FACULDADE DE TECNOLOGIA DE SÃO BERNARDO DO CAMPO
“ADIB MOISÉS DIB”

JULIA MARIA PAULINO VENANCIO
LUCAS LEAL MARSOLA

SEGURANÇA EM NAVEGAÇÃO DE ROBÔS MÓVEIS

São Bernardo do Campo - SP
Novembro/2022

JULIA MARIA PAULINO VENANCIO
LUCAS LEAL MARSOLA

SEGURANÇA EM NAVEGAÇÃO DE ROBÔS MÓVEIS

.

Trabalho de Conclusão de Curso apresentado à Faculdade de Tecnologia de São Bernardo do Campo “Adib Moisés Dib” como requisito parcial para a obtenção do título de Tecnólogo (a) em Automação Industrial.

Orientador: Prof. Dr. Claudio Rodrigo Torres

São Bernardo do Campo – SP
Novembro/2022

JULIA MARIA PAULINO VENANCIO
LUCAS LEAL MARSOLA

SEGURANÇA EM NAVEGAÇÃO DE ROBÔS MÓVEIS

Trabalho de Conclusão de Curso apresentado à Faculdade de Tecnologia de São Bernardo do Campo “Adib Moises Dib” como requisito parcial para a obtenção do título de Tecnólogo (a) em Automação Industrial.

Orientador: Prof. Dr. Claudio Rodrigo Torres

Trabalho de Conclusão de Curso apresentado e aprovado em:29/11/2022

Banca Examinadora:

Prof. Dr. Claúdio Rodrigo Torres

Prof. Esp. Gervásio das Neves Salvador

Prof. Esp. Marcos Vagner Zamboni

Dedicamos esse trabalho para os nossos pais e a todos aqueles que de certa forma nos inspiraram, incentivaram e ajudaram para que conseguíssemos chegar até aqui.

.

Agradecemos ao prof. Dr. Cláudio Rodrigo Torres por todo o tempo, conhecimento e suporte disponibilizado para a execução deste projeto.

“O conhecimento cresce exponencialmente. Quanto mais soubermos, maior a nossa capacidade de aprender, e mais rápido expandimos a nossa base de conhecimento.”

DAN BROWN

RESUMO

Os robôs móveis evoluíram muito nos últimos anos e ganharam muita visibilidade e destaque tanto nas suas aplicações domésticas como nas industriais, surgindo assim a necessidade de colocar a segurança desses aparelhos em pauta de modo semelhante a tratativa dos robôs convencionais. O foco deste trabalho foi produzir um robô que fosse capaz de se locomover de modo autônomo através de ferramentas de programação, monitoramento e mapeamento. Diante dessa demanda, foi proposto o uso do Arduino UNO R3 que, integrado ao sensor ultrassônico adotado, tornou o robô capaz de desviar de objetos ou pessoas e andar numa direção mais segura.

Palavras-chave: Autonomia. Segurança. Robótica móvel. Monitoramento. Mapeamento.

ABSTRACT

Mobile robots have evolved a lot in recent years and have gained a lot of visibility and prominence both in their domestic and industrial applications, thus emerging the need to put the safety of these devices on the agenda in a similar way to the treatment of conventional robots. The focus of this work was to produce a robot capable of moving autonomously through programming, monitoring and mapping tools. Faced with this demand, the use of the Arduino UNO R3 was proposed, which, integrated with the adopted ultrasonic sensor, made the robot capable of deflecting objects or people and walking in a safer direction.

Keywords: Autonomy. Safety. Mobile robotics. monitoring. mapping.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

INTRODUÇÃO

A fabricação e implantação dos robôs surgiram em massa em pró da indústria, de modo a realizar um trabalho com cem por cento de eficiência e em posições pouco ou nada ergonômicas. Atualmente no Brasil, o mercado produtivo tem optado pela implantação de sistemas robotizados, principalmente nos setores automotivos, pelo ótimo retorno financeiro.

O primeiro robô móvel foi criado em 1954, sendo instalado pela indústria Cravens Company, com a ideia de ser um transportador automático. Para seu funcionamento, precisou ser implementado uma infraestrutura, como cabos magnéticos demarcando o caminho pelo qual ele deve seguir, porém problemas como obstruções em seu trajeto acarretam a realidade de um robô automático com falhas que devem ser consertadas com ajuda humana. Devido ao avanço tecnológico, hoje já temos aplicações de robôs móveis autônomos que são equipamentos que conseguem se deslocar sem colidir com outros objetos ou seres humanos sendo altamente implantados para serviços de logística interna. No entanto, algumas infelizes situações como o acidente ocorrido na empresa Amazon no ano de 2018 que deixou 24 funcionários feridos, nos fazem repensar o sistema de segurança deles.

Para robôs móveis autônomos, o problema de controle é concentrado no conceito de navegação. Segundo o autor Nehmzow (2000), a navegação pode ser definida por três habilidades básicas:

- Construção e interpretação de mapas;
- planejamento de trajetória;
- e auto localização.

A auto localização significa a capacidade de determinar e/ou estabelecer sua própria posição em relação ao quadro de referência, já o planejamento da trajetória tem como função designar as posições inicial e final relacionadas ao mesmo quadro. De modo geral, essas duas habilidades necessitam de uma representação do ambiente de uso e a habilidade de interpretá-la, sendo fundamental a construção de um mapa.

O problema básico da construção e interpretação do mapa é entender a posição do robô no ambiente. O método mais comum é por meio de um sistema hodômetro robótico, no qual a posição e a direção são estimadas somando cada movimento

relacionado ao ponto inicial. Como o problema dos erros do hodômetro vem se acumulando, é mais comum encontrar sistemas que também agregam informações sensoriais, pois tais erros não podem ser corrigidos sem a percepção externa.

O objetivo deste trabalho é projetar e construir um robô móvel de baixo custo utilizando o microcontrolador Arduino que seja capaz de desviar de obstáculos ao longo de seu caminho por meio de tomadas de decisões.

A seguir, no primeiro capítulo será abrangido toda a fundamentação teórica do projeto, como por exemplo o histórico da robótica móvel e seus conceitos básicos, principalmente no que diz respeito a veículos autoguiados e robôs móveis autônomos. Além disso também é abordado sobre a segurança necessária nesse modelo de equipamento.

O segundo capítulo consiste em descrever a metodologia de trabalho utilizada, ou seja, todos as técnicas e procedimentos para alcançar o resultado.

O terceiro capítulo é o desenvolvimento do projeto, nele contém dados relacionados a sua especificação técnica, sendo subdividido na arquitetura do projeto, software e testes de integração.

Por fim, nas considerações finais será retomado os objetivos propostos no início do projeto, apresentado os resultados objetivos e realizado uma comparação.

1 FUNDAMENTAÇÃO TEORICA

Para o desenvolvimento do projeto, é necessário conhecimento prévio de alguns assuntos. Portanto, nesse capítulo, será apresentado o referencial teórico sobre os principais componentes utilizados para a execução deste trabalho.

1.1 Histórico

O primeiro manipulador foi criado em 1942, quando cientistas tentavam manipular materiais radioativos como urânio. Devido ao alto teor de radiação emitidos, tinham de manipular através de uma caixa transparente com luvas acopladas nela, porém não era muito eficaz para suas saúdes. Então veio a ideia do primeiro manipulador, que teve grande dificuldade de controle devido ao mapeamento dos comandos na época. Após o fim da segunda guerra os países começaram a investir na tecnologia nuclear com vários intuitos de produção de energia e armas nucleares, desenvolvendo o manipulador com interfaces homem-máquina, assim surgindo o primeiro braço robô em 1956.

A robótica móvel é um tema que vem sendo pesquisado a anos, e evoluído cada vez mais, com o objetivo de produzir um robô autônomo com capacidade de tomar decisões sem ajuda humana.

Em 2004 foi realizado a primeira competição de automóveis autônomos, promovido pelo DARPA *(Defense Advanced Research Projects Agency)*, uma agência americana com o propósito de investimentos nos avanços tecnológicos voltados a segurança nacional (Darpa, 2020). Para ela foram projetados veículos com diversos sensores. O desafio proposto era atravessar um percurso de 240 km em até 10 horas. Nenhum participante conseguiu concluir a prova, o veículo que chegou mais longe alcançou uma faixa de 12 km.

Em 2005, com a estratégia de criar um mapa 3D com tempo real do terreno em sua volta, diversos sensores, câmera de vídeo e diversos microcomputadores. A equipe de Stanford venceu a competição ganhando o prêmio em dinheiro com o tempo de 6 horas e 54 minutos.

Na terceira competição realizada em 2007, o trajeto programado era de percorrer em cidade 60 Km em 6 horas, obedecendo leis de trânsito. A interação dos robôs.

1.2 Veículo autoguiado x robô móvel autônomo

Um veículo autoguiado (AGV) é projetado para se locomover entre diversos locais, geralmente utilizado para logística com transporte de materiais. Devido a complexibilidade de navegação, é necessário a utilização de infraestruturas demarcando o caminho a ser utilizado, como cabos magnéticos ou fitas especiais onde o veículo irá seguir sem sair deste trajeto. Equipado com sensores na frente ele pode “ver” se há alguma obstrução em seu caminho e parar imediatamente, soando um alarme para que algum operador venha e remova o objeto do caminho.

Caso seja necessária uma nova rota a ser seguida, deve-se mudar a trajetória já instalada e implementar outra com o novo caminho até seu destino. Essa estrutura por não ter nenhum tipo de supervisionamento em seu percurso pode ser facilmente vandalizada prejudicando a empresa.

Um robô é uma máquina que realiza diversas tarefas simples ou complexas repetitivas vezes automaticamente. Podendo ser tarefas predefinidas por uma programação de passo a passo, ou de maneira autônoma onde ele toma decisões levando em consideração o ambiente e a situação em que encontra.

Conforme Jung (2005), o robô móvel autônomo, também conhecido como AMR, tem capacidade de se locomover e operar de forma semi ou completamente autônomo, possuindo, então, a vantagem de analisar o ambiente a sua volta e ir direto ao destino sem auxílio humano, podendo desviar de obstruções do caminho analisando o melhor percurso mais seguro, sem precisar de toda uma infraestrutura (um caminho) para seguir e com menores manutenção caso seja necessário um novo destino.

1.3 Sistema de direção

Conforme Heinen (2002), os modelos cinemáticos descrevem a maneira como o robô se locomove pelo ambiente. Um dos principais tipos de sistemas de direção de veículos com rodas é o modelo cinemático Ackerman, também conhecido como acionamento Ackerman. Este é o mais conhecido, principalmente na indústria automobilística e consiste em um motor menor controlando a direção da roda dianteira para realizar curvas e um maior para movimentos para frente e para trás. No entanto, existe um grande problema nele que é a dificuldade de rotacionar em torno do próprio eixo.

Há também o sistema de direção diferencial, no qual um lado é acionado independentemente do outro de modo que, para andar em linha reta, é preciso que ambos possuam mesma velocidade. Entretanto, buscando garantir maior agilidade, foi optado por um terceiro sistema: o omnidirecional. Nela é possível utilizar manobras e movimentar-se de lado sem alterar a posição inicial, tudo isso feito através de rodas omnidirecionais, como ilustra a figura abaixo.

Figura 1.1 – Rodas omnidirecionais

Fonte:https://www.fermarc.com/roda-de-carro-robo-inteligente-omnidirecional-mecanum

1.4 Microcontroladores

Em 1969, uma empresa japonesa fabricante de calculadoras eletrônicas chamada BUSICOM fez um pedido com a Intel para fornecer circuitos integrados para seus artigos e deveria produzir circuitos integrados para vários tipos de produtos. O engenheiro da Intel Marcian Hoff apresentou uma ideia revolucionária, um circuito integrado programável com este dispositivo, bastando alterar a programação podendo ser utilizado em diversas aplicações. Com base nessa ideia, a Intel criou o primeiro microprocessador, chamado 4004 (Breve, 2012) que pode processar 4 bits a 6kHz. A partir desse circuito integrado, surgiu uma competição de desenvolvimento que atingiu a evolução sem fim dos microprocessadores de hoje.

Com a evolução vem a necessidade de conforto e praticidade. TV com controle remoto, máquina de lavar, forno micro-ondas.

Os microcontroladores seguem os mesmos princípios dos microprocessadores, mas não requerem memória externa e barramentos externos para funcionar. Todos esses são construídos em um circuito integrado. Segundo Silva (2006), um microcontrolador é um pequeno dispositivo dotado de inteligência, basicamente composto por uma unidade central de processamento-CPU (unidade central de processamento), memória (dados e programas) e periféricos (portas I / O), I2C, SPI, USART, etc.).

Neste projeto, entre diversos fabricantes e modelos, foi adotado o Arduino UNO R3 em razão da equipe ter conhecimento prévio e afinidade com as plataformas do Arduino além de, em comparação aos outros modelos do mesmo fornecedor, esse, em especial, é o que mais atende a necessidade do protótipo.

1.5 Arduino

O Arduino é uma plataforma de desenvolvimento eletrônico baseada em uma única placa com hardware livre, ou seja, ela permite que o usuário a modifique, personalize e melhore de acordo com sua aplicação (Oliveira, 2018). Esse equipamento é composto por um microcontrolador e circuitos de entrada e saída. Quando foi concebido, o objetivo dos seus criadores era que ele fosse barato, fácil de

programar e funcionar para que, desse modo, fosse mais acessível a programadores iniciantes, projetistas amadores e estudantes.

Para programar esse sistema basta utilizar o seu ambiente de programação gratuito que se chama IDE. Nele você pode escrever e testar o código da sua aplicação e para transferir para o a placa é bem simples: basta conectar um cabo USB R3 e carregar o arquivo.

Na tabela 1.1 é possível verificar algumas das caraterísticas do modelo adotado.

Tabela 1.1 – Características do Arduino UNO R3

Características Arduino UNO R3	
Microcontrolador	ATMega328
Tensão de operação	5V
Tensão de entrada	7-12V
Portas Digitais	14 (6 podem ser usadas como PWM)
Portas Analógicas	6
Corrente Pinos I/O	40mA
Corrente Pinos 3,3V	50mA
Memória Flash	32KB (0,5KB usado no bootloader)
SRAM	2KB
EEPROM	1KB
Velocidade do Clock	16MHz

Fonte: https://www.baudaeletronica.com.br/arduino-uno-r3.html.

1.5.1 Modelos existentes

O sistema Arduino é formado por duas partes: a placa eletrônica e a interface de desenvolvimento (IDE). Existem muitos modelos deste dispositivo no mercado e sua escolha vai depender das necessidades de cada aplicação (Evans, 2013).

As placas mais conhecidas são: Arduino Mega, Pro, Mini, Uno e 101. Todas essas são indicadas para quem está começando por serem de fácil integração. Existem, também, modelos mais complexos como o Arduino Mega 2560, Zero e Due, mas suas aplicações são normalmente em casos que exija uma funcionalidade mais avançada ou maior desempenho. Na tabela 1.2 é possível ver a comparação entre

elas, levando em consideração o clock, números de entradas e saídas, tensão de alimentação, PWM e tamanho da memória flash.

Tabela 1.2 – Comparativo modelos de Arduino

Comparativo modelos de arduino

Modelo	Vin	Vpin	Clock	Digitais	Analógicas	PWM	Flash
Uno - R3	7-12V	5V	16MHz	14	6	6	32Kb
Leonardo	7-12V	5V	16MHz	20*	12	7	32Kb
Romeo V2.0	7-12V	5V	16MHz	20*	12	7	32Kb
Julieta	7-12V	5V	16MHz	14	8**	6	32Kb
Mega 2560 R3	7-12V	5V	16MHz	54	16	15	256Kb
Mega ADK	7-12V	5V	16MHz	54	16	15	256Kb
Due	7-12V	3.3V	84MHz	54	12	12	512Kb
Esplora	5V	5V	16MHz	-	-	-	32Kb
Ethernet	7-12V	5V	16MHz	14	6	4	32Kb
Fio	3,3-12V	3,3V	8MHz	14	8	6	32Kb
Micro	7-12V	5V	16MHz	20	12	7	32Kb
Pro Micro	5-12V	5V	16MHz	12	4	5	32Kb
Pro Mini 3,3V	3,3-12V	3,3V	8MHz	14	6	6	32Kb
Pro Mini 5V	5-12V	5V	16MHz	14	8	6	32Kb

Fonte: https://www.robocore.net/tutoriais/comparativo-arduino.

1.5.2 Expansão

O nome “Shield” é dado aos módulos de expansão das placas de Arduino. De acordo com Evans (2013), são placas especializadas capazes de maximizar a funcionalidade básica do Arduino, o qual o objetivo principal é o desenvolvimento específico em uma aplicação. Alguns dos *shields* mais utilizados são:

- Motor shield: faz o controle de motores conectados a uma placa;
- Wi-fi shield: muito utilizado em projetos voltados para internet das coisas (IoT);
- Ethernet shield: torna possível o uso da internet com um cabo de rede padrão;
- GPS shield: possibilita uso de GSM, GPRS e PGS na mesma placa;
- Bluetooth shield: empregado em operações de comunicação sem fio, com alcance médio de 10 metros dependendo do fabricante.

Para agregar um módulo de expansão à um Arduino, basta conectá-lo na sua parte sua parte superior e realizar a comunicação através dos pinos de entradas e saídas (digitais ou analógicas) ou seriais.

1.6 Sensores

Uns dos elementos cruciais para garantir a movimentação autônoma do robô de modo seguro são os sensores. Segundo Hanapi (2013), o sensor ajuda a fornecer a entrada para o movimento do robô, detectando uma perturbação e envia a informação em forma de sinal eletrônico para o controlador. Ou seja, eles fazem com que a área ao redor do AMR se torne conhecida após realizada a captação e extração de dados significativos adquiridos a partir de suas medições.

Pensando nas necessidades envolvidas nesse projeto, dividiu-se esse tópico em dois objetivos. No primeiro, dados em relação as coordenadas dentro de um plano e sua orientação serão enviadas ao robô. No segundo, deve-se obter a distância em relação aos objetos e posição do cone.

Nesse trabalho utilizamos sensor ultrassônico que será apresentado no capítulo a seguir.

1.5.1 Sensor ultrassônico

O sensor ultrassônico, fundamentalmente, propaga um conjunto de ondas de som e mede o tempo que ele demora para retornar. Com base na propagação do som v (343 m/s) e no tempo t, pode-se descobrir a distância d do objeto causador da reflexão.

$$d=vt/2$$

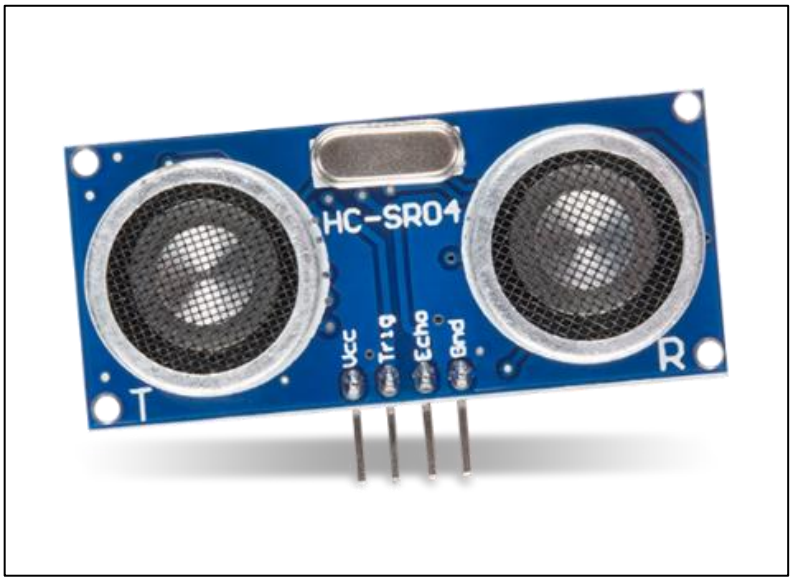
Um ponto importante é que o sensor ultrassônico transmite somente a informação de que existe um objeto a certa distância dentro da área medida uma vez que o som se propaga de forma cônica com uma abertura de 20° a 40° fazendo com que os dados direcionais sejam imprecisos.

A largura da banda é outra limitação deste dispositivo, principalmente em espaços abetos, visto que o tempo de ciclo deste sensor é relativamente lento.

No entanto, vale ressaltar que, a pouca abertura angular é justamente a razão pela qual o sensor ultrassônico foi escolhido, dado que ao utilizá-lo é necessário um menor número de sensores e o tempo de ciclo cai consideravelmente pois ele precisa, para identificar o cone, ele precisa de um tempo menor girando em seu próprio eixo, conseguindo, assim, corrigir o problema da largura de banda.

Neste robô foram adotados dois sensores HC-SR04 (figura 2) de modo a conseguir dar cobertura à toda a frente do robô. Este modelo é capaz de fazer a detecção de pessoas com um alcance entre 1 e 200 centímetros (Oliveira, 2018) e sua tensão de entrada é de 5 V.

Figura 1.2 – Sensor HC-SR04



Fonte: https://br.mouser.com/new/sparkfun/sparkfun-hcsr04-distance-sensor/

2 METODOLOGIA

Neste capítulo encontra-se o percurso percorrido para o desenvolvimento e elaboração deste projeto intitulado “Segurança em navegação de robôs móveis”. Refere-se a um estudo elaborado através da Faculdade de Tecnologia de São Bernardo do Campo e confeccionado na residência dos integrantes do grupo.

2.1 Conceito de metodologia

De acordo Severino (2013), a metodologia é o caminho percorrido para o desenvolvimento de uma pesquisa e enfoca que métodos são procedimentos amplos de raciocínio, enquanto as técnicas são procedimentos mais restritos que operacionalizam os métodos mediante emprego de instrumentos adequados. Diante disso, este estudo foi fundamentado num sistema de pesquisa qualitativa, mediante a uma análise bibliográfica aprofundada visando a entrega de um produto de simulação final.

2.2 Etapas para a elaboração do projeto

O trabalho será desenvolvido observando os seguintes estágios:

a) Levantamento de dados: consiste em levantar dados e bibliografias sobre a atual aplicação da robótica móvel autônoma, segurança e normas;
b) estudo sobre funcionamento do sistema de segurança: estudar equipamentos disponíveis no mercado para realizar tal função;
c) projeto mecânico: criar um visual para o robô, especificar modelo de motor, rodas e tipo de material, além de realizar as adaptações necessárias para prover bom desempenho.
d) conceitualização de funcionamento: impor um fluxo de funcionamento para o sistema e critérios os quais devam ser obedecidos;
e) projeto elétrico: estudar a aplicação do microcontrolador Arduino para o controle, bem como todos os acessórios para ser possível pôr em funcionamento, utilizando a plataforma Tinkercad para esquematizar os circuitos;

f) elaboração da programação: construir um código para o Arduino que seja capaz de atender todos os quesitos já levantados.
g) montagem e testes: realizar a união dos itens especificados, testar suas funções e fazer os ajustes necessários para garantir o funcionamento.

As etapas serão efetivadas nos períodos descritos na tabela a seguir.

Tabela 2.1 – Cronograma de atividades

CRONOGRAMA DE ATIVIDADES

Atividade	Integrante	Início	Conclusão
Entrega cronograma	Todos	16/08/2022	13/09/2022
Primeira entrega monografia	Todos	23/08/2022	13/09/2022
Especificação elétrica	Julia	09/09/2022	09/09/2022
Desenvolvimento elétrico	Julia	10/09/2022	20/09/2022
Primeira entrega relatório parcial de montagem	Todos	13/09/2022	13/09/2022
Desenvolvimento mecânico	Lucas	14/09/2022	20/09/2022
Segunda entrega relatório parcial de montagem	Todos	20/09/2022	20/09/2022
Desenvolvimento código arduino	Julia	21/09/2022	11/10/2022
Montagem e testes	Todos	25/09/2022	26/10/2022
Segunda entrega monografia	Todos	27/09/2022	27/09/2022
Desenvolvimento aplicativo	Todos	12/10/2022	20/10/2022
Terceira entrega relatório parcial de montagem	Todos	18/10/2022	18/10/2022
Quarta entrega relatório parcial de montagem	Todos	26/10/2022	26/10/2022
Elaboração artigo	Julia	01/11/2022	05/11/2022
Entrega final monografia	Todos	07/11/2022	07/11/2022
Elaboração do banner	Lucas	08/11/2022	18/11/2022
Apresentação dos protótipos	Todos	23/11/2022	23/11/2022
Apresentação para banca	Todos	24/11/2022	24/11/2022

Fonte: Autoria própria.

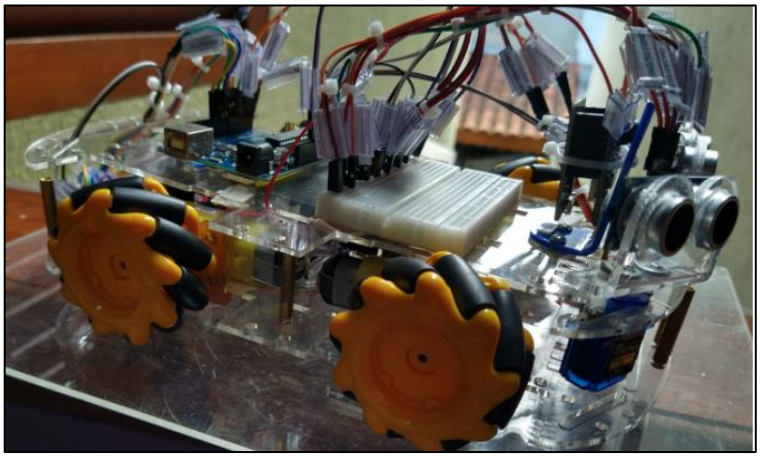
3 DESENVOLVIMENTO DO PROJETO

Este capítulo ilustra o passo a passo de desenvolvimento do robô autônomo desde dimensionamento a testes de integração e segurança.

3.1 Visão geral

Para melhor visualização do projeto, a figuras 3.1 mostra-o em perspectiva

Figura 3.1 – Robô perspectiva



Fonte: autoria própria

O robô ao entrar em funcionamento, fará a leitura do ambiente e caso não detecte nada, seguirá em frente. No entanto, caso os sensores ultrassônicos detectem algum item que possa atrapalhar essa trajetória, o robô é capaz de desviar a rota. Como ele também conta com as rodas omnidirecionais, o desvio acontece de forma mais rápida e eficiente.

3.2 Arquitetura do protótipo

O processo de arquitetar o protótipo foi subdividido em duas etapas, feitas em simultaneidade, sendo elas o dimensionamento mecânico e elétrico abrangidos a seguir.

3.2.1 Dimensionamento mecânico

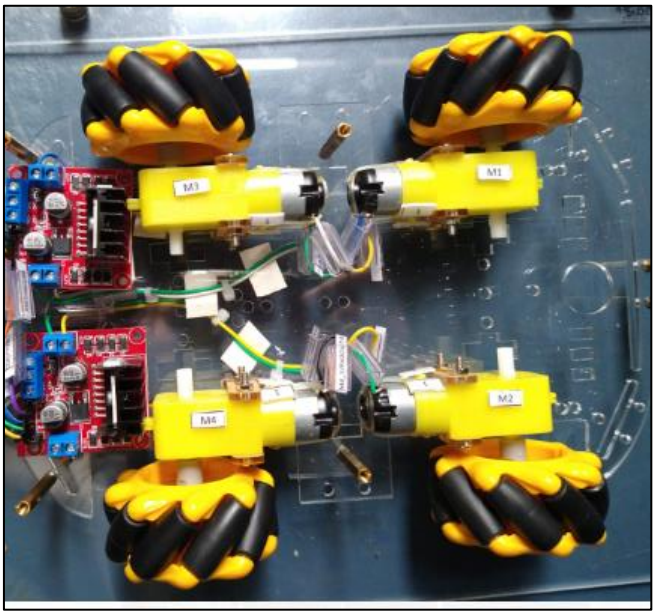
A parte mecânica do projeto é composta basicamente pela estrutura do robô (chapa de fixação, suportes de sensores e suportes de placas), motores utilizados e rodas. Nos tópicos a seguir será explicado um pouco mais a fundo esses temas.

3.2.1.1 Estrutura

O projeto foi estruturado com duas peças de acrílico. Ambas possuem diversos furos e rasgos para melhor eficiência no cabeamento e fixação dos componentes. Para prende-las foram utilizados parafusos M3 em 6 pontos espalhados estrategicamente na borda do robô.

Através dos rasgos, os motores foram colocados juntamente à uma peça de apoio em suas laterais.

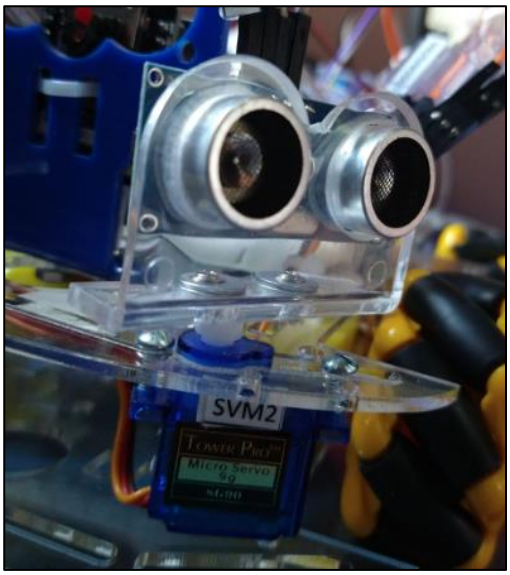
Figura 3.2 – Fixação dos motores



Fonte: autoria própria

Para os sensores ultrassônicos, adquirimos um suporte produzido em acrílico com furos para encaixe no eixo dos servos motores que serão posicionados nas duas laterais do robô visando maior área de vasculhamento.

Figura 3.3 – Suporte sensor ultrassônico



Fonte: autoria própria

3.2.1.2 Motores

Para dar movimento ao robô, foi adotado a utilização de quatro motores DC, levando em conta que cada roda omnidirecional, também conhecida como mecanum, necessita ter seu próprio sentido de giro para utilizar toda a capacidade que ela oferece. O modelo utilizado opera entre 3 e 6V, possui caixa de redução e eixo duplo.

Figura 3.4 – Motor DC 3-6V

Fonte: https://www.robocore.net/motor-motoredutor/motor-dc-3-6v-com-caixa-de-reducao-e-eixo-duplo

Além dos motores responsáveis pelo movimento do robô, existem também outros dois que realizam a movimentação dos sensores ultrassônicos. Para isso, o

modelo escolhido foi o Micro Servo 9g SG90 da fabricante Tower Pro pois possui 180º de rotação, o que garante boa amplitude e cobertura da área.

Figura 3.5 – Micro Servo 9g SG90

Fonte: https://www.eletrogate.com/micro-servo-9g-sg90-towerpro

3.2.1.2 Rodas

Procurando uma roda que fosse capaz de atender as necessidades do projeto e com valor acessível, foi optado um modelo chinês com 60mm de diâmetro. Ela é produzida em plástico, possui 31mm de espessura, 4mm de diâmetro de furo interno, 12mm de acoplamento e 9 pontos de rotação.

Figura 3.6 – Roda mecanum

Fonte: autoria própria

3.2.2 Arranjo elétrico

Como dito anteriormente, adotamos o Arduino UNO R3 para realizar todo o controle do projeto. Nele foram conectados dois servos motores descritos no item 3.2.1.2, dois sensores ultrassônicos, duas pontes H L298 e uma protoboard como pode ser visto no apêndice A.

A função da protoboard nesse projeto é de maximizar as saídas de VCC e GND fornecidas pelo Arduino, além de ser utilizada para fazer uma ligação em série entre as baterias e as duas pontes H. Outro ponto importante é que a alimentação do próprio controlador é feita através de 4 pilhas AA de 1,5V, totalizando 6V e a dos motores é feita externamente com duas baterias alcalina de 9V.

Para otimizar tempo de programação foi definida uma lista de IO que pode ser verificada na tabela abaixo.

Tabela 3.1 – Lista de IO

LISTA DE I/O

ENTRADAS ANALÓGICAS	DESCRIÇÃO	TAG
A0	Capta o som do sensor 1	SU1ECHO
A2	Capta o som do sensor 2	SU2ECHO
SAÍDAS DIGITAIS	DESCRIÇÃO	TAG
D02	Liga servo motor 1	SVM1
D03	Liga servo motor 2	SVM2
D04	Liga IN1 da ponte H 1	PH1IN1
D05	Liga IN2 da ponte H 1	PH2IN1
D06	Liga IN3 da ponte H 1	PH3IN1
D07	Liga IN4 da ponte H 1	PH4IN1
D08	Liga IN1 da ponte H 2	PH1IN2
D09	Liga IN2 da ponte H 2	PH2IN2
D10	Liga IN3 da ponte H 2	PH3IN2
D11	Liga IN4 da ponte H 2	PH4IN2
SAÍDAS ANALÓGICAS	DESCRIÇÃO	TAG
A1	Envia o som do sensor 1	SU1TRIG
A3	Envia o som do sensor 2	SU2TRIG

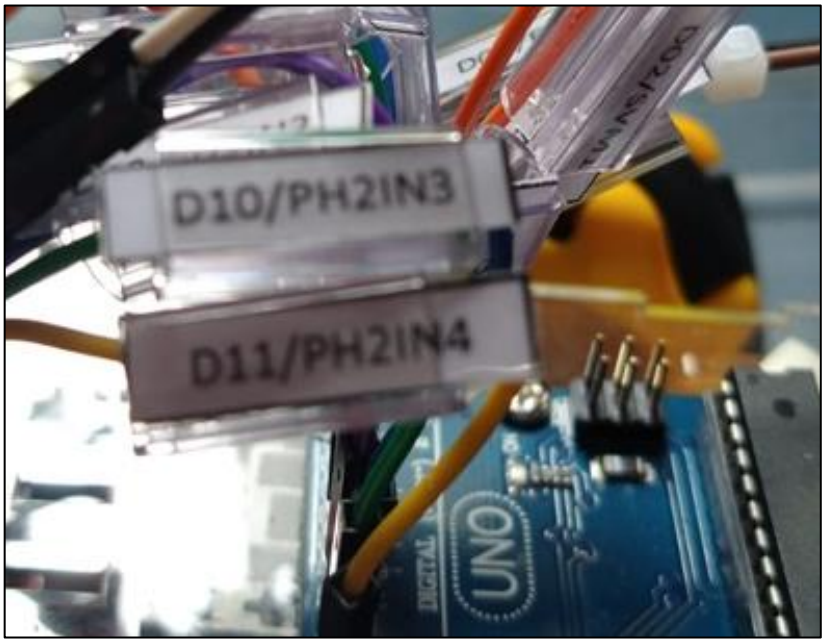
Fonte: autoria própria

Pensando nos testes posteriores e em possíveis manutenções, foi adotado um sistema de identificação dos cabos chamado DE/PARA. O nome é totalmente intuitivo, sendo assim o “DE” é substituído pelo nome do local de origem do cabo e o “PARA”

pelo destino. Por exemplo: o cabo com identificação “D01/SVM1” vem da saída digital 1 do Arduino e é ligado ao servo motor 1.

No apêndice B, podemos encontrar toda a relação “DE/PARA” do projeto.

Figura 3.7 – Exemplo de identificação



Fonte: autoria própria

3.3 Software

Para atenuar riscos e otimizar a qualidade do software, é necessário um planejamento inicial e para tal, é preciso descrever os requisitos modelar a solução e, por fim, codificar a solução.

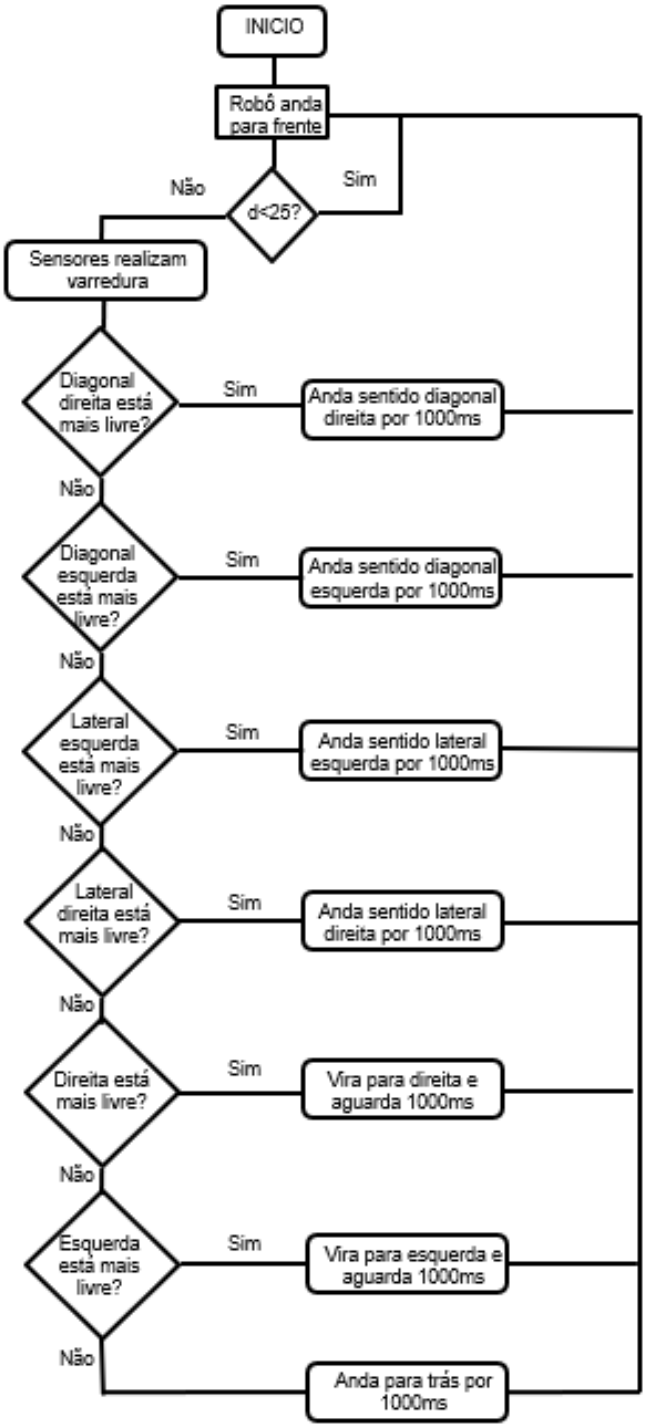
Neste capítulo serão apresentados os requisitos necessários para funcionamento do robô e o código proposto.

3.3.1 Requisitos

Para iniciar a programação, é necessário anteriormente definir suas condições (requisitos) de funcionamento. Diante disso, após análise foi estabelecido que o robô deve manter uma distância mínimo de 25 centímetros de qualquer objeto. Caso algo seja captado pelos sensores, será feita uma varredura para procurar o caminho mais livre e assim seguir.

Este protótipo foi projetado para operar conforme a sequência de eventos proposta no fluxograma da figura abaixo

Figura 3.8 – Fluxograma



Fonte: autoria própria

3.3.2 Código Arduino

Com o fluxograma em mãos, desenvolver o código para rodar no Arduino fica muitos mais simples. O primeiro passo foi adicionar a biblioteca do servo motor e relacionar os objetos lógicos as suas respectivas portas digitais e analógicas.

Em seguir, foram declaradas as variáveis e desenvolvido o código de varredura com o sensor ultrassônico. Com ele em funcionamento, foi elaborado a programação de movimentação do robô em cada uma das direções e realizado testes. O código está disponível para consulta no apêndice C.

CONSIDERAÇÕES FINAIS

Este projeto explorou parte da essência e conceito da robótica móvel autônoma, desde a movimentação até a percepção e desvio de objetos, cujo era o objetivo proposto.

Inicialmente, foi realizado todo o processo de escolha e dimensionamento de elementos mecânicos e eletrônicos . Em seguida, é descrito o sistema de software desenvolvido para manipulação desses elementos e controle do veículo.

Dentre os elementos eletrônicos, vale destacar o Arduino UNO, que foi a principal ferramenta de controle. O ambiente de desenvolvimento usado para codificar o sistema de software foi a própria IDE do Arduino, uma vez que sua interface intuitiva cooperou com a aceleração do processo de programação.

Testes servem para detectar as falhas no sistema, ou seja, as respostas do sistema que não estão de acordo com os requisitos estabelecidos anteriormente. Assim, dada a complexidade dos sistemas de software, uma vez que os componentes do programa estão codificados, o próximo passo consiste em testá-los. Após a realização, constatou-se que o robô alcançou todos os objetivos estabelecidos. No entanto, propõe-se algumas melhorias.

Para futuras versões de robô, será adicionado um hodômetro e adicionado um sistema de visão (como o módulo ESP32-CAM) de modo a utilizar o processamento de imagem para realizar o posicionamento global e recalibrar a posição captada pela hodômetria, evitando que os erros aumentem indefinidamente e permitindo maior movimentação.

REFERENCIAS

BREVE, Matheus Montanini; BERNUY, Miguel Angel Chincaro. **Introdução ao Desenvolvimento de Sistemas com Microcontroladores.** In: ANAIS III Seminário de Pesquisa Jr (SepesqJr**),** 3, 2012. Paraná: 2012, p. 1-6.

DARPA – Defense Advanced Research Projects Agency. **About DARPA**. Disponível em: https://www.darpa.mil/about-us/about-darpa. Acesso em: 07 jun.2022.

EVANS, Martin; NOBLE, Joshua; HOCHENBAUM, Jordan. **Arduino em ação**. 2013. 424p. ISBN 978-85-7522-373-4

HANAPI, Mohd; IZZATI, Hanis Syafiqah. **Development of motion controller system for autonomous robot**. 2013. 46p.Trabalho de conclusão de curso em engenharia elétrica e eletrônica - Universiti Teknologi PETRONAS. Bandar Seri Iskandar, 2013.

HEINEN, J. F**. Sistema de controle híbrido para robôs móveis autônomos.** 2002. 130 f. Dissertação Mestrado em Computação Aplicada – Centro de Ciências Exatas e Tecnológicas, Universidade do Vale do Rio dos Sinos. São Leopoldo, 2002.

Jung, Cláudio Rosito. et al. **Computação embarcada: Projeto e implementação de veículos autônomos inteligentes.** In: Anais do CSBC'05 XXIV Jornada de Atualização em Informática (JAI). São Leopoldo: SBC, 2005, p.1358–1406.

MANUAL DE NORMALIZAÇÃO DE PROJETO DE TRABALHO DE GRADUAÇÃO – FATEC SBCAMPO. **Material didático para utilização nos projetos de trabalho de graduação dos cursos de tecnologia em automação industrial e informática.** São Bernardo do Campo: Fatec, 2017.

NEHMZOW, Ulrich. **Mobile Robotics: A Practical Introduction**. 2ª. ed.London, 2000. 279p. ISBN 978-1-8233-726-1

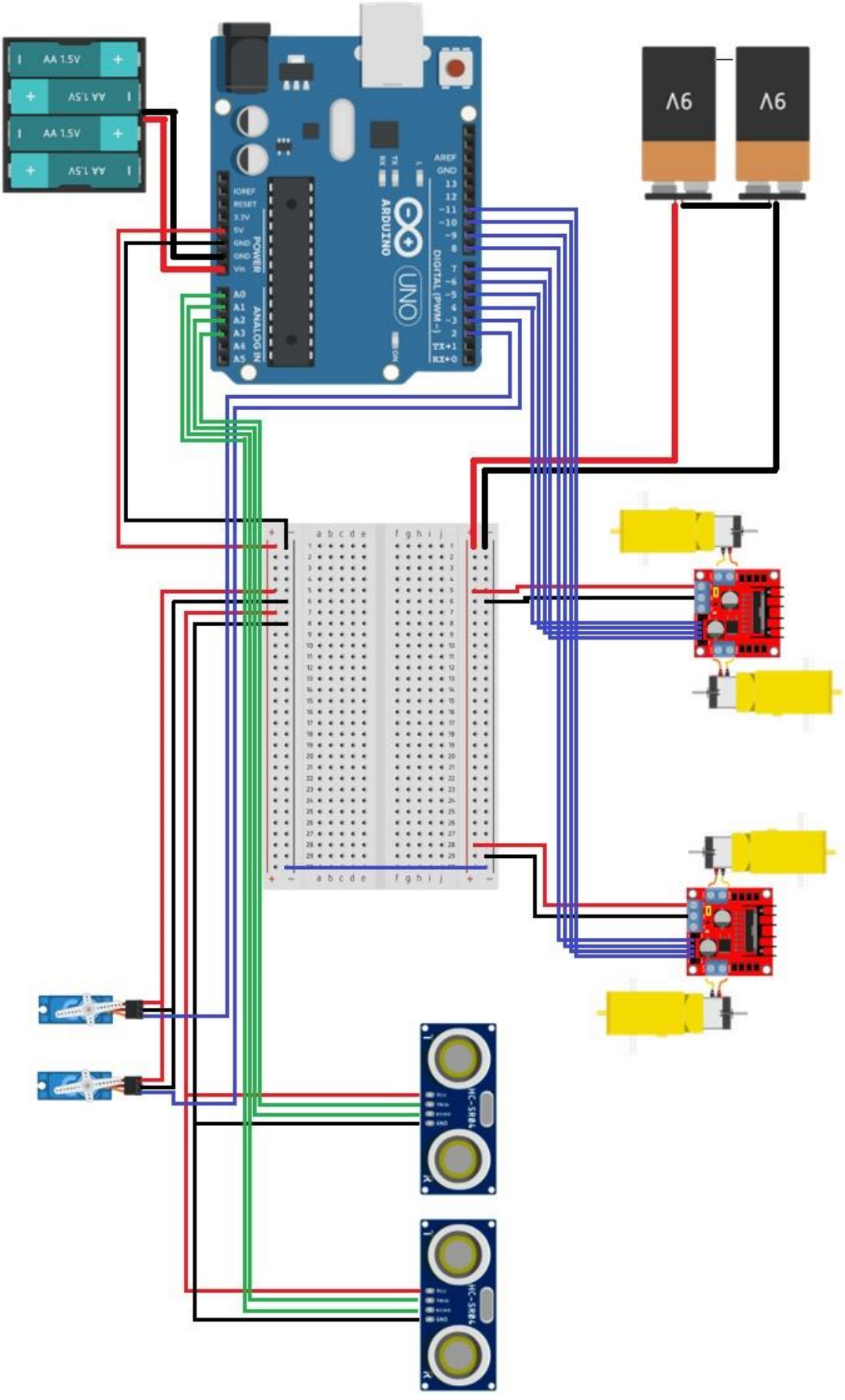
OLIVEIRA, Cláudio Luís Vieira. et al. **Aprenda Arduino: uma abordagem prática.** Duque de Caixas: Katzen Editora, 2018.

SEVERINO, Antônio Joaquim. **Metodologia do trabalho científico**. Antônio Joaquim Severino. -1. ed. -- São Paulo: Cortez, 2013.

SILVA, Renato A. **Programando microcontroladores PIC: Linguagem "C"**– SÉo Paulo : Ensino Profissional, 2006.

SOUZA, Fábio. **Introdução ao Arduino – Primeiros passos na plataforma**. Embarcados, 2013. Disponível em: https://embarcados.com.br/arduino-primeiros-passos/. Acesso em: 03 set. 2022.

APÊNDICE A – DIAGRAMA ELÉTRICO



APÊNDICE B – SISTEMA DE IDENTIFICAÇÃO

”

Sistema de identificação

DE	PARA	TAG
D02	SVM1	D02/SVM1
D03	SVM2	D03/SVM2
D04	PH1IN1	D04/PH1IN1
D05	PH1IN2	D05/PH1IN2
D06	PH1IN3	D06/PH1IN3
D07	PH1IN4	D07/PH1IN4
D08	PH2IN1	D08/PH2IN1
D09	PH2IN2	D09/PH2IN2
D10	PH2IN3	D10/PH2IN3
D11	PH2IN4	D11/PH2IN4
A0	SU1ECHO	A0/SU1ECHO
A1	SU1TRIG	A1/SU1TRIG
A2	SU2ECHO	A2/SU2ECHO
A3	SU2TRIG	A3/SU2TRIG
M1_1	PH1OUT2	M1_1/PH1OUT2
M1_2	PH1OUT1	M1_2/PH1OUT1
M2_1	PH2OUT2	M2_1/PH2OUT2
M2_2	PH2OUT1	M2_2/PH2OUT1
M3_1	PH1OUT4	M3_1/PH1OUT4
M3_2	PH1OUT3	M3_2/PH1OUT3
M4_1	PH2OUT4	M4_1/PH2OUT4
M4_2	PH2OUT3	M4_2/PH2OUT3
VCC	PH1+5	VCC/PH1+5
VCC	PH2+5	VCC/PH2+5
GND	PH1GND	GND/PH1GND
GND	PH2GND	GND/PH2GND
GND	GND	GND/GND
VCC	VCC	VCC/VCC
VCC	SVM1VCC	VCC/SVM1VCC
VCC	SVM2VCC	VCC/SVM2VCC
GND	SVM1GND	GND/SVM1GND
GND	SVM2GND	GND/SVM2GND
VCC	ESP5V	VCC/ESP5V
GND	ESPGND	GND/ESPGND
VCC	SU1VCC	VCC/SU1VCC
VCC	SU2VCC	VCC/SU2VCC
GND	U1GND	GND/SU1GND
GND	SU2GND	GND/SU2GND
BATVCC	VIN	BATVCC/VIN
BATGND	GND	BATGND/GND

APÊNDICE C – CÓDIGO ARDUINO

```
#include <Servo.h>     // Adiciona a biblioteca Servo
#include <Ultrasonic.h>

//Definindo os pinos
float dist;
float dist2;

//Sensor ultrassônico SU1
#define SU1ECHO A0     //Pino ECHO do sensor no pino analógico A0
#define SU1TRIG A1      //Pino TRIG do sensor no pino analógico A1

//Sensor ultrassônico SU2
#define SU2ECHO A2     //Pino ECHO do sensor no pino analógico A2
#define SU2TRIG A3      //Pino TRIG do sensor no pino analógico A3

// Motor um                 //Ligação dos pinos da Ponte H L298N 'PH1'
#define PH1IN1  4        //pino PH1IN1 na porta digital 4
#define PH1IN2  5        //pino PH1IN2 na porta digital 5
// Motor tres               //Ligação dos pinos da Ponte H L298N 'PH1'
#define PH1IN3  6        //pino in3 na porta digital 6
#define PH1IN4  7        //pino in4 na porta digital 7
//Motor dois               // Ligação dos pinos da Ponte H L298N 'PH2'
#define PH2IN1  8       //pino PH1IN1 na porta digital 8
#define PH2IN2  9       //pino PH1IN2 na porta digital 9
//Motor quatro            // Ligação dos pinos da Ponte H L298N 'PH2'
#define PH2IN3  10       //pino in3 na porta digital 10
#define PH2IN4  11       //pino in4 na porta digital 11

Servo SVM1;             //Objeto para controlar servo motor 1
Servo SVM2;             //Objeto para controlar servo motor 2


//Variaveis
int DistanciaDireita, DistanciaEsquerda, DistanciaDiagonalEsquerda,
DistanciaDiagonalDireita, Distanciafrente;  // variavel de Distâncias de ambos os lados

int esquerda = 0;
int diagonalesquerda = 0;
int direita = 0;
int diagonaldireita = 0;
int frente = 0;

void setup() {

 Serial.begin(9600); // inicializa a comunicação serial para mostrar dados

  //Define o servo motor 1 na porta 2
  SVM1.attach(2);
```

```
  //Define o servo motor 2 na porta 3
  SVM2.attach(3);

  // Definir todos os pinos de controle do motor como saídas
  pinMode(PH1IN1, OUTPUT);
  pinMode(PH1IN2, OUTPUT);
  pinMode(PH1IN3, OUTPUT);
  pinMode(PH1IN4, OUTPUT);
  pinMode(PH2IN1, OUTPUT);
  pinMode(PH2IN2, OUTPUT);
  pinMode(PH2IN3, OUTPUT);
  pinMode(PH2IN4, OUTPUT);
  //Configuraçõs do sensor ultrassonico
  pinMode(SU1TRIG, OUTPUT);     //def ine o pino TRIG como saída
  pinMode(SU1ECHO, INPUT);      //define o pino ECHO como entrada
  pinMode(SU2TRIG, OUTPUT);     //define o pino TRIG como saída
  pinMode(SU2ECHO, INPUT);      //define o pino ECHO como entrada




}

void loop() {

//PROGRAMA PRINCIPAL

 SVM1.write (90);                          // Gira o Servo com o sensor a 90 graus
SVM2.write (0);                           // Gira o Servo com o sensor a 0 graus
delay (5);
 distancia();
 delay (5);

//verifica obstaculo
if (dist < 40 or dist2 < 40) {                          // Se há obstáculo encontrado a menos
de 40cm.
    direcao () ;                          // Se Frente estiver bloqueado, mude de direção
  }
else if (dist >= 40 or dist >= 40)  {                   // Se o obstáculo for encontrado entre
a mais de 25cm
    FORWARD ();                           // Robô se move para a direção da Frente.
  }
}

 //estudo de qual é o melhor direcionamento
void direcao () {
  STOP ();                          // O robô Para
  BACK ();                          // O robô vai para tras
  STOP ();                          // O robô Para
```

```
  //esquerda
 SVM2.write (90);                          // Gira o Servo com o sensor a 90 graus
  delay (1000);
  distancia();
  delay(10);
  DistanciaEsquerda = dist2 ;
  delay(500);
    Serial.print ("DISTANCIA ESQUERDA = ");
  Serial.print (DistanciaEsquerda);
 Serial.println();  // MOSTRA O VALOR NA SERIAL

//diagonalesquerda
 SVM2.write (45);                           // Gira o Servo com o sensor a 45 graus
  delay (1000);
  distancia();
  delay(10);
  DistanciaDiagonalEsquerda = dist2;
   delay (500);
     Serial.print ("DISTANCIA DIAGONAL ESQUERDA = ");
  Serial.print (DistanciaDiagonalEsquerda);
 Serial.println();  // MOSTRA O VALOR NA SERIAL

   //direita
   SVM1.write (0);
    delay (1000);
    distancia();
    delay(10);
    DistanciaDireita = dist ;
    delay (500);
      Serial.print ("DISTANCIA DIREITA = ");
  Serial.print (DistanciaDireita);
 Serial.println();  // MOSTRA O VALOR NA SERIAL

    //diagonaldireita
   SVM1.write (45);
    delay (1000);
    distancia();
    delay(10);
    DistanciaDiagonalDireita = dist;
    delay (500);
     Serial.print ("DISTANCIA DIAGONAL DIREITA = ");
  Serial.print (DistanciaDiagonalDireita);
 Serial.println();  // MOSTRA O VALOR NA SERIAL

     CompareDistance ();                    // Encontre a distância mais longa.
}

// Função para calcular qual a distancia é melhor para o robô ir
void CompareDistance ()
{
```

```
 if (DistanciaDireita > DistanciaEsquerda and DistanciaDireita > DistanciaDiagonalDireita and DistanciaDireita > DistanciaDiagonalEsquerda)
 {      // Se a direita está menos obstruída.
   SUPERRIGHT ();                    // O robô vai andar lateralmenta pela direita
 }
 else if (DistanciaEsquerda > DistanciaDireita and DistanciaEsquerda > DistanciaDiagonalDireita and DistanciaEsquerda > DistanciaDiagonalEsquerda)
 {      // Se a esquerda está menos obstruída.
   SUPERLEFT ();                    // O robô vai andar lateralmenta pela direita
 }
 else if  (DistanciaDiagonalDireita > DistanciaDireita and DistanciaDiagonalDireita > DistanciaEsquerda and DistanciaDiagonalDireita > DistanciaDiagonalEsquerda)
 {      // Se a diagonal direita está menos obstruída.
   FORWARDRIGHT ();                  // O robô vai andar pela diagonal direita
 }
  else if (DistanciaDiagonalEsquerda > DistanciaDireita and DistanciaDiagonalEsquerda > DistanciaDiagonalDireita and DistanciaDiagonalEsquerda > DistanciaEsquerda)
 {      // Se a diagonal esquerda está menos obstruída.
   FORWARDLEFT ();                   // O robô vai andar pela diagonal esquerda
 }

   else if (DistanciaDiagonalEsquerda < 10 and DistanciaDiagonalDireita < 10 and DistanciaDireita < 10 and DistanciaEsquerda < 10)
   {
     BACK ();
     }
}
```

```
void distancia()
{
 //SENSOR ULTRASONICO
  delayMicroseconds(5); // TEMPO 5uS

  digitalWrite(SU1TRIG, 1);  // INCREMENTA O TRIGGER
  delayMicroseconds(10); // TEMPO 10uS
  digitalWrite(SU1TRIG, 0);

  dist = pulseIn(SU1ECHO, 1); // RETORNO DO SINAL
  dist = dist/58; // CALCULA DISTANCIA EM CM
```

```
  Serial.print ("DISTANCIA = ");
  Serial.print (dist);
  Serial.print (" cm");
  Serial.println();  // MOSTRA O VALOR NA SERIAL
  delay (100); // TEMPO 1S

//SENSOR ULTRASONICO 2
  delayMicroseconds(5); // TEMPO 5uS

  digitalWrite(SU2TRIG, 1);  // INCREMENTA O TRIGGER
  delayMicroseconds(10); // TEMPO 10uS
  digitalWrite(SU2TRIG, 0);

  dist2 = pulseIn(SU2ECHO, 1); // RETORNO DO SINAL
  dist2 = dist2/58; // CALCULA DISTANCIA EM CM

  Serial.print ("DISTANCIA2 = ");
  Serial.print (dist2);
  Serial.print (" cm");
  Serial.println();  // MOSTRA O VALOR NA SERIAL
  delay (100); // TEMPO 1S
}


// Função para fazer o carro parar
void STOP()
{
  Serial.println("Robô: Parar ");
  digitalWrite(PH1IN1, LOW);
  digitalWrite(PH1IN2, LOW);
  digitalWrite(PH1IN3, LOW);
  digitalWrite(PH1IN4, LOW);
  digitalWrite(PH2IN1, LOW);
  digitalWrite(PH2IN2, LOW);
  digitalWrite(PH2IN3, LOW);
  digitalWrite(PH2IN4, LOW);
  delay(1000);                              //aguarda um tempo
}

// Função para fazer o robô andar para frente
void FORWARD()
{
  Serial.println("Robô: Frente ");
  digitalWrite(PH1IN1, HIGH);
  digitalWrite(PH1IN2, LOW);
  digitalWrite(PH1IN3, LOW);
  digitalWrite(PH1IN4, HIGH);
  digitalWrite(PH2IN1, HIGH);
  digitalWrite(PH2IN2, LOW);
```

```
  digitalWrite(PH2IN3, LOW);
  digitalWrite(PH2IN4, HIGH);
   delay(100);
}

// Função que faz o robô andar para trás
void BACK()
{
  Serial.println("Robô: Ré ");
  digitalWrite(PH1IN1, LOW);                    //Configurar a ponte h
  digitalWrite(PH1IN2, HIGH);
  digitalWrite(PH1IN3, HIGH);
  digitalWrite(PH1IN4, LOW);
  digitalWrite(PH2IN1, LOW);
  digitalWrite(PH2IN2, HIGH);
  digitalWrite(PH2IN3, HIGH);
  digitalWrite(PH2IN4, LOW);
  delay(1000);                                  //aguarda um tempo
}


//Função para o robô andar lateralmente para esquerda
void SUPERLEFT () {
  Serial.println("Robô: Superleft ");
  digitalWrite(PH1IN1, LOW);                    //Configurar a ponte h
  digitalWrite(PH1IN2, HIGH);
  digitalWrite(PH1IN3, LOW);
  digitalWrite(PH1IN4, HIGH);
  digitalWrite(PH2IN1, HIGH);
  digitalWrite(PH2IN2, LOW);
  digitalWrite(PH2IN3, HIGH);
  digitalWrite(PH2IN4, LOW);
  delay (700);                                  //aguarda um tempo

}

//Função para o robô andar lateralmente para direita
void SUPERRIGHT () {
  Serial.println("Robô: Superright ");
  digitalWrite(PH1IN1, HIGH);                    //Configurar a ponte h
  digitalWrite(PH1IN2, LOW);
  digitalWrite(PH1IN3, HIGH);
  digitalWrite(PH1IN4, LOW);
  digitalWrite(PH2IN1, LOW );
  digitalWrite(PH2IN2, HIGH);
  digitalWrite(PH2IN3, LOW);
  digitalWrite(PH2IN4, HIGH);
  delay (1000);                                 //aguarda um tempo
}
```

```
//Função para o robô andar para frente e para direita (diagonal)
void FORWARDRIGHT () {
  Serial.println("Robô: Forwardright ");
  digitalWrite(PH1IN1, HIGH);
  digitalWrite(PH1IN2, LOW);
  digitalWrite(PH1IN3, LOW);
  digitalWrite(PH1IN4, LOW);
  digitalWrite(PH2IN1, LOW);
  digitalWrite(PH2IN2, LOW);
  digitalWrite(PH2IN3, LOW);
  digitalWrite(PH2IN4, HIGH);
  delay (1000);                         //aguarda um tempo
}

//Função para o robô andar para frente e para esquerda (diagonal)
void FORWARDLEFT () {
  Serial.println("Robô: Forwardleft ");
  digitalWrite(PH1IN1, LOW);
  digitalWrite(PH1IN2, LOW);
  digitalWrite(PH1IN3, LOW);
  digitalWrite(PH1IN4, HIGH);
  digitalWrite(PH2IN1, HIGH);
  digitalWrite(PH2IN2, LOW);
  digitalWrite(PH2IN3, LOW);
  digitalWrite(PH2IN4, LOW);
  delay (1000);                         //aguarda um tempo
}
```