TDP - Equipe Red Dragons UFSCar 5v5

Allan Mendonça Cavalcante, André Camilo Terra Costa, Bruno Yoshihara Taira, Carlos Eduardo de Souza, Carlos Eduardo Souza Silva, Daniel Pagliaricci, Davi Batista Silveira, David Amorim Silva, Eduardo Alberto Volpe, Fausto José Francisco Junior, Gabriel Mari Tararam, Gabriel Souza Barbosa, Giovanna Amorim Nascimento, Gregório Sant'Anna,Gustavo Yudi Sagara Nicio, Ítalo Gabriel Bezerra, Jackeline Tomioka, João Carlos Tonon Campi, Julio Raicher Ogando, Kleber Almendro Rodrigues,Lucas Campos de Oliveira Faria, Lucas Henrique Marchiori, Lucas Messias Moura, Lucas Polzella Lourenço Gil, Marcelo Marinoni Carneiro , Mateus José Prado Pereira, Matheus Paladini Pedrinho, Pedro Almeida Gorgatti, Pedro Cassiano Coleone, Pedro Lealdini do Prado Borges, Rafael Vinícius Silvério Rodrigues, Rodrigo Falcão Freddi, Samanta Eunice da Silva, Sérgio Henrique Flório, Tiago Felix Carrare, Tiago Lopes Rezende, Vilson Bueno da Silva Junior, Vinícius Fávero Mioto da Silva

I. Introdução

A equipe Red Dragons buscou desenvolver inovações e estabilizações em hardware para o projeto do ano de 2022. Foram feitos em 2020 e 2021 importantes aprimoramentos no projeto mecânico na equipe, sendo validados em ambiente real no ano de 2022 após o período de distanciamento social. Se destacam neles os testes da implementação de encoder magnético no eixo estendido traseiro do motor e testes da implementação de sistema de engrenagens para acomodar o encoder. Já em relação aos circuitos eletrônicos do robô, buscando um projeto mais simples e estruturação da equipe, um novo projeto de placa de circuito impresso está sendo desenvolvido em base na placa de desenvolvimento ESP32. Enquanto isso, o software de controle e estratégia está se preparando para a transição dos resultados obtidos simulados para aplicação nas competições.

II. Projeto Mecânico

Durante o ano de 2022, a equipe buscou a prototipagem e validação no projeto físico das inovações e projetos desenvolvidos durante os anos de 2020 e 2021, uma vez que esse ano teremos competições presenciais, em que teremos o robô físico. Além disso, buscamos também pensar em novas ideias de inovações e implementações para o projeto com o objetivo de otimiza-lo, dentre essas novas ideias, as quais ainda estão em fase de desenvolvimento e testes, podemos citar a alteração do modelo do *ballcaster*, e a fabricação de novos pneus e rodas, alterando o material anteriormente utilizado.

A. Componentes Gerais Atuais do Robô

1) Motor: Atualmente utilizamos motores da marca Pololu de 6V com redução de 100:1, e é utilizado um *encoder* magnético da mesma fabricante. Na Figura 1, pode-se observar o motor com sua redução, além do *encoder* e seu disco magnético na parte mais a direita da imagem.

2) Rodas: A roda atual utilizada no projeto é feita de alumínio 6351-T6, e usinada na UFSCar. Os pneus são anéis de vedação convencionais de tamanho adequado. Ela pode ser vista na Figura 2

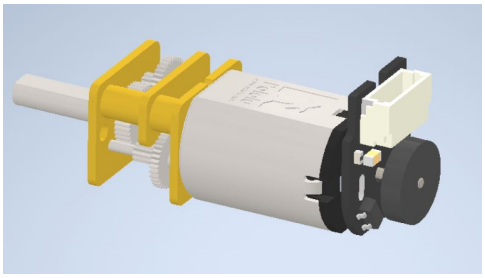

Fig. 1: Motor com Encoder

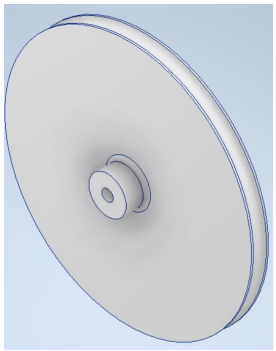

Fig. 2: Roda utilizada no projeto

3) Mancal: O mancal utilizado no nosso projeto tem a função de fixar o motor, sustentar as engrenagens e acomodar os rolamentos. Para tal, precisamos dividi-lo em 3 partes, sendo 2 peças mancais de rolamento para os eixos das rodas, representado em cor vermelha e, sendo 1 peça mancal de fixação do motor representado em cor cinza, e ilustradas na Figura 3.

4) Engrenagens: Os pares de engrenagens utilizados atualmente são feitas de aço, e são do modelo 22T da marca INJORA. O sistema pode ser visto na Figura 4

5) Capa: A capa atual utilizada no nosso projeto é dividida em 4 partes: parte de cima da tampa, responsável por alocar os marcadores da visão, parte de baixo da tampa, responsável pelo encaixe e fixação da tampa com o restante do corpo do robô, parte superior e inferior do corpo da capa, responsáveis pela proteção das peças internas do robô contra

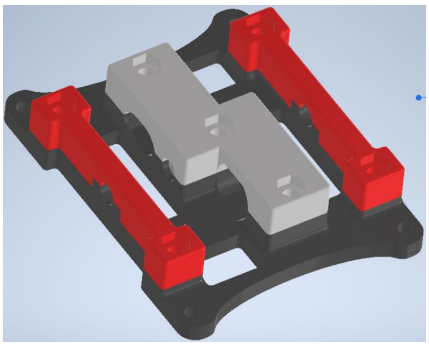

Fig. 3: Estrutura do mancal

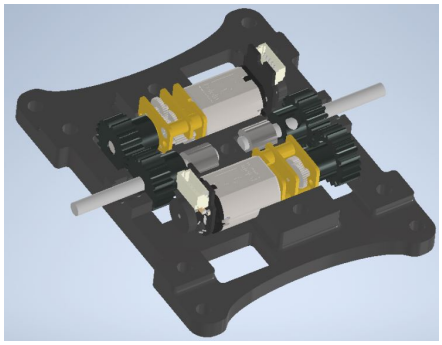

Fig. 4: Sistema de engrenagens

possíveis atritos durante uma partida.

FIGURA DA CAPA SOLID

6) Piso dos motores: O piso dos motores foi confeccionado em PLA, por impressão 3D, e é responsável pela acomodação dos motores e encaixe dos mancais. O piso do motor projetado é visto na Figura 5.

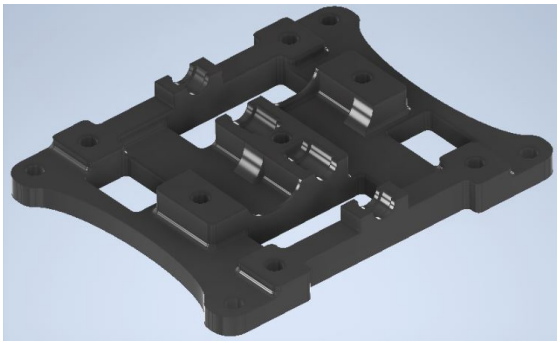

Fig. 5: Piso dos motores

7) Hastes: Atualmente no projeto utilizamos as hastes para unir os componentes do robô, como o piso dos motores, suportes frontais, placa de aço e circuito eletrônico, para isso são utilizados três espaçadores de nylon, dois parafusos sem cabeça, e um parafuso de cabeça chata, além de funcionarem como colunas estruturais, segurando os componentes. As hastes podem ser vistas no centro da Figura 6

8) Suportes frontais: Os suportes frontais utilizados atualmente no nosso projeto são feitos de PLA por impressão 3D,e têm a função de separar o piso dos motores da placa

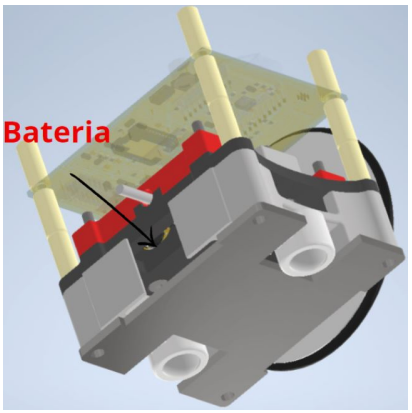

Fig. 6: Robô montado com hastes, suporte frontal e placa de aço

de aço, de modo a se ter espaço para o encaixe da bateria entre essas peças, além de ser a peça onde irá se encaixar o *ballcaster*. O suporte frontal é a parte inferior branca presente na Figura 6.

9) Placa de aço: Na base do robô foi colocada uma chapa de 3mm de aço para abaixar o centro de massa, dar rigidez estrutural e garantir estabilidade. A placa de aço está presente na parte mais abaixo do robô, visto em cinza na Figura 6.

B. Novos Projetos

Após um período de trabalho apenas na modalidade a distância, a equipe aproveitou para desenvolver projetos visando a melhoria de desempenho e montagem dos robôs. Os principais projetos estão apresentados a seguir:

1) Roda e Pneu: A nova roda em desenvolvimento, presente na Figura 7, é de PLA e é feita em impressora 3D, o que traria muito mais velocidade e facilidade para a fabricação da mesma em relação a usinagem.

Fig. 7: Roda nova confeccionada em impressão 3d

2) Ballcaster: O projeto anterior apresentava dificuldades em sua montagem, na qual o *ballcaster* tem parafusos de fixação com suas cabeças opostas à esfera maior e não são transpassantes. Então, desenvolveu-se uma proposta de solução com um *ballcaster* de mercado que tivesse sua fixação feita por parafusos transpassantes e suas cabeças no mesmo lado da esfera maior. Desta forma, seria necessário uma pequena alteração na placa de aço da base do robô

para poder dar espaço para ferramentas de aperto do parafuso. Além disso, seria necessário alteração no suporte do *ballcaster* para acomodar uma porca que funcionaria para o aperto do parafuso à estrutura.

3) Capa: A capa atual sofreu duas principais alterações, sendo uma delas em seu número de divisões e outra em sua forma de fixação. A capa agora apresenta 4 divisões de tal forma que a parte inferior foi dividida em duas, conforme a indicação da seta. O principal objetivo é facilitar a montagem e melhorar a qualidade da impressão 3D. A outra mudança é a forma de fixação que antes era somente por atrito e agora tem presente dois parafusos de cabeça chata na parte superior da capa que transpassam dois furos e se fixam nas hastes de nylon internas à capa. Isso pode ser observado na Figura 8 abaixo.

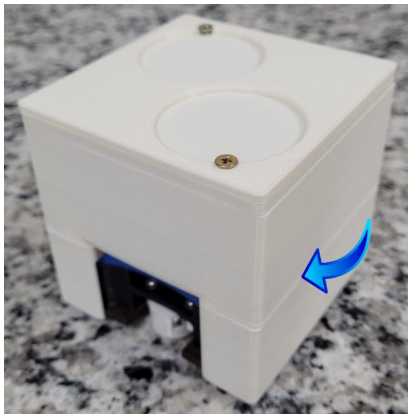

Fig. 8: Capa confeccionada em impressão 3d

4) Hastes de Latão: As atuais hastes do robô, por serem de nylon, tem uma vida útil reduzida de suas roscas após serem muito utilizadas nos processos de montagem e desmontagem. A solução proposta é de utilizar hastes de latão de exatas mesmas dimensões, uma vez que o latão apresenta maior dureza e durabilidade para desgaste com montagem e desmontagens, reduzindo as chances de as roscas falharem, principalmente em um momento de grande estresse mecânico como em competição. No momento, a equipe optou por não utilizar as hastes de latão por ainda ser necessário um estudo mais aprofundado se não ocorreria um aumento da altura do centro gravitacional que poderia piorar o desempenho dinâmico do robô.

III. Eletrônica

A. A Placa

Devido a problemas como impedimento do trabalho presencial e saída de membros experientes, surgiu a necessidade de estruturar um projeto mais básico e sólido antes de finalizar o projeto proposto em 2021, envolvendo o microcontrolador STM32F103 e componentes SMD. Foi retomado o estudo de fabricação de PCBs através de softwares de criação de layout, como Eagle e Altium, e o uso de componentes PTH.

B. Microcontrolador

No ano de 2022 optou-se por utilizar o microcontrolador ESP32, representado na Figura 9, que apresenta algumas vantagens quando comparado a outros microcontroladores. Ele realiza todas as funções necessárias para o projeto, como o controle *PWM* e quantidade suficiente de *timers* e de portas GPIO, além de apresentar uma facilidade de comunicação por conter módulos *bluetooth* e *wifi* integrados à placa. Ele também possui um custo relativamente baixo e tem as dimensões facilitam a sua implementação devido as limitações de espaço na categoria. Outra vantagem desse componente é a versatilidade de plataforma, sendo possível configurar diversas plataformas para sua programação, incluindo a conhecida Arduino IDE.

O chip ESP32 opera sob uma alimentação de 2.7V - 3.6V e 500mA, assim necessitando da implementação de um regulador de tensão, além do regulador interno, para se adaptar à bateria utilizada. Ele possui 30 portas ao total, incluindo portas GPIO (*Input/ Output*), *PWM*, *RX* e *TX*, entre outras, assim permitindo a fácil alimentação e transmissão de sinais de tensão. Nele, ainda encontramos 16 portas ADC (Analog Digital Converter), mais de um *clock* e cristal oscilador internos.

Ainda assim foram mantidas as pesquisas e testes relacionados ao microcontrolador STM32, sendo um objetivo implementá-lo ao projeto assim que possível.

Fig. 9: ESP32

C. BMS

O uso ou recarga inadequada da bateria, pode fazer com que sua vida útil seja diminuta, podendo trazer prejuízos para o robô devido à suas constantes necessidades de recarga. Para contornar tal situação, pesquisas foram feitas visando a viabilidade de carregadores balanceados e placas que controlam a recarga das células da bateria, essas placas são conhecidas como BMS, e se mostraram uma opção satisfatória. Portanto, implementamos um sistema utilizado nesses circuitos em nossa própria placa para realizar a leitura da bateria, fazendo com que o seu carregamento seja controlado, garantindo assim, uma maior longevidade à mesma.

D. UART e USART

Como parte dos estudos para o avanço da Eletrônica da equipe, foi feito uma pesquisa sobre comunicação serial UART e USART, com o objetivo de avaliar seu funcionamento, vantagens e aplicações. A porta UART (Universal Asynchrounous Receiver/Transmiter), assíncrona, tem como principal característica a não utilização de um *clock*, mas um *start bit* e um *stop bit*. Sendo utilizado em comunicações a curta distância, a UART apresenta um baixo consumo de energia e a possibilidade de *parity bit* para verificar erros, porém, limita-se a uma comunicação entre dois dispositivos, apenas. A porta USART (Universal Synchronous Asynchrounous Receiver/Transmiter) tem uma comunicação que pode utilizar ambos os modos assíncrono, como a porta UART, ou síncrono, que utiliza um *clock*. Tal porta possui como principal vantagem uma maior velocidade de transmissão mas é limitada a dois dispositivos e apresenta um maior gasto de energia. A utilização da comunicação UART se tornou a mais eficaz para o nosso uso, já que as informações não são enviadas em um período fixo de tempo.

IV. Visão Computacional

A. Algorítimo

Tendo em vista que nosso algoritmo já apresentava resultados satisfatórios em performance e acurácia, nestes anos buscamos meios de tratar os erros que ocorrem em tempo de execução, como a perda do robô sobre a presença de algum objeto com cor semelhante, luminosidade ruim e distorções que podem atrapalhar a identificação do robô. Para isso, precisamos identificar, dentro do nosso código fonte, indicativos de que a iteração gerou resultados incoerentes como número total de marcadores principais insuficiente, não identificação de centroides para algum marcador secundário ou distância muito grande entre marcador principal e marcador secundário. A partir dessas identificações podemos implementar estratégias para contornar possíveis falhas durante a execução, como manter a posição anterior do robô ou fazer uma projeção baseada nas últimas posições, sendo possível enviar informações aproximadas para o controle quando ocorrer falhas na visão.

B. Calibração

Nestes anos, para reduzir o tempo de calibração foram feitas mudanças em três das etapas existentes. As etapas de calibração da perspectiva e de corte foram unidas em uma só, pensada de modo que o usuário apenas precise marcar as extremidades do campo sem precisar especificar quais são os pontos superiores, inferiores etc. Essa mudança foi possível graças a percepção de que, a partir das coordenadas das extremidades podemos obter os pontos necessários para executar a transformação de perspectiva e para o corte dos cantos do vídeo, bastando apenas traçar as retas que cruzam os pontos fornecidos pelo usuário e identificar os pontos de intersecção entre elas. A Figura 10 ilustra bem como é possível obter as coordenadas faltantes (em vermelho) a partir das extremidades (em azul).

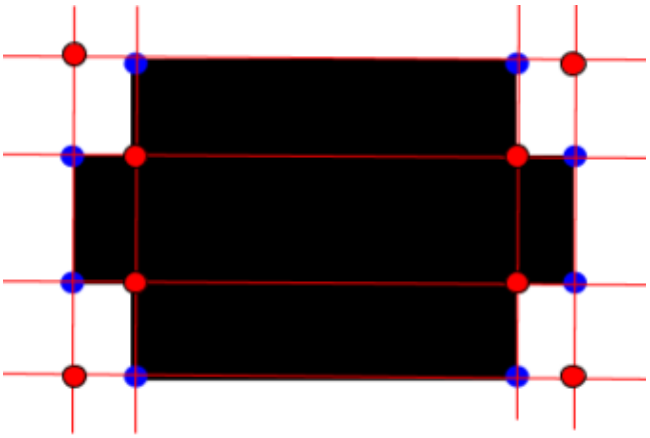

Fig. 10: Obtenção dos pontos de corte

Por ser executado apenas uma vez durante a calibração ou em caso de alteração nas entradas, o cálculo das coordenadas não afeta o desempenho geral do processamento em jogo, se mostrando uma alternativa simples e eficaz para facilitar o processo de calibração.

Além disso, a etapa de identificação dos marcadores também foi modificada pra que o usuário não precise especificar a qual posição corresponde cada marcador. Para isso, a entrada é iniciada com três pontos distribuídos nas extremidades superior esquerda, superior direita e inferior esquerda. A partir daí, a cada clique do usuário o ponto mais próximo é deslocado para a posição indicada.

C. Interface Gráfica

Ano passado foram implementadas diversas melhorias na interface gráfica com objetivo de torna sua utilização mais intuitiva. A fim de tornar o processo de calibração mais intuitivo, a interface foi reescrita de forma que cada etapa da calibração é mostrada sequencialmente para o usuário, forçando-o a seguir a ordem correta e dispensando a necessidade de que o mesmo precise ter conhecimento sobre a sequência de etapas. Nos anos anteriores era necessário escolher qual etapa você gostaria de configurar, sendo muito difícil para pessoas que apresentam pouca experiência com as configuração do algoritmo da visão (a possível entrada de um arquivo .red com uma calibração já pronta também ajuda bastante nesse aspecto).

D. Pré-calibração

A fim de reduzir o tempo necessário para calibração e melhorar sua qualidade, começou a ser desenvolvida, no período do final do ano passado, a implementação do agrupamento k-means na etapa de pré calibração, algoritmo o qual foi dividido em três etapas. Na primeira etapa, o fundo da calibração é removido, de forma a eliminar amostras contaminadas em seu cálculo. Logo em seguida, na segunda etapa, são gerados agrupamentos de cores dentro da imagem, os quais são chamados de clusters, que classificam os pixels da imagem. E por fim, na última etapa, são selecionados os clusters que têm um centro mais próximo das cores de referências dos marcadores.

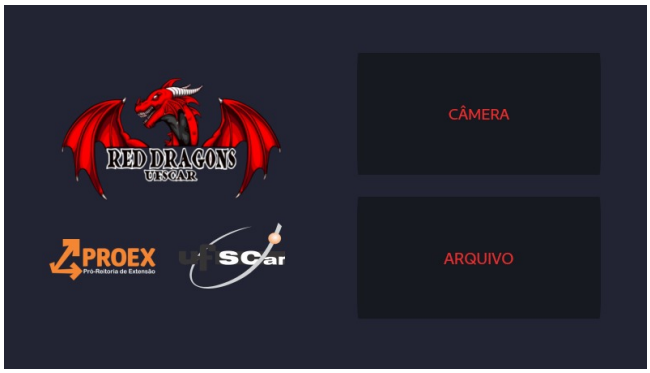

Fig. 11: Nova tela inicial, no qual é selecionada a entrada de vídeo.

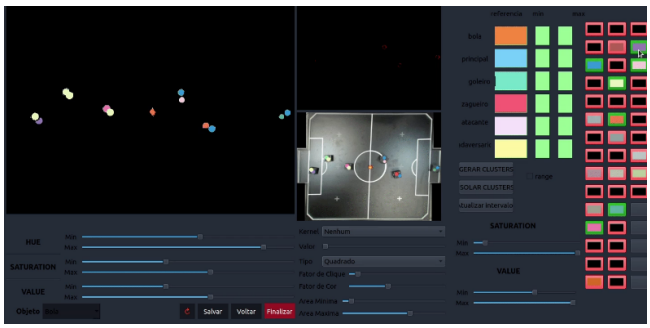

Fig. 12: Interface do algoritmo de agrupamento k-means.

Porém, algumas questões primordiais, para a implementação final desse algoritmo no código, não foram resolvidas. Uma questão se dá no momento de geração dos clusters, onde alguns não são considerados tão próximos das referências e são descartados. E a outra questão é a integração dos clusters selecionados para os parâmetros da calibração em jogo, o que pode ser contornado pela seleção manual dos agrupamentos, mas não é ideal.

V. Controle e Estratégia

A. Simulador

Nesses anos em que ficamos distantes das atividades em ambientes físicos, buscamos uma alternativa para testar novos códigos durante esse período. Uma solução foi a implementação do simulador CoppeliaSim na sua versão EDU. Este simulador, além de possibilitar uma construção do robô bem fiel com os parâmetros mecânicos, também possibilita uma comunicação via API com a linguagem Python, que é a linguagem utilizada pela equipe atualmente. Na Figura 13 a seguir é mostrado o ambiente de simulação de um campo VSSS.

B. Navegação

Houve também a implementação de um algoritmo de navegação, o Univector Field. Resumidamente, esse algoritmo dinâmico determina, em relação a posição do robô no campo, a direção desejada para atingir um alvo com um ângulo de chegada desejado.

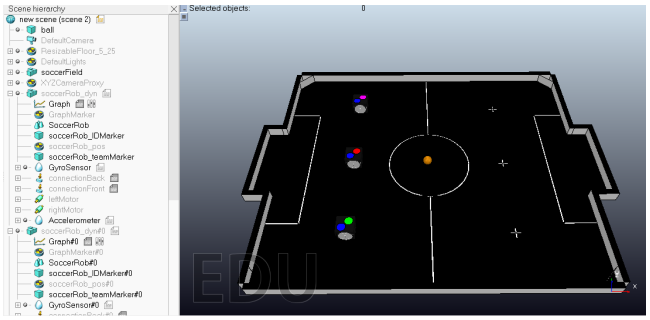

Fig. 13: Simulador CoppeliaSim

A implementação foi baseada no trabalho de [1]. O artigo trás a elaboração de dois tipos de campo: De locomoção até um alvo com um determinado angulo de chegada e o de desvio de obstáculos.

O primeiro é formado pela união de duas espirais hiperbólicas, nas quais são compostas em sentido horário e anti-horário, com o objetivo de levar o robô a uma postura (x, y, θ) desejada. Um campo de espiral hiperbólica gerado pela implementação pode ser visto na Figura 14. O segundo é formado por um simples campo radial com origem no centro do obstáculo. Por fim, esses dois campos são unificados utilizando uma função gaussiana, na qual pesa a distância em que o robô está em relação ao obstáculo.

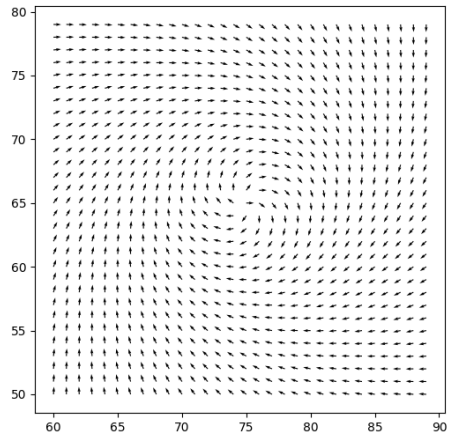

Fig. 14: Espiral hiperbólica no sentido horário

Entretanto, o resultado do campo mostrado é apenas para um angulo de chegada de $0°$. Para englobar todos os ângulos possíveis, precisamos transladar as posições dos robôs usando a posição alvo como origem e rotacionar esse sistema de coordenadas no angulo desejado. As equações

$$\begin{cases} x' = x_r - x_t \\ y' = y_r - y_t \end{cases} \tag{1}$$

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} \tag{2}$$

foram usadas para essa operação, sendo (x_r, y_r) a posição do robô, (x_t, y_t) a posição do alvo, (x', y') a posição transladada e (x_c, y_c) a posição rotacionada.

A Figura 15 mostra o resultado da implementação desse campo, posicionando a postura alvo como $(95, 80, \pi/4)$ e um obstáculo em $(30, 30)$. Nela, é possível perceber a mistura do campo ao redor do obstáculo com o campo de locomoção, além das duas espirais hiperbólicas utilizadas.

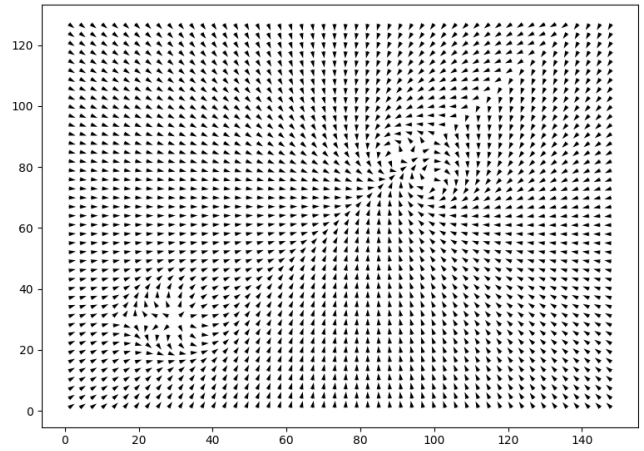

Fig. 15: Campo resultante com obstáculo

C. Controle

A partir do livro [2], foi implementado um controlador cinemático indicado para se trabalhar com o Univector Field olhando diretamente para o erro do ângulo desejado. Mais detalhes sobre esse controlador podem ser encontrados no Capítulo 4.6.1.

D. Estratégia

Para a estratégia do 5x5, adotamos uma formação similar ao futsal, um goleiro, um defensor, dois alas e um pivô. Assim como no 3x3, o goleiro é o jogador mais recuado em nosso time, permanecendo sempre em sua área e também o defensor ou zagueiro, ainda que, fique mais avançado em relação ao goleiro, permanece, principalmente, na área defensiva de nosso time. Os alas, por sua vez, ficam nas regiões laterais do campo, um robô na direita e outro na esquerda, buscando dar suporte tanto na defesa quanto no ataque pelas extremidades. Por fim, o pivô faz o papel de atacante, sendo nosso principal jogador a realizar as jogadas ofensivas para marcação dos gols.

Outra tática que a equipe está empregando é a Strategy Grid, na qual estamos estudando como mapear e realizar a tomada de decisões de uma grid para outra. Nela, o campo seria dividido em 30 partes iguais, como pode ser visto na Figura 16, com o intuito de adaptar o processo decisório das estratégias em tempo real durante o jogo dependendo da localização dos robôs.

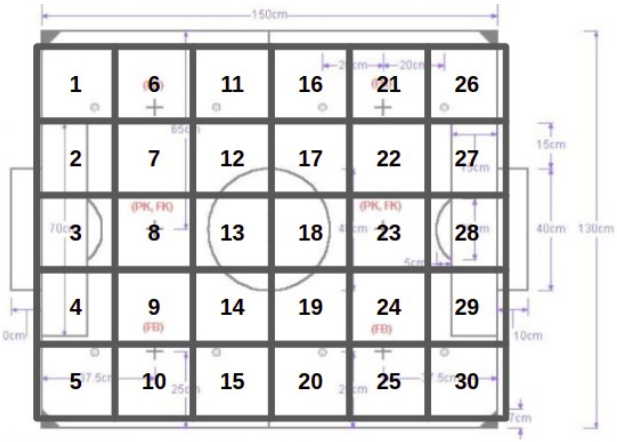

Fig. 16: Campo dividido em grids

Referências

[1] 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.

[2] Jong-Hwan Kim, Dong-Han Kim, Yong-Jae Kim, and Kiam-Tian Seow. Soccer Robotics, volume 11 of Springer Tracts in Advanced Robotics.Springer, 2004.