# *UTBOTS* VSSS 2020 - Desenvolvimento de robô omnidirecional voltado às competições de futebol de robôs na categoria *Very Small Size Soccer**

Gabriel R. Buiar[1], Álefe Felipe G. P. Dias[1], Matheus Diniz de Freitas[1], Misael R. da Costa[1], João A. Fabro[3] , André S. de Oliveira[4] e Jean M. Simão[5]

***Resumo*— Este artigo apresenta o processo de desenvolvimento dos sistemas que compõe o time de futebol de robôs UTBots, representante da Universidade Tecnológica Federal do Paraná(UTFPR) na categoria *Very Small Size Soccer* (VSSS). É apresentado o desenvolvimento mecânico e eletrônico dos robôs, impressos em impressora 3D, controlados por Arduino e com comunicação por módulos de rádio-frequência. Os robôs são omnidirecionais, com quatro servo-motores. O sistema de visão foi construído com as bibliotecas OpenCV e Qt. No sistema operacional *Linux*, o sistema de controle usa a biblioteca *boost interprocess* como meio de memória compartilhada para a comunicação entre módulos. O emergente Paradigma Orientado a Notificações (PON), e sua linguagem de programação LingPON (ambos desenvolvidos na UTFPR), são utilizados para programar a tomada de decisões do time.**

## I. INTRODUÇÃO

A Competição Latino Americana de Robótica (LARC–Latin American Robotics Competition) é um evento anual que reúne diversas competições, dentre elas competições de futebol de robôs. Uma das categorias das competições de futebol é a IEEE Very Small Size Soccer(IEEE-VSSS), categoria em que cada time é composto por 3 robôs pequenos, com tamanho máximo de 80mm de altura, 80mm de largura e 80mm de profundidade. Devido a situação pandêmica deste ano, a competição será realizada em um simulador, para que se possam ser respeitadas as medidas de segurança e, deste modo, evitar o contágio de participantes. O simulador em questão irá simular e tornar visível a partida que está acontecendo, além disso, ele também é responsável por guardar as variáveis do jogo, que serão passadas para o código dos times via um protocolo de comunicação chamado Protobuf, que foi desenvolvido pelo Google. E, posteriormente, o código da estratégia dos times irá retornar os comandos para os devidos jogadores. A equipe UTBots da UTFPR participa desta categoria de maneira efetiva desde 2017, com um robô inovador com movimentação omnidirecional. Em 2018, foi uma das três equipes a competir pela primeira vez na categoria 5 contra 5. O restante deste artigo apresenta as caracterısticas básicas dos robôs, de seus sistemas de aquisição e processamento, e de controle de movimentação. O time UTBots é vinculado a Universidade Tecnológica Federal do Paraná, mais especificamente ao LASER – Laboratório Avançado de Sistemas Embarcados e Robótica [1].

*Projeto suportado pela Universidade Tecnológica Federal do Panará.

[1]Graduandos de Bacharelado em Engenharia de Computação pela Universidade Tecnológica Federal do Paraná (UTFPR) - Campus Curitiba
alefe@alunos.utfpr.edu.br,
gabrielbuiar@gmail.com,
misael@alunos.utfpr.edu.br

[2]Graduando de Bacharelado em Engenharia Mecatrônica pela Universidade Tecnológica Federal do Paraná (UTFPR) - Campus Curitiba
modolojhon@hotmail.com

[3]Professor vinculado ao Departamento Acadêmico de Informática (DAINF) da UTFPR, pesquisador do PPGCA - Programa de Pós-Graduação em Computação Aplicada
fabro@utfpr.edu.br

[4]Professor vinculado ao Departamento Acadêmico de Eletrônica (DAELN) da UTFPR, pesquisador do CPGEI - Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial
andreoliveira@utfpr.edu.br

[5]Professor vinculado ao (DAINF) da UTFPR, pesquisador do CPGEI
jeansimao@utfpr.edu.br

## II. MATERIAIS E MÉTODOS

A categoria IEEE-VSSS envolve a disputa entre dois times em duas subcategorias: 3x3 e 5x5, com 3 e 5 robôs em cada equipe. A equipe participará de ambas as categorias como faz tradicionalmente. Este ano como a competição será realizada virtualmente, os jogos serão rodados em um servidor que receberá os códigos das equipes para executar a partida.

A simulação será realizada no simulador FIRA-SIM, e os robôs de todas as equipes serão os mesmos com duas rodas, como a equipe utilizava robôs omnidirecionais com quatro rodas, parte do código do controle de movimentação dos robôs teve de ser adaptado.

A categoria ainda vai abranger desafios para que os robôs realizem fisicamente com o objetivo de abranger as áreas de mecânica e eletrônica das equipes na competição. Como o acesso ao laboratório é restrito devido à pandemia, desenvolver esses códigos é um desafio para a equipe, para contorná-lo a equipe desenvolverá o código remotamente, e ira ao laboratório apenas para testá-los com os robôs e o campo.

### A. Estrutura do Robô

A estrutura do robô é o sistema responsável pela sustentação dos diversos componentes do robô, por proteger a integridade dos componentes eletrônicos em eventuais colisões durante a partida e pela condução da bola.

Durante as partidas do ano passado, precebeu-se que usar um uniforme para proteger apenas os módulos eletrônicos, deixando as rodas expostas, deixou o robô com o centro de massa muito elevado no eixo Z, deixando-o sem equilíbrio. Juntando o fato da roda construída pela equipe ter ficado bem aderente, quando um robô chegava em uma das laterais do campo e começava a ”escalar” a mesma, ou um robô adversário chocava-se com ele, o mesmo caía. Por isso este

ano foi implementado um uniforme que vai do topo até a base do robô, Com isso, o robô não vai mais tentar subir na parede, a bola será conduzida pelo uniforme ao invés da roda, e o centro de massa é baixado, proporcionando assim maior controle em relação a condução da bola e estabilidade ao robô.

Neste ano mudou-se o software de modelagem 3D para o SolidWorks [1], já que fornece mais ferramentas de trabalho, tornando mais fácil a modelagem de materiais complexos, como parafusos. Na Figura 1 pode ser vista a modelagem atual do robô. O LASER possui uma impressora 3D, onde as peças modeladas são impressas, tornando possível a realização de testes e correções. Na Figura 2 está o resultado correspondente do robô impresso e montado.

Devido ao fato de o robô usar 4 rodas, não há espaço para colocar pontes H na estrutura. Por isso os motores usados no robô são do tipo servo [2], modificados para permitir rotação contínua [3]. Infelizmente se perde a precisão do motor no processo de modificação, tornando o sistema de controle em malha aberta. Testes foram realizados com motores modificados de fábrica, mas o ganho de precisão aumentou apenas de cerca de 10%. Porém, é interessante o uso destes motores, devido ao ganho de tempo por não precisar modificá-los.

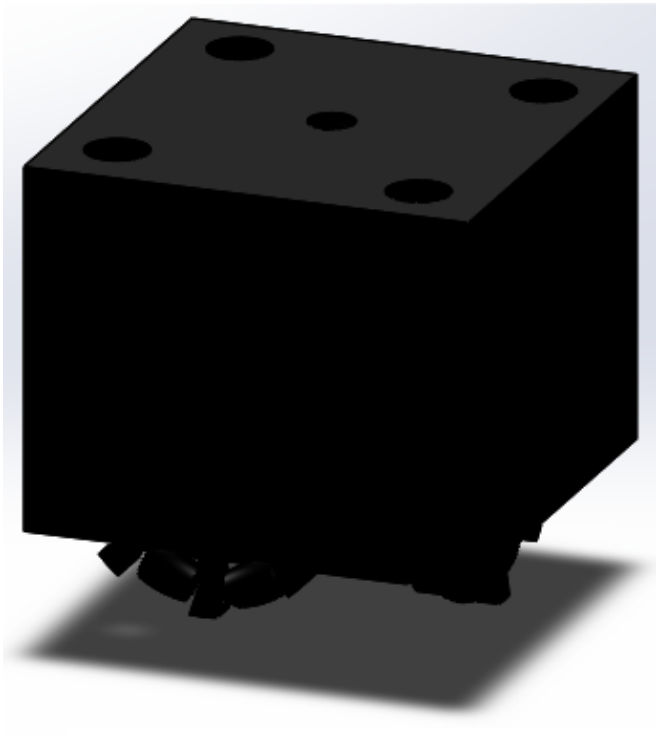

Fig. 1. Foto do modelo 3D do robô

### B. Circuito Elétrico e Placa Mãe

Para captação, interpretação de dados e controle dos motores está sendo usado um Arduino NANO [4], sendo que a comunicação é feita por meio do módulo de rádio

[1]https://www.solidworks.com

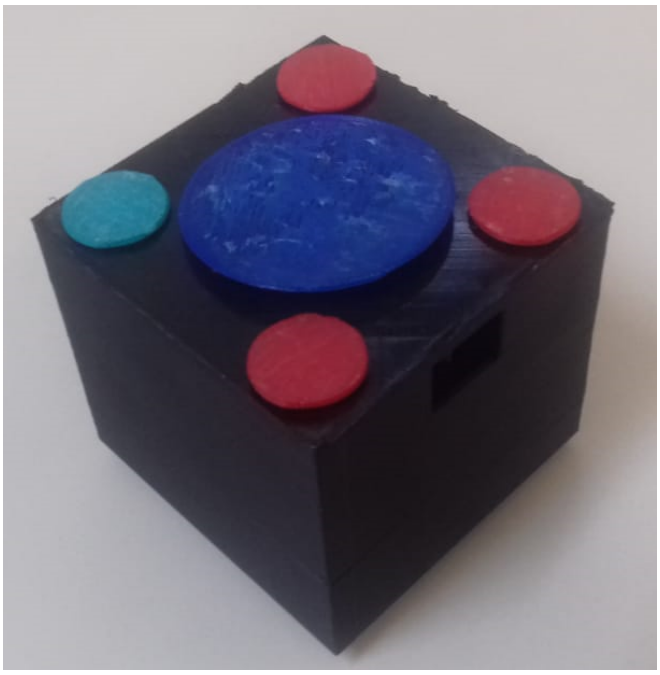

Fig. 2. Foto do robô

NRF24L01 [5]. Há também uma segunda configuração, usando Arduino FIO[6] e Xbee S1[7], mas o programa fonte é o mesmo para ambas, sendo necessário apenas mudar um define para a completa transição entre elas. Os servomotores possuem um sistema interno de controle, então basta enviar um sinal PWM (*Pulse Wide Modulation*, Modulação por Largura de Pulsos) para os mesmos. O circuito elétrico então serve apenas para alimentação dos motores e do Arduino, e também pela transmissão do sinal PWM aos motores. Por conta desse circuito ter uma alta complexidade em questão de fios, está em uso um circuito impresso para alimentação e comunicação entre o Arduino, o rádio e os motores. Na figura 3 pode ser vista a configuração para o rádio NRF24L01.

### C. Rodas Omnidirecionais

Robôs mais comuns com duas rodas podem mover-se em determinada direção apenas se o robô rotacionar para que este adquira a inclinação angular adequada para alinhar-se linearmente, ou seja, o robô movimenta-se apenas em direções paralela as suas rodas. Já robôs omnidirecionais são robôs aptos a deslocarem-se sem a obrigatória rotação, em seu próprio eixo, de direcionamento, podendo assim locomover-se em qualquer direção no plano horizontal com agilidade pois combina movimento linear e angular reduzindo assim o número de manobras necessárias para chegar-se a um determinado ponto de interesse [8]. Apenas a equipe UTBots utilizou este tipo de movimentação nas competições LARC em 2017 e 2018.

Para que esta movimentação omnidirecional, são necessárias rodas especiais. Tal dispositivo é denominado "Roda Omnidirecional", que é composta por uma roda principal que é responsável por fornecer torque para movimentação no plano normal ao eixo do motor. Existem

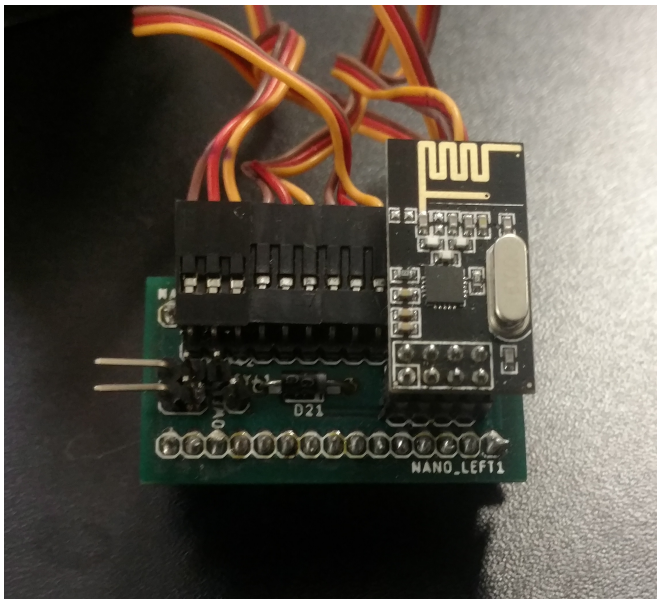


Fig. 3. Circuito impresso e NRF24L01

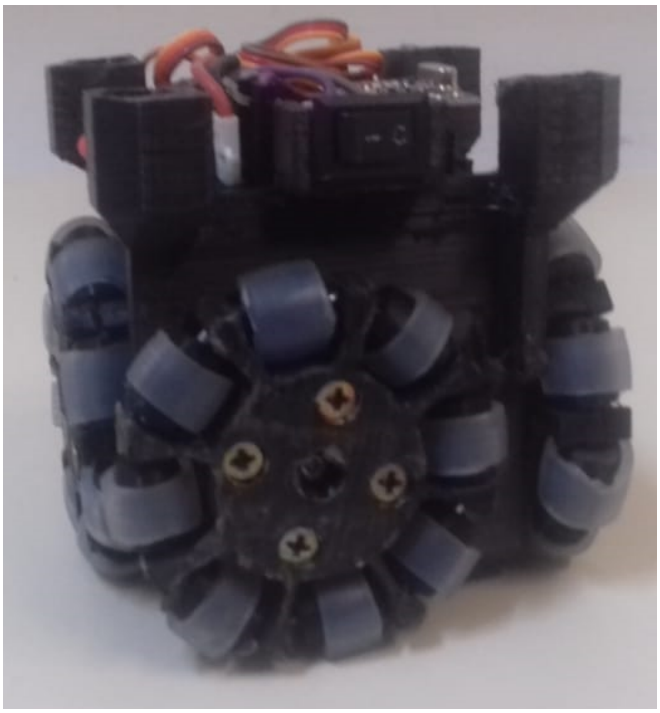

Fig. 4. Foto da roda desenvolvida pela equipe

ainda "rodízios" menores, localizados na periferia da roda principal, sendo que estas rodas secundárias possuem giro livre e eixo de rotação ortogonal em relação ao eixo de rotação da roda principal [9].

Assim como as outras partes estruturais do robô, as rodas foram modeladas pela equipe em 2018 e impressas. Porém, o modo como escolheu-se o eixo interno das rodas menores mostrou-se ineficiente durante a competição, fazendo com que o robô soltasse essas rodas durante os jogos. Com isso, este ano foram feitas modificações para aumentar sua resistência e integridade. Outra questão averiguada foi o atrito entre as rodas menores e a principal, tornando difícil a movimentação. Testes estão sendo feitos para descobrir o melhor formato e revestimento das rodas menores, de modo a ter o melhor de aderência com o chão e pouco atrito com a roda principal. Na figura 4 pode ser vista a roda usada pela equipe.

### *D. Baterias*

São utilizadas duas baterias de Lítio-Polímero (LiPo) capacidade de 850mAh de tensão elétrica de 3,7v, em série. O principal critério de escolha para estas foi o volume, que deveria ser o menor possível, e, dentro disto, foi escolhida então a que possui a maior quantidade de carga. Estas baterias possuem um tamanho de 3,5 x 5 centímetros, a profundidade de menos de 1 centímetro cada, e possuem uma autonomia de cerca de 1 hora com o robô em movimentação com velocidade máxima.

### *E. Sistema de Controle*

Sistema de controle é o nome genérico de todo o processamento feito no hardware auxiliar ao time de robôs até o ponto da transmissão de dados ao sistema embarcado dentro de cada robô pela comunicação sem fio. Esse processo é feito em 3 módulos: Sistema de Visão, Inteligência e Comunicação.

Devido ao fato de nas competições anteriores não se ter uma noção intuitiva do que realmente estava acontecendo durante a partida, a equipe desenvolveu um macrossistema em C++, com funções para o gerenciamento da memória compartilhada usando a biblioteca *boost interprocess* [10], definição do protocolo de uso da memória compartilhada para cada módulo e com uma interface gráfica para a visualização do comportamento dos módulos durante a partida, permitindo assim uma interação de forma intuitiva com os próprios. Com isso é possível saber quando os robôs não estão fazendo o que se pediu, ou se a inteligência está fazendo interpretações errôneas sobre os comandos que deve interpretar. Essa interface gráfica foi feita usando a biblioteca Qt[11], que também proporcionou algumas outras ferramentas úteis para o programa, como seu módulo de threads, o QThread.

Foi desenvolvido também um controle manual na interface principal. Essa função é útil para Debug, ou para testes rápidos, já que ela pode ser usada tanto com quanto sem reconhecimento de imagem, assumindo comportamentos diferentes em cada caso. Uma biblioteca foi desenvolvida para facilitar a integração entre os módulos, usando para configuração da mesma o software PkgConfig [12].

### *F. Sistema de Visão*

O módulo de visão envolve todo o processamento desde a captura da imagem da câmera até a detecção das posições dos robôs e bola no campo. Para a captação da imagem está

sendo usada a câmera Logitech C920 [2].

Cada robô possui obrigatoriamente um identificador, que é uma área em sua superfície com uma determinada cor, com uma cor atribuída ao time, assim pode-se identificar por essa cor seu time e o time adversário. Além disso, há outros identificadores para ser possível determinar o ângulo que o robô está deslocado em relação ao campo. Como não é possível saber de antemão qual é o padrão usado pela equipe adversária, esses outros identificadores são úteis apenas nos robôs aliados. Para identificar esse ângulo do robô a disposição geométrica dos identificadores e suas cores precisa ser tal que haja uma referência para se calculá-lo e ela não seja ambígua.

Em 2017, com o primeiro time efetivo na categoria, houve vários problemas com o controle dos robôs, devido ao fato de um eixo estar espelhado devido à representação da imagem no processamento do sistema de visão, o que não era possível corrigir somente mudando a disposição dos identificadores. Em 2018, isso foi resolvido invertendo a velocidade do eixo espelhado mandada para o robô.

O sistema de visão desenvolvido usa a biblioteca de visão computacional OpenCV [13]. Para identificar os objetos no campo foi utilizado o método de rastreamento de cores, que consiste em utilizar filtros para escolher quais *pixels* da imagem estão na região de uma certa cor, e depois usar um algoritmo do OpenCV que acha contornos em uma imagem binária, assim cada contorno representa um objeto com a cor filtrada. Depois de ter o conhecimento da localização de todos os objetos de cores na imagem, utiliza-se um algoritmo que decide o que é cada entidade no campo, isso é, a bola, robôs adversários e aliados, e ainda para os robôs aliados é calculado o ângulo como descrito anteriormente. Assim essas informações são escritas na memória compartilhada junto com o tempo da amostra para a inteligência poder fazer decisões sobre a movimentação dos robôs aliados. Esse sistema contém também uma interface gráfica desenvolvida com QT para calibragem das cores usadas durante o jogo.

Devido às limitações de processamento de matrizes em CPUs, estávamos com um atraso de cerca de 1 segundo entre a aquisão da imagem e o envio das informações para a inteligência, devido ao fato de estarmos com apenas 10 frames por segundo. Por isso decidiu-se fazer alguns testes GPU, utilizando para isso a biblioteca CUDA [14] e as funções para GPUs disponíveis no OpenCV. Porém não há funções de localização prontas que sejam rápidas como o algoritmo de achar contornos, além de não ter algumas outras funções que fazem várias filtragens na imagem, sendo necessário usar muito mais operações para conseguir o mesmo resultado. Com isso, mesmo com o uso da GPU, o resultado foi praticamente o mesmo. Um ganho maior pode ser obtido se a equipe conseguir desenvolver um algoritmo próprio de localização que seja tão rápido quanto a função usada atualmente (os testes feitos mostraram um ganho de 30% em relação ao uso da CPU, porém ainda muito abaixo do que esperávamos). Com isso, optou-se por diminuir a resolução de FULL HD para HD, obtendo com os algoritmos em CPU os 30 quadros que a câmera usada pode fornecer.

[2]Câmera C920 HD PRO WEBCAM, que possui resolução FULL HD e é capaz de fornecer 30 quadros por segundo. https://www.logitech.com/pt-br/product/hd-pro-webcam-c920

### G. Inteligência

A estratégia a ser seguida pelos robôs é definida pelos módulos de inteligência conforme as informações recebidas do sistema de visão, por meio da memória compartilhada. As estratégias consistem desde o básico e específico de um robô, como levar a bola em direção ao gol ou desviar da bola ao recuar, até estratégias mais elaboradas que podem envolver mais jogadores para realizar uma jogada ensaiada como um ataque ou defesa em conjunto.

As estratégias básicas são desenvolvidas com parâmetros simples como a posição do robô e a posição da bola, ao desenvolver estratégias mais complexas podem ser levados em consideração a posição da bola, a posição de cada jogador assim como a posição de cada oponente, e até o placar e o tempo restante da partida.

A equipe também está desenvolvendo um projeto utilizando o Paradigma Orientado a Notificações (PON), a LingPON, linguagem desenvolvida na UTFPR [15]. Utilizando o PON a quantidade de testes de comparação são reduzidos drasticamente - os testes só ocorrem quando uma regra é notificada da mudança de uma variável de interesse, ao invés de toda iteração – e o código se torna mais otimizado e menos acoplado, assim, sendo fácil adaptá-lo para mudanças de regras da competição e até utilizá-lo para outras modalidades.

Portanto, o objetivo da equipe é ter um número considerável de estratégias e um bom algoritmo para decidir qual a melhor estratégia a ser utilizada em cada momento da partida. Assim sendo, o desempenho pode ser melhorado em função dos seguintes parâmetros: quantidade de estratégias, qualidade das estratégias e qualidade do algoritmo que decide a estratégia.

Para testar a qualidade das estratégias, serão feitos treinos de ataque contra defesa, assim é possível verificar tanto as estratégias de ataque e defesa como também estratégias de contra-ataque e de recuo, que acontecem quando o time atacante vira defensor e vice-versa. Os treinos acontecem com todos os jogadores dos dois times sendo da mesma equipe, podendo constituir de jogos inteiros ou jogadas especificas como pênalti ou uma jogada ensaiada.

Um problema de treinar a equipe com treinos internos é de que não se tem um parâmetro para compará-las com outros times e saber se elas são boas, ou seja, todas as estratégias podem acabar sendo ruins. Logo, quanto melhores os treinos e as modificações das estratégias menores as chances de elas acabarem sendo ruins. Para reduzir ainda mais essa chance, a equipe fará amistosos com pelo menos mais uma equipe da mesma categoria, assim as estratégias ficam mais generalizadas e por consequência tem mais chances de serem melhores para qualquer outra estratégia ao invés de apenas as da mesma equipe.

### H. Comunicação

Os módulos de comunicação leem os comandos escritos na memória compartilhada pela inteligência para transmitir para os robôs via comunicação sem fio.

Para a comunicação, apenas o módulo nrf24l01 é usado, embora o sistema foi contruído de forma a poder se usar o Xbee S1 mudando uma flag no programa. Como o rádio usa comunicação SPI, um Arduino MEGA é usado como intermediário entre o módulo de comunicação e os robôs, lendo o protocolo serial e mandando para os robôs usando SPI.

Nos anos anteriores, notou-se que mesmo com todo o desenvolvimento do sistema de controle, ainda havia um *delay* além do gerado pela visão, fazendo com que os robôs executassem a instrução de maneira muito atrasada. Esse atraso foi resolvido esse ano, melhorando o protocolo de comunicação, o que proporcionou que cada robô receba cerca de 50 instruções por segundo se necessário.

### I. Estratégia

A estratégia da equipe é de possuir três posicões: o goleiro, o defensor ou suporte (dependendo da posição da bola) e o atacante; sendo sempre um jogador o goleiro e um ou dois jogadores (dependendo da subcategoria) nas outras posições.

Os objetivo do goleiro defender o gol de dentro da área, do defensor de defender o gol de fora da área, do suporte de ajudar o atacante no ataque de fora da área e do atacante de fazer o gol. Os robôs ainda trabalham de forma cooperativa, e mudam de posição se necessário, pricipalmente o goleiro e o atacente, o que torna a estratégia mais eficaz.

Como a execução da estratégia pelos robôs não é exata, são utilizadas várias margens de erro no código, mesmo com as margens o ambiente é estocástico, o que dificulta desenvolver e classificar estratégias para o projeto. Uma das soluções encontradas foi simular as estratégias em um simulador.

Para atender os novos padrões da competição, parte do código da estratégia teve que ser remodelado e reescrito, como o cálculo da movimentação dos robôs. Nesse processo de reformulação do código, as principais alterações foram no modo como as variáveis são inicializadas, assim, tornando o código mais otimizado e evitando a repetição, e no algoritmo que controla as velocidades das rodas dos jogadores.

Como todos os robôs serão padronizados para competição deste ano, o código de movimentação dos robôs foi adaptado para que os robôs rotacionem para a direção que precisam ir, o que não era necessário com os robôs omnidirecionais.

## III. Conclusão

No último ano, apesar da velocidade dos robôs ser de menor (cerca de dez vezes menos rotações quando comparado a maioria das equipes) devido aos servo-motores, a equipe conseguiu chegar às quartas de finais, um resultado excelente quando comparado ao histórico. O que explicita como a estratégia utilizada e o fato dos robôs serem omnidirecionais compensaram o motor menos rápido. Além disso, a parte da visão e comunicação se provaram sólidas, e bastante rápidas e eficazes na competição. Na parte da estrutura mecânica dos robôs houve uma melhoria, as peças não caíram nenhuma vez durante a competição e a capa externa não travou as rodas.

Para os motores está sendo desenvolvida uma pesquisa para analisar a possibilidade de utilizar quatro motores DC ao invés de servo-motores, devido à velocidade e maior controle, e como consequência pontes H.

## IV. Agradecimentos

Os autores agradecem à Universidade Tecnológica Federal do Paraná pelo apoio fornecido.

## References

[1] LASER, "Laboratório Avançado de Sistemas Embarcados e Robótica", disponível em http://laser.dainf.ct.utfpr.edu.br, Acesso em: 15/06/2019.

[2] Silveira, Cristiano Bertulucci. "Servo Motor: Veja como Funciona e Quais os Tipos", disponível em: https://www.citisystems.com.br/servo-motor/, acesso em 15/06/2019.

[3] PK2Lab, "Alteração de Servomotor", disponível em: http://pk2lab.blogspot.com.br/2011/12/alteracao-de-servomotor.html, acesso em: 15/06/2019.

[4] Arduino, "Arduino NANO", disponível em: https://store.arduino.cc/usa/arduino-nano, acesso em: 15/06/2019.

[5] Nordic Semiconductor, "nRF24L01 Single Chip 2.4GHz Transceiver Product Specification", disponível em https://www.sparkfun.com/datasheets/Components/nRF24L01_prelimprelim_prod_spec_1_2.pdf, acesso em: 15/06/2019.

[6] Arduino, "Arduino FIO", disponível em: https://store.arduino.cc/usa/arduino-fio, acesso em: 15/06/2019.

[7] Digi, "Xbee/Xbee – PRO S1 802.15.4 (Legacy) RF Modules", disponível em: https://www.digi.com/resources/documentation/digidocs/PDFs/90000982.pdf, acesso em: 15/06/2019.

[8] Pereira do Nascimento, T. "Controle de Trajetória de Robôs Móveis Omni-direcionais: uma abordagem multivariável" disponível em: http://www.ppgee.eng.ufba.br/teses/ac64916ca6ef3bdd2f466bffdd30af7e.pdf, acesso em: 13/06/2019.

[9] John S. Palmisano, "OMNI-WHEEL ROBOT – FUZZY", disponível em: https://www.societyofrobots.com/robot_omni_wheel.shtml, Acesso em: 15/06/2019.

[10] Boost C++ Libraries, "Boost", disponível em https://www.boost.org/, acesso em: 15/06/2019.

[11] Qt Libraries, "Qt", disponível em https://www.qt.io/, acesso em: 15/06/2019.

[12] Nicholson, Dan. "Guide to pkg-config", disponível em: https://people.freedesktop.org/ dbn/pkg-config-guide.html, acesso em: 15/06/2019.

[13] OpenCV Library, "OpenCV", disponível em https://opencv.org/, acesso em: 15/06/2019.

[14] NVIDIA. "CUDA Zone", disponível em: https://developer.nvidia.com/cuda-zone, acesso em: 15/06/2019.

[15] Simão, Jean. "Paradigma Orientado a Notificações (PON)", disponível em: http://www.dainf.ct.utfpr.edu.br/ jeansimao/PON/PON.htm, acesso em: 15/06/2019.

[16] ROS. "ROS", disponível em: https://www.ros.org/, acesso em: 15/06/2019.