CENTRO PAULA SOUZA PAULINO BOTELHO

Técnico em Mecatrônica

Autores:

Abner da Silva Fonseca

Bruno Birce Rodrigues

Bruno Roberto Candelora

Gabriel Fernandes dos Santos

Geovane Mendes de Almeida

CARRINHO AUTOGUIADO (CAG)

São Carlos

Julho 2022

Abner da Silva Fonseca

Bruno Birce Rodrigues

Bruno Roberto Candelora

Gabriel Fernandes dos Santos

Geovane Mendes de Almeida

CARRINHO AUTOGUIADO (CAG)

Trabalho de Conclusão de Curso apresentado ao Curso Técnico em Mecatrônica da Etc. Paulino Botelho, como requisito parcial para obtenção do título de Técnico em Mecatrônica. Orientador: Prof. Mestre Claudio Torres.

São Carlos

Julho 2022

TERMO DE APROVAÇÃO

CARRINHO AUTOGUIADO (CAG)

Abner da Silva Fonseca

Bruno Birce Rodrigues

Bruno Roberto Candelora

Gabriel Fernandes dos Santos

Geovane Mendes de Almeida

Aprovado em_____/___/___

BANCA EXAMINADORA

_____

Prof. Mestre: CLAUDIO TORRES

_____

Prof. Mestre: ANDERSON BELUCO

_____

Prof. Mestre: EVANDRA MARIA RAYMUNDO

AGRADECIMENTOS

Aos professores, pelas correções e ensinamentos que permitiram apresentar um melhor desempenho no processo de formação Técnica ao longo do curso.

À instituição de ensino Centro Paula Souza, essencial no processo de formação Técnica, pela dedicação, e por tudo o que aprendemos ao longo dos anos do curso.

A todos aqueles que contribuíram, de alguma forma, para a realização deste trabalho.

“O talento vence jogos, mas só o trabalho em equipe ganha campeonatos.”

Michael Jordan

RESUMO

Neste relatório propomos apresentar o desenvolvimento do carrinho autoguiado por uma linha preta, com objetivo de aplicar os conhecimentos adquiridos de todas as disciplinas do curso em técnico em Mecatrônica. O projeto consiste, de um carrinho 100% feito de manufatura aditiva, sendo movido por quatros motores servos utilizando rodas omnidirecionais. Além disso o carrinho vai ser autoguiado por uma fita utilizando sensores infravermelhos.

Palavra-chave: Carrinho autoguiado, sensores infravermelhos, manufatura aditiva, roda omnidirecionais.

ABSTRACT

In this report, we propose to present the development of the self-guided trolley with a black line, with the objective of applying the knowledge acquired from all subjects of the Mechatronics technician course. The project consists of a cart 100% made of additive manufacturing, being driven by four servo motors using omnidirectional wheels. In addition, the cart will be self-guided by a tape using infrared sensors.

Keywords: Self-guided cart, infrared sensors, additive manufacturing, omnidirectional wheel.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

Ao longo dos anos, a importância da automação vem crescendo no meio industrial para melhorar os processos, reduzindo assim os trabalhos humanos e aumentando a segurança das pessoas e aperfeiçoando a qualidade de seus produtos.

As indústrias brasileiras vêm buscando possíveis soluções na automação[1], para ajudar a resolver o problema de transição de carregamento de material ou ferramental dentro da empresa. As indústrias americanas, desde dá década de 50, criou uma possível solução e vem aperfeiçoado de décadas e décadas a utilização de um Veículo Autoguiado (AGV), é um robô 100% automatizado que transporta material através de um percurso. Uma das possíveis soluções nas indústrias brasileiras, é a implementação de um veículo autônomo que é capaz de percorrer um percurso (guiado por uma linha) pela fábrica, carregando materiais, ferramentais, reduzindo tempo nas transições de pessoas e aumentando a produtividade da fábrica.

Dessa maneira, a implementação de um veículo autônomo poderá sanar os problemas citado anteriormente, e facilitar nos transportes de ferramental ou material. Em suma, devemos estudar e criar um protótipo chamado de carrinho autoguiado (CAG) por fita, pois abrangerá o conhecimento do assunto em prol da eficiência para o meio industrial.

2. DESENVOLVIMENTO

Nossa pesquisa se iniciou com trocas de ideias e debates, discutimos vários tipos de materiais de manufatura aditiva[2] e modelagem da estrutura do carrinho. Realizamos várias pesquisas e chegamos à conclusão de que o material indicado para a estrutura seria o PLA[3] e os componentes de reforço seria o ABS[4] para suportar o

---

[1] Automação é um processo que realiza tarefas de forma autônoma ou que precisa de auxiliação de pessoas para realizar a tarefas do dia a dia.

[2]“Manufatura aditiva é a impressão 3D, que imprime objetos através da sobreposição progressiva de um material”. (TOTVS).

[3] “O PLA é um termoplástico biodegradável de origem natural e de fontes renováveis, como amido de milho ou cana-de-açúcar”. (3DLAB).

[4] “O ABS e Acrilonitrila Butadieno Estireno. Se trata de um polímero bastante utilizado pelas empresas por conta de suas boas propriedades, como a resistência mecânica”. (3DLAB).

peso do transporte de materiais ou ferramentais.

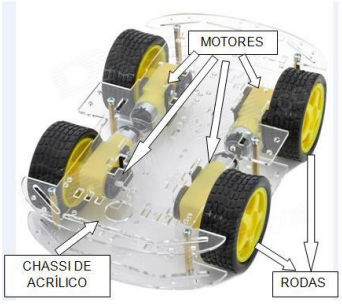
Nosso projeto consiste na criação de um carrinho autoguiado para otimizar o tempo e processo de transporte de materiais e ferramentais usado em linhas de produção nas fábricas.

- Agilizar o Processo
- Produtividade maior, pelo menor tempo
- Lucro
- Reduzir esforços dos trabalhadores
- Facilitar ao público de pessoas com deficiências (PcD).

2.1. MODELAGEM DO CARRINHO AUTOGUIADO (CAG)

A modelagem do carrinho foi desenvolvida utilizando dois programas Solidworks 2018[5] e o Ultimake cura (versão 4.13.1)[6]. O Solidworks converte o arquivo de modelagem em STL, e o Ultimake cura reconhece o arquivo STL e transforma gcode (formato que a impressora reconhece). A manufatura aditiva foi feita na Ender (3 e 5 Pro) [7]. O conceito inicial do carrinho era para ser baseado em um modelo prancha, conforme a figura1: Modelo prancha em acrílico e figura 2: Modelos prancha em ABS.

Figura 1: Modelo prancha em acrílico



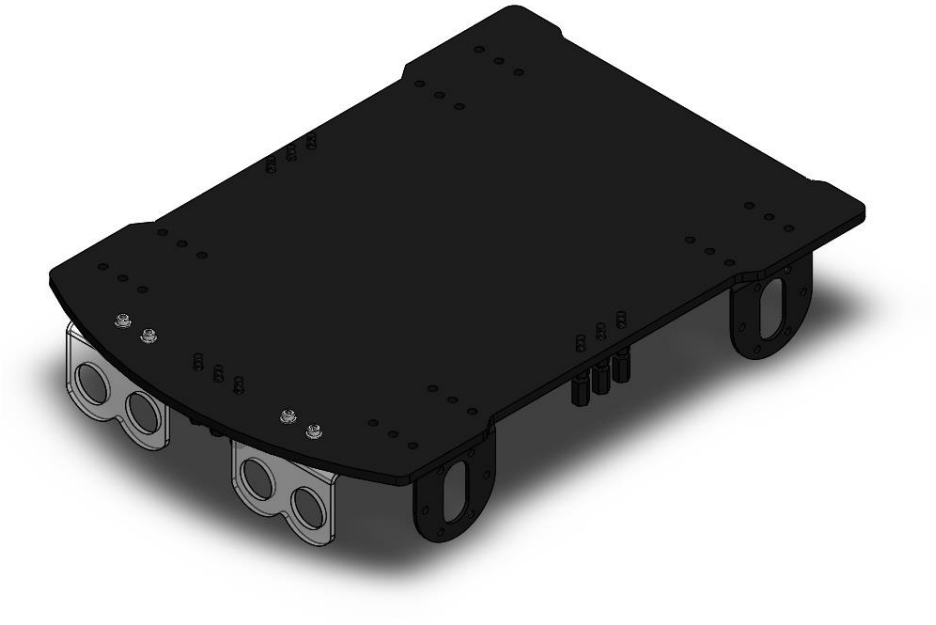
Fonte: Silva (2016, p.22)

[5] “SOLIDWORKS é um software CAD 3D que auxilia na criação e inovação de projetos com foco em reduzir o ciclo de desenvolvimento do produto desde o design até a manufatura”. (SKA).

[6] “Ultimake cura é um software fatiador para impressoras 3D. Ele transforma um modelo 3D em camadas que quando sobrepostas formam o objeto que será impresso”. (3DLAB).

[7] Impressora 3D

Figura 2: Modelo prancha em ABS



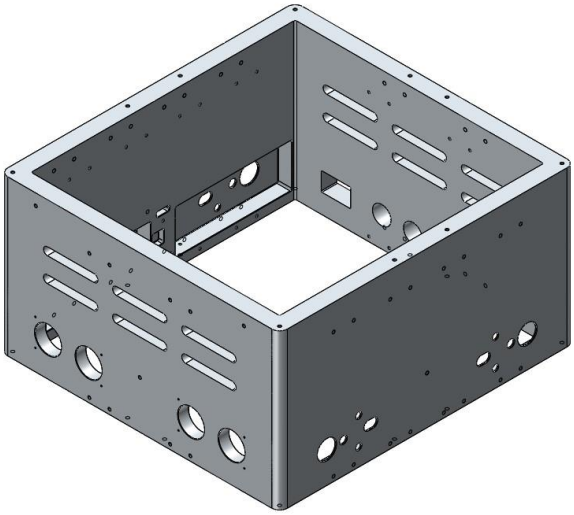
Fonte: Autor

O conceito final do carrinho foi feito próximo ao estilo do AVG, conforme a figura 3: AGV do tipo rebocador e a figura 4: Modelo do carrinho estilo caixa em PLA.

Figura 3: AVG do tipo rebocador

Fonte: Silva (2016, p.12)

Figura 4: Modelo do carrinho estilo caixa em PLA



Fonte: Autor

2.2. MONTAGEM DO CARRINHO AUTOGUIADO (CAG)

A montagem inicial do CAG é constituída de quatros motores de 3 a 6 v, quatros rodas omnidirecionais[8], três sensores de obstáculos Infravermelho (Lm393)[9], dois Sensor Ultrassónico(Hc-Sr04-2020)[10], duas pontes H(L298n)[11], Arduino(Leonardo)[12], Protoboard[13], duas chave (Liga e desliga), Modulo de bateria (duas de 7 e uma de 10) três plugs de carregamento, duas dobradiças, PP-CAG001(Estrutura externa), dois PM-CAG012 (Estrutura interna superior e inferior), PP-CAG007(Tampa de baixo), PP-CAG008(Tampa interna para o suporte do Arduino e ponte H), PP-CAG009(suporte do Arduino e ponte H), PP-CAG010(Tampa para suportar material), PM-

[8] Rodas omnidirecionais, são rodas que não precisa de eixos para esterçar, ela tem a possiblidade de fazer o veículo esterçar 360 graus. (POSITIVO,2018).

[9] É um sensor que trabalha com sistema de reflexão de luz infravermelha. (ARDUINOBELEM).

[10] É um sensor que mede objetos através de emissão de onda sonora. (FILIPEFLOP).

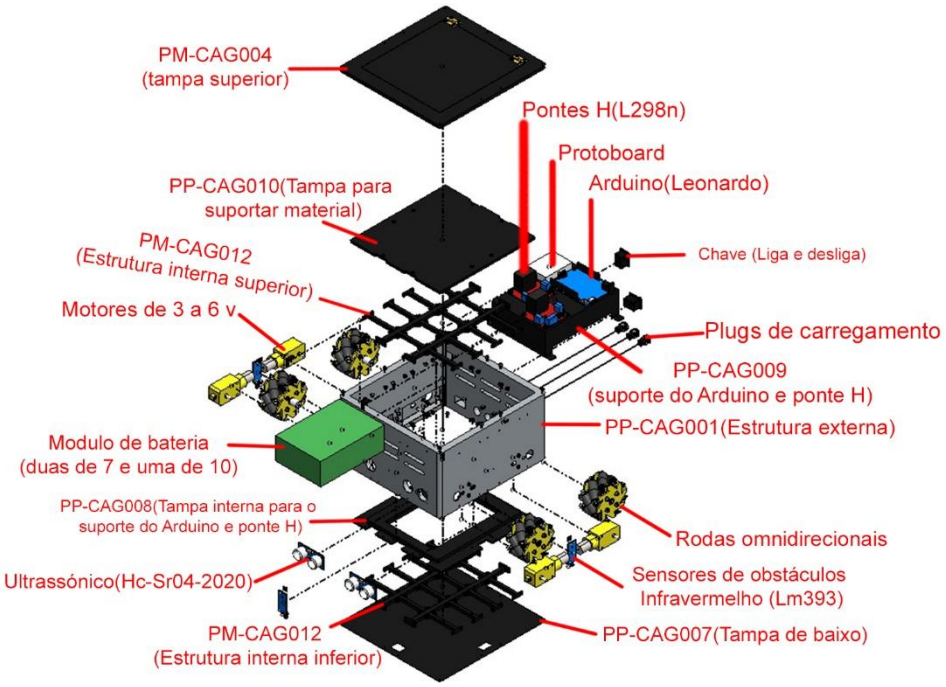
[11] "Ponte H L298N é um módulo de controle para cargas indutivas (corrente que atravessa impulsiona um campo magnético no indutor), relés, solenoides, motores DC e motores de passo". (SARAVATI; ALUGAGERA).

[12] "Arduino é conhecida por conseguir realizar uma série de tarefas relacionadas a automação, robótica etc. Mas sozinha a placa Arduino não consegue realizar muitas tarefas, por isso ele precisará de sensores, módulos, motores, e diversas outras partes para compor um projeto específico". (ELETROGATE).

[13] Placa para montagem de circuito elétrico. (ROBOCORE).

CAG004(tampa superior), e PP-CAG013 (batente da tampa superior). A montagem está representada conforme a figura 5: Montagem geral CAG.

Figura 5: Montagem geral CAG.



Fonte: Autor

O padrão de legenda de fabricação e montagem foi feito conforme o exemplo:

Legenda - PP-XXX(YYY)ZZZ

- PP - Parte De Peças = Fabricação
- XXX - Nome do Projeto
- YYY - Número da Peça De Fabricação
- ZZZ - Ano do Projeto

Obs.: a fabricação de peça, o número é do maior para menor. EX: PP - CAG (001)2100; PP - CAG (002)2100; PP - CAG (003)2100..... PP - CAG (013)2100

Legenda - PM-XXX(YYY)ZZZZ

- PM - Parte de montagem = montagem do projeto
- XXX - Nome do projeto
- YYY - Número da peça de montagem
- ZZZ - ano do projeto

Obs.: montagem de peça, o número é do menor para maior. EX: PM - CAG (0012)2100; PM - CAG (011)2100; PM - CAG (010)2100..... PM - CAG (001)2100.

Na montagem final do CAG, foi substituído os três sensores de obstáculos Infravermelho (Lm393), por dois módulos sensor óptico (TCRT5000) para obter mais precisão na hora de ler branco / preta. houve acrescentação de mais uma chave (Liga e desliga) para a ponte H (esquerda), e a remoção da estrutura interna superior, tampa para suporte de material e os dois Sensor Ultrassónico (Hc-Sr04-2020). A montagem está representada conforme a figura 6: Montagem Final CAG.

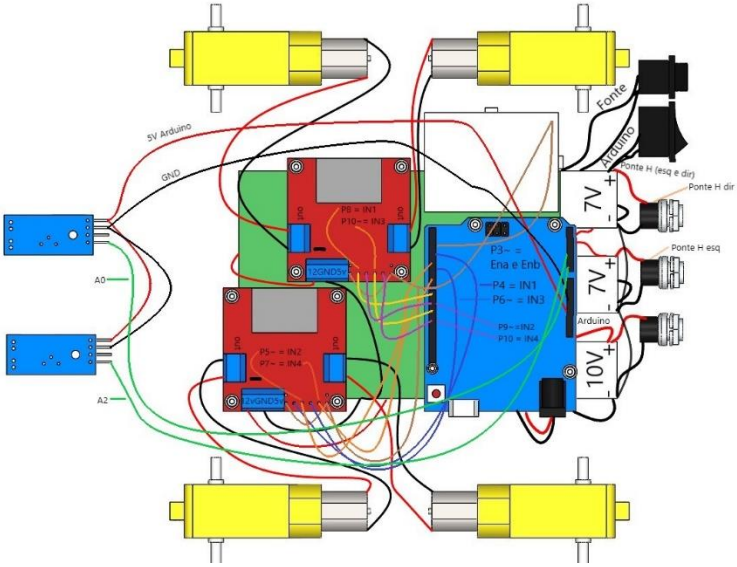
Figura 6: Montagem Final CAG

Fonte: Autor

2.3. MONTAGEM DO ESQUEMA ELETRICO DO CARRINHO AUTOGUIADO (CAG)

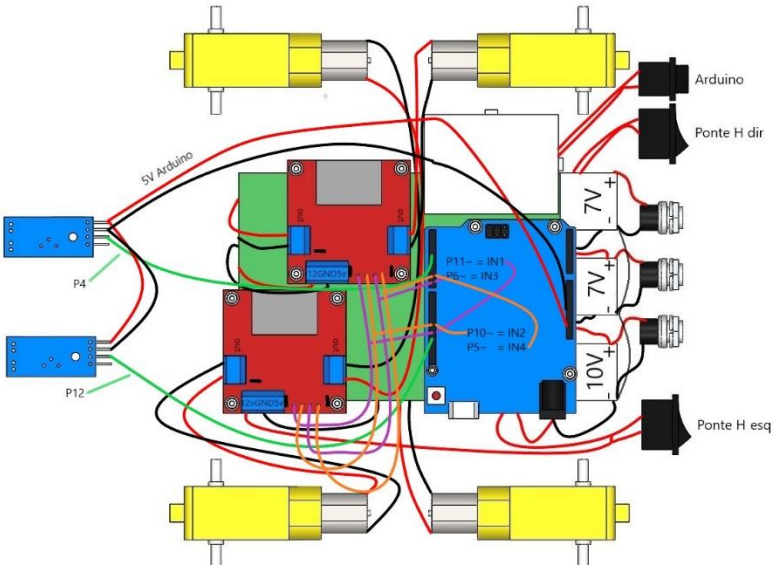
O carrinho autoguiado possuiu dois esquemas elétrico (inicial e final). O esquema elétrico inicial foi feito para primeira programação e o final foi feito para segunda programação. A figura 7 representa esquema elétrico inicial e a figura 8 representa o esquema elétrico final.

Figura 7: Esquema elétrico inicial



Fonte: Autor

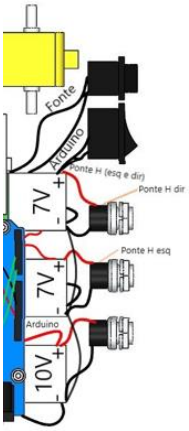
Figura 8: Esquema elétrico Final



Fonte: Autor

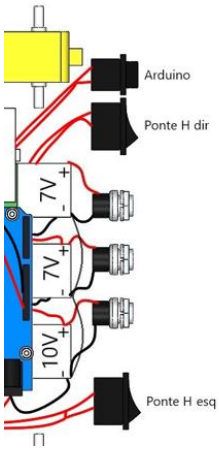
O esquema elétrico inicial tinha um problema de voltar a tensão da ponte H para o Arduino, pois os fios negativos da bateria estavam sendo ligados direto na chave (liga e desliga), sendo assim corrigido no esquema elétrico final, mudando a ligação dos fios negativo para os fios positivo na chave. A comparação de mudança está sendo representado na figura 9 (a): Ligação inicial e figura 9(b): Ligação final.

Figura 9 (a): Ligação inicial



Fonte: Autor

Figura 9 (b): Ligação final



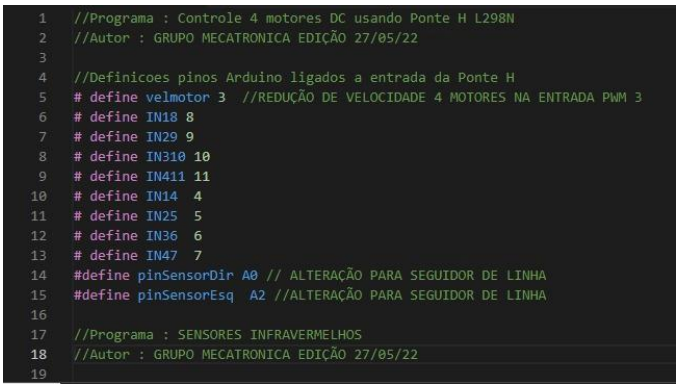
Fonte: Autor

2.4. PROGRAMAÇÃO DO CARRINHO AUTOGUIADO (CAG)

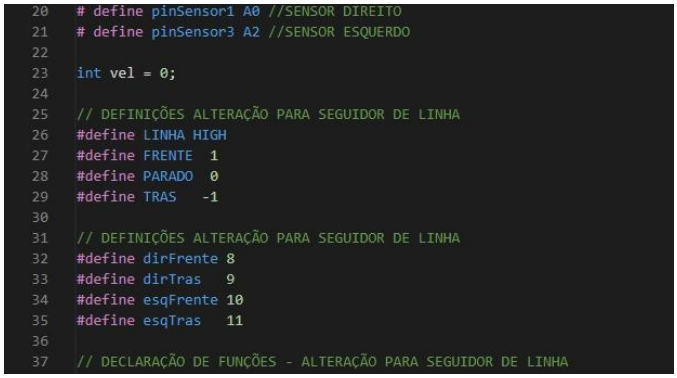
A programação do carrinho autoguiado foi realizada no programa Arduino IDE 1.8.19[14], e possuiu duas programações (inicial e final). (BRINCANDOCOMIDEIAS). A programação inicial apresentou 50% do resultado, pois as vezes o carrinho fazia o trajeto e as vezes ele passava o trajeto nas curvas. A última edição da programação inicial ocorreu no dia 27/05/2022. A figura 10 representa a programação inicial e a figura 11 representa o percurso do carrinho (inicial). O percurso representado na figura 11 foi feito por linha de fita isolante.

[14] O software Arduino (IDE) de código aberto facilita a escrita de código e o upload para a placa. Este software pode ser usado com qualquer placa Arduino. (ARDUINO).

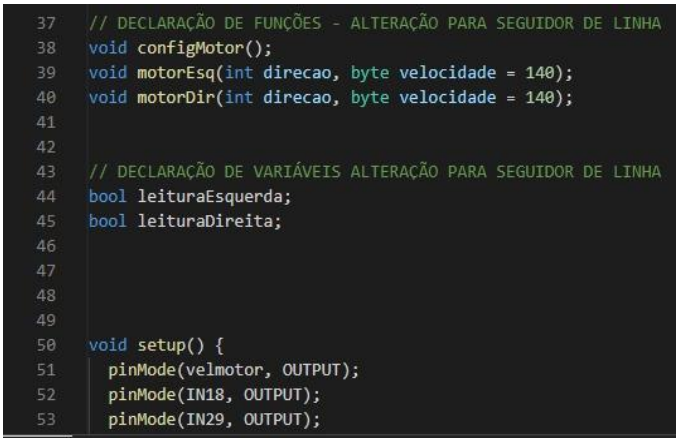
Figura 10: Programação inicial



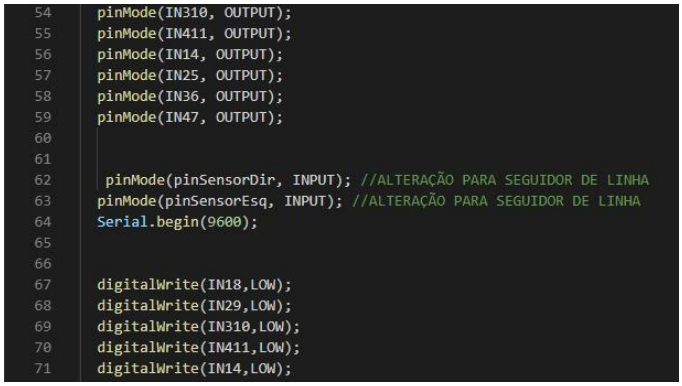
```
//Programa : Controle 4 motores DC usando Ponte H L298N
//Autor : GRUPO MECATRONICA EDIÇÃO 27/05/22

//Definicoes pinos Arduino ligados a entrada da Ponte H
# define velmotor 3  //REDUÇÃO DE VELOCIDADE 4 MOTORES NA ENTRADA PWM 3
# define IN18 8
# define IN29 9
# define IN310 10
# define IN411 11
# define IN14  4
# define IN25  5
# define IN36  6
# define IN47  7
#define pinSensorDir A0 // ALTERAÇÃO PARA SEGUIDOR DE LINHA
#define pinSensorEsq  A2 //ALTERAÇÃO PARA SEGUIDOR DE LINHA

//Programa : SENSORES INFRAVERMELHOS
//Autor : GRUPO MECATRONICA EDIÇÃO 27/05/22
```



```
# define pinSensor1 A0 //SENSOR DIREITO
# define pinSensor3 A2 //SENSOR ESQUERDO

int vel = 0;

// DEFINIÇÕES ALTERAÇÃO PARA SEGUIDOR DE LINHA
#define LINHA HIGH
#define FRENTE  1
#define PARADO  0
#define TRAS   -1

// DEFINIÇÕES ALTERAÇÃO PARA SEGUIDOR DE LINHA
#define dirFrente 8
#define dirTras   9
#define esqFrente 10
#define esqTras   11

// DECLARAÇÃO DE FUNÇÕES - ALTERAÇÃO PARA SEGUIDOR DE LINHA
```



```
// DECLARAÇÃO DE FUNÇÕES - ALTERAÇÃO PARA SEGUIDOR DE LINHA
void configMotor();
void motorEsq(int direcao, byte velocidade = 140);
void motorDir(int direcao, byte velocidade = 140);


// DECLARAÇÃO DE VARIÁVEIS ALTERAÇÃO PARA SEGUIDOR DE LINHA
bool leituraEsquerda;
bool leituraDireita;




void setup() {
  pinMode(velmotor, OUTPUT);
  pinMode(IN18, OUTPUT);
  pinMode(IN29, OUTPUT);
```



```
  pinMode(IN310, OUTPUT);
  pinMode(IN411, OUTPUT);
  pinMode(IN14, OUTPUT);
  pinMode(IN25, OUTPUT);
  pinMode(IN36, OUTPUT);
  pinMode(IN47, OUTPUT);


   pinMode(pinSensorDir, INPUT); //ALTERAÇÃO PARA SEGUIDOR DE LINHA
  pinMode(pinSensorEsq, INPUT); //ALTERAÇÃO PARA SEGUIDOR DE LINHA
  Serial.begin(9600);


  digitalWrite(IN18,LOW);
  digitalWrite(IN29,LOW);
  digitalWrite(IN310,LOW);
  digitalWrite(IN411,LOW);
  digitalWrite(IN14,LOW);
```

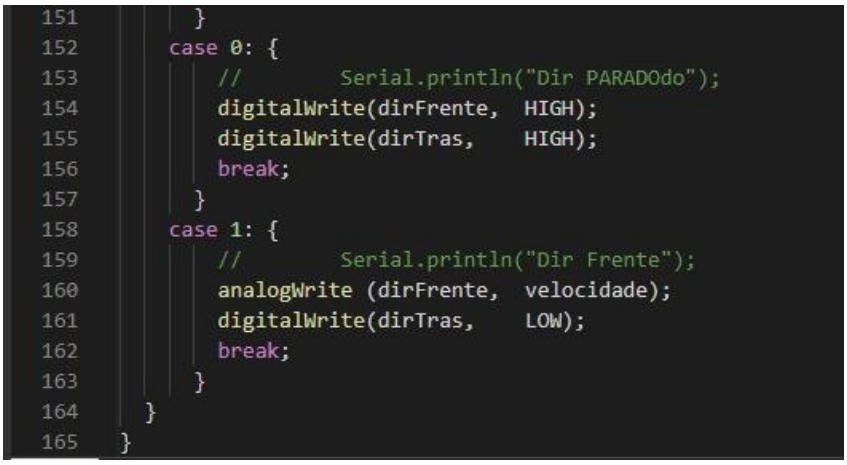
```
  digitalWrite(IN25,LOW);
  digitalWrite(IN36,LOW);
  digitalWrite(IN47,LOW);
  //analogWrite(velmotor,vel);

//ALTERAÇÃO PARA SEGUIDOR DE LINHA
  configMotor();
}

void loop() {
  bool valE = digitalRead(pinSensorEsq);
  bool valD = digitalRead(pinSensorDir);

  if (valE == LINHA && valD == LINHA) {
    motorEsq(PARADO);
    motorDir(PARADO);
```

```
      delay(300);
      motorEsq(TRAS);
      motorDir(TRAS);
      delay(150);
      motorEsq(PARADO);
      motorDir(PARADO);
      delay(3000);
    } else if (valD == LINHA) {
      motorEsq(FRENTE, 100);
      motorDir(TRAS, 100);
    } else if (valE == LINHA) {
      motorEsq(TRAS, 100);
      motorDir(FRENTE, 100);
    } else {
      motorEsq(FRENTE);
      motorDir(FRENTE);
```

```
  }
}

// IMPLEMENTO DE FUNÇÕES

void configMotor() {
  pinMode(dirFrente,  OUTPUT);
  pinMode(dirTras,    OUTPUT);
  pinMode(esqFrente,  OUTPUT);
  pinMode(esqTras,    OUTPUT);

  digitalWrite(dirFrente,  LOW);
  digitalWrite(dirTras,    LOW);
  digitalWrite(esqFrente,  LOW);
  digitalWrite(esqTras,    LOW);
}
```

```
120
121 void motorEsq(int direcao, byte velocidade = 140) {
122   switch (direcao) {
123     case -1: {
124         //        Serial.println("Esq Trás");
125         digitalWrite(esqFrente,  LOW);
126         analogWrite (esqTras,    velocidade);
127         break;
128       }
129     case 0: {
130         //        Serial.println("Esq PARADOdo");
131         digitalWrite(esqFrente,  HIGH);
132         digitalWrite(esqTras,    HIGH);
133         break;
```

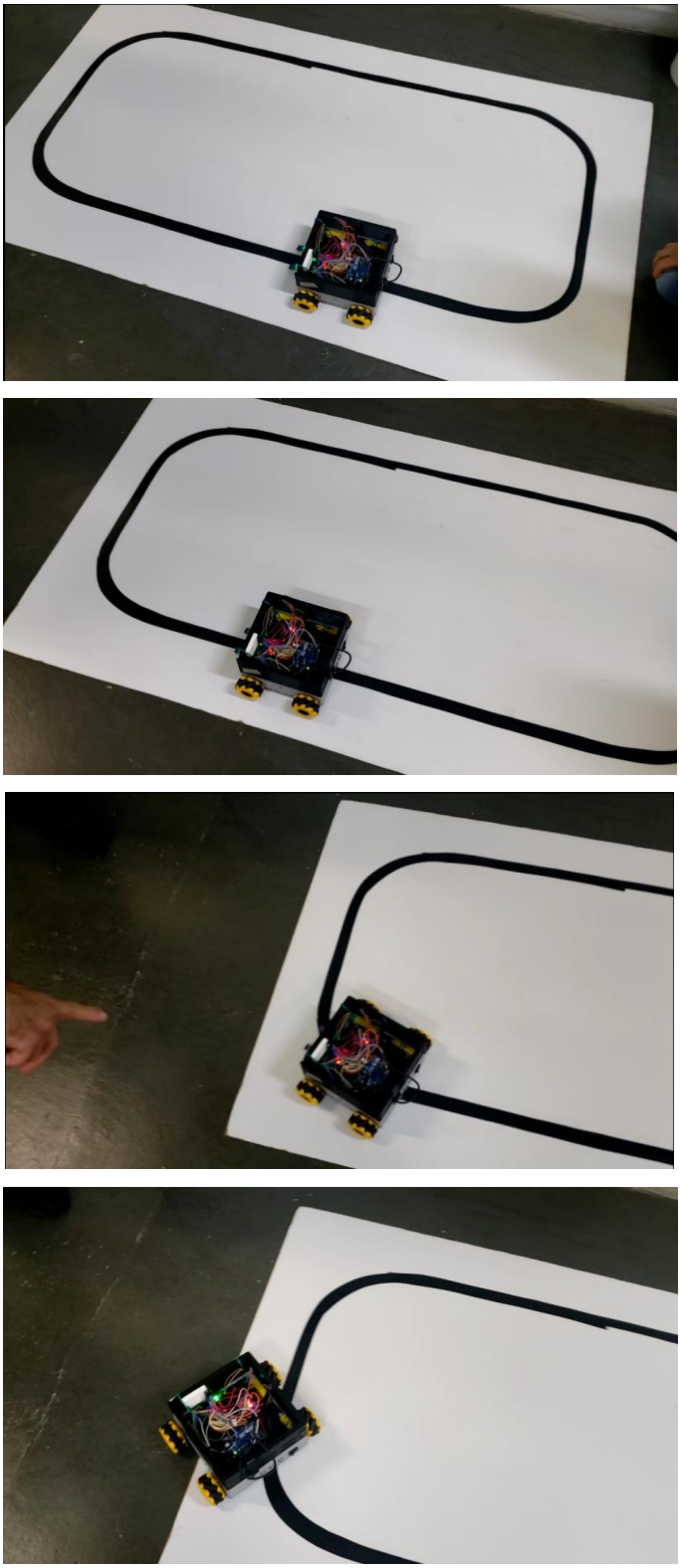
```
134       }
135     case 1: {
136         //        Serial.println("Esq Frente");
137         analogWrite (esqFrente,  velocidade);
138         digitalWrite(esqTras,    LOW);
139         break;
140       }
141   }
142 }
143
144 void motorDir(int direcao, byte velocidade = 140) {
145   switch (direcao) {
146     case -1: {
147         //        Serial.println("Dir Trás");
148         digitalWrite(dirFrente,  LOW);
149         analogWrite (dirTras,    velocidade);
150         break;
```



```
151       }
152     case 0: {
153         //        Serial.println("Dir PARADOdo");
154         digitalWrite(dirFrente,  HIGH);
155         digitalWrite(dirTras,    HIGH);
156         break;
157       }
158     case 1: {
159         //        Serial.println("Dir Frente");
160         analogWrite (dirFrente,  velocidade);
161         digitalWrite(dirTras,    LOW);
162         break;
163       }
164   }
165 }
```

Fonte: Autor

Figura 11: Percurso do carrinho (inicial)



Fonte: Autor

A programação final é uma redução da programação inicial, com umas alterações na pinagem da saída digital do Arduino. Essa programação apresentou

100% do resultado, pois o carrinho conseguiu fazer o percurso total. A última edição da programação final ocorreu no dia 16/06/2022. A figura 12 representa a programação final e a figura 13 representa o percurso do carrinho (final). O percurso representado na figura 13 foi feito por uma linha de tinta preta fosca, pois reduz o brilho ambiente e evitando interferência nos sensores ópticos.

Figura 12: Programação final

```
/*
    AUTOR:      GRUPO CAG
    SKETCH:     PROGRAMAÇÃO GRUPO CAG
    DATA:       16/06/2022
*/

// DEFINIÇÕES DE PINOS
#define pinSensorDir 4
#define pinSensorEsq 12

#define dirFrente 6
#define dirTras   5
#define esqFrente 10
#define esqTras   11

// DEFINIÇÕES
#define LINHA HIGH

#define FRENTE  1
#define PARADO  0
#define TRAS   -1

// DECLARAÇÃO DE FUNÇÕES
void configMotor();
void motorEsq(int direcao, byte velocidade = 80);
void motorDir(int direcao, byte velocidade = 95);

// DECLARAÇÃO DE VARIÁVEIS
bool leituraEsquerda;
bool leituraDireita;

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

  pinMode(pinSensorDir, INPUT);
  pinMode(pinSensorEsq, INPUT);

  configMotor();
}

void loop() {
  bool valE = digitalRead(pinSensorEsq);
  bool valD = digitalRead(pinSensorDir);

  if (valE == LINHA && valD == LINHA) {
    motorEsq(PARADO);
    motorDir(PARADO);
    delay(300);
```

```
48        delay(300);
49        motorEsq(TRAS);
50        motorDir(TRAS);
51        delay(150);
52        motorEsq(PARADO);
53        motorDir(PARADO);
54        delay(3000);
55      } else if (valD == LINHA) {
56        motorEsq(FRENTE, 100);
57        motorDir(TRAS, 100);
58      } else if (valE == LINHA) {
59        motorEsq(TRAS, 100);
60        motorDir(FRENTE, 100);
61      } else {
62        motorEsq(FRENTE);
63        motorDir(FRENTE);
64      }
```

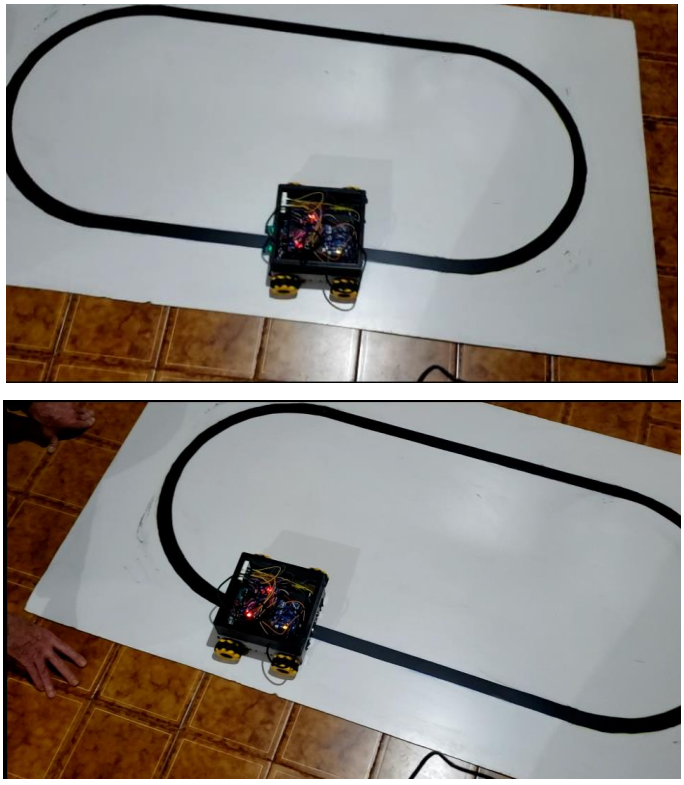
```
67  // IMPLEMENTO DE FUNÇÕES
68
69  void configMotor() {
70    pinMode(dirFrente,  OUTPUT);
71    pinMode(dirTras,    OUTPUT);
72    pinMode(esqFrente,  OUTPUT);
73    pinMode(esqTras,    OUTPUT);
74
75    digitalWrite(dirFrente,  LOW);
76    digitalWrite(dirTras,    LOW);
77    digitalWrite(esqFrente,  LOW);
78    digitalWrite(esqTras,    LOW);
79  }
```

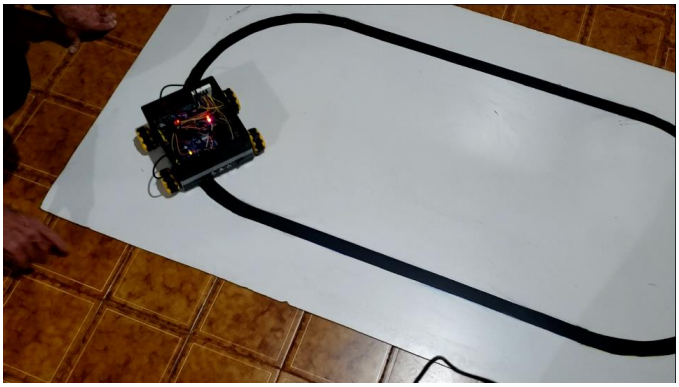
```
81  void motorEsq(int direcao, byte velocidade = 80) {
82    switch (direcao) {
83      case -1: {
84          //        Serial.println("Esq Trás");
85          digitalWrite(esqFrente,  LOW);
86          analogWrite (esqTras,    velocidade);
87          break;
88        }
89      case 0: {
90          //        Serial.println("Esq PARADOdo");
91          digitalWrite(esqFrente,  HIGH);
92          digitalWrite(esqTras,    HIGH);
93          break;
94        }
95      case 1: {
```

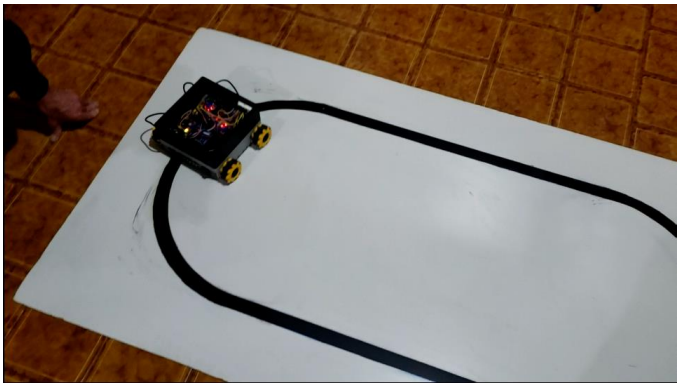
```
95      case 1: {
96          //        Serial.println("Esq Frente");
97          analogWrite (esqFrente,  velocidade);
98          digitalWrite(esqTras,    LOW);
99          break;
100       }
101   }
102 }
103
104 void motorDir(int direcao, byte velocidade = 95) {
105   switch (direcao) {
106     case -1: {
107         //        Serial.println("Dir Trás");
108         digitalWrite(dirFrente,  LOW);
109         analogWrite (dirTras,    velocidade);
110         break;
111       }
111       }
112     case 0: {
113         //        Serial.println("Dir PARADOdo");
114         digitalWrite(dirFrente,  HIGH);
115         digitalWrite(dirTras,    HIGH);
116         break;
117       }
118     case 1: {
119         //        Serial.println("Dir Frente");
120         analogWrite (dirFrente,  velocidade);
121         digitalWrite(dirTras,    LOW);
122         break;
123       }
124   }
125 }
126
```

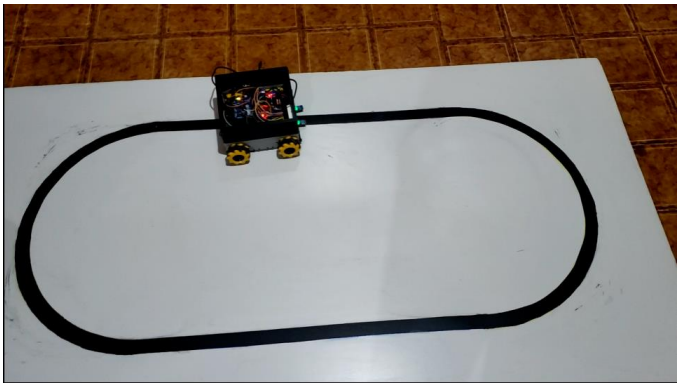
Fonte: Autor

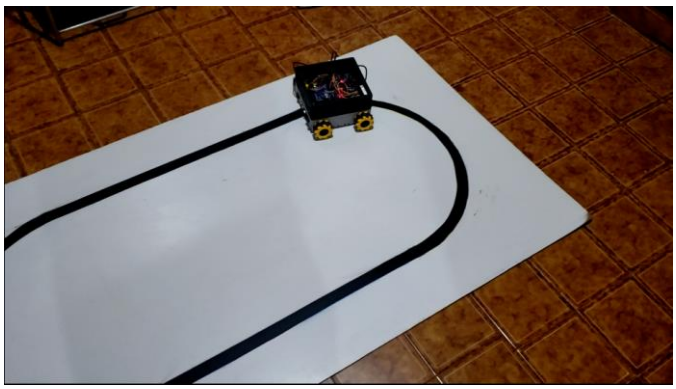
Figura 13: Percurso do carrinho (Final)

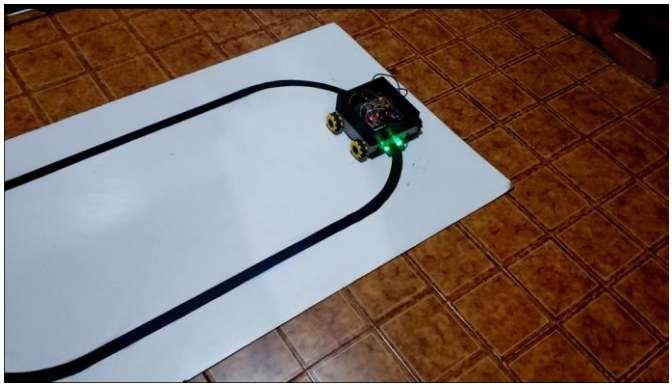


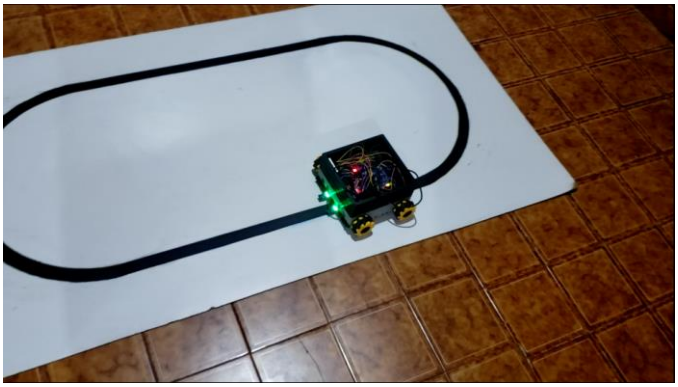












Fonte: Autor