands. *Index Terms*— obile Robotics, RoboCup, Artificial Intelligence, Embedded Electronics, Warthog Robotics.obile Robotics, RoboCup, Artificial Intelligence, Embedded Electronics, Warthog Robotics.M**

I. Introduction

The Warthog Robotics is a research and development Robotics group from the University of São Paulo at São Carlos. The group counts with over 100 members from several knowledge areas, such as Computer Science and Electrical, Mechatronic and Computer Engineering, and develops Robotics technologies in several areas, applying most of them at robotics competitions. Due to budget limitations, the team was not able to participate in the last two editions of the RoboCup, but is still competing in local competitions. The WR Sub-Zero is the IEEE VSS robot of the Warthog Robotics group and its mechanical structure and electronic boards are the same from the last years, of which detailed information can be found in [4]. The next sections briefly describe the robots and its supporting systems, as well as present the newest improvements of the WR Subzero project.

II. Mechanical Structure

The internal mechanical structure of the robots is exactly the same one from the last years, composed of a locomotion system with two faulhaber 2224 [6 Volts] that were implemented with a reduction of 12:120. The 3D-Printed part houses the electronic board, the battery, and all the system wiring. All mechanical structure is made of printed material (painted in black) and composite materials as shown in figure 1.

[1]All authors are with Warthog Robotics Team. University of São Paulo at São Carlos, 400 Trabalhador São-carlense Ave, São Carlos, São Paulo, Brazil info@wr.sc.usp.br

Fig. 1. Mechanical assembly of the 2020 Warthog Robotics VSS robot.

III. Electronic Devices

The electronic devices are the same of last years, composed of a single electronic board. Detailed information about the boards can be found in [1], [2] and [3]. The architecture of the embedded electronics is shown in figure 2. Also the hardware implementation is shown in figure 3

IV. Computer Systems

The WR SubZero project software is based on five sub-projects developed by the group: the GEARSystem library, the WRBackbone server application, the WRCoach strategy application, the WREye vision filtering application, and the WRStation radio communication application.

A. GEARSystem

The GEARSystem distributed library [11] is used to provide communication and distributed execution between system modules, as it focuses on remote procedure calls

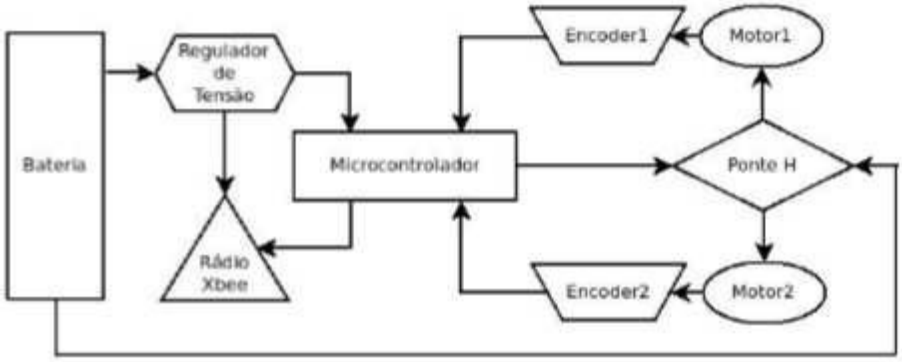

Fig. 2. Block diagram of the embedded electronic systems of the 2020 Warthog Robotics VSS robot.

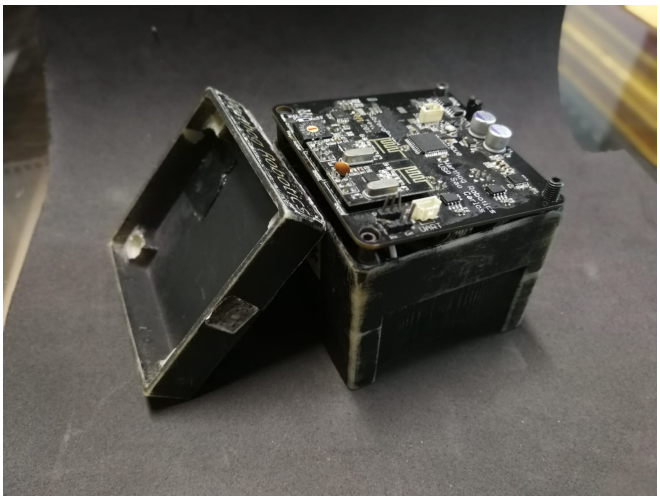

Fig. 3. Implementation of the embedded electronic systems of the 2020 Warthog Robotics VSS robot.

is presented in figure 4. A full description of the software is available in [2], [4] and in Brazilian Portuguese at [13].

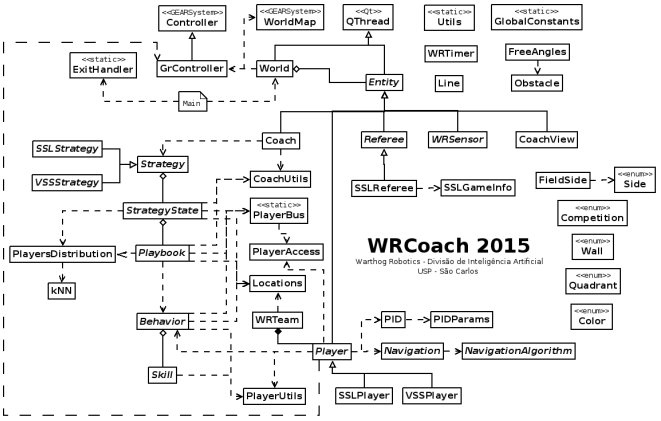

Fig. 4. Simplified diagram of the modules of the WRCoach software.

encapsulated as Robotics-related methods. This architecture allows the easy development of new software based on these main modules, and detailed information can be found in [1] and [2].

In the current implementation of the team, the Sensor module is the WREye, responsible for receiving the data from ssl-vision and inserting it on the system. This module is composed by filters (Kalman, Noise, Loss and Multi Object) and more detailed explanation can be found in [2]. The Server module is WRBackbone, connection all modules on the GEARSystem architecture. The Actuator module in the WRStation, sending commands via USB to a custom station board that wirelessly tunnels them to the robots. The commands sent to the robots are generated by WRCoach, the Controller module on GEARSystem architecture.

B. WRCoach

The coach is responsible for setting the strategy to the team: understanding the world model, defining behaviors, generating navigation paths and sending them to the system actuators. A simplified diagram of the software architecture

V. Improvements for 2020

A. Vision

Our decision-making software (WRCoach) requires a stable set of data from the game (robot positions, ball position) to produce accurate decisions. This data originates from video captured by cameras on top of the playing field. The data is then processed by the standard RoboCup shared vision system: ssl-vision. This system can distinguish different robots based on color patterns that are attached to the top of robots. Additionally, the software is capable of identifying the position of the ball.

Occasionally the data from SSL-vision can be noisy, depending on the light conditions. That's why we developed WREye. WREye is a software that is responsible for an extra layer of data processing, it employs filters on the result data of ssl-vision to decrease potential noise and instability before this information reaches WRCoach. To accomplish its task, WREye uses a combination of filters, which the most important is a Kalman filter implementation.

1) Kalman Filter: Kalman filter is a continuous cycle of predict-update used in WREye to ball and robots position prediction. The following equations form the main loop of the filter:

$$X_{t,t-1} = \Phi X_{t-1,t-1} \tag{1}$$

$$S_{t,t-1} = \Phi S_{t-1,t-1}\Phi^T + Q \tag{2}$$

$$K_t = S_{t,t-1}M^T[MS_{t,t-1}M^T + R]^{-1} \tag{3}$$

$$X_{t,t} = X_{t,t-1} + K_t(Y_t - MX_{t,t-1}) \tag{4}$$

$$S_{t,t} = [I - K_tM]S_{t,t-1} \tag{5}$$

The output from the filter is the result of the state update (Equation 4) and state covariance update equations (Equation 5). These provide the combined estimate from the model (prediction equations) and latest observation (measurements). The state matrix provides the mean value of the distribution for each state variable, and the covariance matrix provides the variances.

In calculating the gain and update, it ix helpful to break the calculations in the following meaningful manner. First, it is calculated the "innovation", which is the difference between the new data and the previous prediction.

$$J = Y_t - MX_{t,t-1} \tag{6}$$

Then calculate the covariance of the innovation. The Equation 8 and Equation 9 calculates the gain matrix and the updated state.

$$\mathrm{COV}(J) = MS_{t,t-1}M^T + R \tag{7}$$

$$K_t = S_{t,t-1}M^T\mathrm{COV}(J)^{-1} \tag{8}$$

$$X_{t,t} = X_{t,t-1} + K_tJ \tag{9}$$

B. Strategy Tuning

Among some minors changes in WRCoach, we realized that the number of players in the field was not being fully explored, we couldn't get all the potential of a full team working together, almost all plays were being executed by only two players. With that in mind it was implemented in code the idea of a outfield goalkeeper, which consists in making the goalkeeper behave as a defender, making it possible to the other defender to make play more aggressively.

This implementation starts from the idea of making a triangulation with the position of the defense players, always leaving a player on the field free to make plays and help others. It consists in making two game states: one state in which the team is attacking, there being an attacker, an assistant and a defender, and another state in which the team is defending, there being a defender, a barrier and a goalkeeper. This type of game makes the robots volatile and adapted to the state the game is in.

For this implementation it is assumed that the change of states will occur and the triangulation will take place as follows: when the game passes from the state of defense to the state of attack the goalkeeper turns the defender and the defender turns the assistant, when the game passes from the state of attack to the state of defense, the assistant who was in the attack turns the goalkeeper making the triangulation complete. This can be done, because while the defender is waiting for the ball to enter the defense field he acting as a blocking agent, meanwhile the assistant will reach the goal in time.

In summary, before the modification we had a static formation designed as: 3 players, each with a behaver - 1 striker, 1 defender, 1 goalkeeper. A solid formation, that had a disadvantage of not giving enough freedom to the robots, limiting the possibility of implementing plays at the strategy level.

At the present time the game is defined as follows: Offense - 1 attacker, 1 assistant, 1 barrier and Defense - 1 defender, 1 barrier, 1 goalkeeper.0

In while in the offense state the striker will take the ball to the goal and the assistant will accompany him so that if there is a rebound, he can turn the striker and finish the goal. In this state he will also have a barrier in defense that will be our first defense in case the enemy team manages to catch the ball. If the enemy team hits the ball to our field or takes the ball to try to score the goal the barrier will turn into a basic defender who will try to block the passage and resume possession of the ball, during this the assistant will go to the goal to make a save in case the defender does not achieve his goal and the attacker who was further away will return to the defense field to help the defender.

At a strategy level the game can be easily changed to cover a greater variety of plays concluding to our main goal and enabling our robots to be even better in the future with the possibility of passing among other ideas.

C. Potential Fields

Also a navigation with Potential Fields was proposed, the Potential Fields algorithm has the goal to take a body(robot) from one point to another, trying to pass through the best path and avoiding obstacles. For this, it is based on the concept of electric charges, in which there are attractive and repulsive forces, which in a final sum, generate a resulting force. This way, it determines the trajectory from a start position to a destiny position.

Its basic functionalities being based on an electric field, makes each obstacle that is added represented as a repulsive charge, and the destination point, as an attractive charge. This way, each of the points that is added makes an imaginary force on the object that is using this algorithm.The direction to be followed is the resulting sum of forces. The attractive point makes a fixed force, and the repulsive one varies according to the distance the robot is from obstacles. This relation is given by an inversely proportional relationship. The module of the repulsive force at the same distance is different for each type of obstacle.

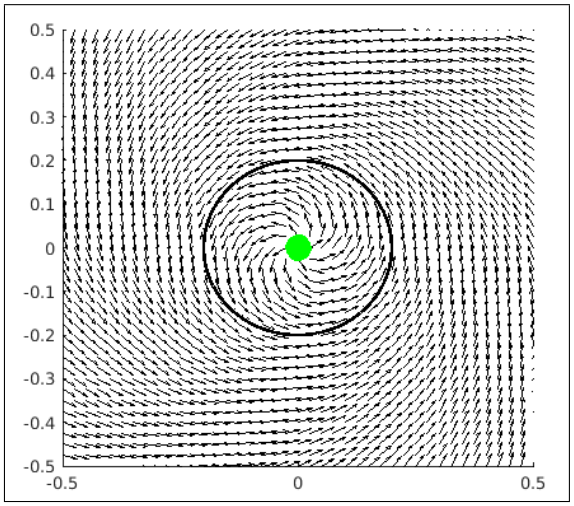

Fig. 5. Hyperbolic spiral univector field generated by our implementation, with the parameter d_e represented by the circle.

As additional functionalities a modification was proposed to solve a problem contained in the Potential Fields algorithm, the algorithm was modified creating features to correct the problem situation and adapt it better to our reality. This situation occurs when the robot and the destination point are aligned with an obstacle between them. This, due to the fact that the attractive vector is, in general, larger than the repulsive one, causes a collision.

To avoid the situation explained above, that our robot collides with an obstacle in its path, a function was created, *PF::isOnCollisionRoute()*, which checks if there will be a collision. To make this happen, it calculates, for each obstacle, whether the current route is a collision route. After that, the function evaluates if the distance to the destination is lower than the distance to the obstacle. If there will be a collision the function returns true as value.

If said collision is bound to happen the function *PF::handleCollision()* is responsible for modifying this vector resulting from Potential Fields, this way avoiding the collision. For this, the Free Angles algorithm is used in order to find a free path to be the new resulting direction. The choice of the free direction is the end of a free angle, so that these ends are already calculated with the purpose that the space is enough for the robot pass with a good margin from the obstacle.

D. Univector field-based algorithm

The Univector-based Field Navigation was proposed in [16] and is consists in a combination of two main potential fields: a *move-to-goal univector field*, and a *avoid-obstacle univector field*. The first one is by itself composed by two *hyperbolic spiral univector fields*, each one in a rotation direction (clockwise and counter clockwise).

The Equation 10 defines the hyperbolic spiral, where p is the current position, θ is the angle from x-axis at the position p, K_r is the adjustable parameter, ρ is the distance between the origin and the position p, and d_e is the predefined radius that decides the size of the spiral. In the 5 it can be seen the parameters generated by our implementation. If K_r becomes larger, the spiral becomes smoother. The notation $\pm$ represents the direction of motion, where $+$ means clockwise, and $-$ means counter clockwise.

$$\phi_h(p, d_e) = \begin{cases} \theta \pm \frac{\pi}{2}(2 - \frac{d_e + K_r}{\rho + K_r}), & \text{if } \rho > d_e \\ \theta \pm \frac{\pi}{2}\sqrt{\frac{\rho}{d_e}}, & \text{if } 0 \leq \rho \leq d_e \end{cases} \tag{10}$$

The move-to-goal univector field is a composition of two hyperbolic spiral univector fields, positioned to form a navigation field that can generate the direction vector to control the robot orientation when reaching the target position. The Equation 12 defines the velocity vector angle for a position, using the ϕ_h from Equation 10, where $N_h(p, d_e)$ is defined in Equation 11, y_l and y_r are defined in Equation 13, and p_l and p_r are defined in Equation 14, with *cw* refering to clockwise, and *ccw* to counter clockwise.

$$N_h(p, d_e) = [cos\phi_h, sin\phi_h]^T \tag{11}$$

$$\phi_{TUF} = \begin{cases} \angle \frac{y_l N_{h(CCW)}(p_l, d_e) + y_r N_{h(CW)}(p_r, d_e)}{2d_e} & \text{if } -d_e \leq y < d_e \\ \phi_{h(CW)}(p_l, d_e) & \text{if } y < -d_e \\ \phi_{h(CCW)}(p_r, d_e) & \text{if } y \geq d_e \end{cases} \tag{12}$$

$$y_l = y + d_e \ , \ y_r = y - d_e \tag{13}$$

$$p_l = [x, y - d_e]^T \ , \ p_r = [x, y + d_e]^T \tag{14}$$

The avoid-obstacle univector field is the same as the classical potential field method, with the exception that there isn't decay with the distance because all the vectors are unitary. We can define it as in Equation 15, without the implementation of the virtual obstacle position described in [16].

$$\phi_{AUF} = \angle(p - p_{obst}) \tag{15}$$

To compose the move-to-goal (ϕ_{TUF}) with the avoid-obstacle univector field (ϕ_{AUF}) a Gaussian function, defined in Equation 16, is used as a compound ratio in the composition process along with the distance between a robot and a obstacle, where d_{min} defines a minimum distance in which the move-to-goal univector field can be applied. The avoid-obstacle univector field is solely applied if the distance is less than d_{min} from the center of the repulsive obstacle, and the composition is applied outside the distance d_{min}.

$$G(r, \delta) = e^{-\frac{r^2}{2\delta^2}} \tag{16}$$

$$\phi = \begin{cases} \phi_{AUF}, & \text{if } R \leq d_{min} \\ \phi_{AUF} G(R - d_{min}, \delta) + & \\ +\phi_{TUF}(1 - G(R - d_{min}, \delta)), & \text{if } R > d_{min} \end{cases} \tag{17}$$

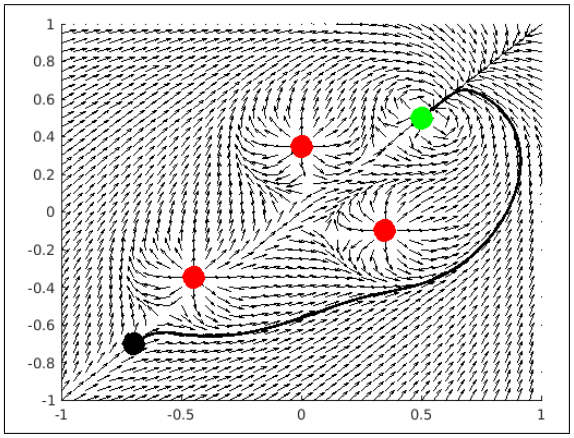

Fig. 6. Composed univector field and generated path for direction vector of $\frac{3\pi}{2}$ with three obstacles.

Using our implementation, the Figure 6 shows the results of the composed univector field with path generation for direction vector of $\frac{\pi}{4}$ with three point obstacles.

VI. Conclusion and Future Work

Following the past few years budget directives of the group, the development made in 2020 focused in software and firmware. Minor and medium improvements and bug fixes were made to the WR Coach software.

The current robot is robust, reliable and provides an excellent platform to the strategy systems. The implemented telemetry system will provide an important tool for further refinement and improvement in newer versions of the firmware.

Future works shall focus in software improvements, including the finalization of the opponent path prediction and a new study on embedded controllers efficiency.

VII. Acknowledgments

The authors would like to thank all Warthog Robotics team for the hard work and friendship; the University of São Paulo for the facilities and financial support; the Embraer, OSG Sulamericana, Griffus and SolidWorks companies for the technical sponsorship; and all others that helped us during the project.

References

[1] Lang, R.G., Bernardo, A.M., Oliveira, G.C., Menezes, H.B.B., Ramos, L.C., Roque, L.G.S., Silva, I.N., Romero, R.A.F.: Description of the Warthog Robotics 2015 project. In: 2015 RoboCup (2015)

[2] Lang, R.G., Oliveira, G.C., Menezes, H.B.B., Rosa, N.S., Correa, R.A., Gomes, V.H., Silva, I.N., Romero, R.A.F.: Description of the Warthog Robotics 2016 project. In: 2016 RoboCup (2016)

[3] Lang, R.G., Oliveira, G.C., Barbosa, H.J., Rosa, N.S., Cepeda, B.H., Siqueira, A.H., Silva, I.N., Romero, R.A.F.: Description of the Warthog Robotics 2017 project. In: 2017 RoboCup (2017)

[4] Lang, R.G., Oliveira, G.C., Barbosa, H.J., Pinto, A.H.M., Rosa, N.S., Siqueira, A.H., Silva, I.N., Romero, R.A.F.: Description of the Warthog Robotics 2019 project. In: 2019 RoboCup (2019)

[5] Nordic Semiconductor: High Frequency 2.4 GHZ Wireless Transnciever. Data Sheet (2007)

[6] Olivera, V.A., Aguiar, M.L., Vargas, J.B.: Sistemas de Controle - Aulas de Laboratório. EESC-USP, Brasil. (2005)

[7] Aguiar, M.L.: SEL359 - Controle Digital,2015. EESC-USP, Brasil. (2015)

[8] Pressman, A.I.: Switching Power Supply Design. McGraw-Hill. (2003)

[9] Mohan, N., Undeland, T.M., Robbins, W.P.: Power Electronics - Convertes Application and Design. Wiley (2002)

[10] Tse, C.K.: Complex Behavior of Switching Power Converter. CRC Press. (2003)

[11] Lang, R.G., Romero, R.A.F., Silva, I.N.: Development of a Distributed Control System Architecture. In: 2014 Latin American Robotics Symposium. (2014)

[12] Furlan, M. S., Silva, I.N.: Comparação de Controladores PID em Sistema de Malha Dupla em Robôs Omnidirecionais. In: 2018 Simpósio Internacional de Iniciação Científica da Universidade de São Paulo. (2018)

[13] WRCoach v2 documentation. Division of Artificial Intelligence - Warthog Robotics, available at `https://www.assembla.com/spaces/warthog-dia/wiki/WRCoach_v2`.

[14] Wikipedia: PID Controller, available at `https://en.wikipedia.org/wiki/PID_controller`.

[15] Control System Labs: Discrete-time PID Controller Implementation, available at `http://controlsystemslab.com/discrete-time-pid-controller-implementation/`

[16] Y. Lim, S. Choi, J. Kim, and D. Kim, "Evolutionary Univector Field-based Navigation with Collision Avoidance for Mobile Robot", The International Federation of Automatic Control, vol. 17, pp. 12787-12792, 2008.

[17] ARCHIVE Robocup. Palo Alto: RoboCup Federation, 2019. Disponível em: ¡http://archive.robocup.info/Soccer/Simulation/2D/logs/RoboCup/¿. Acesso em: 14 out. 2018.

[18] LEE, J.-G.; HAN, J.; WHANG, K.-Y. Trajectory clustering: a partition-and-group framework. In:ACM.Proceedings of the 2007 ACM SIGMOD international conference on Managementof data. [S.l.], 2007. p. 593–604.

[19] YARN, Z.2D Log Mining. [s.n.], 2018. Disponível em: ¡https://github.com/zenoyarn/2DLogMining¿. Acesso em: 14 out. 2018.