

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

MÁRIO ALEXANDRE RODRIGUES

**DESENVOLVIMENTO DE UM ROBÔ MÓVEL DIFERENCIAL DIDÁTICO PARA
O ENSINO DE ROBÓTICA**

PATO BRANCO

2024

MÁRIO ALEXANDRE RODRIGUES

**DESENVOLVIMENTO DE UM ROBÔ MÓVEL DIFERENCIAL DIDÁTICO PARA
O ENSINO DE ROBÓTICA**

Development of a didactic differential mobile robot for teaching robotics

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia da Computação do Curso de Bacharelado em Engenharia da Computação da Universidade Tecnológica Federal do Paraná.

Orientador: Prof^a. Dr^a. Kathya Silvia Collazos Linares

Coorientador: Prof. Dr. Jeferson José de Lima

PATO BRANCO

2024



[4.0 Internacional](#)

Esta licença permite compartilhamento, remixe, adaptação e criação a partir do trabalho, mesmo para fins comerciais, desde que sejam atribuídos créditos ao(s) autor(es). Conteúdos elaborados por terceiros, citados e referenciados nesta obra não são cobertos pela licença.

MÁRIO ALEXANDRE RODRIGUES

**DESENVOLVIMENTO DE UM ROBÔ MÓVEL DIFERENCIAL DIDÁTICO PARA
O ENSINO DE ROBÓTICA**

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia da Computação do Curso de Bacharelado em Engenharia da Computação da Universidade Tecnológica Federal do Paraná.

Data de aprovação:

Jorge Luiz Roel Ortiz
doutorado
Universidade Tecnológica Federal do Paraná

Luciente de Oliveira Marin
doutorado
Universidade Tecnológica Federal do Paraná

PATO BRANCO
2024

Dedico esse trabalho à minha mãe, Silene, ao
meu pai, Clodoaldo e à minha parceira,
Marcela, por todo o apoio e encorajamento.

AGRADECIMENTOS

Agradeço a minha orientadora Profa. Dra. Kathya Silvia Collazos Linares e ao meu co-orientador Prof. Dr. Jeferson José de Lima, pela sabedoria que me guiou nessa trajetória. A todo o centro acadêmico do curso de engenharia de computação, pela cooperação. Também gostaria de agradecer a minha família, pois, sem seu apoio, seria impossível vencer esse desafio.

RESUMO

A robótica educacional é uma área que tem sido abordada desde a década de 70 no Brasil. Introduzida por Seymour Papert e Marvin Minsky, a robótica educacional busca imergir alunos e professores no contexto prático da robótica, facilitando o aprendizado de conceitos como pensamento lógico e resolução de problemas. Um robô móvel diferencial, ou robô de tração diferencial, é um veículo versátil modular que possui rodas sobre um mesmo eixo (virtual) que podem ser acionadas de forma independente. Este trabalho tem como objetivo desenvolver um robô móvel diferencial didático que possa ser utilizado nas disciplinas do curso de engenharia de computação da universidade. Além do robô, também é proposto o desenvolvimento de um algoritmo exemplo para que o veículo autônomo possa desviar de obstáculos, e também a criação de uma biblioteca com funções intermediárias para facilitar o entendimento e a capacidade dos estudantes do curso, possibilitando que estes criem seus próprios programas para o robô móvel.

Palavras-chave: robô móvel diferencial; ensino; robótica educacional.

ABSTRACT

Educational robotics is an area that has been addressed since 1970s in Brazil. Introduced by Seymour Papert and Marvin Minsky, educational robotics seeks to immerse students and teachers in the practical context of robotics, facilitating the learning of concepts such as logical thinking and problem solving. A differential mobile robot, or differential traction robot, is a versatile modular vehicle that has wheels on the same (virtual) axis that can be driven independently. This work aims to develop a didactic differential mobile robot that can be used by the disciplines of the university's computer engineering course. In addition to the robot, it is also proposed the development of an example algorithm so that the autonomous vehicle can avoid obstacles, and also the creation of a library with intermediate functions to facilitate the understanding and ability of course students to design their own programs for the mobile robot.

Keywords: differential mobile robot; teaching; educational robotics.

LISTA DE FIGURAS

Figura 1 – Chassi Robô Móvel – 2WD	11
Figura 2 – Estrutura básica de um robô móvel diferencial	15
Figura 3 – Robô móvel diferencial sobre um plano cartesiano	16
Figura 4 – Funcionamento de um sensor de ultrassom	17
Figura 5 – Representação de um Encoder Óptico	17
Figura 6 – Representação de um <i>Encoder Óptico Incremental</i>	18
Figura 7 – Sinais Gerados pelo <i>encoder</i> óptico de acordo com o sentido de rotação do disco: (a) Sentido Anti-Horário, (b) Sentido Horário	18
Figura 8 – Exemplos de caminhos que o robô irá percorrer.	19
Figura 9 – Representação de uma ponte H.	20
Figura 10 – Representação do uso de uma ponte H.	20
Figura 11 – Kit Curumin	21
Figura 12 – EDUBOT-V2	21
Figura 13 – Robô Tesla	22
Figura 14 – Arduino UNO R3	24
Figura 15 – Arduino IDE	25
Figura 16 – Módulo Ponte H L298N.	26
Figura 17 – Chave óptica do <i>encoder</i>	27
Figura 18 – Sensor de Ultrassom HC-SR04	28
Figura 19 – Projeto do robô móvel com pinagem do Arduino.	28
Figura 20 – Protótipo do robô móvel desenvolvido.	29
Figura 21 – Diagrama da rotação do robô móvel com base no ponto fixo (pivô) definido	32
Figura 22 – Acesso aos algoritmos exemplos da biblioteca desenvolvida após importação	35
Figura 23 – Protótipo do robô móvel diferencial percorrendo o caminho evitando a colisão.	38
Figura 24 – Monitor serial do Arduino Ambiente Integrado de Desenvolvimento, do inglês <i>Integrated Development Environment</i> (IDE) durante a execução do algoritmo <i>wallTracker()</i>	39

LISTA DE TABELAS

Tabela 1 – Relação dos materiais utilizados para construção do robô móvel	23
Tabela 2 – Pinagem da Ponte H L298N	26
Tabela 3 – Mapeamento das portas digitais (D^*) e analógicas (A^*) do Arduino.	29
Tabela 4 – Experimentos realizados com o protótipo.	36

LISTA DE ABREVIATURAS E SIGLAS

Abreviaturas

CC	Corrente Contínua
----	-------------------

Siglas

IDE	Ambiente Integrado de Desenvolvimento, do inglês <i>Integrated Development Environment</i>
ISR	Rotina de Interrupção de Serviço, do inglês <i>Interrupt Service Routine</i>
LED	Díodo Emissor de Luz, do inglês <i>Light Emitting Diode</i>
MIT	Instituto de Tecnologia de Massachusetts, do inglês <i>Massachusetts Institute of Technology</i>
PID	Proporcional Integral Derivativo
POO	Programação Orientada a Objetos
PWM	Modulação de Largura de Pulso, do inglês <i>Pulse Width Modulation</i>
RE	Robótica Educacional
UNICAMP	Universidade Estadual de Campinas
UTFPR	Universidade Tecnológica Federal do Paraná

SUMÁRIO

1	INTRODUÇÃO	11
1.1	Considerações iniciais	11
1.2	Objetivos	12
1.2.1	Objetivo geral	12
1.2.2	Objetivos específicos	12
1.3	Justificativa	13
1.4	Estrutura do trabalho	13
2	REFERENCIAL TEÓRICO	15
2.1	Modelo Cinemático	15
2.2	Sensores	16
2.2.1	Sensor de Ultrassom	16
2.2.2	<i>Encoder</i> Óptico	17
2.3	Microcontrolador	18
2.4	Driver	19
2.5	Trabalhos Similares	20
3	PROTÓTIPO DE UM ROBÔ MÓVEL DIFERENCIAL	23
3.1	Escopo do Protótipo	23
3.2	Materiais	23
3.3	Estrutura Física do Protótipo	24
3.3.1	Arduino	24
3.3.2	Ponte H L298N	25
3.3.3	Sensor Óptico	27
3.3.4	Sensor de Ultrassom HC-SR04	27
3.4	Modelo Esquemático do Protótipo	27
3.5	Algoritmos Para o Protótipo	29
3.5.1	Funções <i>incrementRightCounter()</i> e <i>incrementLeftCounter()</i>	30
3.5.2	Função <i>setupConfig()</i>	30
3.5.3	Função <i>move()</i>	30
3.5.4	Função <i>stop()</i>	31
3.5.5	Função <i>rotate()</i>	31

3.5.6	Funções <i>getRightMotorRPM()</i> e <i>getLeftMotorRPM()</i>	33
3.5.7	Funções <i>getRightMotorDistance()</i> e <i>getLeftMotorDistance()</i>	33
3.5.8	Algoritmo <i>wallTracker()</i>	33
3.6	Disponibilização do Protótipo	34
4	RESULTADOS	36
5	CONCLUSÃO E TRABALHOS FUTUROS	40
	REFERÊNCIAS	41

1 INTRODUÇÃO

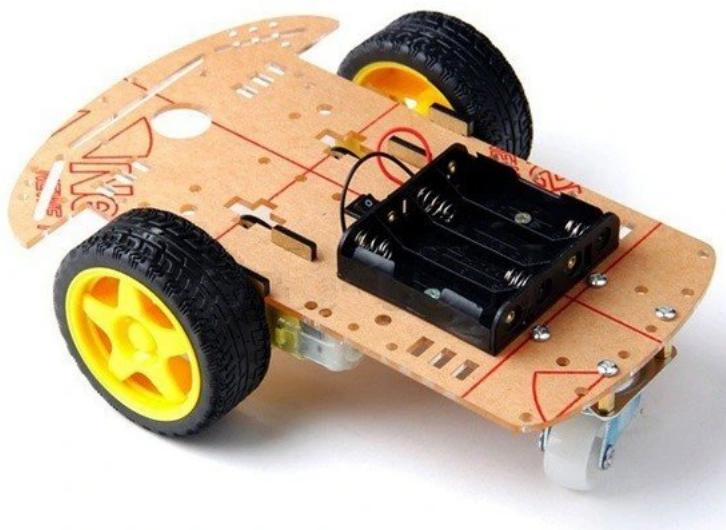
Esta seção busca elucidar sobre as motivações que levaram ao desenvolvimento da proposta, bem como apresentar seus objetivos e justificativas.

1.1 Considerações iniciais

A robótica é uma área em constante expansão, com grande potencial para transformar a forma como são realizadas diversas tarefas em setores multidisciplinares. Nesse contexto, o ensino de robótica se mostra cada vez mais relevante, sendo uma forma de estimular o interesse dos estudantes pela ciência, tecnologia e engenharia, visando o desenvolvimento científico ou para prepará-los para as demandas da indústria.

O robô móvel diferencial é um modelo versátil e com diversas aplicações possíveis. Esse modelo, também conhecido como robô móvel com tração diferencial, possui rodas sobre um mesmo eixo (virtual) que podem acionadas de forma independente. A Figura 1 mostra o chassi de um robô móvel diferencial que pode ser facilmente adquirido no mercado.

Figura 1 – Chassi Robô Móvel – 2WD



Fonte: Silício (2023).

A simplicidade e popularidade desse tipo de robô facilita sua disseminação como veículo de prova para diversos algoritmos ou estratégias. Por exemplo, (BIESEK, 2016) utilizou um robô móvel como protótipo para desenvolvimento de um algoritmo de estacionamento autônomo. Já (CARBONERA, 2021) realiza uma análise sobre navegação em robôs móveis por arbitragem e

fusão em arquiteturas comportamentais. Em adendo, (WELTER, 2022) utiliza de aprendizado por reforço profundo para navegação de veículos autoguiados.

Tais abordagens, desenvolvidas pela própria Universidade Tecnológica Federal do Paraná (UTFPR), permitem explorar conceitos de cinemática, controle de movimento, comunicação entre sensores e microcontrolador, tomada de decisão e aprendizado de máquina. A utilização de um robô móvel como objeto auxiliar de ensino prático pode tornar o aprendizado dos estudantes mais prazeroso, divertido e interessante, permitindo melhor entendimento do conteúdo abordado.

1.2 Objetivos

A seguir serão descritos o objetivo geral, aquele que é desejado ao final do trabalho, e os objetivos específicos, os intermédios que deverão ser alcançados para que se possa chegar ao objetivo geral.

1.2.1 Objetivo geral

Construir um robô móvel diferencial didático que possa ser utilizado para o ensino de robótica nas disciplinas do curso de engenharia de computação da universidade.

1.2.2 Objetivos específicos

- Projetar o robô móvel diferencial.
- Desenvolver uma biblioteca personalizada com funções que auxiliem o aluno na obtenção de dados dos sensores, definição de ações dos atuadores, bem como demais funções necessárias para o desenvolvimento de algoritmos para o robô móvel.
- Implementar um algoritmo de desvio de obstáculos que servirá como código exemplo para implementações individuais dos estudantes.
- Realizar testes e experimentos com o robô móvel diferencial, verificando seu desempenho e funcionalidade em diferentes cenários e situações.
- Documentar todo o processo de desenvolvimento de forma didática do robô móvel diferencial em um repositório *git*, permitindo assim, fácil acesso e oportunidade de replicação do projeto por qualquer um.

1.3 Justificativa

Para um ensino mais eficaz de robótica, é importante que os estudantes possam experimentar e explorar na prática os conceitos teóricos aprendidos. Nesse sentido, o desenvolvimento de um robô móvel diferencial didático é uma alternativa interessante para proporcionar aos estudantes uma experiência prática com um equipamento versátil de fácil montagem e programação.

Segundo Moraes (1993), em 1975, a Universidade Estadual de Campinas (UNICAMP) foi visitada por Seymour Papert e Marvin Minsky. No ano seguinte, professores da UNICAMP visitaram o Laboratório do Instituto de Tecnologia de Massachusetts, do inglês *Massachusetts Institute of Technology* (MIT) em Estados Unidos de América e quando voltaram, iniciaram a investigar o uso de computadores em educação, utilizando a linguagem LOGO, a partir da criação de um grupo interdisciplinar que envolveu especialistas das áreas de computação, linguística e psicologia educacional. Desde então, diversos trabalhos vieram se desenvolvendo na Robótica Educacional (RE), dentre eles, por exemplo: (ZIQUEU; RAMOS; NETTO, 2014), utilizou o *kit* Curumin para facilitar o aprendizado de estudantes do curso técnico de mecatrônica; (RUBIO-TAMAYO; JR.; HENRIQUES, 2014), com o desenvolvimento do protótipo EDUBOT-V2, um robô móvel voltado para ensino de robótica na disciplina de mecatrônica; (SILVA, 2016), que detalha o desenvolvimento do *kit* Tesla para ensino interdisciplinar de robótica em cursos de engenharia e computação.

Por se tratar de um desenvolvimento para uso didático, o protótipo proposto nesse trabalho permitirá ao aluno o acesso aos componentes reais, de modo que este poderá aplicar seus conhecimentos de programação sobre objetos físicos, não se limitando ao contexto de simuladores.

Ao inserir objetos de estudo prático durante toda a graduação, o aluno torna-se mais familiarizado com a visão geral do que a profissão escolhida o proporcionará no futuro. Para o curso de Engenharia de Computação isso não se limita apenas ao *hardware*, ou o robô, mas também ao *software*, o algoritmo embarcado.

Dessa forma, o desenvolvimento de uma biblioteca para mediar a comunicação do algoritmo desenvolvido e o *hardware* torna-se importante para manter uma integridade e coerência do algoritmo e para facilitar o aprendizado dos estudantes e mantê-los motivados a entender os meandros da robótica.

1.4 Estrutura do trabalho

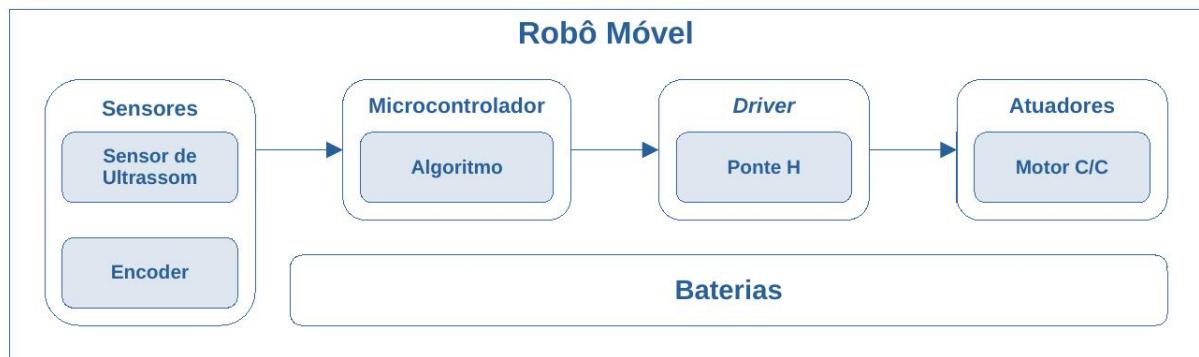
O presente trabalho está estruturado em capítulos. O atual, e primeiro, contém a introdução, justificativa e os objetivos do trabalho. A seguir, o Capítulo 2 apresenta o referencial teórico sobre o qual este projeto se embasa. O Capítulo 3 descreve o protótipo desenvolvido.

Os resultados são apresentados no Capítulo 4 e o Capítulo 5 contém as conclusões e trabalhos futuros.

2 REFERENCIAL TEÓRICO

O robô móvel diferencial é um modelo versátil, que atende a vários propósitos. Este robô é comumente utilizado como modelo de prova para algoritmos que envolvam interação com o mundo real. Por ser um agente modular, é possível personalizá-lo conforme necessário, mantendo a estrutura original. A Figura 2 apresenta a estrutura básica de um robô móvel, com exemplos de componentes utilizados em cada módulo. Essa modulação irá nortear tanto o seccionamento deste capítulo, quanto o desenvolvimento de todo o projeto.

Figura 2 – Estrutura básica de um robô móvel diferencial



Fonte: Autoria própria (2024).

2.1 Modelo Cinemático

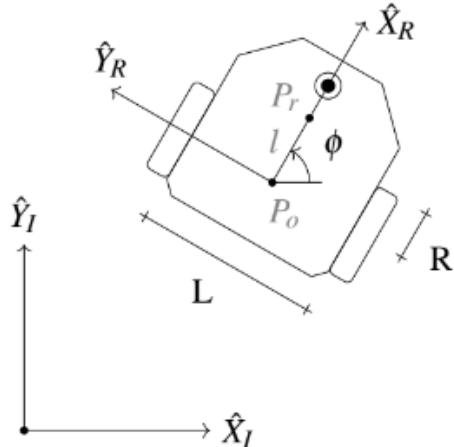
De acordo com Sasaki (2012), robôs móveis diferenciais, ou robôs móveis com tração diferencial, são agentes que possuem rodas sobre um mesmo eixo (virtual), porém acionadas de forma independente. Essa configuração permite que o robô consiga girar sobre seu próprio eixo.

A cinemática é a área da Física que estuda o movimento dos corpos. Em robótica móvel a cinemática estabelece relações entre o deslocamento (locomoção) e a atuação a ele imposta. (LIMA, 2023)

Com base na representação do robô móvel expressa na Figura 3, Carbonera (2021) faz uma análise sobre do modelo cinemático de um robô uniciclo para um robô diferencial, concluindo com as equações Equação 1 e Equação 2 para o modelo cinemático deste último, onde v representa a velocidade linear do robô; ω é a velocidade angular; R representa o raio da roda do veículo; L é a distância entre as rodas, ou a largura do robô móvel; ω_l e ω_r representam as velocidades angulares das rodas esquerda e direita, respectivamente.

$$\omega_l = \frac{2v - L\omega}{2R} \quad \omega_r = \frac{2v + L\omega}{2R} \quad (1)$$

Figura 3 – Robô móvel diferencial sobre um plano cartesiano



Fonte: Adaptado de Carbonera (2021).

$$v = \frac{R}{2}(\omega_l + \omega_r)\omega = \frac{R}{L}(\omega_r - \omega_l) \quad (2)$$

2.2 Sensores

Segundo Balbinot (2019), sensores são dispositivos que recebem um estímulo físico ou químico de certa maneira, produzindo um sinal que pode ser transformado em outra grandeza física para fins de medição e/ou monitoramento. Em outras palavras, fazendo uma analogia com o corpo humano, sensores são como os cinco sentidos, captando informações externas ao corpo para que possam ser processadas internamente.

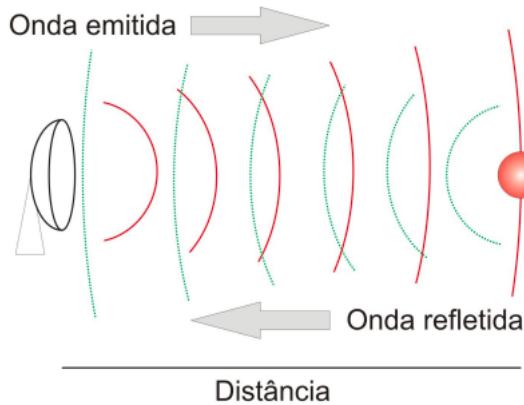
Os sensores podem ser classificados como um tipo de transdutor. Um transdutor é um componente que transforma um tipo de energia em outro. Um motor, por exemplo, é um tipo de transdutor, pois converte energia química ou elétrica em energia mecânica. (PATSKO, 2006a)

2.2.1 Sensor de Ultrassom

De acordo com Patsko (2006a), sensores baseados em ultrassom (sonar) são componentes comumente utilizados para medir a distância de objetos. Ao emitir um pulso sonoro, é possível calcular o tempo que leva para que ele seja refletido e retorne ao sensor (Figura 4). Como a velocidade do pulso emitido no meio é conhecida, é possível então calcular a distância entre o sensor e o objeto cujo pulso foi refletido.

Uma vez realizada a emissão do pulso, a distância D do sensor ao objeto pode ser calculada como sendo a metade da multiplicação do tempo de trânsito do pulso t_t pela velocidade do som no meio $c \approx 1200Km/h$ (PATSKO, 2006a), como mostra a equação Equação 3.

Figura 4 – Funcionamento de um sensor de ultrassom



Fonte: Patsko (2006a).

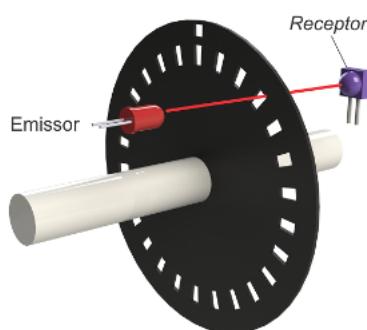
$$D = \frac{c \cdot t_t}{2} \quad (3)$$

2.2.2 Encoder Óptico

De acordo com Almeida (2017), os *encoders* ópticos, também conhecidos como sensores de deslocamento ou sensores odometrícios, são sensores capazes de quantizar distâncias, controlar velocidades, medir ângulos, número de rotações, realizar posicionamentos, rotacionar braços robóticos, dentre outras funcionalidades. O *encoder* óptico é composto basicamente por um componente emissor (Diodo Emissor de Luz, do inglês *Light Emitting Diode* (LED)), um receptor (fotodetector) e um disco opaco com aberturas igualmente espaçadas, assim como mostra a Figura 5.

Este disco, quando fixo ao eixo da roda do veículo, gera um sinal de saída na forma de onda quadrada, ao bloquear e desbloquear o feixe de luz emitido pelo LED para o fotodetector. Logo, é chamado de resolução do *encoder*, ou *clock*, a quantidade de ondas quadradas geradas em uma volta completa, que equivale a quantidade de aberturas no disco.

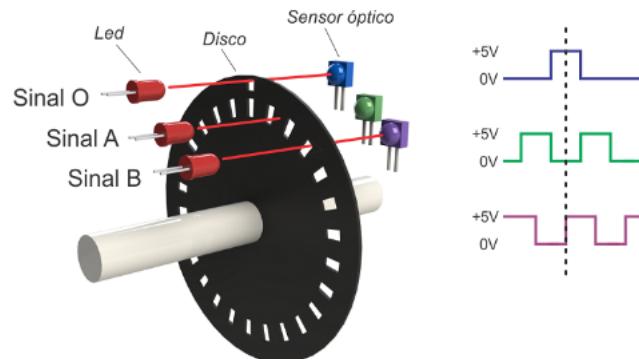
Figura 5 – Representação de um Encoder Óptico



Fonte: Almeida (2017).

Almeida (2017) explica que estes sensores são divididos em duas categorias com relação aos sinais de saída: incremental e absoluto. O *encoder* óptico incremental, Figura 6, indica a posição relativa ao ponto de ativação. Já o *encoder* absoluto sempre indica a posição absoluta através de saídas digitais mais complexas compostas de códigos binários. Esse modelo permite saber a posição sem necessidade de sincronização do sensor. Devido a baixa complexidade e preço do modelo incremental, este é o mais comum no mercado.

Figura 6 – Representação de um *Encoder* Óptico Incremental

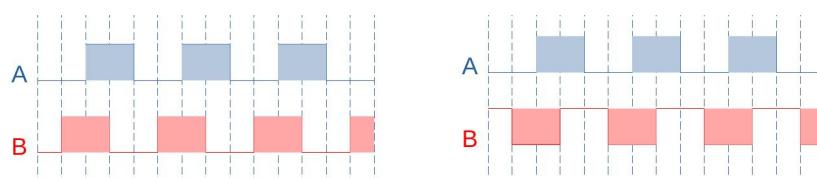


Fonte: Almeida (2017).

Segundo Almeida (2017) , o *encoder* incremental gera 3 sinais distintos de saída. O primeiro sinal (A) gera pulsos de acordo com o giro do disco. O segundo sinal (B) equivale ao primeiro, mas com uma defasagem de $\pm 90^\circ$. Essa defasagem possibilita determinar o sentido da rotação do disco (Figura 7). Quando B está defasado em $+90^\circ$, durante a borda de subida de A, B se encontra como 1, representando o sentido anti-horário. Já, quando quando B está defasado em -90° , durante a borda de subida de A, B aparece como 0, implicando o sentido horário de rotação. Por fim, o último sinal (O), indica o início de uma revolução.

Figura 7 – Sinais Gerados pelo *encoder* óptico de acordo com o sentido de rotação do disco: (a) Sentido Anti-Horário, (b) Sentido Horário

(a) Defasagem do sinal B de $+90^\circ$ (b) Defasagem do sinal B de -90°



Fonte: Adaptado de Almeida (2017).

2.3 Microcontrolador

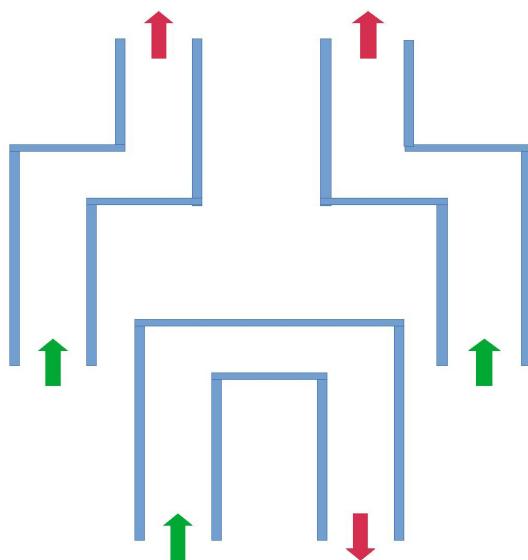
Cardoso (2020) define um microcontrolador como um único circuito integrado composto por um núcleo de processador, memórias voláteis e não voláteis e diversos periféricos para

transferência de dados. Ou seja, ele nada mais é que um computador compacto capaz de realizar diversas tarefas de forma eficaz.

Mantendo a analogia ao corpo humano, um microcontrolador pode ser considerado o cérebro do robô. É nele onde todos os cálculos serão feitos e as ações serão decididas. O módulo do microcontrolador, exposto na Figura 2 é composto majoritariamente pelo algoritmo que o robô móvel executa para realizar o conjunto de tarefas que lhe é imposto.

Auxiliado do modelo apresentado na seção 2.1, será desenvolvida uma estratégia de desvio de obstáculos simples, uma vez que este algoritmo servirá para os estudantes apenas como um exemplo de implementação para o robô desenvolvido. A Figura 8 apresenta alguns exemplos de caminhos que o protótipo a ser desenvolvido neste trabalho deverá percorrer.

Figura 8 – Exemplos de caminhos que o robô irá percorrer.



Fonte: Autoria própria (2024).

2.4 Driver

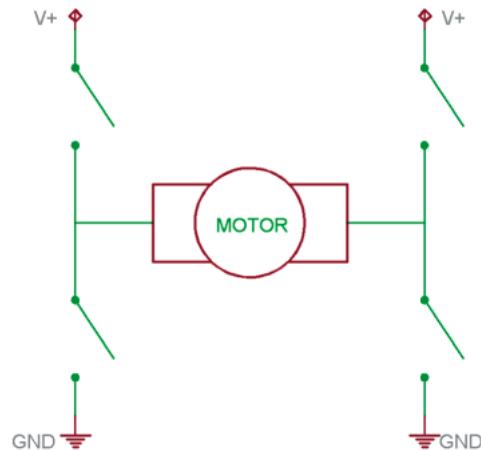
O módulo *driver* se refere a componentes intermediários necessários para o acionamento dos atuadores. Um exemplo desses componentes é a Ponte H.

Segundo Patsko (2006b), a ponte H é um dos circuitos mais importantes no tocante a elaboração de sistemas automatizados. Trata-se de um circuito utilizado para o acionamento de motores Corrente Contínua (CC) através de sinais digitais gerados por um microcontrolador.

Um motor CC, quando conectado diretamente a uma bateria, gira em uma velocidade constante em uma única direção. Para que este mesmo motor gire no sentido inverso, é necessário inverter seus polos. Uma ponte H básica é composta por 4 chaves mecânicas ou eletrônicas posicionadas formando a letra “H”, sendo cada chave alocada em um extremo e o motor centralizado, assim como mostra a Figura 9.

Logo, para que o motor seja acionado, basta acionar um par de chaves diagonais opostas, de forma que a corrente flua de forma direta, ou seja, do polo positivo para o polo negativo do motor. Já para provocar um fluxo reverso da corrente, é necessário acionar apenas o outro par de chaves. Ambos os fluxos são representados na Figura 10.

Figura 9 – Representação de uma ponte H.

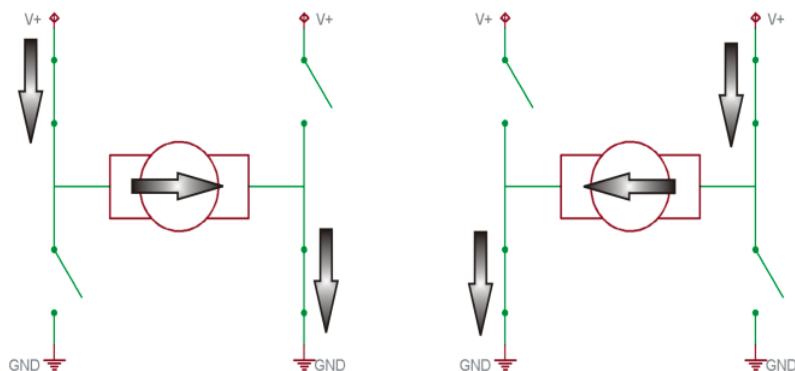


Fonte: Patsko (2006b).

Figura 10 – Representação do uso de uma ponte H.

(a) Fluxo direto

(b) Fluxo reverso



Fonte: Patsko (2006b).

2.5 Trabalhos Similares

De acordo com Zaqueu, Ramos e Netto (2014), o conceito de RE chegou ao Brasil por volta de 1975, com Seymour Papert e Marvin Minsky, que fundamentaram os conceitos da linguagem Logo. Após isso, a robótica educacional tem ganhado força na educação de crianças, jovens e adultos. Os autores também ressaltam que as tecnologias estão cada vez mais presentes no cotidiano dos estudantes, professores e sociedade em geral e que, com os constantes

avanços tecnológicos que estão ocorrendo, as práticas e metodologias adotadas, necessitam de mudanças para adaptar e acompanhar esta evolução natural.

Dessa forma, os autores apresentam uma proposta para ensino de linguagem de programação para o curso técnico em mecatrônica. Ao analisar a dificuldade que os estudantes do curso possuíam em relação ao raciocínio lógico e identificação de problemas, muitas vezes oriundas de um ensino básico deficitário. Zaqueu, Ramos e Netto (2014) utilizaram o kit Curumin (Figura 11), desenvolvido pela empresa Xbot, incluindo uma apostila com atividades que reforçavam conceitos de programação, lógica e resolução de problemas. Vale ressaltar que ao elaborar esse material, o objetivo dos autores era fazer com que o aluno fosse imerso no ambiente vivenciado, sendo capaz de interagir e se envolver com o problema proposto.

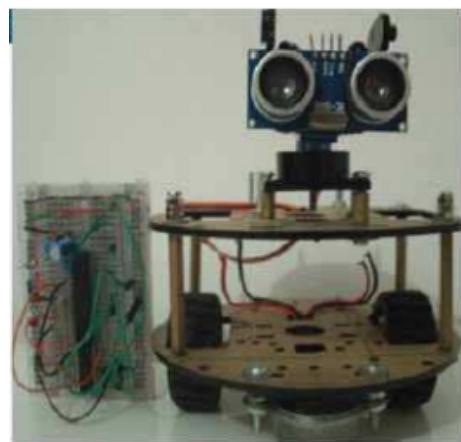
Figura 11 – Kit Curumin



Fonte: Zaqueu, Ramos e Netto (2014).

Ao final dessa experiência, Zaqueu, Ramos e Netto (2014) concluíram que o uso desta metodologia aproximou a relação estabelecida entre aluno e professor, rompendo barreiras que poderiam comprometer o processo de construção do conhecimento.

Figura 12 – EDUBOT-V2



Fonte: Rubio-Tamayo, Jr. e Henriques (2014).

Rubio-Tamayo, Jr. e Henriques (2014) apresentaram o protótipo EDUBOT-V2, 12, um robô móvel para ensino de robótica para a disciplina de mecatrônica. Um diferencial do EDUBOT-V2 é que seu sistema foi desenvolvido utilizando o paradigma da Programação Orientada a Objetos (POO), em contraste aos outros projetos apresentados neste capítulo que im-

plementam o paradigma de programação estruturada. Assim como explicam os autores, a POO têm sido utilizada atualmente como uma alternativa que permite contornar algumas deficiências da análise estruturada, além de fornecer características que permitem melhor entendimento desse modelo e a possibilidade de encapsulamento dos dados, facilitando sua reutilização.

Em 2016, Silva (2016) desenvolveu um protótipo do robô móvel Tesla, um robô de baixo custo interdisciplinar em cursos de engenharia e computação. Além disso, Silva (2016) fez uma análise comparativa entre *kits* de RE disponíveis no mercado. Baseando-se nos conceitos de construtivismo, de Piaget, e de construcionismo, de Papert, o autor ressaltou que o *kit* Tesla é uma ferramenta construída pelo aluno, onde o aprendizado se dá pela execução prática. O protótipo desenvolvido (Figura 13) é composto por duas rodas, uma garra articulada, módulo de potência, de controle, além de diversos sensores.

Figura 13 – Robô Tesla



Fonte: Silva (2016).

Em comparação com os outros *kits* de robótica disponíveis no mercado, o *kit* Tesla se destaca por seu baixo custo, cerca de R\$300,00 (trezentos reais), sua arquitetura aberta, software livre além da diversidade de componentes que podem ser utilizados pelos estudantes para desenvolver seus próprios algoritmos. (SILVA, 2016)

3 PROTÓTIPO DE UM ROBÔ MÓVEL DIFERENCIAL

Neste capítulo será descrito o escopo do protótipo desenvolvido, os materiais utilizados, a implementação física, os algoritmos (funções) e a forma de disponibilização.

3.1 Escopo do Protótipo

O protótipo desenvolvido deve ser capaz movimentar-se por caminhos predeterminados (mostrados na Figura Figura 8). Nestes caminhos, para evitar a colisão com obstáculos, ele deve realizar a conversão tanto em sentido horário quanto anti-horário. Além disso, o protótipo deve permitir o acesso à informações dos sensores, bem como controle dos atuadores, através de funções personalizadas. Por fim, a sua documentação deve estar clara e objetiva com o intuito de auxiliar os estudantes a utilizarem o sistema desenvolvido ou até criarem sua própria versão.

3.2 Materiais

O desenvolvimento do protótipo tem duas partes, a construção física (*hardware*) e a construção abstrata (*textitsoftware*) para o funcionamento.

Para o desenvolvimento dos algoritmos e biblioteca de funções utilizou-se a linguagem de programação C++ e um laptop com *kernel* Linux e sistema operacional Manjaro 64 *bits*, processador Intel Core 7500U com *clock* de 2.5 GHz e 8GB de memória RAM DDR4.

Na Tabela 1 detalham-se as quantidades dos componentes utilizados para a construção física do protótipo.

Tabela 1 – Relação dos materiais utilizados para construção do robô móvel

Material	Quantidade
Arduino UNO R3	1.00
Módulo Ponte H L298N	1.00
Sensor Ultrassom HC-SR04	1.00
Chassi Robô Móvel - 2WD	1.00
Encoder óptico – Chave Óptica	2.00
Disco Encoder 20 Dentes	2.00
Motor CC com caixa de redução	2.00
Roda omnidirecional	1.00
Roda Hobby com pneu	2.00
Bateria LiPo 1300 mAh 7.4V	1.00
Protoboard 170 pontos	2.00
Chave interruptora	1.00

Fonte: Autoria própria (2024).

3.3 Estrutura Física do Protótipo

O modelo conta com um chassi de robô móvel com as devidas perfurações, bem como um *kit* com duas rodas, dois motores e uma roda omnidirecional, a montagem foi simples, necessitando apenas a fixação das partes em seus devidos pontos de acoplagem ao chassi. As demais peças (sensores, microcontrolador, ponte H, placas perfuradas e baterias) são dispostas no chassi conforme a finalidade desejada. Neste protótipo, esses componentes foram dispostos prioritariamente visando a melhor observação das conexões por parte do usuário.

3.3.1 Arduino

Segundo (VIDAL, 2024), Arduino é uma plataforma de desenvolvimento de projetos eletrônicos, ou de prototipagem eletrônica, composta por *hardware* e *software*. Iniciado em 2015, na Itália, o projeto tinha como propósito inicial criar uma plataforma de baixo custo, fácil de trabalhar, e que pudesse ser usada por estudantes para desenvolver seus protótipos de forma eficiente e barata.

Dentre as opções disponíveis no mercado, a placa Arduino UNO R3 (Figura 14) é, sem dúvida, a que melhor se encaixa para este protótipo. Se trata de uma versão para quem está começando a desenvolver projetos eletrônicos e possui 14 portas de entrada/saída, das quais 6 podem ser usados como Modulação de Largura de Pulso, do inglês *Pulse Width Modulation* (PWM) (VIDAL, 2024).

Figura 14 – Arduino UNO R3



Fonte: (VIDAL, 2024).

O *software* Arduino IDE, Figura 15, é um ambiente de programação que faz parte da plataforma Arduino e utiliza uma linguagem de programação baseada em C/C++, com algumas modificações. Possui um alto grau de abstração, além de um leque de bibliotecas que reduzem

consideravelmente a complexidade do microcontrolador. Isso torna a programação mais intuitiva e rápida, pois não é necessário que o desenvolvedor conheça os registradores ou detalhes de memória e a dinâmica do processador (VIDAL, 2024). O *software* está disponível para *download* em: <https://www.arduino.cc/en/software>

Figura 15 – Arduino IDE



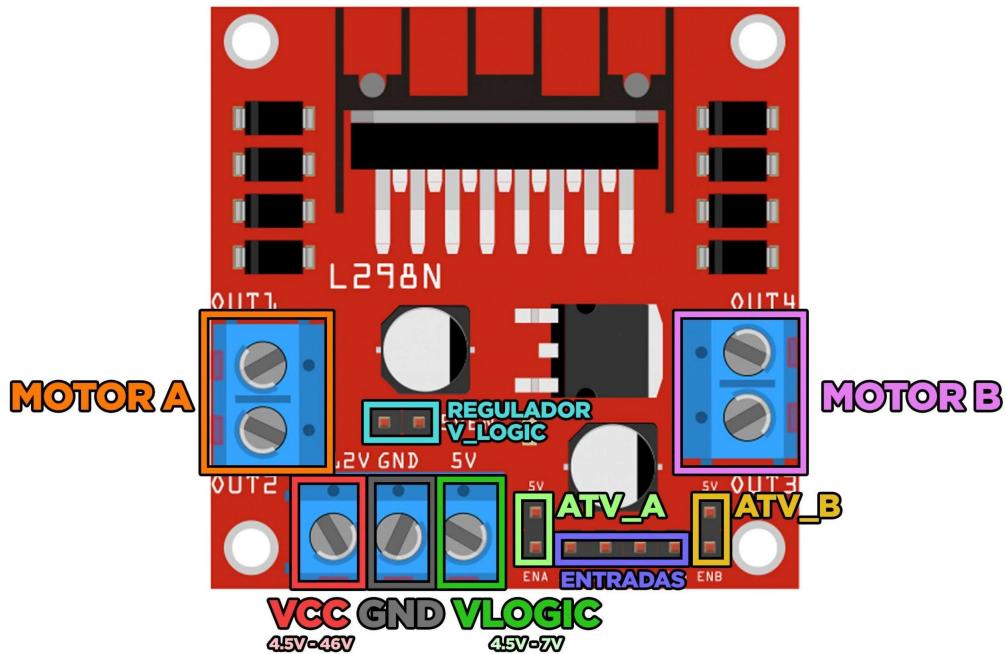
Fonte: (VIDAL, 2024).

3.3.2 Ponte H L298N

A Ponte H L298N é um módulo de controle de motor CC de dois canais. É composto por três pinos de alimentação, seis pinos de controle e quatro pinos de saída. A Figura 16 possibilita a visualização da distribuição dos pinos desse módulo. Já a Tabela 2 descreve as finalidades de cada uma das portas desse componente. Segundo (NERY, 2022), dentre as principais vantagens desse circuito, uma das principais está relacionada a ele permitir uma maior dissipação de potência em relação ao acionamento direto por portas de circuitos microprocessados, que só conseguem fornecer alguns mA de corrente, contra $2A$ por canal da ponte H L298N.

Esse circuito conta com duas entradas digitais para cada um dos motores CC ($IN1$, $IN2$ ou $IN3$, $IN4$) que possibilitam o controle do sentido de rotação desses motores através de sinais lógicos enviados pelo microcontrolador, assim como um pino de entrada adicional (ENA ou ENB) que permite controlar a velocidade de rotação dos mesmos. A tensão de operação desse

Figura 16 – Módulo Ponte H L298N.



Fonte: (NERY, 2022).

Tabela 2 – Pinagem da Ponte H L298N

Pino	Descrição
IN1	Primeira entrada digital do primeiro motor
IN2	Segunda entrada digital do primeiro motor
IN3	Primeira entrada digital do segundo motor
IN4	Segunda entrada digital do segundo motor
ENA	Regulador da tensão de saída do primeiro motor
ENB	Regulador da tensão de saída do segundo motor
OUT1	Primeira saída digital do primeiro motor
OUT2	Segunda saída digital do primeiro motor
OUT3	Primeira saída digital do segundo motor
OUT4	Segunda saída digital do segundo motor
VCC	Alimentação do módulo
GND	Alimentação do módulo
Vlogic	Alimentação lógica do módulo

Fonte: Autoria própria (2024).

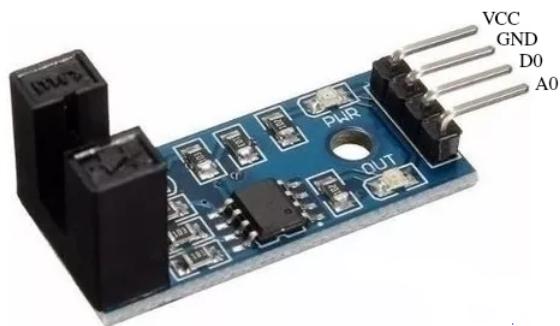
módulo (VCC) está entre 4.5V - 46V e é repassada aos motores, portanto é recomendado que seja aplicada uma tensão suportada pelos mesmos (NERY, 2022).

O pino *Vlogic* é responsável pela alimentação lógica da ponte H. Para que o *chip* L298N presente na placa consiga operar os comandos emitidos pelo microcontrolador e atuar na saída, ele deve operar como um componente lógico, e para isso precisa atuar na faixa de tensão de 4.5V - 7V. Felizmente, o circuito possui um regulador de tensão de 5V, que regula a tensão de VCC para a alimentação lógica do chip em 5V (NERY, 2022). Dessa forma, não é necessário adicionar outra fonte de tensão ao módulo apenas para alimentação lógica.

3.3.3 Sensor Óptico

Na Figura 19, os módulos dos motores estão representados com os *encoders*, isto porque estes sensores são peças dependentes. Já no protótipo, eles foram dispostos de forma que seja possível ler o disco perfurado o qual está acoplado às respectivas caixas de redução dos motores CC. Assim como mostra a Figura 17, estes sensores possuem quatro pinos cada: dois de alimentação: *VCC* e *GND*; uma saída digital: *D0*; e uma saída analógica: *A0*. Como se tratam de circuitos lógicos, com sua tensão de alimentação varia entre 3.3V - 5V, esses componentes podem ser alimentados diretamente pelo Arduino.

Figura 17 – Chave óptica do encoder



Fonte: Autoria própria (2024).

3.3.4 Sensor de Ultrassom HC-SR04

O sensor de ultrassom, utilizado prioritariamente para medir a proximidade do robô móvel com os obstáculos os quais ele deve evitar, é o sensor HC-SR04. Esse é um dos modelos mais comuns desse tipo de sensor e possui quatro pinos: dois para alimentação direta do Arduino: *VCC* e *GND*; e dois pinos de saída digital, *TRIG* e *ECHO*, que informam ao microcontrolador o momento em que a onda sonora é emitida e recebida, respectivamente. A Figura 18 apresenta esse sensor, juntamente com a disposição de seus pinos.

3.4 Modelo Esquemático do Protótipo

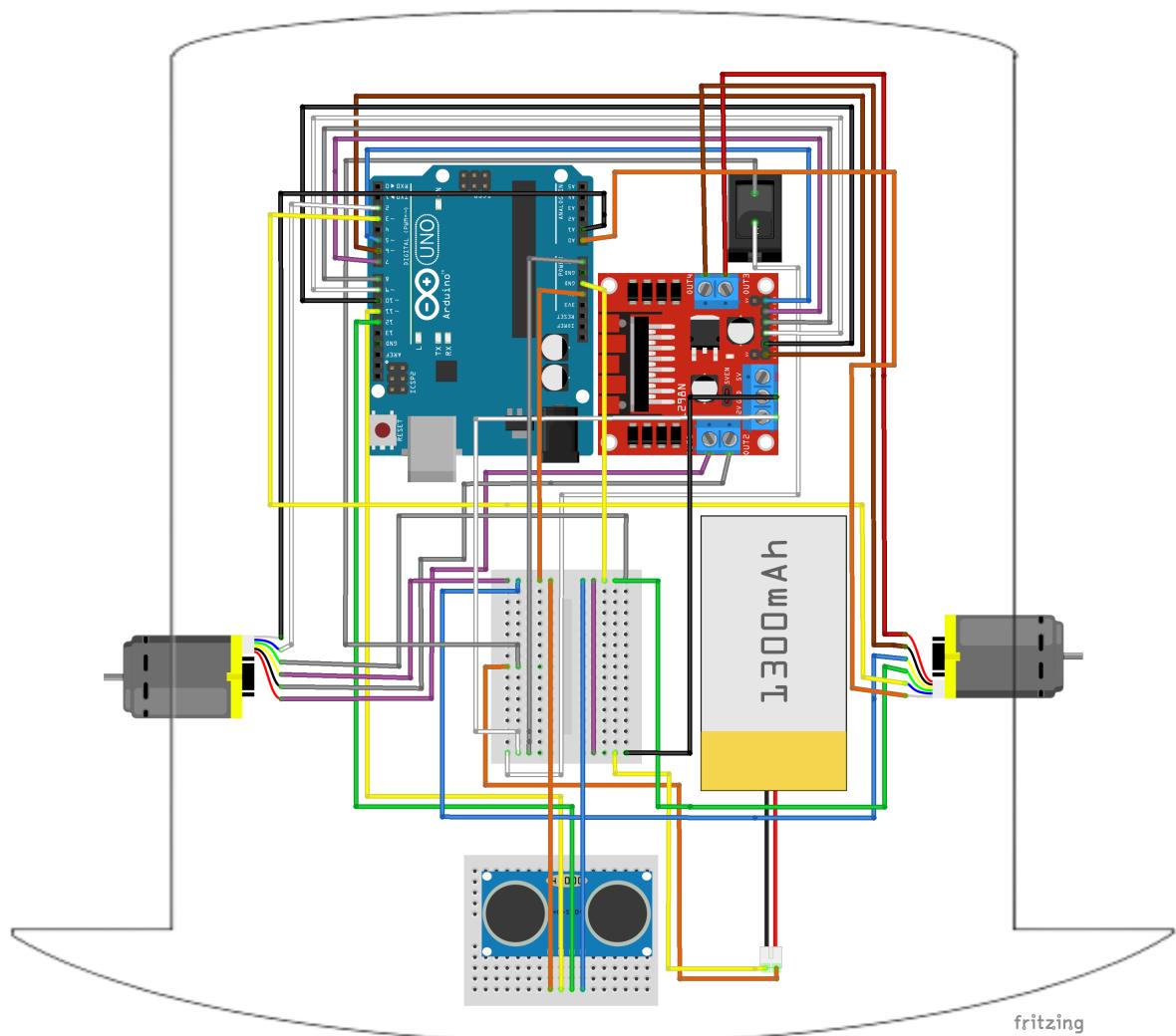
A Figura 19 apresenta o modelo esquemático desenvolvido, disposto da mesma forma que o protótipo, contornado por uma moldura de mesmo formato do chassi utilizado para que seja fácil a assimilação da disposição das peças. Já a Tabela 3 descreve o mapeamento das portas utilizadas do Arduino para os respectivos módulos de entrada/saída e também o modo de configuração (ENTRADA ou SAIDA) das mesmas.

Figura 18 – Sensor de Ultrassom HC-SR04



Fonte: Autoria própria (2024).

Figura 19 – Projeto do robô móvel com pinagem do Arduino.



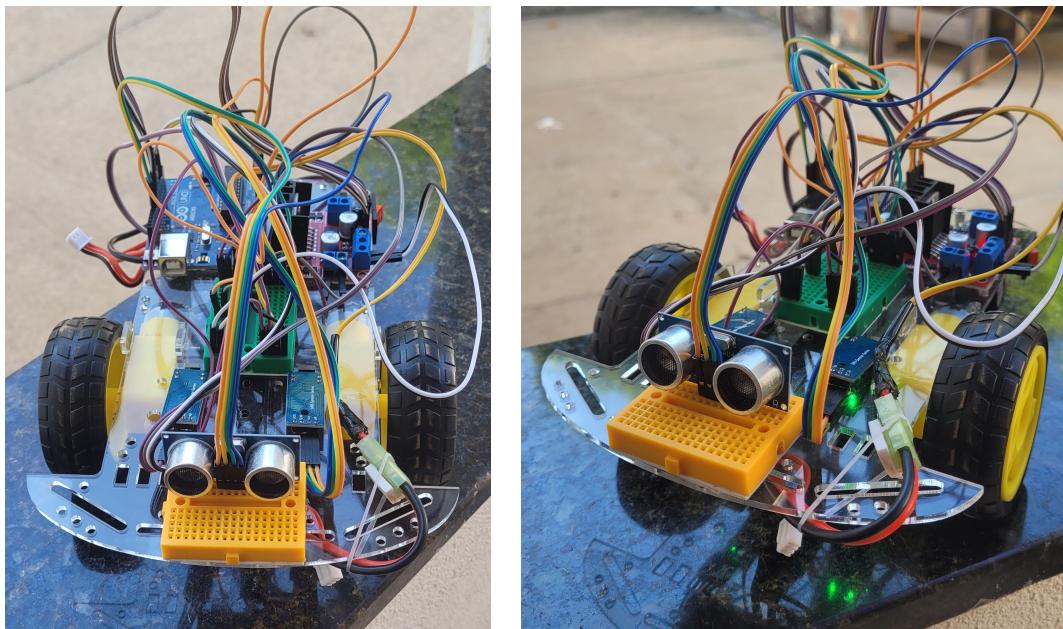
Fonte: Autoria própria (2024).

O protótipo desenvolvido neste trabalho é exibido na Figura 20. Por se tratar de um protótipo que tem por finalidade ser utilizado para auxiliar no ensino de robótica, as conexões foram deixadas visíveis para que possam ser estudadas.

Tabela 3 – Mapeamento das portas digitais (D^*) e analógicas (A^*) do Arduino.

Porta do Arduino	Porta do Componente	Componente Mapeado	I/O do Arduino
D2	D0	Encoder direito	ENTRADA
D3	D0	Encoder esquerdo	ENTRADA
D5	ENB	Ponte H L298N	SAIDA
D6	ENA	Ponte H L298N	SAIDA
D7	IN4	Ponte H L298N	SAIDA
D8	IN3	Ponte H L298N	SAIDA
D9	IN2	Ponte H L298N	SAIDA
D10	IN1	Ponte H L298N	SAIDA
D11	Trig	Sensor Ultrassônico HC-SR04	ENTRADA
D12	Echo	Sensor Ultrassônico HC-SR04	ENTRADA
A0	A0	Encoder esquerdo	N/A
A1	A0	Encoder direito	N/A
Vin	V+	Bateria LiPo 7.4V 1300 mAh	N/A
GND	V-	Bateria LiPo 7.4V 1300 mAh	N/A

Fonte: Autoria própria (2024).

Figura 20 – Protótipo do robô móvel desenvolvido.

Fonte: Autoria própria (2024).

3.5 Algoritmos Para o Protótipo

A seguir serão detalhadas as funções desenvolvidas para o funcionamento do protótipo. Todas as implementações foram realizadas utilizando a plataforma Arduino IDE, na versão 2.3.2, e podem ser encontrados no repósitorio *git* <https://github.com/MrBros/trabalho-conclusao-curso>, juntamente com exemplos de utilização e suas respectivas implementações.

3.5.1 Funções *incrementRightCounter()* e *incrementLeftCounter()*

Como dito anteriormente, os *encoders* são sensores que emitem pulsos digitais, através do pino de saída *D0*, com base na rotação do disco acoplado ao motor. Essa emissão necessita ser identificada pelo Arduino para que este possa armazenar e calcular as métricas inerentes a esse tipo de sensor, como velocidade, distância percorrida, etc.

Para essa finalidade, a plataforma Arduino possui funções que identificam sinais de interrupções, sejam internos ou externos. Essa identificação permite que o Arduino interrompa o fluxo do código principal, execute uma função especial conhecida como Rotina de Interrupção de Serviço, do inglês *Interrupt Service Routine* (ISR) e depois retorna ao fluxo original. Uma ISR é uma função que não possui nenhum parâmetro de entrada e nenhum parâmetro de saída e todas as variáveis globais manipuladas por essa função devem ser do tipo *volatile*, ou seja, podem ser modificadas sem que o programa principal saiba (SCHULTZ, 2022).

Uma interrupção externa se refere a interrupções geradas por agentes externos ao Arduino como, por exemplo, os *encoders*. Segundo (SCHULTZ, 2022), o Arduino UNO R3 possui apenas dois pinos digitais que suportam interrupções externas, os pinos 2 e 3.

Assim, as funções *incrementRightCounter()* e *incrementLeftCounter()* são duas ISRs desenvolvidas, uma para cada sensor de velocidade. Cada função incrementa variáveis globais que são utilizadas pelas próximas funções para calcular as devidas métricas desses sensores.

3.5.2 Função *setupConfig()*

Essa função tem como objetivo realizar a configuração inicial das portas do Arduino, valores iniciais das variáveis globais, inicializar a comunicação serial do microcontrolador e ativar as interrupções de borda dos *encoders*. Recebe o parâmetro de entrada *serialRate*, que determina a frequência da comunicação serial do Arduino.

3.5.3 Função *move()*

A função *move()* foi desenvolvida para facilitar a configuração das portas de entrada do módulo ponte H para acionar os dois motores na mesma direção e com a mesma velocidade, fazendo assim que o robô móvel se movimente linearmente para frente ou para trás. Essa função possui o parâmetro de entrada *direction*, que determina em qual direção o robô irá se movimentar: *forward*, para movimentar o robô para frente; *backward*, para movimentar o robô para trás.

3.5.4 Função *stop()*

Oposta à função anterior, a função *stop()* realiza a parada dos dois motores CC. Essa função não necessita de nenhum parâmetro de entrada para ser acionada.

3.5.5 Função *rotate()*

Esta, por sua vez, configura a saída da ponte H de modo a rotacionar o robô móvel tanto no sentido horário, quanto no sentido anti-horário. Possui três parâmetros de entrada:

- *direction*: define o sentido de rotação do robô móvel. Possui dois valores aceitos: *clockwise*, para sentido horário; e *counterClockwise*, para sentido anti-horário;
- *pivot*: define o centro da rotação do robô móvel. Os valores podem ser: *center*, para que o centro da rotação seja o centro do eixo virtual do robô diferencial; *left*, que torna a roda esquerda o centro da rotação; e *right*, definindo o centro como a roda direita;
- *ang*: dita o ângulo final que o robô irá rotacionar, em graus (0° a 360°), com base na posição inicial do robô e os demais parâmetros da função.

O conceito por trás dessa função se resume, em suma, a calcular o valor do arco do círculo correspondente para cada situação e compará-lo com a distância percorrida por cada motor individualmente. Sabe-se, por definição que:

$$s = r \cdot \theta \quad (4)$$

Onde: s corresponde ao arco do círculo; r é o raio do círculo; e θ é o ângulo, em radianos, correspondente ao arco.

Dessa forma, para rotacionar o robô móvel fixando o centro desse círculo em qualquer uma das rodas (Figuras 21a e 21b), primeiro é necessário encontrar o valor de s . É possível calcular esse valor substituindo as variáveis da função anterior pelo parâmetro *ang* da função e adicionando as medidas do robô:

$$s = L \cdot \left(ang \cdot \frac{\pi}{180} \right) \quad (5)$$

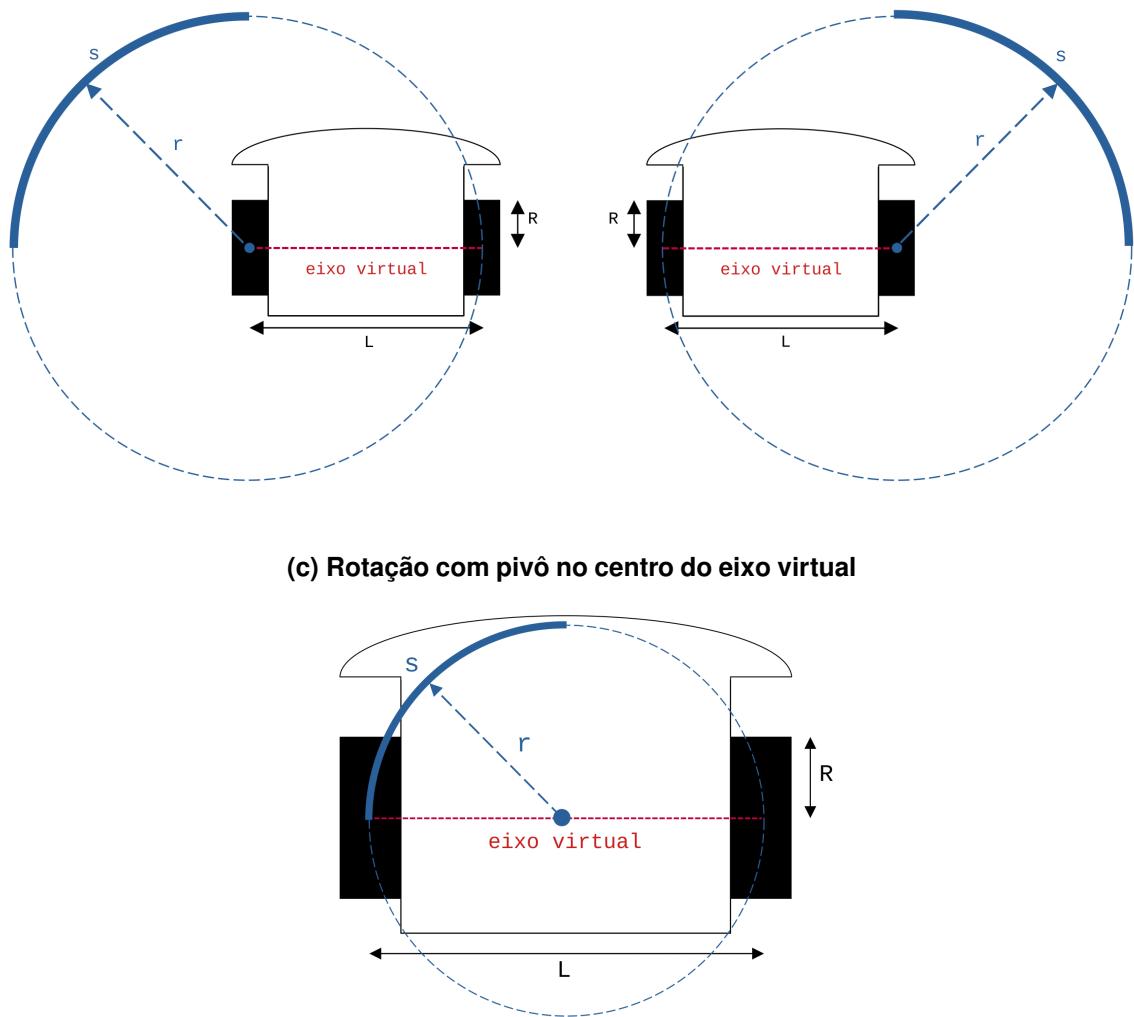
Onde: L é a distância entre as duas rodas do robô móvel; $\frac{\pi}{180}$ realiza a conversão do valor de *ang* de graus para radianos.

Já para a distância percorrida pela roda do robô, a Equação 4 ainda se mostra útil, sendo necessário apenas adicionar o número de voltas realizadas pela roda. A 6 mostra a equação utilizada para calcular a distância percorrida pela roda do robô, onde R é o raio da roda do robô e N corresponde ao número de voltas realizadas pela mesma.

$$s_{roda} = N \cdot 2\pi \cdot R \quad (6)$$

Por fim, a função ativa o motor da roda contrária ao ponto fixo, de forma que o robô rotacione no sentido desejado e realiza um *loop* que monitora, a cada iteração, a distância percorrida por essa roda, de modo que $s_{roda} < s$. Quando essa condição não é mais satisfeita, indica que o robô concluiu sua rotação e é necessário parar a rotação.

Figura 21 – Diagrama da rotação do robô móvel com base no ponto fixo (pivô) definido



Fonte: Autoria própria (2024).

Para calcular a rotação do robô móvel com o ponto fixo no centro do eixo virtual, como mostra a Figura 21c, o raio do círculo de rotação será metade do raio da equação Equação 7, pois agora as duas rodas irão ser ativadas para que o robô rotacione sobre o próprio eixo. Portanto, a equação resultante é:

$$s = \frac{L}{2} \cdot (ang \cdot \frac{\pi}{180}) \quad (7)$$

Outro ponto a ser alterado é a condição do *loop* da função, que agora monitora a distância das duas rodas, ou seja, enquanto $s_{roda\ direita} < s$ ou $s_{roda\ esquerda} < s$. Assim, a rotação do robô móvel só é paralisada quando as duas rodas percorrerem, cada uma, a distância s na orientação definida pelo parâmetro *direction*.

3.5.6 Funções *getRightMotorRPM()* e *getLeftMotorRPM()*

Uma vez que o Arduino armazena os sinais enviados pelos *encoders* através do monitoramento das interrupções de borda, é possível extrair a velocidade de rotação dos motores individualmente. As funções *getRightMotorRPM* e *getLeftMotorRPM* têm a finalidade de obter tal métrica, medida em RPM, divergindo apenas de qual motor a informação é extraída. Por se tratarem de cálculos feitos com variáveis voláteis, que são alteradas de forma assíncrona pelas interrupções de borda, é necessário realizar uma pausa nessas interrupções antes de se efetuar o cálculo. Após feito, a interrupção é ativada novamente.

A lógica por trás dessas funções consiste em obter o número de rotações em um determinado intervalo de tempo e normalizá-lo para encontrar o número de rotações em 1 minuto. A Equação 8 apresenta o cálculo utilizado por essas funções.

$$rpm = 60000 \cdot \frac{\Delta n}{\Delta t} \quad (8)$$

Onde: Δn corresponde à diferença do número de voltas realizadas pela respectiva roda entre duas chamadas da função correspondente; Δt é a diferença de tempo, em milissegundos, entre as chamadas da função.

3.5.7 Funções *getRightMotorDistance()* e *getLeftMotorDistance()*

Outra métrica interessante para se obter utilizando esses sensores é a distância percorrida pelo robô móvel. Para isso, foram implementadas as funções *getRightMotorDistance()* e *getLeftMotorDistance()*, que permitem a obtenção da distância absoluta percorrida por cada roda individualmente. Essas implementações utilizam o mesmo cálculo apresentado pela Equação 6, no entanto, a distância percorrida calculada por essas funções corresponde à distância total percorrida por cada motor individualmente desde o momento que o robô foi ativado.

3.5.8 Algoritmo *wallTracker()*

O algoritmo de desvios de obstáculos simples, apelidado de *wallTracker*, foi desenvolvido buscando não apenas exemplificar o uso das funções citadas anteriormente, mas também demonstrar e avaliar a capacidade do sistema desenvolvido nesse projeto. O código é composto por uma lógica simples, mas eficaz, que realiza a conversão devida para não colidir com o obs-

táculo, como mostra o Algoritmo 1. O robô móvel monitora, utilizando o sensor de proximidade, a distância do obstáculo. Caso a distância seja maior a distância máxima predefinida, o robô seguirá em frente. Caso contrário, o robô realiza uma parada e, logo após, realiza um pequeno recuo e converge 90° no sentido horário com a roda direita como pivô da rotação. Após isso, outra parada é realizada e o robô irá mensurar novamente a distância para o obstáculo. Caso essa distância seja menor que a distância máxima, o robô irá rotacionar 180° no sentido anti-horário com ponto fixo da rotação sendo o centro do eixo virtual do robô móvel. Feito isso, o algoritmo irá se repetir.

Algorithm 1 Algoritmo de desvio de obstáculos *wallTracker()*

```

1: enquanto verdadeiro faça
2:   distancia = mede_distancia()
3:   se distancia < DISTANCIA_MAXIMA então
4:     para();
5:     move_para_tras();
6:     para();
7:     rotaciona_sentido_horario();
8:     distancia = mede_distancia();
9:   se distancia < DISTANCIA_MAXIMA então
10:    para();
11:    move_para_tras();
12:    para();
13:    rotaciona_sentido_anti_horario();
14:  finaliza se
15: senão,
16:   move_para_frente();
17: finaliza se
18: finaliza enquanto=0

```

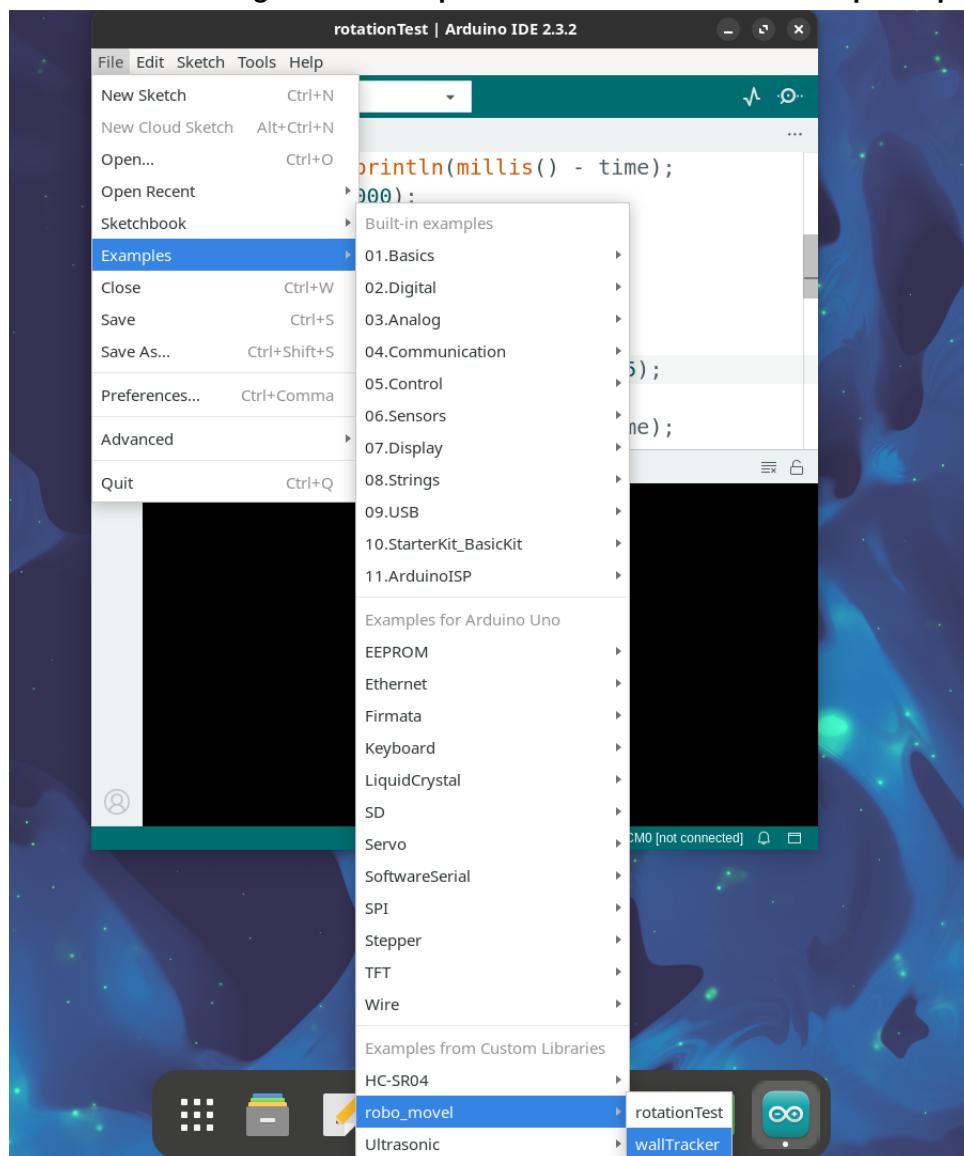
3.6 Disponibilização do Protótipo

Como dito anteriormente, o código fonte do projeto está disponível no repositório *git* <https://github.com/MrBros/trabalho-conclusao-curso>, assim como um vídeo demonstrativo com os testes realizados do protótipo e exemplos de utilizações das funções supracitadas. Para que seja possível utilizar o código desenvolvido, é necessário importá-lo na pasta de biblioteca do programa Arduino IDE. Essa pasta se encontra nos seguintes caminhos, dado o sistema operacional, por padrão:

- Windows 32 bits: 'C:/Program Files (x86)/Arduino/libraries';
- Windows 64 bits: 'C:/Program Files/Arduino/libraries';
- Linux: '/home/Arduino/libraries'

Uma vez importada a biblioteca, será possível acessar a implementação do algoritmo `wallTracker()` através do menu de exemplos das bibliotecas da IDE, como mostra a Figura 22. Ao realizar esse procedimento, a biblioteca ficará disponível para ser utilizada em qualquer projeto, necessitando apenas importá-la no documento do mesmo.

Figura 22 – Acesso aos algoritmos exemplos da biblioteca desenvolvida após importação



Fonte: Autoria própria (2024).

4 RESULTADOS

A Figura 23 apresenta um caso de teste passo-a-passo do protótipo do robô móvel utilizando o algoritmo exemplo *wallTracker()*. Já a Figura 24 mostra a tela do monitor serial com as métricas calculadas para esse mesmo algoritmo exemplo.

Foram realizados oito testes com o protótipo desenvolvido em um caminho determinado, seguindo o exemplo da Figura 8. Dentre os testes realizados, em seis o protótipo deveria percorrer o circuito realizando apenas rotações no sentido horário (T1 a T6), e em dois casos deveria realizar conversões no sentido anti-horário (T7 e T8). O experimento resultou em seis tentativas com sucesso, atingindo uma taxa de 75% de acerto. Os detalhes das tentativas são apresentados na Tabela 4. O vídeo do experimento pode ser encontrado no repositório *git* apresentado na seção 3.6.

Tabela 4 – Experimentos realizados com o protótipo.

Tentativa	Resultado	Detalhes
T1	FALHA	O protótipo colidiu com o obstáculo em um determinado ponto e não finalizou o caminho.
T2	FALHA	O protótipo ficou preso em um ponto e não finalizou o caminho.
T3	SUCESSO	O protótipo não iniciou o trajeto de forma linear, mas finalizou o caminho.
T4	SUCESSO	O protótipo colidiu no fim do percurso, mas finalizou o caminho.
T5	SUCESSO	O protótipo finalizou o caminho.
T6	SUCESSO	O protótipo colidiu no fim do percurso, mas finalizou o caminho.
T7	SUCESSO	O protótipo teve dificuldades em algumas partes do percurso, mas finalizou o caminho.
T8	SUCESSO	O protótipo encostou no obstáculo no final do percurso, mas finalizou o caminho.

Fonte: Autoria própria (2024).

Durante a execução dos testes, foi notado que a roda omnidirecional influencia na direção inicial do robô móvel. Ou seja, mesmo que o robô esteja em uma direção, se a roda omnidirecional não estiver orientada na mesma direção, o robô realiza uma curva logo no início do movimento.

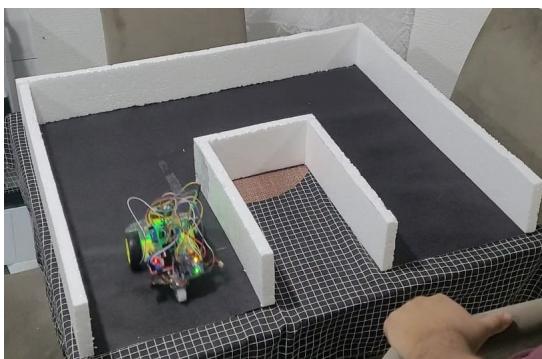
Outro ponto de atenção identificado durante os testes foi leituras descalibradas do sensor de proximidade. Isso aconteceu pois o tempo de execução do algoritmo estava muito rápido, impossibilitando que o sensor pudesse fazer a leitura corretamente. Para resolver esse problema, foi adicionado um tempo mínimo de 10 milissegundos de espera entre as execuções. Esse tempo é relativamente pequeno e não chega a afetar a resposta do robô móvel aos obstáculos.

Notou-se também, durante o experimento, que o projeto desenvolvido não leva em considerações percalços que podem ocorrer durante o translado do robô, como por exemplo o robô móvel não conseguir realizar a rotação corretamente por estar muito perto da parede. Além disso, foi identificado que possíveis interferências ou mal-funcionamento dos sensores podem

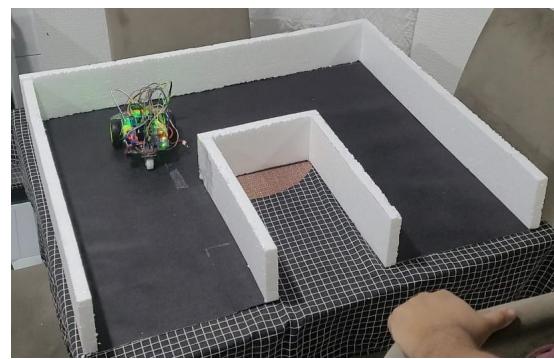
ocorrer. Nesses casos, como se tratam de erros dos componentes e não da implementação, o recomendado é trocar os módulos ou compensar de alguma forma suas leituras.

Figura 23 – Protótipo do robô móvel diferencial percorrendo o caminho evitando a colisão.

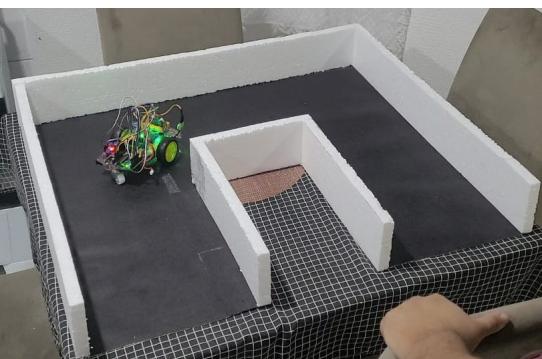
(a) O robô inicia o percurso seguindo em frente



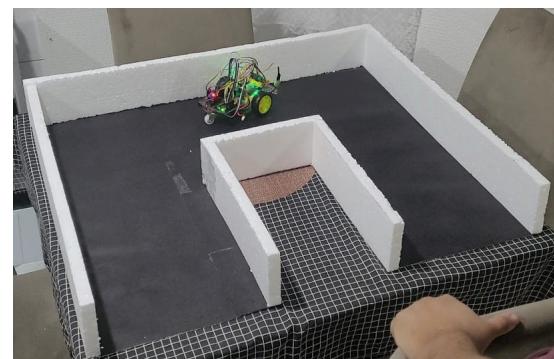
(b) O robô encontra um obstáculo e para



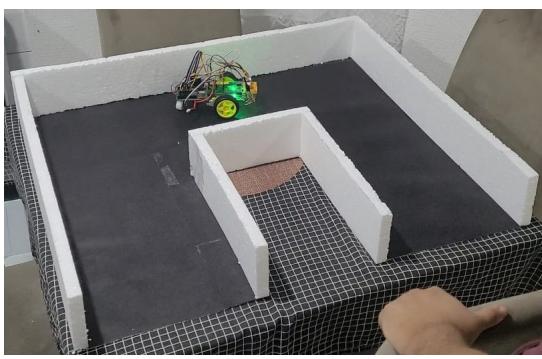
(c) O robô recua um pouco e realiza a conversão à direita



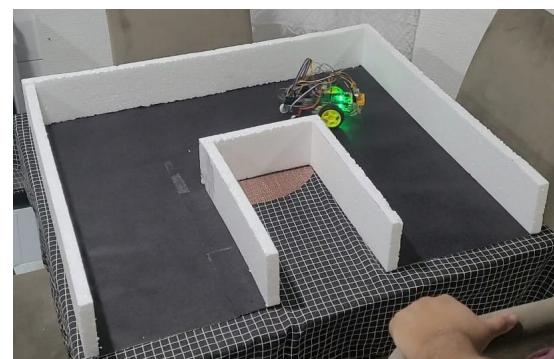
(d) O robô segue em frente, encontra um obstáculo e para



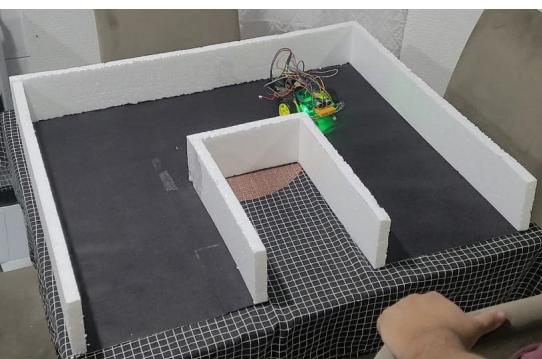
(e) O robô recua um pouco e converge à direita



(f) O robô segue em frente, encontra um obstáculo e para



(g) O robô recua um pouco e rotaciona à direita



(h) o robô segue em frente e finaliza o percurso

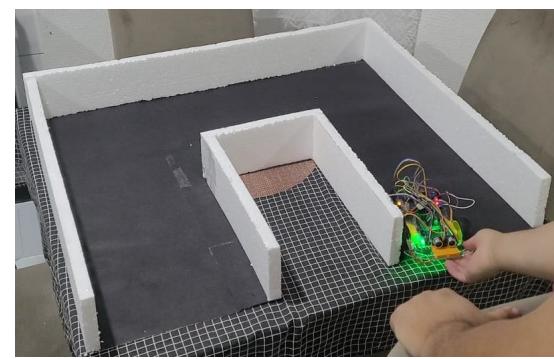
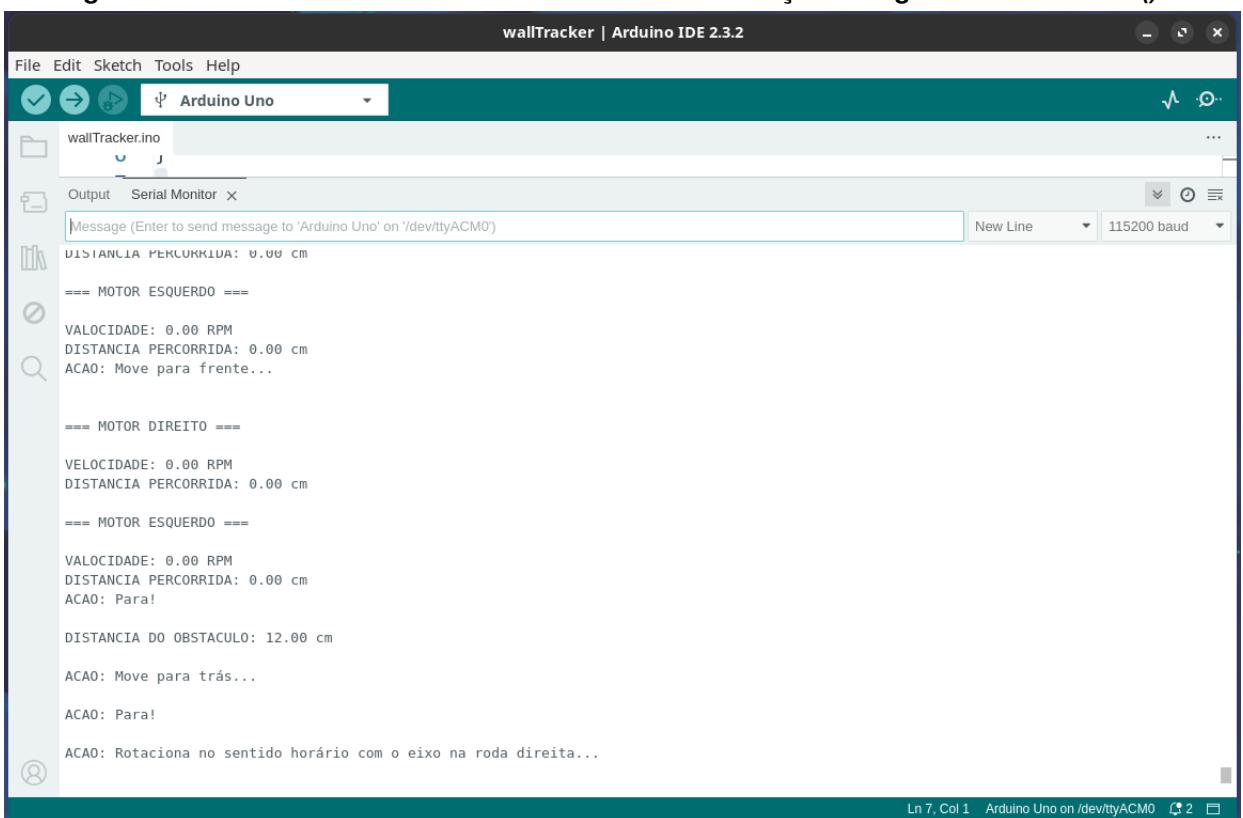


Figura 24 – Monitor serial do Arduino IDE durante a execução do algoritmo *wallTracker()*



The screenshot shows the Arduino IDE 2.3.2 interface with the title bar "wallTracker | Arduino IDE 2.3.2". The menu bar includes File, Edit, Sketch, Tools, Help, and a connection status for "Arduino Uno". The toolbar has icons for upload, refresh, and other functions. The left sidebar shows a file tree with "wallTracker.ino" selected. The main area is the Serial Monitor, which displays the following text:

```
Message (Enter to send message to 'Arduino Uno' on '/dev/ttyACM0')
DISTANCIA PERCORRIDA: 0.00 cm
==== MOTOR ESQUERDO ====
VALOCIDADE: 0.00 RPM
DISTANCIA PERCORRIDA: 0.00 cm
ACAO: Move para frente...

==== MOTOR DIREITO ====
VELOCIDADE: 0.00 RPM
DISTANCIA PERCORRIDA: 0.00 cm
==== MOTOR ESQUERDO ====
VALOCIDADE: 0.00 RPM
DISTANCIA PERCORRIDA: 0.00 cm
ACAO: Para!

DISTANCIA DO OBSTACULO: 12.00 cm
ACAO: Move para trás...
ACAO: Para!
ACAO: Rotaciona no sentido horário com o eixo na roda direita...
```

The status bar at the bottom shows "Ln 7, Col 1" and "Arduino Uno on /dev/ttyACM0".

Fonte: Autoria própria (2024).

5 CONCLUSÃO E TRABALHOS FUTUROS

O desenvolvimento de um robô móvel diferencial didático para o ensino de robótica nos cursos de engenharia de computação se mostra uma necessidade para que os estudantes se familiarizem com conceitos tanto de *hardware* quanto *software* e possam compreender a forma como os dois se relacionam. Além disso, a utilização de materiais comuns no meio da robótica facilita ainda mais esse entendimento.

O projeto em questão se mostrou eficaz na construção de um agente que sirva como base para instigar a curiosidade e criatividade dos estudantes, possibilitando futuras implementações e expansões por parte dos mesmos.

Por se tratar de um protótipo exemplo com o objetivo de instigar novos desenvolvimentos relacionados, o robô móvel desenvolvido apresentou algumas falhas durante seu processo de construção. Entretanto, tais falhas mostram que, mesmo com toda a pesquisa teórica realizada para documentar e evitar erros, estes ainda podem ocorrer. Portanto, não só é interessante apresentar tais pontos de melhoria, como é imprescindível para que sejam corrigidos em trabalhos futuros.

Uma proposta para trabalho futuro que pode ser apontada é a implementação de controladores Proporcional Integral Derivativo (PID) de malha fechada em cada um dos motores, que utilizem as medições dos *encoders* para manter a velocidade de rotação igual nas duas rodas. Essa implementação assegura que sempre as rodas estariam na mesma velocidade, evitando movimentos angulares indesejados no robô móvel diferencial.

REFERÊNCIAS

- ALMEIDA, F. **O que é Encoder? Para Que Serve? Como Escolher? Como Interfacear?** 2017. Disponível em: <https://materiais.hitecnologia.com.br/blog/o-que-%C3%A9-encoder-para-que-servir-como-escolher-como-interfacear/>. Acesso em: 10 jun. 2023.
- BALBINOT, A. **Instrumentação e Fundamentos de Medidas**. Grupo GEN, 2019. ISBN 9788521635888. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788521635888/>. Acesso em: 10 jun. 2023.
- BIESEK, L. J. **Veículo autônomo: uma contribuição para estacionamento**. 2016. Disponível em: <http://repositorio.utfpr.edu.br/jspui/handle/1/14618>. Acesso em: 10 jun. 2023.
- CARBONERA, M. G. **Navegação em robôs moveis por arbitragem e fusão em arquiteturas comportamentais**. 2021. Disponível em: <http://repositorio.utfpr.edu.br/jspui/handle/1/28611>. Acesso em: 10 jun. 2023.
- CARDOSO, M. **O que é um Microcontrolador?** 2020. IEE RAS UFCG. Disponível em: <https://edu.ieee.org/br-ufcgras/o-que-e-um-microcontrolador/>. Acesso em: 10 jun. 2023.
- LIMA, J. J. de. **Mobile Robots**: Robots kinematics. Pato Branco, Brasil, 2023. Disponível em: <https://jeferson.aulas.gitlab.io/mobile-robotics/cinematica-aula.pdf>. Acesso em: 23 jun. 2023.
- MORAES, M. C. **Informática Educativa no Brasil: um pouco de história...** 1993. Disponível em: <https://repositorio.ucb.br:9443/jspui/bitstream/123456789/7727/1/Inform%C3%A1tica%20Educativa%20no%20Brasil%20um%20Pouco%20de%20Hist%C3%B3ria.pdf>. Acesso em: 10 jun. 2023.
- NERY, G. **Guia Definitivo de uso da Ponte H L298N**. 2022. Blog Eletrogate. Disponível em: <https://blog.eletrogate.com/guia-definitivo-de-uso-da-ponte-h-l298n/>. Acesso em: 10 jan. 2024.
- PATSKO, L. F. **Aplicações, Utiliação e Funcionamento de Sensores**. Paraná, Brasil, 2006. Disponível em: https://www.maxwellbohr.com.br/downloads/robotica/mec1000_kdr5000/tutorial_eletronica_-_aplicacoes_e_funcionamento_de_sensores.pdf. Acesso em: 10 Jun. 2023.
- PATSKO, L. F. **Tutorial - Montagem da Ponte H**. Paraná, Brasil, 2006. Disponível em: https://www.maxwellbohr.com.br/downloads/robotica/mec1000_kdr5000/tutorial_eletronica_-_montagem_de_uma_ponte_h.pdf. Acesso em: 10 jun. 2023.
- RUBIO-TAMAYO, J. L.; JR., C. G.; HENRIQUES, R. Robótica para los procesos de enseñanza de la disciplina mecatrónica: Desarrollo del prototipo edubot v-2. In: **III Congreso Internacional Sociedad Digital**. Madrid, Espanha: [s.n.], 2014.
- SASAKI, A. **Estratégia de Desvio de Obstáculo para Navegação Autônoma de um Robô Móvel do Tipo Car-Like**. 2012. Disponível em: https://www3.dti.ufv.br/sig_del/consultar/download/149. Acesso em: 03 jun. 2023.
- SCHULTZ, G. **Interrupção: O que é e Como Utilizar no Arduino**. 2022. Blog Eletrogate. Disponível em: <https://blog.eletrogate.com/interrupcao-o-que-e-como-utilizar-arduin/>. Acesso em: 10 may. 2024.
- SILVA, S. R. X. d. **Protótipo de um robô móvel de baixo custo para uso interdisciplinar em cursos superiores de engenharia e computação**. fev. 2016. 218 p. Dissertação (Mestrado)

- Faculdade de Engenharia Mecatrônica, Universidade Federal da Bahia, Salvador, fev. 2016. Disponível em: <http://repositorio.ufba.br/ri/handle/ri/18614>. Acesso em: 10 jun. 2023.
- SILÍCIO, V. de. **Página Inicial - Loja Vida de Silício**. 2023. Disponível em: <https://www.vidadesilicio.com.br/>. Acesso em: 10 jun. 2023.
- VIDAL, V. **O que é Arduino: Afinal, para que Serve o Arduino?** 2024. Blog Eletrogate. Disponível em: <https://blog.eletrogate.com/arduino-primeiros-passos/>. Acesso em: 12 mar. 2024.
- WELTER, A. R. **Aprendizado por reforço profundo para navegação de um veículo guiado automaticamente**. 2022. Disponível em: <http://repositorio.utfpr.edu.br/jspui/handle/1/30617>. Acesso em: 10 jun. 2023.
- ZAQUEU, A. C. M.; RAMOS, D. C.; NETTO, A. V. Curumim: A robótica educacional como proposta metodológica para o ensino. In: **II Congresso Brasileiro de Informática na Educação (CBIE 2013)**. São Carlos, Brasil: [s.n.], 2014. Disponível em: https://www.researchgate.net/publication/299666251_Curumim_A_Robotica_Educacional_como_Proposta_Metodologica_para_o_Ensino. Acesso em: 14 jun. 2023.