CENTRO PAULA SOUZA

ETEC PAULINO BOTELHO – SÃO CARLOS/SP

Ensino Médio Integrado ao Ensino Técnico em Mecatrônica

Ciro Camargo da Silva Mendes

Emmanoel Irmer Caires

Felipe Pires de Matos Santos

Gustavo Barros Basso

ROBÔ AUTÔNOMO PARA COMPETIÇÕES ACADÊMICAS:

Olimpíada Brasileira de Robótica (OBR)

SÃO CARLOS
2023

Ciro Camargo da Silva Mendes;

Emmanoel Irmer Caires;

Felipe Pires de Matos Santos;

Gustavo Barros Basso.

ROBÔ AUTÔNOMO PARA COMPETIÇÕES ACADÊMICAS: OBR (Olimpíada Brasileira de Robótica)

Trabalho de Conclusão de Curso apresentado ao Curso Técnico em Mecatrônica da Etec Paulino Botelho – São Carlos/SP, orientado pelos professores Cláudio e Eliezer, como requisito parcial para obtenção do título de técnico em Mecatrônica.

SÃO CARLOS
2023

CENTRO PAULA SOUZA

ETEC PAULINO BOTELHO – SÃO CARLOS/SP

Ensino médio integrado ao técnico em mecatrônica

FOLHA DE APROVAÇÃO

Ciro Camargo da Silva Mendes;

Emmanoel Irmer Caires;

Felipe Pires de Matos Santos;

Gustavo Barros Basso.

ROBÔ AUTÔNOMO PARA COMPETIÇÕES ACADÊMICAS: OBR (Olimpíada Brasileira de Robótica)

Aprovado em:_____de_____de 2023.

Banca Examinadora

_____
(Claudio Torres Gonçalves, professor orientador do TCC).

_____
(Eliezer Gibertoni, professor orientador do TCC).

_____
(Magali Teresinha Chiari Alves Araujo, Banca um).

_____
(Celso Hiroshi Tamashiro, Banca dois).

DEDICATÓRIA

Dedicamos este trabalho a Deus, a nossas famílias, a todos que apoiaram nossa jornada e esforços (especialmente todos os professores do técnico e do ensino médio).

AGRADECIMENTOS

Agradecemos aos professores: Ana Cláudia, Angélica Redondo, Antônio Maurilo, Aparecido Moriwaki, Célio Escobar, Celso Tamashiro, Cláudio Gonçalves, Fabio Nakasone, Frederico Jurgensen, Juliana Barbieri, Luís Schiavonne, Magali Teresinha, Márcia Olaio, Nathalia Lara e Valter Carneiro.

Somos gratos pelos seus apoios, ajudas, motivações e incentivos ao nosso projeto, com toda a sua maestria na educação e companheirismo, que foi de suma importância nestes três anos.

“Não creio que haja uma emoção mais intensa para um inventor do que ver suas criações funcionando. Essas emoções fazem você esquecer de comer, de dormir, de tudo.”

NIKOLA TESLA

RESUMO

A equipe de robótica Os Contramestres da Etec Paulino Botelho criou seu primeiro robô para a olimpíada brasileira de robótica do ano de 2022, onde o quesito era ser um robô seguidor de linha. Entretanto não foi obtida a vitória da competição por falhas na parte de programação, desde então a equipe vem trabalhando arduamente para aprimorar o projeto e obter melhores resultados no ano de 2023. Este trabalho de conclusão de curso visa ser uma base para futuras equipes de robótica e apresentação do nosso esforço.

Palavras-chave: Robótica. Competição. Robô. Equipe. Aprimorar. Esforço. Competição.

ABSTRACT

The robotics team, Etec Paulino Botelho's Contramestres, created their first robot for the 2022 Brazilian Robotics Olympiad, where the requirement was to be a line-following robot. However, the victory of the competition was not obtained due to flaws in the programming part, since then the team has been working hard to improve the project and obtain better results in the year 2023. This course conclusion work aims to be a basis for future robotics teams and presentation of our effort.

Key-words: Robotics. Competition. Robot. Team. Effort.

SUMÁRIO

1 INTRODUÇÃO

O projeto, Robô feito para Olimpíada Brasileira de Robótica (OBR), foi desenvolvido com o objetivo de facilitar e auxiliar a participação de futuras equipes na competição, de uma maneira de fácil acesso, desde a programação do Arduino - microcontrolador utilizado- passando pela montagem do projeto e indo até a correção de erros e medidas de precaução.

A ausência de exemplos de certos comandos para determinadas situações foi um grande problema. Por ser uma disputa, as equipes não costumam disponibilizar os seus métodos utilizados para a criação do robô, deixando a equipe à mercê dos próprios conhecimentos e esforços.

O objetivo é desenvolver um carrinho autônomo capaz de se localizar no ambiente disponibilizado pelos organizadores da competição, ou seja, uma simulação de um ambiente de resgate de vítimas após um desastre. Ele deverá seguir as linhas, ultrapassar os obstáculos, fazer desvios, ser capaz de subir terrenos íngremes e resgatar vítimas (pegar bolinhas) na fase final da Olimpíada. Deverá realizar tudo em menos de 5 minutos.

Para a realização desse trabalho, foi utilizado um robô com esteiras ao invés de rodas para se ter mais aderência. Para ser algo viável, foi utilizado o Arduino e seus módulos (sensores e shields de prototipagem).

Ciro Camargo da Silva Mendes, Emmanoel Irmer Caires, Felipe Pires de Matos Santos e Gustavo Barros Basso.

2 FUNDAMENTAÇÃO TEÓRICA

A fundamentação teórica é base do projeto a ser desenvolvido. Todos os esforços, raciocínios e lógicas são criados a partir do conhecimento adquirido sobre os assuntos que serão mencionados a seguir. Irão encontrar alguns termos técnicos e explicações que, muitas vezes, não são lembrados durante o desenvolvimento de projetos, porém é de extrema importância que o técnico desenvolvedor possua conhecimento sobre tais assuntos.

2.1 MECATRÔNICA

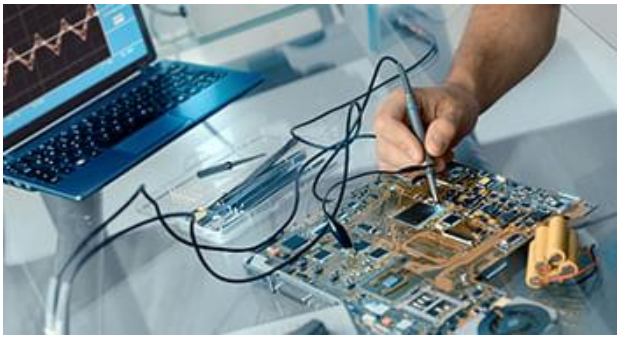
Mecatrônica é a junção de operações mecânicas e eletrônicas, compondo boa parte do ramo de automação, essencial para o projeto a ser desenvolvido.

Figura 12: Exemplo de operação mecatrônica

Fonte: https://www.infomoney.com.br

2.1.1 Eletrônica

Figura 2: Exemplo de operação eletrônica



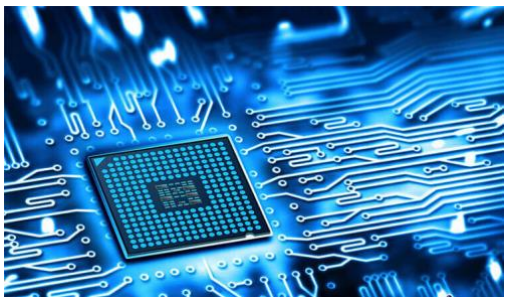
Fonte: https://portaleducacao.vteximg.com.br

É um ramo de estudo aplicado a lógica através das ondas eletromagnéticas, condutores e portadores de carga. Um dos resultados, por exemplo, é a programação.

2.1.1.1 Eletrônica Digital

A eletrônica digital é uma vertente da eletrônica que utiliza circuitos e operações lógicas baseadas em tipos de linguagens de programação como o binário.

Figura 3: Associação a um circuito digital



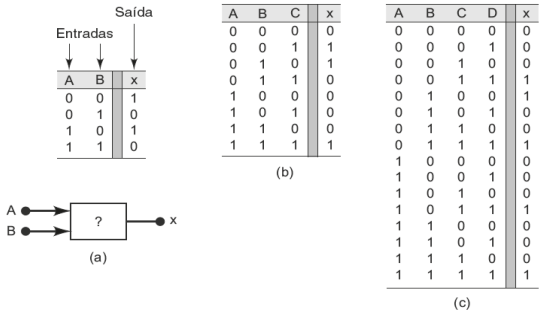
Fonte: https://blog.multcomercial.com.br

2.1.1.1.1 Sistema de Numeração Binário

O sistema de numeração binário é constituído por 0 e 1, sendo sua base o número 2. Muito usado na eletrônica digital, suas combinações geram variações de resultados que podem ser usar em diversas áreas, como na linguagem C do Arduino (HIGH = 1, LOW = 0; ON = 1, OFF = 0; 5V = 1, 0V = 0; Preto = 1, Branco = 0).

Também é possível os transformar em outras linguagens, como a Decimal e Hexadecimal, e é possível escrever caracteres.

Figura 4 – Representação de numeração Binária: a) Operador lógico; b) Tabela verdade com três entradas; c) Tabela verdade com quatro entradas



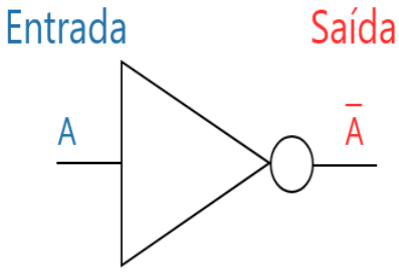
Fonte: https://tecdicas.com

2.1.1.1.2 Portas Lógicas

Há cerca de 7 portas lógicas. A sua principal finalidade é realizar operações lógicas, como multiplicação e soma, assim mostradas nos circuitos integrados (CI).

- ✓ **NOT:**

Figura 5 – Representação do operador NOT

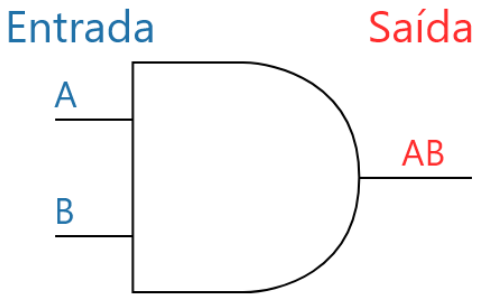


Fonte: https://mundoprojetado.com.br

A porta NOT. Significa negação, ou seja, se for enviado um sinal 1 será recebido 0 e vice-versa. A representação do sinal invertido é o valor, ou incógnita, antecedida por “~”.

- ✓ **AND:**

Figura 6 – Representação do operador AND

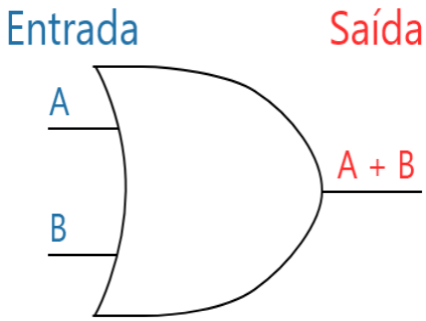


Fonte: https://mundoprojetado.com.br

A porta AND. Tem por afinidade realizar a operação lógica de multiplicação, ou seja, quando introduzidos dois sinais como 1 e 1 esse operador irá multiplicá-los, isso vale para as diversas vertentes de valores como: 1-1, 1-0, 0-1 e 0-0.

✓ **OR:**

Figura 7 – Representação do operador OR

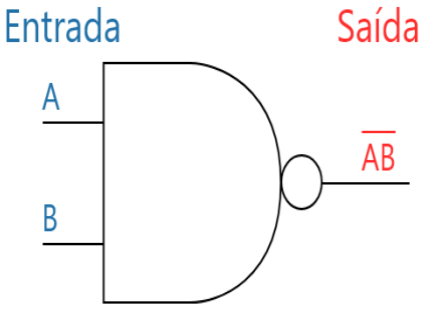


Fonte: https://mundoprojetado.com.br

A porta OR. Ressalta a operação da soma, ou seja, ela irá pegar valores que receber e somá-los. Por exemplo, dois sinais 0 que, somando-os (0+0), retornará o valor 0, e assim por diante com 1-1, 1-0, 0-1 e 0-0.

✓ **NAND:**

Figura 8 – Representação do operador NAND

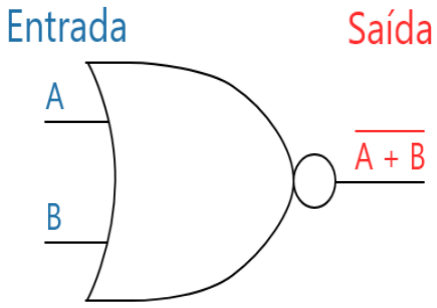


Fonte: https://mundoprojetado.com.br

A porta NAND. Tem o mesmo propósito da porta lógica AND, mas com um único diferencial: o resultado obtido será invertido, pois a porta está em estado de negação (NOT), ou seja, quando o resultado obtido ser 1, ele tornará 0 e vice-versa.

✓ **NOR:**

Figura 9 – Representação do operador NOR

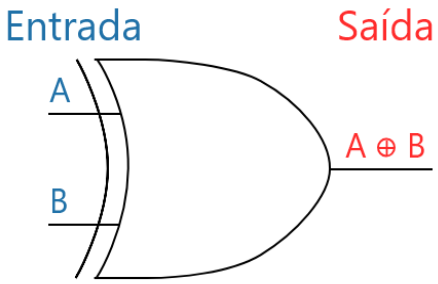


Fonte: https://mundoprojetado.com.br

A porta NOR. Possui a mesma finalidade do operador OR, porém com a mesma ideia do anterior citado: o resultado (soma) será invertido, pois a porta está em negação, com o 1 virando 0 e vice-versa.

✓ ***XOR:***

Figura 10 – Representação do operador XOR

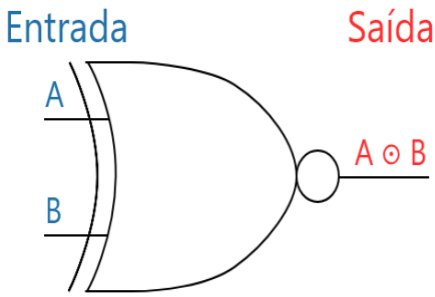


Fonte: https://mundoprojetado.com.br

A porta XOR ou EXCLUSIVO. Quando o operador receber dois sinais iguais, como 1-1 ou 0-0, seus resultados sempre serão 0 e, quando forem diferentes, 1-0 ou 0-1, seu resultado será 1 (as operações lógicas utilizadas por essa porta são: ~A~B + A~B).

✓ **XNOR:**

Figura 11 – Representação do operador XNOR



Fonte: https://mundoprojetado.com.br

A porta XNOR. Tem o mesmo objetivo da porta lógica XOR, com um único diferencial: quando os sinais recebidos pela porta forem iguais ele irá resultar em 1, e quando diferentes irá ser resultado em 0, ou seja, é o resultado de uma XOR inverso.

2.1.1.1.3 Microcontroladores

Os microcontroladores são chips que possuem os componentes necessários para cumprirem sua função automaticamente, necessitando apenas de serem programados e alimentados.

Sobre os microcontroladores são destacáveis as memórias, que são unidades de armazenamentos e leitura de dados em ordem binária. São elas:

✓ **Random Access Memory (RAM):** É capaz de realizar o armazenamento de dados, os quais podem ser escritos e lidos por ela. Ela depende de energia para manter os seus dados íntegros (memória volátil).

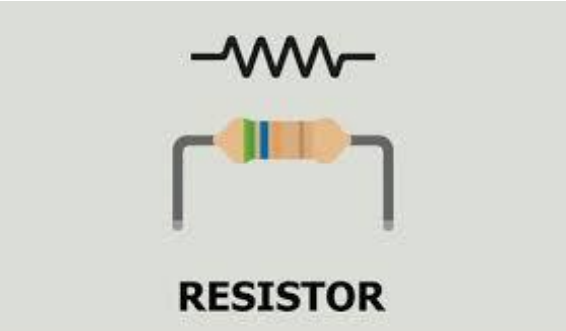
✓ **Read Only Memory (ROM):** É uma memória onde só é feita a leitura dos dados, não é possível escrever nela (como a BIOS relacionada com a inicialização do computador).

✓ **Programmable Read-Only Memory (PROM):** É uma memória ROM programável, porém só é possível programar nela uma vez.

✓ **Electrically-Erasable Programabble Read Only Memory (EEPROM):** São memórias ROM's programáveis e apagáveis, sendo o processor de apagar feito através da presença de luz ultra-violeta ou, em alguns casos, comandos elétricos. A diferença dessas memórias para a memória RAM é que essas não necessitam de energia para manter a integridade dos seus dados e seu espaço de armazenamento é mais demorado e limitado.

2.1.1.2 Componentes Fundamentais da Eletrônica

Figura 12 – Representação de um Resistor

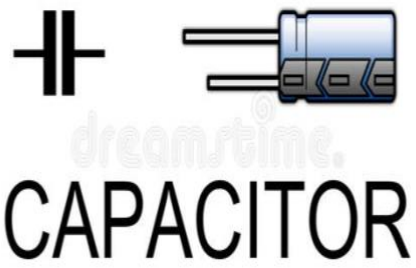


Fonte:https://www.shutterstock.com

No ramo da eletrônica há diversos componentes dentre os quais compõem um circuito elétrico, alguns componentes básicos para a montagem de circuitos:

O resistor é um componente muito comum em diversos circuitos elétricos e eletrônicos. Sua finalidade é impedir a passagem de corrente elétrica em determinadas partes posteriores ao mesmo circuito. A unidade de medida da resistência é OHM, representado pela letra grega OMEGA (Ω).

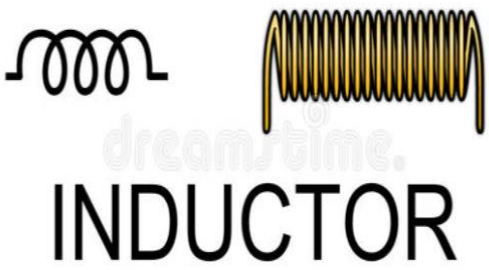
Figura 13 – Representação de um Capacitor Eletrolítico



Fonte: https://thumbs.dreamstime.com

A definição primordial de um capacitor em um circuito elétrico é a capacidade de armazenar e descarregar cargas. Quando trabalhamos com circuitos de Corrente Contínua (CC), o capacitor irá representar esse papel, porém, em circuitos de Corrente Alternada (CA), ele trabalha como uma resistência. Sua unidade de medida é Faraday (F) em homenagem ao físico Michael Faraday.

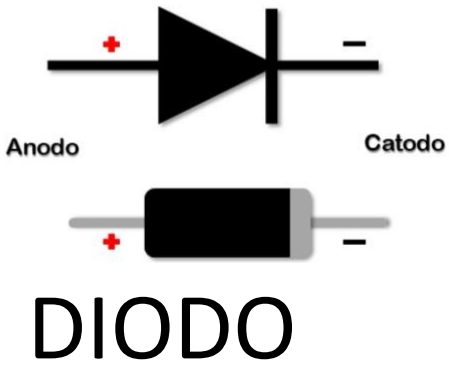
Figura 14 – Representação de um Indutor



Fonte: https://thumbs.dreamstime

O indutor é um componente cuja principal função é induzir a corrente elétrica em função aos campos magnéticos projetados, ou seja, quando for induzido uma corrente elétrica, mais energia passará pelo circuito. O indutor também se comporta como um tipo de resistência em circuitos de CA. Sua unidade de medida é chamada de mH, em homenagem ao seu descobridor.

Figura 15 – Representação de um Diodo



Fonte: https://www.manualdaeletronica.com.br

O diodo é um componente que impede a passagem de corrente contrária à sua polaridade. É comumente utilizado em circuitos retificadores.

2.1.1.3 Linguagem de Programação

É a maneira na qual o programador se comunica com a máquina. O programador controla a lógica da máquina, que ações o software deve executar, quais dados operar e como armazená-los a partir de diversas circunstâncias. Ele cria programas a partir de uma série de ordens, algoritmos, dados e comandos.

Existem diversas linguagens de programação que funcionam através de palavras, símbolos, regras semânticas e sintáticas específicas de cada uma. Entre elas:

- ✓ **Java:** Desenvolvida na década de 1990, é capaz de desenvolver softwares capazes de serem executados em diferentes plataformas (Windows, Linux etc.).
- ✓ **JavaScript:** É uma linguagem de programação criada para o desenvolvimento da web. É a linguagem de programação web mais popular.
- ✓ **Python:** Uma das principais linguagens de programação, o Python é utilizado para fins diversos graças a sua capacidade de suportar diferentes paradigmas e possui uma alta quantidade de recursos. Ganhou destaque pela legibilidade do seu código e da sintaxe moderna.
- ✓ **C:** Foi uma evolução de linguagens anteriores (ALGOL68 e BCPL). Foi criada graças a uma necessidade de facilitar a escrita de programas em linguagem Assembly (linguagem que era usada ainda na época que os computadores possuíam válvulas).
- ✓ **C++:** Baseada em linguagem C, é uma linguagem com capacidade de resolver diversos problemas. Aplicada hoje em dia na criação de jogos, edição de imagens e vídeos.
- ✓ **C#:** Também conhecida como “C Sharp”, foi desenvolvida pela Microsoft. É uma linguagem orientada a objetos. Sua sintaxe foi baseada em C++, Java e Object Pascal.

2.1.2 Mecânica

A Mecânica aplica os princípios da física e ciência dos materiais para produzir, conceber, analisar e realizar a manutenção de sistemas mecânicos. Envolve a produção e operação de máquinas e ferramentas.

Dentro da mecânica, é possível destacar as seguintes áreas:

- ✓ Processos de produção;
- ✓ Sistemas de transmissão;
- ✓ Metrologia;
- ✓ Métodos e processos de usinagem e de conformação.

2.1.2.1 Processos de Produção

O processo de produção se resume ao trabalho, as matérias-primas e os instrumentos de produção.

2.1.2.1.1 Trabalho

É toda atividade realizada pelo homem que resulta em bens ou serviços. Pode ser qualificado, exigindo um certo grau de aprendizagem, e pode ser não qualificado, que qualquer um é capaz de realizar.

2.1.2.1.2 Matérias Primas

São os objetos que, durante o processo de produção, se torna o produto requisitado. Um ótimo exemplo são os recursos naturais, de onde é tirado todos os produtos físicos.

2.1.2.1.3 Instrumentos de Produção

São todas as ferramentas utilizadas para transformar as matérias-primas no produto durante o processo de produção. Exemplo: Em uma tornearia o instrumento de produção é o torno.

2.1.2.2 Sistemas de Transmissão

Os sistemas de transmissão são responsáveis por transferir potência e movimento, respectivamente torque e rodadas por minuto (rpm), para outro sistema (de um eixo para outro eixo), podendo ou não ter alteração na velocidade. Dentro do sistema de transmissão, é importante destacar os elementos de transmissão e a relação de transmissão.

2.1.2.2.1 Elementos de Transmissão

Os elementos de transmissão são de onde vem o trabalho para realizar a variação na rotação entre os eixos. Podem se utilizar os seguintes elementos:

- ✓ Engrenagens (cilíndricas, cônicas);
- ✓ Correias (planas, em “v” ...);
- ✓ Coroa e parafuso sem-fim;
- ✓ Correntes;
- ✓ Rodas de atrito;

2.1.2.2.2 Relação de Transmissão

A relação de transmissão é um valor representado por “i”, onde indica o quanto um elemento é maior que o outro e possui uma relação direta com o aumento ou redução do torque e da velocidade.

Quando o movimento é passado de um elemento maior para um menor é chamado de sistema ampliador, pois ocorre o aumento da rotação e a diminuição do torque. Nele, o valor de “i” é menor do que 1.

Quando o movimento é passado de um elemento menor para um maior é chamado de sistema redutor, ocorre a diminuição na rotação e o aumento do torque. Nele, o valor de “i’ é maior do que 1.

Segue as formas de se calcular as diferentes relações de transmissão:

Tabela 1: Análise matemática sobre os processos de relação de transmissão

	REDUTOR	AMPLIADOR	
	Equações	Equações	Incógnitas
Engrenagens	$i = \frac{Z2\ (maior)}{Z1\ (menor)}$ $i = \frac{N_1\ (maior)}{N_2\ (menor)}$	$i = \frac{Z2\ (menor)}{Z1\ (maior)}$ $i = \frac{N_1\ (menor)}{N_2\ (maior)}$	i=Relação de transmissão Z1= Número de dentes da engrenagem de entrada Z2= Número de dentes da engrenagem de saída N1= RPMs de entrada
Correntes	$i = \frac{Z_2\ (maior)}{Z_1\ (menor)}$ $\pi * d = Z * P$	$i = \frac{Z_2\ (menor)}{Z_1\ (maior)}$ $\pi * d = Z * P$	N2= RPMs de saída d=Diâmetro primitivo da engrenagem Z= Número de dentes da engrenagem
Rodas	$i = \frac{N_1\ (maior)}{N_2\ (menor)}$ $i = \frac{D_2\ (maior)}{D_1\ (menor)}$	$i = \frac{N_1\ (menor)}{N_2\ (maior)}$ $i = \frac{D_2\ (menor)}{D_1\ (maior)}$	P= Passo da corrente D1= Diâmetro de entrada D2= Diâmtero de saída Z1= Número de dentes do parafuso sem-fim Z2= Número de dentes da coroa
Coroa e para-fuso sem fim	$i = \frac{Z_2}{Z_1}$	$i = \frac{Z_2}{Z_1}$	

Fonte: Autoria própria.

2.1.2.3 Metrologia

A metrologia é o estudo das medições. Nele será contemplada a origem da necessidade de se desenvolver uma medida aceita mundialmente e um instrumento de medição: o paquímetro.

2.1.2.3.1 O Metro

Antigamente eram usadas unidades de medidas baseadas em partes do corpo humano (pés, jarda, polegada, palma, côvado etc.). Porém, já que essas medidas variam de pessoa para pessoa, foi desenvolvido uma unidade de medida que se baseava em algo presente na natureza para que todos tivessem acesso a mesma medida: o metro.

O metro teve sua primeira aparição no final do século XVIII, possuindo um valor igual à décima milionésima parte de um quarto do meridiano terrestre. Foi medido esse valor e armazenado em uma barra de platina, a qual passou a ser a referência do metro. Após decorrentes aumentos na precisão das medidas, o valor do metro ia se alterando, e ficando cada vez mais preciso.

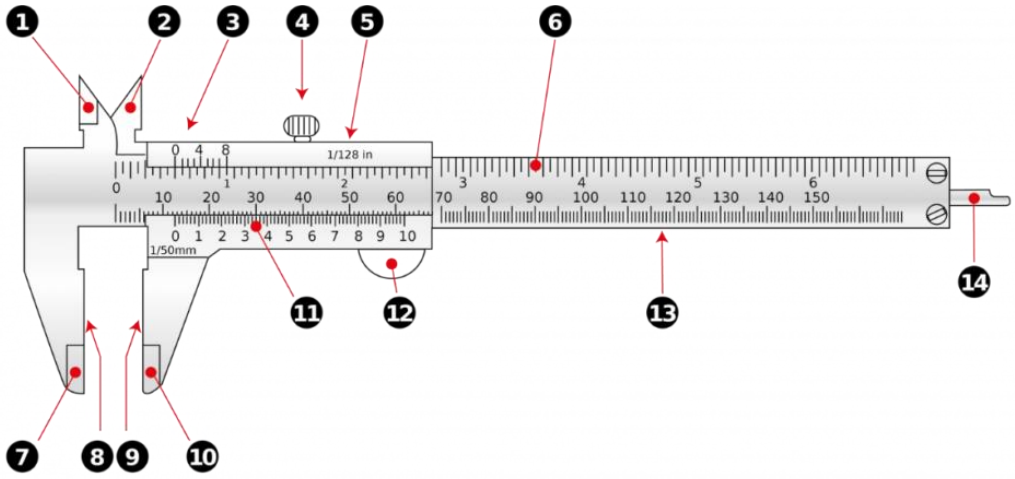
Outras mudanças ocorreram, mas hoje o metro é definido pelo INMETRO como a distância percorrida pela luz no vácuo durante um intervalo de 1/299.792.458 do segundo (3,34 * 10^-9 segundo).

2.1.2.3.2 O Paquímetro

O paquímetro é um instrumento de medição linear que consiste em uma régua graduada com um encosto fixo, sobre o qual é deslizado um cursor com uma escala auxiliar permitindo a leitura fracionada da menor divisão da escala na régua.

Para uma boa utilização desse instrumento, são necessários alguns conhecimentos prévios. Após a imagem:

Figura 16: Representação de um Paquímetro e seus componentes (partes)



❶ orelha fixa
❷ orelha móvel
❸ nônio ou vernier (polegada)
❹ parafuso de trava
❺ cursor
❻ escala fixa de polegadas
❼ bico fixo
❽ encosto fixo
❾ encosto móvel
❿ bico móvel
⓫ nônio ou vernier (milímetro)
⓬ impulsor
⓭ escala fixa de milímetros
⓮ haste de profundidade

Fonte: https://hennings.com.br

- ✓ **Medidas:** Através das orelhas do paquímetro é possível ler medidas de diâmetro internos e canais. Através do bico fixo e bico móvel do paquímetro é possível ler medidas de diâmetros externos e dimensões de fácil acesso (como comprimento de faces). Através da haste de profundidade é possível ler medidas de profundidade de canais e furos.

- ✓ **Leitura de Escala Fixa:** Primeiro deve encaixar a peça entre o bico móvel e o bico fixo do paquímetro. Em seguida deve-se ver a distância que o zero do cursor dista do zero da escala fixa.

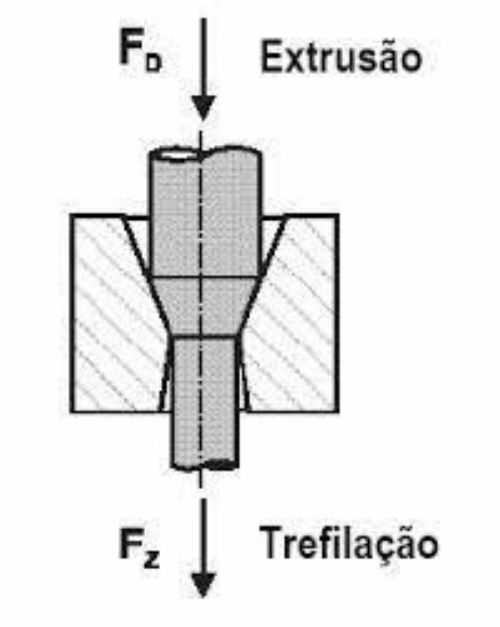
- ✓ **Funcionamento e Leitura do Nônio:** A escala do cursor, nônio, do paquímetro acima possui 50 divisões. Ou seja, se você alinhar o zero da escala fixa com o zero do nônio, a primeira divisão do nônio estará 0,02mm atrasada, a segunda estará 0,04mm atrasada, a terceira estará 0,06mm atrasada e assim sucessivamente. O cursor permite maior precisão na leitura pois, quando ele se movimenta míseros 0,20mm para a esquerda, iremos ver que a décima divisão do nônio estará alinhada com uma das divisões da régua e o zero desalinhado 0,20mm para a esquerda (saberemos a medida de 0,20mm pois a décima divisão do cursor está 0,20mm atrasada em relação à escala fixa e, se ela está alinhada, então o zero foi movimentado 0,20mm para a esquerda). Após realizar a leitura da escala fixa, deve se buscar a divisão do cursor que está alinhada com a régua para possuir uma precisão milimetrada das dimensões da peça. Existem diversas escalas de nônio. Esse exemplo é de um nônio com escala de 1/50mm, mas existem nônios de 1/20mm (cujo atraso nas divisões do cursor equivalem a 0,05mm), 1/10mm (cujo atraso nas divisões do cursor equivalem a 0,1mm) etc. O funcionamento e modo de leitura são os mesmos. Para maior precisão é aconselhado realizar a leitura três vezes e utilizar o valor da média aritmética entre elas.

2.1.2.4 Métodos e Processos de Conformação e Usinagem

Os processos de conformação podem ser divididos em duas vertentes.

A vertente (a) seriam os processos de conformação maciça, no caso: forjamento, extrusão, laminação e trefilação.

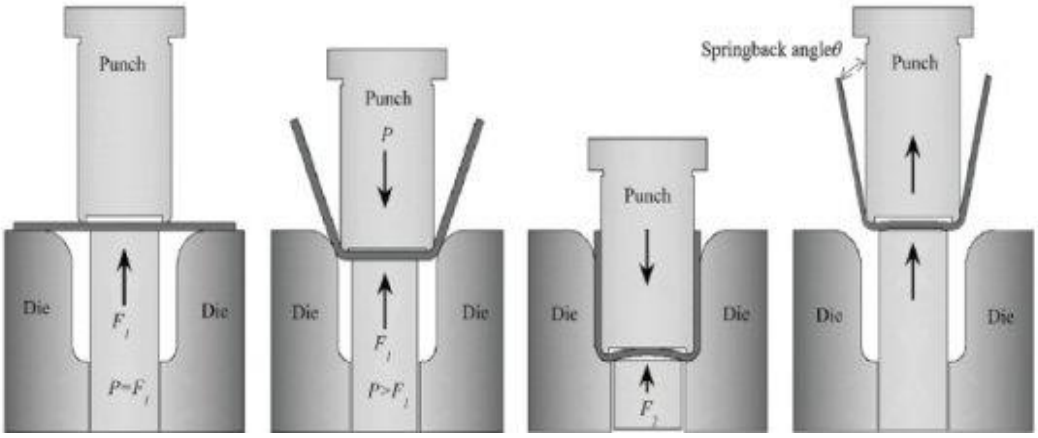
Figura 17: Processo de trefilação



Fonte: https://sistemas.eel.usp.br/A3PIM.pdf

Já a vertente (b), é a conformação de processos de chapas, como por exemplo o dobramento, repuxo e esticamento.

Figura 18: Método de conformação de chapa



Fonte: https://www.inovacaotecnologica.com.br

2.1.3 Desenho Técnico

Desenhos Técnicos são figuras que representam, de forma detalhada, o modo de usinagem e dimensões de uma peça, ou determinado objeto, com medições

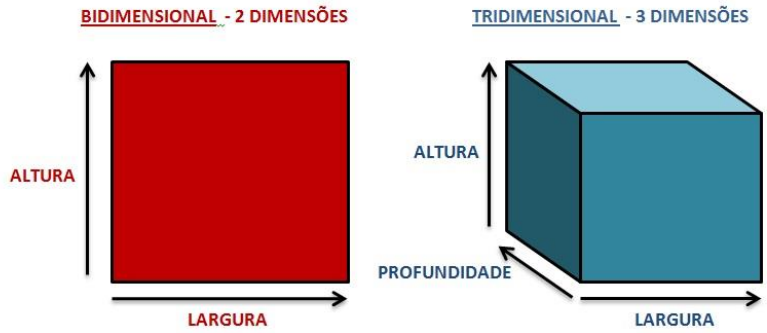
e proporções que devem ser respeitadas. Dentro do assunto sobre desenho técnico serão abordados os seguintes tópicos:

- ✓ Geometria da peça;
- ✓ Vista lateral, frontal e superior;
- ✓ Projeção isométrica;
- ✓ Programas utilizados para desenhos técnicos;
- ✓ Principais funções utilizadas.

2.1.3.1 Geometria da Peça

Quando obtido uma peça usinada, tem de se pensar em como ela saiu do papel para finalmente chegar em sua forma final. Para isso, precisa-se saber como desenhá-la, baseando-se em geometria, e fazer uma relação com sua proporção. Como exemplo, a seguinte peça:

Figura 19: Representação das dimensões 2d e 3d



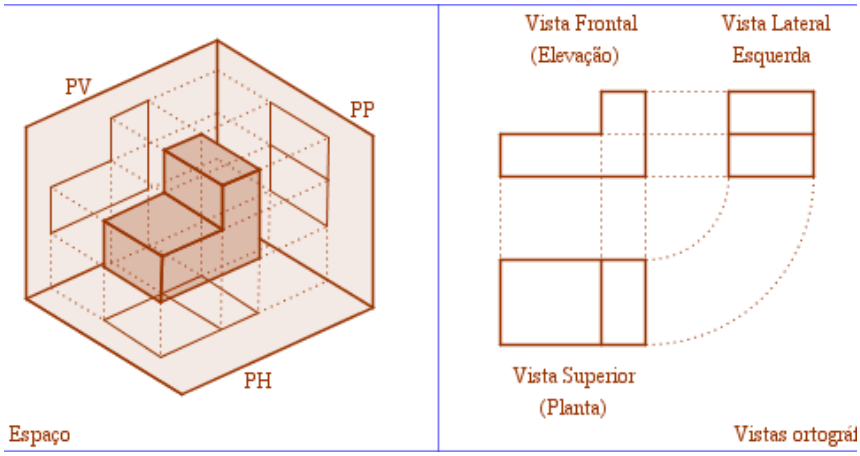
Fonte: https://wwhttps://www.google.com

Nessas peças geométricas, pode-se perceber uma singela diferença que, no caso, é o fato de uma estar em um plano bidimensional e a outra em plano tridimensional, sendo seus respectivos eixos: X, Y e X, Y, Z.

2.1.3.2 Vista Lateral, Frontal e Superior

Quando visualizada uma peça tridimensional é necessário representá-la em três perspectivas, chamadas de projeções ortogonais. Pode-se observar um claro exemplo de uma peça abaixo, e ao lado dela suas três perspectivas ortogonais, que são as três vistas:

Figura 20: Visualização das três vistas ortogonais

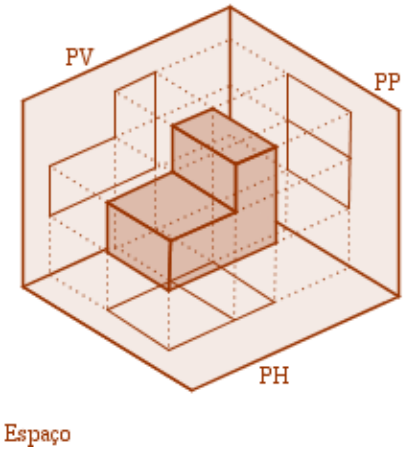


Fonte: https://descritiva.blogspot.com

2.1.3.3 Projeção Isométrica

A projeção isométrica é a junção das três perspectivas citadas anteriormente em um plano só, de maneira que o observador consiga enxergar de forma tridimensional a peça desenhada para, assim, obter uma melhor visualização dela.

Figura 21: Perspectiva Isométrica de uma peça



Fonte: https://descritiva.blogspot.com

2.1.3.4 Programas Utilizados para Desenhos Técnicos

Hoje em dia, desenhos a mão são pouco utilizados. Partimos para os desenhos técnicos digitais realizados por meio de softwares, utilizados com o auxílio do

próprio computador. Dentre os vários programas, iremos citar os mais utilizados. São eles:

- ✓ AutoCad;
- ✓ Inventor;
- ✓ SolidWorks;
- ✓ Fusion 360;
- ✓ Revit;

2.1.3.4.1 AutoCad

O AutoCad é um programa digital usado, em sua maioria, para desenhos técnicos, mas também é utilizado para projetos de planta baixa. O AutoCad acaba sendo uma opção bem versátil para as empresas em geral, pois o custo é baixo comparado a outros programas mais complexos. Em geral, o AutoCad é um programa básico para a prática dos desenhos técnicos, tanto para perspectivas 3D como para 2D.

Figura 22: Imagem do ícone do programa AutoCad

Fonte: https://www.autodesk.com

2.1.3.4.2 Inventor

O Inventor é um programa utilizado nas indústrias para a realização de desenhos técnicos. O seu diferencial para com os demais programas é que possui uma relação conjunta entre o 2D e o 3D bem mais versátil do que o anteriormente citado. Outro aspecto é o fato de ser possível simular os conjuntos de peças

funcionando, como por exemplo para ver se um conjunto de engrenagens funciona corretamente, se os dentes se encaixam e dentre outras demais aplicações.

Figura 23: Imagem do ícone do programa Inventor

Fonte: https://www.autodesk.com

2.1.3.4.3 SolidWorks

O programa SolidWorks é bem similar ao Inventor, contendo apenas uma interface diferente. Contém alguns aspectos semelhantes, como visualização 2D e 3D mais eficiente para os projetistas, possibilita simulações de relações entre as peças criadas, dentre outras semelhanças.

Figura 24: Imagem do ícone do programa SolidWorks

Fonte: https://images.sftcdn.net

2.1.3.4.4 Fusion 360

O Fusion 360 é um dos programas mais completos para design de projetos. Ele é muito utilizado em diversas indústrias automobilísticas devido a sua alta opções de recursos gráficos e projetos que possam ser realizados. O projetista

consegue elevar seus conhecimentos de desenho técnico para uma simulação. Além disso, o Fusion 360 consegue conter todos os recursos já mencionados anteriormente e aprimorá-los exponencialmente.

Figura 25: Imagem do ícone do programa Fusion 360

Fonte: https://www.autodesk.com

2.1.3.4.5 Revit

O Revit possui uma única diferença dele para os demais: o utilizador pode realizar uma simulação em três dimensões sobre o projeto realizado, com até dimensões internas. Graças a isso, é mais versátil para projetos de arquitetura, mas também é usado em projeção de peças, desenhos técnicos e plantas baixas.

Figura 26: Imagem do ícone do programa Revit

Fonte: https://www.autodesk.com

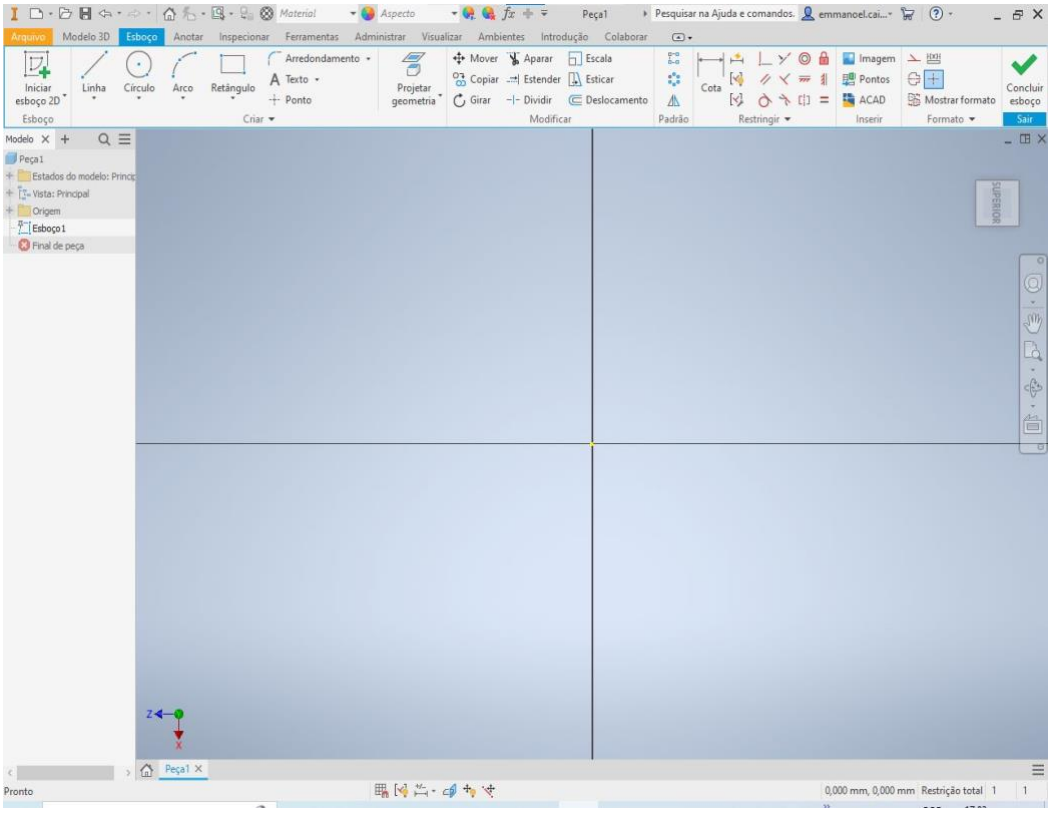
2.1.3.5. Principais Funções Utilizadas

Dentre as principais funções abordadas nos programas acima, destaca-se algumas mais usadas em nosso projeto, pois utilizamos a impressora 3d para a produção das peças do robô.

2.1.3.5.1 Comandos 2D

Os comandos que serão citados abaixo os utilizados em duas dimensões, em outras palavras, comandos 2D. A partir desses comandos, que são primordiais para os desenhos, iremos partir do trabalho 2D (eixos X e Y) e vincular os comandos 3D (eixos X, Y, Z).

Figura 27: Tela de Início do programa Inventor



Fonte: Autoria própria.

Na prática, o software fornece diversos comandos que facilitam as criações. São eles:

- ✓ **Comando Linha:** É dos principais comandos para a realização do desenho técnico, a fim de traçar uma linha para a elaboração do esboço

do desenho, tendo como característica a medida à escolha do projetista. Pode ser selecionado ao clicar, com o mouse, na opção line ou linha.

✓ **Comando Círculo:** Contém a mesma premissa do anteriormente falado, com a pequena diferença que, ao invés de traçar uma linha, fará um círculo. O raio é da escolha do projetista. Pode ser selecionado clicando com o mouse na opção circle ou círculo.

✓ **Comando Arco:** Une duas extremidades de uma reta e deixa-a um pouco mais arredondada, como uma semicircunferência. Pode ser acessada na área de comados 2D, no ícone Arc ou Arco.

✓ **Comando Retângulo:** Desenha um retângulo. Pode ser acessado com o mouse na opção Retângulo.

✓ **Comando Oblongo:** É um comando muito utilizado para realizar a criação de uma figura que parece um retângulo, só que com as extremidades semicirculares. Isso o torna um comando versátil, pois se adapta facilmente à maioria dos desenhos técnicos realizados pela equipe.

✓ **Comando Polígono:** O comando polígono é usado para traçar formas poligonais. Pode ser selecionada no painel criar, na opção retângulo, com o botão direito do mouse.

✓ **Comando Texto:** É um comando que cria caixas de texto para o projetista se organizar melhor, como marcar alguma área específica do desenho técnico para melhor se orientar ou até para evitar que venha se esquecer de algo. Pode ser acessado no menu, no painel “criar”.

✓ **Comando Projetar Geometria:** É um comando muito importante, pois ele ajuda na modelagem da peça ao passá-la do 2D para o 3D. Em geral, o comando cria modelos de arestas e vértices na peça realizada, com o intuito de esboçar as curvas visíveis no plano de trabalho ativo como referência. Pode ser acessado na aba esboço do painel “projetar geometria”.

✓ **Comando Aparar:** Em geral, é um dos mais simples e serve para deletar alguma coisa do desenho, como uma linha. Ele sempre deleta

apenas uma parte do desenho, à escolha do projetista, e nunca o desenho inteiro. É acessado na aba esboço no painel modificar.

✓ **Comando Mover:** Ele consiste em deslocar um objeto do seu esboço, sendo uma linha ou até o próprio desenho inteiro, dependendo do que o projetista deseja. Pode ser acessado no painel modificar.

✓ **Comando Copiar:** Copia partes do esboço, sendo ele 2D ou 3D, e acaba reduzindo o tempo de trabalho. Pode ser acessado na aba esboço, no painel modificar.

✓ **Comando Girar:** Ele gira o projeto, tanto para fins de maior organização ou para fins visuais. Pode ser acessado no painel modificar.

✓ **Comando Estender:** Estende a dimensão da peça, aumentando-a de tamanho. Pode variar a medida de acordo com a decisão do projetista, sendo capaz de alterar muito o esboço. Pode ser acessado no painel modificar.

✓ **Comando Dividir:** Trata-se em o projetista repartir, ou seja, dividir uma parte do desenho técnico a fim de realizar uma dobra ou outro processo de usinagem. É um comando que parte a peça na metade, ou dividindo a escolha do projetista. Pode ser acessada no painel modificar.

✓ **Comando Escala:** Serve para aumentar as dimensões de uma peça sem alterar suas relações de proporção. Pode ser acessado no painel modificar.

✓ **Comando Deslocamento:** Usado a fim de minimizar os erros quando o comando escala é utilizado. Pode ser acessado no painel modificar.

✓ **Comando Padrão Retangular:** Serve para formar uma linha reta através de três pontos alinhados, sem partes circulares como arcos. Assim, economiza-se tempo no preparo do desenho e minimiza-se os erros. Pode ser acessado no painel padrão.

✓ **Comando Padrão Circular:** Trata-se do mesmo conceito do comando anteriormente citado. A diferença é: os três pontos alinhados formando uma reta serão alinhados em forma de um arco ou semicírculo. Ocorre a

mesma função de cópia das formas a fim de minimizar os erros e economizar tempo. Pode ser acessado no painel padrão.

✓ **Comando cota:** É indispensável na criação de um desenho técnico, pois ele irá fornecer a mediada colocada na determinada parte do desenho, seja ela medida de comprimento, raio, diâmetro, largura, altura, ângulo, profundidade, entre outros. Irá mostrar qual unidade de medida está sendo utilizada. Pode ser acessada no painel restringir.

✓ **Comando Restrição Paralela:** Trata-se de uma vertente de um padrão retangular ou até circular, com a diferença de se aplicar apenas em retas paralelas. Consiste em criar uma linha paralela. Pode ser acessada no painel restringir.

✓ **Comando Tangente:** Serve para gerar uma reta que tangencia um raio, a fim de gerar uma reta suporte. Pode ser acessado no painel restringir.

✓ **Comando Lista de Construção:** É um comando que cria uma linha tracejada a fim de estabelecer um tipo de referência auxiliar ao projetista, sendo assim, ela não é lida pelo programa. Pode ser acessado no painel formato.

✓ **Comando Linha de Centro:** Refere-se a uma ação de criar uma linha tracejada que ajuda o projetista ao passar a peça para o modelo 3D, pois auxilia na modelagem. Pode ser acessado no painel formato.

2.1.3.5.2 Comandos 3D

Os comandos 3D são utilizados em diversos programas. Esses comandos possibilitam a visualização tridimensional da peça realizada no esboço. Segue abaixo alguns comandos muito utilizados pela equipe.

✓ **Comando Extrusão:** Permite que o projetista faça uma extrusão, tanto a fim de realizar um preenchimento de uma peça quanto para retirar parte dela. Pode ser acessada no painel criar.

✓ **Comando Revolução:** Aplica-se para o caso de o projetista realizar uma peça cilíndrica ou com padrões circulares, tanto a fim de retirar

material ou para realizar a adição de dimensão na peça. Pode ser acessado no painel criar.

✓ **Comando Furo:** Serve para realizar uma perfuração de uma extremidade à outra na peça, podendo determinar sua largura e profundidade. Pode ser acessado no painel modificar.

✓ **Comando Arredondamento:** Trata-se de uma função utilizada para a junção de duas partes planas em forma de um arredondamento, como um arco em 3D. Pode ser acessado no painel modificar.

✓ **Comando Chanfro:** Trata-se do processo de planificação do vértice da peça projetada, podendo alterar o tamanho dela. Pode ser acessada no painel modificar.

✓ **Comando Padrão Circular:** Trata-se de economizar tempo ao padronizar as medidas. Utilizado para quando se realizará a mesma operação circular anteriormente feita. Pode ser acessado no painel padrão.

✓ **Comando Padrão Retangular:** Assemelha-se com o anterior, com a única diferença de se aplica a aspectos retos e não circulares. Pode ser acessado no painel padrão.

✓ **Comando Casca:** Serve para qualquer peça que seja maciça. Com esse comando, a peça é revestida com apenas uma camada superficial, ou seja, é tornada oca. Pode ser acessada no painel modificar.

✓ **Comando Espelhamento:** Serve para replicar um lado da peça para outro, como uma cópia espelhada. Pode ser acessado no painel padrão.

✓ **Comando Espiral:** Serve para realizar espirais, de forma que a peça fique com uma rosca ou uma abertura espiral, como um recartilhado por exemplo. Pode ser acessada no painel modificar.

✓ **Comando Plano de Trabalho:** Estabelece uma organização melhor no projeto criado, fazendo com que o projetista prepare um outro plano de trabalho onde ele já está, a fim de organizar as referências. Pode ser acessado no painel operações de trabalho.

✓ **Comando Iproperties:** Estabelece uma simulação de resistência da peça, mediante a certos esforços aplicados perante a ela. O projetista pode simular a situação que desejar, a fim de calcular os resultados de pontos de quebra da peça, podendo até mudar o material a ser simulado. Pode ser acessado pesquisando na aba pesquisa do Inventor.

2.1.4 Impressão 3D

A impressão 3D é utilizada na fabricação de peças e elaboração de projetos. Alguns aspectos importantes valem ser ressaltados.

Figura 28: Exemplo de uma Impressora 3d

Fonte: https://img.ibxk.com.br

2.1.4.1 Função da Impressora 3D

O princípio de funcionamento de uma impressora 3D se baseia na criação de um modelo 3D se movimentando nos eixos X, Y e Z a fim de recriar o modelo feito em um programa digital, como o Inventor, passando o mesmo para uma peça real.

Essa máquina sofisticada realiza a impressão com os famosos filamentos, que são tiras bem finas de um polímero, levados automaticamente até uma ponta de extrusão, que esquenta até uma temperatura de aproximadamente 350 graus Celsius. É possível moldar a peça através dos softwares geradores de rotina que a máquina tem que seguir durante minutos, horas ou até dias inteiros.

2.1.4.2 Software da Impressora 3D

Para enviar informação para a impressora realizar o trabalho, a pessoa precisa de um software chamado UltimakerKura. Dentro do mesmo, podemos realizar a edição de como a peça irá ficar na mesa de impressão, preenchimento da peça e modelo 3D.

2.2 ARDUINO

O Arduino é um componente eletrônico muito utilizado em projetos práticos voltados para automação e robótica. No geral, ele é constituído por entradas e saídas, denominados inputs e outputs, e contém um chip que constitui as funcionalidades dele como microcontrolador. Importante citar que a sua linguagem de programação é C++.

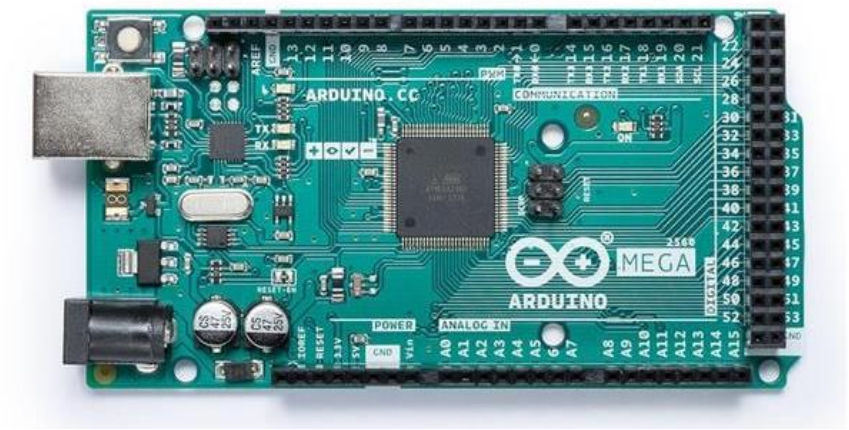
Existem diversos tipos de placas de Arduino. Dentre elas, a Uno é um dos modelos mais básicos que contém um número razoável de portas digitais e analógicas, a Mega é um modelo maior e com mais portas, tanto analógicas quanto digitais, e outro aspecto positivo dessa placa em específico é que a capacidade de armazenamento é bem superior à de outras placas como a Uno e Leonardo.

Por último, a placa chamada Arduino Nano é uma das menores placas. Algumas vantagens são sua utilização em miniprojetos ou sua versatilidade. Um ponto negativo é essa placa não possuir um armazenamento muito grande.

Figura 29: Exemplo de uma placa Arduino Uno

Fonte: https://m.media-amazon.com

Figura 30: Exemplo de uma placa Arduino Mega



Fonte: https://m.media-amazon.com

2.2.1 Fontes de Energia

Em um Arduino, existem duas formas de alimentar a placa. Dentre elas, o cabo Universal Serial Bus (USB) que, além de fornecer energia, tem como principal característica a utilidade de enviar informações para o microcontrolador.

O outro modo consiste em uma entrada de voltagem, Voltage In (VIN), utilizada com fontes externas, como pilhas e baterias avulsas. Tanto para a entrada VIN quanto para a entrada USB a carga máxima é de 12V, caso contrário ele pode vir a ser danificado.

2.2.2 Pinos Digitais

É impossível não falar de pinos digitais quando o assunto é Arduino, pois todo tipo de projeto requer um sinal, sendo ele input (entradas) ou outputs (saídas).

Os pinos digitais desempenham um papel fundamental quando se trata de processamento de dados. A única característica dele é que possui dois tipos de níveis lógicos: o 1, que seria ligado ou que está havendo a passagem energia, e 0, que estaria desligado ou que está com ausência de energia.

2.2.3 Pinos Analógicos

Também não podemos deixar de falar nos pinos analógicos. Os pinos analógicos têm uma única diferença dos digitais: a faixa de informação com que trabalham variam de 0 até 1023, sendo característico para detectar fatores de luminosidade, temperatura, força e outros mais.

2.2.4 Sensores e atuadores

Os Sensores e Atuadores são primordiais em um circuito que envolva o Arduino. Os sensores são componentes que transmitem informações para o Arduino, já os atuadores são componentes que atuam perante as informações enviadas à eles, ou seja, fazem a sua função a partir de um sinal recebido.

2.2.4.1 Sensor Infravermelho

O sensor infravermelho é muito utilizado em projetos pois sua utilização e versatilidade são simples. Existem diversos modelos, mas dentre eles podemos destacar dois. São eles: TCRT-5000 e IR.

Seu modo de funcionamento consiste em dois leds, um sendo transparente que emite um feixe de luz, e um preto que capta a luz do emissor. A luz emitida pelo led está em uma faixa muito pequena, imperceptível a olho nu, mas é refletida por um objeto e detectada pelo led receptor.

Os sinais que os modelos trabalham são valores digitais, sendo: 0 a cor preta, ou ausência de presença, e 1 a cor branca, ou presença de algo.

Uma particularidade de um modelo desse sensor é que o TCRT-5000 também consegue efetuar a leitura em valores analógicos, ou seja, além de ler de 0 até 1, pode também efetuar a leitura de 0 até 1023.

Ele é composto por 4 pinos, sendo eles:

- ✓ Voltagem em Corrente Contínua (VCC): pino de entrada de alimentação para o meu sensor, podendo injetar diferentes valores de voltagem.
- ✓ Terra (GND): pino de saída de energia, ou terra, do sensor.
- ✓ D0: saída de informação digital.
- ✓ A0: saída de informação analógica.

Figura 31: Imagem de um módulo Sensor IR

Fonte: https://images.tcdn.com.br

Figura 32: Imagem de um módulo Sensor TCRT-5000

Fonte: https://images.tcdn.com.br

2.2.4.2 Sensor Inclinação

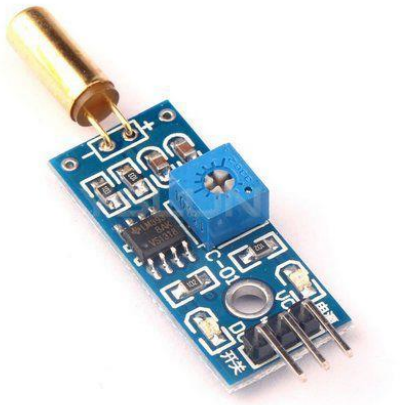
O sensor de inclinação, mais conhecido como sensor Tilt, consegue detectar se uma superfície está inclinada, ou não, e, com isso, mandar sinais para o Arduino.

Seu funcionamento se dá por conta de um cilindro de metal que contém uma pequena esfera de alumínio. Quando há um pequeno deslocamento desse sensor a esfera se move e é possível mandar um sinal para o Arduino. Quando ele detecta que há uma declividade seu valor se altera para 1 e quando a superfície está plana o valor fica 0.

Ele é constituído de 3 portas, sendo elas:

- ✓ OUT: saída de informação digital;
- ✓ VCC: pino de entrada de alimentação;
- ✓ GND: pino de saída de energia, ou terra do sensor.

Figura 33: Imagem de um módulo sensor de inclinação

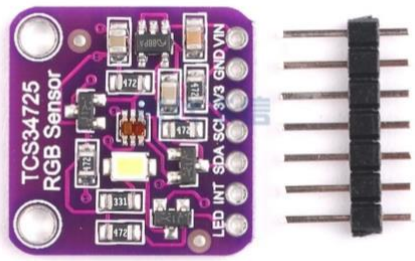


Fonte: https://cdn.awsli.com.br

2.2.4.3 Sensor de Cores

O sensor de cores, também conhecido como TCS-34725, se trata de um detector de cores que trabalha com faixas denominadas Red Green Blue (RGB).

Figura 34: Imagem de um módulo TCS-34725



Fonte: https://imgaz2.staticbg.com

Seu funcionamento se dá por fotodiodos responsáveis por absorver a luminosidade e ao comparar com os parâmetros RGB. Ele identifica, na superfície, a quantidade de R (Red), G (Green) e B (Blue), com o auxílio de um led para a melhor visualização.

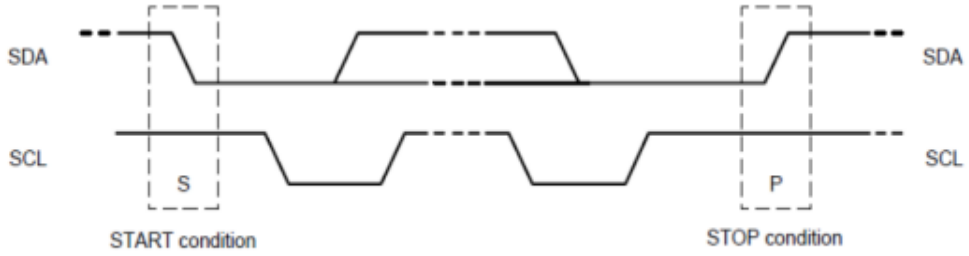
Esse sensor possui um filtro IR, que consiste em filtrar as luzes infravermelhas para a melhor visualização.

É necessário ressaltar que as entradas utilizadas por esse sensor são as I2C, também chamado de endereço I2C. Esse endereço é formado por duas entradas

chamadas Serial Data (SDA) e Serial Clock (SCL). São descritas pelas seguintes funções:

- ✓ SDA: é a linha responsável pelo envio de informações;
- ✓ SCL: é composto por uma lógica que utiliza 3 tipos de sinais, sendo eles o sinal de início, sinal de parada e sinal de resposta. Sua função é determinar o lugar que as informações do SDA serão processadas.

Figura 35: Esquema de sinais SDA e SCL do barramento I2C



Fonte: TCS34725 Color Sensor User Manual

O sinal de início ocorre quando o SCL está ligado e o SDA sendo trocado de sua forma ligada para sua forma desligada, o que resulta na transmissão do chamado sinal de informação.

O sinal de parada consiste no SCL ligado e o SDA desligado, assim interrompendo a transmissão.

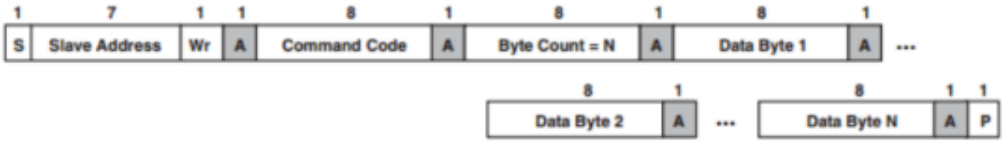
Por fim, o sinal de resposta é quando o IC é enviado de volta ao Arduino em um pulso desligado, assim enviando uma resposta após o processo de receber 8 bits de informação.

O funcionamento da escrita do endereçamento I2C, denominado I2C Write, é detalhado da seguinte maneira:

- ✓ Primeiro: a placa envia um sinal para um byte específico chamado de servo.
- ✓ Segundo: 7 dos 8 bits são nomeados de endereços específicos do servo, já o outro bit que sobrou é denominado de bit de escrita.
- ✓ Terceiro: o servo é responsável pelos endereços de sinais, enviados pela placa, toda vez que receber uma informação.

✓ Quarto: a placa registra os endereços para o servo.

Figura 36: Fluxograma do I2C Write



Fonte: TCS34725 Color Sensor User Manual

Após a descrição da escrita, o próximo processo é a leitura do comando.

✓ Primeiro: o mestre envia um sinal de início, mandando um byte para o servo, sendo os 7 primeiros bits de endereço e o último bit de escrita;

✓ Segundo: o servo responde toda vez que receber qualquer informação;

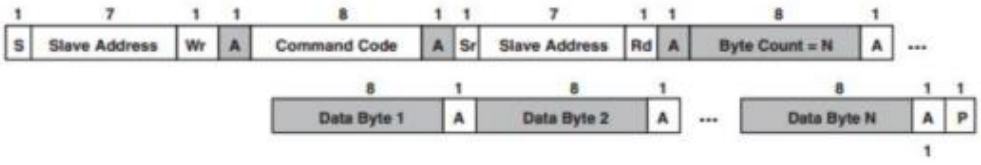
✓ Terceiro: o mestre manda um comando que registra o endereço para o servo;

✓ Quarto: o mestre envia outro sinal de início (1 byte);

✓ Quinto: com isso o servo envia para o mestre um registro dos endereços de informações gravados;

✓ Sexto: o mestre responde com um sinal de parada, encerrando a comunicação entre ambos.

Figura 37: Fluxograma do I2C Read



Fonte: TCS34725 Color Sensor User Manual

2.2.4.4 Sensor ultrassônico

O sensor ultrassônico, também conhecido como HC-SR04, é um emissor de frequências sonoras que, por meio da propagação das ondas emitidas, pode conseguir detectar e calcular a distância de um objeto em suas proximidades.

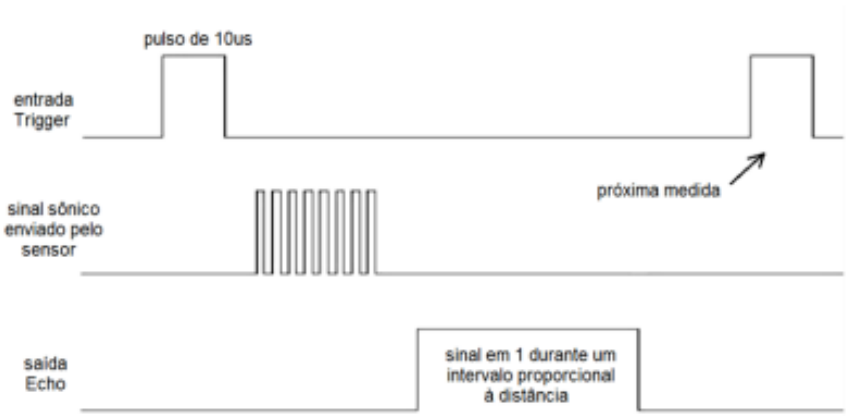
Figura 38: Imagem do sensor HC-SR04

Fonte: EPUSP — PCS 3645 — Laboratório Digital II (USP)

Esse sensor usa uma frequência específica que ondas possui uma velocidade de propagação muito superior à velocidade do som e, devido a isso, não é possível escutar a frequência sendo emitida.

Seu funcionamento consiste em dois objetos que se assemelham a caixas de som, um deles é o emissor de ondas e o outro é o receptor dessas ondas. Ao se refletirem e retornarem ao sensor, é calculado a distância percorrida desde o envio até o retorno.

Figura 39: Esquema de sinais emitidos pelo sensor HC-SR04

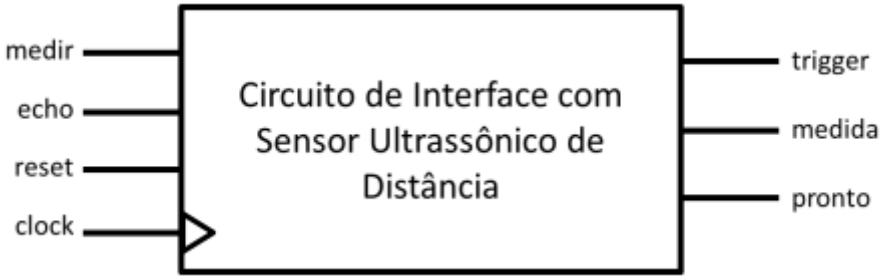


Fonte: EPUSP — PCS 3645 — Laboratório Digital II (USP)

Vale ressaltar que o sensor tem 4 pinos de conexão, sendo eles: GND, VCC, TRIGGER e ECHO.

O circuito desse sensor deve ser ativado por uma operação denominada MEDIR. Após isso, o circuito deve gerar um pulso TRIGGER e aguardar o pulso de resposta do ECHO. Em seguida, deve-se medir a largura do pulso de resposta e calcular a distância percorrida, em uma escala configurável desde milímetros até polegadas.

Figura 40: Representação de entradas e saídas do HC-SR04

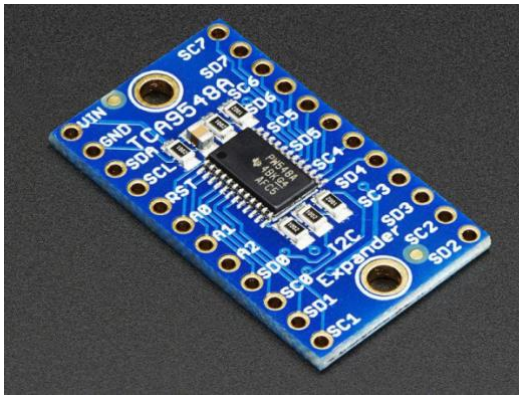


Fonte: EPUSP — PCS 3645 — Laboratório Digital II (USP)

2.2.4.5 Multiplexador de Entradas I2C

Esse multiplexador é também conhecido como TCA9548A I2C Multiplexer. É um atuador muito importante quando possuir mais de um componente com padrão de entradas I2C cujos endereços são idênticos e inalteráveis, já que o Arduino tem apenas uma dessas entradas e não conseguirá diferenciá-los. Para isso, então, o multiplexador aumenta a quantidade de entradas I2C. Essa placa atua como um “porteiro” direcionando os comandos certos para as entradas I2C de seus pinos de comando conectados.

Figura 41: Imagem da placa de circuito TCA9548A I2C Multiplexer



Fonte: Adafruit TCA9548A 1-to-8 I2C Multiplexer

Breakout

Existem diversas classificações para a pinagem dessa placa. Pode-se dividir em três categorias, sendo elas: pinos de energia, pinos de controle I2C e pinos de entrada.

É encontrado nos pinos de energia duas portas, sendo elas:

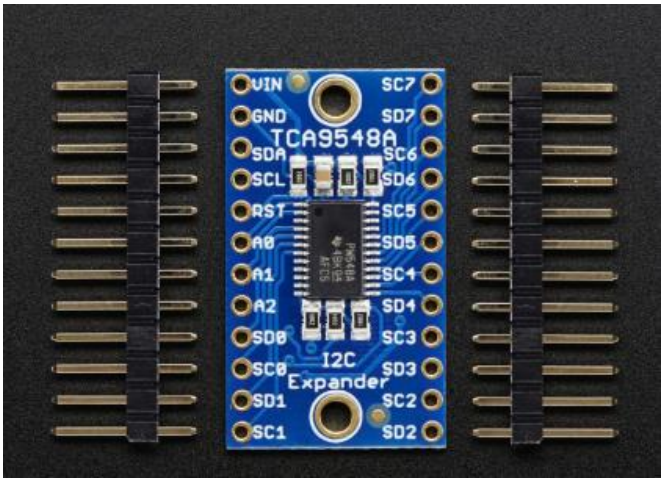
- ✓ VIN: essa é a porta de energia da placa, sua tensão de operação, por conta de seu chip processador, é de 3.5 à 5V;
- ✓ GND: pino comum de terra.

A seguir, os pinos de controle I2C vão encaminhar as informações e endereços cadastrados. Ao todo são:

- ✓ SCL: é o pino da entrada I2C do microcontrolador, sendo responsável pelas linhas de endereço;
- ✓ SDA: é outra entrada I2C do microcontrolador, sendo responsável pelas linhas de informações;
- ✓ Reset (RST): conhecido como pino de resetar, é responsável por reiniciar o chip do multiplexador;
- ✓ A0: esse pino faz parte um subconjunto de pinos de endereçamentos, ou seja, esses pinos são responsáveis pelo endereço do multiplexador. É o menos significativo;
- ✓ A1: é o segundo menos significativo em quesito de endereçamento, então poderíamos classificá-lo como intermediário;
- ✓ A2: é o endereçamento mais significativo dentre os três, então podemos dizer que os bits recebidos por essa entrada são os mais significativos.

Por fim, temos os pinos de entrada do multiplexador. Esses pinos, que são ao todo 16 (sendo 8 de SDA e 8 de SCL), fazem o papel de gerar outras portas I2C, formando sempre um par. Por exemplo, para ligar um sensor em uma entrada “I2C 0”, a placa possui dois pinos: o SDA 0 e o SCL 0, formando a entrada I2C 0. Ao todo temos desde SDA 0 até SDA 7, sendo aplicado o mesmo para o SCL gerando assim outras 8 portas de I2C.

Figura 42: Imagem da placa com a identificação dos pinos



Fonte: Adafruit TCA9548A 1-to-8 I2C Multiplexer

Breakout

2.2.4.6 Botão Switch

O botão switch é muito útil quando se trata de respostas diretas em circuitos. Se trata basicamente de um contato aberto e, quando há uma pressão aplicada nesta haste o contato será fechado, realizando a passagem de sinal.

Figura 43: Imagem de um botão switch

Fonte: https://daeletrica.com.br

2.2.4.7 Servo Motor

O servo motor consiste em um tipo de atuador bem comum para diversos projetos, tanto de fins eletrônicos quanto para mecânicos. Os mais comuns são com rotações até 180 graus, mas existem outros que podem ir até 360 graus.

Figura 44: Imagem de um servo motor

Fonte: https://images.tcdn.com.br

2.2.4.8 Motor DC

O motor Direct Current (DC) é um componente quase indispensável quando se trata de atuadores. Sua sigla, DC, significa que é destinado à corrente elétrica contínua. A partir de um módulo podemos controlar o seu sentido de rotação e velocidade pelo Arduino. O mesmo possui internamente uma caixa de redução (um jogo de engrenagens) para a geração de um torque que varia de motor para motor, podendo um ser mais rápido e conter menos torque ou até um motor lento que possui um torque elevado.

Figura 45: Imagem de um motor DC

Fonte: https://www.robocore.net

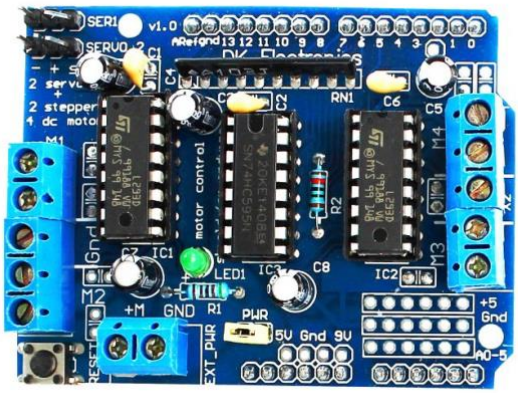
Figura 46: Imagem de um motor DC desmontado

Fonte: https://www.robocore.net

2.2.4.9 Módulo Shield Motor

Essa placa é um pouco complexa, mas com ela é possível controlar, de uma só vez, 4 motores DC e 2 servos motores.

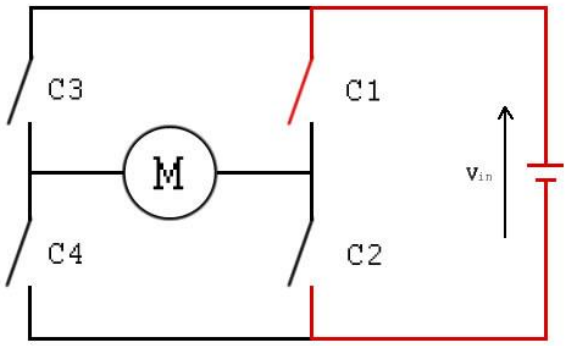
Figura 47: Imagem de um módulo shield motor



Fonte: https://www.usinainfo.com.br

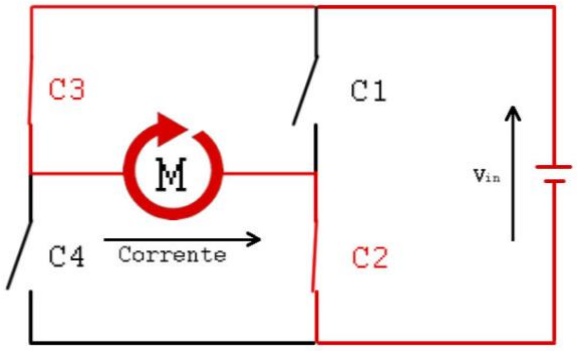
O motor é conectado entre duas séries em paralelo de contatos abertos. Para definir sua rotação basta fechar os contatos certos, como mostrado nas figuras a seguir:

Figura 48: Imagem representando o circuito elétrico sem o motor atuar



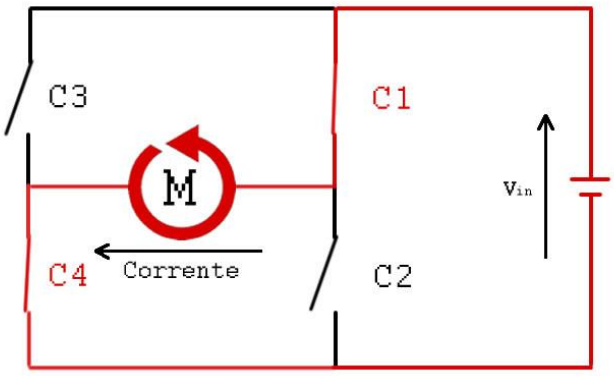
Fonte: https://www.usinainfo.com.br

Figura 49: Imagem representando o motor girando no sentido horário



Fonte: https://www.usinainfo.com.br

Figura 50: Imagem representando o motor girando no sentido anti-horário

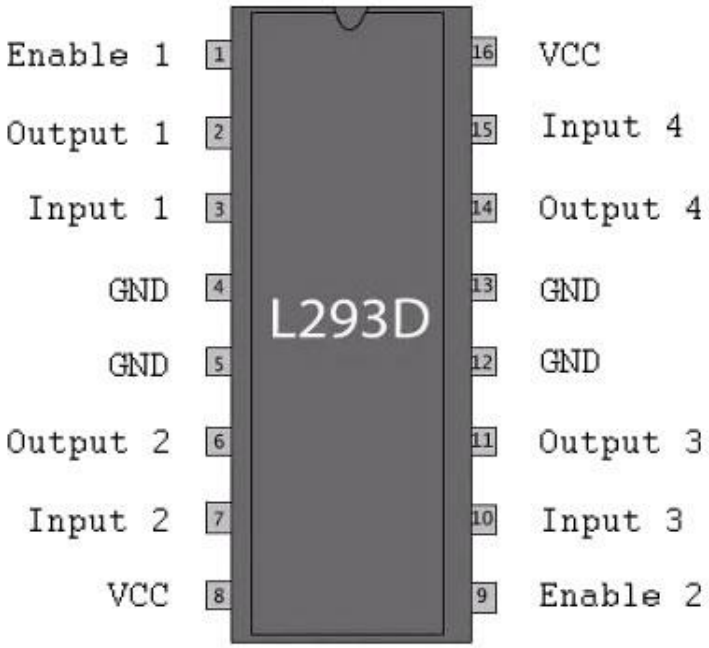


Fonte: https://www.usinainfo.com.br

Para concluir esses comandos é utilizado um circuito integrado (C.I), do qual ele é chamado L293D. Como mostrado na figura abaixo, ele é composto por diversas entradas e saídas. A alimentação do circuito é composta por duas

entradas, denominadas VCC. Outros tipos são as entradas e saídas, totalizando 4 inputs e outputs. Existem quatro portas denominadas de GND. Por fim, o pino “enable” se trata de regular a velocidade dos motores.

Figura 51: CI L293D

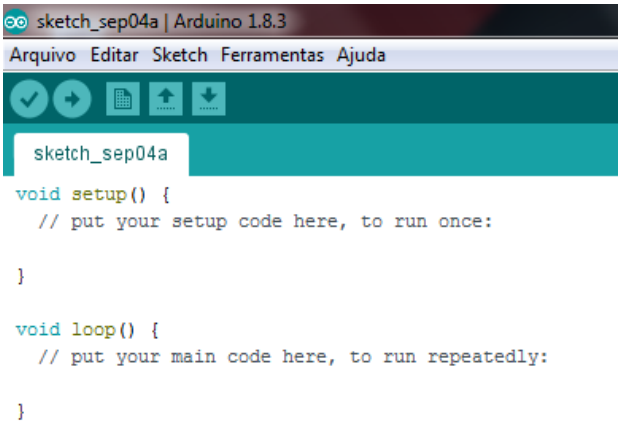


Fonte: https://www.usinainfo.com.br

2.2.5 ARDUINO INTEGRATED DEVELOPMENT ENVIRONMENT (IDE)

O Arduino IDE é um programa composto de uma interface onde é possível digitar as primeiras linhas de código com diversos comandos. É iniciado com os códigos que já vem por padrão quando criado um sketch no Arduino IDE.

Figura 52: Interface do Arduino IDE



Fonte: Autoria própria.

2.2.5.1 Principais Comandos do Arduino IDE

Pode-se digitar diversos comandos no Arduino IDE, de códigos para acender um simples led ou até para atuar motores por sensores ópticos. Serão destacadas funções muito utilizadas, independente do projeto que queira realizar.

2.2.5.1.1 Void Setup

Ao criar um bloco de programação, o void setup vem por padrão. Ele consiste em um código que será lido apenas uma vez, sendo utilizado para realizar definições universais para o programa.

O código pode ser escrito da maneira apresentada:

Figura 53: Representação do código Void setup

```
void setup() {
  // put your setup code here, to run once:

}
```

Fonte: Autoria própria.

Figura 54: Representação do código Void setup com pinos declarados

```
void setup() {
  // Aqui nós escrevemos o que será entrada e o que será saída no circuito

pinMode (led,OUTPUT);

pinMode (botão,INPUT);



}
```

Fonte: Autoria própria.

2.2.5.1.2 Void Loop

O significado é que o código colocado dentro dessa faixa será repetido diversas vezes. No geral, é onde a programação principal é escrita.

Figura 55: Representação do código Void loop

```
void loop() {
 // Aqui nós escrevemos o que será entrada e o que será saída no circuito

}
```

Fonte: Autoria própria.

Figura 56: Programação de motores com controle de velocidade dentro do comando Void loop

```
void loop() {

analogWrite(MD1,40);
analogWrite(ME1,40);

delay(2000);

analogWrite(MD1,80);
analogWrite(ME1,80);

delay(2000);

analogWrite(MD1,120);
analogWrite(ME1,120);

 delay(2000);

analogWrite(MD1,160);
analogWrite(ME1,160);

delay(2000);

analogWrite(MD1,200);
analogWrite(ME1,200);

delay(2000);


}
```

Fonte: Autoria própria.

2.2.5.1.3 Variáreis

Consistem em comandos que se pode colocar um valor variável dentro de um determinado circuito, ou seja, ele ter mais de um valor. Vale ressaltar que, se quiser que uma variável valha para a programação inteira, é colocada antes do setup(), caso ela seja para uma determinada parte da programação, inserida dentro do comando void ao qual for pertinente.

✓ **Variável Int:** Serve para a declaração de uma variável de valores integrais no circuito;

✓ **Variável define:** Consiste no mesmo princípio do comando int, mas será utilizado quando o valor for fixo. Muito utilizado para relacionar o nome de um componente à porta que ele está ligado;

✓ **Variável uint16_t:** Determinar que uma variável venha a ter o valor de 16 bits na memória do Arduino;

✓ **Variável float:** Irá trabalhar com valores decimais. Pode ser utilizado por alguns sensores;

✓ **Variável include:** Consiste em colocar uma biblioteca de algum determinado sensor ou atuador, podendo ser um sensor de cores, display LCD ou até um servomotor.

Figura 57: Representação da variável int para a declaração dos componentes

```
int Led  = 2;
int bot1 = 3;
int bot2 = 4;

void setup() {
pinMode(Led  , OUTPUT);
pinMode(bot1 , INPUT);
pinMode(bot2 , INPUT);

}
```

Fonte: Autoria própria.

Figura 58: Representação da variável define para alguns sensores

```
#define commonAnode true
#define SensorEsqF 22   //azul SENSOR SEGUE FAIXA ESQUERDA DA PONTA
#define SensorEsq 23    //amarelo SENSOR SEGUE FAIXA DIREITA
#define SensorCen 24    //cinza SENSOR SEGUE FAIXA CENTRAL
#define SensorDir 25    //laranja SENSOR SEGUE FAIXA ESQUERDA
#define SensorDirF 26   //marrom SENSOR SEGUE FAIXA DIREITA DA PONTA
#define SensorInc 27    //azul SENSOR INCLINAÇÃO
#define Led 28          //branco LED
#define Buzzer 29       //roxo BUZZER
#define Trig 30         //verde ULTRASSÔNICO
#define Echo 31         //vermelho ULTRASSÔNICO
#define ObsEsqLat 32    //laranja OBSTÁCULOLATERALESQUERDA
#define ObsDirLat 33    //marrom OBSTÀCULOLATERALDIREITA
#define LedEsq 34
#define LedDir 35
#define SensorObsEsq 36 // SENSOR DE OBSTACULO ESQUERDA
#define SensorObsDir 37 // SENSOR DE OBSTACULO DIREITA
#define LedCen 38
#define SensorDtcEsq 62 // SENSOR DETECTA VITIMA ESQUERDA
#define SensorDtcDir 63 // SENSOR DETECTA VITIMA DIREITA
```

Fonte: Autoria própria.

Figura 59: Representação da variável uint16_t utilizado para um sensor de cores

```
void leitura(){


 mux.openChannel(7);
 //esq.setInterrupt(false);
 uint16_t r1, g1, b1, c1, lux1, ct1;
 esq.getRawData(&r1, &g1, &b1, &c1);
 lux1 = esq.calculateLux(r1, g1, b1);
 ct1 = esq.calculateColorTemperature_dn40(r1, g1, b1, c1);
 c1 = c1;
```

Fonte: Autoria própria.

Figura 60: Representação da variável float para a declaração de um sensor

```
float myfloat;
float sensorCalibre = 1.114;

int x;
int y;
float z;

x = 1;
y = x/2;
z = (float)x / 2.0
```

Fonte: Autoria própria.

Figura 61: Representação do comando include para declarar bibliotecas

```
#include <HCSR04.h>
#include <Adafruit_TCS34725.h>
#include <Wire.h>
#include <TCA9548A.h>
#include <VarSpeedServo.h>
#include <AFMotor.h>
```

Fonte: Autoria própria.

2.2.5.1.5 Comandos

- ✓ **Comando pinMode:** Uma função indispensável nas programações com o Arduino. É ele que irá definir quais pinos serão entradas (INPUT) ou saídas (OUTPUT);
- ✓ **Comando Wire:** É utilizado para permitir a conexão pelo barramento I2C;
- ✓ **Comando digitalRead:** Lê na memória um valor digital, podendo ser a informação de um sensor infravermelho por exemplo;

✓ **Comando digitalWrite:** Ao invés de obter a leitura na memória, será realizado a escrita dos valores enviados pela placa;

✓ **Comando analogRead**: Irá enviar valores analógico pela placa;

✓ **Comando analogWriteI:** Irá realizar a leitura de valores analógicos;

✓ **Comando If e Else:** Esses comandos representam uma função lógica que estabelece uma condição em cima de algum valor (geralmente uma variável) e, caso essa condição seja verdadeira, ele executa uma ordem e, caso falsa, pode executar outra ordem ou nenhuma;

Figura 62: Representação do comando pinMode para declarar as entradas e saídas do circuito

```
pinMode(SensorDir,INPUT);
pinMode(SensorCen,INPUT);
pinMode(SensorEsq,INPUT);
pinMode(Buzzer,OUTPUT);
pinMode(SensorObsEsq,INPUT);
pinMode(SensorObsDir,INPUT);
pinMode(SensorDtcEsq ,INPUT);
pinMode(SensorDtcDir, INPUT);
pinMode(SensorInc, INPUT);
pinMode(Led, OUTPUT);
pinMode(SensorDirF, INPUT);
pinMode(SensorEsqF, INPUT);
pinMode(LedDir, OUTPUT);
pinMode(LedEsq, OUTPUT);
pinMode(LedCen, OUTPUT);
```

Fonte: Autoria própria.

Figura 63: Representação do comando Wire para a utilização do barramento I2C

```
void setup() {
 Serial.begin(9600);
 Wire.begin();
 mux.begin(Wire);
 mux.closeAll();
 pinMode(SensorDir,INPUT);
 pinMode(SensorCen,INPUT);
 pinMode(SensorEsq,INPUT);
 pinMode(Buzzer,OUTPUT);
```

Fonte: Autoria própria.

Figura 64: Representação do comando digitalWrite

```
digitalWrite(Buzzer, HIGH);
digitalWrite(Led, HIGH);
delay(250);
digitalWrite(Buzzer, LOW);
digitalWrite(Led, LOW);
delay(250);
digitalWrite(Buzzer, HIGH);
digitalWrite(Led, HIGH);
delay(250);
```

Fonte: Autoria própria.

Figura 65: Representação do comando digitalRead

```
Sensor1 = digitalRead(SensorEsqF);
Sensor2 = digitalRead(SensorEsq);
Sensor3 = digitalRead(SensorCen);
Sensor4 = digitalRead(SensorDir);
Sensor5 = digitalRead(SensorDirF);
Sensor6 = digitalRead(SensorObsEsq);
Sensor7 = digitalRead(SensorObsDir);
```

Fonte: Autoria própria.

Figura 66: Representação do comando analogWrite

```
void loop() {

analogWrite(MD1,40);
analogWrite(ME1,40);

delay(2000);

analogWrite(MD1,80);
analogWrite(ME1,80);

delay(2000);
```

Fonte: Autoria própria.

Figura 67: Representação do comando analogRead

```
void loop() {
  // put your main code here, to run repeatedly:
  valesq = analogRead(A13);
  Serial.println(analogRead(A13));

  }
```

Fonte: Autoria própria.

Figura 68: Representação da condição if e else

```
if(comando == 0){
  vacuocurvaesq = 20;
  vacuocurvadir = 20;
  reto();
}else{
 if(comando == 1){
   esquerdal = 1;
   direital = 0;
   vacuocurvaesq = 20;
   esquerda();
}else{
 if((comando == 2) && (arranquei == 0)){
   //auxilioesq == auxilioesq++;
 motor1.setSpeed(veloplano);
 motor2.setSpeed(veloplano);
 arrancando = 15;
 arranquei = 1;
 arrancadaesq();
```

Fonte: Autoria própria.

3 DESENVOLVIMENTO PRÁTICO

Agora que já foram mencionadas e explicadas as bases teóricas que a equipe possui, será falado dos processos para a formação final do projeto. Os assuntos tratados irão girar em torno de todo o processo de desenvolvimento prático a respeito das montagens e programação do robô.

3.1 Montagem do Chassi

No começo do desenvolvimento do projeto, foi comprado e utilizado um chassi feito de PSAI (um material semelhante ao acrílico). As peças possuíam um bom acabamento, com corte feitos a laser, porém era de pouca resistência e, consequentemente, frágil em possíveis pontos de esforço (locais de impacto e onde normalmente se apoia a mão ao segurá-lo).

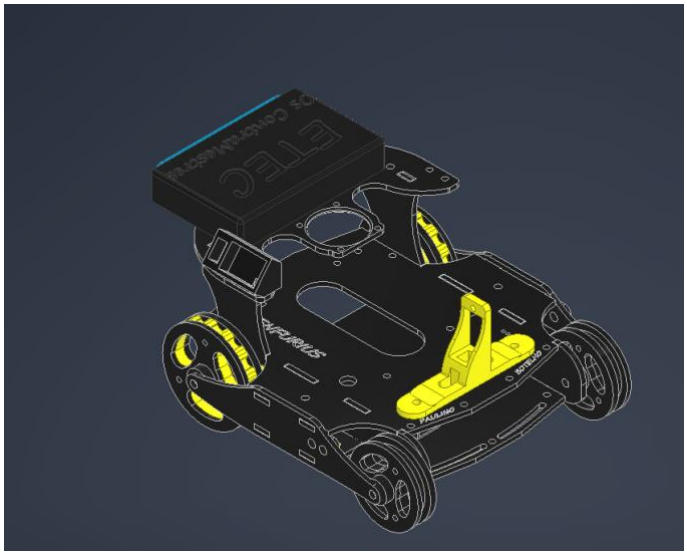
Após um período de 10 meses foi notado sinais de ruptura e, em um treinamento para a competição, a estrutura se desfez na mão de um dos integrantes da equipe.

Figura 69: Primeira versão do chassi

Fonte: https://www.robocore.net

Em seguida, foi realizado um estudo sobre as antigas estruturas e, baseado no desenho técnico disponibilizado no site em que foi efetuado a compra do primeiro modelo, o chassi foi reprojetado, impresso inteiramente em um material de Ácido Poliláctico (PLA).

Figura 70: Segunda versão de Chassi



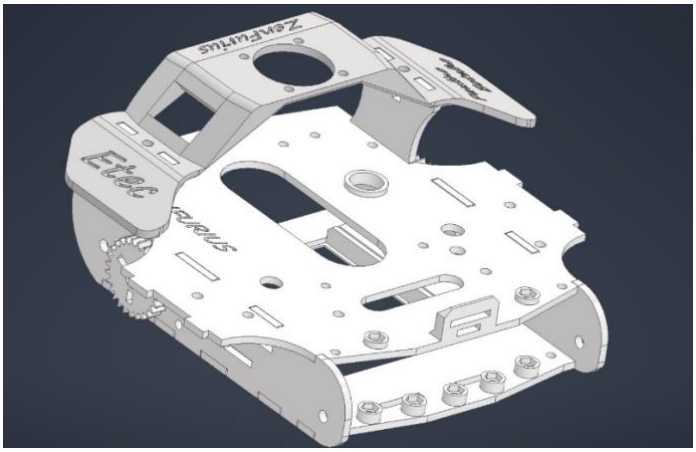
Fonte: autoria própria.

Para evitar os alguns possíveis danos já causados por pontos de esforço, foi adicionado ao robô um sistema de amortecimento.

Seguido de vários testes, a equipe identificou perda de eficiência dos motores devido ao novo material. Devido aos locais onde o peso se concentra, uma curvatura no chassi, que estava gerando atrito nos motores, foi vista. Com a competição do ano de 2023 se aproximando, não foi possível corrigir o erro de forma eficiente.

Com a equipe já classificada para a fase estadual, novos estudos de materiais e estruturas foram feitos o mais rápido possível. Na atualização foram considerados os materiais de Acrilonitrilo-butadieno-estireno (ABS) e Polyethylene-terephthalate-glycol (PETG). A escolha realizada foi o PETG, que se justifica por possuir altíssima resistência mecânica e térmica, decepcionando apenas no acabamento grosseiro.

Figura 71: O Chassi Final



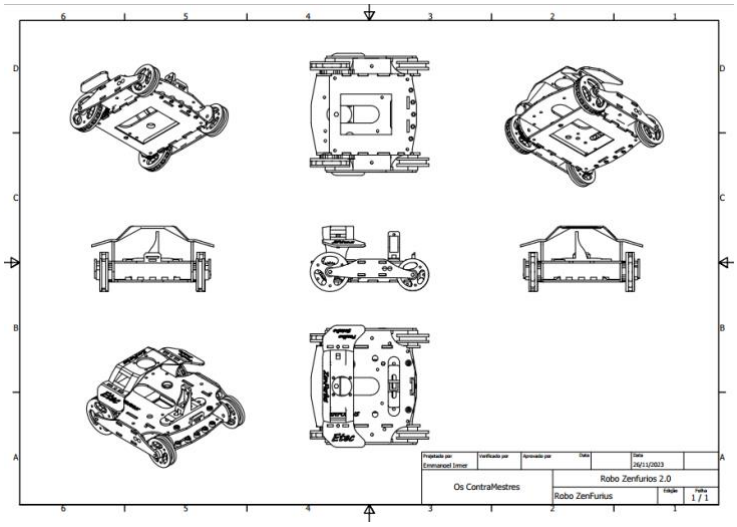
Fonte: autoria própria.

Na última versão, o centro de massa do carrinho foi posicionado nas bases do assoalho ao realocarmos as baterias para lá. Com um centro de gravidade baixo foi evitado tombamentos em terrenos íngremes e curvas. Outras atualizações feitas foram as rodas reforçadas (nas quais foram submetidas a um teste, através de uma dinâmica em sala de aula onde os alunos foram intimados a quebrá-la, que durou 2 horas), estrutura reprojetada visando mais eficiência e melhor portabilidade de novos sensores e a estética.

3.1.1 Desenhos Técnicos

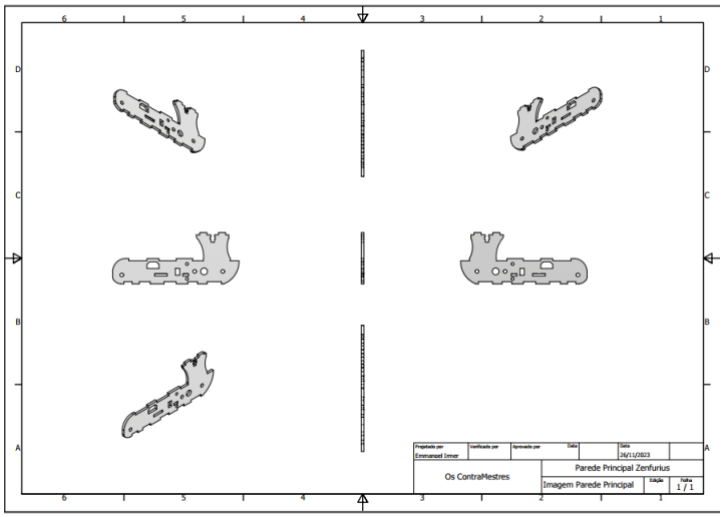
Agora será apresentado os desenhos técnicos desenvolvidos pela equipe para o chassi final.

Figura 72: Robô inteiro



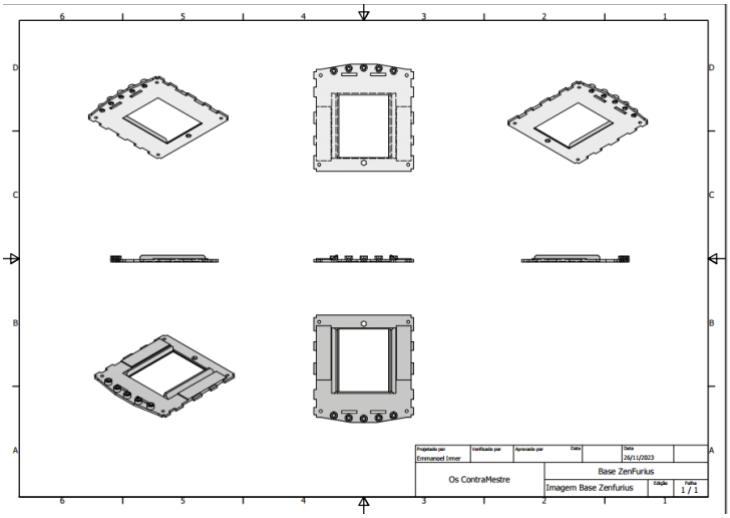
Fonte: Autoria própria.

Figura 73: Parede principal



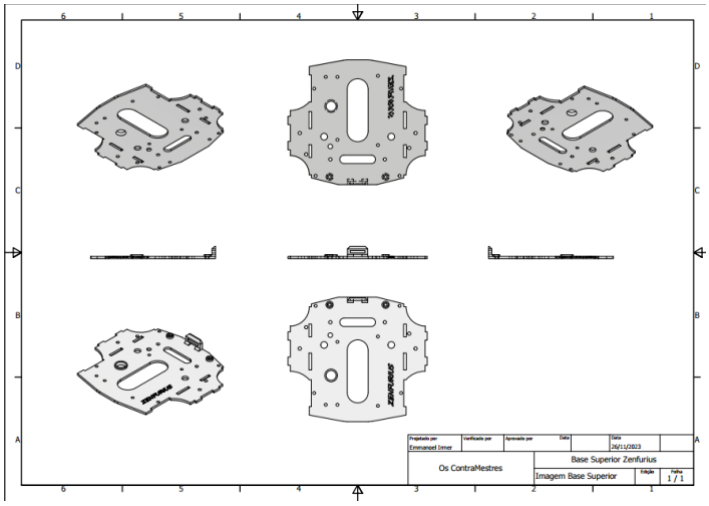
Fonte: Autoria própria.

Figura 74: Base



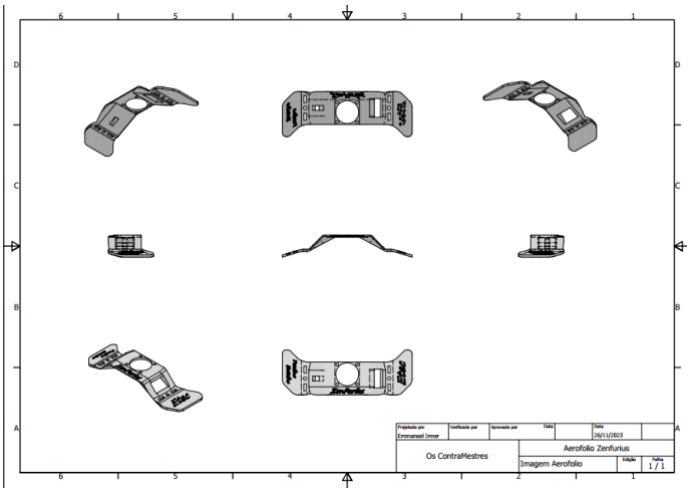
Fonte: Autoria própria.

Figura 75: Base superior



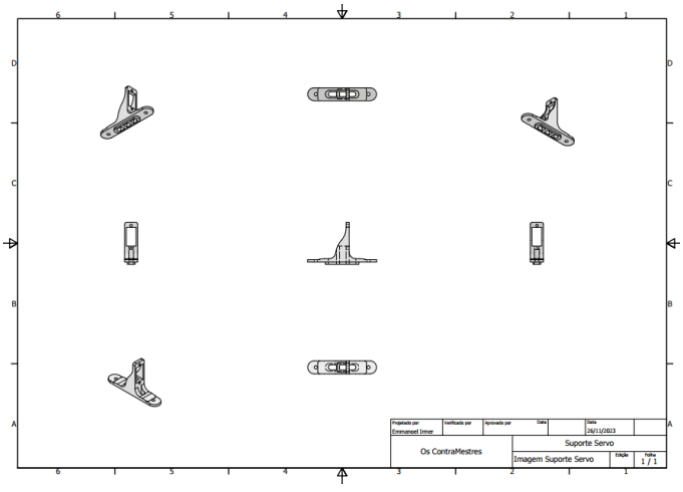
Fonte: Autoria própria.

Figura 76: Aerofólio



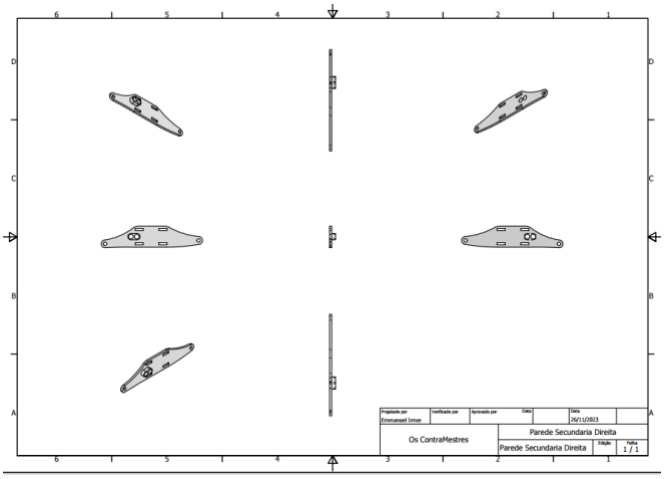
Fonte: Autoria própria.

Figura 77: Suporte do servo



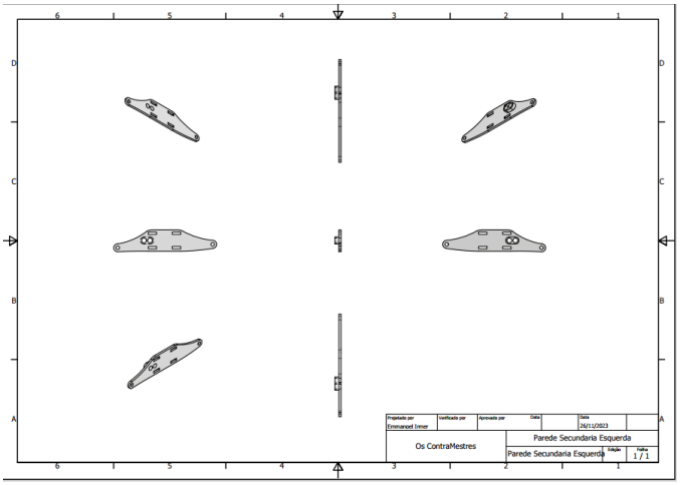
Fonte: Autoria própria.

Figura 78: Parede secundária direita



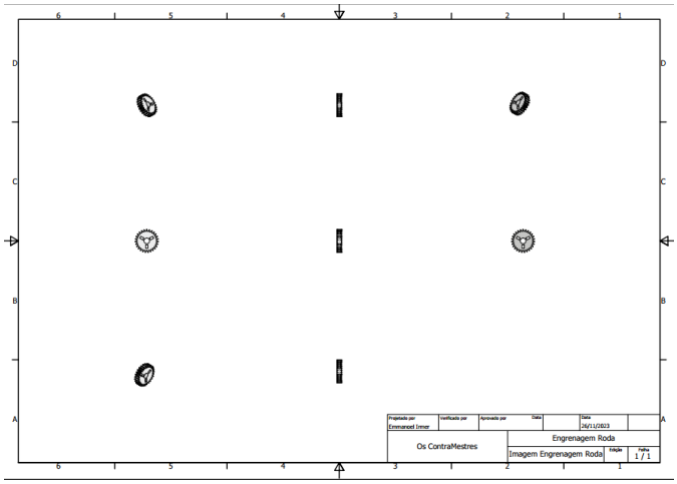
Fonte: Autoria própria.

Figura 79: Parede secundária esquerda



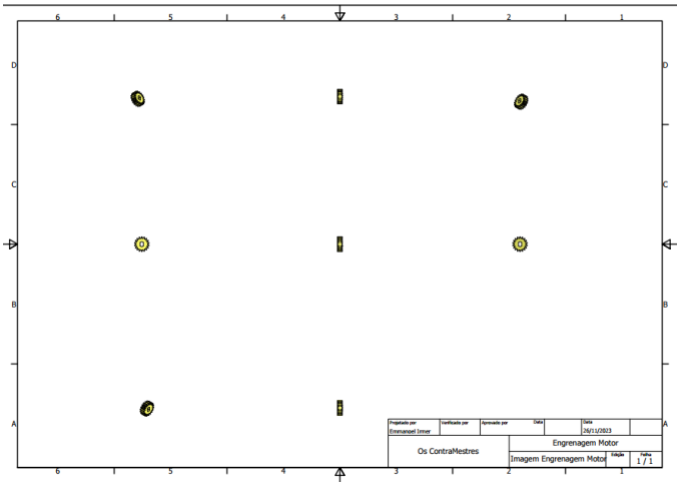
Fonte: Autoria própria.

Figura 80: Engrenagem da roda



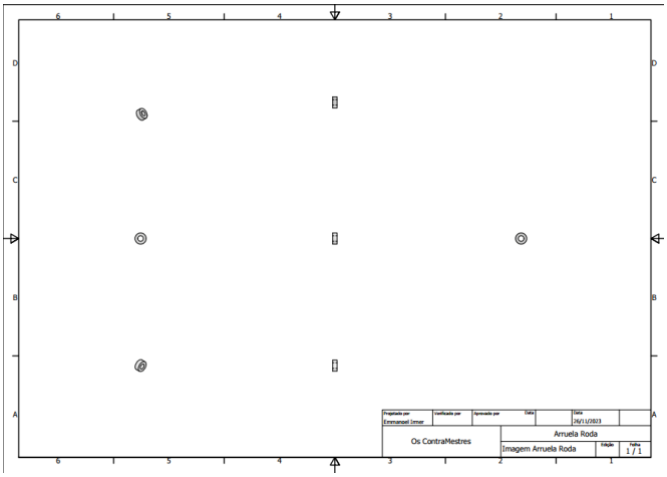
Fonte: Autoria própria.

Figura 81: Engrenagem do motor



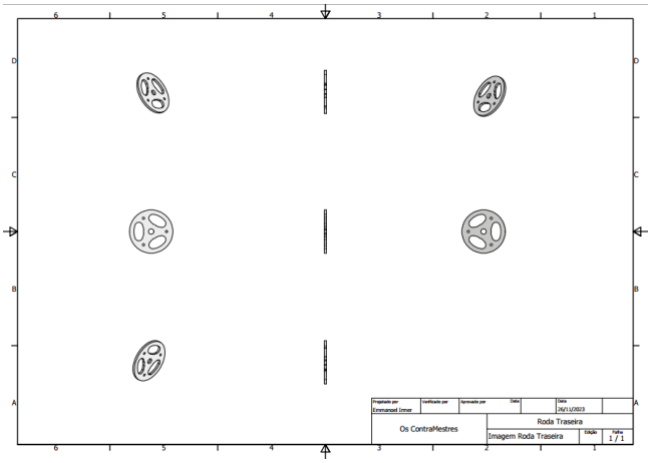
Fonte: Autoria própria.

Figura 82: Arruela da roda



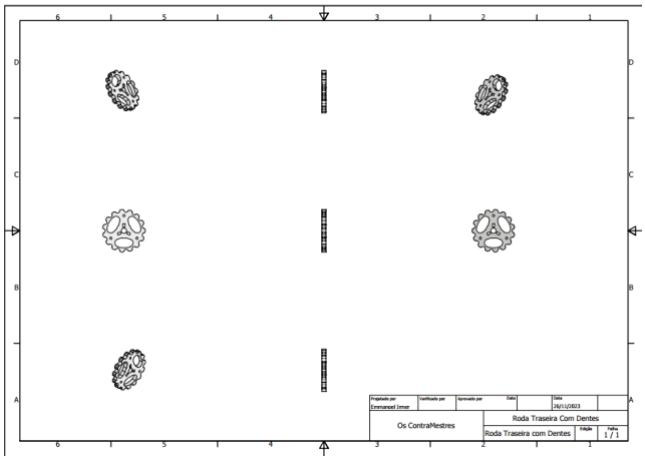
Fonte: Autoria própria.

Figura 83: Roda traseira



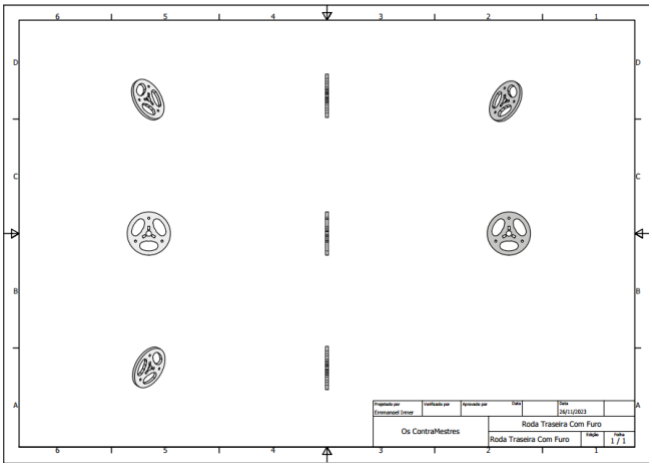
Fonte: Autoria própria.

Figura 84: Roda traseira dentada



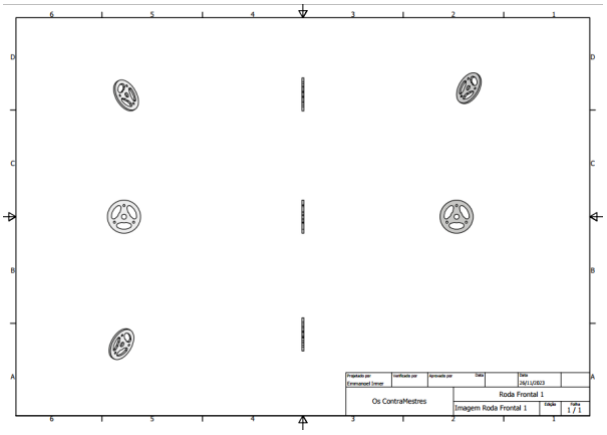
Fonte: Autoria própria.

Figura 85: Roda traseira perfurada



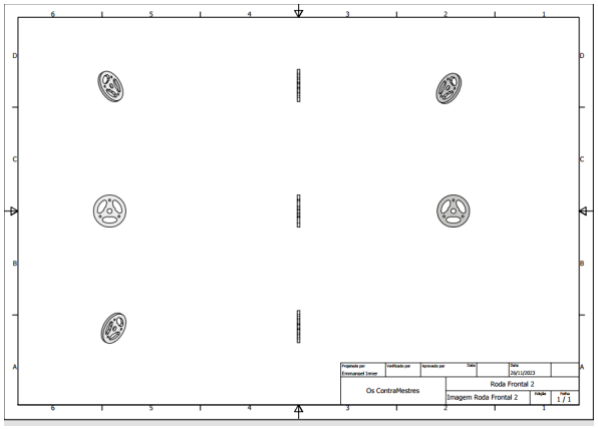
Fonte: Autoria própria.

Figura 86: Roda frontal



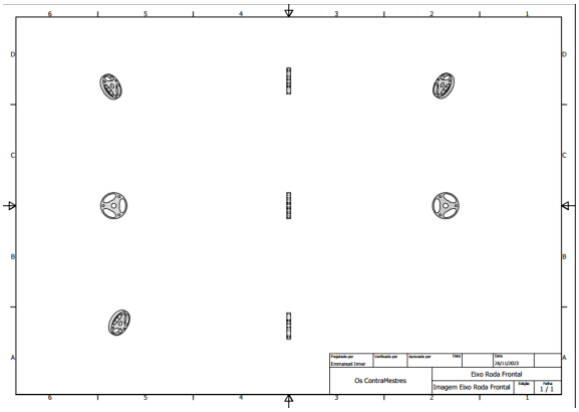
Fonte: Autoria própria.

Figura 87: Segunda roda frontal



Fonte: Autoria própria.

Figura 88: Eixo roda frontal

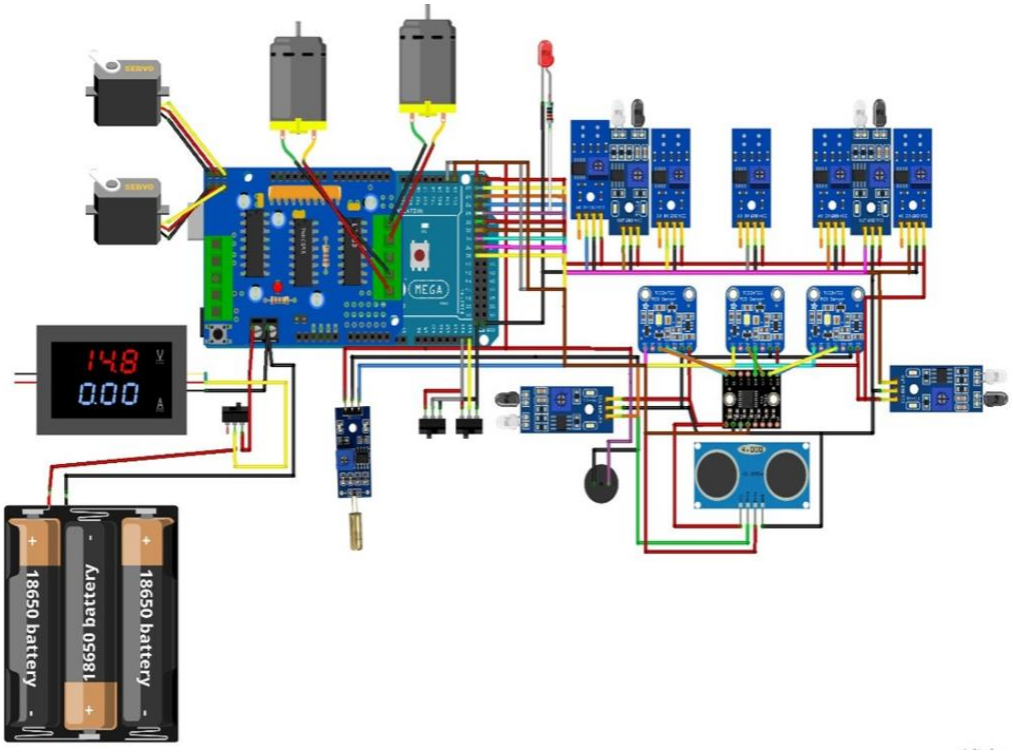


Fonte: Autoria própria.

3.2 Esquema Elétrico

Será mostrado uma imagem do esquema elétrico que contém as ligações eletrônicas feitas.

Figura 89: Esquema elétrico



Fonte: Autoria própria.

3.2.1 A Alimentação

Arduino e Shield

Para alimentar o arduino mega e todos os sensores, que estão ligados em comum no GND e 5V, foi utilizado uma bateria de 12 volts.

Separadamente, foi utilizado três baterias de lítio, em um total de 12 volts, para alimentar a Shield, que ativa os dois motores.

3.3 Programação

3.3.1 O Básico

Agora será falado sobre a programação de alguns dos componentes e, também, a lógica por trás deles.

3.3.1.1 Motores e Shield

O Arduino, através da biblioteca “Adafruit Motor Shield”, se comunica com a Shield, e ela, por conseguinte, controla os motores, que compõe a parte mais essencial do nosso projeto. Assim, o conhecimento de alguns comandos básicos é necessário para podermos controlá-los. Entre eles:

- ✓ **AF_DCMotor ‘nome do motor’ (‘número do pino’):** Esse comando é utilizado para criar um objeto, no caso dar um nome à um dos motores (um nome de uma única palavra), o qual será utilizado para dirigir um comando àquele motor na presente programação;

- ✓ **‘nome do motor’.setSpeed(‘velocidade escolhida’):** Esse comando é utilizado para determinar a voltagem que a Shield deverá liberar aos motores. Ou seja, o número digitado irá indicar um valor que faz variar a voltagem do mínimo ao máximo da bateria. Ademais, se o pino que enviará esse valor para a Shield for pwm, o número digitado estará entre 0 e 253; caso seja analógico, o valor variará de 0 até 1023. Assim, o comando altera a velocidade em que o motor escolhido irá atuar;

- ✓ **‘nome do motor’.Run(‘direção de rotação’):** Esse comando determina a direção em que o motor irá atuar. Caso seja necessário que o motor siga no sentido horário, será digitado FORWARD, caso seja o sentido anti-horário, BACKWARD e, por fim, caso seja ficar parado, RELEASE.

3.3.1.2 Garra

O Arduino, através da biblioteca “Servo”, se comunica com os servos motores que, por conseguinte, movimentam-se e operam o braço e a garra. Logo, para realizarmos essa comunicação, foram utilizados alguns comandos básicos. São eles:

- ✓ ‘**Nome do servo’.attach(‘número do pino’):** Esse comando também determina um nome ao módulo, no caso o servo, a ser utilizado no pino escolhido;

- ✓ **‘Nome do servo’.write(‘grau’):** Esse comando é utilizado para determinar o giro que o servo deverá realizar a partir de uma posição inicial (0). Contudo, esse valor é limitado entre 0 à 180.

3.3.1.3 Sensores

Os sensores são os olhos e ouvidos do robô, são a principal comunicação entre o ambiente e o “cérebro” (Arduino). Portanto, serão utilizados os seguintes

comandos para que possa ser recebido e armazenado as informações enviadas pelos sensores:

- ✓ **UltraSonicSensorDistance ‘nome do sensor’ (‘pino trig’, ‘pino echo’):** Essa é a forma que se cria o objeto do sensor ultrassônico, dando um nome a ele e, desde já, definindo a pinagem;
- ✓ **‘nome do sensor’.measureDistanceCm():** Essa é a função, utilizada através da biblioteca “HASR04”, usada para realizar a leitura do sensor ultrassônico. Logo, o valor recebido é a distância em centímetros do objeto mais próximo.

3.3.1.4 Sensores de Cor e Multiplexador

Para ser possível a comunicação entre o Arduino e os três sensores de cor foi necessário a utilização do multiplexador. Logo, foi necessário o conhecimento das seguintes funções:

- ✓ **TCA9548A ‘nome do multiplexador:** Essa função, utilizada através da biblioteca “TCA9548A”, cria o objeto e dá um nome ao multiplexador;
- ✓ **‘nome do multiplexador. Begin (wire):** Essa função inicia o objeto do multiplexador e determina a utilização da biblioteca “wire”, a qual está incluída a utilização do barramento I2C;
- ✓ **‘nome do multiplexador.close/open(‘número do canal’):** Essa função realiza abertura, ou o fechamento, de um dos canais do multiplexador. Logo, com essa função, é onde encerra e inicia a comunicação do Arduino com um dos sensores de cor, de acordo com o canal escolhido;
- ✓ **Adafruit_TCS34725 ‘nome do sensor’ = Adafruit_TCS34725 (TCS34725_INTEGRATIONTIME_2_4MS, TCS34725_GAIN_1X):** Essa é a função que, além de dar um nome para o sensor de cor através da biblioteca “Adafruit_TCS34725”, define o tempo em que o sensor levará para realizar a leitura e, também, multiplica o valor da leitura;
- ✓ **‘nome do sensor’.getRawData(&‘variável do vermelho’, &‘variável do verde’, &‘variável do azul’, &‘variável da luz’):** Essa

função cria “ponteiros” dentro da memória do sensor, ponteiros esse que dão acesso às variáveis que irão guardar o valor das leituras realizadas pelo sensor;

- ✓ **‘nome do sensor’.calculateLux(‘variável do vermelho, ‘variável do verde’, ‘variável do azul’):** Esse função realiza a leitura de luminosidade dentro do espectro de cada cor;
- ✓ **‘nome do sensor.calculateColorTemperature_dn40(‘variável do vermelho’, ‘variável do verde’, ‘variável do azul’, ‘variável da luz’):** Essa função realiza a leitura da temperatura da cor de acordo com seu espectro.

3.3.2 A Lógica.

Por trás das ações do carrinho existem algumas lógicas que precisam ser destacadas por serem essenciais.

3.3.2.1 As Retas

Dentro do “Void reto()” os motores sempre seguem em frente. Se os sensores de obstáculo detectarem algo, o robô se alinhará e realizará o “Void obstáculo()”. Se a variável de inclinação constar que o carrinho está inclinado, ele entrará em seu “Void rampa()”. Na situação em que os sensores de cor detectarem uma cor verde, o robô realizará um avanço e seguirá para o “void” que curvará para o respectivo lado em que a cor verde se encontra. Em relação às curvas fechadas, o robô realizará um avanço antes de ir ao “void” que o faz curvar (a não ser que precise simplesmente ajustar sua posição na linha).

3.3.2.2 As Curvas

Para isso, os motores são invertidos de acordo com o lado que deve virar.

Muitos “voids” são responsáveis pelas curvas. Antes de realizar curvas fechadas, a programação realiza um pequeno avanço nas esteiras (determinado por um certo tempo) e, então, realiza as curvas. Sendo a intenção apenas ajustar a posição do robô, as esteiras invertem sem realizar o avanço.

3.3.2.3 Os Gaps

Se os sensores não detectarem nenhuma linha após um certo tempo curvando (estar em cima de um gap, por exemplo) os motores voltam para o “Void reto()”. Caso o carrinho esteja fora da linha por um certo tempo (não estava alinhado com a linha após o gap), ele entra no “void” que vira para o lado da última curva que ele fez.

3.3.2.4 Os Obstáculos

Após entrar na programação do obstáculo, o robô realiza um contorno pela esquerda, controlado por delays, e, antes de retornar à linha final, ele para. Ele entra em um “void” específico para seguir em frente até encontrar a linha e avançar um pouco dela. Por fim, curva para a esquerda até se alinhar com ela.

3.3.2.5 A Rampa

Dentro do “void rampa()”, os motores são postos em uma alta velocidade e as curvas são feitas de uma maneira mais leve. Para curvar os motores não invertem, mas, sim, apenas diminuem a velocidade, em relação ao outro, de acordo com a direção desejada. Quando os sensores detectarem a cor prata, foi entrado na arena.

3.3.2.6 A Arena

Na arena, o robô deveria se orientar unicamente com os sensores de obstáculo, o sensor de cor central, o sensor ultrassônico e os sensores da lateral. Inicialmente ele seguiria em frente até que somente o sensor da direita detecte algo, pois se os dois detectarem, é uma parede branca e, nesse caso, é necessário girar 90 graus e reiniciar.

Após detectar a área de resgate, o robô viraria para a esquerda e seguiria um pouco em frente, para se afastar da parede e poder se alinhar com a área. Ao findar, ele curvaria para a direção da área de resgate e, quando o sensor ultrassônico detectasse a menor distância, ele seguiria em frente, pois está alinhado e conseguiria realizar a verificação da cor da arena.

Se fosse verde, ele jogaria o kit de resgate na área, caso contrário, ele viraria para a esquerda até o sensor lateral detectar que o robô está alinhado com a lateral da área e, em seguida, seguiria em frente até encontrar a parede branca,

virar um pouco à esquerda e reiniciar. Depois de entregar o kit de resgate, o robô se afastaria da área em direção ao centro da arena.

No centro, ele realizaria uma volta completa afim de detectar uma vítima com o sensor ultrassônico. Ao se aproximar, a pegaria com a garra, retornaria ao centro e realizaria o caminho reverso até voltar à área. Por fim, a jogaria na área de resgate.

3.3.3 A Prática da Programação

É fácil definir o que o carrinho deve fazer, porém o complicado está em programar o robô para realizar os algoritmos.

3.3.3.1 Os Voids

Os voids foram utilizados para deixar a programação mais organizada. Eles permitem separar um conjunto de ações que o robô irá realizar ou até mesmo permitir uma maior precisão em relação à identificação do robô na linha. Por exemplo, o “void reto()”:

Figura 90: Imagem de um void sendo selecionado

```
if(comando == 0){
  vacuocurvaesq = 20;
  vacuocurvadir = 20;
  reto(); //LINHA RETA
}else{
```

Fonte: Autoria própria.

Para o void ser selecionado, a variável “comando” dever ter valor 0 e, se ele possuir outro valor, será selecionado outro void.

Dentro dos voids ocorre a seleção do novo valor da variável “comando”, a partir da leitura de sensores, para indicar os próximos passos a serem seguidos, ou seja, a lógica está contida no conjunto dos voids. Como cada um chama o próximo? Por exemplo, em que momento que o “void reto()” irá indicar o “void esquerda()”? Veja na imagem:

Figura 91: Imagem do “void reto()” alterando a variável comando

```
 if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0) && (Sensor5 == 0)
  comando = 1;
}else{
 if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0) && (Sensor5 == 0)
  comando = 1;
}else{
  if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0) && (Sensor5 == 0)
  comando = 1;
}else{
```

Fonte: Autoria própria.

Seguindo o modelo proposto, qualquer variação na leitura dos sensores irá indicar as próximas ações para o robô. Dessa forma, através de leituras iguais, o robô reagirá de forma diferente dependendo da situação que ele se encontra. Por exemplo, ele reagirá de uma forma caso os sensores fiquem pretos enquanto anda reto e reagirá de forma diferente caso os sensores fiquem pretos enquanto faz uma curva.

3.3.3.2 “Variáveis de sequência” e “variáveis contadoras”

Algumas vezes, para evitar a criação de novos voids, foi feita uma programação sequencial dentro de um void já existente. Era indicado a ação a ser feita através da variação de uma variável dentro do próprio void.

Quando necessário que uma ação seja realizada por certo tempo, a função “delay” não é recomendada, pois paralisa a programação naquele ponto. Para contornar o problema foram utilizadas variáveis que, a cada momento que a programação rodava, ganhavam um valor e, assim, serviam como contadoras. Era escolhido uma quantidade de repetições e, quando atingida, passava às próximas ações.

Figura 92: Exemplo da “variável de sequência” “passando” e da “variável contadora” “bohr”

```
if((Sensor6 == 1) && (passando == 2)){
  passando = 1;
  bohr = 0;
}

if((Sensor7 == 1) && (passando == 1)){
 passando = 2;
  bohr = 0;

}

  if((bohr > 25) && (passando == 1)){
   motor1.run(FORWARD);
   motor2.run(FORWARD);
   delay(50);
   passando = 2;
   bohr = 0;
 }else{
  if((bohr > 25) && (passando == 2)){
   motor1.run(FORWARD);
   motor2.run(FORWARD);
   delay(50);
   passando = 1;
   bohr = 0;
  }}
```

Fonte: Autoria própria.

O recorte da programação a cima pertence ao “void grandeajuste()”, cuja função seria guiar o carrinho até a primeira área de resgate e posicioná-lo para ler a cor da área. O recorte pertence a uma parte específica que faz o carrinho ir um pouco para frente caso a posição não seja favorável mesmo após 25 repetições do arco para esquerda ou direita.

3.3.3 Programação concluída

```
// !!!! ESSA PROGRAMAÇÃO É ESPECIFICAMENTE PARA O ARDUINO MEGA !!!!
#include <HCSR04.h>
#include <Adafruit_TCS34725.h>
#include <Wire.h>
#include <TCA9548A.h>
#include <VarSpeedServo.h>
#include <AFMotor.h>
```

```
#define commonAnode true
#define SensorEsqF 22   //azul SENSOR SEGUE FAIXA ESQUERDA DA
PONTA
#define SensorEsq 23    //amarelo SENSOR SEGUE FAIXA DIREITA
#define SensorCen 24    //cinza SENSOR SEGUE FAIXA CENTRAL
#define SensorDir 25    //laranja SENSOR SEGUE FAIXA ESQUERDA
#define SensorDirF 26   //marrom SENSOR SEGUE FAIXA DIREITA DA
PONTA
#define SensorInc 27    //azul SENSOR INCLINAÇÃO
#define Led 28          //branco LED
#define Buzzer 29       //roxo BUZZER
#define Trig 30         //verde ULTRASSÔNICO
#define Echo 31         //vermelho ULTRASSÔNICO
#define ObsEsqLat 32    //laranja OBSTÁCULOLATERALESQUERDA
#define ObsDirLat 33    //marrom OBSTÀCULOLATERALDIREITA
#define LedEsq 34
#define LedDir 35
#define SensorObsEsq 36 // SENSOR DE OBSTACULO ESQUERDA
#define SensorObsDir 37 // SENSOR DE OBSTACULO DIREITA
#define LedCen 38
#define SensorDtcEsq 62 // SENSOR DETECTA VITIMA ESQUERDA
#define SensorDtcDir 63 // SENSOR DETECTA VITIMA DIREITA

UltraSonicDistanceSensor ValCm(Trig, Echo);
TCA9548A mux;
Adafruit_TCS34725 esq =
Adafruit_TCS34725(TCS34725_INTEGRATIONTIME_2_4MS,
TCS34725_GAIN_1X);
Adafruit_TCS34725 cen =
Adafruit_TCS34725(TCS34725_INTEGRATIONTIME_614MS,
TCS34725_GAIN_1X);
Adafruit_TCS34725 dir =
Adafruit_TCS34725(TCS34725_INTEGRATIONTIME_2_4MS,
TCS34725_GAIN_1X);
```

```
AF_DCMotor motor1(4); //Motor da Esquerda
AF_DCMotor motor2(3); //Motor da Direita
VarSpeedServo Servo1; //Servo Braço
VarSpeedServo Servo2; //Servo Garra
int Sensor1;
int Sensor2;
int Sensor3;
int Sensor4;
int Sensor5;
int Sensor6;
int Sensor7;
int Sensor8;
int Sensor9;
int Sensor10;
int Sensor11;
int Sensor12;
int ValInc = 0; // SENSOR DE INCLINAÇÃO
int verdeesq = 0;
int verdedir = 0;
int prataesq = 0;
int pratadir = 0;
int cl, cs;
int verdearena;
int vermelhoarena;
int einstein;

int curie = 0;
int tesla = 0;
int tempo = 0;
int veloplano = 100;
int veloincli = 170;
int velocurva = 120;
int velocurvare = 140;
int veloverde = 160;
```

```
int veloturbo = 160;
int arenaplano = 135;
int arenacurva = 160;


int bola;
int avermelhada = 0;
int passando = 0;
int arena;
int arrancando = 100;
int arranquei = 0;
int inclinado;
int inclinacao = 0;
int plano = 0;
int ajustado = 0;
int esquerdal = 0;
int direital = 0;
int comando = 0;
int auxilioesq = 0;
int auxiliodir = 0;
int vacuo = 0;
int vacuocurvaesq = 15;
int vacuocurvadir = 15;
int bohr = 0;


void declive();
void reto();          //0
void esquerda();      //1
void arrancadaesq();  //2-0
void buscaesq();      //2-1
//void curvaesq();    //3
void ageitaesq();     //4
```

```
void direita();        //5
void arrancadadir();   //6-0
void buscadir();       //6-1
//void curvadir();     //7
void ageitadir();      //8
//void obstaculoesq    //9
//void obstaculodir    //10
//void obstaculo       //11
void finalobs();       //12
//void pequenoavanço   //13
void retorno();        //14
void verdesq();        //15
//void giraesq();      //16
void verdir();         //17
//void giradir();      //18
//void verde();        //19
void chegada();        //20
void resgate();        //21
void grandeajuste();   //22
//void pequeno ajuste  //23
void verificaarena (); //24
void arenavermelha();  //25
//void joga;           //26
void newton();         //27
void centro();         //28
void sonar();          //29
void pegar();          //30
void retornoarena();   //31
void leituraarena();
void leitura();


void setup() {
 Serial.begin(9600);
```

```
Wire.begin();
mux.begin(Wire);
mux.closeAll();
pinMode(SensorDir,INPUT);
pinMode(SensorCen,INPUT);
pinMode(SensorEsq,INPUT);
pinMode(Buzzer,OUTPUT);
pinMode(SensorObsEsq,INPUT);
pinMode(SensorObsDir,INPUT);
pinMode(SensorDtcEsq ,INPUT);
pinMode(SensorDtcDir, INPUT);
pinMode(SensorInc, INPUT);
pinMode(Led, OUTPUT);
pinMode(SensorDirF, INPUT);
pinMode(SensorEsqF, INPUT);
pinMode(LedDir, OUTPUT);
pinMode(LedEsq, OUTPUT);
pinMode(LedCen, OUTPUT);

// DEFINIÇÃO DOS SERVOS
Servo1.attach(9); //braço
Servo2.attach(10);//garra
Servo1.write(180);
Servo2.write(0);
// VELOCIDADE DOS MOTORES
motor1.setSpeed(190);
motor2.setSpeed(190);
// MOTORES INICIAM PARADOS
motor1.run(RELEASE);
motor2.run(RELEASE);
delay(500);
// RELEASE = PARADO
// FORWARD = FRENTE
// BACKWARD = TRAS
```

```
 digitalWrite(Buzzer, HIGH);
 digitalWrite(Led, HIGH);
 delay(250);
 digitalWrite(Buzzer, LOW);
 digitalWrite(Led, LOW);
 delay(250);
 digitalWrite(Buzzer, HIGH);
 digitalWrite(Led, HIGH);
 delay(250);
 digitalWrite(Buzzer, LOW);
 digitalWrite(Led, LOW);
 digitalWrite(LedEsq, HIGH);
 digitalWrite(LedDir, HIGH);

}

void loop() {

  /*Serial.print("   verdeesq: "); Serial.print(verdeesq); Serial.print("   verdedir: ");
Serial.print(verdedir);       Serial.print("   comando:   ");    Serial.print(comando);
Serial.print("   vacuocurvaesq: ");  Serial.print(vacuocurvaesq);
  Serial.println();*/
  Serial.print("   latdir: "); Serial.print(Sensor8); Serial.print("   comando: ");
Serial.print(comando); Serial.print("   arenaverde: "); Serial.print(verdearena);
Serial.print("   arenavermelha: "); Serial.print(vermelhoarena);
  Serial.println();
  // 1 = PRETO
  // 0 = BRANCO
 einstein = ValCm.measureDistanceCm(); //Exemplo de uso HCSR04
 Sensor1 = digitalRead(SensorEsqF);
 Sensor2 = digitalRead(SensorEsq);
 Sensor3 = digitalRead(SensorCen);
 Sensor4 = digitalRead(SensorDir);
 Sensor5 = digitalRead(SensorDirF);
```

```
 Sensor6 = digitalRead(SensorObsEsq);
 Sensor7 = digitalRead(SensorObsDir);
 Sensor9 = digitalRead(ObsDirLat);
 Sensor8 = digitalRead(ObsEsqLat);
 Sensor10 = digitalRead(SensorDtcEsq);
 Sensor11 = digitalRead(SensorDtcDir);
 Sensor12 = digitalRead(SensorInc);




 if(Sensor12 == 0){
       inclinacao == inclinacao++;
       tesla = 0;
  }else{
 if((Sensor12 == 1) && (tesla == 0)){
       tesla = 1;
       inclinacao = 10;
       inclinacao == inclinacao--;
  }else{
      if((Sensor12 == 1) && (tesla == 1)){
        inclinacao == inclinacao--;
 }}
 }
if(inclinacao > 10){
  plano = 0;
  inclinado = 1;
}else{
if(inclinacao < 5){
  plano = 1;
  inclinado = 0;
```

```
}
}
```

```
if(comando == 0){
leitura();
}
if(comando == 15){
 leitura();
}
if(comando == 17){
 leitura();
}
if(inclinado == 1){
 leitura();
}
```

```
if((verdeesq == 1) && (verdedir == 1) && (Sensor2 == 1) && (Sensor4 == 1)){
  comando = 19;
}else{
 if((verdeesq == 1) && (Sensor2 == 1)){
   digitalWrite(Buzzer, HIGH);
  comando = 15;
}else{
 if((Sensor4 == 1) && (verdedir == 1)){
   digitalWrite(Buzzer, HIGH);
  comando = 17;
 }}}
```

```
if((inclinado == 1) && (curie == 0)){
  declive();
}else{



 if(comando == 0){
   vacuocurvaesq = 20;
   vacuocurvadir = 20;
   reto();
 }else{
  if(comando == 1){
    esquerdal = 1;
    direital = 0;
    vacuocurvaesq = 20;
    esquerda();
 }else{
  if((comando == 2) && (arranquei == 0)){
    //auxilioesq == auxilioesq++;
  motor1.setSpeed(veloplano);
  motor2.setSpeed(veloplano);
  arrancando = 15;
  arranquei = 1;
  arrancadaesq();
  }else{
   if((comando == 2) && (arranquei == 1)){

    arrancadaesq();
 }else{
  if((comando == 2) && (arranquei == 2)){
```

```
  arranquei = 0;
  buscaesq();
 }else{
 if(comando == 3){
  //curvaesq();
 motor1.setSpeed(velocurvare);
 motor2.setSpeed(velocurva);
 motor1.run(BACKWARD);
 motor2.run(FORWARD);
 comando = 4;
}else{
 if(comando == 4){
  esquerdal = 1;
  direital = 0;
  auxilioesq = 1;
 motor1.setSpeed(velocurvare);
 motor2.setSpeed(velocurva);
  ageitaesq();
}else{
 if(comando == 5){
  esquerdal = 0;
  direital = 1;
  vacuocurvadir = 20;
  direita();
}else{
 if((comando == 6) && (arranquei == 0)){
  //auxiliodir == auxiliodir++;
 motor1.setSpeed(veloplano);
 motor2.setSpeed(veloplano);
 arrancando = 15;
 arranquei = 1;
 arrancadadir();
}else{
 if((comando == 6) && (arranquei == 1)){
```

```
 arrancadadir();
 }
 if((comando == 6) && (arranquei == 2)) {
   arranquei = 0;
   buscadir();
}else{
 if(comando == 7){
   //curvadir();
 motor1.setSpeed(velocurva);
 motor2.setSpeed(velocurvare);
 motor1.run(FORWARD);
 motor2.run(BACKWARD);
 comando = 8;
}else{
 if(comando == 8){

 motor1.setSpeed(velocurva);
 motor2.setSpeed(velocurvare);
   esquerdal = 0;
   direital = 1;
   auxiliodir = 1;

   ageitadir();
}else{
 if(comando == 9){
   //obstaculoesq();
 motor1.setSpeed(veloturbo);
 motor2.setSpeed(veloturbo);
 motor1.run(RELEASE);
 motor2.run(RELEASE);
 delay(400);
 motor1.run(BACKWARD);
 motor2.run(FORWARD);
 delay(200);
```

```
 motor1.run(RELEASE);
 motor2.run(RELEASE);
 delay(70);
 comando = 0;

}else{
 if(comando == 10){
   //obstaculoesq();
 motor1.setSpeed(veloturbo);
 motor2.setSpeed(veloturbo);
 motor1.run(RELEASE);
 motor2.run(RELEASE);
 delay(400);
 motor1.run(FORWARD);
 motor2.run(BACKWARD);
 delay(200);
 motor1.run(RELEASE);
 motor2.run(RELEASE);
 delay(70);
 comando = 0;
}else{
 if(comando == 11){
 motor1.setSpeed(veloturbo);
 motor2.setSpeed(veloturbo);

 motor1.run(BACKWARD);
 motor2.run(BACKWARD);
 delay(300);

 motor1.run(RELEASE);
 motor2.run(RELEASE);
 delay(1000);

 motor1.run(FORWARD);
```

```
motor2.run(FORWARD);
delay(300);

motor1.run(RELEASE);
motor2.run(RELEASE);
delay(1000);

motor1.run(BACKWARD);
motor2.run(FORWARD);
delay(900);

motor1.run(RELEASE);
motor2.run(RELEASE);
delay(1000);

motor1.run(FORWARD);
motor2.run(FORWARD);
delay(1000);

motor1.run(RELEASE);
motor2.run(RELEASE);
delay(1000);

motor1.run(FORWARD);
motor2.run(BACKWARD);
delay(1100);

motor1.run(RELEASE);
motor2.run(RELEASE);
delay(1000);

motor1.run(FORWARD);
motor2.run(FORWARD);
delay(1400);
```

```
 motor1.run(RELEASE);
 motor2.run(RELEASE);
 delay(1000);

 motor1.run(FORWARD);
 motor2.run(BACKWARD);
 delay(1200);

 motor1.run(RELEASE);
 motor2.run(RELEASE);
 delay(1000);

 motor1.setSpeed(veloplano);
 motor2.setSpeed(veloplano);
 comando = 12;
 }else{
  if(comando == 12){
   finalobs();
  }else{
  if(comando == 13){
   //pequenoavano
 motor1.run(FORWARD);
 motor2.run(FORWARD);
 delay(600);
 motor1.run(RELEASE);
 motor2.run(RELEASE);
 delay(500);

comando = 14;
}else{
 if(comando == 14){
    retorno();
```

```
}else{
 if((comando == 15) && (arranquei == 0)){
   arrancando = 15;
   verdesq();
}else{
 if((comando == 15) && (arranquei == 1)){
   verdeesq = 0;
   verdesq();
}else{
 if(comando == 16){
      //void giraesq();

 motor1.setSpeed(velocurvare);
 motor2.setSpeed(veloverde);
 motor1.run(BACKWARD);
 motor2.run(FORWARD);
 delay(500);
 vacuocurvaesq = 15;
 comando = 4;
 digitalWrite(Buzzer, LOW);
}else{
 if((comando == 17) && (arranquei == 0)){
   arrancando = 15;
   verdir();
}else{
if((comando == 17) && (arranquei == 1)){
  verdedir = 0;
   verdir();
}else{
 if(comando == 18){
   //void giradir();

 motor1.setSpeed(veloverde);
 motor2.setSpeed(velocurvare);
```

```
  motor1.run(FORWARD);
  motor2.run(BACKWARD);
  delay(500);
  vacuocurvadir = 15;
  comando = 8;
  digitalWrite(Buzzer, LOW);
 }else{
  if(comando == 19){
verdeesq = 0;
verdedir = 0;
    //void verde;
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  delay(2000);
  motor1.setSpeed(veloverde);
  motor2.setSpeed(veloverde);
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  delay(600);
  motor1.run(BACKWARD);
  motor2.run(FORWARD);
  delay(900);
  motor1.run(BACKWARD);
  motor2.run(BACKWARD);
  delay(200);
  motor1.run(BACKWARD);
  motor2.run(FORWARD);
  delay(700);
  digitalWrite(Buzzer,LOW);
  comando = 27;
 }else{
   if((comando == 20) && (plano == 1)){
    chegada();
}else{
```

```
  if(comando == 21){
   resgate();
}else{
  if(comando == 22){
   grandeajuste();
}else{
  if(comando == 23){
    motor1.run(FORWARD);
    motor2.run(FORWARD);
    delay(500);
    comando = 24;
}else{
  if(comando == 24){
    verificaarena();
}else{
  if(comando == 25){
    arenavermelha();
}else{
  if(comando == 26){
   motor1.run(BACKWARD); //PEQUENA RÉ
   motor2.run(BACKWARD);
   delay(200);
   motor1.run(RELEASE); //PEQUENA RÉ
   motor2.run(RELEASE);
   digitalWrite(LedDir, LOW);
   digitalWrite(LedEsq, LOW);
   Servo1.slowmove(40,80);
   delay(3000);
   Servo2.write(180);
   delay(3000);
   Servo2.write(0);
   delay(2000);
   Servo1.slowmove(180,185);
   delay(6000);
```

```
   motor1.run(RELEASE);
   motor2.run(RELEASE);
   delay(200);
   motor1.run(BACKWARD); //PEQUENA RÉ
   motor2.run(BACKWARD);
   delay(400);
   motor1.run(RELEASE);
   motor2.run(RELEASE);
   delay(200);
   digitalWrite(Buzzer, HIGH); //CHARME PARA DIZER QUE TERMINOU
   digitalWrite(Led, HIGH);
   delay(250);
   digitalWrite(Buzzer, LOW);
   digitalWrite(Led, LOW);
   delay(250);
   digitalWrite(Buzzer, HIGH);
   digitalWrite(Led, HIGH);
   delay(250);
   digitalWrite(Buzzer, LOW);
   digitalWrite(Led, LOW);
   motor1.run(RELEASE);
   motor2.run(RELEASE);
   comando = 28;
   passando = 0;        //AQUIIIIIIIIIIIIIII
}else{
  if(comando == 27){
    newton();
}else{
  if(comando == 28){
    centro();
}else{
  if(comando == 29){
    sonar();
}else{
```

```
  if(comando == 30){
    pegar();
}else{
  if(comando == 31){
    retornoarena();
  }
}
}
}
}
}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}
}
```

```
void declive(){
  motor1.run(FORWARD);
  motor2.run(FORWARD);
 if(plano == 1){
 comando = 0;
 }
```

```
if((prataesq == 1) && (pratadir == 1)){
  curie = 1;
  comando = 20;
}else{
```

```
if(prataesq == 1){
  curie = 1;
  comando = 20;
}else{
if(pratadir == 1){
  curie = 1;
  comando = 20;
}}
}


   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){     // principal
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloincli);


  }else{


    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){     //tudo branco
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloincli);
  }else{
    if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){     //tudo preto
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloincli);
  }else{
    if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //3 centrais
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloincli);
```

```
  }else{



   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0) && (cl >9)){    // extrema esquerda e centro
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloincli);
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    //centro esquerda
    motor1.setSpeed(veloplano);
    motor2.setSpeed(veloincli);
  }else{
    if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //centro e extremo esquerda
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloincli);
  }else{
    if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda e extremo esquerda
    motor1.setSpeed(veloplano);
    motor2.setSpeed(veloincli);
  }else{
    if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda
    motor1.setSpeed(veloplano);
    motor2.setSpeed(veloincli);
  }else{
    if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //extremo esquerda
    motor1.setSpeed(veloplano);
```

```
    motor2.setSpeed(veloincli);
  }else{




   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1) && (cs > 9)){    // extrema direita e centro
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloincli);
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0) && (cs > 9)){    //centro direita
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloplano);
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //centro e extremo direita
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloincli);
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //direita e extremo direita
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloplano);
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //direita
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloplano);
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //extremo direita
```

```
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloplano);
  }else{




   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //incomum
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloincli);
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //incomum
    motor1.setSpeed(veloincli);
    motor2.setSpeed(veloincli);
  }
  }}}}}}}}}}}}}}}}}
  }




void leitura(){



 mux.openChannel(7);
 //esq.setInterrupt(false);
 uint16_t r1, g1, b1, c1, lux1, ct1;
 esq.getRawData(&r1, &g1, &b1, &c1);
 lux1 = esq.calculateLux(r1, g1, b1);
 ct1 = esq.calculateColorTemperature_dn40(r1, g1, b1, c1);
```

```
 cl = c1;

 if((r1 > 0) && (r1 < 5) && (g1 > 1) && (g1 < 6) && (b1 > 1) && (b1 < 6) && (c1 >
2) && (c1< 6) && (plano == 1)){ //verde direita
    verdeesq = 1;
    prataesq = 0;
 }else{
 if((r1 > 7) && (r1 < 12) && (g1 > 7) && (g1 < 14) && (b1 > 6) && (b1 < 13) &&
(c1 > 18) && (c1 < 23) && (inclinado == 1)){ //prata direita
    verdeesq = 0;
    prataesq = 1;
 }else{
  if(c1 > 12){
  verdeesq = 0;
  prataesq = 0;
 }
 }}

 //esq.setInterrupt(true);


mux.closeChannel(7);
mux.openChannel(5);
//dir.setInterrupt(false);
uint16_t r2, g2, b2, c2, lux2, ct2;
 dir.getRawData(&r2, &g2, &b2, &c2);
 lux2 = dir.calculateLux(r2, g2, b2);
 ct2 = dir.calculateColorTemperature_dn40(r2, g2, b2, c2);
 cs = c2;
 if((r2 > 1) && (r2 < 5) && (g2 > 1) && (g2 < 5) && (b2 > 1) && (b2 < 5) && (c2 >
2) && (c2 < 6) && (plano == 1)){ //verde esquerda
    verdedir = 1;
    pratadir = 0;
 }else{
```

```
 if((r2 > 7) && (r2 < 12) && (g2 > 7) && (g2 < 14) && (b2 > 6) && (b2 < 11) &&
(c2 > 17) && (c2 < 23) && (inclinado == 1)){ //prata esquerda
    verdedir = 0;
    pratadir = 1;
 }else{

  if(c2 > 12){
  verdedir = 0;
 pratadir = 0;
 }

}
 }
  /*Serial.print("  - r: "); Serial.print(r1);
  Serial.print("  - g: "); Serial.print(g1);
  Serial.print("  - b: "); Serial.print(b1);
  Serial.print("  - c: "); Serial.print(c1);
  Serial.print("   prataesq: "); Serial.print(prataesq); Serial.print("   pratadir: ");
Serial.print(pratadir);       Serial.print("   comando:   ");    Serial.print(comando);
Serial.print("   arrancando: ");  Serial.print(arrancando);
  Serial.println();*/
 //dir.setInterrupt(true);
mux.closeChannel(5);

}




 void reto(){


  motor1.setSpeed(veloplano);
  motor2.setSpeed(veloplano);
```

```
  motor1.run(FORWARD);
  motor2.run(FORWARD);

 if((vacuo == 25) && (esquerdal == 1) && (Sensor12 == 1)){
  comando = 4;
  vacuo = 0;
}else{
 if((vacuo == 25) && (direital == 1) && (Sensor12 == 1)){
  comando = 8;
  vacuo = 0;
 }}


  if((verdeesq == 1) && (verdedir == 1) && (Sensor2 == 1) && (Sensor4 == 1)){
   comando = 19;
 }else{
  if((verdeesq == 1) && (Sensor2 == 1)){
   comando = 15;
 }else{
  if((Sensor4 == 1) && (verdedir == 1)){
   comando = 17;
  }}}



/*if(auxilioesq == 1){
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  delay(500);
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  delay(200);
  motor1.run(BACKWARD);
  motor2.run(FORWARD);
```

```
  delay(500);
  motor1.run(BACKWARD);
  motor2.run(BACKWARD);
  delay(300);
  auxilioesq == auxilioesq--;
}else{
if(auxiliodir == 1){
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  delay(500);
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  delay(200);
  motor1.run(FORWARD);
  motor2.run(BACKWARD);
  delay(500);
  motor1.run(BACKWARD);
  motor2.run(BACKWARD);
  delay(300);
  auxiliodir == auxiliodir--;
}*/

if((Sensor6 == 0) && (Sensor7 == 0)){
    comando = 11;
  }else{
    if((Sensor6 == 0) && (Sensor7 == 1)){
    comando = 9;
  }else{
    if((Sensor6 == 1) && (Sensor7 == 0)){
    comando = 10;
  }else{
```

```
  if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){     // principal
    comando == comando;
    vacuo = 0;

  }else{




    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){     //tudo branco
    comando == comando;
    vacuo == vacuo++;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){     //tudo preto
    comando == comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //3 centrais
    comando == comando;
  }else{




  if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0) && (cl >9)){    // extrema esquerda e centro
    comando = 1;
  }else{
```

```
  if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)  && (cl >9)){    //centro esquerda
   comando = 1;
 }else{
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //centro e extremo esquerda
   comando = 1;
 }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda e extremo esquerda
   comando = 1;
 }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda
   comando = 1;
 }else{
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //extremo esquerda
   comando = 1;
 }else{
```

```
  if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1) && (cs > 9)){    // extrema direita e centro
   comando = 5;
 }else{
  if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0) && (cs > 9)){    //centro direita
```

```
    comando = 5;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //centro e extremo direita
    comando = 5;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //direita e extremo direita
    comando = 5;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //direita
    comando = 5;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //extremo direita
    comando = 5;
  }else{




    if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //incomum
    comando = 6;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //incomum
    comando = 1;
  }
```

```
}}}}}}}}}}}}}}}}}}}}
}
```

```
void esquerda(){
motor1.setSpeed(velocurva);
motor2.setSpeed(velocurva);
motor1.run(BACKWARD);
motor2.run(FORWARD);
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){     // principal
   comando = 0;
 }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){     //tudo branco
   comando = 0;
 }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){     //tudo preto
   comando = 2;
 }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //3 centrais
   comando = 2;
  }else{
```

```
  if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    // extrema esquerda e centro
   comando = 2;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    //centro esquerda
   comando == comando;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //centro e extremo esquerda
   comando = 2;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda e extremo esquerda
   comando = 2;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda
   comando == comando;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //extremo esquerda
   comando == comando;
  }else{
```

```
  if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){    // extrema direita e centro
    comando = 6;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //centro direita
    comando = comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //centro e extremo direita
    comando = 2;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //direita e extremo direita
    comando = 2;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //direita
    comando = 2;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //extremo direita
    comando = 2;
  }else{
```

```
  if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //incomum
   comando = 5;
 }else{
  if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //incomum
   comando = 1;
 }
 }}}}}}}}}}}}}}}}}
 }
```

```
void arrancadaesq(){
motor1.run(FORWARD);
motor2.run(FORWARD);
arrancando == arrancando--;
arranquei = 1;
if(arrancando == 0){
  arranquei = 2;
}else{
  if((verdeesq == 1) && (verdedir == 1) && (Sensor2 == 1) && (Sensor4 == 1)){
  comando = 19;
```

```
  }else{
  if((verdeesq == 1) && (Sensor2 == 1)){
  comando = 15;
}else{
  if((verdedir == 1) && (Sensor4 == 1)){
  comando = 17;
  }
  }}}
  }
```

```
void buscaesq(){

   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){     // principal
    comando = 0;
    //auxilioesq == auxilioesq--;
  }else{
```

```
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){     //tudo branco
    comando = 4;
  }else{
```

```
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){     //tudo preto
   comando = 4;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //3 centrais
   comando = 0;
  }else{




   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    // extrema esquerda e centro
   comando = 4;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    //centro esquerda
   comando = 4;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //centro e extremo esquerda
   comando = 4;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda e extremo esquerda
   comando = 4;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda
   comando = 4;
  }else{
```

```
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //extremo esquerda
   comando = 4;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){    // extrema direita e centro
   comando = 8;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //centro direita
   comando = 8;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //centro e extremo direita
   comando = 8;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //direita e extremo direita
   comando = 8;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //direita
   comando = 8;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //extremo direita
```

```
    comando = 8;
  }else{

   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //incomum
    comando == comando;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //incomum
    comando == comando;
  }
  }}}}}}}}}}}}}}}}}
  }



void ageitaesq(){

  motor1.run(BACKWARD);
  motor2.run(FORWARD);

  if(vacuocurvaesq == 0){
    comando = 0;
    vacuocurvaesq = 20;
  }else{
```

```
  if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){     // principal
   comando = 0;
 }else{

   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){     //tudo branco
   comando == comando;
   vacuocurvaesq == vacuocurvaesq--;
 }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){     //tudo preto
   comando == comando;
 }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //3 centrais
   comando = 0;
 }else{

  if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    // extrema esquerda e centro
   comando == comando;
 }else{
  if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    //centro esquerda
   comando == comando;
```

```
  }else{
    if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //centro e extremo esquerda
    comando == comando;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda e extremo esquerda
    comando == comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda
    comando == comando;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //extremo esquerda
    comando == comando;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){    // extrema direita e centro
    comando = 0;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //centro direita
    comando = 0;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //centro e extremo direita
   comando = 0;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //direita e extremo direita
   comando = 0;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //direita
   comando = 0;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //extremo direita
   comando = 0;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //incomum
   comando = 0;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //incomum
   comando == comando;
  }
  }}}}}}}}}}}}}}}}}}
  }
```

```
void direita(){
  motor1.setSpeed(velocurva);
  motor2.setSpeed(velocurva);
  motor1.run(FORWARD);
  motor2.run(BACKWARD);
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){     // principal
    comando = 0;
  }else{
```

```
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){     //tudo branco
    comando == 0;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){     //tudo preto
    comando = 6;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //3 centrais
   comando = 6;
  }else{
```

```
  if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    // extrema esquerda e centro
   comando = 2;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    //centro esquerda
   comando == comando;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //centro e extremo esquerda
   comando = 6;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda e extremo esquerda
   comando = 6;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda
   comando = 6;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //extremo esquerda
   comando = 6;
  }else{
```

```
  if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){    // extrema direita e centro
    comando = 6;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //centro direita
    comando == comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //centro e extremo direita
    comando == comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //direita e extremo direita
    comando == comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //direita
    comando == comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //extremo direita
    comando == comando;
  }else{
```

```
  if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //incomum
   comando = 5;
 }else{
  if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //incomum
   comando = 1;
 }
 }}}}}}}}}}}}}}}}}
 }
```

```
void arrancadadir(){
motor1.run(FORWARD);
motor2.run(FORWARD);
delay(tempo);
arrancando == arrancando--;
arranquei = 1;
if(arrancando == 0){
 arranquei = 2;
}else{
 if((verdeesq == 1) && (verdedir == 1) && (Sensor2 == 1) && (Sensor4 == 1)){
 comando = 19;
}else{
 if((verdeesq == 1) && (Sensor2 == 1)){
 comando = 15;
```

```
}else{
  if((verdedir == 1) && (Sensor4 == 1)){
  comando = 17;
  }
  }}}
  }




void buscadir(){


   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){     // principal
    comando = 0;
    //auxiliodir == auxiliodir++;
  }else{




    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){     //tudo branco
    comando = 8;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){     //tudo preto
    comando = 8;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //3 centrais
    comando == comando;
```

```
  }else{


   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    // extrema esquerda e centro
    comando = 2;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    //centro esquerda
    comando = 2;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //centro e extremo esquerda
    comando = 2;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda e extremo esquerda
    comando = 2;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda
    comando = 2;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //extremo esquerda
    comando = 2;
  }else{
```

```
  if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){    // extrema direita e centro
   comando = 8;
 }else{
  if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //centro direita
   comando = 8;
 }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //centro e extremo direita
   comando = 8;
 }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //direita e extremo direita
   comando = 8;
 }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //direita
   comando = 8;
 }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //extremo direita
   comando = 8;
 }else{



  if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //incomum
   comando == comando;
```

```
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //incomum
    comando == comando;
  }
  }}}}}}}}}}}}}}}}}
  }


void ageitadir(){
  motor1.run(FORWARD);
  motor2.run(BACKWARD);


    if(vacuocurvadir == 0){
    comando = 0;
    vacuocurvadir = 20;
  }else{




   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){     // principal
    comando = 0;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){     //tudo branco

   vacuocurvadir == vacuocurvadir--;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){     //tudo preto
   comando = 0;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //3 centrais
   comando = 0;
  }else{




  if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    // extrema esquerda e centro
   comando = 0;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    //centro esquerda
   comando = 0;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //centro e extremo esquerda
   comando = 0;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda e extremo esquerda
   comando = 0;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda
   comando = 0;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //extremo esquerda
   comando = 0;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){    // extrema direita e centro
   comando == comando;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //centro direita
   comando == comando;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //centro e extremo direita
   comando == comando;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //direita e extremo direita
   comando == comando;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //direita
```

```
    comando == comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //extremo direita
    comando == comando;
  }else{




   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //incomum
    comando == comando;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //incomum
    comando = 0;
  }
  }}}}}}}}}}}}}}}}}}
  }




 void finalobs(){
  motor1.run(FORWARD);
  motor2.run(FORWARD);
delay(tempo);
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){
   comando == comando;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){
   comando == comando;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){
   comando = 13;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){
   comando = 13;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){
   comando = 13;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){
   comando = 13;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){
   comando = 13;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){
   comando = 13;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){
```

```
    comando = 13;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){
    comando == comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){
    comando == comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 1)){
    comando == comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 1)){
    comando == comando;
  }
  }}}}}}}}}}}
  }
```

```
void retorno(){
  motor1.setSpeed(140);
  motor2.setSpeed(140);
  motor1.run(BACKWARD);
  motor2.run(FORWARD);
  delay(tempo);
```

```
  if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){     // principal
   comando = 0;
 }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){     //tudo branco
   comando == comando;
 }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){     //tudo preto
   comando = 0;
 }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //3 centrais
   comando = 0;
 }else{
```

```
  if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    // extrema esquerda e centro
   comando == comando;
```

```
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    //centro esquerda
    comando == comando;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //centro e extremo esquerda
    comando == comando;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda e extremo esquerda
    comando == comando;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda
    comando == comando;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //extremo esquerda
    comando == comando;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){    // extrema direita e centro
    comando = 0;
  }else{
```

```
  if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //centro direita
   comando = 0;
 }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //centro e extremo direita
   comando = 0;
 }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //direita e extremo direita
   comando = 0;
 }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //direita
   comando = 0;
 }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //extremo direita
   comando = 0;
 }else{
```

```
  if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //incomum
   comando = 0;
 }else{
  if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //incomum
```

```
    comando == comando;
  }
  }}}}}}}}}}}}}}}}}
  }
```

```
void verdesq(){
  motor1.setSpeed(veloplano);
  motor2.setSpeed(veloplano);
motor1.run(FORWARD);
motor2.run(FORWARD);
arrancando == arrancando--;
arranquei = 1;
if(arrancando == 0 ){
  comando == comando++;
  arranquei = 0;
}else{
  if(verdedir == 1){
    comando = 19;
    arranquei = 0;
  }}
  }
```

```
void verdir(){
  motor1.setSpeed(veloplano);
  motor2.setSpeed(veloplano);
motor1.run(FORWARD);
motor2.run(FORWARD);
```

```
arrancando == arrancando--;
arranquei = 1;
if(arrancando == 0 ){
  comando == comando++;
  arranquei = 0;
}else{
  if(verdeesq == 1){
    comando = 19;
  }}
  }




void chegada(){
 motor1.run(RELEASE);
 motor2.run(RELEASE);
delay(5000);
comando == comando++;
}


void resgate(){
    motor1.setSpeed(arenaplano);
    motor2.setSpeed(arenaplano);

/*if((Sensor6 == 1) && (Sensor7 == 0) && (ajustado == 2)){
    comando = 22;
    ajustado = 0;
}*/

  if((Sensor6 == 1) && (Sensor7 == 1)){
    motor1.run(FORWARD);
```

```
    motor2.run(FORWARD);
    bohr = 0;
  }else{
   if((Sensor6 == 0) && (Sensor7 == 1)){
    motor1.run(FORWARD);
    motor2.run(FORWARD);
    bohr == bohr++;
    auxilioesq = 0;
    auxiliodir = 1;

  }else{
   if((Sensor6 == 1) && (Sensor7 == 0) /*&& (ajustado == 0)*/){
    motor1.run(FORWARD);
    motor2.run(FORWARD);
    bohr == bohr++;
    auxilioesq = 1;
    auxiliodir = 0;
    comando = 22;
    //ajustado = 2;
  }
  }}

  if((auxilioesq == 1) && (bohr > 3)){
   comando = 22;
   auxilioesq = 0;
   bohr = 0;
  }

   if((Sensor6 == 0) && (Sensor7 == 0)){
    motor1.run(BACKWARD);
    motor2.run(FORWARD);
    delay(900);
  }
 }
```

```
void grandeajuste(){

if((Sensor6 == 1) && (passando == 2)){
  passando = 1;
  bohr = 0;
}

if((Sensor7 == 1) && (passando == 1)){
 passando = 2;
  bohr = 0;

}

  if((bohr > 25) && (passando == 1)){
   motor1.run(FORWARD);
   motor2.run(FORWARD);
   delay(50);
   passando = 2;
   bohr = 0;
 }else{
  if((bohr > 25) && (passando == 2)){
   motor1.run(FORWARD);
   motor2.run(FORWARD);
   delay(50);
   passando = 1;
   bohr = 0;
  }}

  if(passando == 0){
```

```
    motor1.setSpeed(arenacurva);
    motor2.setSpeed(arenacurva);
    motor1.run(BACKWARD);
  motor2.run(BACKWARD);
  delay(250);
  motor1.run(BACKWARD);
  motor2.run(FORWARD);
  delay(500);
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  delay(500);
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  delay(700);
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  delay(500);
  motor1.run(FORWARD);
  motor2.run(BACKWARD);
  delay(500);
  passando = 1;
}else{

  if(passando == 1){
    motor1.setSpeed(arenacurva);
    motor2.setSpeed(arenacurva);
  motor1.run(FORWARD);
  motor2.run(BACKWARD);
  bohr == bohr++;
 }


if((Sensor6 == 1) && (Sensor7 == 0) && (passando == 1)){
    motor1.setSpeed(arenacurva);
```

```
    motor2.setSpeed(arenacurva);
    motor1.run(FORWARD);
    motor2.run(BACKWARD);
    bohr == bohr++;
  }else{
if((Sensor6 == 0) && (Sensor7 == 0) && (passando == 1) && (einstein > 4.5)){
    motor1.setSpeed(arenacurva);
    motor2.setSpeed(arenacurva);
    motor1.run(FORWARD);
    motor2.run(BACKWARD);
    bohr == bohr++;
  }else{
if((Sensor6 == 0) && (Sensor7 == 0) && (passando == 1) && (einstein < 4.5)){
   comando = 23;
   ajustado = 0;
   motor1.run(RELEASE);
   motor2.run(RELEASE);
   delay(500);
  }

  if(passando == 2){
  motor1.setSpeed(arenacurva);
  motor2.setSpeed(arenacurva);
  motor1.run(BACKWARD);
  motor2.run(FORWARD);
  bohr == bohr++;
 }

if((Sensor6 == 0) && (Sensor7 == 1) && (passando == 2)){
    motor1.setSpeed(arenacurva);
    motor2.setSpeed(arenacurva);
    motor1.run(BACKWARD);
    motor2.run(FORWARD);
    bohr == bohr++;
```

```
  }else{
if((Sensor6 == 0) && (Sensor7 == 0) && (passando == 2) && (einstein > 4.5)){
    motor1.setSpeed(arenacurva);
    motor2.setSpeed(arenacurva);
    motor1.run(BACKWARD);
    motor2.run(FORWARD);
    bohr == bohr++;
  }else{
if((Sensor6 == 0) && (Sensor7 == 0) && (passando == 2) && (einstein < 4.5)){
   comando = 23;
   ajustado = 0;
   motor1.run(RELEASE);
   motor2.run(RELEASE);
   delay(500);
 }
 }}}}}
 }



void verificaarena(){

  motor1.run(RELEASE);
  motor2.run(RELEASE);
  delay(2000);
  if(Sensor12 == 0){
  motor1.run(BACKWARD);
  motor2.run(BACKWARD);
  delay(100);
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  }
```

```
mux.openChannel(6);
 //esq.setInterrupt(false);
 uint16_t r1, g1, b1, c1, lux1, ct1;
 cen.getRawData(&r1, &g1, &b1, &c1);
 lux1 = cen.calculateLux(r1, g1, b1);
 ct1 = cen.calculateColorTemperature_dn40(r1, g1, b1, c1);


 if((g1 > b1) && (b1 > r1) && (r1 < 900) && (r1 > 650)&& ( g1 < 1500)){ //verde
direita
    verdearena = 1;
    vermelhoarena = 0;
 }else{
 if((r1 > b1) && (b1 > g1) && (c1 < 2600)){ //prata direita
    verdearena = 0;
    vermelhoarena = 1;
 }else{
  verdearena = 0;
  vermelhoarena = 0;
 }
 }

 //esq.setInterrupt(true);


mux.closeChannel(6);
  if(verdearena == 1){
    comando = 26;
 }else{
  if(vermelhoarena == 1){
  comando = 25;

}
```

```
}
}
```

```
void arenavermelha(){

if((Sensor8 == 1) && (avermelhada == 0)){
  motor1.setSpeed(arenacurva);
motor2.setSpeed(arenacurva);
motor1.run(BACKWARD);
motor2.run(FORWARD);
passando = 0;
}else{
if((Sensor8 == 0) && (avermelhada == 0)){
motor1.setSpeed(arenacurva);
motor2.setSpeed(arenacurva);
motor1.run(BACKWARD);
motor2.run(FORWARD);
avermelhada = 1;
}else{
if((Sensor8 == 0) && (avermelhada == 1)){
motor1.setSpeed(arenacurva);
motor2.setSpeed(arenaplano);
motor1.run(BACKWARD);
motor2.run(FORWARD);
}else{
if((Sensor8 == 1) && (avermelhada == 1)){
motor1.setSpeed(arenaplano);
```

```
motor2.setSpeed(arenaplano);
motor1.run(FORWARD);
motor2.run(FORWARD);
avermelhada = 2;
}else{
  if((Sensor8 == 1) && (avermelhada == 2) && (Sensor7 == 1)){
motor1.setSpeed(arenaplano);
motor2.setSpeed(arenaplano);
motor1.run(FORWARD);
motor2.run(FORWARD);
}else{
  if((Sensor8 == 0) && (avermelhada == 2) && (Sensor7 == 1)){
motor1.setSpeed(arenaplano);
motor2.setSpeed(arenaplano);
motor1.run(FORWARD);
motor2.run(FORWARD);
 }else{
if((Sensor7 == 0) && (avermelhada == 2)){
motor1.run(BACKWARD);
motor2.run(FORWARD);
avermelhada = 3;
}else{
if((Sensor7 == 0) && (avermelhada == 3)){
motor1.run(BACKWARD);
motor2.run(FORWARD);
}else{
if((Sensor7 == 1) && (avermelhada == 3)){
motor1.run(BACKWARD);
motor2.run(FORWARD);
avermelhada = 4;
bohr = 0;
}else{
if(avermelhada == 4){
motor1.run(BACKWARD);
```

```
motor2.run(FORWARD);
bohr == bohr++;
}
}}}}}}}}}

if((bohr > 2) && (avermelhada == 4)){
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  avermelhada = 0;
  bohr = 0;
  comando = 21;
}}
```

```
void newton(){
motor1.setSpeed(velocurvare);
motor2.setSpeed(velocurva);
motor1.run(BACKWARD);
motor2.run(FORWARD);

if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0) &&
(Sensor5 == 0)){     // principal
    comando = 0;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){     //tudo branco
   comando == comando;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){     //tudo preto
   comando == comando;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //3 centrais
   comando == 0;
  }else{
```

```
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    // extrema esquerda e centro
   comando = 0;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){    //centro esquerda
   comando = 0;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //centro e extremo esquerda
   comando == comando;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda e extremo esquerda
   comando == comando;
```

```
  }else{
    if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //esquerda
    comando == comando;
  }else{
    if((Sensor1 == 1) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 0)){   //extremo esquerda
    comando == comando;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){    // extrema direita e centro
    comando = 0;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){    //centro direita
    comando = 0;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 1) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //centro e extremo direita
    comando = 0;
  }else{
    if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //direita e extremo direita
    comando = 0;
  }else{
```

```
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //direita
    comando = 0;
  }else{
   if((Sensor1 == 0) && (Sensor2 == 0) && (Sensor3 == 0) && (Sensor4 == 0)
&& (Sensor5 == 1)){   //extremo direita
    comando = 0;
  }else{









   if((Sensor1 == 0) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 1)){   //incomum
    comando = 0;
  }else{
   if((Sensor1 == 1) && (Sensor2 == 1) && (Sensor3 == 1) && (Sensor4 == 1)
&& (Sensor5 == 0)){   //incomum
    comando == comando;
  }
  }}}}}}}}}}}}}}}}}
  }




  void centro(){
    motor1.setSpeed(veloplano);
    motor2.setSpeed(veloplano);
  motor1.run(BACKWARD);
```

```
 motor2.run(BACKWARD);

 if((einstein > 34) && (einstein < 36) && (passando == 0)){
 motor1.run(FORWARD);
 motor2.run(BACKWARD);
 delay(100);
   passando = 1;
   bohr = 0;
   comando == comando++;
   cs = 0;
 }
 }


 void sonar(){
 motor1.setSpeed(veloturbo);
 motor2.setSpeed(veloturbo);
 motor1.run(BACKWARD);
 motor2.run(FORWARD);
 delay(50);
 motor1.run(RELEASE);
 motor2.run(RELEASE);
 delay(25);
 bohr == bohr++;
 cs == cs++;
passando = 0;
 if(einstein < 28){
   comando = 30;
   bohr = 0;
 }
 }
```

```
  void pegar(){
    if(passando == 0){
  motor1.setSpeed(veloplano);
  motor2.setSpeed(veloplano);
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  bohr == bohr++;
    }
  if((einstein > 8) && (einstein < 10) && (passando == 0)){
   motor1.run(RELEASE);
   motor2.run(RELEASE);
   Servo1.write(180);
   delay(1000);
   Servo2.write(180);
   delay(1500);
   Servo1.write(0);
   delay(250);
   Servo2.write(0);
   delay(1500);
   passando = 1;
}
if(passando == 1){
  motor1.run(BACKWARD);
  motor2.run(BACKWARD);
  bohr == bohr--;
}
if(bohr == 0){
  passando = 0;
  bohr = 0;
  comando = 31;

}
```

```
bola = 1;
}


void retornoarena(){
  if(passando == 0){
  motor1.setSpeed(veloturbo);
  motor2.setSpeed(veloturbo);
  motor1.run(FORWARD);
  motor2.run(BACKWARD);
  delay(50);
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  delay(25);
  cs == cs--;
  }

  if((passando == 0) && (cs == 0)){
    passando = 1;
  }

  if(passando == 1){
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  }

  if((Sensor6 == 0) && (passando == 1)){
    comando = 22;
    passando = 1;
    bohr = 0;
  }

}
```

4 CONCLUSÃO

O projeto foi longo, tendo sido iniciado em abril de 2022 e finalizado em setembro de 2023. Grande parte da dificuldade se deu por falta de exemplos a se seguir, a equipe desenvolveu um projeto totalmente do zero. Boa parte dos membros não possuíam conhecimentos sobre programação, ou sobre a competição, e tiveram que aprender durante o desenvolvimento do robô, mas o importante é que conseguiram evoluir o projeto a nível de competição estadual.

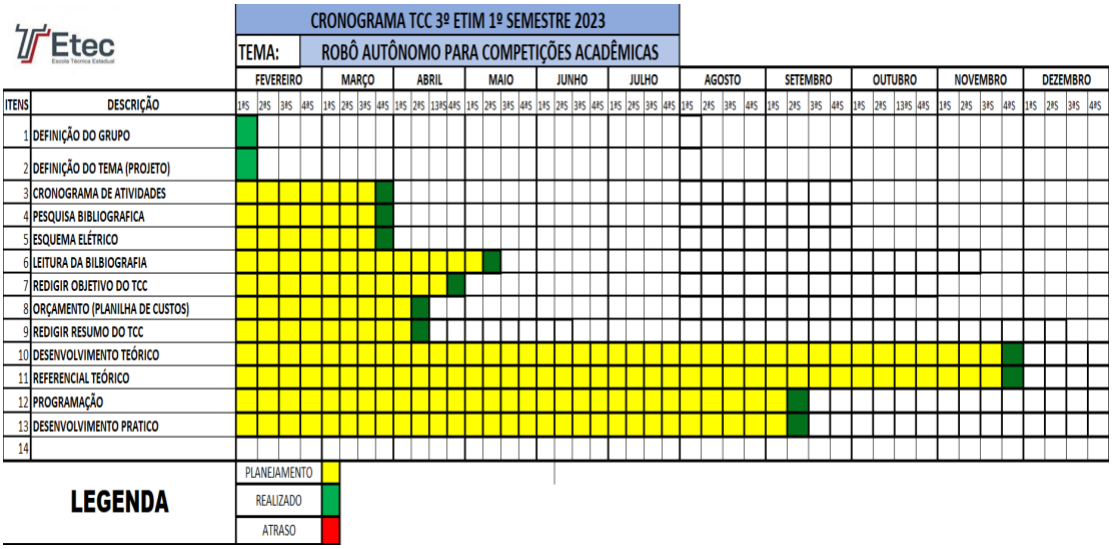
A equipe se classificou para a fase estadual, mas o robô não foi capaz de capturar as vítimas e a performance não foi totalmente eficiente. Foram identificadas mudanças necessárias, como o posicionamento dos sensores, o modelo da garra, a altura do chassi, a alimentação e a lógica desenvolvida para a conclusão da segunda fase da competição (o resgate das vítimas).

Os sensores de cor não eram precisos graças a grande variação de luminosidade e seria muito melhor se fossem posicionados bem no meio do chassi do carrinho, onde não há interferência luminosa. O eixo de rotação do carrinho é na parte traseira e isso prejudicou bastante nas curvas, seria melhor se tanto o eixo quanto os sensores seguidores de linha fossem posicionados mais ao centro do robô. O chassi não era grande o suficiente para colocar equipamentos que facilitassem a captura das vítimas, ou seja, o chassi consegue ser eficaz para a conclusão da primeira fase da competição (seguir a linha e desviar de obstáculos), mas ineficaz para a segunda fase.

Por fim, a conclusão do projeto foi satisfatória. O robô foi capaz de seguir a linha, identificar intersecções, subir a rampa, ultrapassar redutores de velocidades, superar gaps (buraco na linha a ser seguida) e desviar de obstáculos com sucesso. A equipe realizou a montagem e as ligações eletrônicas a partir de seus próprios conhecimentos, recebendo dos professores e da escola apenas o apoio, as aulas provenientes do curso e, quando possível, algum suporte financeiro. A equipe seguiu para a fase estadual e agora serve de exemplo para os próximos competidores em nome da etec, trazendo também acertos, dicas e problemas a serem considerados, cumprindo assim o objetivo do projeto.

Segue abaixo o cronograma e a tabela de custos da equipe.

Figura 93: Cronograma



Fonte: Autoria própria.

Figura 94: Tabela de custos

Componente	Quantidade	Preço
Micro Servo Motor Sg90 9g 1,6kgf	2	R$ 15,09
Botão Switch	2	R$ 6,16
Suporte de 6 pilhas	1	R$ 13,35
Voltímetro Led	1	R$ 12,11
Chave Hh	1	R$ 1,39
Motor DC	2	R$ 9,02
Cooler 5v 25x25mm	1	R$ 17,85
Kit espaçador M3 nylon	1	R$ 52,08
Kit Garra 4 pinças Acrílico	1	R$ 28,96
Kit Jumpers	1	R$ 10,46
Módulo Sensor De Inclinação Sw-520d	1	R$ 11,40
Módulo Sensor Óptico - Segue Faixa - TCRT5000	3	R$ 5,79
Sensor De Obstáculo Reflexivo Infravermelho Ir Lm393 Arduino	2	R$ 9,10
Arduino Mega	1	R$ 95,22
Shield l293d	1	R$ 32,26
Sensor TCS 34725	3	R$ 30,00
Multiplexador I2C	1	R$ 31,00
Total	R$	462,01

Fonte: Autoria própria.

5 REFERÊNCIAS

KERCHBAUMER, Ricardo. Microcontroladores. ed IFSP. IFC – Instituto Federal Catarinense, 2009, 32 p.

ADA, L. Adafruit TCA9548A 1-to-8 I2C Multiplexer Breakout. ed Adafruit Industries. Adafruit, 2021, 16 p.

T. ALTAN, S. OH, H. GEGEL. Processos de conformação dos materiais. Ed USP. USP – Departamento de engenharia de produção, julho de 2022, 132 p.

TCS34725 Color Sensor User Manual, 2019, 16 p.

Izildo Antunes, Marcos A. C. Freire. Elementos de máquina. ed ÉRICA. São Paulo, 2008, 57 p.