|  |  |
| --- | --- |
|  |  |

Coordenadoria de Tecnologia da Informação

Tecnologia em Análise e Desenvolvimento de Sistemas

Veículo autônomo guiado por coordenadas de GPS

Jony Alan de Carvalho Nunes

Sorocaba

Dezembro – 2019



|  |  |
| --- | --- |
|  |  |
|  |  |

**Coordenadoria de Tecnologia da Informação**

Tecnologia em Análise e Desenvolvimento de Sistemas

Veículo autônomo guiado por coordenadas de GPS

Jony Alan de Carvalho Nunes

Prof. Fernando César Miranda – Orientador

Sorocaba

Dezembro - 2019

**Dedicatória**

Dedico este trabalho aos meus pais, minha esposa Rebeca Massoni e aos educadores que foram parte fundamental da estruturação da minha educação, sabedoria e do meu caráter.

**Agradecimento**

Agradeço a Deus por ter me dado força, discernimento, saúde e inteligência para encarar os desafios, e a todos os professores que com dedicação, orientação e brilhantes ensinamentos iluminaram meu caminho para chegar até aqui.

Em especial ao meu orientador Prof.Fernando César Miranda, pela imensa paciência, competência e pelos ensinamentos valiosos que carregarei pelo resto da vida.

# Sumário

[1. Objetivo 2](#_Toc32513959)

[2. Métodos e resultados 3](#_Toc32513960)

[2.1. Motor 4](#_Toc32513961)

[2.2. Display 5](#_Toc32513962)

[2.3. Módulo GPS 6](#_Toc32513963)

[2.4. Arduino 7](#_Toc32513964)

[2.5. Bússola magnética 8](#_Toc32513965)

[2.6. Rodizio giratório 9](#_Toc32513966)

[2.7. Bateria 9](#_Toc32513967)

[2.8. Placa Hoverboard 9](#_Toc32513968)

[3. Estrutura do protótipo 10](#_Toc32513969)

[3.1. Descrição das etapas 10](#_Toc32513970)

[3.1.1 Interpretação das coordenadas 10](#_Toc32513971)

[3.1.2 Estabelecer direção 10](#_Toc32513972)

[3.1.3 Controle de velocidade 10](#_Toc32513973)

[3.1.4 Desviar de obstáculos 10](#_Toc32513974)

[3.2 Tecnologias Disponíveis no Mercado 11](#_Toc32513975)

[3.3 Restrições e Riscos do Projeto 11](#_Toc32513976)

[3.3.1 Limitações Operacionais 11](#_Toc32513977)

[4. Bibliotecas utilizadas 12](#_Toc32513978)

[4.1 SoftwareSerial.h 12](#_Toc32513979)

[4.2 TinyGPS.h 12](#_Toc32513980)

[4.3 HMC5883L \_Simple.h 12](#_Toc32513981)

[4.4. SPI.h 13](#_Toc32513982)

[4.5. Adafruit\_ST7735.h 13](#_Toc32513983)

[4.6. Adafruit\_GFX.h 13](#_Toc32513984)

[5. Arquitetura da aplicação 14](#_Toc32513985)

[5.1. Ligação dos componentes Arduino1 14](#_Toc32513986)

[5.2. Ligação dos componentes Arduino2 15](#_Toc32513987)

[5.3. Comunicação entre Arduino1 e Arduino2 15](#_Toc32513988)

[6. Percurso de Teste 16](#_Toc32513989)

[7. Exemplos do código 19](#_Toc32513990)

[7.1 Algoritmos de posicionamentos 19](#_Toc32513991)

[7.2 Algoritmos de envio da direção 20](#_Toc32513992)

[7.3 Algoritmos de velocidade 21](#_Toc32513993)

[7.4. Algoritmos de desvio de obstáculos 22](#_Toc32513994)

[8. Considerações Legais 23](#_Toc32513995)

[9. Código fonte 24](#_Toc32513996)

[10. Conclusão 25](#_Toc32513997)

# Lista de Figuras

[Figura 1 – Visão geral do protótipo 3](#_Toc24754938)

[Figura 2 - Motores de Hoverboard 4](#_Toc24754939)

[Figura 3- Display ST7735 LCD TFT 1.8 5](#_Toc24754940)

[Figura 4 - Módulo GPS GY-GPS6MV2 6](#_Toc24754941)

[Figura 5 - Arduino UNO 7](#_Toc24754942)

[Figura 6 - Bússola eletrônica HMC5883L 8](#_Toc24754943)

[Figura 7- Placa mãe Hoverboard 9](#_Toc24754944)

[Figura 8 - Ligação dos componetes ao Arduino1 14](#_Toc24754945)

[Figura 9 - Ligação dos componentes ao Arduino2 15](#_Toc24754946)

[Figura 10 - Comunicação entre Arduino1 e Arduino2 15](#_Toc24754947)

[Figura 11 - Percurso de teste 16](#_Toc24754948)

[Figura 12 - Algoritmos de posicionamentos 19](#_Toc24754949)

[Figura 13 - Algoritmos de posicionamentos 20](#_Toc24754950)

[Figura 14 - Algoritmos de posicionamentos 21](#_Toc24754951)

[Figura 15 - Algoritmos de posicionamentos 22](#_Toc24754952)

# Lista de Tabelas

[Tabela 1 - Lista de limitações operacionais 11](#_Toc24754895)

[Tabela 2- Percurso de teste 16](#_Toc24754896)

**Veículo autônomo guiado por coordenadas de GPS**

# Resumo

Este trabalho tem como objetivo mostrar e documentar as etapas envolvidas no processo de desenvolvimento de um protótipo de carro autônomo, que através de um destino estabelecido consiga calcular sua rota e assim estabelecer uma direção e velocidade precisa e se deslocar até ela desviando de obstáculos que possam vir a aparecer pelo caminho, tudo isso utilizando equipamentos eletrônicos de baixo custo ou reaproveitados, sendo codificados com bibliotecas de plataformas abertas e software livre, espera-se que os estudos e análises descritos aqui, possam ser implementados e melhorados em novos projetos que necessitem de deslocamento sem controle visual, rotas de monitoramento, ou qualquer lugar que não se possa adentrar por algum tipo de obstáculo.

Palavras-chave: Módulo GPS; Bússola eletrônica, Autonomia, Monitoramento, Desvio de obstáculos, Sensores, Arduino, Cálculo de rotas, Tomada de decisão

# 1. Objetivo

O objetivo principal deste projeto é utilizar as tecnologias disponíveis e de fácil acesso por estudantes e pesquisadores, para se elaborar um protótipo a fim de testar a viabilidade de um sistema de locomoção autônoma, cuja eficácia possa ser comparada com os sistemas que estão sendo desenvolvidos hoje por grandes empresas tecnológicas.

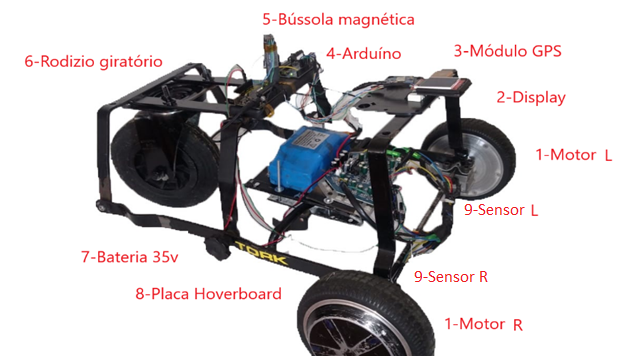
Todos os códigos, imagens, bibliotecas, exemplos ficarão disponíveis em um endereço eletrônico, para que se possa ser baixado adaptado e utilizado por todos os interessados nestas abordagens.

Protótipo é o termo usado para se referir ao que foi criado pela primeira vez, servindo de modelo ou molde para futuras produções, no desenvolvimento de produtos, por exemplo, a confecção de protótipos é parte essencial do projeto, consistindo a fase onde são realizados todos os testes práticos, antes que este possa ser disponibilizado para produção em larga escala ou comercialização.

# 2. Métodos e resultados

A estrutura final do projeto ficou como apresentado na figura 1, nela é possível ver todas as ligações entre os componentes, assim como sua alimentação e botões de segurança que no desenvolver do projeto foram muito utilizados para desligar em casos de panes causadas por erros de parâmetros e reconfigurações.

Figura – Visão geral do protótipo



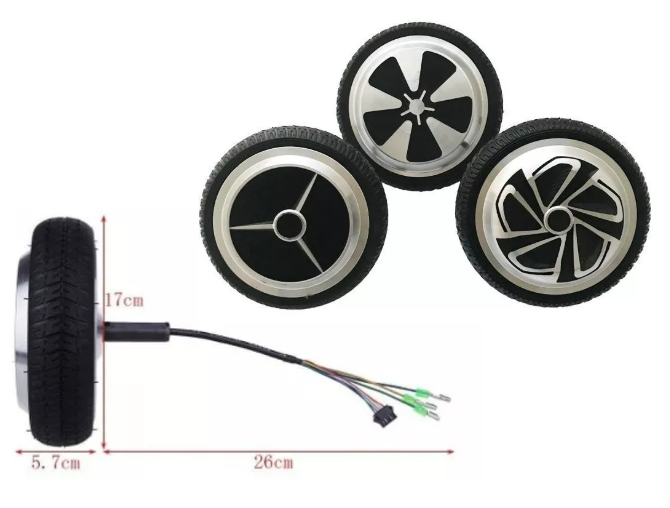
Fonte: Autor.

## 2.1. Motor

Após muitos testes com diferentes tipos e potência de motores, chegou-se à conclusão que o motor a ser usado precisava ser um motor tipo PWM (Pulse Width Modulation), este motor que melhor se adapta aos comados precisos enviados por um controlador. Neste projeto foram utilizados os motores de um brinquedo chamado Hoverboard cujas rodas de 6.5 polegadas são motores de 350w cada.

PWM é uma forma de controlar a quantidade de energia entregue a um dispositivo elétrico ligando e desligando a tensão e quanto maior a frequência do ciclo, mais estável será o valor obtido na coisa controlada.

Figura - Motores de Hoverboard



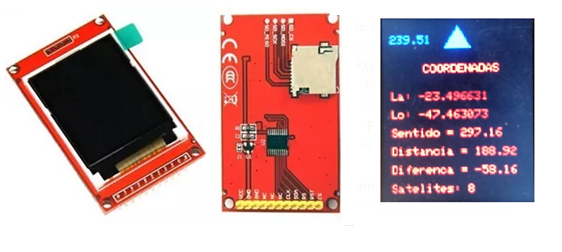
Fonte: Adaptado de <https://pt.aliexpress.com/popular/motor-hoverboard.html>.

## 2.2. Display

O display utilizado para exibição dos dados enviados pelos módulos arduinos é o ST7735 LCD TFT 1.8 128×160, cuja tensão de operação é de 3,3VDC à 5VDC, usa interface SPI para comunicação, e pode ser utilizado não só com arduino, mas também com placas como Raspberry, Beaglebone, módulos ESP8266, etc.

Nele são exibidas as coordenadas do ponto a ser atingido em LA (Latitude) e LO (Longitude), Sentido (direção) que o carro deve se deslocar, distância do ponto atual até o próximo ponto de deslocamento, diferença em graus, e a quantidade de satélites disponíveis no momento do recebimento das coordenadas

Figura - Display ST7735 LCD TFT 1.8



Fonte: Autor.

## 2.3. Módulo GPS

O Módulo GPS (Global Positioning System) modelo GY-GPS6MV2 é um dispositivo eletrônico especialmente desenvolvido para aplicação em projetos robóticos ou domóticos que tenham como base plataformas de protipagem, entre elas o arduino e o Raspberry PI.

O Módulo GPS funciona em conjunto com a plataforma microntroladora, informando ao arduino, se for o caso, a localização exata de determinado objeto em que o Módulo GPS esteja instalado, enviando dados referentes a latitude e longitude, data, hora e velocidade de deslocamento.

Outra importante aplicação que pode ser dada ao Módulo GPS é comandar um robô para executar diferentes tarefas em localizações e horários predefinidos.

Uma informação importante do Módulo GPS é sua interface serial de 3,3V, não tolerando 5V. Deste modo, se o microcontrolador tiver um GPIO de 5V, será necessário o uso de um level shift, trabalha com uma taxa de transmissão padrão de 9600 bps, sendo observado o mal funcionamento em outras velocidades.

Figura - Módulo GPS GY-GPS6MV2



Fonte: https://www.filipeflop.com/produto/modulo-gps-gy-neo6mv2/

## 2.4. Arduino

Segundo o site www.arduino.cc o arduino é uma plataforma de prototipagem eletrônica de hardware livre e de placa única, projetada com um microcontrolador Atmel AVR com suporte de entrada/saída embutido, uma linguagem de programação padrão, que tem origem em Wiring, e é essencialmente C/C++. O objetivo do Arduino é criar ferramentas que são acessíveis, com baixo custo, flexíveis e fáceis de usar por principiantes e profissionais. Principalmente para aqueles que não teriam alcance aos controladores mais sofisticados e ferramentas mais complicadas.

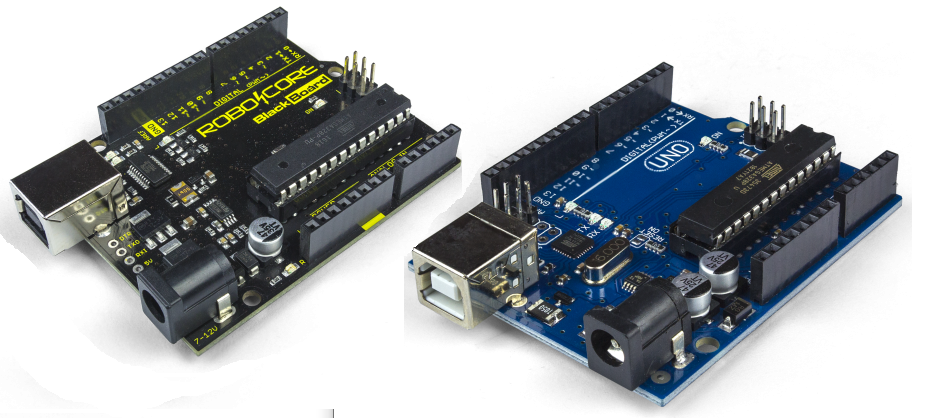
Graças a sua IDE de fácil interação foram possíveis às inúmeras alterações correções e adaptações até que se começasse a ter os resultados esperados,

Neste projeto foram utilizados 2 modelos, um Arduino Uno e uma BlackBoard UNO R3 fablicada pela RoboCore. Ambas possuem as mesmas configurações e vem equipadas com um microcontrolador ATmega328P.

Este Arduino UNO é o modelo mais popular, possui vasta documentação, diversos blogs e tutoriais, por esses e alguns outros motivos é o mais utilizado por quem está iniciando no mundo Arduino.

Eles possuem o microcontrolador Atmega328 versão encaixável, esse detalhe é muito bem visto pelos iniciantes, pois caso ocorra um erro em um teste que cause algum dano ao microcontrolador, como a ligação de um VCC e GND, por exemplo, normalmente, você resolve apenas substituindo o chip atmega328, facilmente encontrado em lojas e na internet.

Figura - Arduino UNO



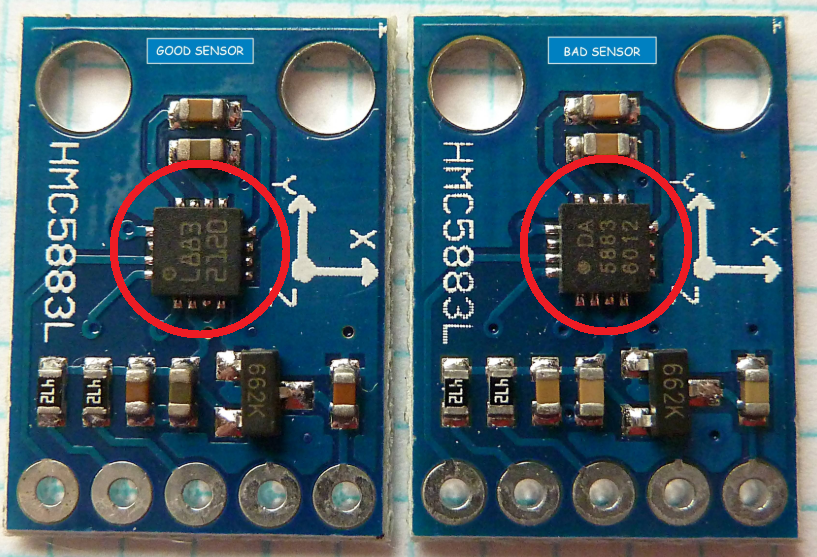
Fonte: Adaptado da internert

## 2.5. Bússola magnética

O módulo HMC5883L GY-273 é uma bússola digital de três eixos, este sensor age da mesma forma que a agulha de uma bússola, se alinha com o campo magnético mais forte. O funcionamento interno do sensor depende de resistores sensíveis a campos magnéticos, instalados de maneira a formar um sistema de coordenadas ortogonal (x, y, z). O seu barramento serial I2C permite fácil interface, trabalha em tensão de alimentação de 3 á 5V, seu chip HMC5883L é de grande precisão, diferente do modelo com o chip QMC5883L de qualidade inferior.

A figura 6 ilustra a diferença entre as placas com o HMC5883L e QMC5883L.

Figura - Bússola eletrônica HMC5883L



Fonte: adaptado de [https://surtrtech.com/2018/02/01/interfacing-hmc8553l-qmc5883-digital-compass-with-arduino](https://surtrtech.com/2018/02/01/interfacing-hmc8553l-qmc5883-digital-compass-with-arduino%20%20)

## 2.6. Rodizio giratório

Nos testes ele se mostrou o mais eficiente, já que permite fazer uma volta em cima do eixo central do carrinho, ou seja, houve um ganho em dirigibilidade.

## 2.7. Bateria

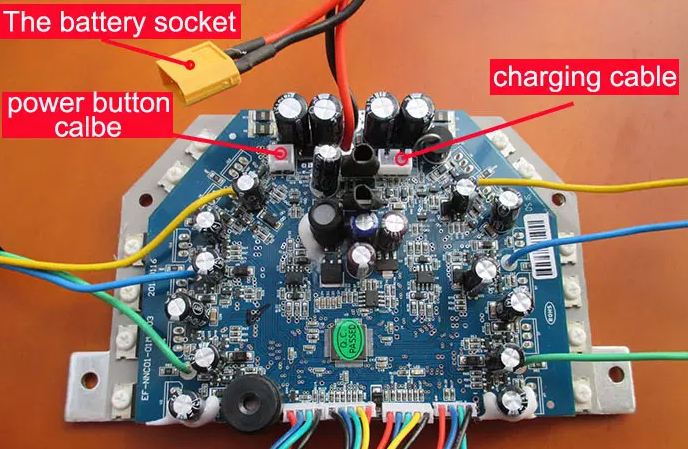
A bateria de Lithium-ion 18650 10S2, possui voltagem nominal de 36V, capacidade de 4400mAh, voltagem de carregamento de 42V, peso bruto de 1Kg sua autonomia no projeto foi de até 20KM com uma carga, alimentando todos os módulos e os motores, ela foi reaproveitada de um hoverboard.

## 2.8. Placa Hoverboard

Esta placa, também chamada de placa mãe hoverboard é a grande responsável pelo controle dos motores.

Recebe os 36V da batreria e distribui através de controladores PWM os pulsos para o controle dos motores, ela não precisou ter seu firmware modificado para receber comandos do arduino ou de outros dispositivos, o método para tal tarefa será assunto de outro tópico.

Figura - Placa mãe Hoverboard



Fonte: adaptado de https://github.com/NiklasFauth/hoverboard-firmware-hack

# 3. Estrutura do protótipo

## 3.1. Descrição das etapas

As etapas foram desenvolvidas e testadas isoladamente até que se estivesse o resultado esperado para então serem acopladas ao projetos, nos itens a seguir são descritos os resultados esperados em cada uma das etapas.

### 3.1.1 Interpretação das coordenadas

Do módulo GPS são recebidos pacotes de dados da malha de satélites GPS, mas para funcionamento do protótipo são necessárias apenas as coordenadas de latitude e longitude.

Através da posição inicial e posição final são estabelecidos o sentido em graus que ele deve percorrer,

### 3.1.2 Estabelecer direção

A direção é estabelecida a cada 1 segundo de acordo com o algoritmo das coordenadas GPS, tem controle principalmente sobre o controle de velocidade, a direção e estabelecida da seguinte forma, de 338° a 23° Norte, de 24° a 67° Nordeste, de 68° a 112° Leste, de 113° a 156° Sudeste, de 157° a 202° Sul, de 203° a 247° Sudoeste, de 248° a 292° Oeste, de 293° a 337° Noroeste.

### 3.1.3 Controle de velocidade

O controle de velocidade é continuo e pré-determinado no momento de iniciação do percurso, só é interrompido por novos sentidos estabelecidos ou pelo sensor de obstáculo.

### 3.1.4 Desviar de obstáculos

Após a rota estabelecida o algoritmo de desvio de obstáculos começa a funcionar, mantendo o sentido programado até o sensor detectar algum obstáculo a uma distancia anteriormente programada, então ele assume a direção até que o obstáculo seja desviado, após o sensor não detectar mais nada é liberado para o algoritmo de sentido e velocidade.

## 3.2 Tecnologias Disponíveis no Mercado

No momento é possível identificar tecnologias parecidas em alguns drones, como por exemplo, o X25 Pro da marca Syma que possui um sistema GPS que permite que ele volte automaticamente para o ponto de partida quando a bateria esta acabando, ultrapassando o limite máximo de distancia do Radio Controle ou acionando pelo Controle remoto , por se tratar de uma tecnologia fechada, não é possível comparar os parâmetros e equipamentos utilizados para calcular o posicionamento e ativar o deslocamento, mas em seus anúncios ficam claro que sistema se baseia em sinal de geoposicionamento.

## 3.3 Restrições e Riscos do Projeto

### 3.3.1 Limitações Operacionais

Tabela - Lista de limitações operacionais

|  |  |
| --- | --- |
| LIMITAÇÃO | DESCRIÇÃO |
| Velocidade | Velocidade máxima dos motores está limitada a 20 km/h |
| Peso | Estrutura feita de metal, mas os motores possuem uma limitação especificada pelo fabricante, máximo -100 quilos. |
| Arduino | Ele conta com 32 KB de Flash (mas 512 Bytes são utilizados pro bootloader), 2 KB de RAM e 1 KB de EEPROM. Pode operar a até 20 MHz, porém na placa Arduino UNO opera em 16 MHz, valor do cristal externo que está conectado aos pinos 9 e 10 do microcontrolador. |
| Sensores | Devido ao tempo de resposta dos sensores e a sincronização com as respostas dos módulos, foram possíveis a instalação de apenas 2 componentes estes sendo instalado na frontal do protótipo. |

Fonte: Autor.

# 4. Bibliotecas utilizadas

## 4.1 SoftwareSerial.h

O hardware do Arduíno possui suporte interno para comunicação serial nos pinos 0 e 1 (que também vai para o computador via conexão USB). O suporte serial nativo acontece por meio de um hardware (incorporado ao chip) chamado UART. Esse hardware permite que o chip Atmega receba comunicação serial mesmo enquanto trabalha em outras tarefas, desde que haja espaço no buffer serial de 64 bytes.

A biblioteca SoftwareSerial foi desenvolvida para permitir a comunicação serial em outros pinos digitais do Arduino, usando o software para replicar a funcionalidade (daí o nome " SoftwareSerial "). É possível ter várias portas seriais de software com velocidades de até 115200 bps. Um parâmetro ativa a sinalização invertida para dispositivos que exigem esse protocolo.

## 4.2 TinyGPS.h

O TinyGPS foi projetado para fornecer a maior parte da funcionalidade do GPS NMEA que um usuário do Arduino desejaria - posição, data, hora, altitude, velocidade e curso - sem o tamanho grande que parece acompanhar corpos de código semelhantes. Para manter baixo o consumo de recursos, a biblioteca evita qualquer dependência obrigatória de ponto flutuante e ignora todos, exceto alguns campos-chave do GPS.

## 4.3 HMC5883L \_Simple.h

Essa biblioteca simplifica muito a obtenção de uma direção da bússola de um magnetômetro HMC5883L, incluindo o ajuste da declinação magnética local, que pode ser inserida diretamente em graus minutos e direção.

## 4.4. SPI.h

A Interface Periférica Serial (SPI) é um protocolo de dados serial síncrono usado pelos microcontroladores para se comunicar com um ou mais dispositivos periféricos rapidamente em curtas distâncias. Também pode ser usado para comunicação entre dois microcontroladores.

Com uma conexão SPI, sempre há um dispositivo mestre (geralmente um microcontrolador) que controla os dispositivos periféricos. Normalmente, existem três linhas comuns a todos os dispositivos.

## 4.5. Adafruit\_ST7735.h

Esta é a biblioteca gráfica principal de todos os displays adafruit, fornecendo um conjunto comum de primitivas gráficas (pontos, linhas, círculos etc.). Ele precisa ser emparelhado com uma biblioteca específica de hardware para cada dispositivo de exibição que transportamos (para lidar com as funções de nível inferior), requer a biblioteca Adafruit\_GFX para Arduino.

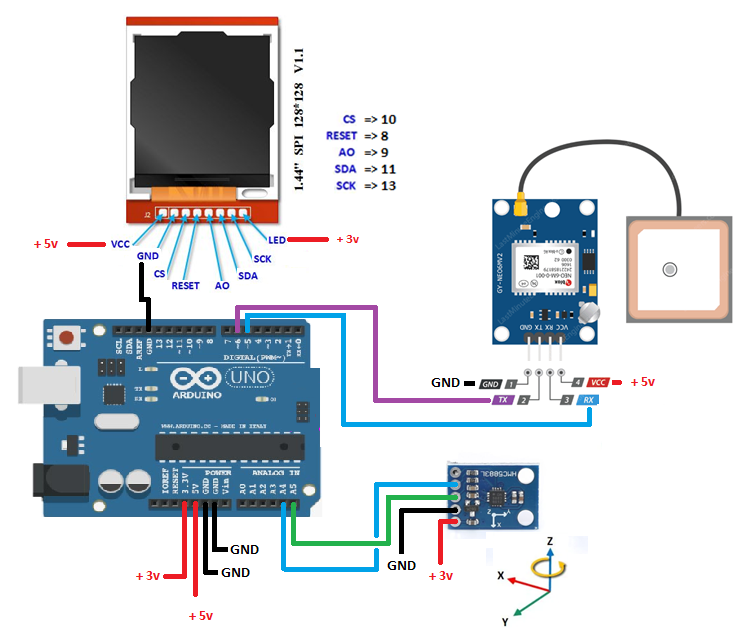
## 4.6. Adafruit\_GFX.h

Esta é a biblioteca gráfica principal de todos os displays adafruit, fornecendo um conjunto comum de primitivas gráficas (pontos, linhas, círculos etc.). Ele precisa ser emparelhado com uma biblioteca específica de hardware para cada dispositivo de exibição que transportamos (para lidar com as funções de nível inferior).

# 5. Arquitetura da aplicação

## 5.1. Ligação dos componentes Arduino1

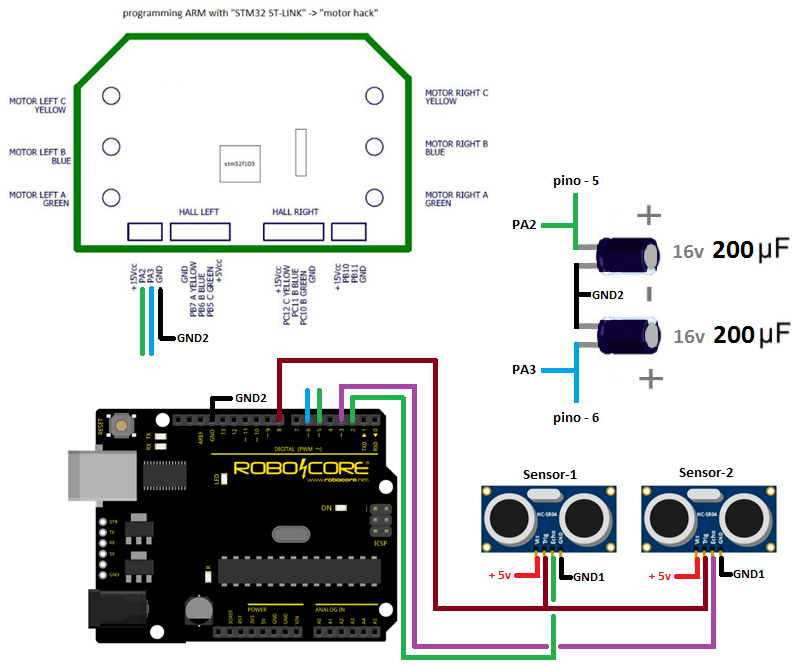
Figura - Ligação dos componetes ao Arduino1



Fonte: Autor.

## 5.2. Ligação dos componentes Arduino2

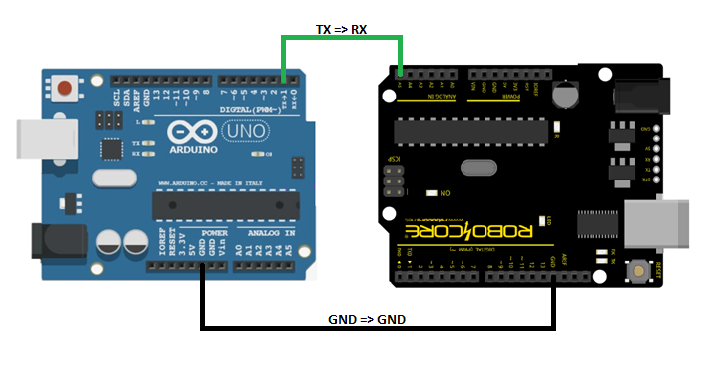
Figura - Ligação dos componentes ao Arduino2



Fonte: Autor.

## 5.3. Comunicação entre Arduino1 e Arduino2

Figura - Comunicação entre Arduino1 e Arduino2

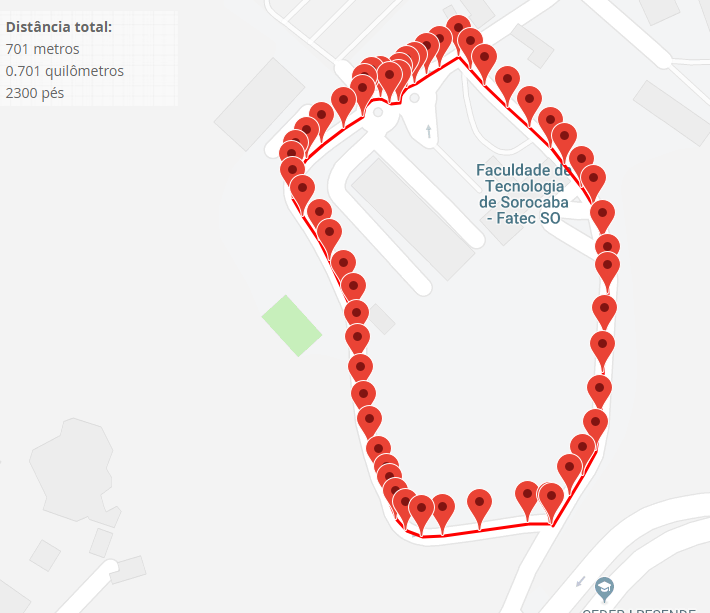


Fonte: Autor.

# 6. Percurso de Teste

Os pontos foram traçados com ajuda de uma API do google maps.

Figura - Percurso de teste



Fonte: Adaptado de https://www.google.com.br/maps

Tabela - Percurso de teste

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | Latitude | Longitude | **ID** | Latitude | Longitude |
| **1** | -23.480395 | -47.425745 | **51** | -23.481604 | -47.426829 |
| **2** | -23.480338 | -47.425799 | **52** | -23.481677 | -47.426818 |
| **3** | -23.480282 | -47.425854 | **53** | -23.481748 | -47.426798 |
| **4** | -23.480234 | -47.425911 | **54** | -23.481821 | -47.426779 |
| **5** | -23.480183 | -47.425969 | **55** | -23.481891 | -47.426756 |
| **6** | -23.480132 | -47.426029 | **56** | -23.48196 | -47.426728 |
| **7** | -23.480082 | -47.426087 | **57** | -23.482027 | -47.426694 |
| **8** | -23.480031 | -47.426145 | **58** | -23.482098 | -47.426672 |
| **9** | -23.479977 | -47.4262 | **59** | -23.482163 | -47.426634 |
| **10** | -23.479922 | -47.426255 | **60** | -23.482206 | -47.426573 |
| **11** | -23.479866 | -47.426307 | **61** | -23.482227 | -47.426497 |
| **12** | -23.479915 | -47.426399 | **62** | -23.482223 | -47.426414 |
| **14** | -23.479944 | -47.426469 | **63** | -23.482213 | -47.426334 |
| **15** | -23.479987 | -47.426533 | **64** | -23.482203 | -47.426255 |
| **16** | -23.48 | -47.426561 | **65** | -23.482192 | -47.426175 |
| **17** | -23.480013 | -47.426598 | **66** | -23.482181 | -47.426095 |
| **18** | -23.480049 | -47.426623 | **67** | -23.48217 | -47.426016 |
| **19** | -23.480082 | -47.426623 | **68** | -23.482161 | -47.425936 |
| **20** | -23.480105 | -47.426684 | **69** | -23.482155 | -47.425854 |
| **21** | -23.48008 | -47.426712 | **70** | -23.482154 | -47.425801 |
| **22** | -23.480071 | -47.426746 | **71** | -23.482124 | -47.425782 |
| **24** | -23.480077 | -47.426782 | **72** | -23.482069 | -47.425747 |
| **25** | -23.4801 | -47.426809 | **73** | -23.482005 | -47.425706 |
| **26** | -23.480135 | -47.426819 | **74** | -23.481943 | -47.425662 |
| **27** | -23.480152 | 47.426821 | **75** | -23.481886 | -47.425623 |
| **28** | -23.480166 | -47.426846 | **76** | -23.481815 | -47.425586 |
| **29** | -23.480192 | -47.426885 | **77** | -23.481742 | -47.42557 |
| **30** | 23.480234 | -47.42695 | **78** | -23.481674 | -47.425556 |
| **31** | -23.480277 | -47.427015 | **79** | -23.481595 | -47.425552 |
| **32** | -23.480322 | -47.427078 | **80** | -23.481522 | -47.425547 |
| **33** | -23.480373 | -47.427138 | **81** | -23.481448 | -47.425542 |
| **34** | -23.480429 | -47.427187 | **82** | -23.481374 | -47.425538 |
| **35** | -23.4805 | -47.427207 | **83** | -23.4813 | 47.425533 |
| **36** | -23.480547 | -47.427205 | **84** | -23.481227 | -47.425528 |
| **37** | -23.48059 | -47.427186 | **85** | -23.481153 | -47.425524 |
| **38** | -23.480634 | -47.427154 | **86** | -23.48108 | -47.425519 |
| **39** | -23.480696 | -47.42711 | **87** | -23.481005 | -47.425514 |
| **40** | -23.480761 | -47.427069 | **88** | -23.480961 | -47.425511 |
| **41** | -23.480826 | -47.427029 | **89** | -23.480917 | -47.425518 |
| **42** | -23.48089 | -47.426992 | **90** | -23.48086 | -47.425527 |
| **43** | -23.480958 | -47.426959 | **91** | -23.480804 | -47.42554 |
| **44** | -23.481093 | -47.426896 | **92** | -23.480723 | -47.425558 |
| **45** | -23.481142 | -47.426878 | **93** | -23.480642 | -47.425577 |
| **46** | -23.481236 | -47.426861 | **94** | -23.480588 | -47.425597 |
| **47** | -23.48131 | -47.426857 | **95** | -23.480541 | -47.425628 |
| **48** | -23.481385 | -47.426854 | **96** | -23.480479 | -47.425679 |
| **49** | -23.481458 | -47.426846 | **97** | -23.480398 | -47.425742 |
| **50** | -23.481532 | -47.426839 |  |  |  |

Fonte: Autor.

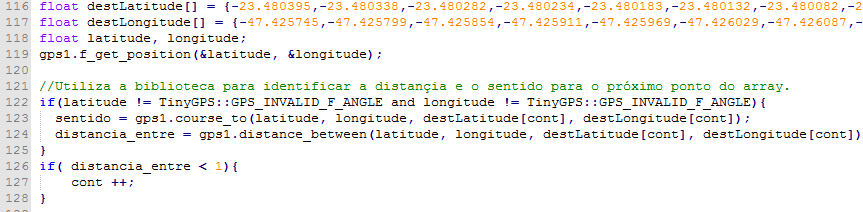
# 7. Exemplos do código

Para fins de explicação foi retirado trechos de partes cruciais do código, a fim de exemplificar a lógica por trás da ideia principal do protótipo.

## 7.1 Algoritmos de posicionamentos

Nas linhas 116, 117, 118 são declaradas as variáveis utilizadas para identificar o sentido e a distância a ser percorrida até o próximo elemento do array, em seguida é verificado se existe dados corretos para o cálculo, então são passados para a biblioteca TinyGPS os valores das variáveis, e como resposta temos o sentido e a distância até o próximo ponto, e por fim na linha 126 temos uma comparação entre a distância para identificar se o trajeto da posição cont foi concluído.

Figura - Algoritmos de posicionamentos

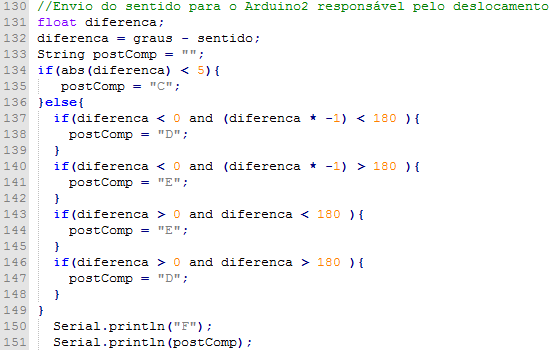


Fonte: Autor.

## 7.2 Algoritmos de envio da direção

Aqui são feitas comparações entre o ângulo de deslocamento, para assim estabelecer a mais precisa tomada de direção, em seguida é setado pela porta TX do arduino1 para a porta RX do arduino2 que então interpreta da seguinte maneira direção F => frente e o lado a ser deslocado onde D => direita e E => esquerda.

Figura - Algoritmos de posicionamentos

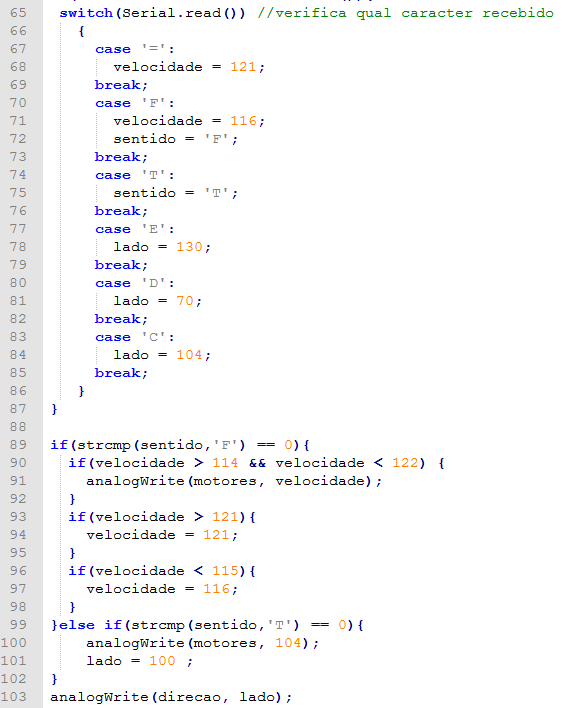


Fonte: Autor.

## 7.3 Algoritmos de velocidade

Esta parte do código se encontra no arduino2, responsável pelo envio dos dados para a placa hoverboard através de pulsos PWM em conjunto com uma linha de capacitores responsáveis por filtrar ruídos e assim estabelecer uma CC (corrente continua) para controlar os motores, para os testes foi identificado a necessidade de uma velocidade de 5km/h para garantir a segurança dos equipamentos em caso de perda de controle, assim sendo a velocidade => 121 equivale a 5km/h e velocidade => 104 para os motores.

Figura - Algoritmos de posicionamentos

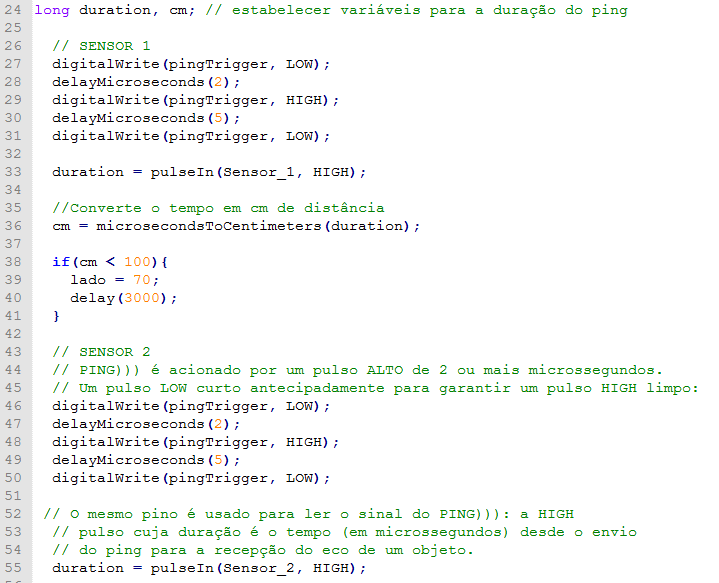


Fonte: Autor.

## 7.4. Algoritmos de desvio de obstáculos

O código de desvio de obstáculos tem prioridade sobre todos os eventos dos algoritmos, para evitar que um acidente possa ocorrer, quando um objeto é identificado na frente do carro ele aborta a velocidade e gira para o lado oposto do objeto, assim que o objeto não se encontrar mais no radar ele devolve o controle para o arduino1.

Figura - Algoritmos de posicionamentos



Fonte: Autor.

# 8. Considerações Legais

O resultado gerado por este projeto possui fins acadêmicos, assim sendo, é liberada a cópia e/ou reprodução sem a prévia autorização de seu desenvolvedor, por quaisquer meios ou processos existentes. Em caso de acidentes, o autor está isento de quaisquer responsabilidades.

# 9. Código fonte

O código fonte e imagens desenvolvidas durante o projeto estão disponíveis no DVD em anexo, e no repositório “https://github.com/JonyAlan/TG-Prot-tipo-carro-aut-nomo”.

# 10. Conclusão

Se tratando de um protótipo conclui-se que ainda não é capaz de percorrer grandes trajetos, e percursos com grandes obstáculos isso sem as análises e consequentemente alterações das configurações, mais precisamente por conta dos sensores de obstáculos que tem posicionamento fixo e frontal, e do tamanho das rodas, pois podem ficar presas em crateras ou mesmo em desníveis no solo.

Para resolver o primeiro caso, seria interessante implantar sensores com subsistemas ligados à Machine Learning para reconhecimentos de padrões, para que possa identificar obstáculos simulando uma visão humana.

Para o segundo caso é preciso reestruturar a parte física do protótipo, primeiro fazendo com que possua uma extensão maior para que possa sustentar rodas maiores e em seu interior maior poder de processamento com mais equipamentos, por exemplo carregar um computador para processamento dos dados.

Outra item que precisa ser melhorado é a forma com que o percurso é inserido no protótipo, momentaneamente sendo através de um array de coordenadas GPS, sendo necessário pegar estas coordenadas previamente através de sites como o “google maps” site este que foi utilizado, para solucionar é possível utilizar de uma Api para traçar a rota através do início e fim do trajeto e então enviar direto para o protótipo para que ele possa iniciar seu percurso.

Descrevendo as etapas bem sucedidas do projeto tem-se itens como a precisão do posicionamento através do Módulo GPS GY-GPS6MV2 com coordenadas enviadas sem interferência a da um segundo e o sentido estabelecido pelo módulo HMC5883L - bússola eletrônica cuja precisão se dá entre 1 e 2 graus e através desses itens foi preciso desenvolver um algoritmo capaz de percorrer trajetos sem o controle humano ou pelo menos simular isso, também é preciso reconhecer que apesar de pequeno e com pouca capacidade de processamento o microcontrolador ATEMEGA 328p acoplado ao arduino supriu as necessidades momentâneas deste protótipo recebendo os dados processando e enviando os comandos para a placa mãe do hoverboard responsável pelos motores do protótipo.

Não foi identificado se empresas que estão na corrida do desenvolvimento dos carros autônomos como Google, Apple, Hyundai, Tata Elixsi, Tesla, Toyota, Uber, dentre outras e também a brasileira IARA, utilizam de um sistema parecido para traçar suas rotas, como era de se esperar os códigos e grande parte de sua tecnologia permanece em segredo já que isto tende a ser mais uma grande revolução nos sistemas de transportes do mundo.

**Bibliografia**

**Bibliotecas arduino**. Site, 2018. Disponível em: <https://www.arduino.cc/reference/pt/language/structure/further-syntax/include/>. Acesso em: 10 de out. de 2019.

**Dibble**, Drew Dibble, How I hacked the Self Balancing Scooter. Tecnoblog, 2019. Disponível em: <<http://drewspewsmuse.blogspot.com/>> Acesso em: 10 de jun. de 2019.

**Laboratório de Garagem**, Bússola eletrônica com HMC5883L. Blogs, 2019. Disponível em: <http://labdegaragem.com/profiles/blogs/tutorial-bussola-eletronica-com-hmc5883l>Acesso em: 20 de jun. de 2019.

**LANA**, Hellynson Cásio. **Projetos Maker:**Arduino, Eletrônica, Robótica, Automação residencial. São Paulo: Novatec, 2018. 208 p. (1).

**MCROBERTS**, Michael. **Arduino Básico.**2. ed. [eua]: Novatec, 2015. 46 p.

**Simova**, Isabelle Simova, Reprogramming hoverboards for robots. Site, 2019. Disponível em:<https://hackaday.io/project/158256-hoverbot/log/146067-reprogramming-hoverboards-for-robots> Acesso em: 15 de jun. de 2019.

**Glossário**

**GPS -** Global Positioning System (Sistema de Posicionamento Global).

**LA** – Latitude (Angulo entre o plano do equador à superfície de referência).

**LO** - Longitude (Distância entre um ponto e o Meridiano de Greenwich).

**BPS** - bits per second (bits por segundo).

**RX** - Receiver (portas digitais utilizadas para receber dados via comunicação serial).

**TX** - Transmitter (portas digitais utilizadas para enviar dados via comunicação serial).

**Machine Learning –** (aprendizado de máquina) subcampo da Engenharia e da ciência da computação que evoluiu do estudo de reconhecimento de padrões e da teoria do aprendizado computacional.

**GND** – Ground – Terra ou polo negativo.

**Hardware** - Termo técnico que foi traduzido para a língua portuguesa como equipamento, pode ser definido como um termo geral da língua inglesa que se refere a equipamentos físicos.

**I2C -** Barramento serial Barramento multimestre desenvolvido pela Philips que é usado para conectar periféricos de baixa velocidade a uma placa mãe, a um sistema embarcado ou a um telefone celular.

**Firmware -** Classe específica de software de computador que fornece controle de baixo nível para o hardware específico do dispositivo.

**PWM -** Pulse Width Modulation ou Modulação de Largura de Pulso, através da largura do pulso de uma onda quadrada é possível o controle de potência ou velocidade.