ISBN: 978-65-86901-74-0

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

# TUTORIAL DE CONFECÇÃO DE UM ROBÔ SEGUIDOR DE LINHA PARA INICIAÇÃO À ROBÓTICA EDUCACIONAL

Renan Corrêa Basoni
Mestre pelo Curso de Engenharia Elétrica da Universidade Federal do Espírito Santo – UFES, renan.basoni@ifc.edu.br;

Humberto Pontes van Ool de Sousa
Graduando do Curso de Tecnologia em Mecatrônica Industrial do Instituto Federal do Ceará – IFCE, humberto.pontes12@gmail.com;

César Augusto Victor
Graduando do Curso de Tecnologia em Mecatrônica Industrial do Instituto Federal do Ceará – IFCE, cesar-tri@hotmail.com;

Harrison Lucas Paula Lacerda
Graduando do Curso de Tecnologia em Mecatrônica Industrial do Instituto Federal do Ceará – IFCE, harrisonlucas81@gmail.com.

## RESUMO

Objetivando demonstrar a construção de um robô seguidor de linha, que possa ser utilizado na iniciação à Robótica Educacional, este trabalho visa tornar a prática da robótica mais acessível e presente na vida de jovens e adolescentes. Nesse sentido, esta pesquisa descreve os processos de planejamento, produção e funcionalidade de um robô seguidor de linha de baixo custo, disponibilizando os meios necessários para que o projeto seja replicado, com o intuito de incentivar a interação entre professores e estudantes por meio da robótica educacional. Estabelecendo, assim, um vínculo interdisciplinar com a matemática, física e computação, as quais, a título de exemplo, são algumas das diversas áreas que englobam a Robótica. O estudo e desenvolvimento do projeto foi realizado pelo Grupo de Extensão e Pesquisa em Robótica

ISBN: 978-65-86901-74-0

(GEPRO) do IFCE Campus Sobral. Durante os estudos realizados para o desenvolvimento deste projeto, foram considerados diversos modelos de robô seguidor de linha, propositando escolher um modelo que tivesse sua complexidade nivelada com os níveis de ensino médio, técnico e superior. Através da metodologia adotada, visualiza-se a proposta para a construção do robô seguidor linha, sendo demonstrados todos os componentes necessários para sua confecção, assim como o funcionamento de cada um dos seus componentes. Além do detalhamento da parte construtiva do projeto, através da metodologia, é possível verificar o funcionamento das principais linhas de código utilizadas – o código fonte utilizado está disponível na seção de anexo deste artigo. Analisando os resultados obtidos, observa-se que o robô seguidor de linha apontado neste trabalho é um modelo viável para a iniciação à robótica, visto que o protótipo se mostrou eficaz ao percorrer os desafios da pista de testes. Diante disso, constata-se que fomentar a Robótica Educacional possibilitará que os estudantes possam desenvolver habilidades de aprendizado prático, trabalho em equipe e resolução de problemas.

**Palavras-chave:** Robótica educacional, Robô seguidor de linha, Sistemas embarcados.

ISBN: 978-65-86901-74-0

## INTRODUÇÃO

Na última década, a robótica tem aguçado o interesse de docentes e pesquisadores como um importante recurso para o desenvolvimento cognitivo e habilidades sociais de alunos da educação infantil ao ensino médio, e no embasamento para o aprendizado de ciências, matemática, tecnologia, computação e outros saberes (CAMPOS, 2017).

Este artigo tem como propósito apresentar o processo de desenvolvimento de um robô seguidor de linha, de fácil execução e baixo custo, para que, de forma interdisciplinar, os professores possam interagir com seus alunos, dando incentivo à prática da robótica e suas tecnologias, uma vez que a robótica interage com diversas áreas do conhecimento.

Com o intuito de promover a interação entre professores e alunos, e seus respectivos cotidianos, a interdisciplinaridade pode ser trabalhada de diversas formas, de modo a facilitar a associação do conhecimento adquirido com o dia-a-dia. Ou seja, aplicada como uma proposta de trabalhar um determinado assunto abordando várias disciplinas, ampliando os horizontes dos alunos (LIMA, 2020).

## METODOLOGIA

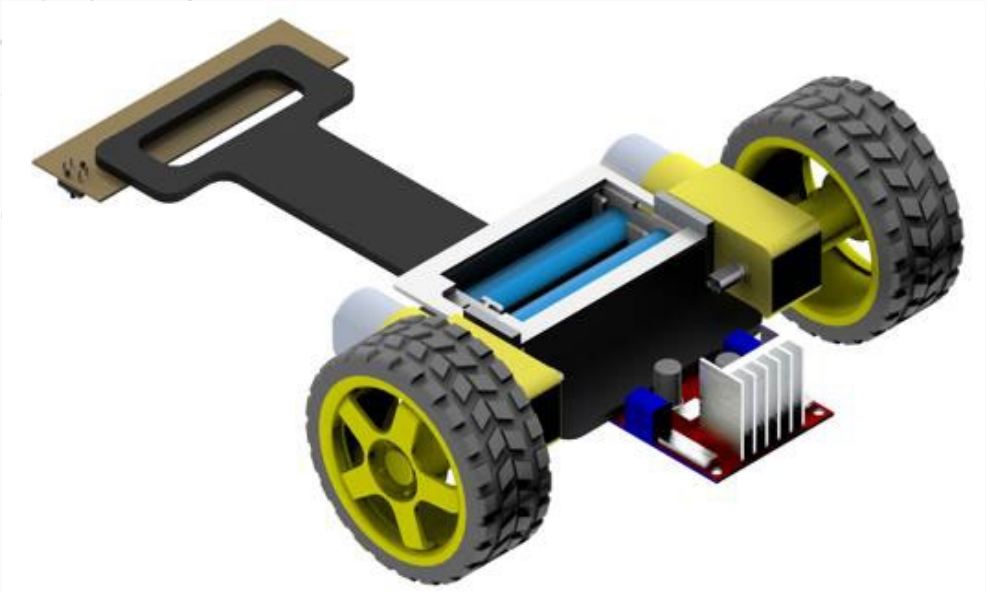
O estudo e desenvolvimento do projeto foi realizado no Laboratório de Eletrônica 2 do IFCE Campus Sobral e o seguidor de linha proposto neste artigo foi desenvolvido pelo Grupo de Extensão e Pesquisa em Robótica (GEPRO).

A motivação pelo desenvolvimento deste projeto se deu por causa da necessidade de criar um material didático relacionado com um protótipo de robô seguidor de linha que pudesse ser replicado por qualquer estudante dos níveis de ensino médio, técnico e superior (graduação). Atualmente, existem diversos modelos de robô seguidor de linha; geralmente, a complexidade de cada modelo se encaixa em pelo menos um dos três níveis de ensino citados. No entanto, para o protótipo proposto, decidiu-se desenvolver um tutorial de construção de um robô seguidor de linha que pudesse ser utilizado nos três níveis de ensino mencionados. O robô projetado pode ser visualizado através da Figura 1.

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

ISBN: 978-65-86901-74-0

Figura 1 – Seguidor de linha proposto desenvolvido no *software* Solid Edge.



Fonte: Próprio autor (2022).

O robô proposto é composto por diversos elementos de circuitos elétricos, como, por exemplo: Arduino; motores de corrente contínua; ponte H; sensores ópticos reflexivos, etc. Além dos componentes elétricos, também é necessária a utilização de um chassi. A seguir, será demonstrado cada um dos componentes que compõem o seguidor de linha.

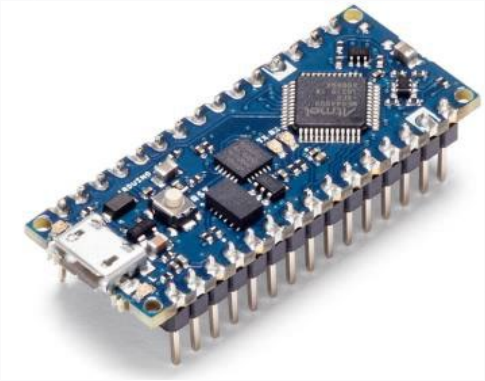
## ARDUINO

O Arduino é uma plataforma *open source* de prototipagem eletrônica, ou seja, o *hardware* é livre. Neste projeto utilizou-se o Arduino Nano, o qual é composto por 14 pinos digitais, que trabalham na tensão de 5 volts e 40 miliamperes, sendo que os pinos digitais podem ser utilizados como entradas ou saídas digitais. Além disso, o Arduino nano possui 6 portas analógicas que também podem ser utilizadas como entrada ou saída digitais.

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

ISBN: 978-65-86901-74-0

**Figura 2** – Arduino Nano.



**Fonte:** Arduino (2022).

## MOTORES DE CORRENTE CONTÍNUA

Os motores escolhidos funcionam com tensão de 6 V e são acoplados em uma caixa de redução. Os motores utilizados no projeto podem ser visualizados através da Figura 3.

**Figura 3** – Motores de corrente contínua com caixas de redução e rodas.

**Fonte:** Usinainfo (2022).

## PONTE H

Para a realização do acionamento dos motores de corrente contínua utilizou-se a ponte H L298N. Sua utilização é justificada porque o Arduino suporta uma corrente máxima por terminal na escala de miliamperes, de modo que não seria possível acionar os motores sem a utilização do L298N ou *drivers* similares.

ISBN: 978-65-86901-74-0

**Figura 4** – Ponte H L298N.

**Fonte:** FilipeFlop (2022).

## SENSOR ÓPTICO REFLEXIVO

O sensor óptico reflexivo utilizado foi o TCRT-5000. Conforme Figura 5, este sensor possui, em um mesmo dispositivo, um LED infravermelho (emissor) e um fototransistor (receptor).

**Figura 5** – Sensor Óptico Reflexivo TCRT-5000.

**Fonte:** Casa da Robótica (2022).

## BATERIA DE LÍTIO

A alimentação do seguidor de linha foi realizada através da utilização de duas baterias de lítio conectadas em série. Cada bateria tem 3,7 V e uma capacidade de 2600 mAh.

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

ISBN: 978-65-86901-74-0

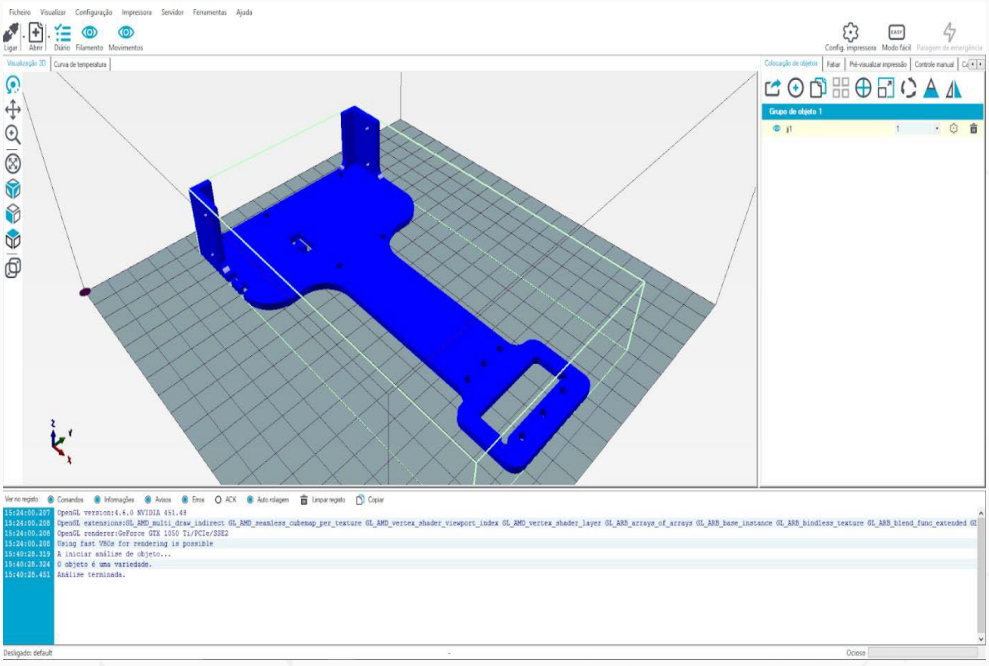
**Figura 6** – Bateria de lítio.


**Fonte:** Casa da Robótica (2022).

## CHASSI

O Chassi foi desenvolvido através da utilização de uma impressora 3D. O filamento utilizado foi o ABS de 1,75 mm. Para a impressão do Chassi foi utilizado o *software* Repetier-Host. O arquivo utilizado para impressão pode ser encontrado em (Andrew Linden, 2018).

**Figura 7** – *Software* Repetier-Host.



**Fonte:** Próprio autor (2022).

ISBN: 978-65-86901-74-0

Através da Tabela 1, é possível visualizar o preço médio dos componentes eletrônicos utilizados para o desenvolvimento do robô seguidor de linha.

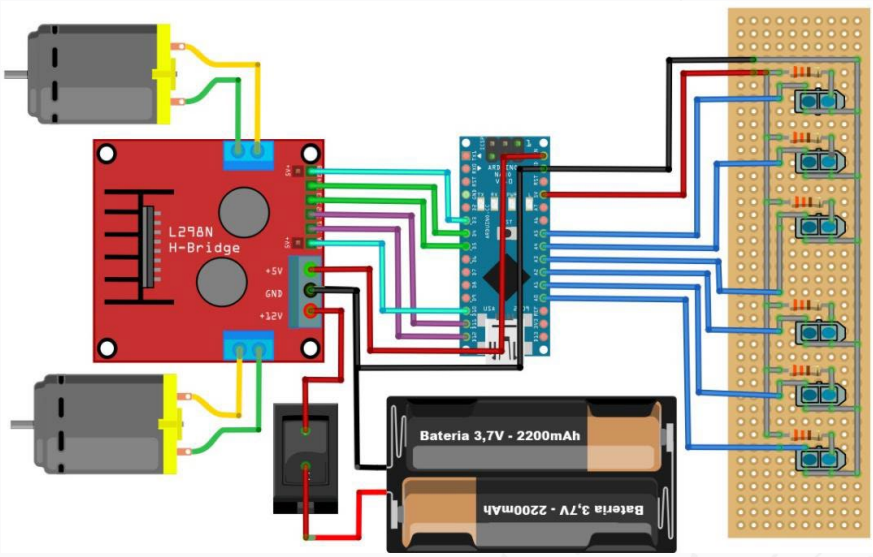
**Tabela 1** – Lista de componentes.

| Componente | Quantidade | Preço médio (R$) |
|---|---|---|
| Arduino Nano | 1 | 39,90 |
| L298N | 1 | 18,50 |
| Sensor óptico TCRT-5000 | 6 | 10,70 |
| Resistor 330 Ω e ¼ W | 6 | 0,50 |
| Bateria de lítio de 3,7 V e 2200 mAh | 2 | 22,00 |
| Roda + Motor | 2 | 36,00 |
| Chave gangorra com 2 terminais | 1 | 1,80 |
| Placa ilhada 10x5 cm | 1 | 6,90 |
| Suporte para 2 baterias de 3,7 V | 1 | 12,90 |
| **Total** | **21** | **149,20** |

**Fonte:** Próprio autor (2022).

A Figura 8 demonstra o esquemático do robô seguidor de linha proposto, bem como a utilização de todos os componentes eletrônicos listados na Tabela 1.

**Figura 8** – Esquemático do robô seguidor de linha proposto.



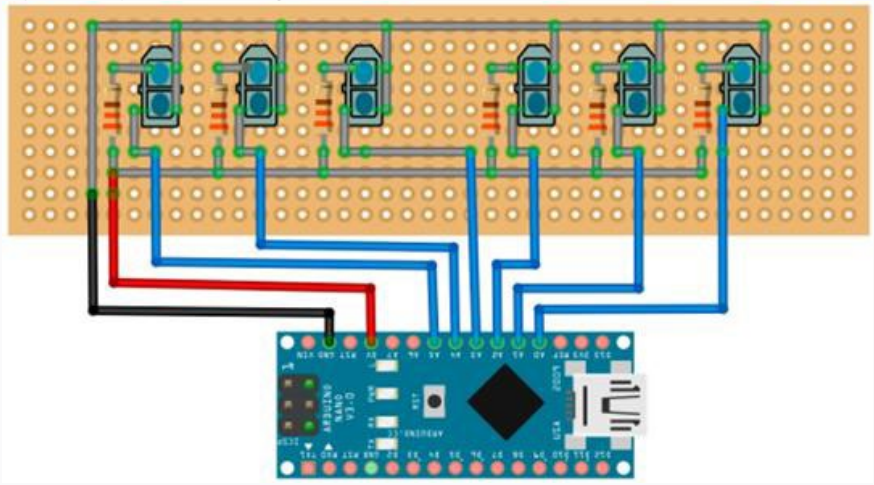

**Fonte:** Próprio autor (2022).

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

ISBN: 978-65-86901-74-0

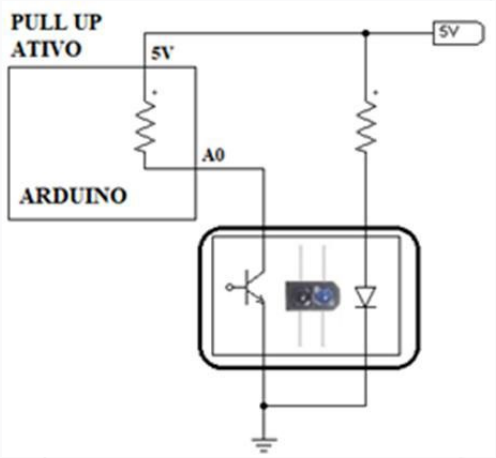
Para a confecção da placa de sensores deve ser utilizado uma placa ilhada que possua 10 x 5cm, podendo ser reduzida para fins estéticos. Os sensores e resistores devem ser soldados à placa como mostra o esquema no *Software* Fritzing na Figura 9, atentando-se à polaridade dos sensores, conforme é visualizado na Figura 10.

**Figura 9** – Ligação elétrica entre os sensores ópticos reflexivos e o Arduino Nano.



**Fonte:** Próprio autor (2022).

**Figura 10** – Alimentação do TCRT-5000.




**Fonte:** Adaptado de SQUIDS Arduino e de Casa da Robótica pelo próprio autor (2022).

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

ISBN: 978-65-86901-74-0

O seguidor de linha é composto por 6 sensores; cada um deles transmite ao Arduino um valor entre 0-1023, e este valor será proporcional à absorção de luz do sensor receptor. O Arduino utiliza o valor de 0-1023 (conversão analógica/digital) para acionar os motores através da modulação por largura de pulso (PWM). Levando em consideração a utilização de seis sensores, existirão 64 possibilidades de detecção de linha. Sendo que cada possibilidade irá gerar um sinal PWM para cada um dos motores, e este sinal será baseado no valor de erro de posição entre o robô e a linha. Mais detalhes podem ser visualizados na Tabela 2.

No caso em que os *status* dos sensores forem 000000 (totalmente fora da linha) o programa detectará quais foram os últimos sensores acionados. Desta forma, o microcontrolador será capaz de estabelecer se o robô saiu pela direita ou pela esquerda da linha, e, a partir disso, poderá estabelecer aos motores as mesmas condições de quando os status dos sensores forem 000001 ou 100000. E nos demais status, que não foram demonstrados na Tabela 2, como é o caso de 100001, o programa utilizará o último valor válido capturado, já que os status como 100001 não tem significado físico para o seguidor de linha. Mais detalhes podem ser visualizados na Tabela 2.

**Tabela 2** – Sinal PWM gerado a partir dos principais status dos sensores ópticos reflexivos.

| Status dos Sensores | PWM motor direito | PWM motor esquerdo |
|---|---|---|
| **100000** | 100% | 0% |
| **110000** | 100% | 71% |
| **010000** | 100% | 78% |
| **001000** | 100% | 94% |
| **000001** | 0% | 100% |
| **000010** | 100% | 78% |
| **000100** | 94% | 100% |
| **001100** | 100% | 100% |
| **000110** | 84% | 100% |
| **000011** | 71% | 100% |
| **111000** | 100% | 0% |

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

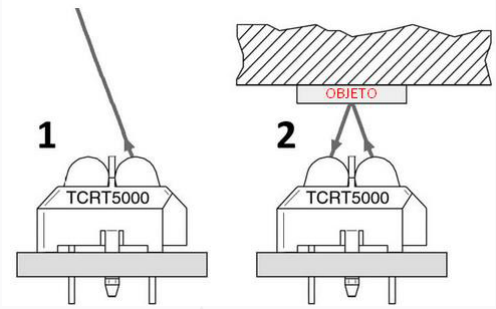
ISBN: 978-65-86901-74-0

| Status dos Sensores | PWM motor direito | PWM motor esquerdo |
|---|---|---|
| **011100** | 100% | 71% |
| **000111** | 100% | 0% |
| **001110** | 71% | 100% |
| **111100** | 100% | 00% |
| **001111** | 0% | 100% |
| **111110** | 100% | 0% |
| **011111** | 0% | 100% |

**Fonte:** Próprio autor (2022).

A Figura 11 exemplifica o funcionamento dos sensores. Considerando a aplicação dos sensores ópticos reflexivos, o "Objeto" da Figura 11 equivale a linha da pista de competição dos seguidores de linha.

**Figura 11 –** Funcionamento do TCRT-5000.




**Fonte:** SQUIDS Arduino (2022).

Para evitar mau contato entre as ligações do circuito recomenda-se que os *jumpers* sejam soldados diretamente nas ilhas. A Figura 12 exemplifica o correto processo de soldagem.

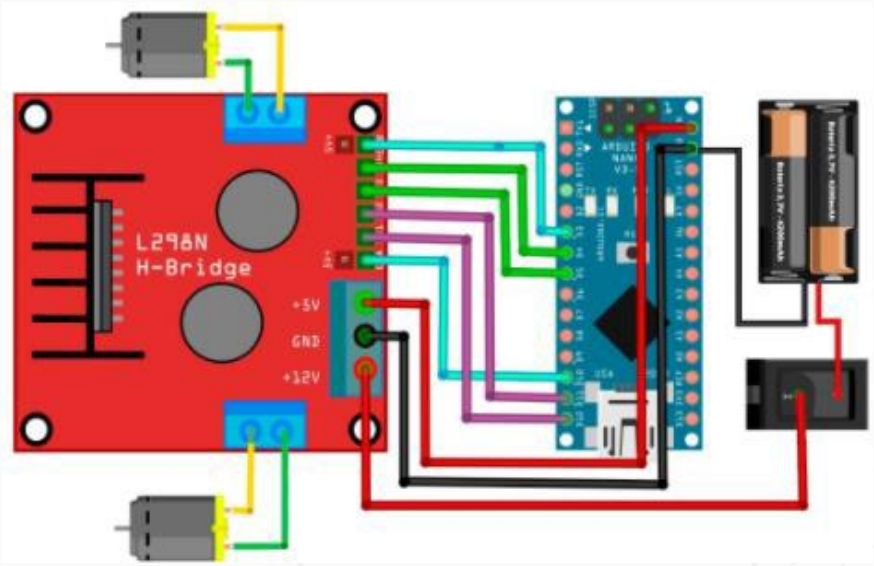
ISBN: 978-65-86901-74-0

Figura 12 – Tutorial de soldagem de componentes eletrônicos.


Fonte: Tarzan Soluções em Eletrônica pelo próprio autor (2022).

Conforme dito, escolheu-se o driver L298N para realizar o acionamento dos motores. A Tabela 3 exibe as principais características do *driver* L298N. Através da Figura 13 e da Tabela 3 é possível visualizar as conexões entre o Arduino Nano e o driver L298N.

Figura 13 – Conexões entre o Arduino Nano e o *driver* L298N.




Fonte: Próprio autor (2022).

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

ISBN: 978-65-86901-74-0

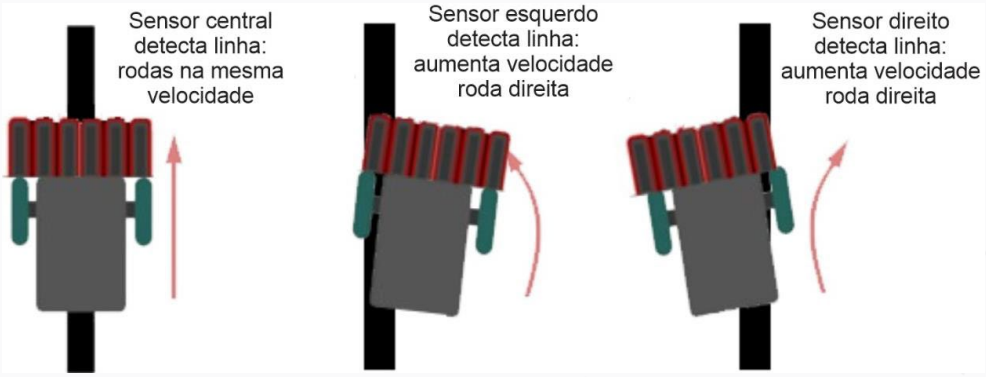
**Tabela 3** – Conexões entre o Arduino Nano e o *driver* L298N.

| L298N | Arduino Nano |
|---|---|
| ENA | D10 (PWM) |
| IN1 | D12 |
| IN2 | D11 |
| IN3 | D5 |
| IN4 | D4 |
| ENB | D3 (PWM) |

**Fonte:** Próprio autor (2022).

A velocidade dos motores será máxima quando os dois sensores do centro estiverem sobre a linha; a velocidade do motor direito será máxima quando somente o sensor mais à esquerda estiver acionado, e a velocidade do motor esquerdo será máxima quando somente o sensor mais à direita estiver acionado. Mais detalhes podem ser visualizados na Figura 14.

**Figura 14** – Funcionamento dos sensores e dos motores.

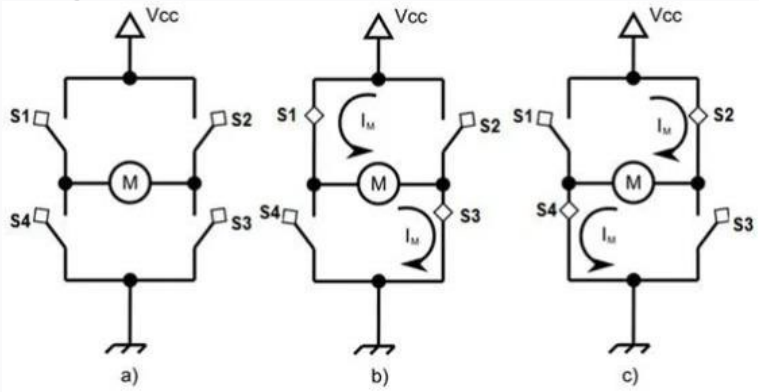



**Fonte:** Adaptado de FilipeFlop (2022).

Através da Figura 15 é possível visualizar três condições de funcionamento de uma ponte H. Na condição (a) não circula corrente elétrica pelo motor, pois as chaves S1, S2, S3 e S4 estão abertas, e, por fim, nas condições (b) e (c), o motor irá girar respectivamente nos sentidos horário e anti-horário.

ISBN: 978-65-86901-74-0

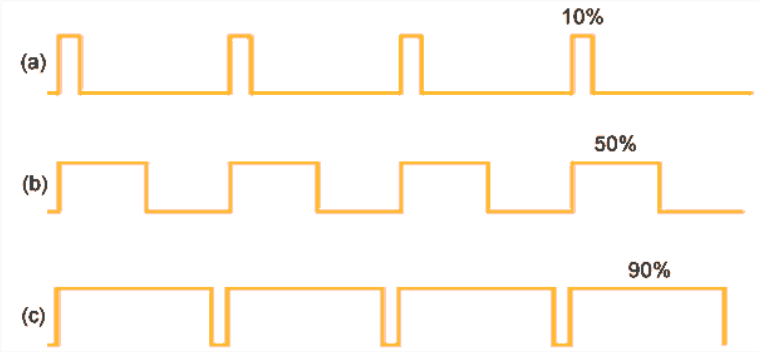
Figura 15 – Funcionamento da ponte H.




Fonte: Arduino Portugal (2022).

A forma de acionamento dos motores é através do PWM (modulação por largura de pulso). A Figura 16 exibe 3 valores diferentes de razão cíclica, 10%, 50% e 90%. É possível perceber que quanto maior for a razão cíclica, maior será o valor da tensão média de saída dos terminais do Arduino configurados como PWM, assim, maior será a velocidade dos motores.

Figura 16 – Modulação por largura de pulso.




Fonte: Citisystems (2022).

A razão cíclica dos sinais PWM que acionam os dois motores é definida com base nos status dos seis sensores ópticos reflexivos, conforme a Tabela 2. A razão cíclica do PWM pode variar numa faixa entre 0 a 100%. No Arduino utilizado, o PWM é de 8 bits, portanto a variação ocorre numa faixa entre 0 a 255.

ISBN: 978-65-86901-74-0

O conversor analógico digital do Arduino possui uma resolução de 10 bits, isso significa que a leitura dos sensores será interpretada em uma faixa entre 0 a 1023. Nesse sentido, com 100% de reflexão no receptor, a leitura contará o valor máximo (1023), e com 0% de reflexão, ela contará o valor mínimo (zero). No código do seguidor de linha proposto – conforme demonstrado na Figura 17 –, a variável *treashold* (limite) servirá para converter leituras analógicas em digitais; valores obtidos abaixo do valor setado em *treashold* serão entendidos como zero e valores obtidos acima serão entendidos como 1.

**Figura 17 –** Manipulação da variável *treashold*.

```
void Scan()
{
  cont = 0;
  irSensors = B000000;
  for (int i = 0; i < quantSensores; i++)
  {
   irSensorDigital[i] = (analogRead(irPins[i]) >= treashold);
   cont = cont + irSensorDigital[i];
   int b = (quantSensores - 1 ) - i;
   irSensors = irSensors + (irSensorDigital[i] << b);
  }
}
```

**Fonte:** Próprio autor (2022).

No trecho do código exibido através da Figura 17 pode-se observar a conversão analógica/digital como descrito anteriormente. A utilização da variável *treashold* permite o controle da sensibilidade do sensor via *software*. Essa condição é muito importante, considerando que diferentes ambientes podem ter diferentes índices de iluminação. Um vetor de 6 posições com valores 0 ou 1 correspondentes aos status de cada um dos 6 sensores é obtido após a conversão analógica/digital realizada a partir do valor setado para a variável *treashold*.

Para cada um dos vetores construídos, será atribuído um valor de erro (Figura 18), este erro assume valores entre -180 a 180. Para valores positivos de erro, o robô curvará para a esquerda e, para valores negativos de erro, o robô curvará para a direita.

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

ISBN: 978-65-86901-74-0

Figura 18 – Manipulação dos erros de posição.

```
void UpdateCorrection()
{
 if(error >= 0 && error < 30)
 {
  correction = 0;
 }
 else
    if(error >= 30 && error < 60)
    {
     correction = 15;
    }
    else
       if(error >= 60 && error < 90)
       {
        correction = 40;
       }
       else
          if(error >= 90 && error < 120)
          {
           correction = 55;
          }
          else
             if(error >= 120 && error < 150)
             {
              correction = 75;
             }
             else
                if(error >= 150 && error < 180)
                {
                 correction = 255;
                }
                else
                   if(error >= 180)
                   {
                    correction = 305;
                   }
}
```

Fonte: Próprio autor (2022).

Os erros estabelecidos através da Figura 18 são convertidos em valores de razão cíclica dos PWM's que atuam em cada um dos motores, conforme pode ser visualizado através do código fonte disponibilizado no Anexo 1 deste artigo.

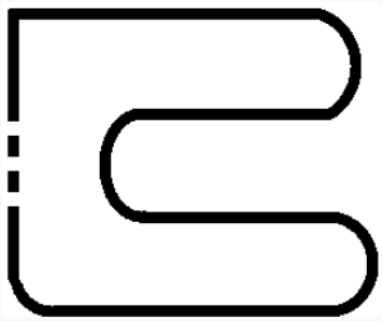
## RESULTADOS E DISCUSSÃO

Para efetuar os testes do robô seguidor de linha, podem ser utilizadas pistas de testes construídas em superfícies brancas e linhas pretas. As linhas pretas podem ser feitas com fita isolante. A superfície onde deve ser feita a pista deve ter uma boa aderência para garantir que o robô não derrape e/ou patine. Conforme o necessário, com base no índice de iluminação do ambiente, deve-se

ISBN: 978-65-86901-74-0

ajustar o valor da variável *treashold* no código fonte para garantir um melhor desempenho. A Figura 19 mostra um modelo básico de pista com a linha medindo 2,5 cm de espessura, no entanto em competições esta espessura pode variar.

**Figura 19** – Modelo de pista sugerida para a realização de testes iniciais do seguidor de linha.

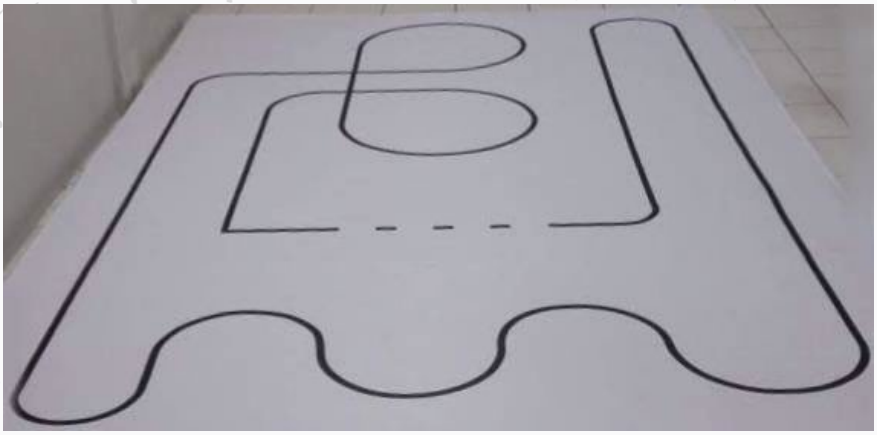


**Fonte:** Próprio autor (2022).

Para validar os resultados finais do projeto, foi utilizada a pista da Figura 20, pode-se perceber que esta pista é mais desafiadora do que a pista sugerida para testes iniciais. O seguidor de linha desenvolvido percorreu todo o trajeto da pista sem falhas, tanto na parte tracejada da linha quanto na curva de 90°, em um tempo médio de 45 segundos. Deste modo, ficou comprovado que o seguidor de linha proposto está apto a percorrer pistas com partes da linha tracejadas e curvas em 90°. Como projeto futuro, os estudantes podem modificar o código e pensar em pistas com novos desafios, como por exemplo, adicionar partes da pista com zigue-zague e etc.

ISBN: 978-65-86901-74-0

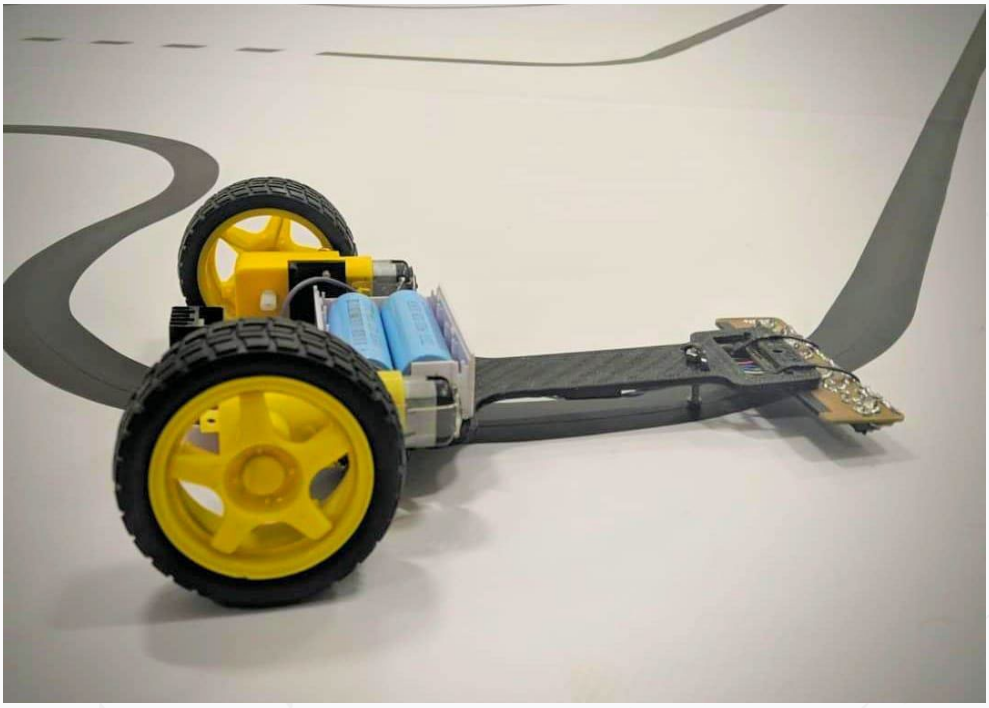
Figura 20 – Pista utilizada na realização de testes do seguidor de linha.



Fonte: Próprio autor (2022).

Após a conclusão dos testes, pode-se afirmar que o robô seguidor de linha desenvolvido foi finalizado com êxito. A Figura 21 exibe o robô seguidor de linha desenvolvido.

Figura 21 – Robô seguidor de linha desenvolvido.



Fonte: Próprio autor (2022).

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

ISBN: 978-65-86901-74-0

## CONSIDERAÇÕES FINAIS

Através do projeto proposto é possível quebrar paradigmas, mostrando que a robótica pode ser de fato utilizada para fins didáticos, de modo que não pareça ser tão assustadora e complexa, criando deste modo um vínculo maior entre os alunos e as áreas das ciências exatas, esta que é uma área que normalmente, devido a sua complexidade, não desperta tanto interesse dos alunos, por isso é fundamental a utilização de ferramentas lúdicas aplicadas na construção de meios e mecanismos que aproximem os alunos das ciências exatas. O propósito deste trabalho é a simplificação de forma intuitiva da construção de robôs seguidores de linha.

## REFERÊNCIAS

42 Bots. **Project 2: Arduino Line Following teste 3.** Disponível em: ‹https://42bots.com/competitions/arduino-line-following-code-video/›. Acesso em: 21 de fevereiro de 2021.

Andrew Linden. **Robô Seguidor de Linha**. Disponível em: ‹https://www.thingiverse.com/thing:2831729/files›. Acesso em: 09 de setembro de 2022.

Arduino. **Arduino Nano**. Disponível em: ‹https://store-usa.arduino.cc/products/arduino-nano-every-with-headers?selectedStore=us›. Acesso em: 14 de julho de 2022.

Arduino Portugal. **Módulo Ponte H L298N – O primeiro passo para montar seu robô com Arduino**. Disponível em: ‹https://www.arduinoportugal.pt/modulo-ponte-h-l298n-primeiro-passo-montar-robo-arduino/›. Acesso em: 12 de junho de 2022.

CAMPOS, Flavio Rodrigues. **Robótica Educacional no Brasil: Questões em Aberto, Desafios e Perspectivas Futuras**. Revista Ibero-Americana de Estudos em Educação, Araraquara, v.12, n.4, p. 2108-2121, out./dez. 2017.

ISBN: 978-65-86901-74-0

Casa da Robótica. **100x Peças de Sensor Reflexivo IR TCRT5000 TCRT.** Disponível em: <https://www.casadarobotica.com/sensores-modulos/sensores/movimento-proximidade/100x-pecas-de-sensor-refletivo-ir-tcrt5000-tcrt>. Acesso em 21 de janeiro de 2021.

Casa da Robótica. **1x Bateria 3,7V 18650 de Lítio Recarregável Gimex.** Disponível em: <https://www.casadarobotica.com/fontes-conversores/fontes/outros/1x-bateria-3-7v-18650-de-litio-recarregavel>. Acesso em 14 de julho de 2022.

Citisystems. **O que é PWM e Para que serve?.** Disponível em: <https://www.citisystems.com.br/pwm/>. Acesso em 20 de junho de 2022.

Codebender. **Autonomous line follower using Arduino Uno, H-Bridge, 2 DC geared motors and a custom made line sensor from 6 TCRT5000 IR diodes / collectors.** Disponível em: <https://codebender.cc/sketch:11197#Line%20Follower.ino>. Acesso em 10 de julho de 2021.

FilipeFlop. **Como montar um Robô Seguidor de Linha com Arduino Motor Shield.** Disponível em: <https://www.filipeflop.com/blog/projeto-robo-seguidor-de-linha-arduino/>. Acesso em: 21 de junho de 2022.

FilipeFlop. **Driver Motor Ponte H L298N.** Disponível em: <https://www.filipeflop.com/produto/driver-motor-ponte-h-l298n/>. Acesso em: 14 de julho de 2022.

LIMA, Eduardo Oliveira. **Interdisciplinaridade no Ambiente Escolar: Matemática e saúde.** Disponível em: <https://www.ufjf.br/emem/files/2015/10/INTERDISCIPLINARIDADE-NO-AMBIENTE-ESCOLAR-MATEM%c3%81TICA-E-SA%c3%9aDE.pdf>. Acesso em 15 de fevereiro de 2021.

SQUIDS Arduino. **Usando o Sensor Óptico Reflexivo TCRT5000 como interruptor – Arduino.** Disponível em: <http://www.squids.com.br/arduino/index.php/projetos-arduino/projetos-squids/intermediario/286-i01-sensor-optico-reflexivo-tcrt5000-comointerruptor-arduino>. Acesso em: 21 de junho de 2022.

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

ISBN: 978-65-86901-74-0

TARZAN SOLUÇÕES EM ELETRÔNICA. **Soldar é Fácil – Dicas Básicas para Soldar Componentes Eletrônicos**. Disponível em: ‹https://www.facebook.com/tarzancomponentes/photos/a%C3%AD-vai-uma-dica-de-como-soldar-bem-os-componentes-nesse-carnaval/2117923934988991/›. Acesso em: 12 de fevereiro de 2021.

Usinainfo. **Roda com Caixa de Redução e Motor 120:1 80 RPM.** Disponível em: ‹https://www.usinainfo.com.br/rodas-roboticas/roda-com-caixa-de-reducao-e-motor-1201-80rpm-2297.html?gclid=CjwKCAjw_b6WBhAQEiwAp4HyIAnCmJ0DFCY9LeIRM2I-qbwYNWce KPM2-dduF3Mzse1SP7ytcpAeePhoCSmoQAvD_BwE›. Acesso em: 14 de julho de 2022.

ISBN: 978-65-86901-74-0

# ANEXO 1

Algoritmo adaptado de (42 Bots, 2013) e (Codebender, 2013).

```
const int quantSensores = 6;
const int motorRPin1 = 12;
const int motorRPin2 = 11;
const int motorREnable = 10;

const int motorLPin1 = 5;
const int motorLPin2 = 4;
const int motorLEnable = 3;

const int irPins[quantSensores] = {A0, A1, A2, A3, A4, A5};

int irSensorDigital[quantSensores] = {0, 0, 0, 0, 0, 0};

int treashold = 100; //depende da iluminação do ambiente

int irSensors = B000000;

int cont = 0;
int error = 0;
int errorLast = 0;
int correction = 0;

int maxSpeed = 255;
int motorLSpeed = 0;
int motorRSpeed = 0;

void setup()
{
 pinMode(motorLPin1, OUTPUT);
 pinMode(motorLPin2, OUTPUT);
 pinMode(motorLEnable, OUTPUT);
 pinMode(motorRPin1, OUTPUT);
 pinMode(motorRPin2, OUTPUT);
 pinMode(motorREnable, OUTPUT);

 for (int i = 0; i < quantSensores; i++)
 {
  pinMode(irPins[i], INPUT_PULLUP);
 }
}

void loop()
{
 Scan();
 UpdateError();
 UpdateCorrection();
 Drive();
}

void Scan()
{
 cont = 0;
 irSensors = B000000;

 for (int i = 0; i < quantSensores; i++)
 {
  irSensorDigital[i] = (analogRead(irPins[i]) >= treashold);

  cont = cont + irSensorDigital[i];
  int b = (quantSensores - 1 ) - i;
  irSensors = irSensors + (irSensorDigital[i] << b);
 }
}

void UpdateError()
{

 errorLast = error;

 switch (irSensors)
 {
  case B000000:
   error = 0;
  break;

  case B110000:
   error = -120;
  break;

  case B011000:
   error = -60;
  break;

  case B001100:
   error = 0;
  break;

  case B000110:
   error = 60;
  break;

  case B000011:
   error = 120;
  break;

  case B111000:
   error = -150;
  break;

  case B011100:
   error = -120;
  break;

  case B000111:
   error = 150;
  break;

  case B001110:
   error = 120;
  break;

  case B111100:
   error = -150;
  break;

  case B111010:
   error = -150;
  break;

  case B001111:
   error = 150;
  break;

  case B010111:
   error = 150;
  break;

  case B111110:
   error = -150;
  break;

  case B011111:
   error = +150;
```

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

ISBN: 978-65-86901-74-0

```
    if (errorLast <= -30)
    {
      error = -150;
    }
    else
      if (errorLast >= 30)
      {
        error = 150;
      }
    break;

  case B100000:

  error = -150;
  break;

  case B010000:   error = -90;
  break;

  case B001000:   error = -30;
  break;

  case B000001:   error = 150;
  break;

  case B000010:   error = 90;
  break;

  case B000100:   error = 30;
  break;
```

```
    break;

   default:
   error = errorLast;
  }
}

void UpdateCorrection()
{
 if (error >= 0 && error < 30)
 {
  correction = 0;
 }
 else
   if (error >= 30 && error < 60)
   {
     correction = 15;
   }
   else
     if (error >= 60 && error < 90)
     {
       correction = 40;
     }
     else
       if (error >= 90 && error < 120)
       {
         correction = 55;
       }
       else
         if (error >= 120 && error < 150)
         {
           correction = 75;
         }
         else
           if (error >= 150 && error < 180)
           {
             correction = 255;
           }
           else
             if (error >= 180)
             {
               correction = 305;
             }
 if (error <= 0 && error > -30)
 {
  correction = 0;
 }
 else
   if (error <= -30 && error > -60)
   {
    correction = -15;
   }
   else
     if (error <= -60 && error > -90)
     {
```

```
  motorRSpeed = 255;
 }
 else
   if (motorRSpeed < -255)
   {
     motorRSpeed = -255;
   }

 if (motorLSpeed > 255)
 {
  motorLSpeed = 255;
 }
 else
   if (motorLSpeed < -255)
   {
     motorLSpeed = -255;
   }

 if (motorRSpeed > 0)
 {
  analogWrite(motorREnable, motorRSpeed);
  digitalWrite(motorRPin1, HIGH);
  digitalWrite(motorRPin2, LOW);
 }
 else
   if (motorRSpeed < 0)
   {
```

DOI: 10.46943/VIII.CONEDU.2022.GT19.018

ISBN: 978-65-86901-74-0

```
        }
        else
          if (error <= -90 && error > -120)
          {
            correction = -55;
          }
          else
            if (error <= -120 && error > -150)
            {
              correction = -75;

            }
            else
              if (error <= -150 && error > -180)
              {
               correction = -255;
              }
              else
                if (error <= -180)
                {
                  correction = -305;
                }
  if (correction >= 0)
  {
    motorRSpeed = maxSpeed - correction;
    motorLSpeed = maxSpeed;
  }
  else
    if (correction < 0)
    {
      motorRSpeed = maxSpeed;
      motorLSpeed = maxSpeed + correction;
    }
}

void Drive()
{
  if (motorRSpeed > 255)
  {
    abs(motorRSpeed));
    digitalWrite(motorRPin1, LOW);
    digitalWrite(motorRPin2, HIGH);
  }
  else
    if (motorRSpeed == 0)
    {
      digitalWrite(motorREnable, HIGH);
      digitalWrite(motorRPin1, LOW);
      digitalWrite(motorRPin2, LOW);
    }
  if (motorLSpeed > 0)
  {
    analogWrite(motorLEnable, motorLSpeed);
    digitalWrite(motorLPin1, HIGH);
    digitalWrite(motorLPin2, LOW);
  }
  else
    if (motorLSpeed < 0)
    {
      analogWrite(motorLEnable,
      abs(motorLSpeed));
      digitalWrite(motorLPin1, LOW);
      digitalWrite(motorLPin2, HIGH);
    }
    else
      if (motorLSpeed == 0)
      {
        digitalWrite(motorLEnable, HIGH);
        digitalWrite(motorLPin1, LOW);
        digitalWrite(motorLPin2, LOW);
      }
}
```

DOI: 10.46943/VIII.CONEDU.2022.GT19.018