FACULDADE SATC

ANA CAROLINA PATRÍCIO DANIELSKI
ISAAC DEBIASI DE SOUZA
IURY RIBEIRO ZANELATO
MAIARA ZANONI MARTINELLO
PAULO RICARDO ANDRÉ DA LUZ

PROJETO DE ELETRÔNICA I: ROBÔ SUMO

Criciúma
Junho – 2019

ANA CAROLINA PATRÍCIO DANIELSKI

ISAAC DEBIASI DE SOUZA

IURY RIBEIRO ZANELATO

MAIARA ZANONI MARTINELLO

PAULO RICARDO ANDRÉ DA LUZ

PROJETO DE ELETRÔNICA I: ROBÔ SUMO

Relatório apresentado ao Curso de Graduação em Engenharia Elétrica da Faculdade SATC, como parte dos requisitos avaliativos da disciplina de Eletrônica I, solicitado pelo professor Claudio Ernesto Ponce Saldias

Criciúma

Junho – 2019

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Este relatório tem como intuito apresentar o projeto do robô sumo capacitado para participar da categoria Beetlewight, que tem como regra a limitação das dimensões do robô, não podendo ultrapassar 20 cm em qualquer direção e podendo usar rampas que somadas não ultrapassem 10 cm do eixo em que estão ligadas e, ter peso máximo de 3 kg.

O projeto divide-se em parte estrutural, eletrônica e software. Explorando essas partes, pode-se dizer que o robô está dividido basicamente em carcaça e estrutura, motores, sensores, bateria e software, sendo o último totalmente embasado em um Arduino visando a sua autonomia.

Alguns materiais de reuso foram utilizados para minimizar os custos do robô, assim como se fez necessária a compra de equipamentos para a confecção do protótipo.

1.1 JUSTIFICATIVA E CONTRIBUIÇÕES

A competição é mais uma forma de incentivo, sendo o projeto uma maneira de assimilar os conhecimentos até aqui obtidos no curso de Bacharelado em Engenharia Elétrica e, sobretudo, mostrá-los na prática como funcionam.
Conceitos de matérias como Eletricidade Básica, Técnicas de Programação II, Desenho Técnico, Eletrônica I, Circuitos Elétricos, Resistência dos Materiais e, ambas as cadeiras de Física, por mais que alguns sejam básicos, foram abordados no trabalho, o que mostra a importância desse tipo de avaliação.

1.2 OBJETIVO

- Qualificar-se as regras do Beetlewight, sem exceções;
- Aplicar conhecimentos práticos de diversas áreas, com foco na matéria de Eletrônica;
- Criar um robô que cumpra seus objetivos com excelência.

2 FUNDAMENTAÇÃO TEÓRICA

A montagem do robô divide-se em algumas partes. Esse capítulo tem o propósito de demonstrar o detalhamento da montagem, os componentes e seus comportamentos nos testes e a viabilidade de cada um.

2.1 ESTRUTURA

Como o objetivo é uma competição de robôs em um ringue (Dohyō), a estrutura é de eximia importância, pois, além de que a esquematização dos componentes afeta diretamente nos resultados, sendo em proporção de peso ou possíveis curtos, também se precisa de resistência para aguentar a batalha.

2.1.1 Chassi e Carcaça

Por conta das limitações do peso para a categoria, precisa-se de um material que seja resistente, leve e de fácil manuseio. Sendo assim, decidiu-se usar o acrílico de 6mm para a confecção da carcaça, chassi e estruturação interna.

Além de sua densidade de 1.19 g/cm³, relativamente baixa, é resistente a fragmentação, de fácil manuseio, o que permite maior elaboração na construção de todo o robô. Para a rampa, preferiu-se o uso do inox, por sua grande resistência para choques contra os adversários e maleabilidade que agrega facilidade na montagem, além da segurança em casos de queda da arena.

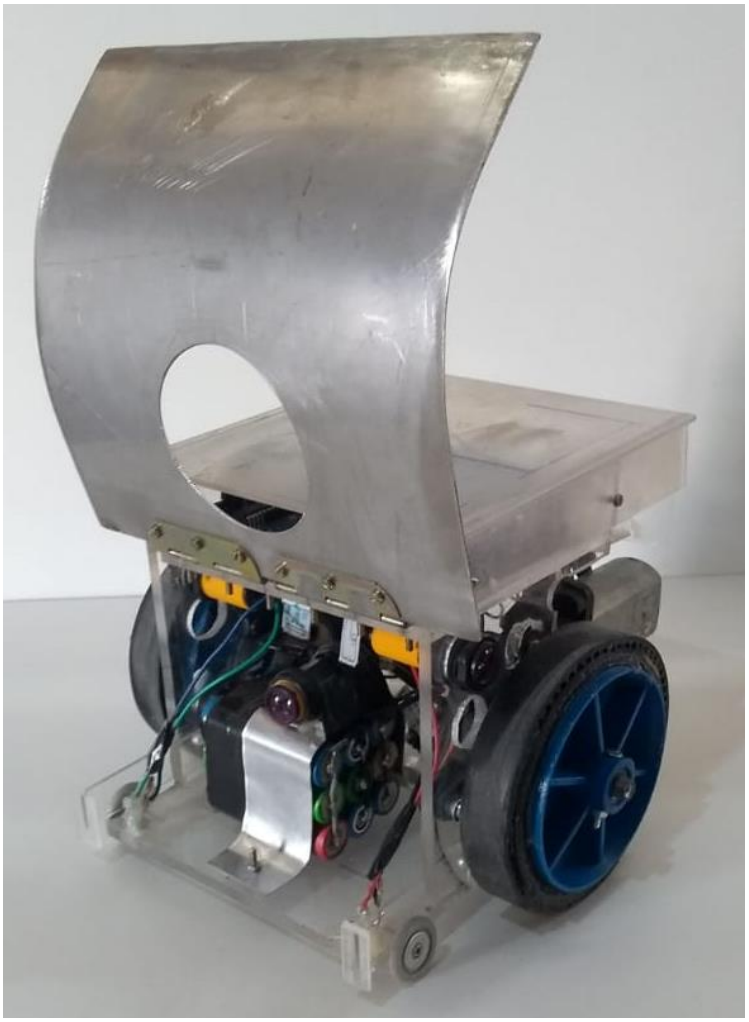

Fig. 1 – Estrutura do robô

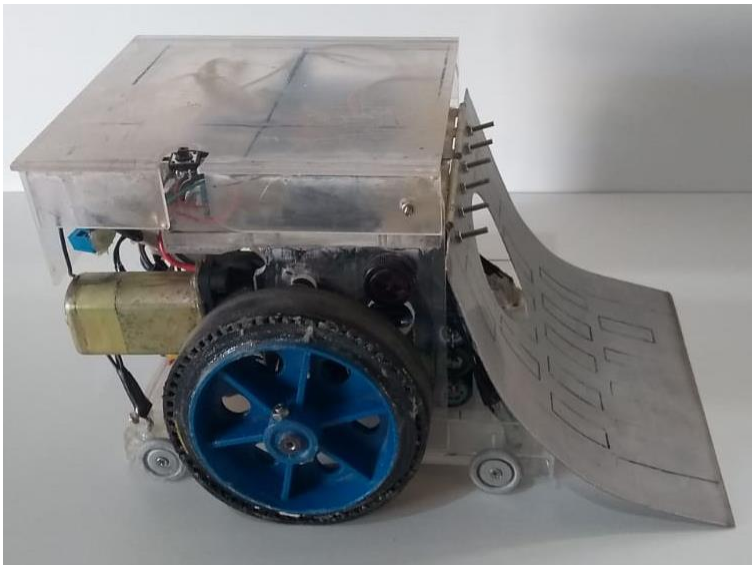

Fig. 2 – Testes de inclinação da rampa

2.1.2 Movimentação

O objetivo do combate é retirar o robô oponente do ringue (Dohyō), portanto, fluidez na movimentação é imprescindível. Para que não haja falhas nesse quesito, foi feito um dimensionamento das rodas, de forma que tivessem alto desempenho em tração. Desta forma, foi utilizado as rodas com diâmetro de 11 cm e borracha antiaderente de 130 mm com objetivo de evitar que o robô derrape quando estiver exercendo força contra o adversário.

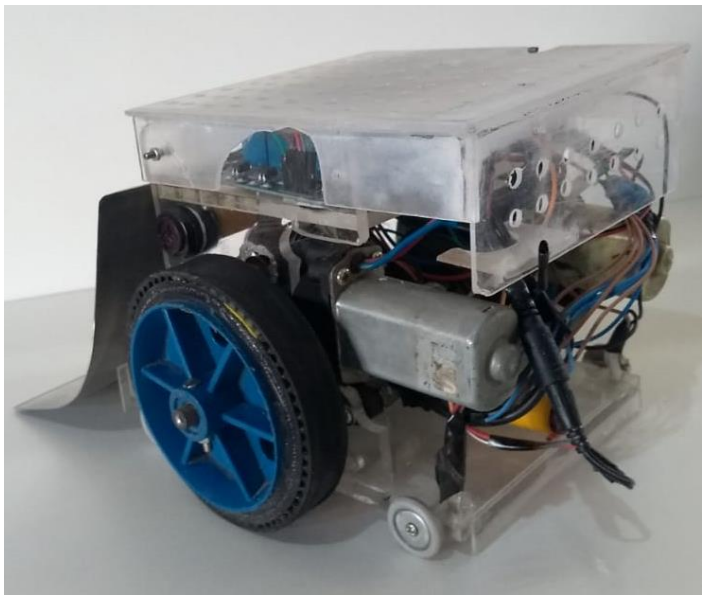

Fig. 3 – Rodas e parte da estrutura

Fig. 4 – Rodas

2.2 MOTORES

Entre os motores avaliados, o que se mostrou mais eficaz foi o motor usado em vidros elétricos automotivos. A necessidade que o motor precisa suprir é de gerar torque e velocidade consideráveis, além disso, é interessante que o motor possa travar os eixos em certos momentos, o que garante mais controle em partes do combate.

Na Fig. 5 o par de motores pode ser visualizado e junto a Tab. 1 com suas características.

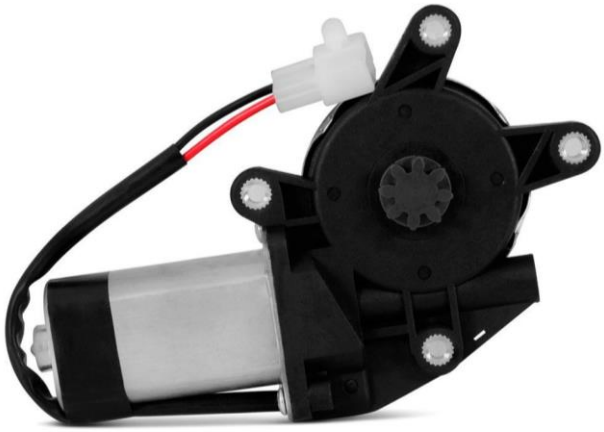

Figura 5 - Motor de Vidro Elétrico [4]

Tab. 1 – Características do Motor [4]

MOTOR	**ESQUERDO**	**DIREITO**
Torque (N.m)	9,12	9,12
Corrente nominal (A)	1,3	1,3
Tensão (V)	12	12

2.3 SENSORES

2.3.1 Sensor Reflexivo Infravermelho

O sensor infravermelho possui um circuito transmissor e um receptor, posicionados um ao lado do outro. Quando o robô do oponente for avistado pelo sensor, o sinal IR é refletido e detectado pelo receptor, que coloca o pino de saída em nível baixo (LOW), e aciona um LED localizado na parte traseira do sensor.

No protótipo foram instalados sensores deste modelo nas partes traseira, dianteira e nas laterais, para garantir que o robô possa localizar o adversário em qualquer posição, além de não ser surpreendido pelo mesmo.

A figura 6 mostra uma ilustração do sensor utilizado.

Fig. 6- Sensor Reflexivo Infravermelho [5]

2.3.2 Sensor Óptico

Para a detecção da linha branca ao redor da pista, utilizou-se este tipo de sensor porque possui embutido um emissor infravermelho e um fototransistor. O emissor é um led infravermelho que emite um sinal nessa faixa do espectro. Já o fototransistor é o receptor que faz a leitura do sinal refletido. Ou seja, o led emite um feixe infravermelho que pode ou não ser refletido por um objeto. Caso o feixe seja refletido, o fototransistor identifica o sinal refletido e gera um pulso em sua saída.

A detecção máxima não é a longa distancias, ficando em torno de 2,5 cm. O fototransistor vem com um filtro de luz ambiente, o que maximiza a identificação do feixe infravermelho refletido.

No protótipo, fora fixados quatro sensores de linha, um em cada ponta da estrutura da base quadrada do robô.

A figura 7 mostra uma ilustração do sensor utilizado.

Fig. 7–Sensor Óptico [5]

Tab. 1 – Características técnicas do sensor.

SENSOR TCRT5000	
Tensão reversa do LED emissor	5 VDC
Corrente direta do LED emissor	60mA
Tensão máximo coletor emissor transistor	70V
Corrente máxima de coletor	100mA
Distancia de detecção	2,5cm
Dimensões	10,2 x 5,8 x 7 mm

2.4 BATERIAS

Para a alimentação de todos os componentes foram utilizadas nove células de notebook. A montagem das mesas, deu-se de forma paralela em três

grupos de três células cada, e finalmente ligadas em série no final, garantindo a tensão necessária para os motores e não proporcionando uma queda de tensão tão rápida. A projeção foi feita para que a tensão chegue a 12V, que é a tensão que os motores trabalham.

Por serem células usadas podem apresentar problemas, o robô conta com um conjunto de baterias reserva, que poderão ser substituídas facilmente para as batalhas caso a principal apresente qualquer tipo de problema.

Para a alimentação do Arduino será utilizado uma bateria de 9V.

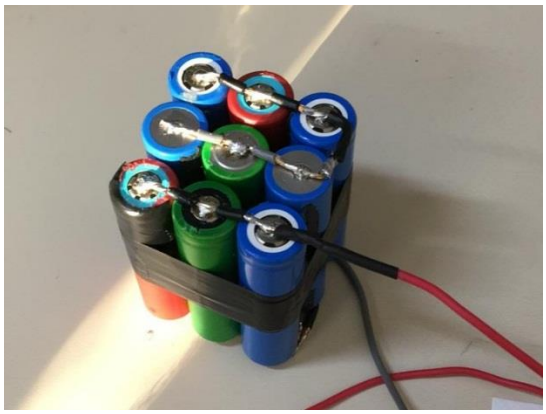

Fig. 8- Conjunto de células.

2.5 HARDWARE

Para automação e controle do robô será utilizado o microcontrolador Arduino que controla de forma eficaz os sensores e as outras partes elétricas do robô e uma ponte H usada para controlar o sentido de rotação do motor e que é capaz de aguentar as correntes altas dos motores.

2.5.1 Arduino

O Arduino foi criado com a ideia de diminuir o uso de outros aparelhos eletrônicos e substitui-los por um programa, isso diminui custos, perdas e principalmente tempo, pois, facílima na hora de encontrar erros. Um exemplo é a diminuição do uso das portas digitais como as XOR, XNOR, ADN e OR que trabalham com sinais lógicos, portanto, necessitam da programação física das

placas. No projeto ele será responsável pelo comando da ponte H que por sua vez controla os motores.

Entre os modelos de Arduino existentes, optou-se por usar o Arduino Uno R3, por ser uma plataforma aberta, onde a equipe pode ter acesso a ele com facilidade. Entre seus componentes estão um cristal feito de quartzo 16Mhz, 6 entradas analógicas, entrada de 5V, conexão USB para salvar programas, um micro controlador que se baseia em ATmega328P e possui 14 pinos digitais de entrada / saída e um botão de reset. [1]

Na Fig. 9 encontra-se o esquema com a descrição de cada parte do arduino escolhido, logo após na Tab.3, as suas características.

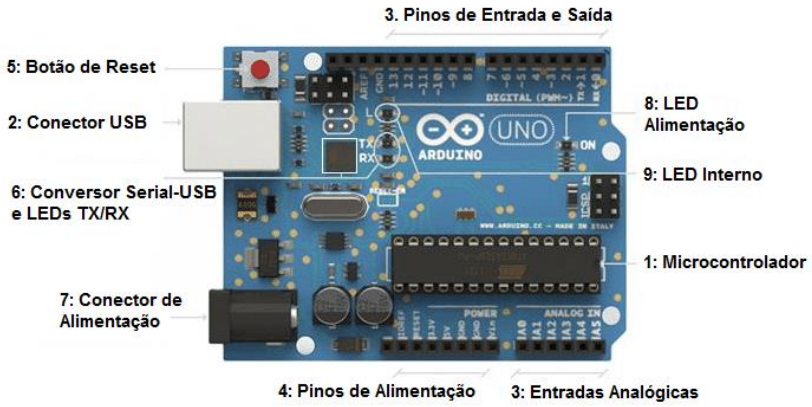

Figura 9 - Arduino UNO R3 [1]

Tab. 3 - Características Arduino UNO R3 [3]

ARDUINO UNO R3	
Tensão de operação	5V
Tensão de Alimentação recomendada	7 a 12 V
Tensão de Alimentação Limite	9,12 V
Corrente CC por pino I/O	40 mA
Corrente para pino de 3,3V	50mA

2.5.2 Ponte H

Para o robô, foi projetada uma ponte H que é um circuito capaz de controlar motores de corrente contínua a partir de sinais elétricos, permitindo a realização da inversão da direção da corrente e assim mudando a sentido de rotação do motor[2].

A ponte H será feita a partir de um módulo relé de quatro canais e que opera a tensão de 5V. Será ligado cada motor a dois relés que controlaram a sua direção, ou para frente ou para trás. O modulo será conectado tanto a uma bateria para ser energizado, como ao Arduino, para receber os comandos de direção dos motores necessários durante a luta.

Na Fig. 10 pode ser o modelo escolhido, na Tab. 2 as características do mesmo.

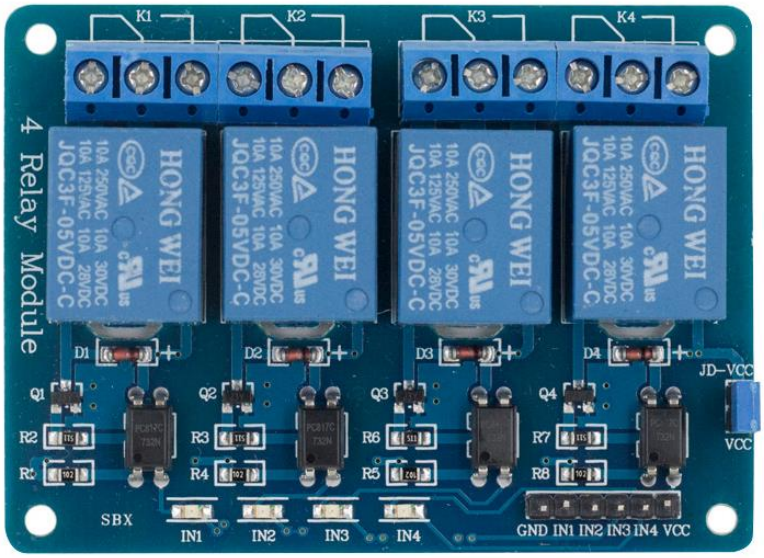

Fig.10 - Módulo Relé [4]

Tab. 4 - Características do Módulo Relé [4]

MÓDULO RELÉ	
Tensão de sinal	5V
Tensão de operação	5V
Canais	4
Capacidade do relé	(30V DC e 10A) ou (250V AC e 10A)

Tempo de resposta	5~10ms
Dimensões	71 mm x 53 mm x 20 mm

2.6 SOFTWARE

A programação foi realizada para controlar os sensores, os motores e todos os componentes utilizados para a autonomia do robô. Nela se definiu cada situação possível de acontecer na luta, criando uma resposta para a mesma. A programação completa está no apêndice A.

3 MONTAGEM E TESTES

Neste capitulo será apresentado como foi executada a montagem interna do robô, os testes realizados pela equipe e as estratégias empregadas na programação do robô.

3.1 MONTAGEM

Como falado no capítulo anterior, o robô usara um Arduino UNO, um módulo rele de 4 canais, três sensores de obstáculos, dois sensores de linha, dois motores de vidro elétrico e duas baterias. Eles foram conectados da seguinte maneira:

O módulo rele é conectado ao Arduino e a bateria de 12V. E os dois motores são conectados ao módulo. Como pode-se ver na Fig. 14.

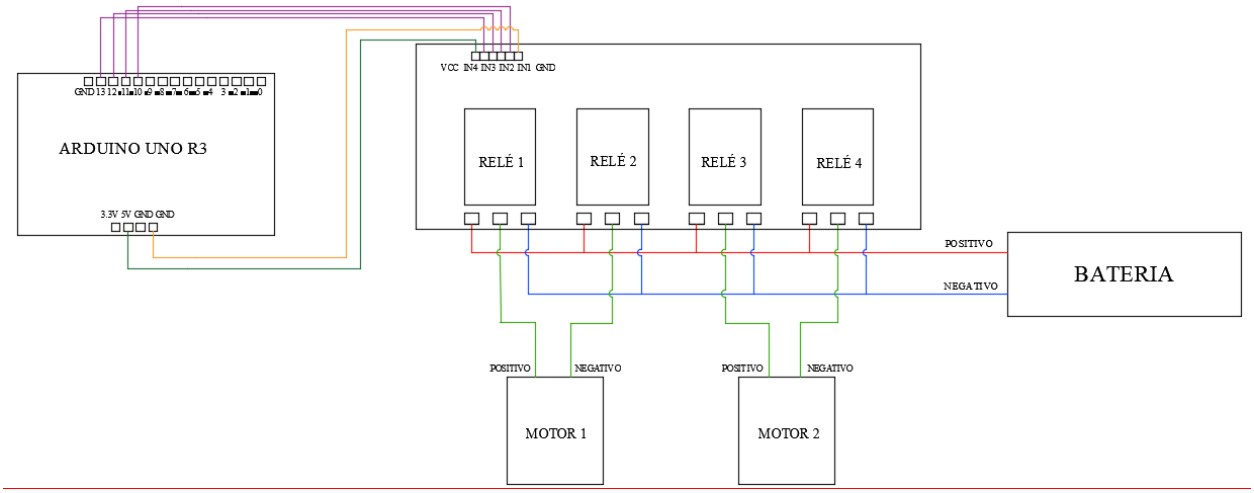

Fig. 2 - Esquema Módulo Relé

Os sensores de obstáculos são conectados somente ao Arduino, e como são três foram postos um na esquerda, um na direita e o outro na frente do robô. O esquema pode-se ver na Fig. 15.

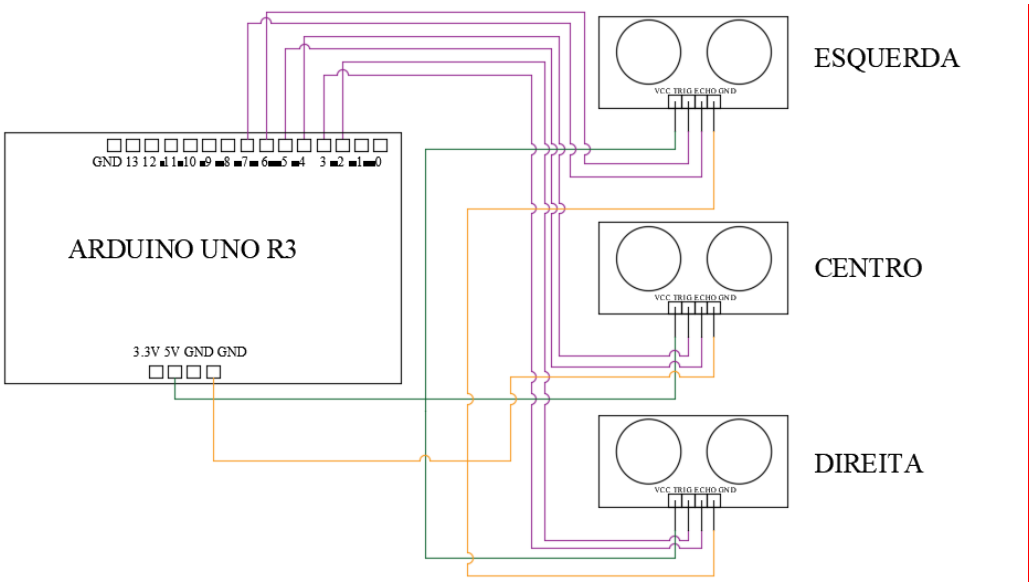

Fig. 3 - Esquema Sensor de Obstáculo

Os sensores de linha também são conectados somente no Arduino. São dois, um na direita e outro na esquerda do robô. O esquema pode ser visto na Fig. 16.

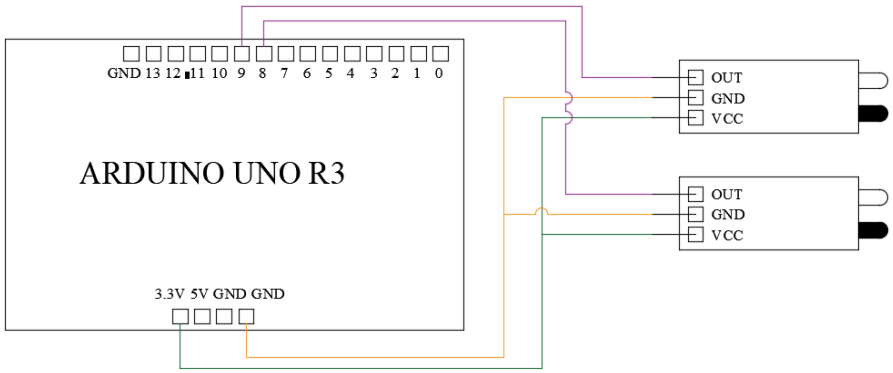

Fig. 4 - Esquema Sensor de Linha

A parte interna juntando esses componentes descritos no início ficou da seguinte maneira:

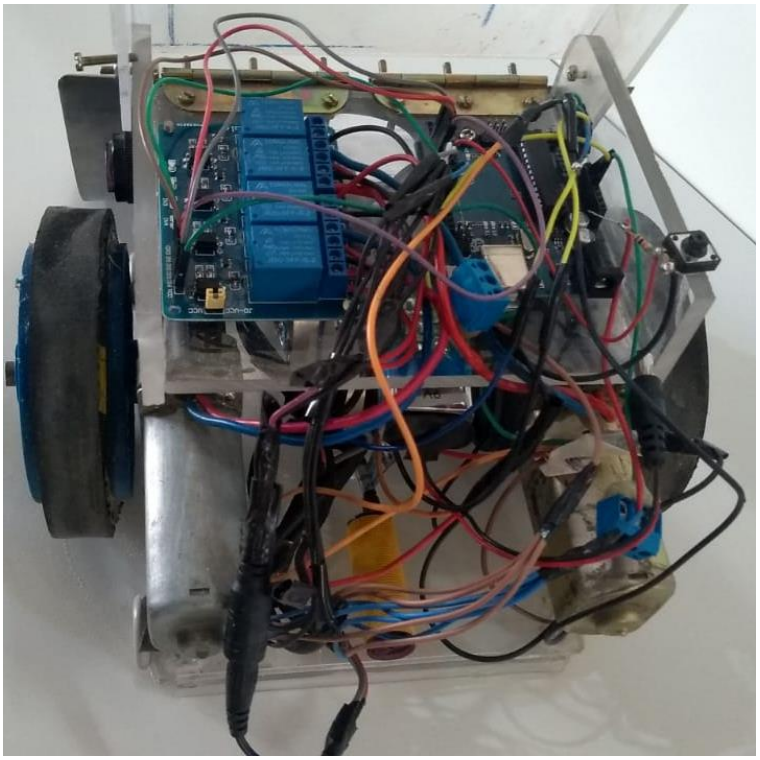

Fig. 5 - Parte Interna do Robô

3.2 TESTES

Com uma pista confeccionada pela equipe como mostra a figura 11, foi possível a realização dos testes em todos os componentes do robô. Os sensores de linha diferenciam a cor branca da borda da pista, da cor preta de toda a superfície do ringue e voltam quando a primeira é detectada.

Os sensores reflexivos de obstáculo detectam qualquer movimento e respondem conforme foram programados. E todos os componentes respondem bem e executam suas funções corretamente.

Fig.11 – Pista de testes

3.3 ESTRATÉGIAS PARA A LUTA

A estratégia de luta foi pensada de forma que após os cinco segundos iniciais saia em busca do adversário por meio dos quatro sensores em suas extremidades para localização mais rápida e eficiente do adversário, como também para sua própria segurança para não ser surpreendido.

Caso algum adversário tente empurrá-lo para fora da arena ele usará tração contrária para evitar que saia derrotado da batalha.

Os sensores das laterais estão programados para que ocorra um contato longo, que possivelmente seria o oponente o empurrando para fora, ele usará o mecanismo de defesa e atuará para frente ou para trás para tentar escapar.

Para atacar os adversários sua resposta de atuação também será inteligente de forma que independente do sensor que localizá-lo, ele se projetará de frente, de modo que sua rampa possa tirar parte do oponente do chão, facilitando o objetivo que é de empurrar o adversário, claro que com o cuidado de identificar a linha branca da arena para não acabar fora também.

Suas rodas ante aderentes terão forte influência para que ele não derrape e perca o torque.

CONSIDERAÇÕES FINAIS

Após vários testes e considerando tudo o que foi apresentado nos relatórios anteriores a este, o projeto chega na sua fase final contemplando todas as condições para participar da competição. Desde o início houve alterações de componentes e na programação do Arduino, sempre se preocupando com as condições da competição e o bom desempenho durante a competição.

Não resta muito a dizer além de que o robô atendeu as expectativas da equipe. A estrutura mostrou-se resistente, os circuitos comportaram-se bem e a programação deve boa resposta aos sinais, assim como a ponte H teve uma boa montagem de forma que os motores funcionam conforme o desejado.

REFERÊNCIAS

[1] MOTA, Allan. **O que é Arduino e como funciona? 2017.** Disponível em: <https://portal.vidadesilicio.com.br/o-que-e-arduino-e-como-funciona/>. Acesso em: 23 março 2019

[2] ROBOLIV.RE. **Ponte H.** Disponível em: <http://www.roboliv.re/conteudo/ponte-h>. Acesso em: 23 março 2019.

[3] SILÍCIO, Vida de. **Arduino Uno R3.** Disponível em: <https://www.vidadesilicio.com.br/arduino-uno-r3>. Acesso em: 27 março 2019.

[4] PARTS, Connect. **Motor Vidro Elétrico Mabuchi 8 Dentes 12V.** Disponível em: <https://www.autopartsonline.com.br/www-autopartsonline-com-br/motor-maq-vidro-8-dentes-esquerdo>. Acesso em: 01abr. 2019.

[5] CANDIDO, Gradimilo. **Robô seguidor de linha com sensor infravermelho e PWM. 2018.** Disponível em: <https://portal.vidadesilicio.com.br/robo-seguidor-de-linha-sensor-infravermelho-e-pwm/>. Acesso em: 28fev. 2019.

[6]THOMSEN, Adilson. **Detector de proximidade com sensor infravermelho.** Disponível em: <https://www.filipeflop.com/blog/sensor-infravermelho-arduino/>. Acesso em: 25 ago. 2018.

[7] OLIVEIRA, Euler. **Como usar com Arduino – Sensor Óptico Reflexivo TCRT5000.** Disponível em: < http://blogmasterwalkershop.com.br/arduino/arduino-utilizando-o-sensor-reflexivo-tcrt5000/>. Acesso em: 30março 2019.

[8] PRODUTOS E SOLUÇÕES EM ALUMÍNIO, Belmetal. **Características do Acrílico.** Disponível em:<https://www.aecweb.com.br/cls/catalogos/belmetal/chapas-acrilico.pdf>. Acesso em 01 abr.2019.

APÊNDICES

APÊNDICE A – PROGRAMAÇÃO DO ARDUINO

```
#define trava 1500//CONSTANTES usadas para facilitar mudanças na programação
#define tre  500//Usadas no valor de leitura dos sensores de linha, e nas equações de tempo.
#define tgiro 300
#define tgirototal 700
#define vslin1 400
#define vslin2 400
#define vslin3 400
#define vslin4 400
//DEFINE os pinos digitais como ENTRADA dos sensores de obstáculos.
int SobsF = 11;  //FRENTE
int SobsT = 12;  //TRASEIRO
int SobsE = 9;   //ESQUERDO
int SobsD = 10;  //DIREITO

//DEFINE os pinos analógicos como ENTRADA dos sensores de linhas.
int Slin1 = A1;      //NOROESTE
int Slin2 = A0;      //NORDESTE
int Slin3 = A2;      //SUDOESTE
int Slin4 = A3;      //SUDESTE

int MdirF = 5;       //Define o SENTIDO de rotação dos motores.
int MdirT = 6;
int MesqF = 3;
int MesqT = 4;
int botao = 8;       //Define o pino digital 8 como SINAL do botão de inicio.

//VARIAVEIS de tempo necessárias para alguns pontos da programação.
```

```
//É usado mais de 1 variável para não haver conflito, por uma variável ser usada 2 vezes ao //mesmo tempo.

unsigned long Tinicio;
unsigned long Tinicio2;
unsigned long Tinicio3;
int estado;          //Variável responsável pelo inicio ou parada da movimentação do robô.


//FUNÇÕES criadas para cada tipo de ação, separadas da parte principal da programação.
void parar(void);
void start(void);
void resq(void);
void rdir(void);
void frente(void);
void tras(void);
void inimigoesquerda(void);
void inimigodireita (void);
void inimigoatras(void);

void setup() {


  pinMode(MdirF, OUTPUT);      //DEFINE os pinos dos motores como saidas
  pinMode(MdirT, OUTPUT);
  pinMode(MesqF, OUTPUT);
  pinMode(MesqT, OUTPUT);

  estado = 0;                  //Estado inicial, parado.
}


void loop() {

// Se o estado for igual a 0, ele ira para a função parar(), seguida da função start().
  if (estado == 0) {
  parar();
```

```
 start();
 }



 do { //Executara tudo dentro deste paretese, enquanto estado for igual a 3.


//Esta função IF aparecera muitas vezes ao longo do codigo,
// ela quem assegurara que apos o robo estar ligado,
// quando o botao ser pressionado novamente, ele ira parar e esperar ser reiniciado.

 if (digitalRead(botao) == HIGH) {
     estado = 0;
     parar();
   }
   else {
//Le os valores dos sensores de linha,
//Dependendo dos resultados(se forem positivos ou nao), eles executam uma ação
//para que a leitura seja negativa, ou seja, esteja fora da linha branca
     if ( analogRead(Slin2) < vslin2) {
       resq();
     }
     if (analogRead(Slin1) < vslin1 ) {
       rdir();
     }
     if (analogRead(Slin3) < vslin3) {
       frente();
     }
     if (analogRead(Slin4) < vslin4) {
       frente();
     }

//Se nenhum sensor de linha esta ativado, ele passa a ler os sensores de obstaculo.
```

```
    if (analogRead(Slin1) > vslin1 && analogRead(Slin2) > vslin2 && analogRead(Slin3) >
vslin3 && analogRead(Slin4) > vslin4) {
      frente();
//O codigo da prioridade ao sensor de obstaculo frontal, logo, esta a um IF
superior aos demais.
      if (digitalRead(SobsF) == LOW) {
        frente();       }
      else {
//Dependendo de qual sensor for ativado, ele executa uma função diferente.
        if (digitalRead(SobsE) == LOW) {
          inimigoesquerda();
        }
        if (digitalRead(SobsD) == LOW) {
          inimigodireita();
        }
        if (digitalRead(SobsT) == LOW) {
          inimigoatras();
        }
      }
    }
  }

  } while (estado == 3);
  return (0);
}




void start() {


//Parte do codigo para o deligamento do robo, se estado for 3(ligado) e o botao
ser pressionado, estado passa a ser 0(desligado).

  if (digitalRead(botao) == HIGH  && estado == 3 ) {
    estado = 0;
```

```
}

do {
//O botao precisa ser mantido pressionado por pelo menos 0.5s para iniciar o codigo de movimentação.
  Tinicio = millis();
  do {
    if ((millis() - Tinicio) > 500 ) {
      estado = 1;
    }
    else {
      delay(100);
    }
  } while (digitalRead(botao) == HIGH );
} while (estado == 0);


do {  //Apos pressionado por 0.5s, quando solto iniciara a contagem de 5s para o inicio do movimento.
  if (digitalRead(botao) == LOW) {
    estado = 2;
  }
} while (estado == 1);

do {        // inicio do funcionamento apos soltar o botao
  delay(5000);
  estado = 3;                              //define a variavel estado = 3, ligado.

  tras();       // apos os 5s, dara um impulso com RE+FEENTE para cair a rampa.

  delay(100);

  frente();

} while (estado == 2);
```

```
}



void parar() {

  digitalWrite(MdirT, LOW);
  digitalWrite(MdirF, LOW);

  digitalWrite(MesqT, LOW);
  digitalWrite(MesqF, LOW);



}

void frente() {

  if (digitalRead(botao) == HIGH) {
    estado = 0;
    parar();
    return (0);
  }

  digitalWrite(MdirT, HIGH);
  digitalWrite(MdirF, LOW);

  digitalWrite(MesqT, HIGH);
  digitalWrite(MesqF, LOW);



}
```

```
void tras() {

  if (digitalRead(botao) == HIGH) {
    estado = 0;

    return (0);
  }

  digitalWrite(MdirT, LOW);
  digitalWrite(MdirF, HIGH);

  digitalWrite(MesqT, LOW);
  digitalWrite(MesqF, HIGH);

}
// Todas as funções a seguir VERIFICAM todos os sensores de linha antes de qualquer movimento.

void resq() {

  Tinicio = millis();
  do {   // Inicia um movimento de RÉ por um determinado tempo

    if (analogRead(Slin3) > vslin3 && analogRead(Slin4) > vslin4) {
      if (digitalRead(botao) == HIGH) {
        estado = 0;
        return (0);
      }
      digitalWrite(MdirT, LOW);
      digitalWrite(MdirF, HIGH);
```

```
    digitalWrite(MesqT, LOW);
    digitalWrite(MesqF, HIGH);


   }
   else {
    return (0);
   }
  } while ((millis() - tre) < Tinicio);


  Tinicio2 = millis();
   //Apos o movimento de RÉ, gira para a esquerda por um determinado tempo.
do {
   if (analogRead(Slin3) > vslin3 && analogRead(Slin4) > vslin4) {
    if (digitalRead(botao) == HIGH) {
     estado = 0;


     return (0);
    }
    digitalWrite(MdirT, LOW);
    digitalWrite(MdirF, HIGH);

    digitalWrite(MesqT, HIGH);
    digitalWrite(MesqF, LOW);


   }
   else {
    return (0);
   }
  } while ((millis() - tgirototal) < Tinicio2);
  return (0);

}
```

```
void rdir() {

  Tinicio = millis();

  do {     // Inicia um movimento de RÉ por um determinado tempo
    if (analogRead(Slin3) > vslin3 && analogRead(Slin4) > vslin4) {
      if (digitalRead(botao) == HIGH) {
        estado = 0;

        return (0);
      }
      digitalWrite(MdirT, LOW);
      digitalWrite(MdirF, HIGH);

      digitalWrite(MesqT, LOW);
      digitalWrite(MesqF, HIGH);

    }
    else {
      return (0);
    }
  } while ((millis() - tre) < Tinicio);

  Tinicio2 = millis();
//Apos o movimento de RÉ, gira para a direita por um determinado tempo.
do {
    if (analogRead(Slin3) > vslin3 && analogRead(Slin4) > vslin4) {
      if (digitalRead(botao) == HIGH) {
        estado = 0;
```

```
      return (0);
     }
     digitalWrite(MdirT, HIGH);
     digitalWrite(MdirF, LOW);

     digitalWrite(MesqT, LOW);
     digitalWrite(MesqF, HIGH);
    }
    else {
     return (0);
    }
  } while ((millis() - tgirototal) < Tinicio2);

  return (0);

}

void inimigoesquerda() {

  Tinicio = millis();
  do {//Gira para a esquerda por um determinado tempo.
    if (analogRead(Slin3) > vslin3 && analogRead(Slin4) > vslin4 && digitalRead(SobsF) ==
HIGH) {
     if (digitalRead(botao) == HIGH) {
      estado = 0;
      return (0);
     }
     digitalWrite(MdirT, HIGH);
     digitalWrite(MdirF, LOW);

     digitalWrite(MesqT, LOW);
     digitalWrite(MesqF, HIGH);
```

```
    }
    else {
      return (0);
    }
  } while ((millis() - trava ) < Tinicio);


  if (digitalRead(SobsE) == LOW) {
  //Se o tempo determinado anteriormente for atingido, e mesmo assim continuar com o sensor de obstaculo ativo, inicia um movimento de RÉ, seguido de um giro para esquerda novamente.

    Tinicio2 = millis();
    do {
      if (analogRead(Slin3) > vslin3 && analogRead(Slin4) > vslin4
&& digitalRead(SobsF) == HIGH) {
        if (digitalRead(botao) == HIGH) {
          estado = 0;


          return (0);
        }
        digitalWrite(MdirT, LOW);
        digitalWrite(MdirF, HIGH);

        digitalWrite(MesqT, LOW);
        digitalWrite(MesqF, HIGH);


      }
      else {
        return (0);
      }
    } while ((millis() - tre) < Tinicio2);

    Tinicio3 = millis();
    do {
```

```
    if (analogRead(Slin3) > vslin3 && analogRead(Slin4) > vslin4 && digitalRead(SobsF)
== HIGH) {
      if (digitalRead(botao) == HIGH) {
        estado = 0;

        return (0);
      }
      digitalWrite(MdirT, HIGH);
      digitalWrite(MdirF, LOW);

      digitalWrite(MesqT, LOW);
      digitalWrite(MesqF, HIGH);

    }
    else {
      return (0);
    }
  } while ((millis() - tgiro) < Tinicio2);
  return (0);



 }
 return (0);
}

void inimigodireita() {

 Tinicio = millis();
 do {   //Gira para a direita por um determinado tempo.
  if (analogRead(Slin3) > vslin3 && analogRead(Slin4) > vslin4 && digitalRead(SobsF) ==
HIGH) {
    if (digitalRead(botao) == HIGH) {
      estado = 0;
      return (0);
```

```
    }
    digitalWrite(MdirT, LOW);
    digitalWrite(MdirF, HIGH);

    digitalWrite(MesqT, HIGH);
    digitalWrite(MesqF, LOW);

   }
   else {
    return (0);
   }
  } while ((millis() - trava ) < Tinicio);

  if (digitalRead(SobsE) == LOW) {
//Se o tempo determinado anteriormente for atingido, e mesmo assim continuar com o sensor de obstaculo ativo, inicia um movimento de RÉ, seguido de um giro para direita novamente.

   Tinicio2 = millis();
   do {
    if (analogRead(Slin3) > vslin3 && analogRead(Slin4) > vslin4 && digitalRead(SobsF) == HIGH) {
      if (digitalRead(botao) == HIGH) {
       estado = 0;

       return (0);
      }
      digitalWrite(MdirT, LOW);
      digitalWrite(MdirF, HIGH);

      digitalWrite(MesqT, LOW);
      digitalWrite(MesqF, HIGH);

    }
    else {
```

```
        return (0);
      }
    } while ((millis() - tre) < Tinicio2);

    Tinicio3 = millis();
    do {
      if (analogRead(Slin3) > vslin3 && analogRead(Slin4) > vslin4
&& digitalRead(SobsF) == HIGH) {
        if (digitalRead(botao) == HIGH) {
          estado = 0;

          return (0);
        }
        digitalWrite(MdirT, LOW);
        digitalWrite(MdirF, HIGH);

        digitalWrite(MesqT, HIGH);
        digitalWrite(MesqF, LOW);

      }
      else {
        return (0);
      }
    } while ((millis() - tgiro) < Tinicio2);
    return (0);


  }
  return (0);
}

void inimigoatras() {
//Inicia um movimento de giro para a esquerda, com objetivo de rotacionar o
robo em 180graus.
```

```
  Tinicio = millis();
  do {
    if (analogRead(Slin3) > vslin3 && analogRead(Slin4) > vslin4 && digitalRead(SobsF) ==
HIGH) {
      if (digitalRead(botao) == HIGH) {
        estado = 0;

        return (0);
      }
      digitalWrite(MdirT, HIGH);
      digitalWrite(MdirF, LOW);

      digitalWrite(MesqT, LOW);
      digitalWrite(MesqF, HIGH);

    }
    else {
      return (0);
    }

  } while ((millis() - tgirototal) < Tinicio);
  return (0);
}
```