

**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA
SOUZA**

ETEC SYLVIO DE MATTOS CARVALHO

Curso Técnico em Mecatrônica M-tec

GUILHERME HENRIQUE RUFFO SOARES

LAYANE KAROLINE SILVA LEAL

MARIA ISABELLA CASSEMIRO

MATHEUS HENRIQUE MUNHOZ

**ROBÔ NO RINGUE: DESENVOLVENDO TECNOLOGIA PARA O
SUMÔ**

Matão, SP

2024

1	INTRODUÇÃO.....	3
2	OBJETIVOS (GERAL E ESPECÍFICOS).....	4
3	DESCRIÇÃO DO PROJETO	5
3.1	Apresentação Geral do Projeto	5
3.2	Materiais Utilizados e suas Especificações Técnicas.....	5-10
4	CRONOGRAMA DE ATIVIDADES	11
4.1	Atividades e Metas 1º Semestre de 2024	11
4.2	Atividades e Metas 2º Semestre de 2024	12
5.	DESENVOLVIMENTO DO PROJETO	13
5.1	Descrição do Funcionamento.....	13-15
5.2	Montagem Física do Projeto	15-20
5.3	Desenvolvimento do programa do robô	21
6	CONSIDERAÇÕES FINAIS	22
	APÊNDICE	23-30

1 INTRODUÇÃO

Este documento apresenta o Projeto Técnico ROBÔ NO RINGUE: DESENVOLVENDOTECHNOLOGIA PARA O SUMÔ, referente ao Trabalho de Conclusão de Curso elaborado pela equipe de estudantes do curso de Mecatrônica. O projeto foi escolhido por proporcionar aos alunos a oportunidade de aplicar e aprimorar os conhecimentos adquiridos ao longo do curso, especialmente nas áreas de robótica e automação, fundamentais para a formação de um profissional capacitado na atual indústria 4.0. Além disso, o projeto visa estimular a resolução de problemas complexos, o trabalho em equipe e o desenvolvimento de soluções práticas e inovadoras, preparando os estudantes para desafios profissionais futuros.

A proposta deste projeto envolve o desenvolvimento de um **robô de sumô**, um tipo de robô projetado para competir em lutas contra outros robôs, com o objetivo principal de empurrar o oponente para fora de uma área delimitada. O robô será desenvolvido com foco na parte mecânica e eletrônica, integrando sistemas de movimentação e controle. O projeto visa criar uma estrutura robusta e eficiente, capaz de competir de forma eficaz em torneios de robôs de sumô.

Este tipo de robô exige o desenvolvimento de sistemas de movimentação, com rodas motorizadas e um sistema de alimentação de energia, além de uma estrutura física resistente o suficiente para suportar os impactos durante as competições. A construção do robô envolve o dimensionamento adequado de sua estrutura, a escolha de materiais para garantir resistência e leveza, e o desenvolvimento de circuitos de controle para gerenciar a movimentação.

Este documento encontra-se assim dividido:

- Capítulo 2: Apresentação dos objetivos geral e específicos deste projeto.
- Capítulo 3: Descrição geral do projeto, incluindo a lista de materiais utilizados para a montagem física do robô de sumô.
- Capítulo 4: Apresentação dos cronogramas de atividades planejados e cumpridos durante os 1º e 2º semestres de 2024, que orientaram a execução do projeto.
- Capítulo 5: Descrição detalhada do projeto físico, incluindo o funcionamento do robô, a montagem de seus componentes e o desenvolvimento do programa aplicativo que controla suas ações.

- Capítulo 6: Considerações finais, discutindo as dificuldades encontradas ao longo do desenvolvimento, as soluções adotadas para superá-las e os resultados obtidos com a conclusão do projeto.
- Referências Bibliográficas: Relacionamento de toda a bibliografia utilizada para o embasamento teórico e prático durante o desenvolvimento deste projeto.

2 OBJETIVOS (GERAL E ESPECÍFICOS)

2.1 Objetivo Geral

O objetivo geral do projeto é:

- Construir um robô sumô.

2.2 Objetivos Específicos

Os objetivos específicos do projeto são:

- Definir os componentes ideais para o robô;
- Elaborar o projeto estrutural do robô;
- Montar o robô, efetuando as conexões dos seus componentes;
- Desenvolver a programação do robô.

3 DESCRIÇÃO DO PROJETO

A tabela 1 mostra os componentes e a quantidade que utilizamos para desenvolvimento do projeto.

Tabela 1 – Lista de Materiais

Material	Quantidade
Arduino Nano	1
Ponte h dupla L298n	1
Bateria pack 12 v	1
Motor de Corrente Continua	2
Bluetooth HC-05	1

FONTE: Elaborado Pelos Autores (2024).

3.1 ARDUINO NANO

O Arduino Nano é uma placa de desenvolvimento compacta baseada no microcontrolador ATmega328P. As principais características são: Microcontrolador: ATmega328P (8 bits, 16 MHz). Memória: 32 KB de Flash, 2 KB de SRAM, 1 KB de EEPROM. I/O: 14 pinos digitais (6 com PWM), 8 pinos analógicos (A0-A7). Comunicação: USB para programação, Serial (RX/TX), I2C (A4, A5), SPI (D10-D13). Alimentação: 5V via USB ou VIN (6-12V).

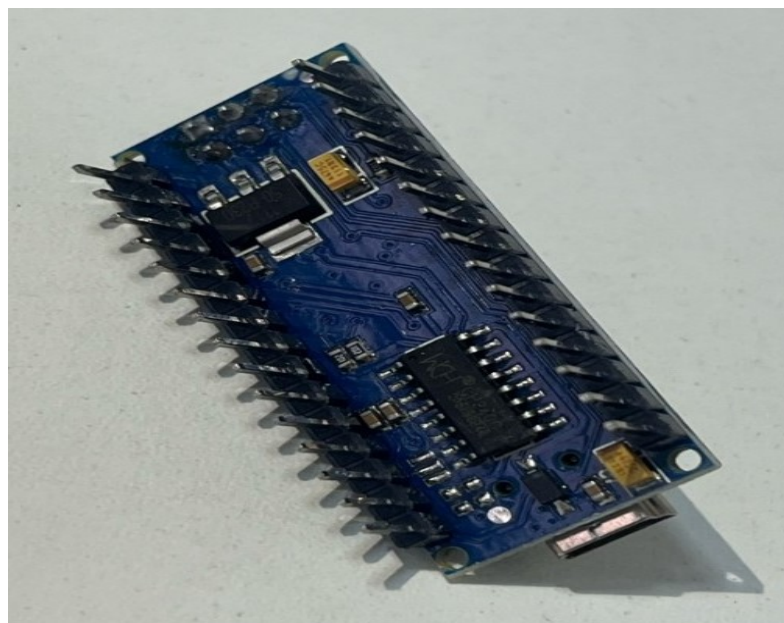


Figura 1: Arduino Nano ATmega328

FONTE: Autoria Própria (2024)

3.2 Ponte H Dupla L298N

A Ponte H Dupla L298N é um circuito integrado utilizado para controlar motores de corrente contínua (DC) e motores de passo, permitindo o controle da direção e da velocidade desses motores. Ela possui duas pontes H internas, o que a torna ideal para controlar dois motores de forma independente.

Principais características:

Tensão de operação: 4,5 V a 46 V (para os motores).

Corrente máxima: Até 2 A por canal (contínuo), com pico de 3 A.

Controle: Permite controle bidirecional (frente e reverso) e controle de velocidade (via PWM).

Proteção: Possui proteção térmica e proteção contra sobrecorrente, o que aumenta a durabilidade e segurança do circuito.

Aplicações: Usada principalmente em robótica, controle de movimentação de robôs e automação, além de ser muito comum em projetos que envolvem motores DC.

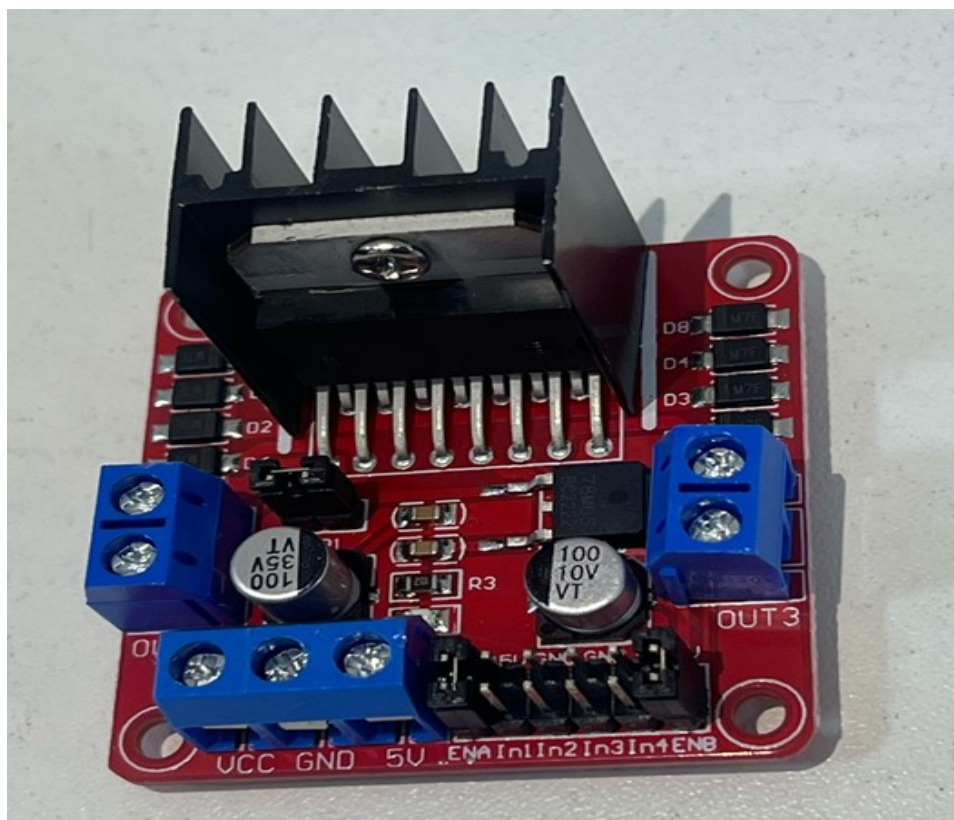


Figura 2: Ponte H Dupla L298N
FONTE: Autoria Própria (2024)

3.3 Bateria Pack 12 V;

A bateria 12V 5200mAh é uma escolha bastante adequada para projetos de robôs sumô, pois oferece uma boa combinação de tensão e capacidade que atende às necessidades típicas desses robôs, como alta potência para motores e suficiente autonomia para competições curtas. Além disso, dependendo do tipo de bateria (Li-ion ou LiPo), ela também pode proporcionar uma taxa de descarga alta, o que é importante para garantir um desempenho consistente durante as lutas. No entanto, é importante considerar o peso e o espaço disponível no robô, além de manter a bateria em boas condições para garantir a longevidade e o desempenho durante o uso.



Figura 3: Bateria Pack 14.4v
FONTE: Joacir de Oliveira Silva (2024)

3.4 Motor de Corrente Contínua

Um motor de corrente contínua (CC) 3-6V converte energia elétrica em movimento rotacional usando um campo magnético gerado pelas bobinas no rotor e pelos ímãs permanentes no estator. Quando uma voltagem DC é aplicada aos terminais do motor, a corrente flui pelas bobinas, gerando um campo magnético que interage com o campo do estator, fazendo o rotor girar. A direção de rotação pode ser invertida trocando a polaridade da corrente, e a velocidade é controlada pela variação da voltagem ou usando modulação por largura de pulso (PWM).

Terminais:

Positivo (+): Conectado ao polo positivo da fonte de alimentação.

Negativo (-): Conectado ao polo negativo da fonte.



Figura 4: Motor Corrente Continua
FONTE: Tallis Mendonça Santos e Loretron (2024)

3.5 Bluetooth HC-05

O módulo Bluetooth HC-05 é um dispositivo utilizado para comunicação sem fio entre microcontroladores (como Arduino) e outros dispositivos via Bluetooth. Ele opera com a tecnologia Bluetooth 2.0 e pode ser configurado como mestre (master) ou escravo (slave). Sua principal aplicação é em projetos de robótica, automação e controle remoto. A comunicação ocorre por meio de uma interface UART (RX/TX), e ele pode ser alimentado com tensões de 3,3V a 6V, com uma distância de operação de até 10 metros.

O módulo Bluetooth HC-05 é de grande importância em projetos de eletrônica e robótica, pois possibilita a comunicação sem fio entre dispositivos como microcontroladores (como o Arduino), smartphones, computadores e outros módulos Bluetooth. Sua capacidade de estabelecer uma conexão sem fio é essencial para muitos tipos de aplicações que exigem controle remoto ou monitoramento de dispositivos.

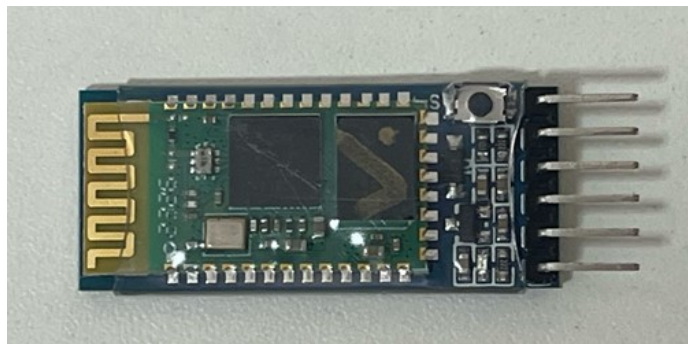


Figura 5: Bluetooth HC-05
FONTE: Autoria Própria (2024)

3.6 Roda

A Roda Amarela de 68mm com pneu plástico, ótimo para aplicações de robótica

Especificações:

- Diâmetro: 6,8cm;
- Largura: 2,6cm;
- Furo central: 5,3 x 3,66mm (Semicírculo);
- Peso: 50g



Figura 6: Roda
FONTE: Autoria Própria (2024)

3.7 Chassi

Especificações do Chassi:

- Altura: 11 cm
- Comprimento: 20,5 cm
- Largura: 18 cm
- Material do Chassi: MDF (Medium Density Fiberboard)

O chassi é composto por duas placas de MDF, com espessura de 2,5 cm, sendo um material amplamente utilizado em robótica pela sua rigidez e facilidade de corte.

Detalhes da Parte Frontal:

Na parte frontal do robô, há uma placa de MDF inclinada com as seguintes dimensões:

- Altura: 13 cm

- Comprimento: 18 cm
- Espessura: 2,5 cm

Esta placa inclinada contém uma placa reta de alumínio, fixada sobre ela, com as seguintes especificações:

- Largura: 3 cm
- Comprimento: 18 cm
- Espessura: 0,1 cm

Observações:

MDF: Material utilizado devido à sua fácil manipulação e boas propriedades mecânicas para suportar componentes leves e moderados, mas que pode ser menos resistente a impactos fortes.

Alumínio: O uso do alumínio na parte frontal do robô pode melhorar a resistência ao desgaste e a leveza da estrutura.



Figura 7: Chassi (modelo)

FONTE: Autoria própria (2024)

4 CRONOGRAMA DE ATIVIDADES

4.1 Atividades e Metas 1º Semestre de 2024

No primeiro semestre planejamos a escolha dos integrantes como também dos itens que utilizamos ao longo do ano, a separação de afazeres de cada pessoa e a realização do boneco.

Tabela 2: Cronograma de atividades e metas no 1º Semestre de 2024

Atividade	Fev.		Março		Abril		Maio		Junho	
	1ª Quinzena	2ª Quinzena	1ª Quinzena	2ª Quinzena	1ª Quinzena	2ª Quinzena	1ª Quinzena	2ª Quinzena	1ª Quinzena	2ª Quinzena
Montagem da equipe	X									
Definição do nome do robô		X								
Definição dos objetivos do projeto		X	X							
Definição da lista de materiais / orçamentos				X	X					
Levantamento de dados					X	X	X	X	X	X
Escrita da parte teórica						X	X	X	X	X
Análise dos dados resultados					X	X	X	X	X	X
Discussão dos resultados								X	X	X
Elaboração da conclusão e referências									X	X
Batalha de Robôs										X

FONTE: Elaborado pelos autores (2024).

4.2 Atividades e Metas 2º Semestre de 2024

No segundo semestre realizamos a compra dos itens do robô, fazendo testes e ajustes para sua melhoria, finalizando o boneco de acordo com as normas BNT, além disso, ensaiando e fotografando para os resultados finais.

Tabela 3: Cronograma de atividades e metas no 2º Semestre de 2024

Atividade	Julho		Agosto		Setembro		Outubro		Novembro	
	1ª Quinzena	2ª Quinzena	1ª Quinzena	2ª Quinzena	1ª Quinzena	2ª Quinzena	1ª Quinzena	2ª Quinzena	1ª Quinzena	2ª Quinzena
Entrega final do boneco										X
Revisão do diário de bordo/Boneco		X	X	X	X	X	X	X	X	X
Ensaio para apresentação								X	X	
Teste do robô								X	X	X
Finalização do diário de bordo/boneco							X	X	X	X
Finalização do robô								X	X	X
Apresentação do TCC finalizado										X

FONTE: Elaborado pelos autores (2024).

5 DESENVOLVIMENTO DO PROJETO

5.1 Descrição do Funcionamento

A Figura 8 apresenta o esquema elétrico de toda a parte eletrônica do projeto.

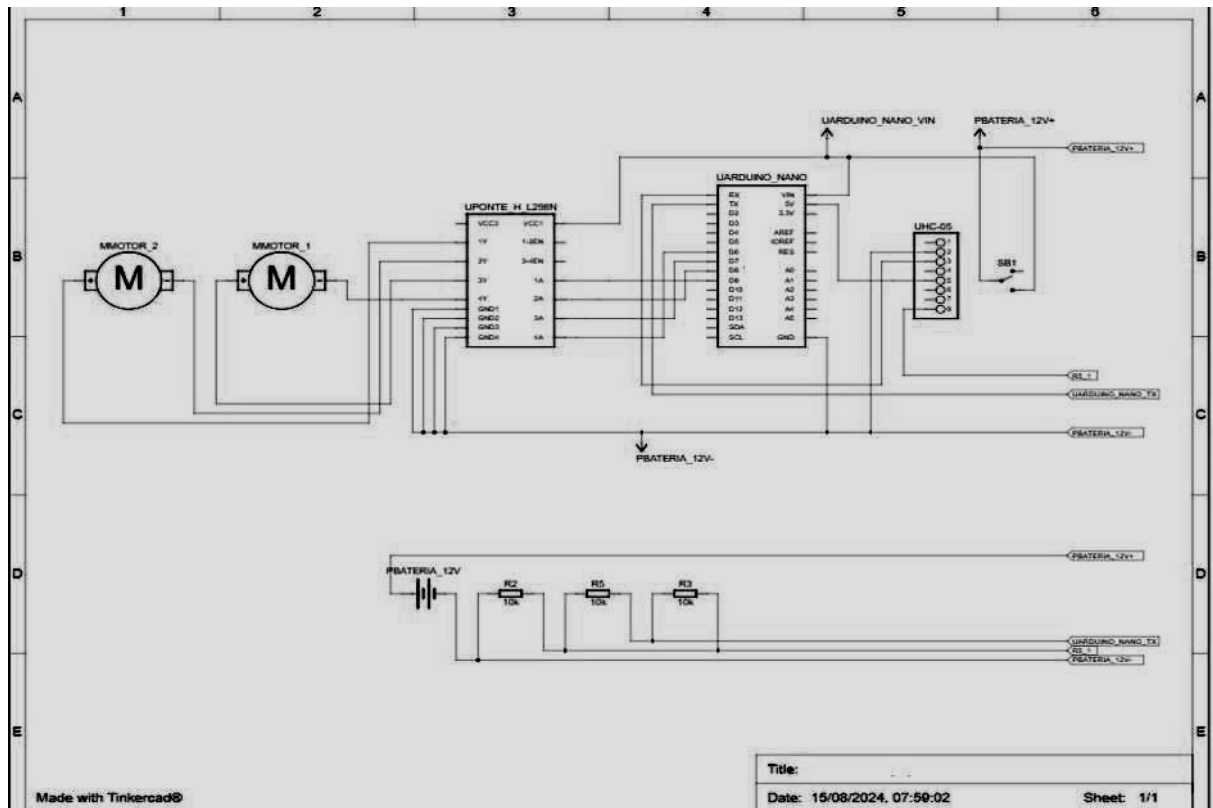


Figura 8 – Esquema elétrico do robô
FONTE: Elaborado pelos autores (2024).

1. Motores (MOTOR_1 e MOTOR_2):
 - Esses motores estão conectados ao driver de ponte H (L298N), que permite controlar a direção e a velocidade dos motores ao aplicar tensões em diferentes sentidos nos terminais.
2. Driver de Ponte H (L298N):
 - Esse componente recebe sinais do Arduino Nano para controlar os motores. O L298N tem quatro pinos de entrada (IN1, IN2, IN3, IN4), que recebem sinais lógicos do Arduino para controlar o movimento dos motores.

- Também há conexões de alimentação: o L298N é alimentado pela bateria de 12V, o que permite fornecer energia suficiente para os motores.

3. Arduino Nano:

- O Arduino Nano está conectado ao driver de ponte H e ao módulo Bluetooth (HC-05). Ele executa o código que controla os motores com base nos sinais recebidos do módulo Bluetooth.
- O pino VIN do Arduino Nano está conectado à bateria de 12V, que alimenta o sistema.

4. Módulo Bluetooth (HC-05):

- Esse módulo permite comunicação sem fio com o Arduino Nano, possibilitando o controle remoto.
- Ele está conectado aos pinos RX e TX do Arduino, permitindo enviar e receber comandos de controle.

5. Botão e Resistores:

- O botão, junto com os resistores, provavelmente funciona como um reset ou controle adicional. Ao pressionar o botão, o sinal pode ser lido pelo Arduino para ativar ou desativar certas funções.

— Funcionamento Elétrico do Sistema

1. Alimentação:

- Quando a bateria de 12V está conectada, ela alimenta tanto o Arduino Nano quanto o driver de ponte H (L298N). O Arduino Nano, por sua vez, controla os motores e o módulo Bluetooth.

2. Acionamento do Botão:

- Quando o botão é pressionado, ele envia um sinal para o Arduino. Dependendo da programação, esse botão pode ser configurado para inicializar, resetar, ou acionar funções específicas do robô.

3. Arduino e Programação:

- O Arduino Nano executa o código carregado, que interpreta os sinais do módulo Bluetooth e envia comandos ao driver de ponte H para controlar os motores. Com base nos sinais Bluetooth recebidos, o Arduino decide a direção e a velocidade dos motores.

- Se o código estiver programado para controlar a direção do robô (frente, trás, esquerda, direita), o Arduino ajustará os sinais lógicos enviados ao L298N, que alterará a polaridade dos motores e controlará sua velocidade.

4. Comunicação com o Bluetooth:

- Quando o módulo Bluetooth HC-05 recebe um comando (por exemplo, de um celular ou controle remoto), ele envia os dados ao Arduino através da comunicação serial (pinos RX e TX).

- O Arduino interpreta esses comandos e, conforme programado, aciona os motores para movimentar o robô na direção desejada.

5.2 Montagem Física do Projeto

As Figuras 9 a 12 apresentam a sequência da montagem física do nosso projeto.



Figura 9 – Peças utilizadas no robô
 FONTE: Elaborado pelos autores (2024).

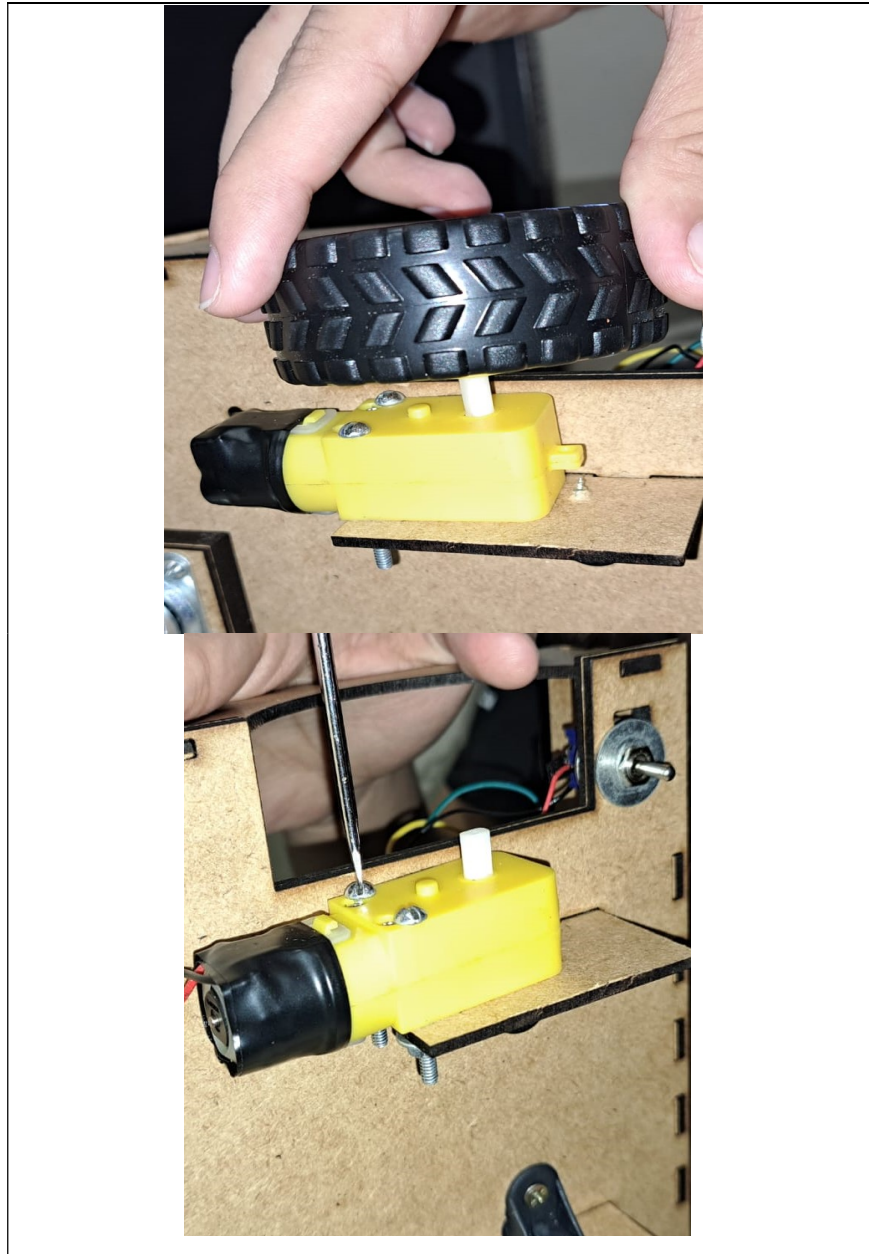


Figura 10 – Instalação das rodas e motores no chassi
FONTE: Elaborado pelos autores (2024).

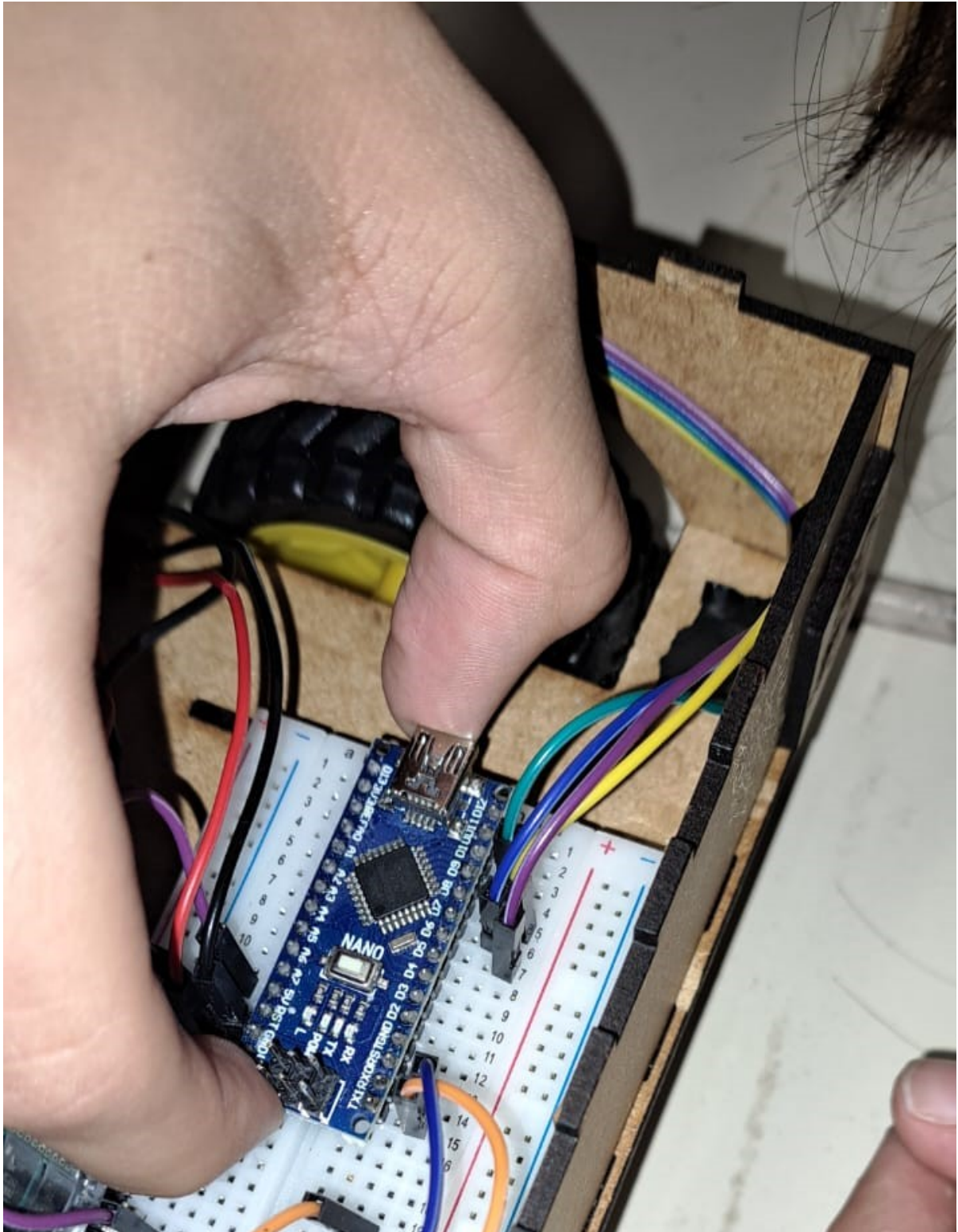


Figura 11 – Instalação do Arduino e módulos no chassi
FONTE: Elaborado pelos autores (2024).

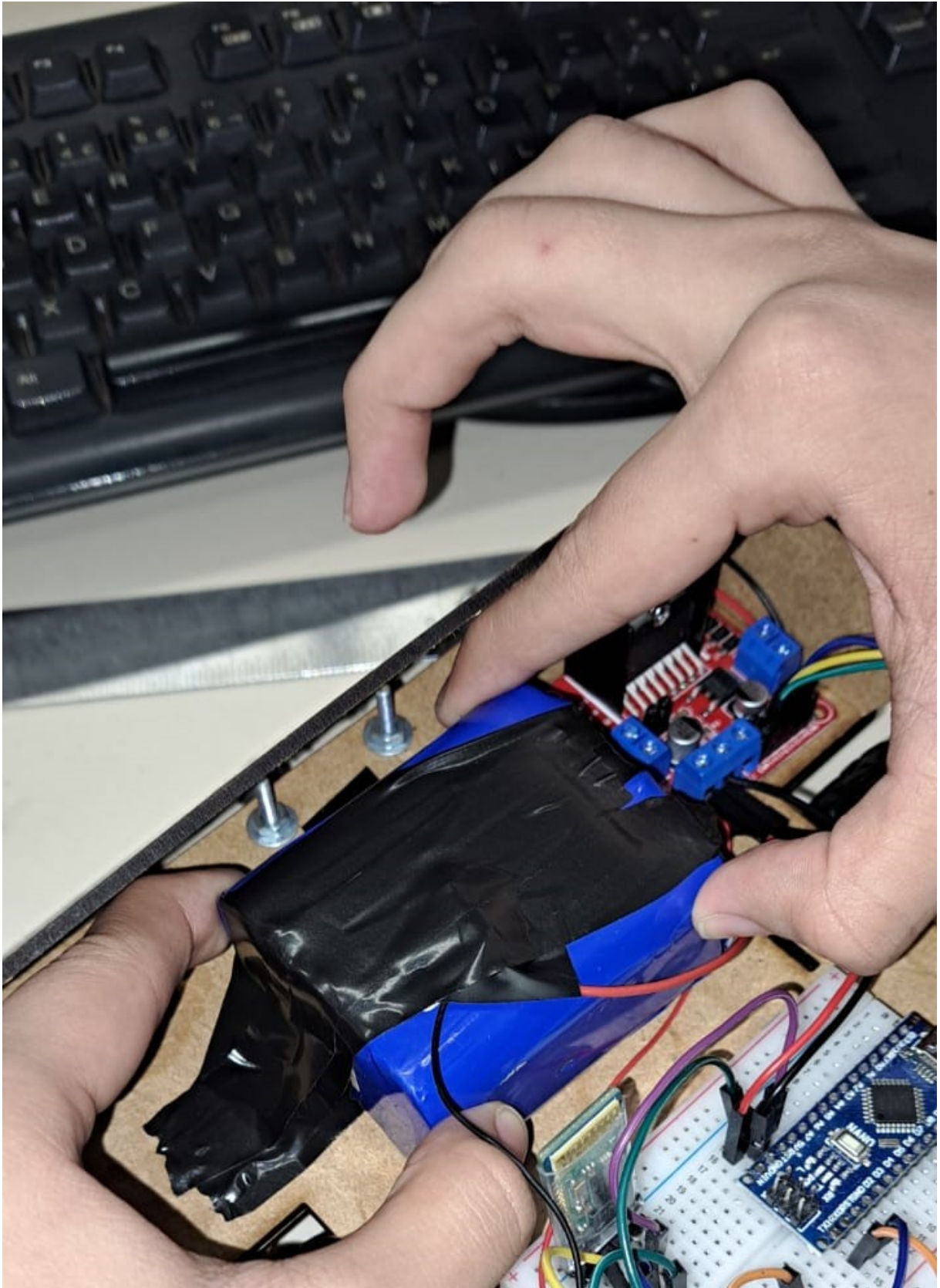


Figura 12 – Instalação da bateria no chassi
FONTE: Elaborado pelos autores (2024).

As Figuras 13 e 14 apresentam os testes finais efetuados em nosso projeto.



Figura 13 – Teste do hardware
FONTE: Elaborado pelos autores (2024).

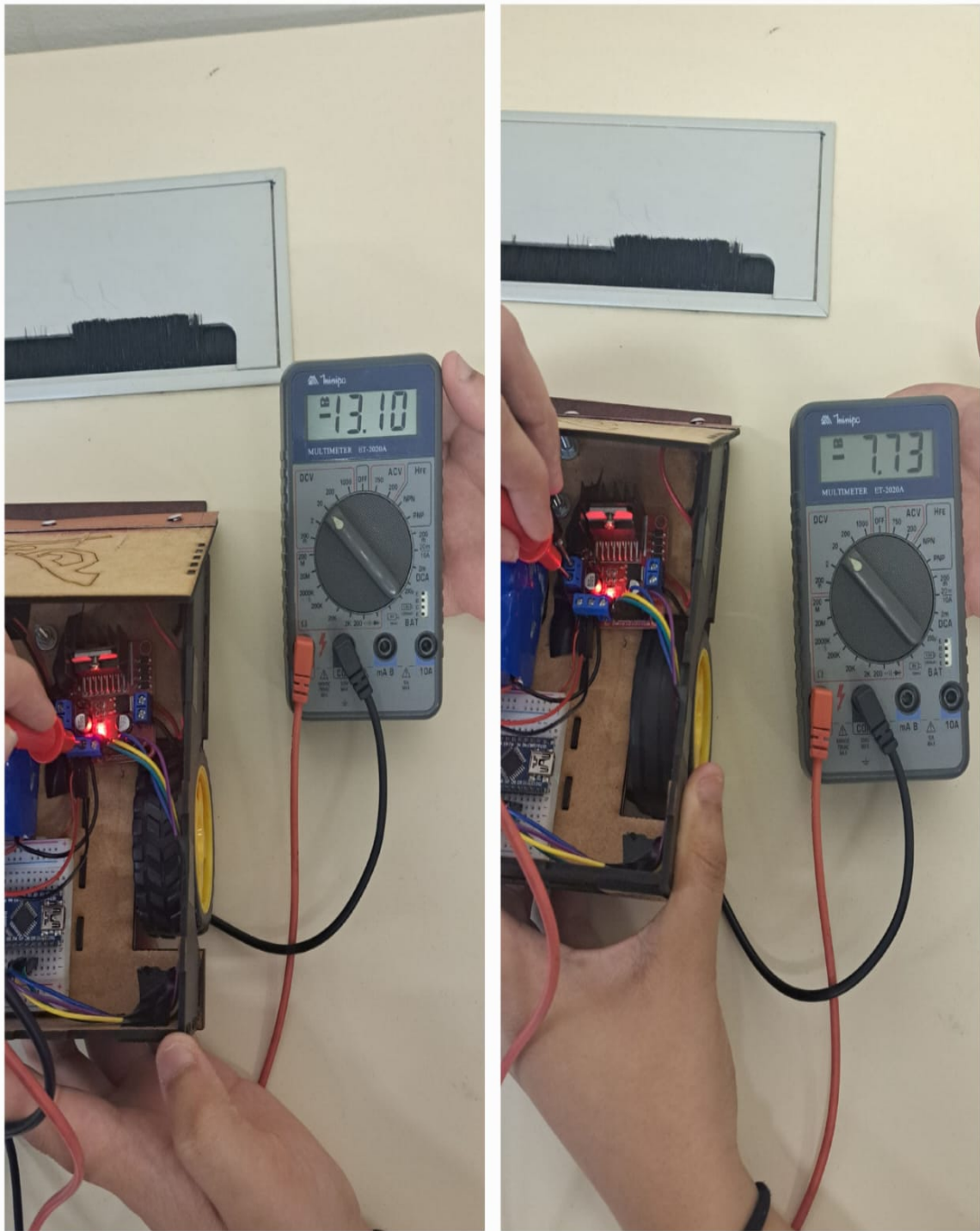


Figura 14 – Teste geral do robô
FONTE: Elaborado pelos autores (2024).

5.3 Desenvolvimento do programa do robô

A programação do robô foi desenvolvida no aplicativo Arduino IDE versão 1.8.5.

A Figura 15 apresenta o fluxograma do programa.

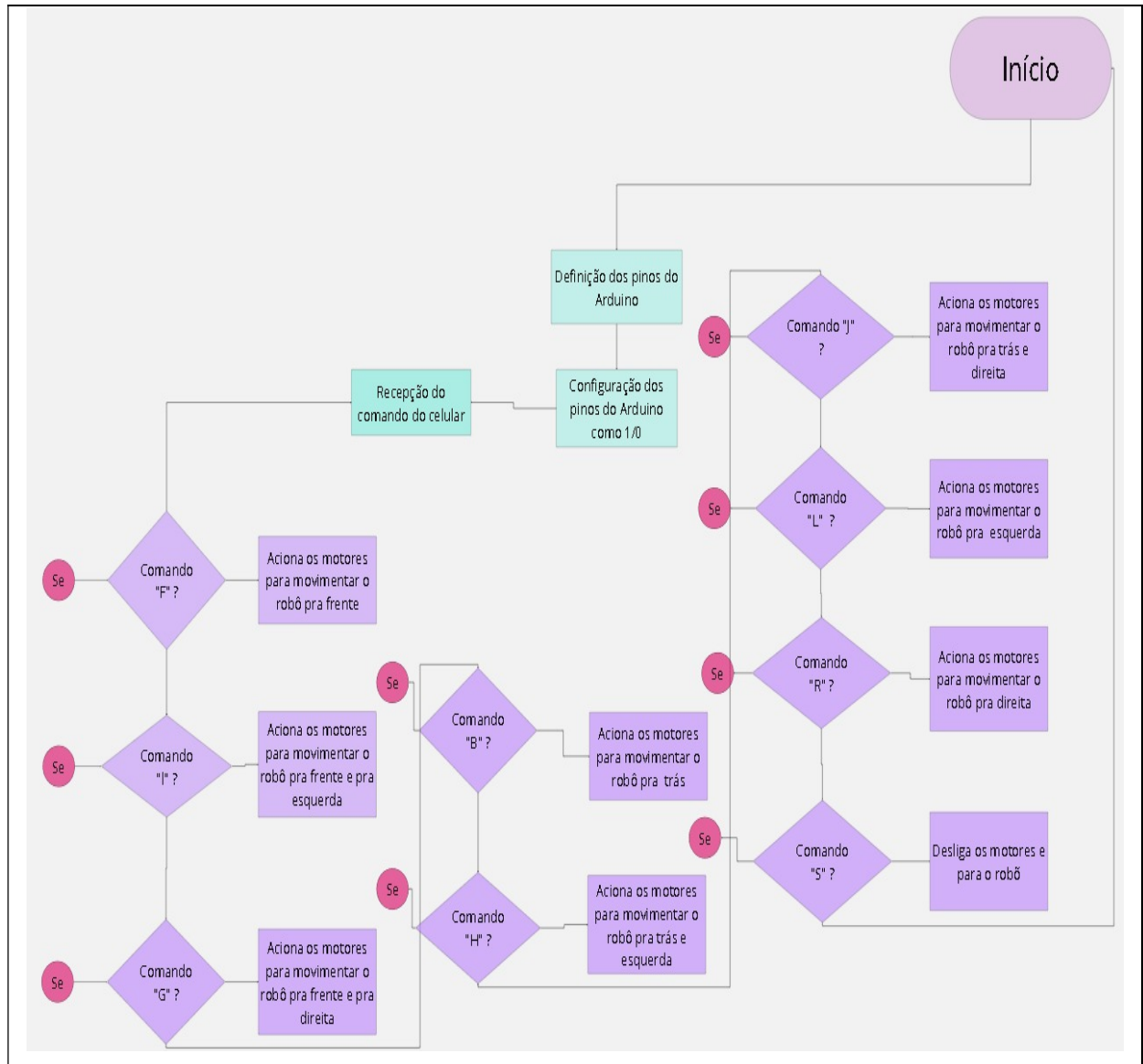


Figura 15 – Fluxograma do programa do robô
FONTE: Elaborado pelos autores (2024).

Conforme apresentado no fluxograma o programa é executado de forma contínua dentro do loop principal, onde o Arduino aguarda dados do módulo Bluetooth. Quando os dados são recebidos, o comando é processado e, dependendo do valor do comando, o carro é movimentado de acordo com as direções solicitadas. O código fonte desenvolvido no Arduino IDE é apresentado no Apêndice A.

6 CONSIDERAÇÕES FINAIS

O desenvolvimento deste projeto de robô sumô, controlado via Bluetooth com Arduino, foi uma experiência de grande aprendizado no campo da robótica e automação. O principal objetivo foi criar um robô capaz de competir em uma arena de sumô, onde a missão é empurrar o robô adversário para fora da área delimitada. O projeto envolveu a construção do robô, a programação de seu controle via Bluetooth e a realização de testes para otimizar seu desempenho.

Durante a montagem do robô, a construção do chassi e a instalação dos motores e do módulo Bluetooth não apresentaram grandes dificuldades. A integração dos componentes foi realizada de forma direta, e o uso do driver L298 para controlar os motores proporcionou a funcionalidade necessária para movimentar o robô de forma eficiente. A programação no Arduino IDE também foi uma etapa relativamente tranquila, já que o controle do robô foi implementado por meio de comandos simples, como avanço, retrocesso, e movimentos laterais, usando o módulo Bluetooth para enviar comandos.

O maior desafio encontrado foi, na verdade, na elaboração dos arquivos e na programação do controle do robô. A criação da lógica de movimentação do robô sumô exigiu ajustes constantes para garantir que ele respondesse corretamente aos comandos enviados via Bluetooth e se comportasse de forma eficiente durante a competição. A programação envolveu o ajuste da velocidade, direção e a resposta adequada a cada comando recebido, como frente, trás, esquerda, direita e parar. Além disso, o controle do robô de forma precisa, para que ele não saísse da arena ou ficasse em desvantagem, também exigiu algumas iterações.

A parte mais trabalhosa foi a organização do código e da documentação, uma vez que cada função de controle precisava ser claramente definida e documentada para garantir o funcionamento adequado. A elaboração do fluxograma para representar o funcionamento do sistema foi importante para esclarecer o processo de tomada de decisão do robô, além de permitir uma melhor visualização da lógica implementada.

REFERÊNCIAS BIBLIOGRÁFICAS

1. ARDUINONANO. Datasheet. ETC. Disponível em: <<https://html.alldatasheet.com/html-pdf/1424859/ETC/ARDUINONANO/66/1/ARDUINONANO.html>>. Acesso em: 20 nov. 2024.
2. DBF81H903. Datasheet bluetooth. SOSHIN. Disponível em: <<https://html.alldatasheet.com/html-pdf/1179463/SOSHIN/DBF81H903/253/1/DBF81H903.html>>. Acesso em: 20 nov. 2024.
3. L298N. Datasheet. STMICROELECTRONICS. Disponível em: <<https://html.alldatasheet.com/html-pdf/22440/STMICROELECTRONICS/L298N/1619/1/L298N.html>>. Acesso em: 20 nov. 2024.
4. MOTORDC. Datasheet. ETC. Disponível em: <<https://html.alldatasheet.com/html-pdf/96347/ETC/MOTORDC/46/1/MOTORDC.html>>. Acesso em: 20 nov. 2024.
5. KURI, T.; MARTIN, D. Robótica: princípios, técnicas e aplicações em automação. Campinas: Editora Unicamp, 2018.
6. MILLER, R. L. Fundamentos de eletrônica para robótica. 2. ed. Porto Alegre: Bookman, 2019.
7. MICROCHIP TECHNOLOGY INC. ATmega328P: 8-bit microcontroller with 32K bytes of in-system programmable Flash. 2019. Disponível em: <<https://www.microchip.com/wwwproducts/en/ATmega328P>>. Acesso em: 13 set. 2024.

8. MIRO. Miro - A plataforma de colaboração visual. Miro. Disponível em: <<https://miro.com/pt/>>. Acesso em: 21 set. 2024.

APÊNDICE

Programa do Robô

// CARRO CONTROLADO POR BLUETOOTH

// === Pinos utilizados no driver L298 === //

Apesar dessas dificuldades, o projeto foi concluído com êxito, e o robô demonstrou um desempenho satisfatório durante os testes, sendo capaz de executar os movimentos básicos necessários para competir em uma arena de sumô. A experiência adquirida foi fundamental para a compreensão da interação entre hardware e software em robôs controlados remotamente, além de proporcionar aprendizado valioso sobre a programação de controle de motores e a utilização de módulos Bluetooth.

```
#define ENA 10    //o pino ENA do Driver será ligado ao pino 10 do arduino (motor da esquerda)
```

```
#define IN1 9     //o pino IN1 do Driver será ligado ao pino 9 do arduino
```

```
#define IN2 8     //o pino IN2 do Driver será ligado ao pino 8 do arduino
```

```
#define IN3 7     //o pino IN3 do Driver será ligado ao pino 7 do arduino
```

```
#define IN4 6     //o pino IN4 do Driver será ligado ao pino 6 do arduino
```

```
#define ENB 5     //o pino ENB do Driver será ligado ao pino 5 do arduino (motor da direita)
```

```
// ===== Variáveis utilizadas ===== //
```

```
char Texto_Recebido; //variável do tipo caracter (char) - recebe os dados do bluetooth/comunicação serial
```

byte velocidade = 70; //variável do tipo byte - usada no controle da
velocidade

```
void setup() {
// === Definição dos pinos de saída === //
  pinMode(ENA, OUTPUT);           //configura o pino do ENA como saída
  pinMode(IN1, OUTPUT);           //configura o pino do IN1 como saída
  pinMode(IN2, OUTPUT);           //configura o pino do IN2 como saída
  pinMode(IN3, OUTPUT);           //configura o pino do IN3 como saída
  pinMode(IN4, OUTPUT);           //configura o pino do IN4 como saída
  pinMode(ENB, OUTPUT);           //configura o pino do ENA como saída
  pinMode(13, OUTPUT);            //configura o pino do 13 (ligado ao led L) como
saída
  Serial.begin(9600);              //Inicia a comunicação serial
}

void loop() {
// === Dados recebidos na comunicação === //
  //if (Serial.available() == 0) {    //"SE" não tiver dados disponíveis na
comunicação, faça...
    // Texto_Recebido = 'S';           //Situação necessária para parar o carro
(incluindo perda de sinal)
  // }

  if (Serial.available()>0) {        //"SE" tiver dados disponíveis na comunicação,
faça...
    Texto_Recebido = Serial.read();    //o valor da leitura serial é atribuído à
variável Texto_Recebido
  }

// === Movimentação do carro === //
  if (Texto_Recebido == 'F') {        //"SE" a letra recebida for "F", faça...
    Serial.println("Para Frente");    //imprime o texto "Para Frente" no monitor serial
  }
}
```

```

    analogWrite (ENA, velocidade);      //envia ao pino ENA o valor da velocidade
do motor 1
    analogWrite (ENB, velocidade);      //envia ao pino ENB o valor da velocidade
do motor 2
    digitalWrite(IN1, 1);                //envia nível lógico alto para o pino IN1  //liga o
motor 1
    digitalWrite(IN2, 0);                //envia nível lógico baixo para o pino IN2  //para
frente
    digitalWrite(IN3, 1);                //envia nível lógico alto para o pino IN3  **liga o
motor 2
    digitalWrite(IN4, 0);                //envia nível lógico baixo para o pino IN4 **para
frente
    digitalWrite(13,1);                  //liga o led L ligado ao pino 13
}

```

```

else if (Texto_Recebido == 'I') {      //"MAS SE" a letra recebida for "I", faça...
    Serial.println("Para Frente e Esquerda"); //imprime o texto "Para Frente e
Esquerda" no monitor serial
    analogWrite (ENA, velocidade);      //envia ao pino ENA o valor da velocidade
do motor 1
    analogWrite (ENB, velocidade-150);  //envia ao pino ENB o valor da velocidade-
150 do motor 2
    digitalWrite(IN1, 1);                //envia nível lógico alto para o pino IN1  //liga o
motor 1
    digitalWrite(IN2, 0);                //envia nível lógico baixo para o pino IN2  //para
frente
    digitalWrite(IN3, 1);                //envia nível lógico alto para o pino IN3  **liga o
motor 2
    digitalWrite(IN4, 0);                //envia nível lógico baixo para o pino IN4 **para
frente
}

```

```

else if (Texto_Recebido == 'G') {      //"SE" a letra recebida for igual a 'G', o carro
se movimenta para Frente Direita.

```

```

    Serial.println("Para Frente e Direita"); //imprime o texto "Para Frente e Direita" no
monitor serial
    analogWrite (ENA, velocidade-150);    //envia ao pino ENA o valor da velocidade-
150 do motor 1
    analogWrite (ENB, velocidade);        //envia ao pino ENB o valor da velocidade
do motor 2
    digitalWrite(IN1, 1);                  //envia nível lógico alto para o pino IN1    //liga o
motor 1
    digitalWrite(IN2, 0);                  //envia nível lógico baixo para o pino IN2    //para
frente
    digitalWrite(IN3, 1);                  //envia nível lógico alto para o pino IN3    **liga o
motor 2
    digitalWrite(IN4, 0);                  //envia nível lógico baixo para o pino IN4    **para
frente
}

else if (Texto_Recebido == 'B') {        // "SE" a letra recebida for igual a 'B', o carro
se movimenta para Trás.
    Serial.println("Para Trás");          //imprime o texto "Para Trás" no monitor serial
    analogWrite (ENA, velocidade);        //envia ao pino ENA o valor da velocidade
do motor 1
    analogWrite (ENB, velocidade);        //envia ao pino ENB o valor da velocidade
do motor 2
    digitalWrite(IN1, 0);                  //envia nível lógico baixo para o pino IN1    //liga o
motor 1
    digitalWrite(IN2, 1);                  //envia nível lógico alto para o pino IN2    //para
trás
    digitalWrite(IN3, 0);                  //envia nível lógico baixo para o pino IN3    **liga o
motor 2
    digitalWrite(IN4, 1);                  //envia nível lógico alto para o pino IN4    **para trás
}

else if (Texto_Recebido == 'H') {        // "SE" a letra recebida for igual a 'H', o carro
se movimenta para Trás e esquerda.

```

```

Serial.println("Para Trás e Esquerda"); //imprime o texto "Para Trás e Esquerda"
no monitor serial
    analogWrite (ENA, velocidade-150);    //envia ao pino ENA o valor da velocidade-
150 do motor 1
    analogWrite (ENB, velocidade);        //envia ao pino ENB o valor da velocidade
do motor 2
    digitalWrite(IN1, 0);                  //envia nível lógico baixo para o pino IN1    //liga o
motor 1
    digitalWrite(IN2, 1);                  //envia nível lógico alto para o pino IN2    //para
trás
    digitalWrite(IN3, 0);                  //envia nível lógico baixo para o pino IN3    **liga o
motor 2
    digitalWrite(IN4, 1);                  //envia nível lógico alto para o pino IN4    **para trás
}

```

```

else if (Texto_Recebido == 'J') {        //"SE" a letra recebida for igual a 'J', o carro se
movimenta para Trás e direita.

```

```

    Serial.println("Para Trás e Direita"); //imprime o texto "Para Trás e Direita" no
monitor serial
    analogWrite (ENA, velocidade);        //envia ao pino ENA o valor da velocidade
do motor 1
    analogWrite (ENB, velocidade-150);    //envia ao pino ENB o valor da velocidade-
150 do motor 2
    digitalWrite(IN1, 0);                  //envia nível lógico baixo para o pino IN1    //liga o
motor 1
    digitalWrite(IN2, 1);                  //envia nível lógico alto para o pino IN2    //para
trás
    digitalWrite(IN3, 0);                  //envia nível lógico baixo para o pino IN3    **liga o
motor 2
    digitalWrite(IN4, 1);                  //envia nível lógico alto para o pino IN4    **para trás
}

```

```

else if (Texto_Recebido == 'L') {        //"SE" a letra recebida for igual a 'L', o carro
se movimenta para a esquerda.

```

```

    Serial.println("Para Esquerda");    //imprime o texto "Para Esquerda" no monitor
serial
    analogWrite (ENA, velocidade);      //envia ao pino ENA o valor da velocidade
do motor 1
    analogWrite (ENB, velocidade);      //envia ao pino ENB o valor da velocidade
do motor 2
    digitalWrite(IN1, 0);               //envia nível lógico baixo para o pino IN1    //liga o
motor 1
    digitalWrite(IN2, 1);               //envia nível lógico alto para o pino IN2    //para
trás
    digitalWrite(IN3, 1);               //envia nível lógico alto para o pino IN3    **liga o
motor 2
    digitalWrite(IN4, 0);               //envia nível lógico baixo para o pino IN4    **para
frente
}

else if (Texto_Recebido == 'R') {      //"SE" a letra recebida for igual a 'R', o carro
se movimenta para a direita.
    Serial.println("Para Direita");    //imprime o texto "Para Direita" no monitor serial
    analogWrite (ENA, velocidade);      //envia ao pino ENA o valor da velocidade
do motor 1
    analogWrite (ENB, velocidade);      //envia ao pino ENB o valor da velocidade
do motor 2
    digitalWrite(IN1, 1);               //envia nível lógico alto para o pino IN1    //liga o
motor 1
    digitalWrite(IN2, 0);               //envia nível lógico baixo para o pino IN2    //para
frente
    digitalWrite(IN3, 0);               //envia nível lógico baixo para o pino IN3    **liga o
motor 2
    digitalWrite(IN4, 1);               //envia nível lógico alto para o pino IN4    **para trás
}

else if (Texto_Recebido == 'S') {      //"SE" a letra recebida for igual a 'S', o carro
deve parar.

```

```

    Serial.println("Parado");           //imprime o texto "Parado" no monitor serial
    digitalWrite(IN1, 0);               //envia nível lógico baixo para o pino IN1
//desliga
    digitalWrite(IN2, 0);               //envia nível lógico baixo para o pino IN2    //o
motor 1
    digitalWrite(IN3, 0);               //envia nível lógico baixo para o pino IN3    **desliga
    digitalWrite(IN4, 0);               //envia nível lógico baixo para o pino IN4    **o
motor 2
    digitalWrite(13,0);                 //desliga o led L ligado ao pino 13
}
}

```