UM SISTEMA PARA PROGRAMAÇÃO E CONTROLE REMOTO DE UM ROBÔ 2WD USANDO UM APLICATIVO PARA ANDROID

Daniel Igor Alves Lima[1], Paulo Henrique Lopes Silva [2]

Resumo: Nos últimos anos, sensores e comunicadores foram capazes de gerar uma grande quantidade de dados sobre o ambiente, concedendo à nova geração de robôs uma inteligência e percepção de ambiente avançadas. Essa pesquisa é motivada pelo crescente aumento do número de dispositivos com tecnologia substancial para processamento e conexão com a Internet, o que eleva as suas possibilidades de uso em aplicações de diversas áreas. O presente trabalho apresenta o processo de desenvolvimento de um sistema que viabilize a programação e controle de um robô carro 2WD (duas rodas). Ele tem sua concepção na capacidade de executar ações de monitoramento de um ambiente, podendo se locomover por ele e se comunicar com dispositivos via wireless. A programação e controle do robô ocorre por meio de um aplicativo, no qual usuários podem se conectar, programar e disparar comandos a serem executados pelo robô carro. Tal processo de desenvolvimento envolve temas como engenharia de software, sistemas distribuídos, sistemas embarcados e internet das coisas e a viabilização de tal sistema se coloca como um desafio interessante e factível em muitos cenários de utilização. A implementação do sistema é validada por meio da observação de funcionamento de seus componentes, registrada em imagens e vídeos. A partir disso, conclui-se que o robô projetado é capaz de executar ações simples de locomoção e conexão com outros dispositivos em um ambiente controlado, comandadas remotamente por usuários. Vislumbra-se para o futuro a incorporação de novas funcionalidades, possibilitando ao robô carrinho realizar tarefas de monitoramento mais abrangentes.

Palavras-chave: Programação; Conexão *wireless*; Robô 2WD.

Abstract: In recent years, sensors and communicators have been able to generate a large amount of data about the environment, giving the new generation of robots advanced intelligence and environment perception. This research is motivated by the increasing number of devices with substantial technology for processing and connection to the Internet, which increases its possibilities of use in applications in several areas. The present work presents the process of developing a system that enables the programming and control of a 2WD robot car (two wheels). It has its conception in the ability to perform monitoring actions of an environment, being able to move around it and communicate with devices via wireless. The robot's programming and control takes place through an application, in which users can connect, program and trigger commands to be executed by the car robot. Such a development process involves topics such as software engineering, distributed systems, embedded systems and internet of things and the feasibility of such a system is an interesting and feasible challenge in many usage scenarios. The implementation of the system is validated through observation of the functioning of its components, recorded in images and videos. From this, it is concluded that the designed robot is capable of performing simple actions of locomotion and connection with other devices in a controlled environment, remotely controlled by users. The incorporation of new functionalities is envisaged for the future, enabling the cart robot to perform more comprehensive monitoring tasks.

Keywords: Programming; Wireless connection: 2WD robot.

1. INTRODUÇÃO

No passado, as três revoluções industriais contribuíram para o avanço da sociedade através da criação de diversas inovações, desde a mecanização de trabalhos anteriormente manuais ao desenvolvimento da internet. Atualmente, o mundo passa pelo processo da quarta revolução industrial, também chamada de Indústria 4.0, onde a robótica e a automação se tornaram o principal pilar (KARABEGOVIĆ et al., 2019).

Sensores e comunicadores foram um dos maiores beneficiados com o aumento do poder computacional dos últimos anos, e se utilizados em conjunto são capazes de fornecer inteligência simulada e uma percepção avançada do ambiente. Essa capacidade de gerar grandes quantidades de dados sobre o ambiente, em conjunto do uso da tecnologia *wireless,* concederam aos robôs de ações específicas como os utilizados há algumas décadas na indústria de automóveis, uma liberdade nunca antes vista, possibilitando a criação de um robô mais abstrato capaz de realizar tarefas organicamente, com a capacidade de se adaptar e responder a imprevistos, ou seja, semelhante a como humanos resolvem e respondem a suas tarefas. (PRUTHI, 2012). Tais fatos, motivam a academia e a indústria a criar modelos, arquiteturas e experimentos no contexto do desenvolvimento de

[1] *Discente do Bacharelado em Ciência da Computação da UFERSA - Campus Mossoró.*
[2] *Professor Adjunto IV do Departamento de Computação da UFERSA - Campus Mossoró.*

aplicações onde, cada vez mais percebe-se a presença de dispositivos computacionais interagindo e tomando decisões via conexões em rede.

Este trabalho se coloca como um dos esforços no desenvolvimento de uma arquitetura de sistema que propõe, dentre outras coisas, que um usuário final possa programar e controlar um ou mais dispositivos em uma dada aplicação ou tarefa (SILVA e BURLAMAQUI, 2016). Trata-se de um trabalho de escopo grande e que necessita de ações articuladas como a que é proposta neste estudo.

Sendo assim, o objetivo deste trabalho é desenvolver um sistema onde um robô do tipo carrinho pode executar ações de escopo simples via programação prévia feita por meio de um aplicativo. Além disso, o sistema ainda oferece suporte à conexão do robô com outros dispositivos, possibilitando interação e colaboração distribuída na execução de tarefas.

O processo de desenvolvimento e validação deste trabalho é apresentado nas seções a seguir.

2. DESENVOLVIMENTO

2.1. Fundamentação teórica

A motivação para essa pesquisa, foi ocasionada pelo grande aumento de pesquisas na área de *Internet of Things,* ou IoT, tecnologia essa, representada por sistemas distribuídos intrinsecamente ligados à constante comunicação entre seus dispositivos (DA XU et al., 2014).

Diante disso, foi traçado o objetivo de criação de um arquitetura com algoritmos capazes de controlar esses sistemas de forma simplificada, beneficiando qualquer nova pesquisa centrada na comunicação de robôs móveis em um ecossistema IoT. Para a sua criação, o projeto utilizou-se das ideias de desenvolvimento de algoritmos de sistemas distribuídos por SANTORO (2016). Em sua obra, o autor define que um sistema distribuído consiste em um conjunto finito de entidades, representados pelos integrantes computacionais do sistema, que se comunicam por meio de mensagens, uma sequência finita de bits, para atingir um objetivo, seja ele uma tarefa programada, uma solução computacional, ou uma requisição de um usuário ou outra entidade.

2.2. Material e Métodos

2.2.1. Software

No que tange ao aplicativo, foi utilizada para todas as suas funções a linguagem C# em conjunto da plataforma de desenvolvimento de jogos e software Unity (UNITY, 2022). A plataforma foi escolhida por fornecer uma praticidade na portabilidade do aplicativo para diferentes plataformas e utilizar a linguagem C# que facilitou o processo de comunicação com o robô por fornecer uma robusta biblioteca de conexão via *Sockets*.

Na testagem, o simulador de dispositivo da Unity foi utilizado para fornecer um dispositivo similar a um *smartphone* sem sair da interface de desenvolvimento, facilitando a implementação, de maneira que elimina a necessidade de conectar e enviar um protótipo do programa a um *smartphone* real. Na versão final, foi utilizado um *smartphone* modelo Mi 9T Pro, com processador Snapdragon 855 e 6GB de memória RAM.

Para o desenvolvimento do robô foi utilizada a linguagem *Arduino Programming Language* que consiste em um framework construído utilizando a linguagem C++. Em conjunto disso, foi utilizado a plataforma Arduino IDE (ARDUINO IDE, 2022), que fornece todas as funções necessárias para o desenvolvimento e testagem de uma placa de prototipagem baseada no sistema Arduino.

2.2.2. Placa de prototipagem eletrônica

A placa de prototipagem utilizada foi a WeMos D1 R2 (Figura 1): uma solução compatível com Arduino que possui o microprocessador ESP8266 integrado, capaz de realizar comunicação wireless, além de ter uma capacidade computacional maior. Essa integração foi essencial na escolha da placa no lugar de um Arduino tradicional com um módulo ESP8266 separado, não só por tornar todo o processo de conexão com dispositivos mais fácil, mas também por contribuir para a miniaturização do projeto, essencial em um robô móvel e compacto. Outro ponto para a escolha da placa foi a quantidade superior de portas digitais disponíveis comparada a outras soluções arduino com conectividade wireless.

O ESP8266 implementado no WeMos D1 R2 possui as seguintes especificações: uma tensão de operação de 3,3V, tensão de entrada suportada entre 9V e 24V, 11 portas digitais, 1 porta analógica e 4MB de memória *Flash* (SILVEIRA, 2018).

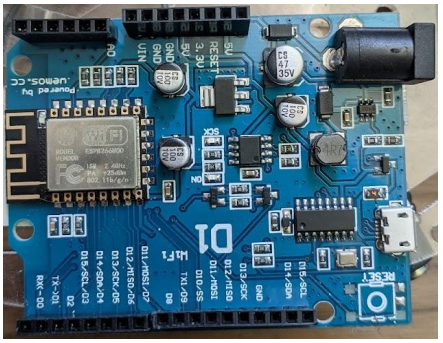

Figura 1**.** WeMos D1 R2. (Autoria própria)

2.2.3. Ponte H

Para controlar os motores DC escolhidos para o projeto, foi necessário a utilização de uma Ponte H, um circuito responsável por três funções: fornecer a alimentação necessária aos motores, possibilitar o uso de tecnologia PWM, ou seja, controlar a intensidade dos motores por meio de uma porta digital do arduino, e por último fornecer uma linha de alimentação de 5V, que foi utilizada para alimentar a placa de prototipagem escolhida. Portanto, o modelo de Ponte H escolhido foi a L298N (Figura 2) que é capaz de exercer todas as funções necessárias e também manter a miniaturização para o projeto.

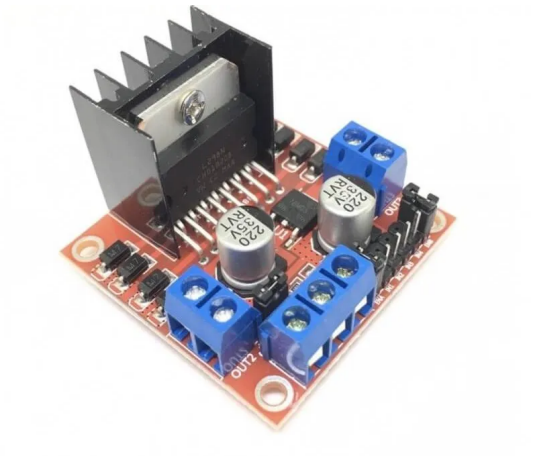

Figura 2. Ponte H. (XYKYO, 2022)

2.2.4. Alimentação elétrica.

A alimentação utilizada foi uma bateria (Figura 3) de tensão de 12V, corrente de 2.2A e capacidade de 2200mAh. Foi preterida em lugar de outras opções como pilhas AA convencionais, por ser mais compacta, possuir uma corrente maior e pela praticidade de ser recarregada apenas por um conector, onde as pilhas teriam que ser trocadas ou removidas para serem recarregadas.

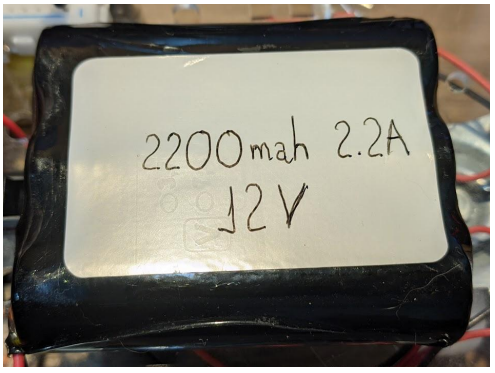

Figura 3. Bateria. (Autoria própria)

2.2.5. Integração e materiais complementares.

A integração final dos materiais utilizados foi feita visando não só a miniaturização robô, mas também a distribuição homogênea do peso, necessária para a estabilidade na locomoção de um robô móvel. Alguns outros componentes menores foram utilizados na integração final, são esses: conector 12v, interruptor, dois motores DC, chassi de acrílico, rodas, roda boba, protoboard, jumpers macho-macho e macho-fêmea.

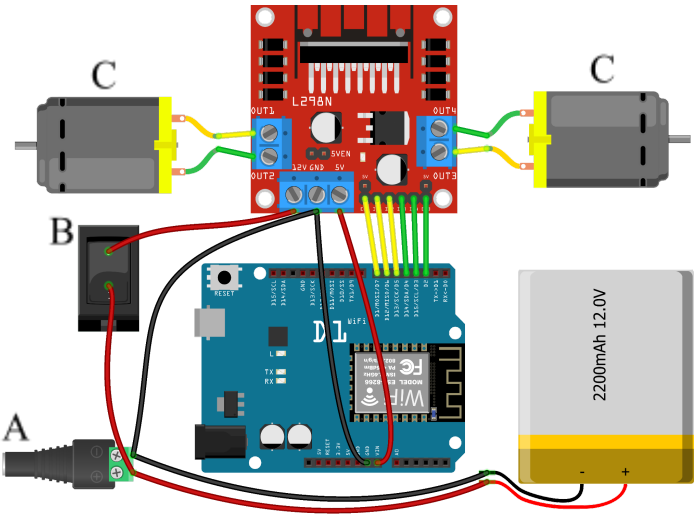

Figura 4. Representação do circuito final. (Autoria própria)

O conector (Figura 4A) foi implementado com o intuito de recarregar a bateria e possibilitar o uso do robô com alimentação direta por uma fonte 12V, que fez-se necessário na fase de testagem do projeto, possibilitando o prototipação das ações sem utilizar a carga da bateria.

O interruptor (Figura 4B) foi implementado com dois objetivos: garantir o desligamento do dispositivo evitando a perda de carga da bateria e fornecer segurança na montagem e prototipagem do robô onde os fios e bateria são manipulados pelo montador.

Dois motores DC (Figura 4C) foram utilizados com intuito de conceder mobilidade ao robô, apesar da sua simplicidade pelo baixo custo, em conjunto da função *PWM* da ponte H é capaz de realizar todas as funções utilizadas no projeto.

Um chassi de acrílico foi utilizado como base de todo o carro, onde os componentes foram alocados em suas posições. O material acrílico foi escolhido por não conduzir corrente elétrica, permitindo que os componentes sejam colocados diretamente no chassi sem isolamento extra. O acrílico, pela sua transparência, também fornece uma melhor visualização do robô.

Dois tipos de rodas foram utilizadas, duas rodas comuns utilizadas diretamente nos motores sem a capacidade de manobrabilidade, ou seja, possuem posições paralelas fixas em relação ao chassi do carro. O outro tipo utilizado foi a roda boba, uma roda acoplada na frente do carro com raio de giro de 360º, capaz de realizar rotação sem estar implementada a um eixo mecânico. Isso é possível, porque a roda boba responde a diferença de velocidade dos motores, ou seja, em um exemplo, onde o motor da esquerda esteja funcionando a 100% de sua capacidade e do da direita a 80% de sua capacidade, a roda boba auxiliará uma rotação no sentido horário, para direita.

Uma protoboard foi utilizada para fornecer uma linha de alimentação 5V para o arduino, vindo da ponte H. Ela também pode ser utilizada para possibilitar a adição de outros sensores com o avanço da pesquisa, desde que, sejam compatíveis com a tensão de 5V.

Integrando todos os componentes foram utilizados 12 *jumpers*, fios com terminais em suas pontas, sendo 6 macho-macho e 6 macho-fêmea.

2.2.6. Etapas de desenvolvimento

Para garantir que o projeto seja capaz de realizar suas funcionalidades corretamente, foi realizada a testagem individual de cada uma de suas funções. Para a obtenção desse objetivo, o projeto foi dividido em diferentes fases de desenvolvimento.

Inicialmente, foram levantados os requisitos para o projeto, ou seja, quais materiais e consequentemente os softwares seriam utilizados no seu desenvolvimento. Onde os requisitos foram moldados pelas necessidades do projeto e filtrados por limitações de custo.

Após o levantamento e a aquisição dos materiais, foi realizada uma montagem inicial do robô, um protótipo, onde a posição dos componentes podem ser mudadas e a miniaturização do projeto não é de grande importância nessa fase. Este protótipo foi utilizado para o desenvolvimento e testagem das funções distintas do robô, chamadas de módulos. Dois módulos foram criados: o de conectividade, responsável por realizar a conexão e envio e recebimento de mensagens, e o de gerenciamento, que é responsável por realizar os movimentos de rotação e aceleração do robô. Os módulos serão discutidos mais detalhadamente no tópico 3.1 sobre o aplicativo. Somente com a conclusão dos testes, o protótipo é modificado para a montagem final, visando a sua miniaturização e balanço de peso. A versão final do robô pode ser visualizada na Figura 5.

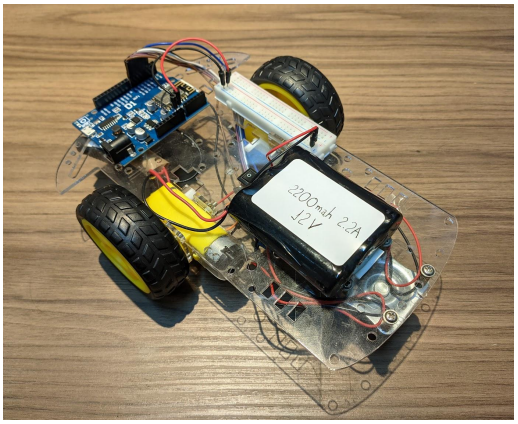

Figura 5. Resultado da montagem do robô. (Autoria própria)

Seguidamente, as funções do aplicativo também foram particionadas em módulos, que foram implementados separadamente. Três módulos foram criados: o de conectividade, responsável por se conectar com os robôs, o controlador manual e o controlador de tarefas. Após a testagem de todos os módulos do aplicativo, eles foram integrados em um único aplicativo com a interface de usuário final.

Por fim, ocorre a integração final entre robô e aplicativo, onde são estipulados os valores para cada operador utilizado nos comandos, onde são implementados tanto no aplicativo quanto no robô. Após isso, iniciou-se a testagem final onde a integração é avaliada e é checado se todos os operadores realizaram suas funções corretamente.

A seguir, na Figura 6, são apresentadas as etapas de desenvolvimento utilizadas no projeto

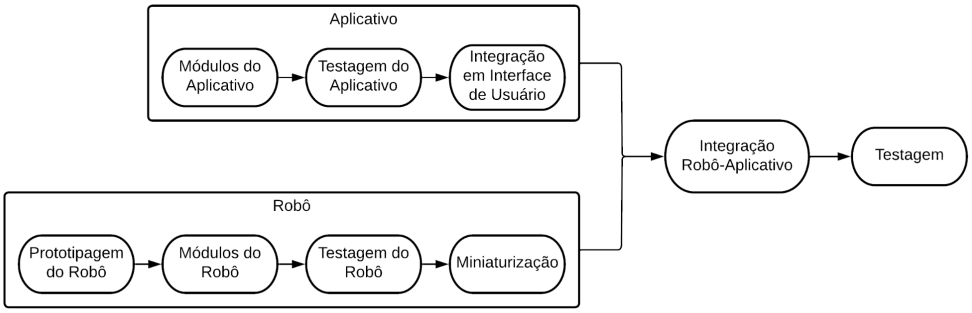

Figura 6. Ordenação das etapas de desenvolvimento do projeto.

Em suma, inicialmente o projeto separou completamente a implementação do robô e aplicativo com o objetivo de obter uma versão largamente testada de ambos, que tornaria a sua integração mais simples por permitir que qualquer erro encontrado nesta fase do desenvolvimento seja mais facilmente encontrado e resolvido, pelo fato dos testes individuais realizados antes garantirem o seu funcionamento individualmente.

3. RESULTADOS

3.1. Arquitetura do sistema

Para controle das funções do robô foi escolhido o desenvolvimento de um aplicativo para smartphones Android. A plataforma foi escolhida por tornar o projeto mais intuitivo e de mais fácil acesso aos usuários. Toda a arquitetura do sistema foi desenhada com o intuito de simplificar a comunicação entre o aplicativo e o robô. Para tanto, o programa centrou todo o envio de comandos no servidor e abstraiu as funções que o programa pode

exercer, sendo nomeadas de modos, de maneira que qualquer modo criado tem que se comunicar com o servidor para enviar os seus dados.

Diante disso, o passo a passo de operação do sistema proposto ocorre da seguinte forma: após o usuário escolher um modo e realizar uma ação, é gerado uma cadeia de dados que são enviados ao servidor. O servidor checa os dados, insere o operador lido do modo e monta um comando que é enviado ao robô. O robô lê o operador, escolhe a operação e lê o restante da cadeia de dados de acordo com a operação escolhida. Dessa forma, um modo pode criar o dado da forma, tamanho e segmentos livremente e, desde que o comando esteja referenciado no robô, a operação será executada independentemente.

É possível observar na Figura 7 que a arquitetura foi implementada particionando o uso do sistema em 4 partes, o usuário, que utiliza o sistema, o aplicativo, com as funções de servidor e controle, os robôs que realizam as ações programadas, e os dispositivos finais operados pelos robôs.

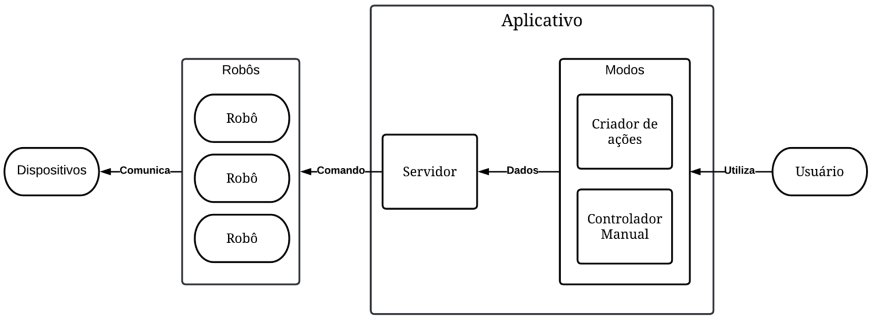

Figura 7. Representação geral da arquitetura. (Autoria própria)

3.1.1. Aplicativo

Para realizar as ações do robô, foi desenvolvido um aplicativo na engine Unity, utilizando linguagem C#. Ele foi implementado com a capacidade de realizar três funções: ser o servidor para múltiplos dispositivos, fornecer controle manual sobre um robô e criar e enviar tarefas. Essas funções estão representadas no diagrama de classes na Figura 8.

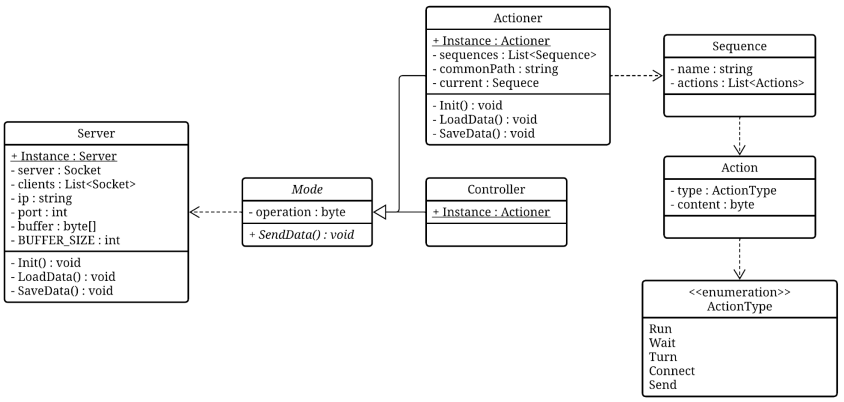

Figura 8. Diagrama de classes da implementação do Aplicativo

O servidor foi implementado utilizando comunicação de rede via *Sockets* padrão da linguagem C# e é capaz de se comunicar individualmente com os seus clientes ou de forma conjunta, possuindo, também, capacidade de receber mensagens dos seus clientes. Tudo isso deve ocorrer de forma assíncrona para que não haja problemas de lentidão na comunicação mesmo com múltiplos dispositivos conectados. Na Figura 9A é mostrado a tela inicial de atribuição de porta do servidor para inicializá-lo.

No sistema de tarefas, é possível atribuir uma lista de ações a um ou múltiplos robôs a qual realizam em sequência. Dentro dessas ações, o robô pode acelerar, virar, esperar, se conectar a uma rede, e enviar uma mensagem a outro dispositivo. As tarefas podem ser salvas para facilitar a reutilização de tarefas repetitivas. Nas Figuras 9B, 9C e 9D é mostrada todas as telas relacionadas a criação e administração das tarefas.

O controlador manual é outra opção de controle do robô, onde se fornecem ordens por meio de ajustes manuais de velocidade e direção. Foram implementados como recurso de reforço para situações nas quais as

tarefas não têm o desempenho esperado, seja por dificuldade de locomoção no ambiente ou por de uma conexão de rede insuficiente. Na Figura 9E é mostrada a tela controle manual, com os controladores de velocidade e direção.

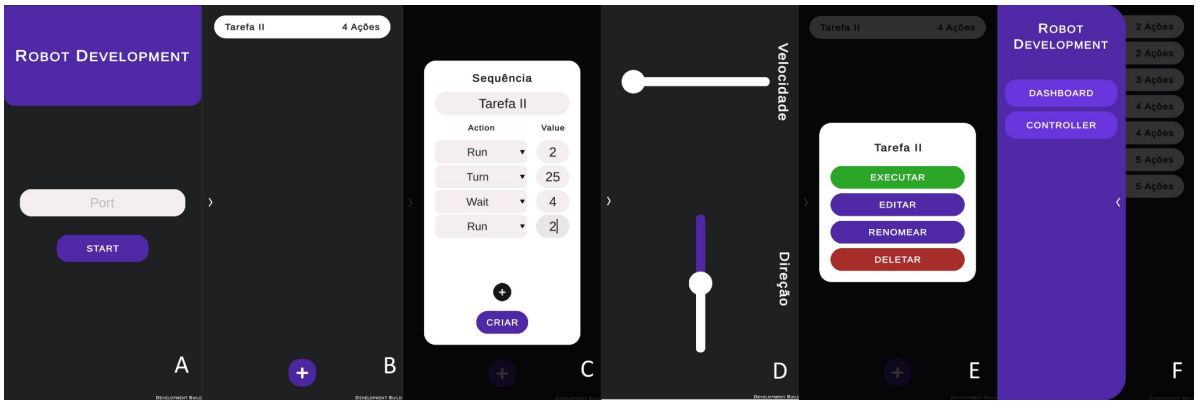

Figura 9. (A) Tela para atribuição da porta. (B) Tela de listagem de tarefas. (C) Tela de criação tarefas (D) Tela de opção de tarefa. (E) Tela do controlador manual. (F) Painel lateral de todos os modos implementados. (Autoria própria)

Inicialmente, o usuário interage com a interface de usuário, inserindo a porta de rede que o aplicativo utilizará, essa porta deverá condizer com a utilizada no robô, na versão atual do projeto foi utilizada a porta 1907. Após inserir uma porta válida, o usuário é transferido para o painel de tarefas, onde poderá ver as tarefas criadas, ou criar novas ações por meio de um botão com o símbolo de adição. Caso opte por criar um nova tarefa, será mostrado em uma janela *popup*, todos os campos necessários, como o nome da tarefa, e a cadeia de ações que será realizada, sendo possível adicionar mais ações por meio de outro botão com o símbolo de adição. No caso de um *swipe* para direita na *Dashboard* (Figura F), será mostrado ao usuário um menu lateral com todos os modos implementados, no caso do projeto, a própria *Dashboard* e o controlador manual. O usuário também pode interagir clicando diretamente sobre uma tarefa, onde poderá executar, editar, renomear ou deletar.

Por último, na tela do controlador, que foi implementada com o intuito de ser utilizada com o dispositivo no modo paisagem, o usuário verá dois ajustadores: O de velocidade, capaz de fornecer qualquer valor de aceleração entre 0% e 100%; e o de direção, onde o centro representa nenhuma rotação, a esquerda fornece uma rotação anti-horária e a direita que fornece uma rotação horária.

3.1.2. Comunicação

A comunicação utilizada entre os dispositivos foi feita utilizando protocolo TCP em rede local WiFi. Todos os comandos são fornecidos pelo servidor ao seus clientes, por meio de mensagens formadas por um operador e uma cadeia de dados, estruturadas conforme representada na Figura 10.

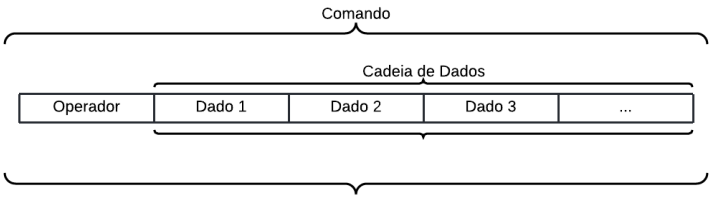

Figura 10. Estrutura de um comando. (Autoria própria)

Foi atribuído ao operador um valor de 8 bits, que carrega a informação de qual ação deverá ser realizada pelo robô. O número atual de operadores distintos suportados pelo aplicativo é de 256, onde oito operações já estão sendo utilizadas pela implementação atual do aplicativo.

A respeito da cadeia de dados, o seu tamanho é delimitado pela implementação da operação, onde dependendo da ação que será enviada a cadeia poderá ser vazia ou conter dados até um limite de 2KB, onde cada dado assim como o operador tem um valor de 8 bits, ou seja, uma cadeia pode conter até 256 dados. Na Tabela 1 é possível visualizar como foram estruturados os comandos do utilizados no aplicativo.

Tabela 1. Ação e dados para cada operador utilizado no projeto. (Autoria própria)

Operador	Ação	Dados	
0	Iniciar uma sequência de ações	N/A	
1	Ativar modo de controle manual	Velocidade do Motor 1	Velocidade do Motor 2
2	Ativar aceleração	Tempo de duração da aceleração	
3	Realizar rotação	Porcentagem de rotação, sendo 100% uma rotação completa de 360º	
4	Espera	Tempo de duração da espera	
5	Conectar-se a uma rede	Nome da rede	Senha da rede
6	Enviar mensagem	IP do dispositivo	Mensagem
255	Sair da sequência de uma tarefa	N/A	

Também, é importante ressaltar, que nessa arquitetura um comando é enviado ao robô em um único pacote. Essa implementação é importante por possibilitar que o robô se desconecte do servidor após receber um comando, podendo cobrir longas distâncias, realizar longas cadeias de ações e se conectar a outras redes para realizar uma tarefa, fornecendo uma grande versatilidade.

3.2. Custos do projeto

Em razão do projeto ter como objetivo oferecer um robô por um baixo custo, faz-se necessário a exposição dos valores dos componentes utilizados no projeto. Os valores presentes na Tabela 2 foram obtidos após uma pesquisa pelo menor valor no mercado brasileiro em 16 de junho de 2022.

Tabela 2. Custos dos componentes. (Autoria própria)

Componente	**Custo**
Esp8266 Wemos D1 R2	R$ 47,49
Kit Chassi para Robô 2WD	R$ 56,90
Kit com 10 jumpers macho-macho	R$ 12,73
Kit com 10 jumpers macho-fêmea	R$ 11,10
Protoboard de 400 pontos	R$ 18,51
Driver Motor Ponte H	R$ 20,11
Bateria li-ion de 12v, 2.200 mah	R$ 60,86
Total	**R$ 227,70**

4. CONCLUSÕES

Alguns desafios foram encontrados durante o desenvolvimento do projeto. Dentre eles, encontra-se a impossibilidade de utilização de um carrinho 4WD, que consiste em um modelo de carrinho composto por quatro motores, um para cada roda, onde os dois frontais são capazes de realizar as manobras do robô. Esse modelo de carrinho é capaz de realizar movimentos muito mais precisos, necessário em locais de difícil locomoção, mas trariam um custo elevado ao projeto, ferindo o propósito de robô de baixo custo. A escolha do modelo dos

motores também passou pela mesma limitação, onde os modelos com torque superior, capazes de superar pequenos obstáculos e de realizar movimentos em uma velocidade menor, gerando uma precisão maior, são normalmente encontrados por um valor muito acima do que os utilizados no projeto. Portanto, visando manter o baixo custo do projeto, foi tomada a decisão da implementação de motores de baixo torque utilizando o modelo de carrinho 2WD, que possui os dois motores de direção fixa, e a roda boba.

Diante do exposto, mais pesquisas podem ser feitas utilizando-se do sistema desenvolvido. Dentre essas, uma pesquisa sobre um modelo de robô sem as limitações de custo descritas seria benéfica para a testagem dos limites da tecnologia. Outra recomendação, seria a utilização do sistema de tarefas desenvolvido, no ambiente educacional, onde alunos poderiam utilizar-se do sistema para realizar ações no robô. Essa pesquisa poderá trazer ótimos benefícios para alunos sem conhecimento prévio de linguagens de programação, onde o desenvolvimento lógico utilizado poderia ser utilizado para uma transição natural para a programação tradicional.

Por fim, conclui-se que o projeto foi capaz de produzir um robô autônomo multiuso de baixo custo, capaz de realizar diversas tarefas em um ambiente, sendo possível sua implementação em cenários diversos, como industrial ou doméstico. O aplicativo para gerenciamento do robô demonstrou-se de fácil uso e intuitivo, podendo ser utilizado por usuários sem conhecimento na área de pesquisa do projeto.

5. REFERÊNCIAS BIBLIOGRÁFICAS

ARDUINO IDE: Open-source electronic prototyping platform. Versão 1.8.19. [S.l.]: Arduino, 2022. Disponível em: <https://www.arduino.cc>. Acesso em: 15 de maio de 2022.

DA XU, Li; HE, Wu; LI, Shancang. Internet of things in industries: A survey. **IEEE Transactions on industrial informatics**, v. 10, n. 4, p. 2233-2243, 2014.

FRITZING: *Eletronics made easy*. Versão 0.9.10. [S.l.]: Fritzing, 2022. Disponível em: <https://fritzing.org/>. Acesso em: 15 de maio de 2022.

KARABEGOVIĆ, Isak; TURMANIDZE, Raul; DAŠIĆ, Predrag. Robotics and automation as a foundation of the fourth industrial revolution-industry 4.0. In: **Grabchenko's International Conference on Advanced Manufacturing Processes**. Springer, Cham, 2019. p. 128-136.

PRUTHI, Sanil. Wireless robotics: A history, an overview, and the need for standardization. **Wireless Personal Communications**, v. 64, n. 3, p. 597-609, 2012.

SANTORO, Nicola. **Design and analysis of distributed algorithms**. John Wiley & Sons, 2006.

SILVA, Paulo Henrique Lopes; BURLAMAQUI, Aquiles Medeiros Filgueira. System for creation, programming and availability of distributed robot teams using collective knowledge. **IEEE Latin America Transactions**, v. 14, n. 10, p. 4392-4401, 2016.

SILVEIRA, André Luis. Arduino, Internet das Coisas e Computação vestível. 2018. Disponível em: <http://www.um.pro.br/arduino/index.php?c=Wemos_D1_R2_Wifi_ESP8266>. Acesso em: 12 de junho de 2022.

UNITY: Plataforma de desenvolvimento em tempo real. Versão 2020.3.1.f1. [S.l.]: Unity Technologies, 2022. Disponível em: <https://unity.com/pt>. Acesso em: 04 de junho de 2022.

XYKYO. Usando um módulo L298N com Arduino. 2022. Disponível em: <https://www.aranacorp.com/pt/usando-um-modulo-l298n-com-arduino/> . Acesso em: 23 de junho de 2022.