Robotech-UFPA IEEE Very Small Size Soccer (VSSS) Team Description

Ailton Oliveira[1], Yuri Silva[1], Lucas Damasceno[2], Virgínia Brioso[1], Carnot Braun[2], Jhonys Moura[1] e Adalbery Castro[2]

***Abstract*— Este trabalho descreve o desenvolvimento da equipe de robótica da Universidade Federal do Pará, Robotech-UFPA, para a categoria IEEE Very Small Size Soccer. Neste artigo serão apresentadas as técnicas e tecnologias empregadas pelo time, para a solução dos problemas apresentados, de modo que os robôs desenvolvidos tenham um desempenho eficiente, e que os temas abordados possam contribuir com a evolução de outros times.**

***Index Terms*— VSSS, Robótica, Visão Computacional, Controle NIT.**

I. INTRODUÇÃO

A Robotech-UFPA surgiu a partir de uma iniciativa de alunos da graduação em Engenharia da Universidade Federal do Pará, no ano de 2019, com o intuito de promover a integração dos participantes com a robótica através do estudo, pesquisa e desenvolvimento de soluções e tecnologias nessa área. A equipe também busca participar de competições para validar seus projetos dentro da robótica. Nas competições, há diversas categorias, como a IEEE VSSS (Very Small Size Soccer), cujo esse artigo tem o enfoque principal.

A categoria IEEE VSSS (Very Small Size Soccer) propõe a construção de um time de futebol de robôs. Cada equipe é formada por três robôs autônomos de tamanho limitado (7.5 x 7.5 x 7.5cm), controlados por um computador externo, utilizando uma câmera de vídeo para aquisição de dados visuais do campo e enviando comandos para os robôs através de uma interface de comunicação sem fio [1], como mostrado na Fig. 1.

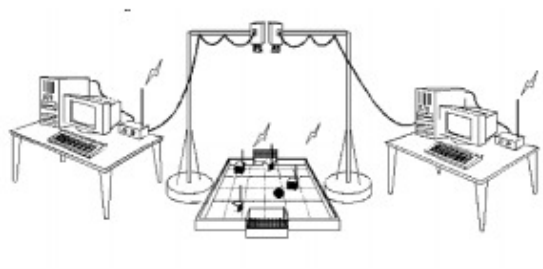

Fig. 1. Ilustração de uma partida IEEE VSSS.

[1] A. Oliveira, J. Moura, V. Brioso e Y. Silva estão com FEEB - Faculdade de Engenharia Elétrica e Biomédica, Belém-PA, Brasil (e-mails: [ailton.pinto, jhonys.moura, virginia.tavares, yuri.souza] @itec.ufpa.br).

[2] A. Castro, C. Braun e L. Damasceno estão com FCT - Faculdade de Computação e Telecomunicações, Belém-PA, Brasil (e-mails: adalbery@ufpa.br, [carnot.filho, lucas.damasceno.silva] @itec.ufpa.br).

O diferencial do VSSS é que os jogadores do time (robôs) devem possuir independência, ou seja, através das imagens enviadas pela câmera, localizada suspensa sobre o campo, o computador do time deve decidir quais as ações os jogadores do seu time devem tomar. Essa decisão é realizada através de um algoritmo previamente desenvolvido capaz de interpretar dados de cada robô da partida e estabelecer uma estratégia de jogo para cada robô do time.

O restante deste trabalho está organizado da seguinte forma: A Seção II descreve a estratégia de tomada de decisão utilizada pela equipe. Na Seção III é tratada a Visão Computacional, responsável pela localização e identificação dos robôs, além do tratamento das imagens capturadas pela câmera. A Seção IV descreve o método de movimentação e trajetória do robô em campo. A Seção V descreve um sistema de controle baseado no controlador não-linear invariante no tempo (NIT). Na Seção VI são descritas as características dos componentes e circuitos utilizados no sistema eletrônico dos robôs. A Seção VII apresenta o simulador utilizado pela categoria para a competição deste ano. Por fim, a Seção VIII encerra este trabalho.

II. ESTRATÉGIA

A estratégia utilizada pela equipe para a tomada de decisão, de cada robô, é realizada através do método Árvore de Decisão. As Árvores de Decisão são uma forma muito eficiente de criar sistemas complexos que são modulares e reativos. Essas propriedades são cruciais em muitas aplicações, o que levou à disseminação desse método da programação de jogos de computador para muitos ramos da Inteligência Artificial e da Robótica [2]. De um modo geral em computação, as Árvores de Decisão são estruturas de dados formadas por nós, que podem ser do tipo: Condicional, Seletor, Sequencial, ou de Ação. Toda Árvore possui um nó principal, ele possui o maior nível hierárquico (é o ponto de partida) e ligações para outros elementos, denominados nós filhos. O nó condicional verifica sempre que uma condição é satisfeita ou não, retornando com ”sucesso” ou ”fracasso” de acordo com a situação. O nó condicional nunca retorna ”em execução”. O nó seletor chama cada nó filho em sua lista até que um desses nós retorne ”sucesso”, então ele mesmo retorna como sucesso. Se nenhum de seus sub-nós chamar ”sucesso”, o seletor retornará ”fracasso”. Um nó sequencial irá chamar cada um de seus sub-nós, um após o outro, até que um nó falhe (retorne ”fracasso”) ou todos os nós sejam chamados. Se uma chamada de nó falhar, o nó sequencial retornará ”fracasso”, caso contrário, chamará ”sucesso”. Um nó de ação executa uma dada atividade, retornando ”sucesso”

se a ação for concluída [3]. Essas propriedades do método trazem segurança, robustez e eficiência para a tomada de decisão dos robôs jogadores, tornando os mesmos sistemas autônomos com design eficientes e fáceis de usar. A tomada de decisão dos robôs da categoria IEEE VSSS são baseadas nas coordenadas de todos os robôs da partida, a partir de cada coordenada é possível analisar individualmente como cada robô está alocado dentro da partida de uma maneira que seja possível rapidamente classifica-lo dentro de condições dos nós da Árvore de Decisão, e assim então poder realizar as ações mais adequadas para cada robô jogador do time. Na Fig. 2 é possível ver a estratégia criada pela equipe.

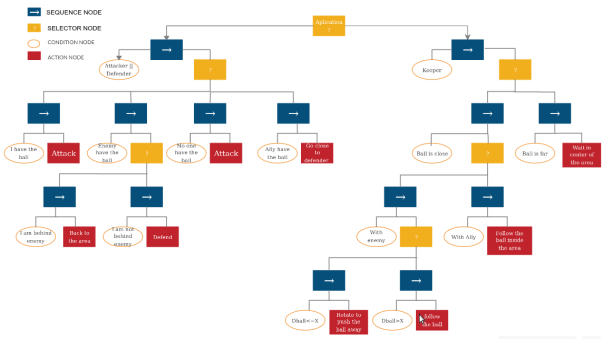

Fig. 2. Estratégia aplicada pelo método Árvore de Decisão.

III. VISÃO COMPUTACIONAL E PROCESSAMENTO DE IMAGEM

Neste tópico serão abordardos as informações obtidas por uma câmera centralizada acima do campo, de modo que os robôs sejam capazes de identificar e localizar os elementos no campo, permitindo assim o planejamento da trajetoria a partir do tratamento do vídeo, que é feito através de técnicas de processamento digital de imagem.

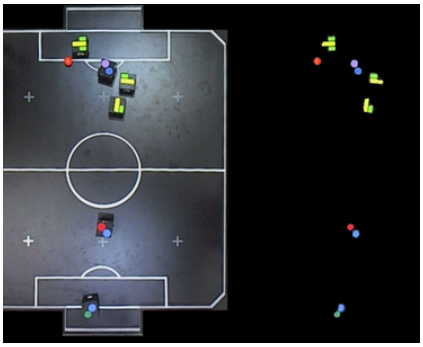

Fig. 3. Tratamento da imagem capturada pela câmera

A. Conversão RGB para HSV

A identificação dos objetos em campo não é uma tarefa simples quando utilizamos o sistema de cores RGB, principalmente pelo modo que a representação de uma cor é definida. Para facilitar esse processo, é utilizado o sistema de cores HSV, que representa a imagem pelos componentes matiz (Hue), saturação (Saturation) e valor (Value), possibilitando a filtragem pela matriz de cor (Thresholding) requerer uma calibração menos burocrática.

B. Filtragem de ruídos

São definidos intervalos de cores de acordo com as camisas dos jogadores. Após isso, é realizada a filtragem por thresholding, a qual executa uma varredura em todos os pixels da imagem, verificando quais estão dentro dos limites dos intervalos, sendo esse setados como branco e os demais, não pertencentes a esse domínio, como preto. Essa filtragem gera uma classificação binária, mas que pode ser separada por classes. Após essa classificação, são realizadas transformações morfológicas de abertura e dilatação, de modo a retirar pixels ruidosos que possam ter passados pela filtragem anterior.

C. Identificação

Para realizar a diferenciação dos times, e dos robôs que os integram, o algoritimo busca os contornos que correspondem as camisas dos robôs, essa técnica consiste em seguir bordas até que ambos os lados se encontrem [4], e após isso ordena topologicamente os contornos encontrados, facilitando assim o descarte de algum contorno interno que venha a ser causado por ruídos. Por ultimo é feito um calculo para encontrar o centro de massa dos robôs com base no formato dos contornos.

IV. PATH PLANNING

Com base nas informações adquiradas pela visão, o método de navegação que demonstrou melhor eficácia nos robôs, foi o APF (Artificial Potential Field) como mostra a Fig. 8, o qual consiste em definir o caminho a ser seguido usando campos de vetores unitários que podem ditar a direção de movimento do robô.Um campo potencial é qualquer campo físico que obedeça à equação de Laplace, um algoritmo de campo potencial usa o campo potencial artificial para regular um robô em um determinado espaço. Duas espirais hiperbólicas são utilizadas para gerar os vetores, uma para fazer o robô se direcionar à bola e uma para fazer o robô desviar de obstáculos, ou seja, o robô é tratado como um ponto no espaço de configuração, sendo puxado pelo potencial atrativo em direção a um alvo definido e empurrado para longe dos obstáculos pelo potencial repulsivo. O resultado do processamento do módulo de planejamento de trajetória será um conjunto ordenado de pontos no plano que descrevem a trajetória a ser executada. A partir dessa trajetória, é gerado um objetivo e angulação instantâneos, que são pontos do caminho alcançáveis num intervalo de frame.

V. SISTEMA DE CONTROLE

Nesta seção, é apresentado o sistema de controle baseado apenas no modelo cinemático [5] para a fim de gerar as velocidades desejadas para o problema de acompanhamento de trajetória de referência de robôs móveis.

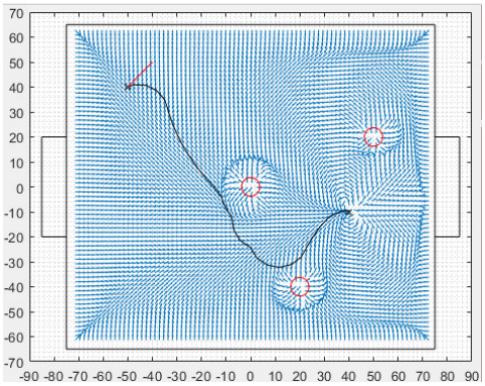

Fig. 4. Modelagem do ambiente utilizando APF.

O modelo cinemático para a movimentação do robô no ambiente pode ser modelado a partir das medições de velocidades angulares das rodas. Dessa forma, pode-se determinar a velocidade linear e angular do robô, através da equação [6]

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} r/2 & r/2 \\ r/d & -r/d \end{bmatrix} \begin{bmatrix} \omega_e \\ \omega_d \end{bmatrix} \tag{1}$$

Sendo d a metade da distância entre as rodas e ω_e e ω_d a velocidade angular da roda esquerda e da roda direita, respectivamente.

A Fig. 5 ilustra a movimentação do robô móvel em um período de amostragem T_s. Cada robô possui um vetor postura **q** dado pela tripla (x_c, y_c, θ_c), onde (x_c, y_c) são as coordenadas do centro de massa do robô. Dessa meneira, entre duas posturas consecutivas, temos a seguinte equação [6]

$$\begin{bmatrix} x_c\,(k+1) \\ y_c\,(k+1) \\ z_c\,(k+1) \end{bmatrix} = \begin{bmatrix} x_c(k) \\ y_c(k) \\ z_c(k) \end{bmatrix} + \begin{bmatrix} cos(\theta(k)) & 0 \\ sen(\theta(k)) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_c(k) \\ \omega_c(k) \end{bmatrix} \tag{2}$$

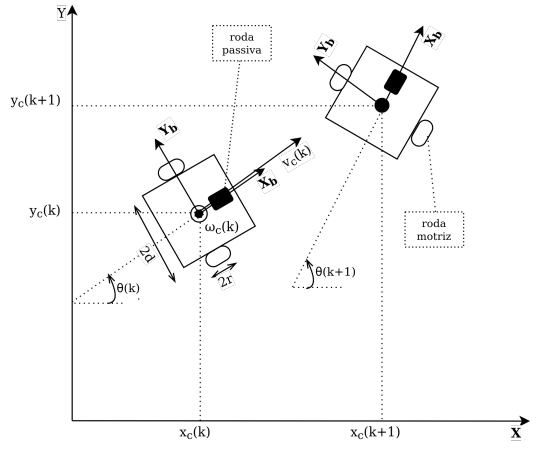

Fig. 5. Representação de dois estados consecutivos do robô móvel e seu sistema de coornedanas.

Para que o robô realize uma trajetória planejada deve-se projetar um bom controlador de posição e passar referências $x_r(t)$ e $y_r(t)$ variantes no tempo de acordo com a trajetória a ser seguida. Para isso, ao longo da trajetória as velocidades de referência angular ω_r e linear v_r devem ser [7]

$$\begin{aligned} v_r(t) &= \pm\sqrt{\dot{x}_r^2(t) + \dot{y}_r^2(t)} \\ \omega_r(t) &= \frac{\ddot{y}_r(t)\dot{x}_r(t) - \ddot{x}_r(t)\dot{y}_r(t)}{\dot{x}_r^2(t) + \dot{y}_r^2(t)} \end{aligned} \tag{3}$$

Nesse sistema, as variáveis de controle são as velocidades linear v_c e angular ω_c aplicadas a dinâmica do robô. Para o controle cinemático utilizamos um controle não linear invariante no tempo, também conhecido como NTI (*nonlinear time-invariant control*), na qual possui como lei de controle as seguintes equações [7]

$$\begin{aligned} v_s &= v_r cos(E_3) + k_1 E_1 \\ \omega_s &= \omega_r - k_2 v_r G_\theta E_2 + k_3 E_3 \end{aligned} \tag{4}$$

Sendo

$$\begin{aligned} E_1 &= (x_r - x)cos(\theta) + (y_r - y)sin(\theta) \\ E_2 &= (y_r - y)cos(\theta) + (x_r - x)sin(\theta) \\ E_3 &= \theta_r - \theta \\ \theta_r &= tan^{-1}(dy_r/dx_r) \\ \theta &= tan^{-1}(dy/dx) \\ G_\theta &= sin(E_3)/E_3 \end{aligned} \tag{5}$$

VI. ELETRÔNICA

Os componentes eletrônicos utilizados serão apresentados a seguir, sendo cada robô composto por um microcontrolador, uma ponte H, um par de motores, um par de encoders e um módulo de comunicação em rádio-frequência. O papel dos eletrônicos é bem simples e direto: o robô irá receber os comandos do servidor, controlar o acionamento dos motores e com auxílio dos encoders, realizar as estrategias previamente calculadas.

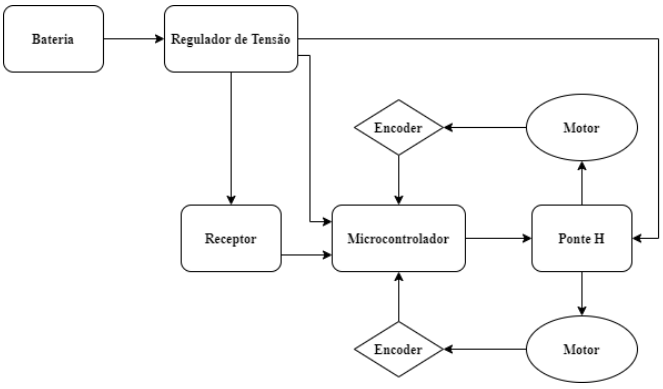

Fig. 6. Esquemático dos componentes e suas respectivas conexões

- Microcontrolador: o STM32F103C8T6 [Fig. 7] fará o papel de envio e coordenação das ”tarefas”. Um estará conectado ao computador e um módulo de comunicação, e passará as informações via serial ao comunicador, que então enviará para cada um dos robôs. Os outros, receberão as informações e vão ”comandar”

seus respectivos robôs: acionarão ou não os motores, enviarão valores de PWM para a ponte H, e farão a leitura e controle da movimentação utilizando os dados dos encoders.

- Transmissor/Receptor: o módulo de comunicação permite a comunicação computador-robô. Basicamente, o módulo ”principal” envia as informações necessárias para os módulos ”operantes”, aqueles que estão nos robôs. Para controlar qual robô ”escuta” receberá as informações adequadas, os módulos dos robôs permanecem fixados um em um dos 125 canais de frenquência operantes do NRF24L01 [Fig. 7], pois estes apenas recebem informações (Receptores), enquanto o módulo conectado ao computador pode variar de frequência pra frequência, comunicando-se com diferentes robôs (Transmissor). Trabalha com $2,4$ GHz de frequência, suportando 2000 kbps via radio-frequência e 1 Mbps a 2 Mbps via serial, com alcance de 100 m até 1000 m em ambientes abertos, variando com o uso de antenas.
- Ponte H: Este componente permite o controle de velocidade e acionamento dos motores, recebendo esses comandos do microcontrolador e então repassando para os motores, o modelo utilizado é o TB6612FNG [Fig. 7].
- Motor: Os motores são responsáveis pela movimentação, sendo acionados em velocidades específicas para ter o controle do movimento do robô. O modelo escolhido foi o N20 [Fig. 7].
- Encoder: os encoders são responsáveis por devolver ao microcontrolador a velocidade e a posição dos motores com precisão, permitindo assim um melhor controle de movimentação dos motores. Foram escolhidos encoders magnéticos do modelo TLE4946-2K, devido a instabilidade do ambiente em que estarão no momento da competição e pela capacidade dos mesmos de manter sua precisão nessas condições. Esses encoders se firmam no back-shaft do motor, de forma a não exceder as medidas do mesmo e conseguem até 12 contagens por revolução.

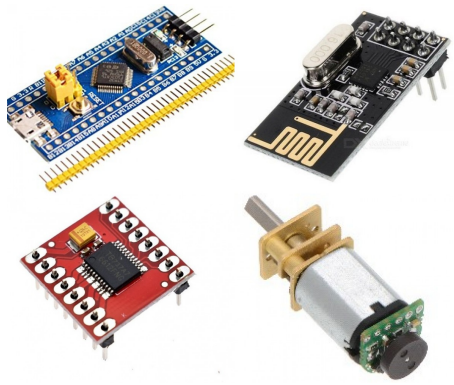

Fig. 7. Componentes eletrônicos utilizados

VII. SIMULADOR

Com o cenário atual da pandemia do COVID-19, os simuladores tornaram-se o principal ambiente de testes e aplicações, substituindo o ambiente físico, que para preservar a saúde dos membros, permanecem com seu acesso limitado. Foi utilizado o simulador FIRASIM, originalmente criado para a categoria SSL, e adaptado pelo equipe RoboCIN, com a ajuda de outras equipes, para atender as demandas da categoria IEEE-VSSS [8]. O simulador consegue representar a interface de comunicação, baseada em modelos que representam o domínio do problema de futebol de robôs e métodos uteis. Em seu sistema, receber os dados do FIRASIM é semelhante a receber dados do SSL-Vision usando a biblioteca Protobuf. Ademais, o envio de dados para o simulador também é possível com o uso das bibliotecas, já os clientes de amostra estão incluídos na pasta de clientes, os quais existem dois clientes disponíveis, baseado em qt e baseado em Java.

Fig. 8. API gráfica do simulador.

VIII. CONCLUSÃO

Este trabalho consistiu na descrição do projeto de futebol de robô da categoria *IEEE Very Small Size Soccer* desenvolvido pela equipe Robotech-UFPA para participar da competição *Latin American Robotics Competition 2020*, que será realizada remotamente devido ao cenário atual de pandemia. Neste primeiro trabalho foi possível colocar em prática várias áreas de conhecimento que possibilitaram o aprendizado técnico e organizacional dos integrantes da equipe. Este ano, apesar de todas a circunstâncias negativas, a equipe Robotech-UFPA atinge seu objetivo de formar um time competitivo de *VSSS* e busca estar adquirindo mais experiência com as equipes adversárias.

REFERENCES

[1] IEEE, IEEE Very Small Size Soccer (VSSS) Rules. Disponível em: ¡http://200.145.27.208/cbr/wp-content/uploads/2020/04/sslrules.pdf¿ Acessado em 28 de setembro de 2020.

[2] M. Colledanchise and P. Ogren, “Behavior trees in robotics and AI: an introduction,” CoRR, vol. abs/1709.00084, 2017.

[3] Trees for Autonomous Agents with Hybrid Constraints Evolution. Appl. Sci. 2018, 8, 1077.

[4] Suzuki, Satoshi. ”Topological structural analysis of digitized binary images by border following.” Computer vision, graphics, and image processing 30.1 (1985): 32-46.
[5] AICARDI, Michele et al. Closed loop steering of unicycle like vehicles via Lyapunov techniques. IEEE Robotics Automation Magazine, v. 2, n. 1, p. 27-35, 1995.
[6] CORKE, Peter. Robotics, vision and control: fundamental algorithms in MATLAB® second, completely revised. Springer, 2017.
[7] VIEIRA, Frederico Carvalho. Controle dinâmico de robôs móveis com acionamento diferencial. 2006. Dissertação de Mestrado. Universidade Federal do Rio Grande do Norte.
[8] IEEE, IEEE fira-simurosot/FIRASim . Disponível em: ¡https://github.com/fira-simurosot/FIRASim¿