Relatório Final

Projeto AGV - Entregador de Escritório

Campinas, SP
2019

EA075 - Introdução a Projeto de Sistemas Embarcados

Gildo Santos Rodrigues - 155544
Ângelo Henrique Barbosa - 166527
José Antônio Moreira Schewinsky Junior - 176574
Matheus Dias Lopes Cordeiro - 156717

Objetivo

Os AGV's (Automated Guided Vehicle), amplamente empregados na logística industrial, são exemplos de sistemas autônomos capazes de aumentar a eficiência dos processos de transporte de materiais dentro das plantas fabris. O avanço da tecnologia permitiu que esse tipo de sistema fosse empregado em aplicações fora da indústria, como é mostrado na figura abaixo:

Figura 1. Veículo Autônomo para Escritórios

Neste projeto, o princípio do sistema AGV será implementado para a entrega de correspondências e materiais de pequeno porte para colaboradores em escritórios com percursos bem definidos, necessitando assim uma estrutura menos robusta porém com o mesma lógica funcional.

Aplicação

Edifícios comerciais costumam possuir um grande número de colaboradores e uma das consequências é o grande volume de correspondências e pacotes que necessitam ser entregues aos respectivos destinatários.

O AGV proposto tem como objetivo realizar a entrega das correspondências e pacotes de pequeno porte para cada colaborador.

Premissas de Funcionamento

Como hipótese, haverá um operador responsável por definir as correspondências que o AGV deverá entregar, ou seja, definir quais serão os colaboradores que possuem entrega pendente. Com os destinatários definidos, o AGV irá se deslocar dentro do escritório para realizar as entregas.

Supõe-se que, para cada andar, a ordem dos colaboradores ao longo do percurso do AGV seja conhecida e previamente programada através de uma matriz de entregas, onde cada célula indique se há entrega naquela posição, e que seu estado possa ser alterado ao longo do percurso.

A navegação será feita utilizando como referência de direção uma fita de cor detectável, adesivada ao chão do escritório, desenhando rotas bem definidas e seguras à integridade do AGV. Cada posição contida na matriz de entregas será unicamente detectável pelo AGV e irá guiar o sistema em sua rota. O AGV irá se localizar baseando-se em sua última posição na matriz de entregas, ou seja, a última posição detectada será sua referência para se deslocar para a próxima entrega e ao passar por uma nova posição, irá atualizar sua referência.

Caso o AGV saia da rota determinada, o veículo irá parar e emitir um alarme sonoro contínuo com o objetivo de chamar a atenção de algum colaborador próximo. Através de instruções indicadas na estrutura do AGV, o operador irá pressionar o botão de emergência, onde o alarme irá desligar e os motores serão liberados, assim o operador poderá reposicionar o AGV na rota e, novamente pressionar o botão de emergência para retomar seu funcionamento.

Durante o percurso, o AGV, ao detectar as posições sem entrega pendente, irá apenas atualizar sua referência e continuar o trajeto. Nos locais com entrega pendente, o AGV irá parar e emitir um alarme sonoro intermitente por 2 minutos para chamar a atenção do destinatário. Através do acionamento de um botão (devidamente sinalizado), o destinatário irá indicar o recebimento da entrega, interrompendo o alarme. Caso o destinatário não se encontre, ninguém receba por ele ou assim que o botão de recebimento for pressionado, o AGV irá emitir um alarme sonoro contínuo por 10 segundos com o objetivo de indicar que irá retomar seu percurso e evitar a colisão com colaboradores que possam estar nas redondezas durante esse processo.

Considera-se que a rota a ser percorrida não tenha cruzamentos, possua fim detectável e que toda a trajetória possa ser realizada somente em um sentido. Assim, ao fim do percurso, o operador retornará o AGV a sua estação de carregamento, onde será reabastecido e reprogramado para realizar novamente sua função.

No caso de não terem sido realizadas todas as entregas pendentes, será a critério do operador mantê-las na rota ou tomar outras providências em relação a entrega do conteúdo.

Durante o percurso, o sistema AGV está sujeito a interferências físicas (pessoas circulando, objetos deixados na rota etc), logo o sistema deverá ser capaz de detectar os obstáculos a uma distância segura e como resposta, irá parar e emitir um som contínuo para que colaboradores próximos possam agir no sentido de desobstruir a rota.

Caso o AGV venha a sair da rota, ele irá parar e emitir um sinal sonoro intermitente e irá liberar seus motores para que assim, algum colaborador possa retorná-lo a rota.

Com o objetivo de gerenciar todos os AGV's distribuídos pelo edifício, é necessária alguma forma de geolocalização remota, seja por RFID ou GPS, porém este tópico não será explorado nesse projeto.

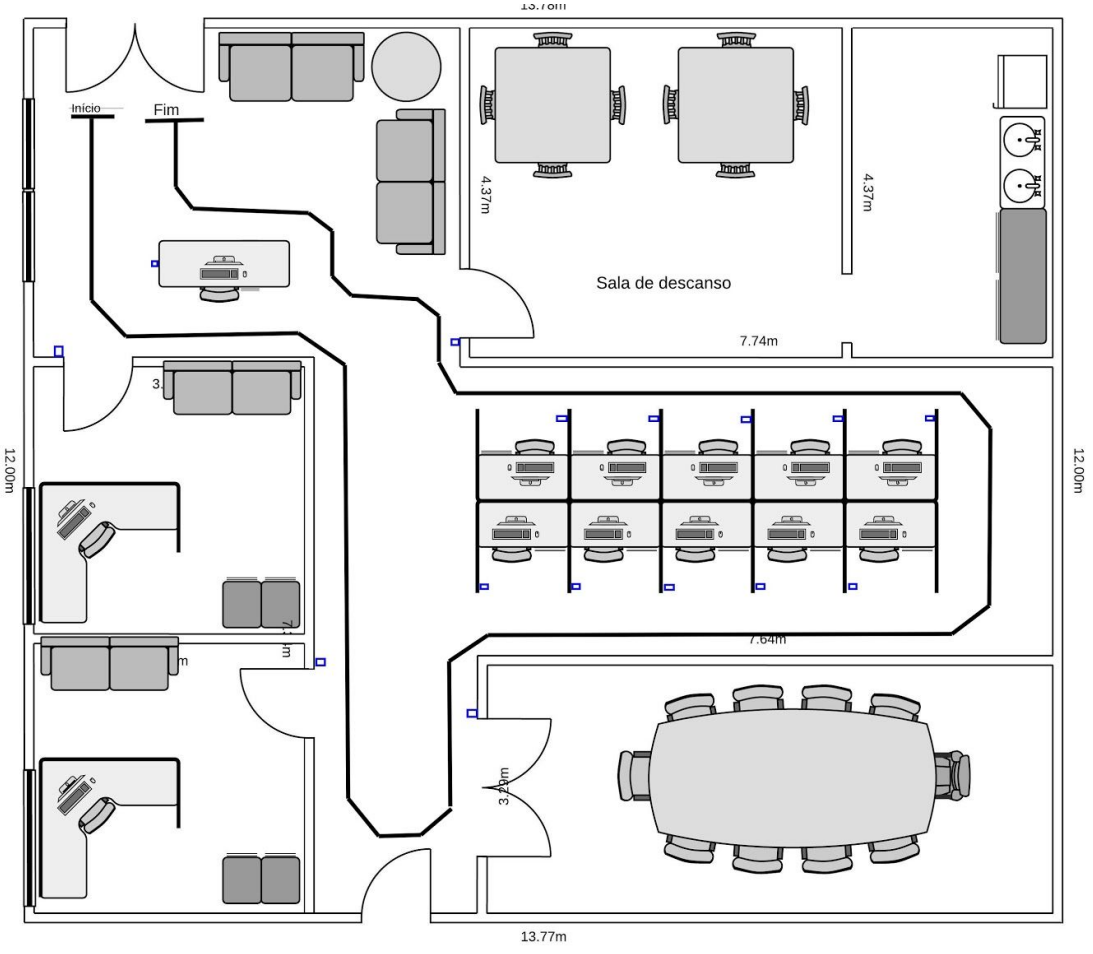


Figura 2. Trajetória AGV com os indicadores de posição em azul

Implementação

Estrutura

Mecânica

Este sistema AGV é projetado para a entrega de pequenos objetos, logo, a massa do sistema somada à massa das entregas deve ser baixa, assim a estrutura do sistema AGV não necessita de muita robustez.

Para este projeto toma-se como hipótese que a massa do AGV junto com o máximo de entregas esteja em torno de 5 kg.

Devido ao tipo de trajetória sugerido para o funcionamento do sistema AGV (trajetória descrita por uma linha bem definida através de fita), com o objetivo de simplificar a estrutura mecânica e a lógica de controle, é possível utilizar uma estrutura com 3 pontos de apoio, sendo eles 2 motores e 1 roda livre. A imagem abaixo demonstra este conceito:

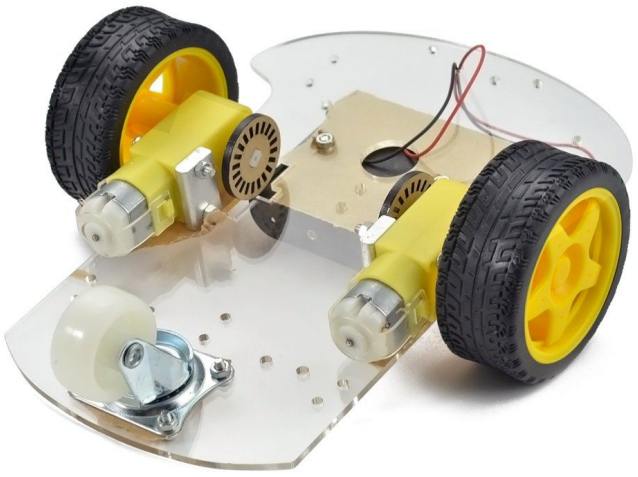


Figura 3. Carro elétrico de 3 rodas

É razoável supôr que para se aumentar a capacidade de carga do sistema demonstrado na Figura 3, é necessário aperfeiçoar o projeto mecânico e elétrico mas, ainda assim, o princípio de funcionamento se manterá.

Compartimento para as entregas

A forma do objeto poderá variar entre envelopes e pequenas caixas (caixa tipo N0, por exemplo), assim pode-se utilizar a estrutura representada na Figura 4 para se obter combinações de compartimentos adequados a cada andar:

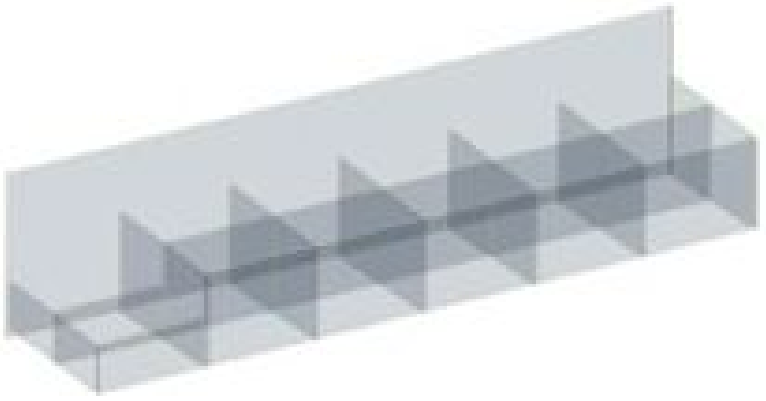


Figura 4. Compartimento da entrega

Hardware

Microcontrolador: Arduino UNO

O Arduino Uno é uma placa baseada no processador ATmega328. Ele tem 14 pinos de entrada/saída digital (dos quais 6 podem ser usados como saídas PWM), 6 entradas analógicas, um cristal oscilador de 16MHz, uma conexão USB, uma entrada de alimentação, uma conexão ICSP, um botão de reset e todos os componentes necessários para suportar as funcionalidades do microcontrolador.

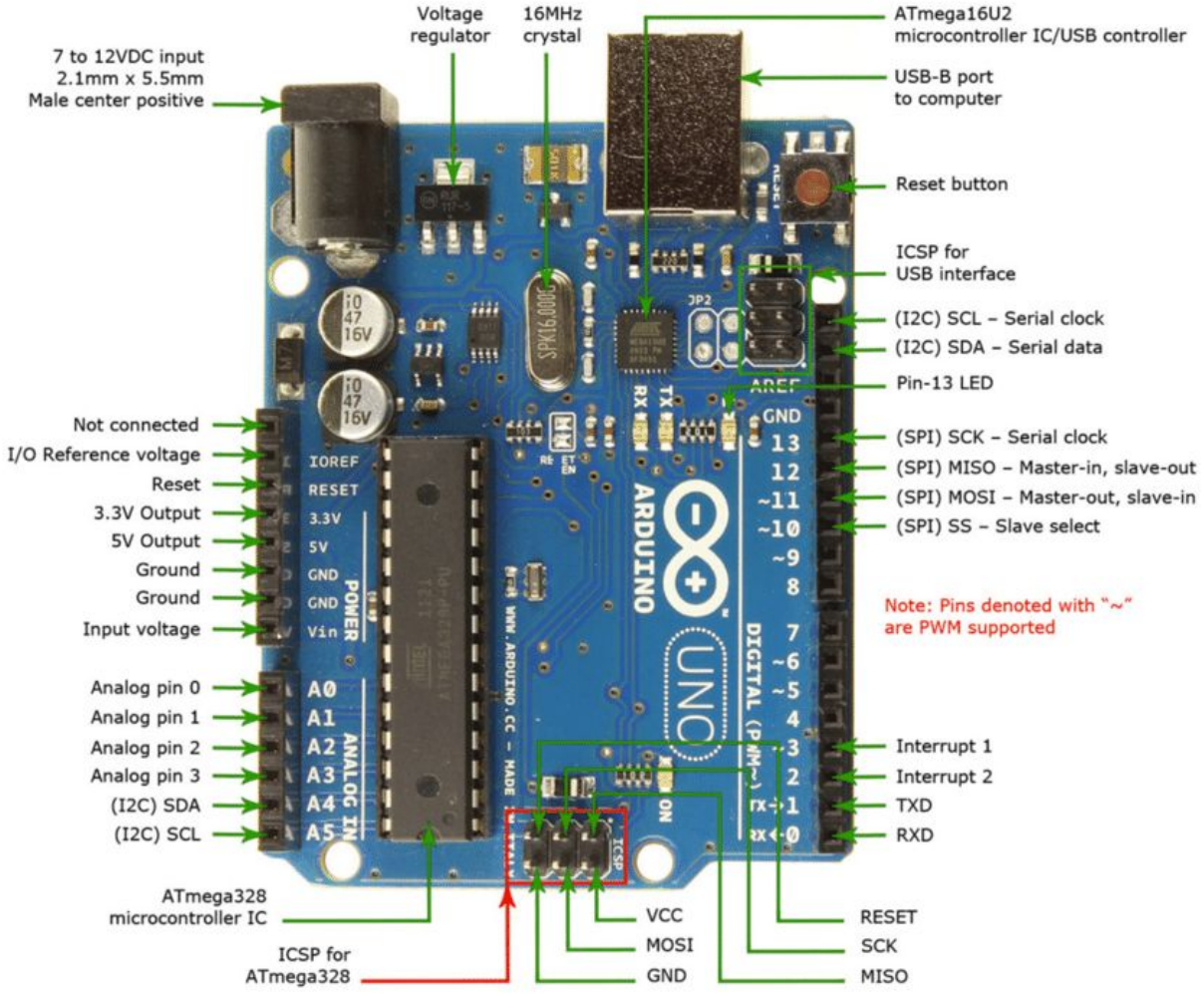


Figura 5: Arduino UNO

A largura e comprimento máximos da placa do Uno são 68,58 e 53,34 mm respectivamente (2,7" x 2,1"). Quatro orifícios para parafusos permitem que a placa seja fixada a estrutura mecânica.

O Arduino Uno pode ser alimentado pela conexão USB ou com uma fonte de alimentação externa. Neste projeto será utilizado um pack de pilhas semelhante ao demonstrado abaixo:

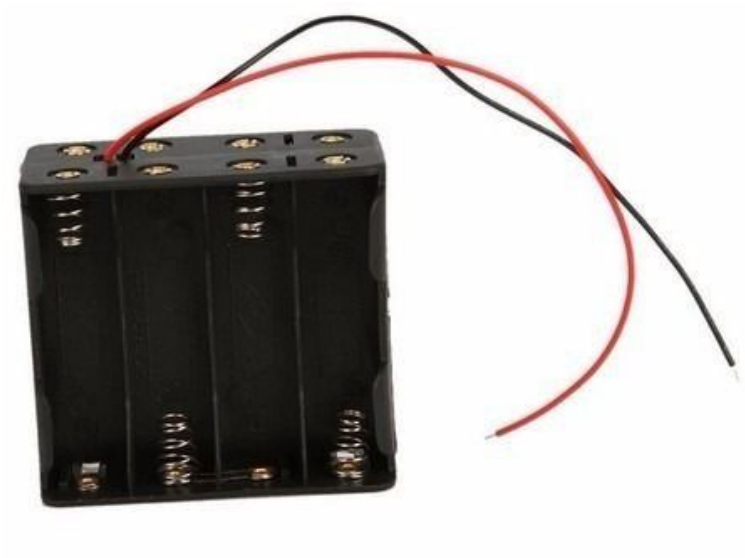


Figura 6: Pack de baterias

A alimentação é selecionada automaticamente e pode ser tanto de um adaptador CA para CC ou bateria. O conector para alimentação é de 2,1mm, com o positivo no centro ou cabos vindos de uma bateria podem ser inseridos diretamente nos pinos Gnd e Vin do conector de alimentação.

Esta placa pode funcionar com uma fonte de alimentação externa de 6 a 20 V. No entanto se a alimentação for inferior a 7V, o pino 5V pode fornecer menos de cinco volts e a placa pode se mostrar instável. E se a alimentação for maior do que 12V o regulador de tensão pode superaquecer e danificar a placa. A faixa recomendada é de **7 a 12 V**.

O ATmega328P têm 32KB (dos quais 0,5KB são utilizados pelo bootloader) de memória FLASH, 2KB de SRAM e 1KB de EEPROM.

Cada pino pode fornecer ou receber um máximo de 40 mA e tem um resistor pull-up interno (desconectado por padrão) de 20-50kΩ.

Além disso há pinos com funções especializadas:

Serial: 0 (RX) e 1 (TX). Usados para receber (RX) e transmitir (TX) dados seriais TTL. Estes pinos são conectados aos pinos correspondentes do chip serial USB-para-TL ATmega8U2.

Interrupções Externas: 2 e 3. Estes pinos podem ser configurados para disparar uma interrupção de acordo com alguma variação sensível pelo circuito.

SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Estes pinos dão suporte à comunicação SPI.
I2C: 4 (SDA) and 5 (SCL). Fornecem suporte a comunicação I2C (TWI).
AREF: Tensão de referência para as entradas analógicas.
Reset: Envio do valor LOW para esta linha para resetar o microcontrolador.

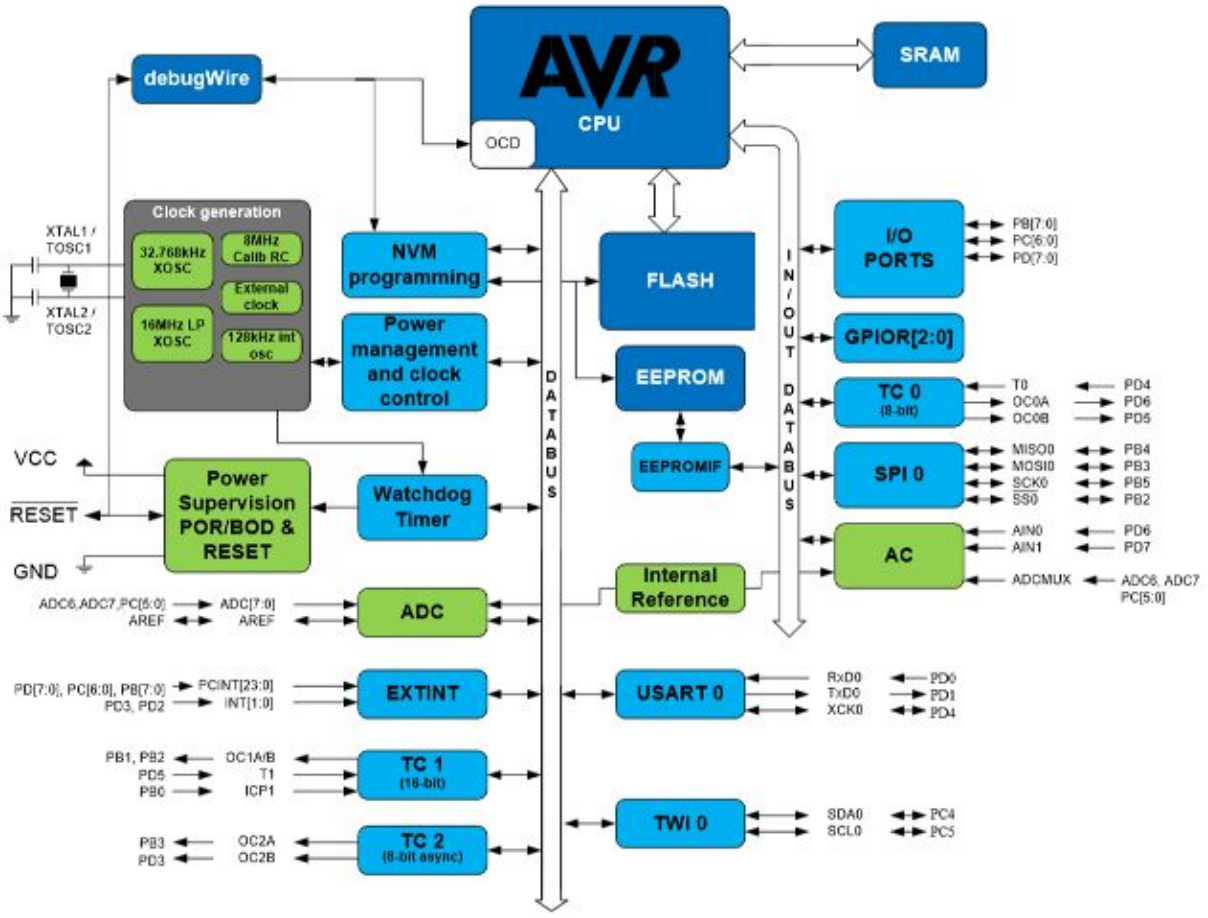


Figura 7: Diagrama de Blocos do Microcontrolador

O Arduino Uno tem 6 entradas analógicas, etiquetadas de A0 a A5, cada uma tem 10 bits de resolução (i.e. 1024 valores diferentes). Por padrão elas medem de 0 a 5V, embora seja possível alterar o limite superior utilizando o pino AREF.

O ATmega328P fornece comunicação serial **UART TTL (5V)** que está disponível nos pinos digitais **0 (RX) e 1 (TX)**. Um ATmega8U2 na placa canaliza esta comunicação para a USB e aparece como uma porta virtual para o terminal no computador. O software do Arduino inclui um monitor serial que permite dados textuais serem enviados e recebidos da placa.

O ATmega328P no Arduino Uno vem pré-gravado com um bootloader que permite enviar código novo para ele sem a utilização de um programador de hardware externo. Ele se comunica utilizando o protocolo original STK500.

Este microcontrolador foi escolhido por possuir todas as funcionalidades necessárias para a implementação do projeto além de extenso conteúdo disponível para consulta.

Sensores

Emissor/Sensor InfraVermelho: O conjunto emite uma luz infravermelha por um LED negro e capta o reflexo com um LED receptor (LED claro). A luz reflete em superfícies claras e é absorvida em superfícies negras. Sendo assim o LED receptor irá detectar a luz infravermelha no branco e não detectar no preto, por exemplo. Para uma melhor eficácia do sensor, a superfície em contraste com a faixa preta deve ser branca.

Para o projeto será utilizado o módulo TCRT5000 o qual possui as seguintes características:

- Controlador: LM393
- Tensão de operação: 3,3 – 5VDC
- Saída Digital e Analógica
- LED indicador de sensor ativado
- LED indicador de tensão no sensor
- Sensibilidade ajustável através de trimpot

A conexão com o microcontrolador será feita através de um pino digital, Vcc e GND.

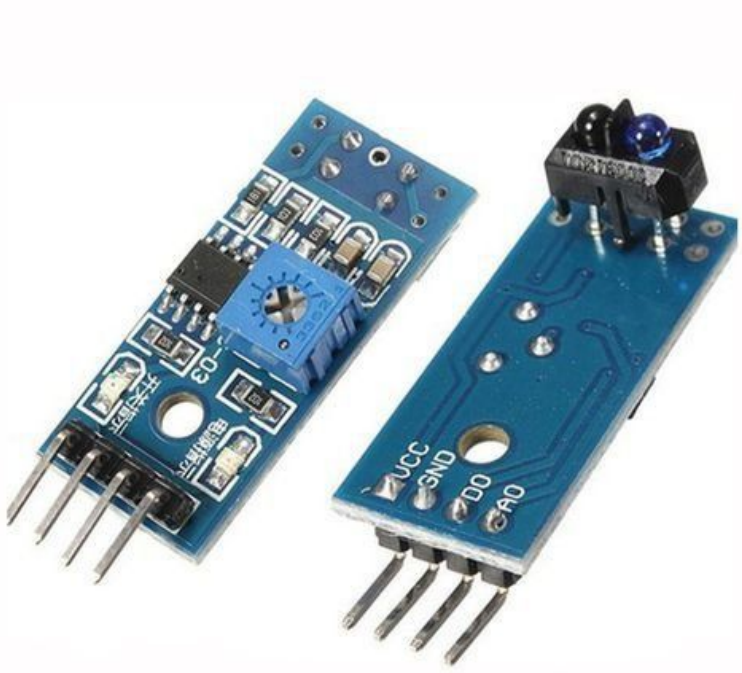


Figura 8: Módulo TCRT5000

Com o objetivo de simplificar a implementação, a forma de detecção da posição de cada entrega será através da utilização de mais um sensor IR posicionado mais ao extremo do AGV, de maneira que se possa identificar marcações feitas ao longo do percurso. Colocamos aqui como ponto de melhoria a utilização de algum sistema de identificação única como por exemplo RFID, Bluetooth, QR Code e outras tantas formas para caracterizar o local da entrega.

Sensor Ultrassônico: O Sensor Ultrasonico HC-SR04 é aplicado com mais frequência em projetos de robótica e é capaz de medir com precisão (3mm de margem de erro) distâncias de 2cm até 4m.

A composição do Sensor Ultrasonico HC-SR04 é feita de um emissor e um receptor ultrassônico, onde o sensor emite (emissor) sinais ultrassônicos que serão refletidos no obstáculo / objeto retornando ao sensor (receptor). Com base no tempo que o sinal emitido levou para retornar ao receptor, o mesmo efetua o cálculo da distância.

A conexão com a placa do microcontrolador será feita através de dois pinos digitais, Vcc e GND.

O módulo possui as seguintes características:

- Tensão de operação: 5VDC
- Corrente de operação: 15mA
- Faixa de detecção (ângulo): ±15º
- Alcance: 2cm a 4m
- Margem de erro: ±3mm

Figura 9: Módulo HC-SR04

Botão: Os botões utilizados neste projeto serão do tipo Chave Táctil 12×12 com Capa. Esta chave é um tipo de interruptor pulsador (conduz somente quando está pressionado).

Este botão é composto basicamente por uma chave táctil 12x12x7.3mm e os pinos para conexão a plataforma. Além disso, a chave táctil possui uma capa que deixa o botão com um melhor acabamento e facilita o pressionamento.

Características:

– Tensão de operação: 3,3 a 5VDC

– Saída de sinal (digital): nível TTL

A conexão com o microcontrolador será feita através de um pino digital, VCC e GND.

Figura 10: Chave Táctil 12x12 com Capa

Atuadores

LED: São diodos semicondutores que, quando energizados, emitem luz. Utilizados para indicar o funcionamento de partes do circuito ou para sinalização.

Figura 11: LED

Buzzer: O **Buzzer 5V** é um componente utilizado para emitir sinais sonoros. O mesmo possui um circuito oscilador que produz o som e só necessita ser energizado.

A conexão com a placa do microcontrolador será feita através de um pino digital e GND.

Figura 12: Buzzer

Motor de Corrente Contínua: Motores elétricos convertem energia elétrica em mecânica. Existem motores de corrente contínua (CC) e de corrente alternada (CA), cada um com diversas variações. Motores CA são geralmente usados para máquinas grandes e recebem energia diretamente da rede de distribuição de energia. Robôs móveis usam tipicamente CC, pois sua fonte de energia pode ser uma bateria.

Para as necessidades de locomoção do AGV, o motor gira em uma velocidade muito alta e com um torque muito baixo. Para inverter essa relação o motor deve ser ligado a uma caixa de redução que produz uma saída com rotação menor, porém com um torque maior.

É possível encontrar motores CC que já possuem a caixa de redução instalada e com acoplamentos adequados a sua aplicação.

Figura 13: Motor CC com caixa de redução e acoplamento adequado a uma roda

Os motores CC possuem pelo menos dois terminais que, ao se aplicar tensão nestes terminais, o motor gira em um sentido e invertendo-se a polaridade o motor gira no sentido contrário.

Ponte H: A Ponte H é um tipo de circuito que possibilita o controle dos motores de corrente contínua. Com ela é possível controlar não só a velocidade do motor, mas também o sentido de rotação do motor.

Para o controle de velocidade utiliza-se os pinos PWM do Arduino para modular a corrente controlada por transistores e para o controle do sentido utiliza-se a lógica de acionamento dos transistores. A figura 13 demonstra a lógica de acionamento.

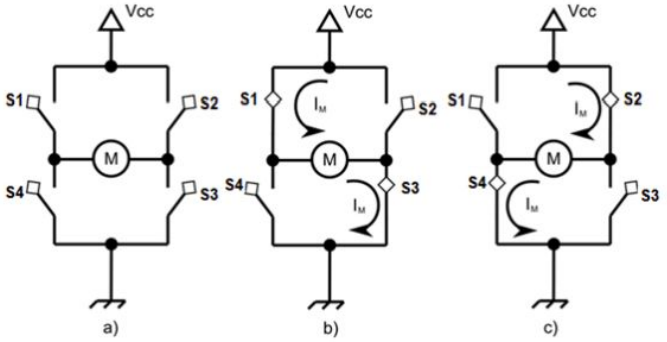


Figura 14: Ponte H - Lógica de Funcionamento

Há quatro transistores (S1, S2, S3 e S4), que podem ser ativados aos pares (S1 e S3) ou (S2 e S4). Para cada configuração, a corrente que fluirá pelo motor terá um sentido, o que fará com a rotação seja no sentido horário ou anti-horário. Caso nenhuma chave seja ativada, o motor permanecerá desligado.

Para o projeto considerado, utilizando um AGV com rodas de 10 cm de diâmetro e a massa de 5kg, é possível aproximar o torque mínimo necessário para o motor em 7,5 kgf.cm (torque necessário para o carrinho sair da inércia).

Para exemplificar o desempenho, é possível supôr a utilização do Micro Motor DC 12V 13RPM 8,5Kgf.cm que possui as seguintes especificações:

- Corrente: 4.4A;
- Potência: 3.7W;
- Tensão nominal: 12V;
- Torque: 8.5Kgf.cm;
- Velocidade (sem carga): 13RPM;
- Velocidade (carga máxima): 11.7RPM;
- Peso unitário: 188g;

Com os valores de corrente exigidos pelo MicroMotor é possível dimensionar os transistores a serem utilizados na ponte H.

Considerando uma velocidade de operação de 12RPM, tem-se que a velocidade máxima atingida pelo sistema seria de aproximadamente 6 cm/s, o que corresponde a uma velocidade razoável exigida para a aplicação.

Figura 15: MicroMotor DC

Pseudocódigo:

O seguinte pseudocódigo tem como objetivo descrever o funcionamento projetado para o sistema AGV:

```
Início:
- Receber como entrada do operador o número inteiro correspondente aos 'endereços(int[n])' dos destinatários (em ordem), bem como o conteúdo a ser entregue;
- int contador de endereços = 0;
int i (ordem de destinatários) = 0;
- Movimento normal:
    - Acionar os dois motores (transistorizados) com a mesma potência PWM (logo mesma velocidade e está andando reto);
    - Se o sensor ultrassônico localizado na frente do carrinho detectar algo:
        - Parar movimento, enquanto ainda estiver detectando;
        - Tocar buzzer com o sinal 1, enquanto o botão do carrinho não for pressionado ou o objeto deixado de ser detectado pelo sensor;
    - Se IV1 (sensor infravermelho localizado na lateral do carrinho) detectar um retorno:
        - Incrementar contador de endereços;
        - Se endereço[i] inserido pelo operador == contador d endereços:
            - Parar movimento de ambos os motores;
            - Se pressionado o botão (pelo receptor):
                - Voltar ao Movimento normal;
    - Se IV2 (sensor infravermelho localizado na parte de baixo do carrinho, à esquerda de IV3 e IV4) deixar de detectar a linha (não obtiver retorno de sinal):
        - Se IV4 não detectar retorno:
            - Se IV3 não detectar retorno:
                - Parar movimento;
                - Tocar buzzer com sinal 2;
            - Se IV3 detectar retorno:
                - Andar durante 3 segundos com a potência da roda esquerda menor que a da direita;
                - Se IV2 e IV4 detectarem retorno, e IV3 não detectar, voltar ao movimento normal, com a mesma potência em ambos os motores;
                - Se IV4 passar a detectar retorno, e IV3 parar:
                    - Voltar a roda da esquerda para a potência normal de operação, e a da direita com metade da potência normal; (Mover para a direita)
                    - Se IV2 e IV4 detectarem retorno, e IV3 não detectar, voltar ao movimento normal, com a mesma potência em ambos os motores;
            - Se IV4 detectar retorno:
                - Andar durante 3 segundos com a potência da roda direita menor que a da esquerda;
                - Se IV2 e IV4 detectarem retorno e IV3 não detectar, voltar ao movimento normal, com a mesma potência em ambos os motores;
    - Se IV4 (sensor infravermelho localizado na parte de baixo do carrinho, à direita de IV3 e IV2) deixar de detectar a linha (não obtiver retorno):
        - Se IV2 não detectar retorno:
            - Se IV3 não detectar retorno: (está 'perdido')
                - Parar movimento e não enviar potência PWM a nenhum dos motores;
                - Tocar buzzer com o sinal 2;
                - Se apertado o botão:
                    - Parar de tocar;
                - Se voltar a detectar o caminho corretamente:
                    - Voltar ao movimento normal;
            - Se IV3 detectar retorno:
                - Andar durante 3 segundos com a potência normal na roda direita, e metade da potência na roda esquerda;
                - Se IV2 e IV4 detectarem retorno, e IV3 deixar de detectar
                    - Voltar ao movimento normal, com a mesma potência em ambos os motores;
                - Se IV4 passar a detectar retorno e IV3 parar, com IV2 ainda não detectando:
                    - Andar durante 6 segundos a roda da esquerda para a potência normal e diminuir a potência da roda direita para a metade;
                - Se IV2 e IV4 detectarem retorno, e IV3 deixar de detectar
                    - Voltar ao movimento normal, com a mesma potência em ambos os motores;
            - Se IV2 detectar retorno:
                - Andar durante 3 segundos com a potência da roda direita menor que a da esquerda;
                    - Se IV2 e IV4 detectarem retorno, e IV3 não:
                        - Voltar ao movimento normal, com a mesma potência em ambos os motores;
```

Funções Auxiliares da principal:

```
41  void LeSensorUltrassonico() {
42    float cmMsec, inMsec;//Le as informacoes do sensor, em cm e pol
43    long microsec = ultrasonic.timing();
44    cmMsec = ultrasonic.convert(microsec, Ultrasonic::CM);
45    inMsec = ultrasonic.convert(microsec, Ultrasonic::IN);
46    Serial.print("Distancia em cm: ");//Exibe informacoes no serial monitor
47    Serial.print(cmMsec);
48    Serial.print(" - Distancia em polegadas: ");
49    Serial.println(inMsec);
50    delay(100);
51  }
52  void LeSensorInfravermelho() {
53    if (digitalRead(pinoSensorInfravermelho) == LOW) //SE A LEITURA DO PINO FOR IGUAL A LOW, FAZ
54          digitalWrite(pinoLedInfravermelho, HIGH); //ACENDE O LED
55    else//SENÃO, FAZ
56          digitalWrite(pinoLedInfravermelho, LOW); //APAGA O LED
57    delay(100);
58  }
59  boolean LeBotao() {
60    if (digitalRead(pinoBotao) == LOW) //SE A LEITURA DO PINO FOR IGUAL A LOW, FAZ
61          return true;
62    else//SENÃO, FAZ
63          return false;
64  }
65  void TocaBuzzer(int frequencia) {
66    if (frequencia==132) // frequência correspondente a notá Dó
67      tone(9,frequencia);
68    else if (frequencia==148) // frequência correspondente a notá Ré
69      tone(9,frequencia);
70    else if (frequencia==166) // frequência correspondente a notá Mi
71      tone(9,frequencia);
72    else if (frequencia==176) // frequência correspondente a notá Fá
73      tone(9,frequencia);
74    else if (frequencia==198) // frequência correspondente a notá Sol
75      tone(9,frequencia);
76    else if (frequencia==222) // frequência correspondente a notá Lá
77      tone(9,frequencia);
78    else if (frequencia==249) // frequência correspondente a notá Si
79      tone(9,frequencia);
80    delay(1);
81  }
```

```
86  void MoveMotor(string motorSelecionado, string instrucao, string speed)
87  {
88    if (motorSelecionado=="A") {
89      if (instrucao=="Horario") {       //Gira o Motor A no sentido horario
90        digitalWrite(pinoMotor1, HIGH);
91        digitalWrite(pinoMotor2, LOW);
92      }
93      if (instrucao=="Anti-Horario") {       //Gira o Motor A no sentido anti-horario
94        digitalWrite(pinoMotor1, LOW);
95        digitalWrite(pinoMotor2, HIGH);
96      }
97      if (instrucao=="Parar") {     //Para o motor A
98        digitalWrite(pinoMotor1, HIGH);
99        digitalWrite(pinoMotor2, HIGH);
100     }
101     if (speed =="Lento")
102       delay(2000);
103     if (speed =="Rapido")
104       delay(1000);
105   }
106   if (motorSelecionado=="B") {
107     if (instrucao=="Horario"){  //Gira o Motor B no sentido horario
108       digitalWrite(pinoMotor3, HIGH);
109       digitalWrite(pinoMotor4, LOW);
110     }
111     if (instrucao=="Anti-Horario") {//Gira o Motor B no sentido anti-horario
112       digitalWrite(pinoMotor3, LOW);
113       digitalWrite(pinoMotor4, HIGH);
114     }
115     if (instrucao=="Parar") {//Para o motor B
116       digitalWrite(pinoMotor3, HIGH);
117       digitalWrite(pinoMotor4, HIGH);
118     }
119     if (speed =="Lento")
120       delay(2000);
121     if (speed =="Rapido")
122       delay(1000);
123   }
124 }
```

Esquemáticos:

FSMD

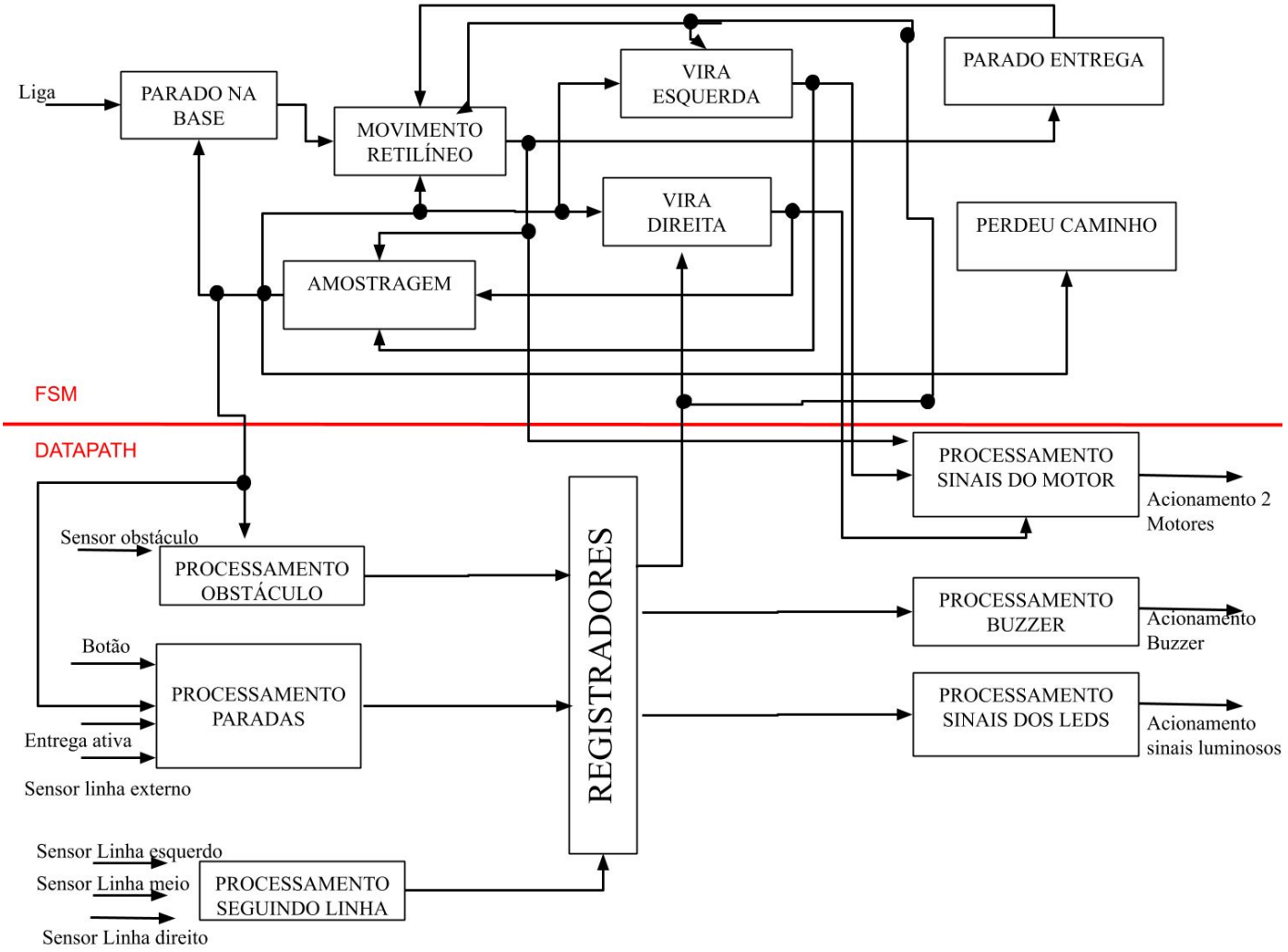
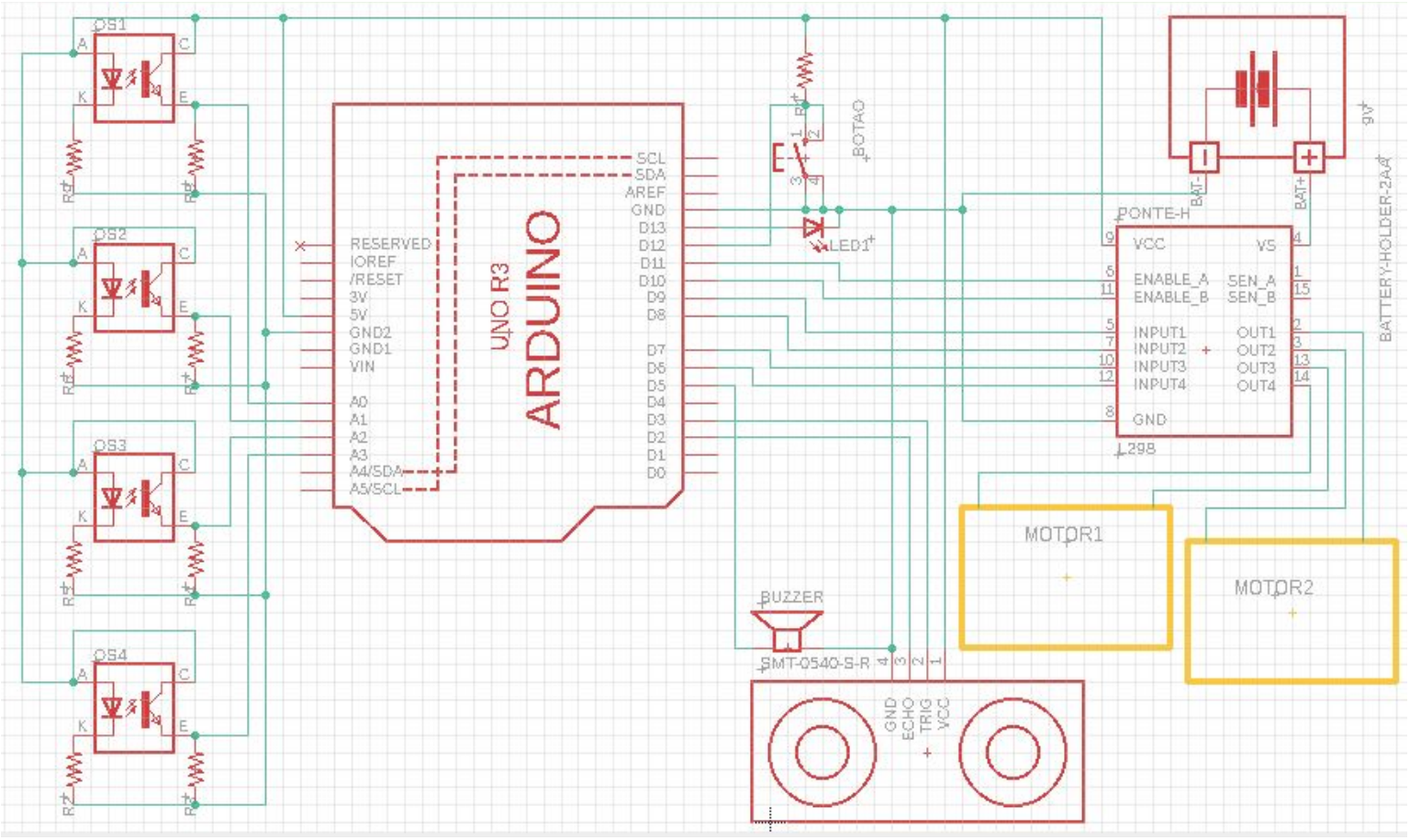


Figura 16: FSMD

Circuito



Conclusão

Neste projeto foi possível demonstrar as necessidades básicas de um sistema AGV e as lógicas para seu funcionamento. Para prototipagem e experimentações o sistema apresentado pode ser considerado uma boa fonte de dados e observações porém é notável que, para aplicações reais, seja necessário o aperfeiçoamento das características mecânicas e elétricas, além de um bom estudo de eficiência energética aliado à utilização de uma bateria adequada a aplicação.

Bibliografia:

https://www.bastiansolutions.com/solutions/technology/automated-guided-vehicles/lynx-mobile-robot/

https://www.newtoncbraga.com.br/index.php/microcontrolador/138-atmel/14863-conhecendo-o-cerne-do-microcontrolador-atmega328p-arduino-uno-mic165

https://blogmasterwalkershop.com.br/arduino/arduino-utilizando-o-sensor-ultrasonico-hcsr04-e-buzzer-5v/

https://blogmasterwalkershop.com.br/arduino/como-usar-com-arduino-sensor-seguidor-de-linha-tcrt5000/

https://blogmasterwalkershop.com.br/arduino/como-usar-com-arduino-modulo-botao-chave-tactil-12x12-com-capa/