ERUS Team - Team Description Paper 2020

Fernando Bisi[1] Lorena Bassani[2] Luiza Bolonha[3] Rafael Silva[4] Pedro Lelis[5]

***Abstract*— This paper presents the submission of the ERUS team to the IEEE OPEN category of CBR 2020, which this year will be completely virtual. The paper will be presenting a brief history of the team, the strategy adopted to solve the proposed challenge, as well as the description of the models and software developed for the challenge simulation, and a short introduction of how the team visualized their work being reproduced in reality.**

***Resumo*—Este trabalho apresenta a submissão da equipe ERUS para a categoria IEEE OPEN da CBR 2020, que neste ano será completamente virtual. Será apresentado um breve histórico da equipe, a estratégia adotada para solucionar o desafio proposto, assim como a descrição dos modelos e software desenvolvidos para a simulação do desafio, e uma pequena introdução de como a equipe visualizou seu trabalho sendo reproduzido na realidade.**

I. Introdução

A equipe ERUS foi fundada em 2012 por estudantes da UFES do curso de engenharia da computação e ex-membros do Laboratório de robótica educacional (LRE), os quais já haviam obtido excelentes resultados em competições da LARC: quatro vezes campeões na categoria IEEE SEK e três vezes campeões da Competição Brasileira de Robótica (CBR). Entretanto, visando participar de categorias mais diversas, a ERUS foi então fundada tendo como objetivo principal usar as competições de robótica para colocar em prática a teoria desenvolvida em sala de aula além de popularizar a robótica na comunidade capixaba.

Desde sua criação, a ERUS contou com o apoio do Programa de Educação Tutorial (PET) da engenharia da computação, que fez com que já nos primeiros anos a equipe obtivesse bons resultados: 7º lugar na LARC IEEE OPEN 2012 e posteriormente 2º lugar na LARC IEEE OPEN 2013. Conforme a equipe expandiu as operações para diversos projetos, em 2019 tornou-se o Programa de Extensão ERUS, vinculado ao Centro de Tecnologia da Universidade Federal do Espírito Santo, assumindo a importante missão de reunir diversos cursos da Ufes em prol do desenvolvimento da Robótica no estado do Espírito Santo.

Além de participar de competições, a ERUS também organiza o seu próprio evento: O Torneio de Robótica da UFES (TRUFES) o qual terá sua 7ª edição realizada em 2020, completamente online.

A categoria IEEE OPEN propõe um problema (geralmente ligado a alguma questão ambiental ou que pode trazer algum benefício para a sociedade) no qual cada equipe é livre para utilizar qualquer ferramenta para solucionar o problema. Este fato abre uma gama de opções e também de desafios para solução do problema, pois é necessário lidar com todos os detalhes e contratempos na construção dos robôs.

Neste ano, o desafio foi simular a locomoção de pacotes, realizada por robôs, dentro de um armazém industrial como visto na Figura 1. O robô terá de se locomover pelo ambiente e transportar caixas dos locais de origem, uma de cada vez, aos respectivos locais pré-determinados, que seriam prateleiras ou áreas marcadas no cenário. Para saber qual o local de destino, a caixa possui uma identificação de código de barras, coloração específica ou um conjunto de dígitos, que determina para qual região ou prateleira a caixa está destinada.

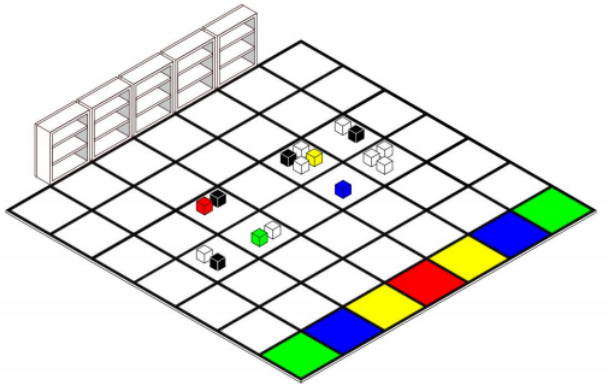

Fig. 1. Referência do ambiente do desafio IEEE OPEN 2020

II. Estratégia adotada

Para resolver o desafio de forma virtual, não foi necessário apenas projetar um robô, mas também estudar os diversos simuladores disponíveis e realizar a escolha mais adequada ao desafio e aos recursos da equipe.

O robô foi projetado de forma que realizasse o desafio com o mínimo de recursos, mesmo que estes fossem virtuais, mantendo a implementação o mais realista possível. Para isso, a equipe desenvolveu um robô capaz de navegar através de sensores de cor e reconhecer os padrões e cores dos pacotes através de uma câmera. Uma garra foi projetada para que o robô possa levar os cubos ao local adequado, estando

Universidade Federal do Espírito Santo - Ufes - http://ufes.br/
Equipe de Robótica da Ufes - ERUS - http://erus.ufes.br/
[1]fernando.b.vieira@aluno.ufes.br, Aluno do departamento de Informática da Ufes
[2]lorena.bassani@aluno.ufes.br, Aluna do departamento de Informática da Ufes
[3]luiza.onofre@aluno.ufes.br, Aluna do departamento de Elétrica da Ufes
[4]rafael.silva.37@aluno.ufes.br, Aluno do departamento de Elétrica da Ufes
[5]pedro.lelis@aluno.ufes.br, Aluno do departamento de Informática da Ufes

em um sistema elevador que permite carregar e posicionar os pacotes na região de descarregamento ou na prateleira mais adequada.

Além de hardware modelado de forma realista, o software utilizado foi projetado para ser facilmente adaptado ao mundo real, com a lógica de alto nível altamente desacoplada das partes dependentes de simulação (e automaticamente de hardware ao reproduzir o robô na realidade).

III. Hardware modelado para simulação

Durante o desenvolvimento do modelo de simulação, teve-se a preocupação de aproximar as principais características do projeto à realidade, seja no momento de construção do hardware ou software do robô.

A equipe procurou posicionar os componentes de modo que fosse deixado um local vago para o posicionamento do controlador utilizado. Além disso, antes de iniciar as simulações do projeto, foram feitas duas versões em softwares de modelagem 3D, procurando garantir que o modelo cumprisse as necessidades impostas pelo desafio.

O primeiro modelo, originalmente desenvolvido em plataforma Tinkercad[4], foi realizado com o intuito de conceptualizar as partes necessárias para completar o desafio, como também para referência no projeto que seria utilizado no simulador. A partir deste modelo, a equipe produziu um novo, utilizando o software Blender[1], que foi exportado para simulação, junto com todos os sensores e atuadores devidamente instalados durante esse processo.

A. *Desafio de manuseio de objetos*

Um dos maiores desafios deste projeto foi o manuseio de objetos por parte do robô, visto que para levar cada caixa até sua região ou prateleira, é necessário muito mais do que somente saber qual local deve-se depositá-los, é preciso estabelecer também uma forma segura de manusear tais objetos. Por isso, foi adotada a estratégia de empregar uma garra modelada de modo a otimizar o manuseio de objetos cúbicos, como os pacotes que o robô deve carregar, como mostrada na Figura 2.

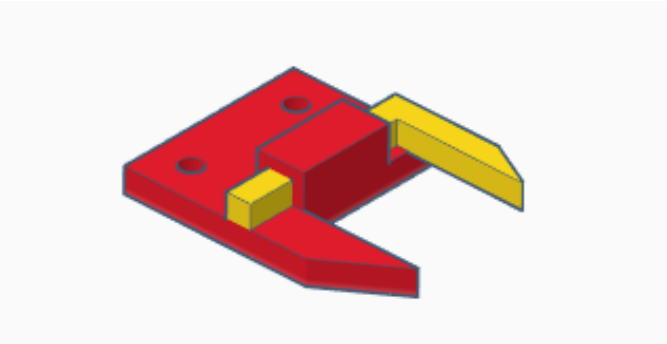

Fig. 2. Detalhe da garra projetada

Dentre as principais opções consideradas para o projeto da garra de manuseio dos pacotes, a primeira foi de fazê-la acoplada a um motor, para permitir a rotação do cubo com o propósito de alinhá-lo à ferramenta responsável pela detecção do modelo de pacote, aumentando sua eficácia. No entanto, concluímos que um sistema onde a ferramenta identifica o cubo anteriormente à etapa de atuação da garra seria mais eficiente, permitindo uma operação mais simples do robô ao eliminar a necessidade de rotacionar o cubo.

Além disso, pensando no posicionamento de outros elementos do robô e na necessidade de se elevar os pacotes para certas posições da prateleira, optou-se por acoplar a garra a um sistema elevador, como pode ser visto na Figura 3. Assim, evita-se uma possível obstrução desse objeto nas imagens da câmera e torna-se viável o adequado posicionamento dos objetos em qualquer ponto da prateleira. Desse modo, enquanto a câmera está sendo utilizada, a garra mantém-se elevada.

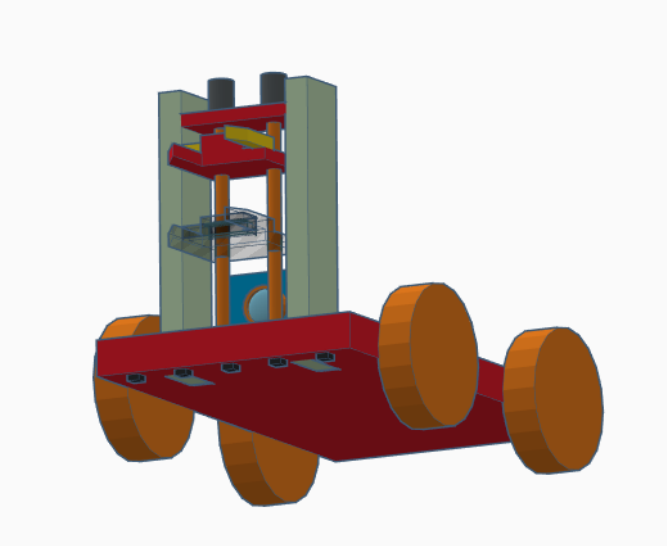

Fig. 3. Detalhe do sistema elevador da garra

B. *Desafio de detecção e reconhecimento de padrões*

Para detecção e reconhecimento de padrões, a equipe percebeu a necessidade do uso de uma câmera. Este também é um sensor modelado e disponibilizado no simulador utilizado pela equipe, de forma que o maior desafio foi projetar o posicionamento deste no agente robótico. Com a utilização de uma garra para o manuseio de objetos, a equipe tomou o devido cuidado de posicionar a câmera de modo que o elevador e a garra exercessem o mínimo de interferência no processamento de imagem, como visto da Figura 4, que apresenta a vista frontal do robô projetado.

C. *Desafio de navegação*

Para a navegação do robô móvel desenvolvido, utilizamos sensores de linha disponíveis no simulador Webots[3]. Com o posicionamento de cinco sensores na parte inferior do robô, como é possível perceber na Figura 5, ele é capaz de navegar pelo ambiente de simulação através das características deste, se orientando pelas delimitações encontradas no chão do ambiente simulado e mapeando-o através de um grafo em grade, deixando a complexidade maior para resolução via software.

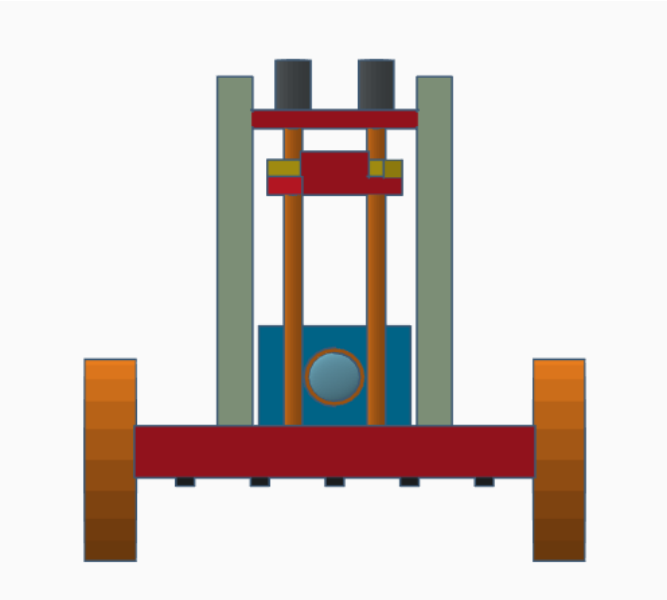

Fig. 4. Vista frontal do robô demonstrando posicionamento do elevador e da câmera

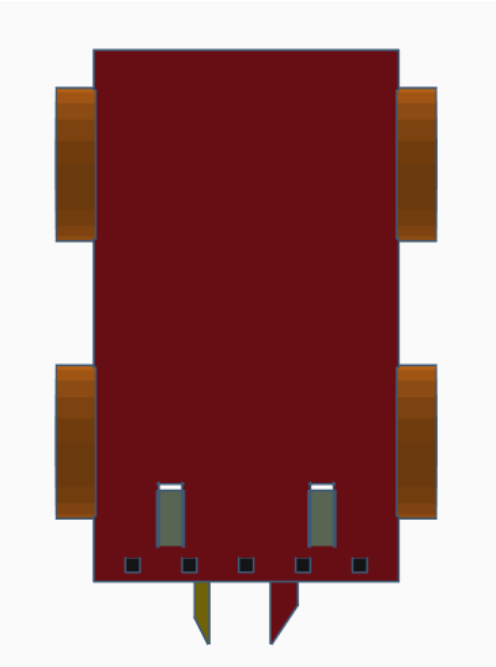

Fig. 5. Detalhe dos sensores de linha utilizados para navegação

Além disso, este também utiliza de detecção de objetos e padrões para se orientar através de *landmarks* (pontos de referência). Para isso, houve a necessidade de instalar uma câmera no robô, que se encontra atrás do elevador da garra.

IV. Software desenvolvido

Na simulação, o Software foi refletido com base no Hardware e arena desenvolvida no desafio, e chegou-se a decisão de realizar o processamento necessário para completar o desafio em uma arquitetura baseada em camadas.

Para isso, utilizaremos uma arquitetura multicamada vertical para a análise do problema, de forma que as etapas do desafio sejam facilitadas em nível de complexidade. Neste desafio, foram utilizadas 3 camadas vistas na Figura 6, que serão explicadas posteriormente em etapas, respectivamente, a camada de decisão, a camada de execução e a camada do movimento, sendo cada uma responsável por monitorar ou realizar uma das etapas de processamento necessárias para o sucesso do robô no desafio.

Cada camada envia informações para o nível de controle imediatamente subjacente, dessa maneira, sempre haverá coesão entre os comandos executados e os dados obtidos no código.

Como a entrada do processamento ocorrerá através de captura de imagem de uma câmera, para o tratamento das imagens será usada a biblioteca OpenCV[2].

A. Camada de decisão

A primeira camada tem como intenção permitir a realização de decisão por parte do robô. Este decide sua próxima ação, analisando as informações sobre onde este se encontra e o que fazer a seguir. Para isto é adotado uma máquina de estados[5], vista de forma simplificada na Figura 7, no qual o robô se torna capaz de decidir os passos

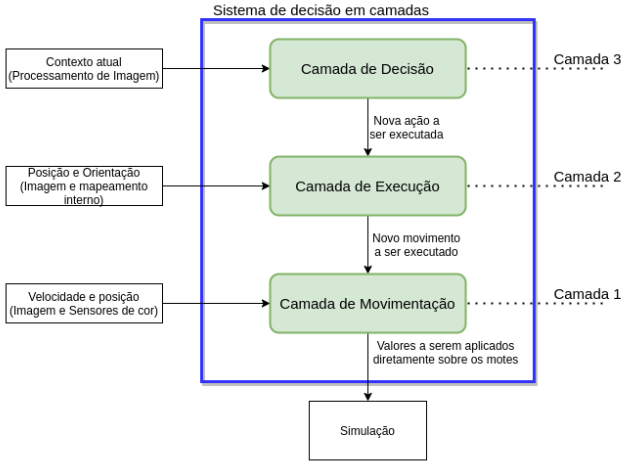

Fig. 6. Arquitetura multicamada utilizada no robô projetado para o desafio IEEE OPEN

futuros de forma determinística e consistente, analisando o contexto atual no qual ele se encontra.

B. Camada de execução

Com o resultado da camada de decisão, os passos de execução podem variar a partir desta. Para a decisão de mover-se, ele realizará processamentos de navegação. A partir da informação de para onde o robô deve ir, baseado nas condições atuais do ambiente e de seu ponto atual, ele fará um cálculo de como chegar ao ponto desejado, de forma que evite obstáculos, minimize a necessidade de alterar sua velocidade ou mudar de direção, e encontrar o caminho mais curto.

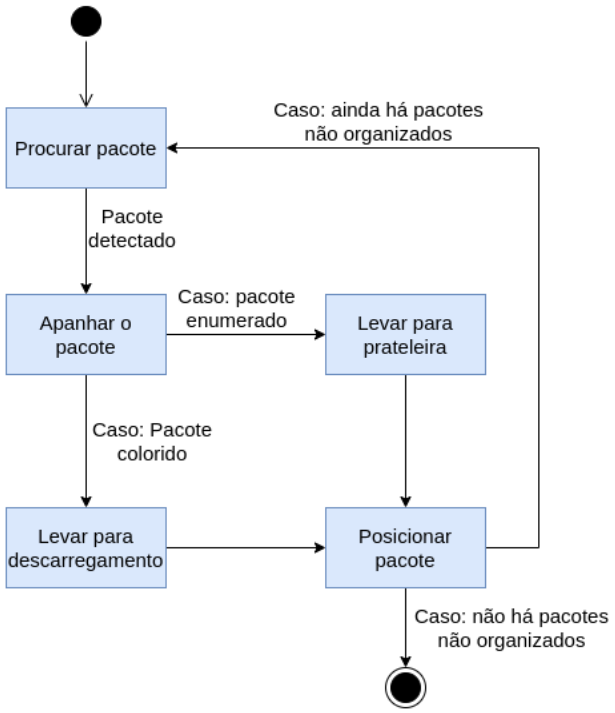

Fig. 7. Diagrama simplificado da máquina de estados de decisão do robô projetado.

Tal processo se baseia no conceito de Path-Planning[6], que considera os fatores mencionados anteriormente e a partir de algoritmos obtém o caminho mais adequado para se utilizar, como exemplificado pela Figura 8. O algoritmo implementado pela equipe para Path-Planning é o algoritmo A*, um algoritmo que combina técnicas de busca heurística com busca pelo menor caminho[6], dando o menor caminho entre dois pontos, caso este exista, de forma rápida e eficaz.

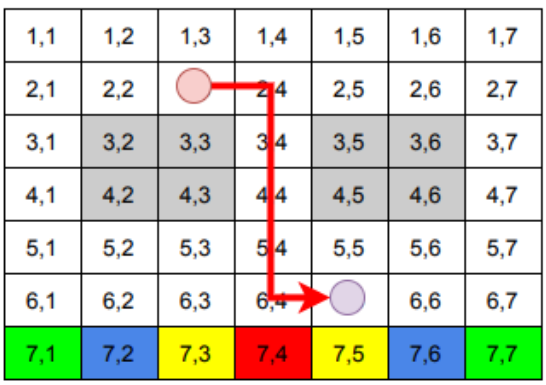

Fig. 8. Exemplo de caminho calculado via Path Planning

C. Camada de movimentação

A camada de movimentação é responsável pela atuação direta sobre o hardware. Esta irá modificar o estado do robô, atuando sobre o ambiente e controlando seus movimentos. Isto pode acontecer em dois contextos: mudança de posição do robô e manuseio dos objetos por parte deste.

Para a mudança de posição, temos rotinas de cálculo das velocidades adequadas dos motores que será utilizada em cada trecho de forma que tenha um balanceamento entre velocidade e precisão, com o intuito de completar a tarefa no menor tempo possível porém minimizando ao máximo o risco de realizar um erro de trajetória, tombar ou atingir algum dos pacotes devido a velocidade excessiva.

Para manuseio de objetos, temos rotinas análogas às de mudança de posição, porém com foco em realizar o manejo dos pacotes do ambiente da forma mais segura, precisa e ágil possível. O robô tem foco em não derrubar os pacotes, para que, em uma situação real, o conteúdo não fosse prejudicado, além de completar a tarefa ligada a cada pacote no menor tempo possível.

V. Simulação Projetada

Para realizar o desafio de forma remota, a equipe procurou desenvolver um ambiente de simulação que fosse o mais realista possível. Para isto, um dos primeiros desafios foi a escolha do simulador, que precisava levar em consideração os recursos pessoais de cada integrante, para que fosse possível que todos desenvolvessem suas tarefas em segurança, sem necessidade de deslocamento para utilização de recursos computacionais mais poderosos.

Após uma detalhada pesquisa dos principais produtos open source disponíveis, foi escolhida a plataforma Webots[3], que roda nos principais sistemas operacionais, permite o desenvolvimento do robô em diversas linguagens de programação e é compatível com os recursos pessoais dos envolvidos no projeto.

Foi projetada uma arena 3D de forma mais fiel possível a especificada no edital do desafio, que pode ser vista na Figura 9, utilizando o software de modelagem gratuito e open source Blender[1]. A partir dela e dos modelos produzidos do robô e dos pacotes, a equipe é capaz de desenvolver sua solução para o desafio proposto.

VI. Projeto de Desenvolvimento de Hardware Real

O desenvolvimento da solução por parte da equipe da ERUS passou pelo princípio de proporcionar um projeto final que pudesse ser reproduzido em hardware real. Para isto, foi feito um planejamento prévio de quais materiais seriam utilizados para o hardware de forma que minimizasse a necessidade de modificação de outras partes do projeto, em especial do software desenvolvido.

Com as necessidades de Hardware e Software definidos, foi escolhida a peça central de controle do robô. Optamos pelo Raspberry Pi[7], vista na Figura 10, que é um micro-computador acessível e com poder de processamento e suporte suficientes para o Software definido, além da presença

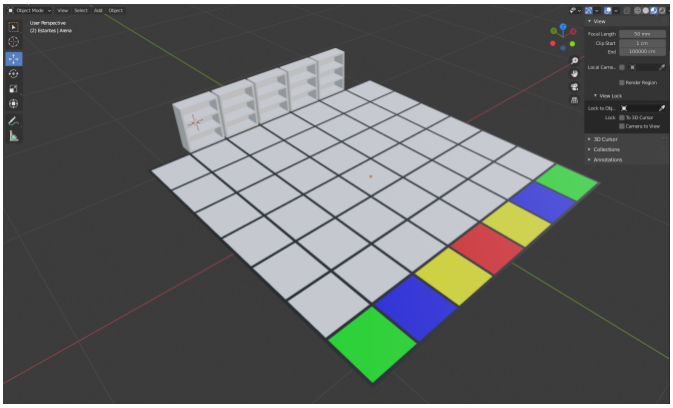

Fig. 9. Arena projetada pela equipe em ambiente de modelagem 3D.

de portas para comunicação serial compatíveis com protocolo UART (*Universal Asynchronous Receiver/Transmitter*), permitindo a utilização de módulos, sensores e microcontroladores diversos.

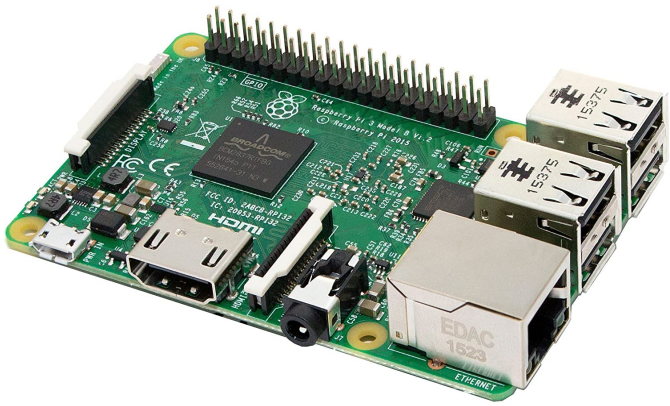

Fig. 10. Raspberry Pi 3 model B

Para a locomoção do robô, foi adotado quatro rodas com motores CC dedicados, porém, para manter uma separação entre os circuitos de tensão e de controle, o projeto contaria com dois módulos ponte H L298N (um para cada dois motores)[8]. Para os sensores de linha, foram escolhidos cinco módulos baseados no sensor TCRT5000[9] posicionados na parte inferior do robô. Para o sistema de garras e elevador, poderia ser utilizado um sistema com três motores de passos.

A equipe não desenvolveu o robô final com hardware real por motivos de segurança, mas prezou pelo desenvolvimento de um projeto que permita ser reproduzido assim que a equipe puder retornar às atividades normais.

VII. Conclusão

A proposta da equipe se caracteriza pela simulação do projeto mecânico na plataforma Webots e implementação de algoritmos, objetivando desenvolver um robô autônomo que cumpra as propostas feitas pela competição de maneira otimizada.

Ademais, pensou-se nas melhores estratégias para lidar com os desafios de Hardware, dos quais foram: manuseio de objetos, utilizando a garra; e detecção e reconhecimento de padrões, pela câmera. Em seguida, baseado no Hardware e em características da área de trabalho do robô, desenvolveu-se o Software com uma arquitetura de multi camada vertical para processamento das informações necessárias, sendo elas as camadas de decisão, de execução e de movimentação do robô.

Por fim, a competição se apresenta como um grande desafio para a equipe, levando em conta a crise global enfrentada no momento, que impossibilita uma competição presencial. Fundamentado nisso, busca-se apresentar uma simulação que possa ser reproduzida em Hardware real. Portanto, com base nas informações compartilhadas ao longo deste documento, a equipe encontra-se confiante para competir e otimista quanto aos resultados.

References

[1] “Blender 2.79,” https://www.blender.org/download/releases/2-79/, acessado em: 10-09-2020.

[2] G. Bradski et al., “The opencv library,” Doctor Dobbs Journal, vol. 25, no. 11, pp. 120–126, 2000.

[3] ”Webots 2020a rev1,” https://github.com/cyberbotics/webots/releases/tag/R2020a-rev1, acessado em: 25-09-2020.

[4] ”Tinkercad,”https://www.tinkercad.com/, acessado em: 10-09-2020.

[5] Rumbaugh, James, Grady Booch, and Ivar Jacobson. The unified modeling language user guide. Addison-wesley, 1998.

[6] Duchoň, František et al. Path planning with modified a star algorithm for a mobile robot. Procedia Engineering, v. 96, p. 59-69, 2014.

[7] Raspberry Pi 3 Model B. Disponível em https://www.raspberrypi.org/products/raspberry-pi-3- model-b/, acessado em 13-09-2019.

[8] Driver para ponte H L298N. Disponível em https://www.sparkfun.com/datasheets/Robotics/L298_H_Bridge.pdf, acessado em 25-09-2020.

[9] Sensor de Linha TCRT5000. Disponível em https://www.vishay.com/docs/83760/tcrt5000.pdf, acessado em 24-09-2020.