RoboCupJunior Brasil – Rescue Maze – TDP Fran Robots

1˚ Arthur Mendes Lopes
SESI-SP
Franca, Brasil
arthur.mendes2801@gmail.com

2° Fernanda de Lima Gonçalves
SESI-SP
Franca, Brasil
fernandalima99995788@gmail.com

3˚ Guilherme Baptista Canteiro
SESI-SP
Franca, Brasil
guilhermebcanteiro@gmail.com

4° João Otávio Silva Ribeiro
SESI-SP
Franca, Brasil
silvaribeirojoaootavio@gmail.com

***Resumo* — Este artigo apresenta o desenvolvimento de nosso robô para a competição RoboCupJunior Rescue Maze, realizado ao longo de um ano e três meses. Para a navegação, tratamos o labirinto como uma grade de vértices e arestas (grafos), permitindo ao robô construir um mapa eficiente para localizar-se e encontrar rotas inteligentes. Optamos por priorizar tarefas no core (núcleo) 0 da ESP32, configurando-as com prioridade 1, o que resultou em um aumento significativo na velocidade de execução do programa. No hardware, além de um para-choque sensível equipado com botões que detectam paredes e obstáculos ao colidir, fazemos o uso de um sensor ultrassônico que valida a identificação de vítimas pela Raspberry Pi 4, garantindo que elas estejam em uma parede próxima. Isso previne os problemas anteriores com a detecção de vítimas falsas em paredes distantes. Também desenvolvemos um sistema de disparo com mecanismo biela-manivela para lançar kits de resgate em ambas as direções (direita e esquerda), utilizando apenas um servo motor. Essas inovações visam aumentar a eficiência e a precisão nas missões de resgate, reduzindo o tempo de resposta e aumentando a capacidade de adaptação do robô ao ambiente.**

***Palavras-chave* — Labirinto, RoboCupJunior-Brasil, Robótica-Educacional, Machine-Learning, K-Means.**

I. Visão Geral

É a segunda vez que estamos participando dessa competição, mas, nos inspiramos muito nos conhecimentos que adquirimos anteriormente na competição RoboCupJunior Rescue Line. Nesse ano, aprimoramos o design do nosso robô, buscando maior simplicidade e eficiência. A estrutura modular, com peças transparentes e 3D, facilita a manutenção e o diagnóstico de problemas. As esteiras, antes com borracha de câmara de ar, agora com silicone, garantem maior aderência em diferentes terrenos do labirinto.

Para a navegação autônoma, desenvolvemos um sistema em C++ [01] que representa o labirinto como um grafo [02], permitindo a construção de mapas e a busca de rotas otimizadas e inteligentes. Sensores detectam paredes, caminhos e vítimas nas paredes, auxiliando na localização e na tomada de decisões.

A visão computacional, implementada em Python [03] com OpenCV [04], utiliza os 7 Momentos de Hu [05] e o algoritmo K-Means [06] para identificar e classificar vítimas com alta precisão, mesmo em condições de iluminação variável. Essa combinação de técnicas permite ao robô reconhecer padrões visuais complexos e adaptar-se às diferentes situações encontradas durante a competição.

Os aprimoramentos realizados no robô desde o ano passado resultaram em um desempenho consideravelmente melhor. Sua maior estabilidade e precisão na identificação de vítimas demonstram um avanço significativo em direção ao objetivo de criar um sistema de resgate autônomo mais eficiente.

II. Planejamento Geral do Projeto

O principal objetivo da nossa equipe nesta competição é aprimorar de forma significativa as capacidades de reconhecimento de imagem do nosso robô, buscando um desempenho otimizado. Para isso, definimos metas específicas que incluem melhorar a precisão do sistema de reconhecimento e reduzir o tempo necessário para a conclusão das tarefas.

Nossas atividades estão registradas no Diário de Engenharia (Figura 1), desenvolvido no Canva [07], o que nos permitiu documentar nossas ações diárias com eficiência e manter uma organização clara. O diário também funciona como um planejador de metas a curto e longo prazo, além de incluir detalhes importantes de tarefas específicas. Registrar essas informações torna a execução dessas atividades mais prática no futuro, já que todos os procedimentos e soluções anteriores estão devidamente registrados.

O uso do Canva, que reúne todos os componentes do robô, também foi essencial durante o processo de montagem, proporcionando acesso simplificado e prático a informações críticas, como datasheets dos sensores.

Figura 1
Páginas do Diário de Engenharia da Fran Robots, capa geral, capa da área, página aleatória, respectivamente.

III. Estrutura do Robô e Eletrônica

Nosso robô foi projetado com uma arquitetura robusta e modular, permitindo adaptações rápidas e soluções eficazes durante a competição. O hardware principal é composto por uma série de componentes cuidadosamente selecionados e otimizados para garantir o máximo desempenho no labirinto.

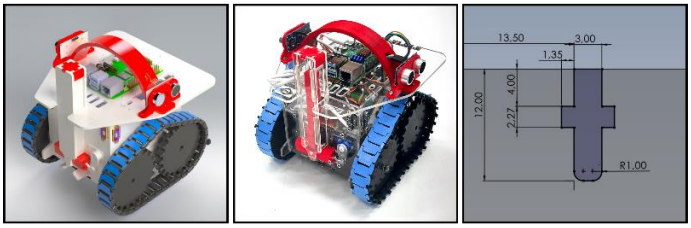
A. Mecânica e Manufatura

1. Estrutura Principal

Projetado em CAD [08] (Figura 2a e 2b), o robô prioriza confiabilidade e estabilidade enquanto mantém um tamanho compacto, simplificando o desenvolvimento do software e da

eletrônica. Ele possui fácil acesso por cima e por baixo, com poucos parafusos para manutenções rápidas. Porcas coladas com supercola em "junções T" (Figura 2c) evitam perdas e facilitam a montagem rápida. Os sensores e as câmeras são meticulosamente alinhados dentro das restrições físicas (tamanho e campo de visão), garantindo uma visão abrangente do labirinto.

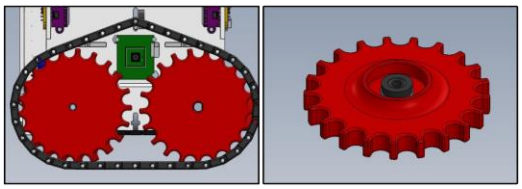
FIGURA 2
(A) MODELO 3D DO ROBÔ; (B) ROBÔ REAL; (C) JUNÇÃO EM T.



2. Rodas, Motores e Encoder

As engrenagens, uma acionada por um motor e a outra livre (Figura 3a), movimentam as esteiras do robô. Um parafuso e uma porca são usados para fixação da "roda boba". Os motores operam com tração nas rodas dianteiras, priorizando um melhor desempenho ao subir rampas, ao mesmo tempo que protegem o disparador de kits durante as descidas. Um encoder [09] é colocado no lado direito com um ímã, dentro da roda livre (Figura 3b), Ele lê as rotações e as conta para validação dos ladrilhos e movimentos mais rápidos do robô

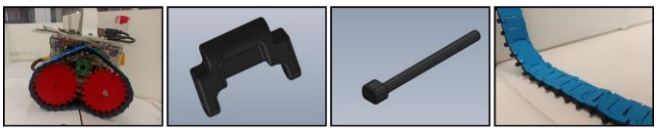
FIGURA 3
(A) RODA LIVRE (ESQUERDA) E RODA MOTORA (DIREITA); (B) RODA LIVRE E ÍMÃ DO ENCODER.



3. Esteiras Modulares

As esteiras do nosso robô foram baseadas em nossa estratégia anterior na Olímpiada Brasileira de Robótica [10]. Elas garantem maior estabilidade geral, especialmente ao subir escadas (Figura 4a), superar redutores de velocidade e navegar por detritos espalhados pela arena. A parte superior garante que as rodas não batam nas paredes do labirinto e as escalem, ajudando no alinhamento. Nossa esteira é modular. De maneira similar à elos de uma corrente, utilizar dentes impressos em filamento ABS e conectá-los por pinos (Figuras 4b, 4c) garante uma manutenção rápida e eficaz. A tensão da esteira pode ser facilmente ajustada por meio de um abre-longo na parte superior da peça, ele também serve de guia à esteira para que ela fique fora da visão da câmera. O tensionamento adequado se revelou crucial, pois afetou diretamente a movimentação prevista do robô. Para melhorar a aderência, adicionamos uma camada de silicone na superfície dos dentes (Figura 4d).

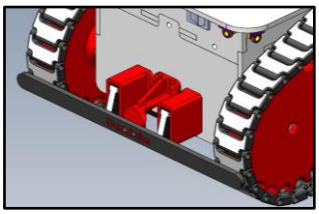
FIGURA 4
(A) ESTEIRAS DO ROBÔ SE MOLDANDO NA RAMPA; (B) DENTE DA ESTEIRA; (C) PINO DA ESTEIRA; (D) ESTEIRA COM SILICONE NO TOPO.



4. Para-choque táctil

O que anteriormente eram duas peças frontais passivas se tornou um para-choque com micro-switchs (Figura 5). Além da função anterior de prevenir um possível tombo frontal devido ao peso dos motores, agora ele também é capaz de, junto ao teto, ajudar nos alinhamentos e identificar obstáculos em ambas as extremidades do mesmo.

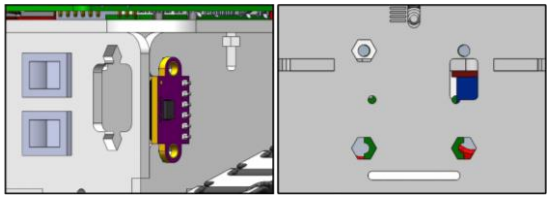
FIGURA 5
PARA-CHOQUE TÁTIL



5. Módulos

Sensores de distância (Figura 6a), câmeras (Figura 6b), a placa de refletância, a parte superior, inferior e frontal inferior são modulares para permitir uma remoção independente e fácil do chassi. A fixação delas com parafusos e porcas embutidas foi aprendida através de experiências em competições passadas, pois facilita a manutenção e o diagnóstico.

FIGURA 6
(A) SENSORES DE DISTÂNCIA MODULARES; (B) ENCAIXE DA CÂMERA MODULAR.

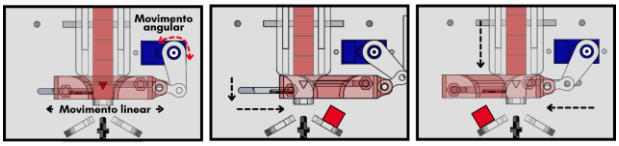


6. Mecanismo de Liberação dos Kits de Resgate

Melhoramos o mecanismo de disparo de kits de resgate do nosso robô, passando de um design unidirecional para um bidirecional, aumentando a precisão e eficiência. O novo mecanismo de biela-manivela (inspirado em uma ideia antiga de disparo por mola), é acionado por um servo motor que desliza, por meio de um êmbolo, suavemente os kits para ambos os lados sem que haja a necessidade de curvas do robô (Figura 7). Sua posição na traseira do robô evita que as esteiras atropelem os kits. Paredes laterais de limite que os impedem de baterem nas irregularidades da roda e terem seu escape alterado e abre-longos na manivela e biela do mecanismo que diminuem que travamentos ocasionais são as mais novas adições. Muitas versões foram criadas e registradas para garantir confiabilidade e precisão, prevenindo que os kits

travem ou fiquem fora da área pontuável. Alguns registros estão na seção de mecânica do nosso GitHub [11].

FIGURA 7
SEQUÊNCIA DE OPERAÇÃO DO MECANISMO DE BIELA MANIVELA.



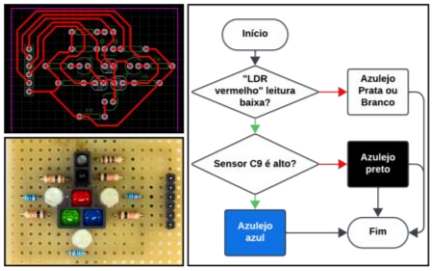
7. *Materiais*

- **Placas de acrílico:** O acrílico transparente proporciona visibilidade dos circuitos internos, auxiliando em diagnósticos, enquanto o acrílico preto, na parte inferior da tampa do chassi, previne a interferência nos sensores de cor da placa de refletância causada pela iluminação interna.
- **Filamento ABS:** O filamento de ABS possui baixa tolerância durante a impressão, durabilidade e resistência. Suas propriedades bem conhecidas facilitaram a confecção 3D e a simulação com CAE [12].

B. Design Eletrônico e Manufatura

- **I2C:** O sensor de distância infravermelho VL53L0X utiliza I2C [13] para conectividade com múltiplos sensores, detectando paredes e obstáculos. Um multiplexador de oito canais PCA9548A consolida os sensores em um único endereço, prevenindo interferências. Além disso, via I2C, o sensor giroscópio 9-DOF de orientação absoluta GY-BNO055 [14] rastreia a inclinação com precisão, protegido com papel alumínio aterrado para mitigar a interferência eletromagnética, funcionando como uma gaiola de Faraday [15].
- **Câmeras:** Duas webcams desmontadas nas laterais se conectam a uma Raspberry Pi 4 [16] via USB para identificação de vítimas. As webcams são genéricas e de baixo custo, conectadas com cabos USB, isso permite uma fácil substituição.
- **Placa de refletância:** Uma placa de circuito removível (Figura 8a) feita em placa ilhada, é posicionada no nível mais baixo do robô, agora, internamente. Ela contém um sensor emissor e receptor de infravermelho (C9L3), LEDs brancos de alta luminosidade de 5mm e LDRs com filtros de gelatina para cores vermelha, verde e azul. O sensor infravermelho detecta rapidamente a luminosidade da superfície e os filtros de gelatina atuam como barreiras ópticas RGB, simplificando a separação de cores no software, como mostrado no fluxograma (Figura 8b). Fisicamente, a luz consegue passar apenas pelo filtro de seu espectro luminoso, assim, quando deparado com uma superfície inteiramente azul a reflexão da luz branca dos LEDs adquiri essa característica e chega somente ao LDR coberto por esse filtro, gerando uma leitura de nível alto (circuito pulldown [17]), enquanto os outros se mantem com uma leitura de nível baixo. Com os três espectros RGB e uma leitura analógica estamos preparados para cores conhecidas e potenciais surpresas do desafio Super Teams e Técnico. Sua instalação interna agora permite o robô passar diretamente por cima de redutores paralelos às laterais do mesmo, solucionando problemáticas enfrentadas na fase mundial da última temporada.

FIGURA 8
(A) ROTEAMENTO DA PLACA DE REFLETÂNCIA E SUA FOTO; (B) FLUXOGRAMA DA LÓGICA DE IDENTIFICAÇÃO DE CORES.



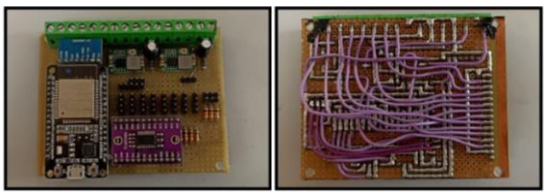
1. EasyEDA

Para criar esquemáticos, diagramas elétricos e rastreamento de circuitos, usamos o software de design eletrônico gratuito EasyEDA [18]. Ele preveniu erros de produção e economizou tempo e recursos. Conforme necessário, os circuitos foram modificados fisicamente e depois atualizados online, mantendo a documentação atualizada.

2. Controladores

- **ESP32:** O núcleo do robô, uma microcontroladora ESP32 [19], gerencia movimentos e decisões, processando dados dos sensores para distância, cor, orientação e classificação de vítimas. Foi escolhida por sua capacidade de multitarefa, velocidade, memória e tamanho compacto, quando comparado com um Arduino [20]. Um capacitor eletrolítico de baixa capacitância entre os pinos "EN" e "GND" permite a entrada automática no modo de programação do chip.
- **Raspberry Pi 4:** A Raspberry Pi 4 8GB [16] foi selecionada por seu alto processamento, velocidade, liberdade no código, dimensões compactas e confiabilidade. Ela envia todas as classificações das vítimas para a ESP32 via o protocolo UART [21], que executa a tarefa de resgate.
- **Placa mãe:** A placa central (Figura 9) agrupa os principais componentes, permitindo a substituição eficiente do "cérebro" do robô. A fiação interna e as trilhas de solda facilitam as conexões com bornes e conectores posicionados estrategicamente, minimizando os comprimentos dos fios e garantindo substituições rápidas em caso de falhas ou defeitos.

FIGURA 9
FRENTE E VERSO DA PLACA MÃE



3. Sistemas e Subsistemas de Alimentação

- **Alimentação:** Uma bateria LiPo de 12V 2.2Ah passa por dois reguladores de tensão Step-Down [22] Mini 360 e um Step-Up [23] Mt3608. Um dos reguladores Step-Down

fornece 5V para um servomotor, uma fita de LEDs e a Raspberry Pi [16], e o outro fornece 3V3 para a ESP32 [19] e sensores gerais. Eles foram escolhidos devido ao seu tamanho compacto e capacidade de 3A. O regulador Step-Up Mt3608 eleva a tensão para 15V para o circuito de alimentação da Ponte-H [24], aumentando a velocidade e a eficiência do motor enquanto mantém sua potência. Cada regulador inclui capacitores de 220uF em suas saídas como filtros passa-baixa para reduzir os efeitos de chaveamento em alta frequência. Um display voltímetro (MH-DL18S) mostra os níveis da bateria, ajudando a manter a tensão do robô e prevenir seu uso excessivo durante os treinos.

- **Motores e Drivers:** Dois motores TEK8 12V 50 RPM, com redução de engrenagem e travas, são controlados por uma ponte H L298N para navegação eficiente e à prova de falhas. A redução de engrenagem do motor determina sua velocidade, enquanto suas travas físicas evitam rotações indesejadas durante o desligamento, aumentando a precisão especialmente em descidas. A maior tensão em relação aos níveis lógicos reduz o consumo de corrente, estendendo a vida útil dos componentes.

IV. Software

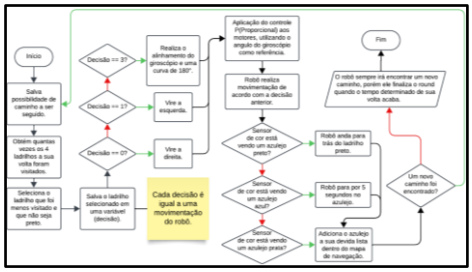
O *software*, desenvolvido em C++ [01] e Python [03], apresenta algoritmos avançados de navegação, processamento de imagens em tempo real com OpenCV [04] e sistemas de controle precisos para operação eficiente e inteligente do robô. A navegação é programada em C++ e organizada em segmentos distintos. Para reconhecimento de vítimas, foram empregadas técnicas puras de processamento de imagem sem machine learning [25] pré-desenvolvido ou IA seguindo as etapas: Captura de Imagem, Segmentação e Classificação.

Uma biblioteca de funções personalizada foi desenvolvida, contendo algoritmos frequentemente usados para simplificar o código final. Esta biblioteca inclui funções práticas para movimento, leitura de sensores e mapeamento de labirintos. Para sensores e atuadores específicos, utilizamos suas respectivas bibliotecas públicas, aproveitando suas implementações validadas para economizar tempo de desenvolvimento.

A. Navegação no Labirinto

O mapa é atualizado a cada novo azulejo visitado. Isso permite que o robô tome decisões ao escolher o próximo caminho, priorizando áreas não exploradas. Também focamos em movimentos precisos, enfatizando alinhamentos, cálculos de movimento e controles PD [26]. A lógica pode ser dividida em duas etapas, representadas pelos fluxogramas na Figura 10.

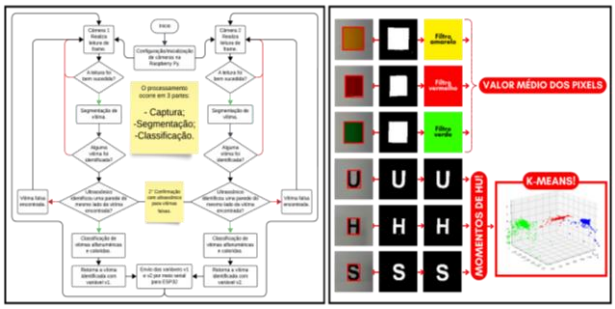
Figura 10
Analise dos ladrilhos de cor; processo de decisão; tomada de atitude.



B. Identificação de Imagem

A captura de imagem utiliza a biblioteca OpenCV [04] para processamento em tempo real das webcams de dois lados conectadas a uma Raspberry Pi 4 [16]. Primeiro, configuramos a resolução e a exposição das duas câmeras no Raspberry Pi e capturamos uma imagem de ambas, enviando-a para um filtro de segmentação de vítimas (baseado em thresholds adaptativos e de forma). Se alguma vítima é segmentada, a imagem resultante é enviada para a etapa de classificação. Para vítimas coloridas, consiste em filtros de faixa de três cores (verde, amarelo, vermelho) das imagens segmentadas convertidas para HSV [27], que são sujeitas a um limiar baseado no valor médio dos pixels. Para vítimas alfanuméricas (letras), as características da imagem são obtidas através dos 7 Momentos de Hu [05] e aplicadas à classificação estatística por meio de centroides obtidos por um algoritmo de agrupamento K-Means [06] em um espaço tridimensional (aproximadamente 1500 imagens (500 amostras de cada letra H, S e U em diferentes ângulos e tamanhos) usadas para convergir os centroides de classificação do algoritmo K-Means). Os valores resultantes são enviados à microcontroladora ESP32 [19] via o barramento serial (protocolo UART [21]). As Figuras 11a e 11b representam esses métodos. Os resultados foram satisfatórios, com a maioria dos valores de confiança acima de 90%.

Figura 11
(a) Fluxograma do algoritmo geral de processamento de imagem; etapas de segmentação e classificação de vítimas.



V. *Inovações*

A. Inovações Mecânicas e Elétricas:

- **Silicone nos Dentes da Esteira:** A camada de silicone nas esteiras do robô melhorou a tração em rampas e escadas do labirinto, se mostrando superior aos pedaços de câmara de ar anteriormente utilizados, pois é mais poroso e lida melhor com a sujeira.
- **Formato Chanfrado do Último Andar:** Os chanfros na extremidade traseira evitam que o robô colida com as paredes especialmente durante curvas sobre seu eixo. Solucionando muitos problemas que ocasionavam em desalinhamentos.
- **Biela-manivela no Disparador de Kits de Resgate:** Para dispensar os kits de forma bidirecional usando apenas um Servo motor, um mecanismo de biela-manivela aciona um êmbolo que direciona os kits para a direita ou esquerda. Abre-longos na estrutura mantêm o êmbolo sempre alinhado, enquanto um na manivela garante um movimento mais suave e evita travamentos, solucionando problemas das versões anteriores.
- **Kit de Resgate Antiquique:** O kit de resgate preenchido com bolinhas de estanho para solda solucionou nosso

problema em que os mesmos quicavam para fora da área pontuável de 15 centímetros.

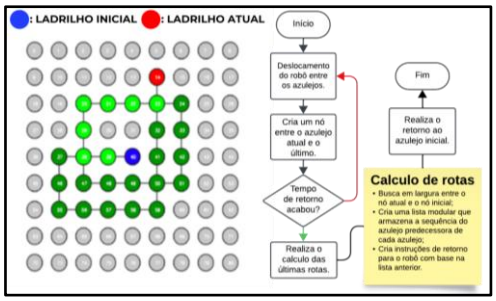
- **Para-choque Tátil:** O para-choque sensível ao toque manteve sua função de proteger as esteiras durante os alinhamentos e, além disso, detectam obstáculos no percurso.
- **Sensores Modulares:** Os sensores de distância e câmeras não são soldados diretamente. Em vez disso, são conectados |por barras pinadas, permitindo substituições rápidas e fáceis. A posição e o alinhamento são garantidos por parafusos e porcas previamente fixadas, o que facilita ainda mais a troca desses componentes.
- **Placa de Refletância com Filtros de Cor:** Para identificar os ladrilhos coloridos, utilizamos uma placa de refletância desenvolvida internamente, equipada com sensores LDR encapsulados e cobertos por filtros de gelatina luminosa nas três cores do espectro RGB. Esses filtros permitem a passagem apenas das cores correspondentes ao seu espectro. No código, quando um dos sensores registra um valor alto, os outros registram valores baixos, utilizando um circuito de pulldown.
- **Placa Mãe com Componentes Modulares:** Para identificar os ladrilhos coloridos, utilizamos uma placa de refletância desenvolvida internamente, equipada com sensores LDR encapsulados e cobertos por filtros de gelatina luminosa nas três cores do espectro RGB. Esses filtros permitem a passagem apenas das cores correspondentes ao seu espectro. No código, quando um dos sensores registra um valor alto, os outros registram valores baixos, utilizando um circuito de pulldown [17].

B. Inovações na Programação:

- **K-Means:** Utilizamos o conceito de K-Means para o processamento de imagem no momento de classificação das vítimas alfanuméricas. Utilizando de referência centroides gerados a partir de um banco de dados de cerca de 1500 imagens com vítimas em diferentes ângulos e posicionamentos podemos agrupar e separar os frames coletados das vítimas (H, S, U), classificando-as.
- **BFS:** É o algoritmo de busca no qual aplicamos ao mapeamento que nos retorna os ladrilhos identificados, e qual direção foi menos explorada, juntamente com a ordem de descoberta dos grafos.
- **Biblioteca de Navegação no Labirinto:** A biblioteca *"Maze_FranRobots.h"* foi desenvolvida por nós para auxiliar no controle de variáveis, rotas e mapeamento durante a navegação, além de implementar uma gama de funcionalidades essenciais.
- **Gerenciamento de Conexões de Nó:** A classe *Node Connections* gerencia as conexões entre diferentes nós no labirinto, armazenando até quatro conexões por nó e fornecendo métodos para defini-las e imprimi-las.
- **Gerenciamento de Listas:** A classe *List* é uma implementação personalizada de listas dinâmicas, semelhante ao que a biblioteca NumPy [28] faz em Python [03]. Ela permite alocação dinâmica de memória, redimensionamento, inserção, remoção **e** acesso a elementos, entre outras funcionalidades.
- **Estrutura de Ponto:** Define coordenadas bidimensionais dentro do labirinto, com métodos para inicialização e acesso às posições.
- **Algoritmos de Navegação e Busca:** A classe *"MazeRunner"* é crucial para a navegação do robô. Ela mantém o estado atual do robô, posição, direção e nós visitados. Além disso, implementa algoritmos de busca, como BFS [29], para encontrar os caminhos mínimos entre nós e gerar instruções de movimento.

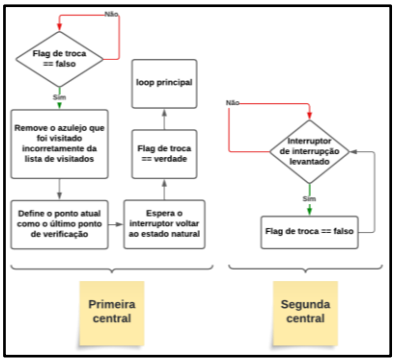
Esta biblioteca utiliza alocações dinâmicas de memória, que são essenciais para gerenciar grandes conjuntos de dados, como nós e conexões visitadas no labirinto (Figura 12). Ela oferece uma solução escalável para o controle do robô, com foco em navegação autônoma, utilizando listas inovadoras e algoritmos de busca robustos para maior eficiência.

FIGURA 12
À ESQUERDA ESTÁ O GRÁFICO DO LABIRINTO: VERDE ESCURO PARA OS VISITADOS, VERDE CLARO PARA OS CAMINHOS MAIS CURTOS. À DIREITA, ESTÁ O FLUXOGRAMA DE NAVEGAÇÃO DO ROBÔ.



- **Task Direto no Núcleo (Core) 0:** Uma otimização de desempenho foi alcançada ao mover a execução do código no ESP32 [19] do *"void loop"* para uma tarefa de prioridade 1 no núcleo zero. Essa mudança interrompeu a de-priorização causada pelo sistema operacional, resultando em uma execução mais rápida.
- **Botão de Interrupção Multitarefa:** O segundo núcleo do nosso microcontrolador [19] lê um interruptor físico que congela o robô durante falhas de progresso, eliminando a necessidade de desligamentos e perda do mapa. Ele atualiza uma variável para *'false'* quando ativado, solicitando que o código principal pule tarefas até que o interruptor seja desativado (Figura 13). Durante essa pausa, variáveis são ajustadas para refletir a posição atual do robô e a direção cardinal com base no último ponto de verificação.

FIGURA 13
INFOGRÁFICO DO PROCESSO DE INTERRUPÇÃO.



- **Ultrassônico para Dupla Verificação de Vítima:** Realizamos uma 2° verificação com o sensor ultrassônico no momento de identificação de vítimas, para evitar o reconhecimento de vítimas falsas.

  No momento em que o sinal PWM chega para ESP32, analisamos se o sensor ultrassônico detecta uma parede a frente, caso sim ele dispara os kits, caso não ele identificou uma vítima falsa e não dispara os kits.

VI. Avaliação de Performance

Na nossa inauguração na competição *RoboCup Junior Rescue Maze*, enfrentamos desafios relacionados ao tamanho do robô e às portas I2C [13] para os sensores de distância a laser. As dimensões do robô se mostraram problemáticas devido ao nosso treinamento em ladrilhos ligeiramente maiores, o que nos levou a reduzir o tamanho da Placa Mãe, do chassi e das rodas para melhor manobrabilidade. Além disso, enfrentamos uma sobrecarga de canais I2C, o que exigiu a integração de um multiplexador para consolidar os dados dos sensores em um único canal alternado. Identificar e resolver esses problemas foi essencial para nosso progresso e evolução contínua na robótica.

Além disso, durante nossas sessões de treinamento, encontramos várias limitações com nosso robô. Contas detalhadas desses desafios estão meticulosamente registradas em nosso Diário de Engenharia [30], complementadas por vídeos dos problemas e suas soluções correspondentes, acessíveis através do nosso repositório GitHub [11] nas seções de mídia de cada área de trabalho respectiva. Uma tabela (Figura 14) mostra as principais mudanças feitas no disparador, montagem, modelos 3D, eletrônica, navegação no labirinto e reconhecimento de imagem desde que começamos a trabalhar na categoria (agosto de 2023), bem como ideias que tivemos, testamos, e que funcionaram ou não.

Figura 14
Tabela de ideias durante as temporadas de 2023-2024 e 2024-2025

	IDEIAS DESCARTADAS	VERSÃO INICIAL	VERSÃO FINAL
MODELAGEM 3D, DISPARADOR E MONTAGEM	• Disparador de elástico; • Disparador com dois servos; • Disparador deslizante; • Abre-longo para a roda movida; • Suporte angular para a câmera impresso em 3D; • Encoder para a roda traseira.	• Tamanho do robô retangular (= 28 cm no lado maior); • Disparador de kit unidirecional; • Disparador frontal; • As esteiras não completamente dentro do perímetro do robô; • Espaçamento menor e mais curto dos sensores; • Câmera de ar mas esteiras; • Porcas soltas nos sensores de distância.	• Tamanho do robô quadrado (= 18 cm de lado); • Disparador biela-manivela bidirecional; • Disparador traseiro; • Esteiras completamente dentro do perímetro do robô; • Espaçamento maior dos sensores para melhor detecção de paredes e obstáculos; • Silicone nas esteiras; • Porcas embutidas nos sensores de distância.
ELETRÔNICA	• Fabricação de PCBs; • Encoder dedicado com roda traseira; • Necessidade de terminal pré-isolado; • Duas ESPs; • Motores de 100 RPM; • Duas baterias; • Sensor de peso para inclinações; • Placa lihada de dupla camada.	• Três reguladores de tensão step-down maiores; • Múltiplos endereços I2C sem o multiplexador; • Filtro LDR de cor única; • Menos capacitores de filtro.	• Dois reguladores de tensão mini step-down e um regulador step-up para ambos os motores; • Encoder na roda livre; • Uso de um multiplexador; • Filtros LDR de cor RGB; • Capacitores para cada regulador de tensão; • Motores de 500RPM; • Caixa de Faraday no giroscópio.
NAVEGAÇÃO E RECONHECIMENTO DE IMAGEM	• Interrupção externa na ESP32; • Parar o robô após a segmentação de imagens; • Lógica A* para encontrar o melhor caminho de retorno; • Lógica de mouse para calcular distância; • Python rodando em um smartphone.	• Matriz para mapeamento; • Alinhamento do ângulo dos sensores de distância; • Função linear para identificar a distância do ladrilho entre o robô e a parede mais próxima; • Prioridade de decisão de movimento não modular.	• Mapeamento por grafo; • Alinhamento do ângulo do giroscópio; • Valores predefinidos; • Decisão inteligente com base no ladrilho menos visitado.

Conclusão

Com esta iniciativa, nosso objetivo é ter um impacto positivo na comunidade de robótica e avançar em nossa jornada profissional ao abraçar experiências e aprender com os erros. Identificamos desafios que permitiram que nosso projeto evoluísse de forma gradual e consistente. Nossos planos envolvem migrar o processamento de imagem para o Raspberry Pi [16] Zero para reduzir custos e consumo de energia. Além disso, estamos dedicados a expandir nossa proficiência em áreas críticas, como documentação, modelagem 3D, eletrônica, navegação e reconhecimento de imagem, para aprimorar continuamente nossos projetos futuros.

Em resumo, este trabalho representa uma evolução substancial desde nossa última participação na competição *RoboCup Junior* em julho de 2024. Reconhecemos que a jornada científica é contínua e dinâmica. Avançando, continuaremos a iterar e implementar melhorias em todas as facetas do nosso projeto, enquanto colaboramos ativamente com outras equipes e educadores. **A ciência nunca para, e nós também não.**

Referências

[1] C++ , o que é e por que você precisa aprender essa linguagem, https://www.dio.me/articles/c-o-que-e-e-por-que-voce-precisa-aprender-essa-linguagem, acessado por último em 17/10/2024.

[2] Teoria dos Grafos —Introdução, Definições, Matriz e Lista de Adjacência, https://medium.com/@anwarhermuche/teoria-dos-grafos-introdução-definições-matriz-e-lista-de-adjacência-2252d4800a44, acessado por último em 17/10/2024.

[3] O que é Python?, https://aws.amazon.com/pt/what-is/python, acessado por último em 17/10/2024.

[4] Uma introdução ao OpenCV, https://www.dio.me/articles/uma-introducao-ao-opencv, acessado por último em 17/10/2024.

[5] Shape Matching using Hu Moments (C++/Python), https://learnopencv.com/shape-matching-using-hu-moments-c-python, acessado por último em 17/10/2024.

[6] Entendendo Clusters e K-Means, https://medium.com/cwi-software/entendendo-clusters-e-k-means-56b79352b452, acessado por último em 17/10/2024.

[7] Sobre o Canva, https://www.canva.com/pt_br/about, acessado por último em 17/10/2024.

[8] Software CAD, https://www.autodesk.com/br/solutions/cad-software, acessado por último em 17/10/2024.

[9] Encoder Incremental, https://blog.lri.com.br/encoder-incremental-como-funciona-e-aplicacoes, acessado por último em 17/10/2024.

[10] Sobre a OBR, https://obr.robocup.org.br/sobre, acessado por último em 17/10/2024.

[11] Sobre o Git, https://docs.github.com/pt/get-started/using-git/about-git, acessado por último em 17/10/2024.

[12] Engenharia Assistida por Computador: o que é e como funciona?, https://www.esss.com/blog/engenharia-assistida-por-computador-o-que-e-e-como-funciona, acessado por último em 17/10/2024.

[13] Comunicação I2C, https://embarcados.com.br/comunicacao-i2c, acessado por último em 17/10/2024.

[14] Adafruit 9-DOF Absolute Orientation IMU Fusion Breakout - BNO055, https://adafruit.com/product/2472, acessado por último em 17/10/2024.

[15] Gaiola de Faraday, o que é? Qual a sua aplicação?, https://www.mundodaeletrica.com.br/gaiola-de-faraday-o-que-e-qual-a-sua-aplicacao, acessado por último em 17/10/2024.

[16] O que é o Raspberry Pi, https://diolinux.com.br/tutoriais/o-que-e-o-raspberry-pi.html, acessado por último em 17/10/2024.

[17] Entendendo os Resistores de Pull-Up e Pull-Down, https://blog.eletrogate.com/entendendo-os-resistores-de-pull-up-e-pull-down, acessado por último em 17/10/2024.

[18] About EasyEDA, https://easyeda.com/page/about, acessado por último em 17/10/2024.

[19] ESP32 Pinout: Detalhes e Conexões,

https://clubedomaker.com/esp32-pinout, acessado por último em 17/10/2024.
[20] Arduíno: o que é, funções, tipos e mais, https://blog.kalatec.com.br/arduino-o-que-e, acessado por último em 17/10/2024.
[21] Compreender UART, https://www.rohde-schwarz.com/br/produtos/teste-e-medicao/essentials-test-equipment/digital-oscilloscopes/compreender-uart_254524.html, acessado por último em 17/10/2024.
[22] Conversor step down, https://www.wesen.com.br/conversor-step-down, acessado por último em 17/10/2024.
[23] Conversor step up, https://www.wesen.com.br/conversor-step-up, acessado por último em 17/10/2024.
[24] Ponte H – O que é e como funciona!, https://www.manualdaeletronica.com.br/ponte-h-o-que-e-como-funciona, acessado por último em 17/10/2024.
[25] O que é machine learning (ML)?, https://cloud.google.com/learn/what-is-machine-learning?hl=pt-BR, acessado por último em 17/10/2024.
[26] O Controlador Proporcional-Derivativo (PD), https://www.ece.ufrgs.br/~jmgomes/pid/Apostila/apostila/node29.html, acessado por último em 17/10/2024.
[27] Teoria da cor aplicada aos sistemas digitais, https://sidigicor.blogspot.com/2011/02/modelo-hsv.html, acessado por último em 17/10/2024.
[28] Quem Somos, https://numpy.org/pt/about/, acessado por último em 17/10/2024.
[29] Métodos de Busca em Grafos — BFS & DFS, https://medium.com/@anwarhermuche/métodos-de-busca-em-grafos-bfs-dfs-cf17761a0dd9, acessado por último em 17/10/2024.
[30] Diário de Engenharia Fran Robots, https://www.canva.com/design/DAGOUH4spuc/E9yPPvWn3bpYneac6GS2cQ/edit, acessado por último em 24/10/2024.