FACULDADE DE TECNOLOGIA DE AMERICANA "MINISTRO RALPH BIASI"

Curso Superior de Tecnologia em Jogos Digitais

Bruno De Andrade Patrocinio

DISPOSITIVO PARA CONTROLE DE INCLINAÇÃO NA MONTAGEM DE VIDROS AUTOMOTIVOS

Americana, SP

2020

FACULDADE DE TECNOLOGIA DE AMERICANA "MINISTRO RALPH BIASI"

Curso Superior de Tecnologia em Jogos Digitais

BRUNO DE ANDRADE PATROCINIO

DISPOSITIVO PARA CONTROLE DE INCLINAÇÃO NA MONTAGEM DE VIDROS AUTOMOTIVOS

Trabalho de Conclusão de Curso desenvolvido em cumprimento à exigência curricular do Curso Superior de Tecnologia em Jogos Digitais, sob a orientação do Prof. Me. Benedito Aparecido Cruz

Área de concentração: Programação

Americana, SP

2020

BRUNO DE ANDRADE PATROCINIO

DISPOSITIVO PARA CONTROLE DE INCLINAÇÃO NA MONTAGEM DE VIDROS AUTOMOTIVOS

Trabalho de graduação apresentado como exigência parcial para obtenção do título de Tecnólogo em Jogos Digitais pelo CEETEPS/Faculdade de Tecnologia – FATEC/ Americana.

Área de concentração: Programação

Americana, 02 de dezembro de 2020.

Banca Examinadora:

_____

Me. Benedito Aparecido Cruz (Presidente)

Mestre

Faculdade de Tecnologia de Americana

_____

Me. Silvia Aparecida José e Silva (Membro)

Mestre

Faculdade de Tecnologia de Americana

_____

Dr. Renato Kraide Soffner (Membro)

Doutor

Faculdade de Tecnologia de Americana

AGRADECIMENTOS

Agradeço ao Prof. Dr. Daives Arakem Bergamasso pela grande dedicação ao presente trabalho na forma de orientações gerais.

Ao meu amigo Valdeci Ferreira Bertonha, agradeço o apoio fornecido em materiais que auxiliaram na montagem dos protótipos .

Agradeço a minha esposa Larissa Aparecida Fernandes Hidalgo Patrocinio por toda a motivação e auxilio nas pesquisas acadêmicas e normatizações de formatação do trabalho.

"Talk is cheap. Show me the code"

(Linus Torvalds)

RESUMO

Nos dias atuais pode se perceber diversos avanços na área de tecnologia, e é possível observar que em áreas educacionais esse avanço se encontra presente. A comunidade de T.I (Tecnologia da Informação) possui de modo geral inúmeras iniciativas *open-source* ou *software* livre que certamente auxiliam diretamente no avanço de desenvolvimento de aplicações tecnológicas e o acesso à informação. Na busca de uma aplicação que visasse atenuar e auxiliar a diminuir o nível de defeitos gerados no processo produtivo de montagem de vidros automotivos, foi escolhido para atuar sobre o problema proposto, uma plataforma *open-source* para desenvolvimento de protótipos eletrônicos e robótica, muito utilizada para fins educacionais nas mais diversas esferas de ensino, o Arduino. A solução se deu na construção de três protótipos-base com a mesma finalidade básica, auxiliar os montadores do processo produtivo a atenuarem as inclinações excessivas durante a montagem dos vidros, sendo posteriormente implementado funcionalidades adicionais para obtenção de uma melhoria contínua no processo. O projeto buscou implantar uma metodologia no que diz respeito a pesquisa de um método para implantar uma interface de usuário gráfico, que possibilitasse a supervisão dos dados gerados e de uma representação do dispositivo. Após a implementação da comunicação da aplicação por radiofrequência, foi observado que o protótipo possui potencial para se tornar um projeto voltado para IoT (Internet das Coisas), além de outras melhorias voltadas ao código fonte do dispositivo, incluindo a filtragem dos dados lidos do acelerômetro, pelo filtro de Kalman, inclusão do controle em um eixo adicional, o eixo Y e melhorias na representação gráfica do dispositivo, efetuado no Canvas da aplicação desenvolvido em Python.

Palavras-chave: Controle; Inclinação; Arduino.

ABSTRACT

Currently it is possible to realize several advances at technology field, and it is possible to observe in educational areas this advance is located. The I.T community (Information Technology), has in general purpose countless open-source or free software initiatives that surely directly helps the search development of technologic applications and information access. When looking for an application which would smooth and reduce the defects about the automotive glasses assemble, it was chosen to act on the problem, the open-source platform to development of electronic and robotic prototypes, very used in education field and in the most diverse areas of teaching, the Arduino. The Solution for the purposed issue was to build three prototypes with the same properties and goals, to help the production process assemblers to prevent or smooth excessive inclinations during the glasses assemble, and afterwards implementing additional functionalities, trying to obtain a continuous improvement in the process. The project sought to implement a methodology of searching a way to implant a user graphic interface that could keep track of the process on a computer, the generated data. After the Radio frequency communication implementation, it was observed that prototype has potential to become an IoT project (Internet of Things), in additional, other source codes improvement of the dispositive may be applied, including some filter enhancements of the accelerometer data axis, using Kalman's Filter and making use of the Y axis. Improvements in the Canvas, inside the user's interface, developed in Python may be applied too.

Keywords: Control; Inclination; Arduino.

LISTA DE FIGURAS

SUMÁRIO

1. INTRODUÇÃO

Problemas de montagem em processos automobilísticos podem ocorrer, pois em muitos casos em uma linha de produção são encontrados uma mescla de processos automatizados e manuais, onde tanto existem falhas humanas quanto de equipamentos. Na montagem manual de para-brisas automotivos dianteiros e traseiros, se faz necessário possuir cautela e atenção durante a montagem, pois os vidros são peças que possuem um custo considerável e que com base nas observações ao longo de anos de montagem, foram vistos inúmeros sucateamentos de peças e defeitos que geram retrabalhos demorados, ocasionados por falhas operacionais. Felizmente existem soluções parciais ou totais dos mais diferentes tipos e valores. Pode se automatizar o processo por completo, eliminando por completo o fator humano, provavelmente por um alto custo, ou também é possível criar uma solução criativa mais barata e que utilize da experiencia exercida pelos profissionais do processo.

Nos dias de hoje existem plataformas de desenvolvimento de prototipagem de microcontroladores que oferecem baixo custo e fácil aprendizagem. Uma delas é o Arduino que oferece uma boa compatibilidade com uma infinidade de componentes, programação bastante similar a linguagem C, que é muito disseminada, além disso é *opensource*. A facilidade de acesso e grande quantidade de material útil disponível pela internet, aliado aos fabricantes clones que se baseiam na plataforma do Arduino, fazem com que seja muito relevante decidir a favor da utilização destes recursos para desenvolvimento de protótipos de soluções de problemas de automação e controle, projetos educacionais, entre outros.

Para a atuação no presente trabalho, foi decidido pela utilização da plataforma Arduino, utilizando de maneira agregada componentes adicionais, auxiliando e fornecendo dados para o processamento de funções que visassem atuar no problema encontrado, explanando alguns conceitos básicos necessários para o bom entendimento do trabalho, a evolução dos protótipos desenvolvidos e a apresentação dos esquemas de montagem, assim como o código desenvolvido, buscando deixar o leitor mais confortável com o tema.

1.1 Objetivos

O desenvolvimento dos protótipos deste trabalho faz com que se busque uma solução, otimização de um processo ou simplesmente uma melhoria aplicada que reduza ocorrências de problemas que ocasionem defeitos de qualidade de montagem. No que diz respeito a este trabalho, a ocorrência a ser tratada é um defeito em potencial visto em um processo fabril na indústria automobilística, na montagem de vidros automotivos.

Basicamente, a meta do trabalho é desenvolver um projeto que atue na redução de defeitos relacionados a infiltrações de água na montagem de vidros automotivos, por meio da emissão de alertas sonoros e visuais nas inclinações que ofereçam riscos de defeitos. Desenvolver um sistema que mostre os dados para os usuários do dispositivo e que os dados possuam a possibilidade de serem visualizados a uma certa distância, não estabelecida, através de comunicação via radiofrequência e uma representação de uma imagem que altera sua posição conforme o equipamento inclina. Além dessas características, também é objetivo deixar relatado possíveis trabalhos futuros que visem o aprimoramento dele.

1.2 Justificativa e Motivação

A decisão de atuar sobre o tema e elaborar um projeto para atuar sobre, veio da observação do problema e a possibilidade de aplicar conhecimento técnico na elaboração de um trabalho de melhoria de processo produtivo, sendo uma boa oportunidade para praticar o conhecimento adquirido ao longo da matriz curricular do curso de Tecnologia em Jogos Digitais, que oferece uma boa bagagem para atuar em diversas frentes.

Pode se compreender como motivação para este trabalho a realização da aplicação do conhecimento em benefício da empresa em que foi feita a sugestão de melhoria e de crescimento pessoal mediante da prática do conhecimento.

2. REVISÃO BIBLIOGRÁFICA

2.1 Tecnologia a Serviço da Humanidade

A tecnologia faz parte do cotidiano da humanidade há muitos séculos. A ampliação do conhecimento científico de modo a fornecer métodos e ferramentas para a resolução de problemas, emancipação do conforto, bem estar da espécie humana e outras diversas aplicações que não cabem a este trabalho detalhar mais precisamente são motivações que fazem com que cada vez mais a raça humana busca o saber científico.

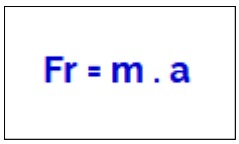
O processo tecnológico, aliado a inovações ao longo do tempo desde a revolução industrial, modelou a economia no decorrer dos séculos XVIII e XIX. A economia contemporânea atualmente conta com avanços muito significativos em diversas áreas, os quais podem ser citados com maior ênfase o de telefonia e computação. A computação passou por sucessivas evoluções durante décadas dentro do século XX, em que os primeiros computadores funcionavam através de válvulas eletrônicas e consumiam uma abundante quantidade de energia. Algumas gerações depois, durante a década de 70, geração que fez uso dos famosos circuitos integrados, que a população começou a ter acesso dos computadores pessoais. Atualmente um dos maiores objetivos dos desenvolvedores de processadores, além do ganho de processamento é a economia de gasto energético.

2.2 Conceitos de Aceleração

A aceleração é uma grandeza física ou simplesmente um fenômeno estudado pelas ciências naturais. Essa força da natureza é explicada pela segunda lei de Isaac Newton, físico brilhante do século XVII e XVIII, formulador da teoria da gravitação e das três leis do movimento, conhecidas também por “leis de Newton”. Especificamente, a segunda lei de Newton está relacionada com a aceleração. Uma

boa explicação simples pode ser entendida como “a definição técnica de aceleração é a taxa de mudança de velocidade em relação ao tempo” (EVANS et al,2016, p.251). De acordo com essa lei “A soma das forças que atuam sobre um corpo é igual ao produto de seu coeficiente de inércia pela sua aceleração” (ANTUNES; GALHARDI; HERNASKI, 2018, p.2). Pode ser visto a equação representada na Figura 1.

Figura 1 – Equação da 2° lei de Newton.



Fonte: Própria autoria

A equação acima é descrita por força resultante (Fr), massa (m) e aceleração (a). A aceleração, como visto anteriormente, é a variação de velocidade em relação ao tempo. Para alterar o estado do objeto é necessário aplicar uma determinada força sobre ele e dependerá da massa deste objeto. A unidade de medida da força aplicada é o Newton (N), atribuído pelo Sistema Internacional de Unidades (SI). Se for aplicado uma força de 2N em um objeto com uma massa de 1 Kg, será gerado uma aceleração resultante de 2m/s, ao passo que se o objeto tiver uma massa inferior, como por exemplo 0,5 Kg, a aceleração resultante será bem superior, sendo de 4m/s, conforme a fórmula.

Dessa maneira, atualmente, existem diversos componentes eletrônicos que possuem a função de atuar na medição de acelerações. Conhecidos como acelerômetros, eles podem ser de três ou seis eixos e podem medir “a aceleração como sendo a variação no movimento angular do controlado, e ele pode detectar mudanças de ângulo muito pequenas” (EVANS et al,2016, p.251). Esses aparatos eletrônicos são perfeitos para muitas finalidades, podem oferecer a quem desejar manipulá-lo, juntamente com um microcontrolador, muitas formas de resolução de problemas com automação.

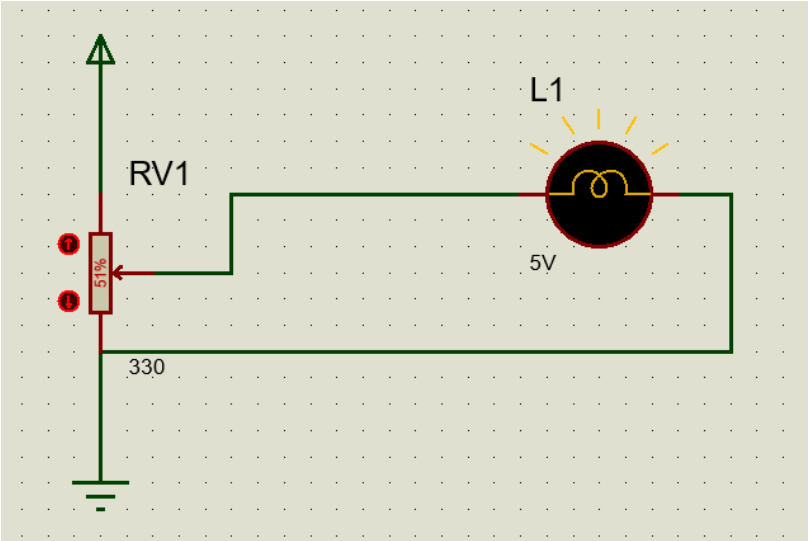
2.3 Circuitos Elétricos

Um circuito elétrico é um circuito ou trajeto fechado com uma série de componentes agregados a ele, como resistores, interruptores, transistores, entre diversos outros. Os circuitos eletrônicos possuem as mesmas definições, porém trabalham na maior parte dos casos em baixa tensão, podendo ser classificados entre circuitos de tensão e corrente contínua ou alternada. Dessa maneira, conforme Alexander e Sadiku (2013, p.4) “Um circuito simples [...], é formado por três elementos básicos: Uma bateria, uma lâmpada e fios para interconexão”.

A montagem de circuitos eletroeletrônicos possui um papel preponderante em diversos projetos que executam prototipagem com microcontroladores, podendo chegar a ser complexos inclusive. “Os circuitos elétricos são usados em inúmeros sistemas elétricos para realizar diferentes tarefas.” (ALEXANDER; SADIKU, 2013, P.4).

A imagem do esquema, apresentado na Figura 2, desenvolvido no Proteus 8 demonstra um circuito simples, para exemplificação, com uma lâmpada com tensão nominal de 5 volts, ligada a uma fonte de alimentação com o respectivo ponto de aterramento e um potenciômetro para controlar ou limitar o fluxo de corrente.

Figura 2 – Esquema de um circuito básico eletrônico.



Fonte: Própria autoria

Um potenciômetro é um dispositivo eletrônico muito usado em diversos projetos e aparatos eletrônicos e demonstra muito bem o uso de um controle analógico de um dispositivo, que pode ter diversas funções.

Os circuitos eletrônicos podem ser muito complexos. É exigido aos projetistas que conheçam bem as mais diversas nomenclaturas e representações técnicas para a elaboração de um circuito eletrônico.

A imagem apresentada na Figura 2 é apenas um exemplo bem básico destes circuitos. A realidade de desenvolvedores e projetistas de dispositivos eletrônicos, que visam solucionar problemas encontrados em diversas áreas de atuação profissional, inevitavelmente acaba passando por um aliado poderoso no campo de processamento de informações, os circuitos integrados e microcontroladores.

2.3.1 Microcontroladores e internet das coisas

Um microcontrolador pode ser comparado a um computador, pois ele contém tudo o que um computador simplificado possui, podendo ser destacados a memória, o processador, os pinos de entrada e saída. É importante ressaltar que existem diversos fabricantes de microcontroladores e que estes pequenos dispositivos possuem uma gama de tipos e capacidades. Seus usos estão em praticamente tudo hoje em dia, como automóveis, eletrodomésticos, equipamentos industriais, entre uma grande quantidade de produtos encontrados no cotidiano da população.

Os microcontroladores utilizam linguagens de programação para fazer a comunicação com o hardware, para execução das tarefas desejadas pelo programador. Na maior parte dos casos de uso de microcontroladores, a linguagem mais difundida para a programação é a linguagem C. C é uma linguagem de programação compilada e estrutural desenvolvida entre 1969 e 1973 no AT&T Bell Laboratories.

Os microcontroladores quando agregados com outros componentes eletroeletrônicos, incluindo dispositivos eletromecânicos, servo motores, motores DC (que converte corrente contínua em capacidade de trabalho, ou energia mecânica), podem fazer uma espécie de envolvimento entre áreas do conhecimento cientifico e

engenharias para a criação de projetos diversos. Os microcontroladores podem dar vida a robôs que executam tarefas manuais, dispositivos de acessibilidade, automação residencial, podendo estar presentes em caixas eletrônicos, aviões, carros, entre outros. Em suma, estão presentes em sistemas projetados para executar funções específicas únicas, conhecidos como Sistemas Embarcados.

Existem vários kits de desenvolvimento que são criados para auxiliar a criação de protótipos de projetos que possam ser aperfeiçoados e testados para posteriormente se tornar um produto ou sistema de controle e automação. Os Kit PICGenios PIC18F e PIC16F da Microchip e o Arduino são exemplos destes kits, muito utilizados para fins educacionais.

Sabendo se da grande utilização dos microcontroladores, existem áreas em que estes dispositivos podem oferecer uma grande quantidade de possibilidades de conexão e interação com outros dispositivos e pessoas. A Internet das coisas (IoT) é um grande exemplo de um futuro certo destas atuações.

> Internet das Coisas é muito mais que apenas ligar lâmpadas pelo smartphone. Não é somente ligar as “coisas” pela internet, mas também torná-las inteligentes, capazes de coletar e processar informações do ambiente ou das redes às quais estão conectadas. (OLIVEIRA, 2017, p15).

A sociedade moderna utiliza vários dispositivos diariamente que fazem uso de conexão com a Internet. A Internet das Coisas certamente é um tema que cada vez mais estará presente no cotidiano da população, mantendo as pessoas conectadas com uma grande quantidade de dispositivos que fazem uso de uma série de sensores que se comunicam com outros dispositivos que estejam conectados remotamente e interagem com um usuário para auxiliá-lo melhor em suas atividades. Os microcontroladores podem fazer uso desta rede para aprimorar e possibilitar um uso mais eficaz dos produtos que são usados diariamente pelas pessoas e indústrias. Uma maneira mais eficiente e personalizada, que atenda um uso mais avançado de produtos, equipamentos e eletrodomésticos tradicionais pode ser alcançado em dias atuais e existem inúmeros microcontroladores e placas para uso, que oferecem possibilidades para o desenvolvimento dessas soluções, acoplados com componentes que possibilitam a conexão com o mundo externo. Uma série de produtos e equipamentos como ares-condicionados, fornos, geladeiras, televisores, equipamentos industriais enquadram-se como exemplos de produtos que a internet

das coisas está presente, tornando-os mais prestativos, inteligentes, agregando valor ao produto e claro maior conforto e comodidade para os usuários.

Para o uso e criação de soluções voltadas a ampla interatividade, existem modelos de baixo custo que oferecem interfaces *WiFi*, que possuem um papel fundamental para Iot (*Internet of Things*). Oliveira diz (2017, p.17) "O microcontrolador ESP8266 da Espressif [...] é um circuito totalmente integrado, com interfaces de I/O digitais e analógicas, e ainda interface Wi-Fi, com um processador de 32 bits [...]". No que diz respeito baixo custo, tanto o Arduino e seus clones quanto o ESP8266 são vantajosos, podendo ser adquirido por pouco dinheiro.

O conceito de IoT está presente em muitas plataformas. Como já citado o ESP8266 e o NodeMCU, que também é baseado na arquitetura do Esp8266 possuem embutido um suporte a WiFi (Oliveira, p.18). O Arduino, através de módulos adicionais acoplados, possibilita a interatividade com os outros dispositivos pelo meio de conexões com uma rede também. Conforme dito por Oliveira (2017, p.18) "As aplicações com suporte a rede WiFi e a todos os módulos desenvolvidos no Arduino abrem muitas possibilidades de desenvolvimento de aplicações". Dessa maneira, é possível utilizar um módulo que faz uso da arquitetura de rede Ethernet, além do WiFi.

Em suma, "A internet das Coisas, em poucas palavras, nada mais é que uma extensão da internet atual, que proporciona aos objetos do dia a dia [...], mas com capacidade computacional e de comunicação, se conectarem a internet" (SANTOS et al.,2016, p.2).

2.4 Arduino

"O conceito Arduino surgiu na Itália, em 2005, com o objetivo de criar um dispositivo de controle para protótipos construídos de forma menos dispendiosa do que outras soluções disponíveis" (FONSECA; VEGA, 2011, p.3).

O Arduino possui vários recursos, além de ser uma placa voltada para o desenvolvimento de aplicações para fins educacionais e construção de protótipos. São encontrados no mercado vários kits de desenvolvimento voltados para os mais diversos níveis educacionais. Possui conjuntos para iniciantes, estudantes do ensino

regular e engenheiros, no site oficial do produto possui maiores informações (www.arduino.cc/education).

> [...] o Arduino é uma pequena placa de microcontrolador que contém um conector USB que permite ligá-la a um computador, além de diversos pinos que permitem a conexão com circuitos eletrônicos externos, como motores, relés, sensores luminosos, diodos laser, alto-falantes, microfones, etc.(MONK SIMON, 2014, p1).

Esta pequena placa de desenvolvimento possui várias versões, sendo os mais conhecidos o Leonardo e o Uno. Dependendo do nível de complexidade de um projeto, pode ser necessário o desenvolvimento em uma placa com uma quantidade de recursos maior. Para isso, existe uma versão do Arduino, conhecida como Mega. Está na sua versão 2560 Rev3 e possui vantagens em relação aos demais modelos, como maior *layout*, maior quantidade de pinos de entrada e saída, totalizando 54, mais poder de processamento em seu microprocessador, o ATmega2560 e uma memória flash de 256 KB, o dobro em relação aos 128 KB do Mega anterior, entre outras coisas. Para Mc Roberts (2018, p.4): " A maior vantagem do Arduino em relação a outras plataformas de desenvolvimento de microcontroladores é sua facilidade de utilização", essa facilidade faz com que pessoas que não possuem um conhecimento amplo de áreas da engenharia possam encontrar uma maior facilidade para criar projetos. O Uno possui 14 pinos digitais programáveis tanto quanto entrada e saída, sendo que seis destes pinos podem ser modulados por largura de pulso (PWM). Evans, Noble e Hochenbaum relatam (2013, p.26): " Diversos protocolos de comunicação estão disponíveis, incluindo serial, bus serial de interface periférica (SPI) e I2C/TWI". Uma outra vantagem do uso deste *hardware* é o seu baixo custo e a facilidade de encontrar arquiteturas similares ao Arduino oficial e a compatibilidade oferecida. É uma plataforma *open-source* e existem diversas placas baratas que compartilham o mesmo ambiente integrado de desenvolvimento para a programação, que é a IDE do Arduino e pode ser baixado no site oficial do produto (www.arduino.cc). As placas podem ser vistas nas figuras 3 e 4.

Evans, Noble e Hochenbaum ainda observam (2013, p.27): "O Arduino Uno é uma boa opção multiuso e é sua melhor aposta para uma placa de partida com fonte de alimentação auto chaveada e tensão integrada de 3,3 V regulada".

Como poderá ser observado, este trabalho inicia se no Uno e acaba evoluindo o projeto e optando por outra versão em algum momento compacta, e posteriormente mais abrangente no quesito recursos disponíveis.

Figura 3 - Arduino Uno Rev3

Fonte: Disponível em: www.arduino.cc

Figura 4 – Arduino Mega2560 Rev3

Fonte: Disponível em: www.arduino.cc

É possível que em relação a peso e facilidade de uso, não haja muita diferença entre a versão Mega 2560 e o Uno, além de não possuírem diferenças consideráveis entre tamanho da placa.

2.4.1 Apresentação Básica da IDE

Para Alves et al.(2012, p.167), é evidente que “O projeto Arduino também envolve um ambiente de desenvolvimento integrado [...] para geração dos programas (denominados de *sketches*) que serão enviados para a placa eletrônica”. Em geral é uma interface de fácil utilização para seus usuários, inclusive aqueles que não possuem muita experiência com sua utilização. A linguagem que a plataforma utiliza para desenvolvimento dos programas que se comunicam com o Arduino, é toda baseada em uma das linguagens mais utilizadas na atualidade ainda, o C++. Para todo usuário que pretenda desenvolver projetos com a plataforma Arduino e seus clones, é necessário um conhecimento no mínimo básico de programação e que esteja atento a sua documentação de referência.

> O IDE do Arduino foi desenvolvido em linguagem JAVA baseado no projeto *Processing,* na bibliotec AVR-gcc (para microcontroladores da família AVR) e em outros softwares livres. A linguagem de programação do Arduino é baseada no projeto Wiring e pode rodar nas plataformas Windows e Linux. (ALVES et al., 2012, p.167).

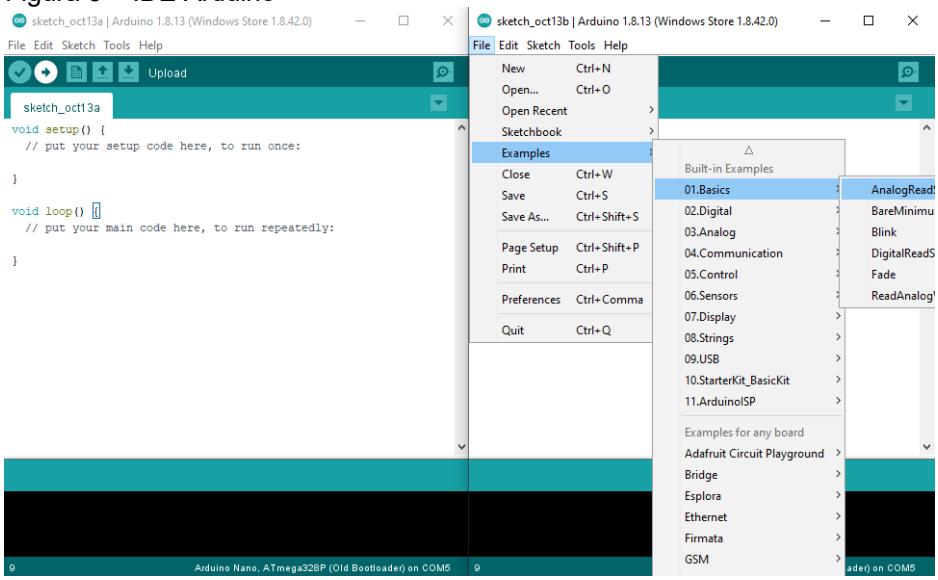
Ambos os projetos, Wiring e Processing podem ser encontrados em maiores detalhes em *http://processing.org/* e *http://wiring.org.co/* respectivamente.

A plataforma ainda oferece um recurso interessante no que diz respeito ao processo de aprendizagem e uso, o *Arduino Web Editor.* Um ambiente online em que é possível utilizar os recursos de diversas plataformas disponíveis (Uno, Mega, Mini), incluindo suas bibliotecas de recursos e componentes. Para maiores informações consultar a página direta do editor, em *https//create.arduino.cc/editor*.

O código fonte é inserido na área da janela onde é possível ver duas funções básicas iniciais em praticamente qualquer programa desenvolvido com Arduino, que são *Start* e *Loop.* Em Start é inserido geralmente tudo o que seja necessário para o programa iniciar e em Loop fica instaurado o processo de repetição infinito até que o programa seja interrompido por um outro processo, em outras palavras, uma espécie de *While(True).* Na Figura 5, é possível verificar o local específico onde são escritos os programas no modelo de *Sketch* e mais ao lado na aba *File > Examples,* pode-se

encontrar diversos exemplos iniciais para aprendizagem do funcionamento da plataforma. As imagens abaixo buscam auxiliar na visualização.

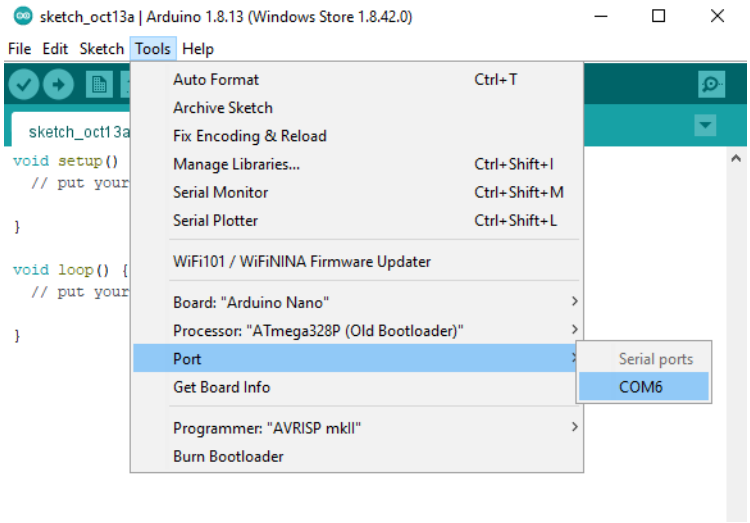
Figura 5 – IDE Arduino



Fonte: Própria autoria

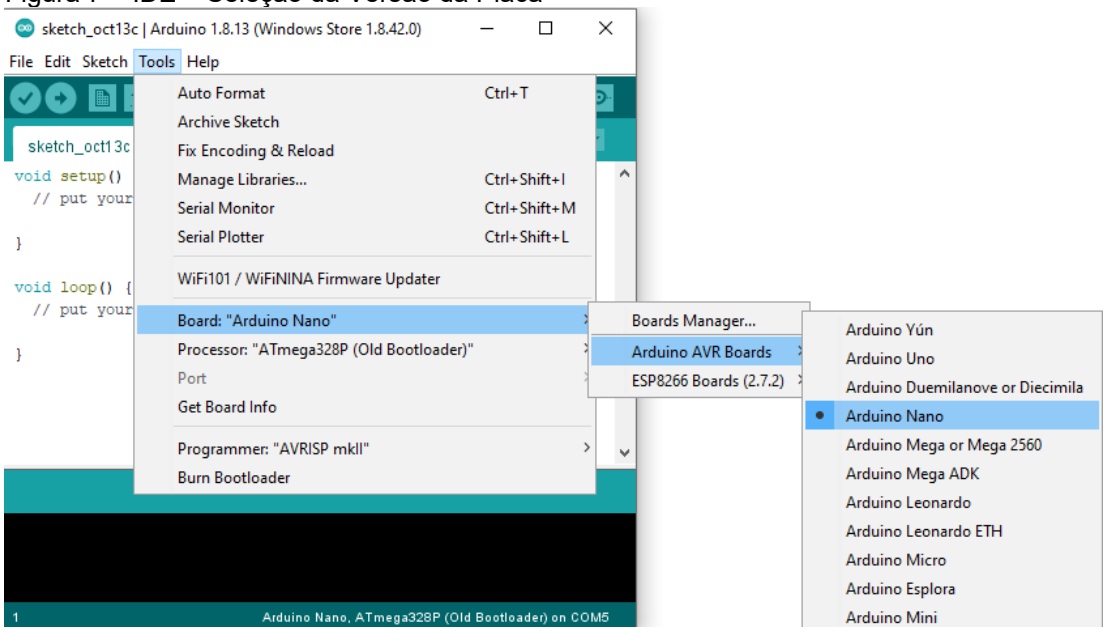
Após o código fonte ser escrito, é necessário que seja selecionado a porta USB que o Arduino se encontra conectado e a versão correta seja selecionada corretamente na aba *Tools*. Posteriormente precisa ser carregado para a placa antes de sua execução, através do botão Upload. As imagens abaixo novamente buscam permitir uma melhor compreensão do leitor. Maiores informações são podem ser encontradas na página oficial do Arduino www.arduino.cc/.

Figura 6 – IDE – Seleção de Portas



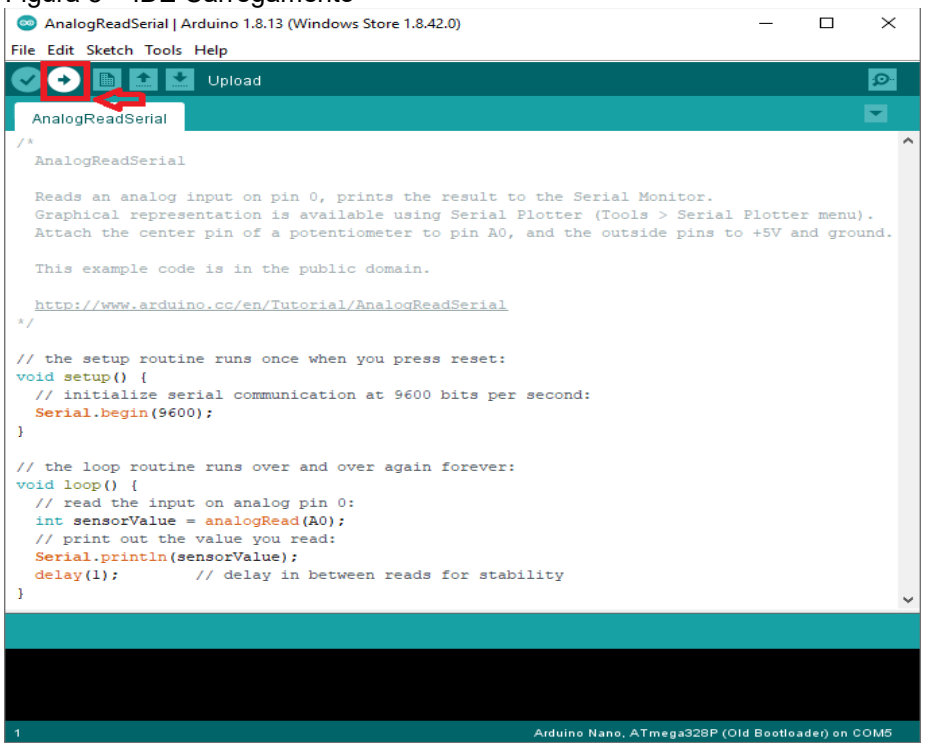
Fonte: Própria Autoria

Figura 7 – IDE – Seleção da Versão da Placa



Fonte: Própria Autoria

Figura 8 – IDE Carregamento



Fonte: Própria autoria

3. APRESENTAÇÃO DO PROBLEMA

Muitas soluções de problemas surgem de necessidades. Existem ocasiões em que problemas são encontrados em processos de fabris, como o observado na proposta deste trabalho, e necessitam de soluções assertivas, sejam elas inovadoras ou não. Diversas questões que dependem exclusivamente de habilidades e aptidões do ser humano ou precisam de acompanhamento, podem ser auxiliados por processos automatizados ou até mesmo substituídos completamente, como geralmente ocorre. Muitos processos de produção podem ser automatizados ao ponto de não haver mais a necessidade de um ser humano precisar ter o contato direto. Por outro lado, pode-se encontrar casos em que processos ou procedimentos sejam mais vantajosos que exista um certo contato com o processo produtivo, por parte do intermédio de uma pessoa, e que este esteja apto a garantir a qualidade de uma montagem específica, sendo enfatizado neste caso o custo. Existem processos de manufatura que a plena automatização pode custar uma alta quantia de capital, ao contratar uma empresa especializada para desenvolver uma solução definitiva, porém se houver um certo nível de criatividade aliada com uma boa dose de competência técnica, pode-se resolver ou atenuar algo indesejado no processo ou problemas de qualidade.

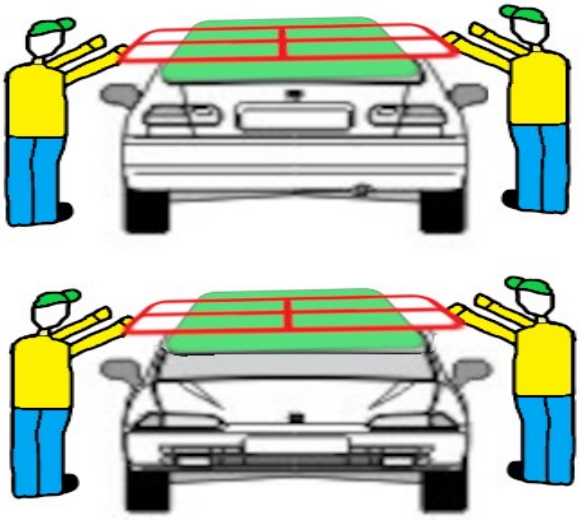
Observando o que fora citado acima, compreendeu-se que através da necessidade de atuar sobre um problema existente no chão de fábrica, mesmo que não o resolvendo por completo, poderia se melhorar de maneira muito significativa a um custo extremamente baixo, atacando um problema conhecido há anos dentro da fábrica automotiva em que o procedimento de montagem é efetuado.

No processo produtivo de montagem de veículos nota-se uma mistura de procedimentos automatizados e manuais. Ocorre o encontro também de várias atividades em que é visto a combinação do ato de montagem manual e automatizado juntos, em um mesmo processo de montagem, não sendo retirado completamente o fator humano da questão e fazendo com que o operador responsável pela montagem precise dispender bastante atenção na manufatura em questão. A montagem de vidros automotivos, que no caso é feito manualmente, demonstra ser bem delicado quando levado em consideração a questão da qualidade, retrabalhos, e sucatas de

peças. Um erro durante a manufatura, seja um movimento brusco ou uma inclinação excessiva, pode gerar incertezas e complicações na qualidade do produto. A precisão do procedimento de montagem e o segmento correto das especificações estabelecidas pelo processo fabril são fundamentais, devido ao fato de ser uma peça bastante visível ao consumidor e que pode gerar defeitos consideráveis como ruídos, infiltrações de água, entre outros. Quando há processos fabris que exigem atenção, geralmente procedimentos padronizados e checagens são habituais, pois auxiliam a evitar que problemas de qualidade avancem sem nenhum tipo de conhecimento por parte do corpo que atua na montagem, checagem dos processos e liderança. Procedimentos como este são fundamentais para se manter o padrão de qualidade sempre elevado.

Buscando compreender a raiz dos problemas que podem ocasionar defeitos de qualidade, embora que por questões de sigilo industrial não possam ser reproduzidas no presente trabalho, refletindo sobre possibilidades de atuação sobre um dos maiores problemas relacionados a montagem dos vidros, foi observado que o maior deles é o de infiltrações de água. Iniciou-se uma profunda reflexão dos mecanismos possíveis para atacar a questão. Fora observado que inclinações durante o processo de montagem dos vidros nas carrocerias poderiam incidir na geração de defeitos de montagem. Defeitos estes que não são muito facilmente identificados pelos operadores que executam o trabalho. Com base na experiência que um indivíduo possa adquirir ao longo de anos trabalhando em um determinado tipo de atividade, pode se observar um *feeling* do trabalho e isso pode maximizar a acurácia de uma atividade, não se pode afirmar que nunca existe a possibilidade de um erro, falha ou defeito ocorrer, seja qual motivo for. No processo de manufatura em questão, foi observado que diversos defeitos durante a montagem ocorreram sem a percepção dos operadores que atuam nele. A figura 9 logo abaixo demonstra a forma de montagem no processo.

Figura 9 – Processo de Montagem de Vidros



Fonte: Própria Autoria

Nas extremidades dos vidros são postas uma cola especializada que atua na fixação e vedação. Se a fixação dos vidros ocorrer de maneira desproporcional, com inclinações excessivas, problemas de infiltrações de água poderão ocorrer. É muito importante que a cola posicionada nas extremidades dos vidros permaneça de maneira uniforme durante a descida dos vidros na carroceria.

Plataformas de prototipagem, como o Arduino, são exemplos ótimos de baixo custo de investimento e possibilidades de buscar soluções para problemas das mais variadas naturezas, seja na indústria ou fora dela. Se levar em consideração que cursos de graduação tecnológicas e do âmbito das engenharias possuem em suas grades curriculares, matérias que aplicam o conhecimento de robótica, automatização, aprendizagem em programação em várias linguagens. É possível obter uma gama de possibilidades a serem aplicadas para a resolução de problemas e até inovações tecnológicas. Dessa maneira buscou-se utilizar uma plataforma de desenvolvimento de projetos que oferecesse vantagens, como citado acima, para desenvolver um protótipo que pudesse atuar sobre o problema proposto pelo trabalho.

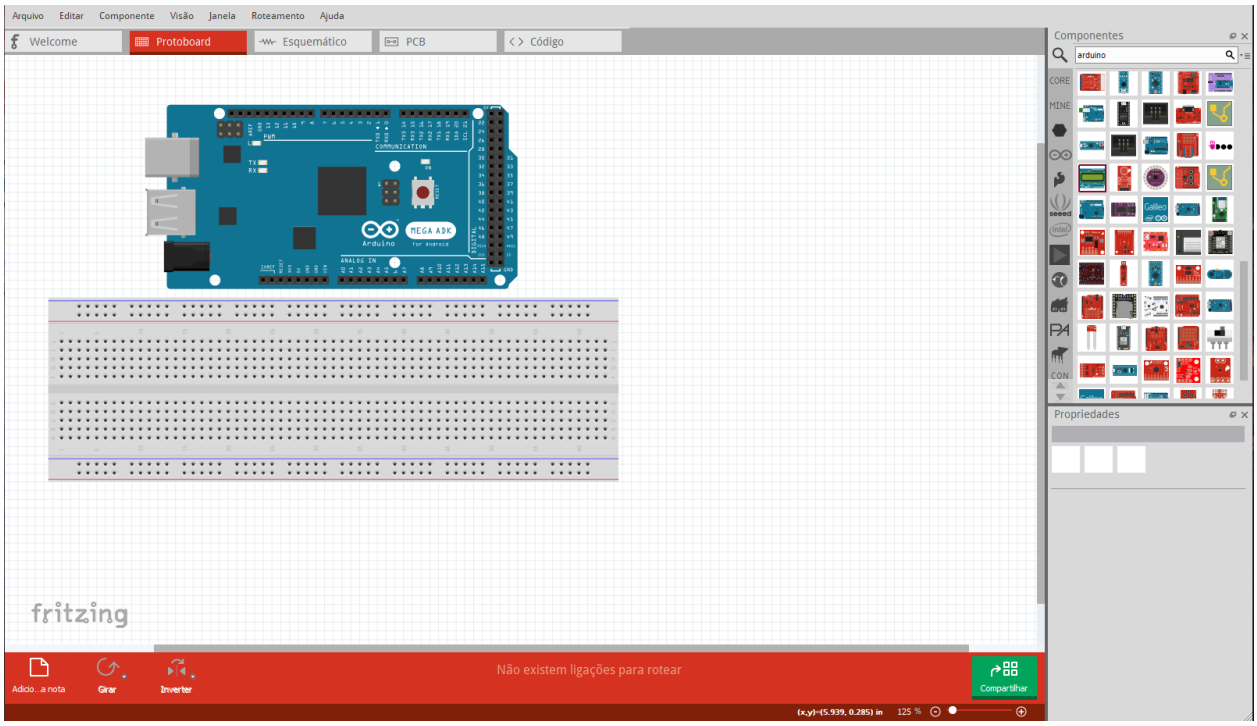
4. MÉTODOS E PROCESSOS

Ao decorrer do desenvolvimento deste projeto, o que inclui a montagem e programação dos protótipos desenvolvidos e seus componentes agregados, nesta seção é possível conferir um pouco a respeito. A utilização dos recursos de software e bibliotecas auxiliou no desenvolvimento deste projeto, provendo uma programação mais facilitada. Foi desenvolvida uma interface Gráfica com Python e a biblioteca Tkinter (disponível nativamente), que oferece recursos para a criação de Interfaces de Usuário Gráficas nativas da linguagem. A interface integrada ao computador tem como objetivo prover uma interação entre o usuário e o dispositivo de uma forma mais agradável, para o acompanhamento dos dados e funcionamento, embora não tenha sido o foco do projeto. A linguagem de programação Python, em sua versão 3.8, possibilita a comunicação com um computador, pelo protocolo de comunicação serial, utilizando a biblioteca PySerial, e o documento de sua instalação se encontra de maneira oficial em *https://pyserial.readthedocs.io*, fazendo com que haja uma boa facilidade de comunicação entre um computador e um dispositivo externo que possibilite a comunicação Serial. Não é intenção deste trabalho apresentar e guiar os leitores quanto a instalação e utilização da linguagem Python, mas é possível encontrar as informações necessárias na página oficial em *python.org/* e não é difícil encontrar informações a respeito da utilização da biblioteca *Tkinter* pela internet, através de pesquisas. É importante salientar que apenas no terceiro protótipo do projeto que a interação entre Homem-Máquina de maneira gráfica está disponível. Foi alvo de uso também as bibliotecas OpenCV, uma biblioteca de processamento de imagens muito utilizada em visão computacional e a Numpy (Numeric Python).

Além da IDE do Arduino, citado anteriormente, foi utilizado para a esquematização do projeto um software de fácil utilização e montagem de circuitos. O Fritzing é uma iniciativa de código aberto, *open-source* que auxilia na documentação de projetos com Arduino, podendo ser encontrado em *https://fritzing.org/*. Na sua utilização, mais precisamente na versão 0.9.4, é possível montar os circuitos usando os recursos disponíveis, como componentes eletrônicos, *protoboards*, placas de Arduino e interligá-los com os cabos *Jumper* que estão

incluídos para a prototipagem com o *software*. A utilização de aplicações que possibilitem esquematizar projetos, favoreceu e muito a montagem posterior.

Figura 10 – Ambiente de Desenvolvimento Fritzing



Fonte: Própria Autoria

Para efetuar a esquematização do circuito dos protótipos deste trabalho, não necessariamente precisa ser utilizado o Fritzing, podendo ser feito em outros softwares do mercado, como o Proteus.

4.1 Apresentação básicas dos Principais Componentes Utilizados

Além dos recursos de *software*, como o ambiente de desenvolvimento do Arduino, o PyCharm para a criação da interface gráfica do usuário e Fritzing para desenho do esquema, foi utilizado vários componentes eletrônicos no projeto. Existe uma infinidade de sensores e componentes compatíveis com o Arduino e a intenção desta parte do trabalho é demonstrar um pouco as características básicas dos que foi usado neste trabalho.

4.1.1 Acelerômetros

Um componente pequeno pode fazer uma enorme diferença. É o caso dos acelerômetros. Estes pequenos dispositivos estão presentes em muitos objetos que são úteis para a humanidade, como celulares, *notebooks*, sistemas de *airbags,* controladores de videogames, entre outras coisas.

“Acelerômetros são sensores fantásticos, que permitem todos os tipos de interações gestuais emocionantes” (EVANS et al,2016, p.221). Estes pequeninos aparatos podem auxiliar no desenvolvimento desde *softwares* que utilizam interações gestuais para controlar um aparelho remotamente e até jogos digitais na indústria de entretenimento.

Para o propósito do presente trabalho, acelerômetros se mostraram componentes úteis para o controle de inclinações de um dispositivo de uso manual, utilizado por montadores do processo produtivo de montagem de vidros nos carros, na fábrica. Vale ressaltar novamente que o processo é puramente manual e é extremamente delicado ao ponto de precisão ser um fator bem relevante. Isso faz com que inclinações possam gerar problemas de qualidade durante o processo de montagem. Essas inclinações são indesejadas e durante fixação dos vidros devem ser evitadas ao máximo. Esse é um grande reforço da necessidade de uma espécie de controle das inclinações, mantendo em um nível mínimo possível as obliquidades.

Inicialmente o acelerômetro utilizado foi o modelo conhecido como ADXL-335, que é um modelo com três saídas analógicas e três eixos e taxa de medida de aproximadamente 3g. Um acelerômetro que ofereceu uma fácil manipulação dos seus recursos, porém utilizava 3 portas analógicas, fazendo com que fosse optado por utilizar posteriormente outro modelo de acelerômetro que oferecesse uma conexão mais facilitada, como a I2C.

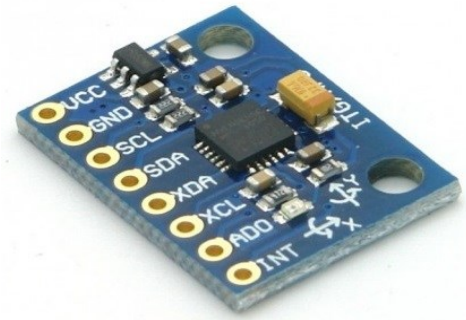
O acelerômetro MPU6050 (Figura 10) é um componente bastante utilizado onde seja necessário o uso de componentes Micro eletromecânicos, que fundem características elétricas e mecânicas e possuem capacidades de obter dados sensoriais e reagir aos estímulos recebidos. Possui 6 eixos, sendo 3 para o acelerômetro e 3 para o giroscópio. Os valores de saída do acelerômetro são referentes aos eixos X, Y e Z.

Um acelerômetro do tipo eletromecânico funciona de forma muito similar a um transdutor piezoelétrico. Sensores deste tipo trabalham com ação resultante de pressão, no caso dos piezoelétricos, sua ação ocorre literalmente quando algo o pressiona. Nos acelerômetros do tipo MEMS (*Micro Electro Mechanical Systems)* uma tensão é gerada em resposta a pressão mecânica exercida, mas no caso o que interessa é quando há uma inclinação no dispositivo.

"Um acelerômetro, como seu nome sugere, mede as mudanças na aceleração" (EVANS et al,2016, p.251). Esses dispositivos medem a variação ou mudança da velocidade em relação ao tempo, o quão rápido um determinado corpo está em relação a outro que possa ter uma aceleração menor. Este poderia ser um assunto bastante extenso, mas não é o propósito deste trabalho.

Este pequeno objeto citado acima, é de fundamental importância para o trabalho e sua forma de atuar para auxílio no controle das inclinações do manipulador de montagem dos vidros nos carros, sendo escolhido para utilizar no segundo e terceiro protótipos por sua facilidade de conexão, a I2C.

Figura 11 – Acelerômetro MPU6050



Fonte: Disponível em: https://www.vidadesilicio.com.br/. Acesso em 13 jun. 2020

4.1.2 Sensor Ultrassônico

O sensor de ultrassom utilizado é o HC SR04 da Elecfreaks.c, um sensor frequentemente empregado para fins didáticos e um dos motivos é por ser de baixo custo. Possui uma precisão relativamente boa, mas não tão exata. A figura 11 ilustra esse sensor.

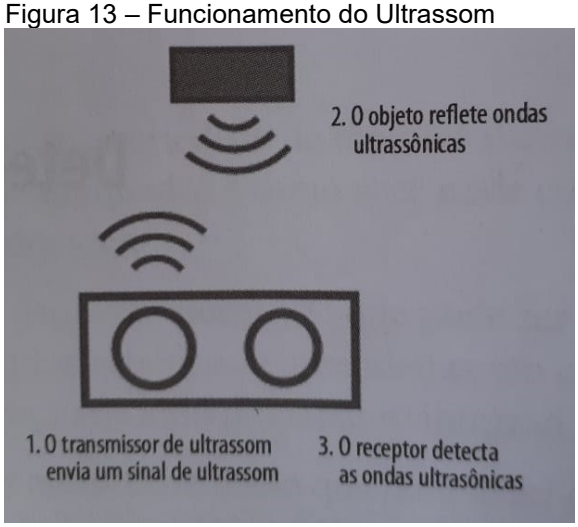
Figura 12 – Sensor HC SR04

Fonte: Datasheet Ultrasonic ranging module HC SR04

Este sensor é frequentemente encontrado em projetos voltados para a robótica e muito utilizado na detecção de objetos, além de ser de fácil utilização e possuir diversas bibliografias que tratam a respeito e diversos materiais e projetos pela internet que o utilizam.

Conforme descrito por Evans, Noble e Hochenbaum (2013, p.157): “Você emite um som, espera para ouvir um eco e, se você possuir o tempo certo, saberá se algo está lá fora e qual a sua distância”. Este pequeno componente se mostra útil para projetos em que é necessário a detecção de objetos que se aproximam e é conhecido como eco localização, no meio ambiente existem diversos animais que se guiam através desse princípio como os morcegos.

Figura 13 – Funcionamento do Ultrassom



Fonte: Livro Arduino em Ação, p.158

Conforme Evans, Noble e Hochenbaum (2013, p.158): “Um sensor consiste em dois componentes separados: um que envia o sinal e outro que recebe o sinal de volta.” Um microcontrolador, ou uma plataforma que contenha um microcontrolador embutido que acaba sendo responsável por calcular o período entre o pulso emitido e o recebimento.

4.1.3 Transceptor de Rádio Frequência RF24L01

O NRF24L01 é um transceptor de Rádio Frequência de 2.4GHz e baixo consomo de energia produzido pela Nordic Semiconductors. Possui amplificador de potência (PA), um amplificador de baixo ruído (LNA) e antena externa de 2 DB (Decibéis). Utiliza a comunicação com os microcontroladores por meio da interface SPI (Serial Peripheral Interface). Existem dois tipos para esse modulo. Um possui antena interna e outro modelo (Figura 13) possui antena adicional externa, intercambiável de 2 dBi (decibéis).

Transceptores de radiofrequência podem se mostrar muito úteis no que diz respeito a conexão de dados e criação de uma rede de envio e recebimento de informações oriundos de sensores. Para Neto, Pereira, Pereira (2015, p.1) “Tais redes são tecnologias baratas e podem ser implantadas em uma área física formando redes entre si e com a internet”.

Figura 14 – Modulo Transceptor RF24L01 e Antena Externa

Fonte: Disponível em: http://arduinoinfo.mywikis.net/wiki/Nrf24L01-2.4GHz-HowTo

4.1.4 Display OLED 128x64 de 0.96 polegadas I2C

Segundo o fabricante, O Display OLED de 0.96 polegadas é uma pequena tela, mas oferece boa qualidade na apresentação de dados. Possui interface de comunicação por dois fios (I2C) ou SPI e resolução de 128 por 64 *pixels*. Pode ser utilizado para exibir informações para um usuário como visualização de temperatura, tensão, pequenas mensagens e dados em geral. Opera em nível lógico de 3,3 a 5 *volts* e é fabricado pela Solomon Systech. Abaixo segue uma imagem do componente.

Figura 15 – Display OLED SSD1306 0.96”

Fonte: Disponível em: https://www.tecnotronics.com.br/display-oled-0-96-grafico-128x64-i2c.html

4.2 Introdução e Características Básicas do Python

> A linguagem foi criada em 1990 por Guido van Rossum, no Instituto Nacional de Pesquisa para Matemática e Ciência da Computação da Holanda (CWI), e tinha originalmente foco em usuários como físicos e engenheiros. O Python foi concebido a partir de outra linguagem da época ,chamada ABC. (BORGES, 2014, p.15).

Python é uma das linguagens de programação de alto nível que mais cresce atualmente, estando em terceiro lugar em um *ranking* de pesquisa mensal do *TIOBE index* – da *TIOBE Programming Community* com 11.28 % de representatividade. O sucesso da linguagem e sua ótima aceitação podem estar ligados a facilidades de sua sintaxe, amplo material disponível para aprendizagem, simplicidade e agilidade, podendo facilitar a aprendizagem nas universidades e desenvolvedores em geral. Sua aplicação é bastante extensa, passando por diversas áreas do conhecimento científico

como Data Science, machine learning, sistemas back-end Web e possibilitam facilidades de integração entre camadas de um projeto, como o objeto deste trabalho, por exemplo.

Borges afirma que (2014, p.14): “Também inclui recursos encontrados em outras linguagens modernas, tais como geradores, introspecção, persistência, metaclasses e unidades de teste.” É uma linguagem interpretada, diferentemente de C/C++ que é uma linguagem compilada. Pode ser visto que “A linguagem é interpretada através de bytecode pela máquina virtual Python, tornando o código portável”. (BORGES, 2014, p.14). Além dessas características, a linguagem é de código aberto pela licença Pública Geral (GPL) no âmbito de caracterização de software livre.

No desenvolvimento deste trabalho, como será abordado e demonstrado mais a frente, Python teve um papel importante, pois permitiu realizar uma boa interação entre usuário e máquina.

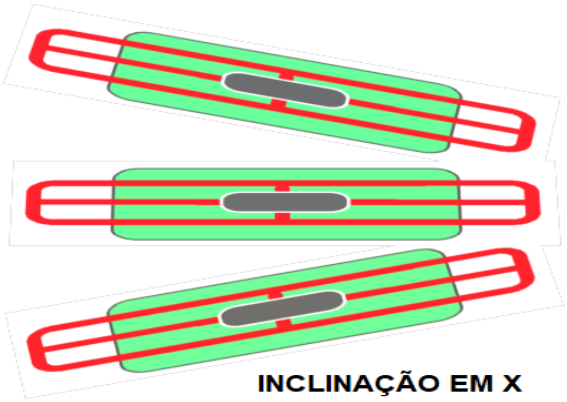
4.3 Desenvolvimento do Protótipo I

A plataforma Arduino é um exemplo amplo de uso, na construção de protótipos, devido em parte pela vasta quantidade de material e ensinamentos de uso encontrados pela Internet. É uma plataforma bastante popular e seu propósito é exatamente esse, por isso utiliza-se muito no âmbito do ensino. Claramente e somente a plataforma não é suficiente dependendo do projeto a ser construído. Muitas vezes é necessário a utilização de diversos componentes agregados a uma *protoboard* para a construção do layout do circuito a ser utilizado em conjunto com a placa do microcontrolador. Pode ser visto na figura 17 o modelo de protoboard, ou placa de prototipagem.

Uma preocupação deste trabalho foi procurar exemplificar por intermédio de figuras e imagens que pudessem ilustrar de maneira satisfatória a montagem do trabalho e a forma de utilização, além de seus componentes e funcionamento. Dessa maneira buscando aprimorar mais a visualização conceitual a figura abaixo demonstra o equipamento de montagem manual de vidros, com uma representação básica do

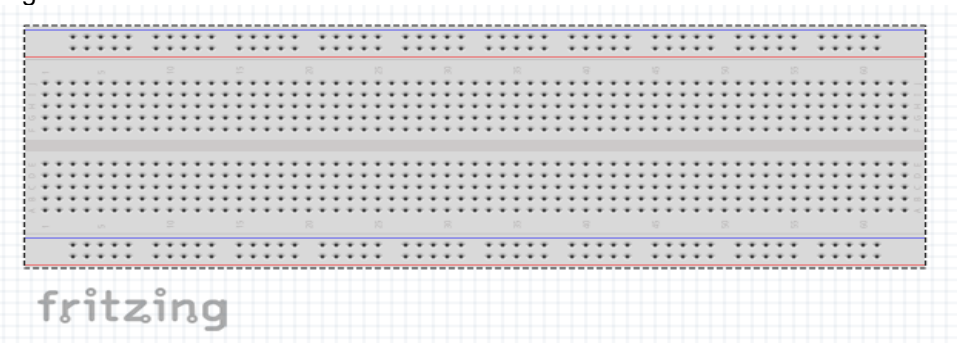
local onde seria mais bem posicionado o modelo de protótipo (em cinza) e a inclinação no eixo X que o protótipo busca atuar.

Figura 16 – Inclinação do Manipulador com o protótipo acoplado



Fonte: Própria Autoria

Figura 17 – Placa Protoboard



Fonte: Própria Autoria

Além da placa para prototipagem, do circuito para a utilização dos componentes eletrônicos necessários, observado na figura 17, os componentes acoplados na protoboard precisam ser ligados através do circuito por cabos conhecidos como *jumper.*

Além dos acelerômetros, existem uma grande quantidade de componentes que trazem muitos benefícios para quem deseja explorar a área de automação e robótica. Neste trabalho são explorados diversos recursos ao longo da trajetória de desenvolvimento, agregando mais funcionalidades.

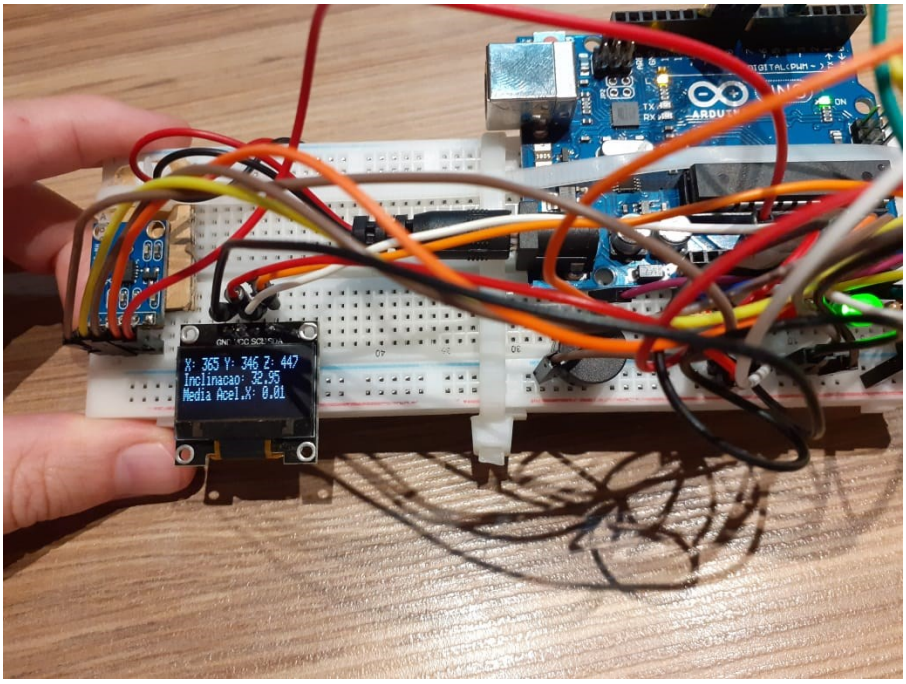
O modelo de protótipo I utiliza como componente fundamental para a obtenção dos dados de inclinação o acelerômetro ADXL335, que é um componente de fácil

utilização no âmbito da programação e de baixo custo, oferecendo excelente custo-benefício para a prototipagem.

O projeto que irá atuar no equipamento utilizado para montagem de vidros, será acoplado ao próprio dispositivo de montagem, funcionando como uma espécie de guia de montagem. O dispositivo tem a responsabilidade de soar um alarme, através do *buzzer* e acender um *LED* vermelho quando o equipamento apresentar inclinações que estejam fora dos limites de tolerância nos eixos X e Y estabelecidos como padrão mínimo de qualidade. O padrão máximo de tolerância representa um papel fundamental nas inclinações acusadas pelo acelerômetro, pois é por esses valores que um usuário do equipamento, o qual o protótipo esteja acoplado poderá manter o equipamento em um nível estável, sem inclinações mais íngremes. Quando o equipamento se encontra nivelado adequadamente, o dispositivo permanece com *LEDs* verdes acesos e não soa nenhum tipo de alarme ou ruídos. No protótipo, foi colocado um pequeno display de *LCD* para um acompanhamento dos valores em tempo real, embora não houvesse sido decidido o local ou *layout* que este ficaria acomodado, o display faz com que os dados sejam melhores apresentados, ao invés de somente poder ser acompanhado os valores pelo monitor Serial da IDE do Arduino. O dispositivo conta com a conexão de um adaptador para uma alimentação através de uma bateria de 9V.

Em resumo, o objetivo principal de qualquer protótipo desenvolvido pelo presente trabalho é acusar e alertar um usuário que esteja manipulando um equipamento, possa ser auxiliado pelo mesmo a fim de controlar as inclinações durante a montagem de vidros automotivos. Na Figura 18 é possível ver o protótipo por meio de uma imagem.

Figura 18 – Protótipo 1



Fonte: Própria autoria

Não é possível identificar facilmente, mas o acelerômetro deste primeiro modelo de projeto se encontra ao lado esquerdo e acima do *display LCD*. O monitor LCD apresenta os valores de leitura do acelerômetro em tempo real, os *buzzers* executam um efeito sonoro quando os limites de tolerância dos parâmetros definidos no dispositivo são ultrapassados.

A leitura dos dados obtidos pelo acelerômetro é feita a partir de três portas analógicas, representando os eixos X, Y e Z, sendo o mais vital para o presente trabalho o eixo X, pois observou-se que durante o processo de montagem dos vidros no veículo amplas inclinações nesse eixo horizontal ocasionavam muitos defeitos de qualidade. O esquema de montagem do protótipo deixa mais claro e fácil a observação do dispositivo (apêndice B).

O código de programação foi desenvolvido todo na própria IDE – Ambiente de Desenvolvimento Integrado do Arduino e algumas bibliotecas de recursos também foram aplicadas. Existem bibliotecas de recursos que não acompanham a instalação padrão da IDE do Arduino, dessa forma, instalação de bibliotecas adicionais deve ser feita via gerenciador de bibliotecas, apresentado na aba ferramentas da IDE. No primeiro protótipo, foram usadas as bibliotecas ADXL335, recurso fornecido pelo fabricante de acelerômetros, Wire e a Adafruit_SSD1306, utilizada para configuração do display de informações. Após a finalização do desenvolvimento deste primeiro

protótipo, foram identificadas algumas dificuldades, pois houve problemas ao realizar testes aplicados na montagem dos vidros, devido dificuldades de manuseio e a fixação dos componentes envolvidos, o que inclui a facilidade dos componentes e cabos se soltarem da placa de protótipo, pois quando não há soldagem a possibilidade de isso ocorrer é alta. Fora observado também que o dispositivo I apresentava oscilações excessivas nas leituras obtidas pelo acelerômetro, o que tornava o dispositivo sujeito a falsos positivos, pois quando se efetuava uma aceleração brusca do protótipo em qualquer direção o dispositivo atuava e esse resultado não é útil para o objetivo de controle das inclinações deste trabalho. Dessa forma, precisava se de um meio para atuar sobre o problema de oscilação das leituras dos eixos do acelerômetro e atuação indesejada do dispositivo.

4.4 Desenvolvimento do Protótipo II

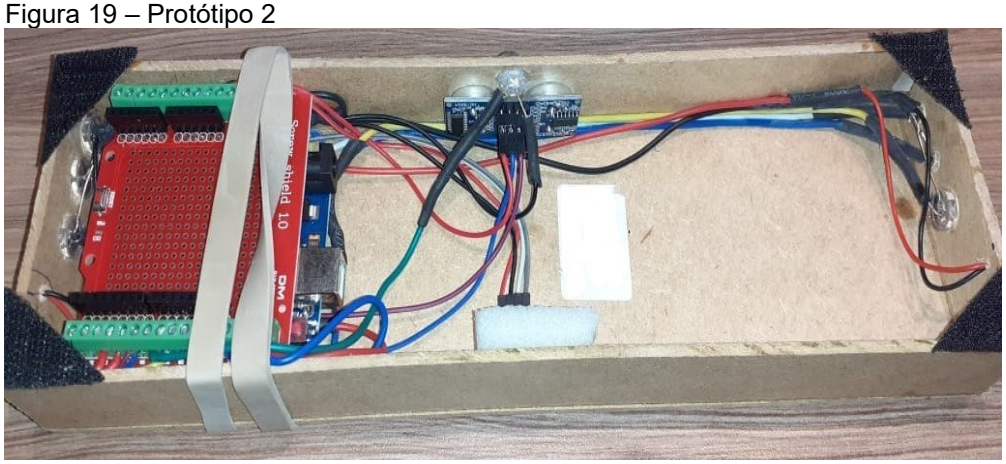
A proposta para o protótipo seguinte é oferecer um layout mais organizado, ter uma montagem mais robusta e trazer alguns recursos adicionais que pudessem ajudar a tornar o projeto mais assertivo quanto a montagem. Uma outra questão importante, verificado após testes de funcionamento do protótipo anterior era as oscilações das leituras. Para atuar este problema foi realizado uma espécie de filtro *passa-baixo*, com a finalidade de minimizar o máximo possível a instabilidade que havia nos valores obtidos do acelerômetro, mesmo quando o dispositivo se encontrava em repouso. Para implementar o filtro foi realizado um cálculo a partir de um determinado número de amostras alocadas em um vetor, obtidas das leituras brutas em tempo real do sensor e feito uma média da soma destes valores.

O filtro executado, embora simples, estabilizou melhor as leituras obtidas, porém para auxiliar essa estabilização e restringir a acusação por parte do protótipo a movimentos indesejados durante a movimentação livre do dispositivo, buscou se efetuar uma condição interna que se as inclinações no eixo X fossem maiores a um determinado valor, o programa aguardaria um determinado tempo e verificaria novamente se as inclinações realmente eram maiores que tal valor, pois o que importa nessa situação é sempre o estado final da condição, atuando em uma espécie de

Debounce (o código fonte pode ser encontrado no apêndice A). Existem meios de alterar sensibilidade dos sensores e estes são ajustáveis via código, porém os erros de medida verificados não puderam ser atenuados apenas alterando a sensibilidade.

O segundo protótipo foi montado em uma caixa de proteção de madeira, com o circuito devidamente fixado sem uso de *protoboard,* porém a placa principal do microcontrolador foi acomodada em uma placa do tipo *borne.* Foi acrescentado um sensor ultrassônico de distância como recurso adicional para identificar o vidro posicionado no dispositivo. O sensor ultrassônico atua de forma identificar, pelo tempo de retorno do eco, a distância do objeto posicionado acima dele e dessa forma é possível fazer com que o dispositivo somente exerça suas funcionalidades se houver um vidro posicionado na proximidade do sensor. Com base nesse princípio definiu se uma distância aproximada que o sensor identifica o vidro. Os fios foram soldados devidamente, com exceção da parte que fora conectado e parafusado através do borne presente do *shield* acoplado na placa micro controladora. O monitor *LCD* foi retirado desta segunda versão, pois fora repensado que não seriam necessários os operadores do processo de montagem, ao manipularem o dispositivo, focarem a atenção nos valores apresentados pelo *display.* O acelerômetro desta versão de projeto é o modelo MPU6050 da TDK Invensense. Os *LEDs* usados e as buzinas (*buzzers*) foram posicionados nas partes laterais do estojo de proteção, totalizando 6 *LEDs*. Os *LEDs* vermelho sinalizam quando há uma inclinação fora do padrão estabelecido como limite de tolerância seguro para a qualidade de montagem do processo. O *LED* azul amarelo indica a presença de um vidro posicionado na proximidade do sensor e o verde sinaliza que a inclinação no eixo X de atuação se encontra dentro do limite de tolerância do dispositivo. É importante salientar que existe um *shield* acoplado a placa principal, que permite a construção de um circuito adicional, para realizações futuras no projeto. Na Figura 17 é possível conferir a evolução da montagem do protótipo. Na parte traseira da caixa protetora do circuito possui um velcro para fixação do dispositivo no manipulador de montagem de vidros, passando a ser um equipamento somente. As bibliotecas de recursos utilizadas foram as bibliotecas padrões do Arduino e uma adicional, a MPU6050, necessária para a programação do acelerômetro MPU6050. Com base nisso, os operadores podem executar a montagem com maior confiança em relação a inclinações fora do padrão estabelecido.

Figura 19 – Protótipo 2



Fonte: Própria Autoria

Este protótipo se mostrou mais apto a realização de testes de funcionamento, sendo mais manuseável e fácil para posicioná-lo no equipamento de montagem de vidros, mais detalhes do desenvolvimento podem ser encontrados no apêndice A e B.

4.5 Desenvolvimento do Protótipo III

A montagem do segundo protótipo fez com que o projeto tivesse a capacidade de ser manuseado mais facilmente em relação ao antecessor, obteve um formato mais característico de um protótipo propriamente dito e mostrou uma organização de *layout* bem mais robusta, porém deixava em aberto algumas possibilidades de melhoria na comunicação do protótipo e um computador, pois a única maneira de obter os dados em uma interface era pela intervenção do monitor serial do Arduino. Uma melhoria em relação a comunicação, poderia ser oferecida com a comunicação do dispositivo sem fios, seja por um modulo de conexão *Wi-fi* conectado à internet ou através de comunicação por Rádio Frequência e uma interface gráfica para observação dos dados de maneira mais agradável, pela interface desenvolvida para rodar no computador.

Uma melhoria na comunicação, embora não fosse crucial para o principal objetivo do dispositivo, daria uma interatividade maior a ele, pois os dados poderiam

ser acompanhados por alguém que quisesse supervisionar o processo de montagem dos vidros automotivos a uma certa distância.

A opção de desenvolver um terceiro protótipo traz também o objetivo de agregar novas funcionalidades ao projeto para fins de estudo, que além de aprimorar e obter melhorias continuas, ajuda a desenvolver novas habilidades que possam ser uteis.

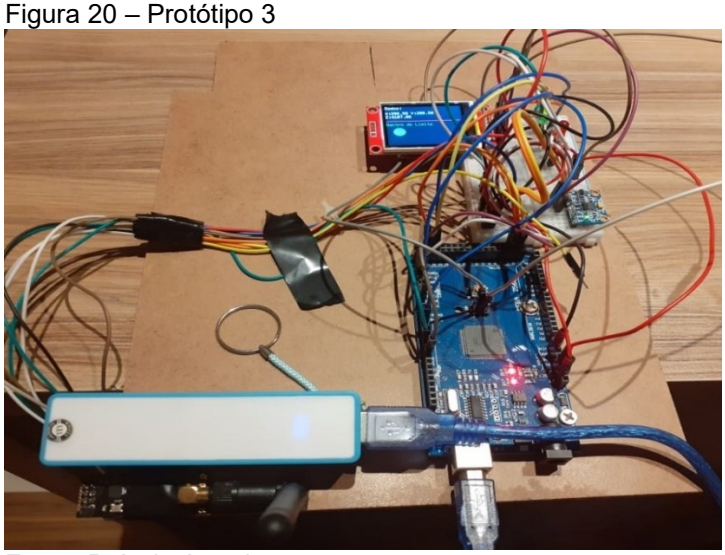
Foi escolhido para desenvolver essa versão do protótipo, o Arduino Mega 2560 rev.3, que permite que um projeto possa evoluir bastante posteriormente devido ao seu maior número de portas de conexão. O esquema de montagem pode ser encontrado no apêndice B e o código de desenvolvimento está localizado no apêndice A.

Esta versão inclui a maior parte dos tipos de componentes utilizados anteriormente, removendo o sensor ultrassônico HC-SR04, os *buzzers* e o *Led* amarelo, usado anteriormente para sinalização de presença de vidros. O protótipo foi montado sobre uma placa de madeira de MDF com uma protoboard fixada para facilitar as conexões do acelerômetro MPU-6050, os *LEDs* indicativos vermelho e verde, o *display* LCD TFT 1.8 polegadas. Também foi colocado ao canto da placa de madeira um modulo transceptor de radiofrequência NRF24L01 com antena embutida para o envio das informações para uma outra placa micro controladora, conectado ao computador.

Já em relação ao display, ele tem a finalidade de mostrar os valores do eixo X, Y e Z em tempo real. Possui também uma condição em que se o valor do eixo X ultrapassar o limite de tolerância é impresso na tela uma mensagem sinalizando que que o dispositivo se encontra fora do estabelecido, incluindo um círculo no display que fica vermelho, caso contrário, o mesmo permanece verde e uma mensagem de que o dispositivo se encontra dentro do limite estabelecido aparece.

A opção de alimentação de energia escolhida para esta versão do projeto ficou definido em utilizar um carregador portátil de celular, muito prático e leve, pois existem versões mais compactas para serem posicionados no *layout* do projeto. A figura 20 mostra uma imagem do protótipo relacionado.

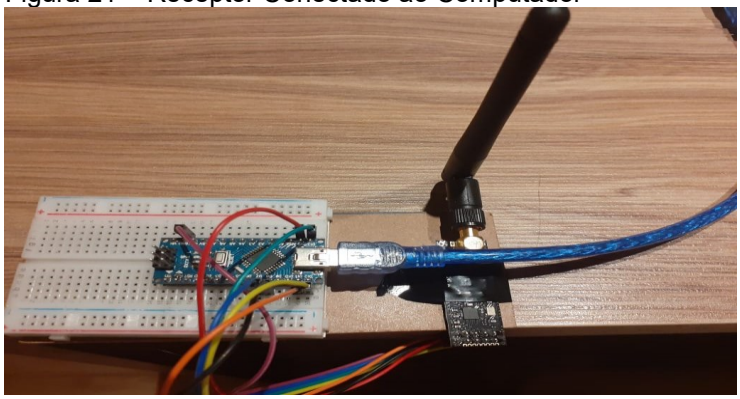
Figura 20 – Protótipo 3



Fonte: Própria Autoria

No desenvolvimento deste modelo , devido o envio de dados ser por radiofrequência, foi necessário desenvolver de maneira adicional um protótipo para atuar como receptor dos dados enviados a partir do protótipo principal. O receptor recebe os dados enviados pelo protótipo emissor, através do modulo transceptor de radiofrequência NRF24L01, devidamente estabelecido para apenas receber dados, com antena embutida e envia para o computador pela porta de comunicação USB para o computador. Em suma, são utilizados dois transceptores RF para efetuar a comunicação entre as duas placas do Arduino basicamente. No item B.5 do apêndice B pode se obter informações adicionais referente a conexão dos pinos do modulo de radiofrequência NRF24L01 nas versões Uno, Nano e Mega 2560 do Arduino.

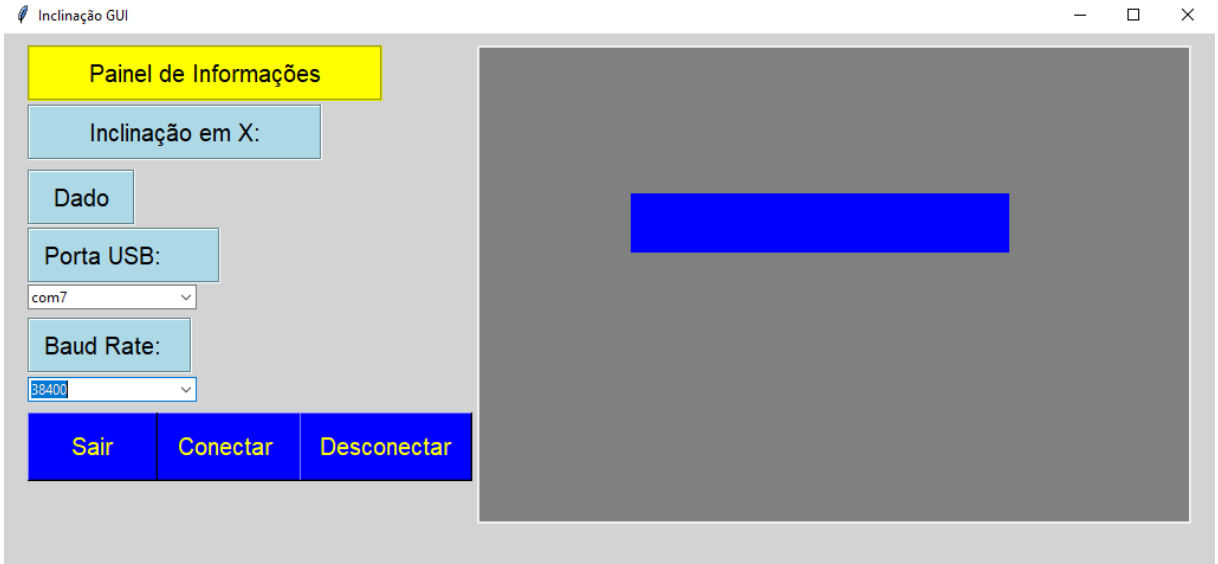
Figura 21 – Receptor Conectado ao Computador



Fonte: Própria Autoria

Foi desenvolvido uma pequena interface gráfica em Python, com a finalidade de demonstrar os valores obtidos do eixo X do acelerômetro e mostrar a um usuário que esteja interagindo com um computador. Na interface é apresentada uma imagem de um equipamento de montagem de vidros e é possível vê-lo se movimentando conforme o protótipo é inclinado no eixo X do acelerômetro. O objetivo desta representação foi apresentar visualmente um modelo para que exista uma facilidade em acompanhar os movimentos realizados enquanto o protótipo é manipulado. Para o desenvolvimento dessa aplicação foi utilizada a biblioteca de recursos OpenCV, numpy e a Tkinter, essa última para a criação dos *Widgets* da tela. A interface oferece botões com funcionalidades de efetuar a conexão do dispositivo com o computador, sair, seleção de portas USB para selecionar a porta correspondente que o Arduino se encontra conectado e uma caixa de seleção para selecionar a taxa de *Baud Rate*. Também foi incluído um *Canvas* de maneira experimental para fins de estudo de movimentação de objetos a partir do acelerômetro para trabalhos futuro. A barra azul, que pode ser visualizada na figura 22, para este protótipo, exibe a movimentação para cima ou para baixo, conforme exista inclinações no eixo X. Abaixo é possível conferir a modelo da interface atual. Posteriormente a barra desenhada no canvas poderá representar uma possível liberdade de movimentos em todos os eixos do acelerômetro.

Figura 22 – Interface de Usuário



Fonte: Própria Autoria

Figura 23 – Equipamento de Montagem de vidros

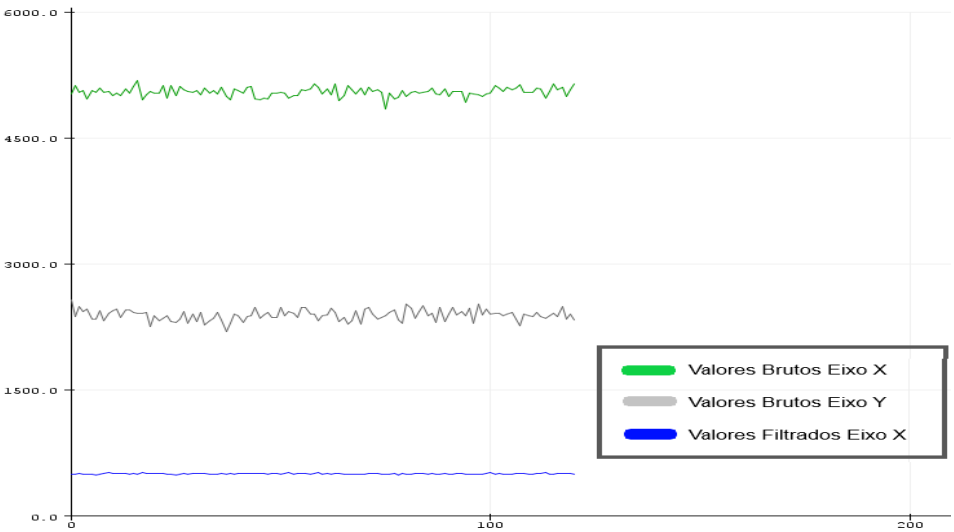
Fonte: Própria Autoria

Para efetuar a comunicação do Arduino com a aplicação foi utilizada a biblioteca PySerial, disponível para Python, que pode ser facilmente instalado no Windows.

5. RESULTADOS OBTIDOS

O processo de desenvolvimento e montagem das possíveis soluções do problema apresentado na abordagem da problemática ocorreu em três etapas. Durante o desenvolvimento dos protótipos procurou se concentrar inicialmente na solução para criar um sistema que auxiliasse os montadores de vidros veiculares a controlar a montagem dos vidros na carroceria, a partir de exibição de alertas, por intermédio de via sonora e visual, ajudando a manter a montagem estável, sem inclinações excessivas e buscando redução de defeitos. Foram feitas algumas simulações de uso do protótipo 1 e 2 e embora a função para controlar as inclinações tenha funcionado de acordo, foi percebido que oscilações e ruídos nas leituras do acelerômetro atrapalhavam no bom funcionamento do protótipo, fazendo com que fosse acusado falsos positivos durante o processo de montagem. No segundo protótipo, com a filtragem pelo cálculo da média de amostras obtidas do sensor, verificou-se uma melhora considerável em relação ao anterior. Com a verificação de *debounce* melhorou mais, pois o dispositivo aguardava um determinado tempo mínimo para confirmar se o dispositivo se encontrava mesmo na posição observada no eixo X. Através da ferramenta de plotagem de dados do Arduino, pode ser observado os níveis de oscilações dos eixos do acelerômetro e o resultado obtida com a filtragem.

Figura 24 – Plotagem de Dados do Acelerômetro - Arduino



Fonte: Própria Autoria

Embora os valores filtrados apresentaram uma melhoria na oscilação, essa melhora foi mais bem verificada quando o dispositivo se encontrava em repouso. Durante o trajeto de movimentação livre no processo de montagem dos vidros na carroceria, essa relevância da filtragem não se mostrou muito eficaz, sendo melhorado mais pelo *debounce,* neste caso. Quando os montadores efetuavam movimentações bruscas, o equipamento identificava as acelerações exercidas, captadas pelo acelerômetro, acusando como inclinações, embora em muitos casos não fossem.

O terceiro protótipo não foi submetido a testes, pois até o momento do desenvolvimento deste trabalho não foi possível adaptar no momento uma estrutura que possibilitasse acoplar no dispositivo de montagem de vidros, e por enquanto o objetivo principal do terceiro protótipo é ampliar o desenvolvimento de recursos adicionais. Este protótipo permitiu de maneira satisfatória na visualização de dados pelo display e o envio e recebimento de dados por radiofrequência, pelo uso de um computador com interface gráfica (*GUI)* e a visualização dos dados nela.

6. CONSIDERAÇÕES FINAIS

Com base na observação dos resultados obtidos e o valor agregado no projeto, foi percebido que existem diversas melhorias que podem ser implementadas que vão desde a montagem de uma estrutura mais definitiva para abrigar os protótipos, melhorias no algoritmo de filtragem dos dados, sendo uma boa opção aplicar a filtragem pelo método de Kalman, de modo a possibilitar uma melhor estabilização do sensor inercial. A implementação do controle das inclinações no eixo Y poderia ajudar no processo de montagem dos vidros, auxiliando na atuação dos montadores do processo produtivo referente ao trabalho em obter mais controle do processo em si.

Os dois primeiros protótipos desenvolvidos buscaram operar de modo a desempenhar um papel importante no que diz respeito ao foco da causa das adversidades levantadas no capítulo que foi abordado o problema. A partir do essencial do trabalho realizado, foi possível perceber que havia mais direções praticáveis para serem aplicados no trabalho.

Para o processo de aprendizagem, o fator de busca por conhecimento adicional e novos horizontes de aplicação para o projeto, fizeram com que fosse testado várias formas de criar novas funcionalidades para o trabalho, explorando outras linguagens de programação não listadas neste trabalho e componentes que possuem características positivas para a prática de prototipagem, inclusive que possam ser aplicáveis em outros projetos no futuro e que agregam valor no trabalho.

Obteve se um ganho de interatividade no projeto a partir do terceiro protótipo, pois a integração com uma interface de usuário e transmissão de dados sem fio tornam o projeto mais dinâmico e rico.

6.1 Trabalhos Futuros

O projeto em sua reta final de desenvolvimento deixou uma questão interessante em aberto para exploração futura. No *Canvas* da interface de usuário, mais precisamente na barra azul posicionada no centro do Canvas permite uma

implementação e aprimoramento da movimentação estabelecida pelo acelerômetro. Até o momento ela se encontra se movimentando apenas em duas direções, porém pode ser interessante empreender uma funcionalidade de um movimento mais livre e amplo para o acelerômetro e em tempo real, para objetivar e ampliar o estudo com acelerômetros.

Uma outra proposta válida para este trabalho pode ser enxergada através da expansão das capacidades do dispositivo desenvolvido para não somente se conectar e comunicar com um computador ou outros aparatos por radiofrequência, mas sim pela internet, dessa maneira se torando um dispositivo *IoT,* abrindo mais o leque ainda e possivelmente armazenando dados em um banco de dados compartilhado.

REFERÊNCIAS BIBLIOGRÁFICAS

ALEXANDER, C.K; SADIKU, N. O. **Fundamentos de Circuitos Elétricos.** 5.ed. Porto Alegre: AMGH Editora Ltda, 2013.

ALVES, R. M. et al. **Uso do Hardware Livre Arduino em Ambientes de Ensino-aprendizagem.** In: Anais da Jornada de Atualização em Informática na Educação – JAIE, 2012. P.167.

ANTUNES, C.A; GALHARDI, V. B; HERNASKI, C. A. **As leis de Newton e a estrutura Espaço-temporal da Mecânica Clássica:** Revista Brasileira de Ensino de Física, vol. 40, nº 3, e3311, 2018.

ARDUINO. **Arduino**, c2020. Página Inicial. Disponível em <www.arduino.cc>. Acesso em: 12 de jun. de 2020.

ARDUINO. **Arduino,** c2020. Arduino Products. Disponível em <www.arduino.cc/en/Main/Products>. Acesso em: 12 de jun. de 2020.

BORGES, L. **Python Para Desenvolvedores.** 1.ed. São Paulo: Novatec, 2014.

ELECFREAKS. **Electro Schematics** Ultrassonic Ranging Module HC-SR04.

Datasheet. Disponível em <www.electroschematics.com/hc-sr04-datasheet>. Acesso em: 15 de jun. de 2020.

EVANS, M.; NOBLE, J.; HOCHENBAUM, J. **Arduino em Ação.** 3.ed. São Paulo: Novatec, 2016.

FONSECA, E. G. P.; VEGA, S. A. **Tutorial Sobre Introdução a Projetos Utilizando o Kit de Desenvolvimento Arduino.** In: Congresso Brasileiro de Educação em Engenharia, XXXIX, 2011, Blumenau. Niterói: Universidade Federal Fluminense - Escola de Engenharia, 2011. p.3.

Instituto de Computação – Universidade Federal do Mato Grosso, Cuiabá, 2015. Tiobe Company – **TIOBE PROGRAMMING INDEX OF OCTOBER 2020.** Disponível em**:** < https://www.tiobe.com/tiobe-index> . Acesso em 02 out. 2020.

MCROBERTS, M. **Arduino Básico.** 2.ed. São Paulo: Novatec, 2018.

MONK, S. **30 Projetos com Arduino.** 2.ed. Porto Alegre: Bookman, 2014.

NETO, Evaldo J. K; PEREIRA, Mauricio F. L; PEREIRA, Roberto B. O. **Utilizando o transceptor NRF24L01 em redes de sensores sem fio operando sobre TCP/IP.**

OLIVEIRA, S. **Internet das Coisas com ESP8266, Arduino e Raspberry Pi.** 1ed. São Paulo: Novatec, 2017.

SANTOS. B.P et al. **Internet das Coisas: da Teoria à Prática.** Departamento de Ciência da Computação – Universidade Federal de Minas Gerais (UFMG), Belo Horizonte, 2016.

APÊNDICE A – CÓDIGO FONTE DO DESENVOLVIMENTO DO PROJETO

Neste apêndice, com intuito de manter a clareza e organização das informações relevantes a programação , o código fonte de todos os dispositivos pode ser encontrados aqui.

As listagens de código apresentam as versões dos três protótipos desenvolvidos ao longo da trajetória do trabalho, as versões do Arduino que foram utilizadas e a interface desenvolvida em Python. Os códigos fonte seguem todos comentados de maneira a esclarecer o máximo possível da lógica criada

A.1 Código Fonte do Protótipo I

```
/*
    * Author/Coder : Bruno A. Patrocinio
    * Version : 2.0.2 (Com acelerômetro para captura de dados e Display LCD)
    */

   #include <Wire.h>
   #include <Adafruit_GFX.h>
   #include <Adafruit_SSD1306.h>

   Adafruit_SSD1306 display = Adafruit_SSD1306(); //

   //definindo as constantes:
   #define pinAcelX A0
   #define pinAcelY A1
   #define pinAcelZ A2
   const int pinoLedRed1 = 10;
   const int pinoLedRed2 = 9;
   const int pinoLedGreen = 8;
   const int pinoSom = 5;
   int eixoX, eixoY, eixoZ;
   int i;
    void setup() {
     Serial.begin(9600);

     //config iniciais do lcd Oled
```

```
  Wire.begin(); //inicializando biblioteca wire.h
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.clearDisplay();

  //config iniciais dos leds e buzzer sound
  pinMode(pinoLedRed1, OUTPUT); //setando pino led verm. como saida.
  pinMode(pinoLedRed2, OUTPUT); //setando pino led verm. como saida.
  pinMode(pinoLedGreen, OUTPUT); //setando pino led verde como saida.
  pinMode(pinoSom, OUTPUT); //setando pino do som como saída.
  //os pinos de led verm iniciam em LOW e o verde em HIGH
  digitalWrite(pinoLedRed1, LOW);
  digitalWrite(pinoLedRed2, LOW);
  digitalWrite(pinoLedGreen, HIGH);
}

void loop() {
  //inicia as leituras
  eixoX = analogRead(pinAcelX);
  eixoY = analogRead(pinAcelY);
  eixoZ = analogRead(pinAcelZ);

  //printa os valores no monitor serial(tela)
  Serial.print("X: ");
  Serial.print(eixoX);

  Serial.print(" Y: ");
  Serial.print(eixoY);

  Serial.print(" Z: ");
  Serial.print(eixoZ);
  Serial.print("\n");

  //Aqui será realizado as condições ou seja, chama se a função.
  verificaPos();

  //Aqui printa os valores no lcd oled, chama se a função
  printaData();
}

int verificaPos(){
  if(eixoZ <= 310)
  {
  if(eixoX >= 351){
    Serial.println("não vira!!!!\n");
    digitalWrite(pinoLedRed1, HIGH);
    tone(pinoSom, 2000, 800); //speaker apita em frequencia diferente
    digitalWrite(pinoLedGreen, LOW);
  }
```

```
  else if(eixoX <= 331){
    Serial.println("não vira!!!!\n");
    digitalWrite(pinoLedRed2, HIGH);
    tone(pinoSom, 2000, 800); //speaker apita em frequencia diferente
    digitalWrite(pinoLedGreen, LOW);
  }
  else{
    digitalWrite(pinoLedRed1, LOW); // deixa o pino dos leds em LOW
    digitalWrite(pinoLedRed2, LOW);
    digitalWrite(pinoLedGreen, HIGH);
    noTone(pinoSom);
  }
  }else{
    digitalWrite(pinoLedRed1, LOW); // deixa o pino dos leds em LOW
    digitalWrite(pinoLedRed2, LOW);
    digitalWrite(pinoLedGreen, HIGH);
    noTone(pinoSom);
  }
}//fim verificaPos

void printaData(){
  int vetor[3];
  char frase[] = "Disp. virado p/ colagem...";
  vetor[0] = eixoX;
  vetor[1] = eixoY;
  vetor[2] = eixoZ;
  display.setCursor(1,2);
  display.print("X: ");
  display.print(vetor[0]);
  display.print(" Y: ");
  display.print(vetor[1]);
  display.print(" Z: ");
  display.print(vetor[2]);
  display.print("\n");
  if(eixoZ <= 310){
    display.print(frase);
    display.display();
  } else{
    //nothing

  }
 display.display();
 display.clearDisplay();
}//fim printaData
```

A.2 Código Fonte do Protótipo II

```
/*
 * Author/Coder : Bruno A. Patrocinio
 * Version : 3.0.5 (Acelerometro MPU6050) 11/05/2019
 */
#include <I2Cdev.h>
#include<MPU6050.h>
#include <Wire.h>

//constantes sensor ultrassom
#define trigger 5  //pino de trigger na porta 5 por constante emitindo o pulso
#define echo 4    //pino echo na porta 4 atraves de uma constante ouvindo o pulso

//constantes dos pinos led e buzzer
const int pinoLedRed1 = 3;
const int pinoLedRed2 = 2;
const int pinoLedGreen = 6;
const int pinoLedYellow = 7;
const int pinoLedBlue = 12;
const int pinoSom = 13;

double mediaX = 0;
double mediaGiro = 0;
double distCent = 0;
double distRefinada = 0;
double eixoZ;
long timer = 0;

//variaveis sensor ultrassom
float distancia = 0; // essa variavel receberá a distancia em metros
float centimetros = 0; //variavel de tempo para calcular qtd de centimetros
float milimetros = 0;
float tempo = 0;  //variavel de medida de retorno do pulso
bool traseiroCivic = false; //identifica vidro especifico
bool traseiroCity = false;  //identifica vidro especifico
bool frontalCivic = false;  //identifica vidro especifico
bool frontalCity = false; //identifica vidro especifico
```

```
bool frontalFit = false;  //identifica vidro especifico
bool ehTraseiro = false;   //var. bool. que generaliza o vidro traseiro

//Protótipos de Função
void printSerial();
void verificaVidro();
void verificaPosX();

MPU6050 mpu6050; // objeto acelerometro

unsigned int ax, ay, az;
unsigned int gx, gy, gz;

void setup() {
  #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
      Wire.begin();
   #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
      Fastwire::setup(400, true);
   #endif

   Serial.begin(38400);
   Wire.write(0b00000000);

   // config iniciais pinos ultrassom
   pinMode(trigger, OUTPUT);
   digitalWrite(trigger, LOW);
   pinMode(echo, INPUT);

   //config iniciais dos leds e buzzer sound
   pinMode(pinoLedRed1, OUTPUT); //setando pino led verm. como saida.
   pinMode(pinoLedRed2, OUTPUT); //setando pino led verm. como saida.
   pinMode(pinoLedGreen, OUTPUT); //setando pino led verde como saida.
   pinMode(pinoLedYellow, OUTPUT);//setando pino led amarelo como saida.
   pinMode(pinoLedBlue, OUTPUT);//setando pino led azul como saida.
   pinMode(pinoSom, OUTPUT); //setando pino do som como saída.

   //os pinos de led verm e amarelo iniciam em LOW e o verde em HIGH
   digitalWrite(pinoLedRed1, LOW);
   digitalWrite(pinoLedRed2, LOW);
   digitalWrite(pinoLedYellow, LOW);
   digitalWrite(pinoLedBlue, LOW);
   digitalWrite(pinoLedGreen, HIGH);

   //iniciando disp.
   Serial.println("Iniciando...");
   mpu6050.initialize();

   //Executa o teste do dispositivo
   Serial.println("Testando conexão do dispositivo...");
```

```
    Serial.println(mpu6050.testConnection() ? "MPU6050 conectado com sucesso!" :
"MPU6050 conexão falhou.");

}

void loop() {
  // leituras brutas
  mpu6050.getMotion6(&ay, &ax, &az, &gx, &gy, &gz);

  //remap do eixo Z
  eixoZ = map(az, 0, 1023, 0, 255);

  //chama funções
  printSerial();
  verificaVidro();
  verificaPosX();

}//fim de loop

void printSerial(){
  if(millis() - timer > 1000){
  Serial.println("Dados:");
  Serial.print("X raw: ");Serial.print(ax);
  Serial.print(" Y raw: ");Serial.print(ay);
  Serial.print(" Z raw: ");Serial.print(az);
  Serial.print(" Media Giro X: ");Serial.print(mediaGiro);
  Serial.print(" Media X: ");Serial.print(mediaX);
  Serial.print(" Centimetros: ");Serial.print(centimetros);
  Serial.print(" Dist Metros: ");Serial.print(distCent);
  Serial.print(" Milimetros: ");Serial.print(milimetros);
  Serial.print("\n");

  timer = millis();
  }
}//fim de printSerial

void verificaPosX(){
  //faz uma media de 100 amostras para estabilização dos dados do eixo X
  double amostras[10];
  double soma = 0;
  int i, o;
  for(int i = 0; i <= 9; i++){
    amostras[i] = ax;
  }
  soma = soma += amostras[i];
  mediaX = soma / 10;

  //faz uma media de 100 amostras para estabilizar o giroscopio
  double amostraGiro[10];
  double somaGiro = 0;
```

```
  for(int o=0; o<=9; o++){
    amostraGiro[i] = gx;
  }
  somaGiro = somaGiro += amostraGiro[i];
  mediaGiro = somaGiro / 10;

  if(eixoZ >= 2300 && ehTraseiro == true)
  {
  if(mediaX <= 215){
    delay(500);
    if(mediaX <= 215){
    Serial.print("Não Vira\n");
    digitalWrite(pinoLedRed1, HIGH);
    tone(pinoSom, 2000, 800); //speaker apita
    digitalWrite(pinoLedGreen, LOW);
   }
 }
  else if(mediaX >= 722){
    delay(500);
    if(mediaX >= 722){
    digitalWrite(pinoLedRed2, HIGH);
    Serial.print("Não Vira\n");
    tone(pinoSom, 2000, 800); //speaker apita
    digitalWrite(pinoLedGreen, LOW);
  }
 }
  else{
    digitalWrite(pinoLedRed1, LOW); // deixa o pino dos leds em LOW
    digitalWrite(pinoLedRed2, LOW);
    digitalWrite(pinoLedGreen, HIGH);
    noTone(pinoSom);
  }
  }else{
    digitalWrite(pinoLedRed1, LOW); // deixa o pino dos leds em LOW
    digitalWrite(pinoLedRed2, LOW);
    digitalWrite(pinoLedGreen, HIGH);
    noTone(pinoSom);
  }
}//fim de verificaPosX

void verificaVidro(){
  int i;
  double soma = 0;
  double amostrasD[100];
  int contador = 0;
  digitalWrite(trigger,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigger,LOW);
  tempo = pulseIn(echo,HIGH);
  tempo = tempo/1000000;
```

```
distancia = (tempo * 340)/2; //calcula distancia em metros
centimetros = distancia * 100; //calcula distancia em centimetros
milimetros = centimetros / 0.10;//calcula em milimetros
 for(int i = 0; i<=99; i++){//pega 100 amostras
  amostrasD[i] = distancia;
}
  soma = soma += amostrasD[i]; //faz media
  distRefinada = soma / 100;
  distCent = distRefinada *100;

if(centimetros >= 2 && centimetros <= 10){
  Serial.println("Possui um vidro traseiro do city");
  digitalWrite(pinoLedYellow,HIGH);
  traseiroCivic = true;
  ehTraseiro = true;
}
else if(centimetros >=11 && centimetros <= 14){
  Serial.println("Possui um vidro traseiro do civic");
  digitalWrite(pinoLedYellow, HIGH);
  traseiroCity = true;
  ehTraseiro = true;
}
else if(centimetros >=15 && centimetros <=18){
  Serial.println("Possui um vidro frontal do Civic");
  digitalWrite(pinoLedBlue, HIGH);
  ehTraseiro = true; //aqui está assim só para acionar a verificação da posiçãol
  frontalCivic = true;
}
else if(centimetros >=19 && centimetros <=22){
  Serial.println("Possui um vidro frontal do Fit");
  digitalWrite(pinoLedBlue, HIGH);
  ehTraseiro = true; //aqui está assim só para acionar a verificação da posição
  frontalFit = true;
}
else if(centimetros >=23 && centimetros <=26){
  Serial.println("Possui um vidro frontal do City");
  digitalWrite(pinoLedBlue, HIGH);
  ehTraseiro = true; //aqui está assim só para acionar a verificação da posição
  frontalCity = true;

}
else{
  digitalWrite(pinoLedYellow,LOW);
  digitalWrite(pinoLedBlue, LOW);
  traseiroCivic = false;
  traseiroCity = false;
  frontalCivic = false;
  frontalCity = false;
  frontalFit = false;
  ehTraseiro = false;
```

```
  }

}//fim verificaVidro
```

A.3 Código Fonte do Protótipo III

```
/*
 * Author/Coder : Bruno A. Patrocinio
 * Version : 3.5.0 (Acelerometro MPU6050 e LCD Display de OLED 1.8") 19/07/2020
 */
#include <Wire.h>
#include <SPI.h>
#include <MPU6050.h>
#include <nRF24L01.h>
#include <RF24.h>
#include <Ucglib.h>//biblioteca de uso do display

#define cs 53
#define dc 3
#define rst 14
#define WHITE 0xFFFF

//Instância do objeto da classe RF24
RF24 radio(7, 8);

//objeto acelerômetro
MPU6050 mpu;

//Objeto da classe UCGlib do display
Ucglib_ST7735_18x128x160_HWSPI ucg(3, 53, 14);

// I2C address of the MPU-6050
const int endereco = 0x68;

// endereço para envio das informações Transm e recep. devem ter o mesmo endereço
const byte address[6] = "00002";

//variáveis de leitura brutas estas não permitem valores negativos
unsigned int acelX, acelY, acelZ, temperatura;
unsigned int giroX, giroY, giroZ;

//variáveis de contador
long timer = 0;
```

```
//variaveis globais gerais
float data;
double eixoX, eixoY, eixoZ;
double mediaX = 0;
double mediaY = 0;

//constantes dos pinos digitais LEDs
const int pinoLedRed1 = 24;
const int pinoLedRed2 = 25;
const int pinoLedGreen = 26;

//prototipos de função
void printaSerial();
void filtro();
void printaDisplay();
void verificaPos();

void setup() {
  //RF
  Serial.begin(38400);
  radio.begin(); //inicia a comunicação sem fio RF

  radio.openWritingPipe(address); //define o endereço p/ envio de dados ao recept.
  radio.setPALevel(RF24_PA_HIGH); //define o nível do amplificador de potência
  radio.stopListening(); //define o modulo como transmis. apenas

  //Acelerômetro
  #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
      Wire.begin();
   #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
      Fastwire::setup(400, true);
   #endif

  Wire.write(0b00000000);
  Wire.beginTransmission(endereco);//inicia a transmissão de Wire
  Wire.write(0x6B); // PWR_MGMT_1 register
  Wire.write(0);    // estabelece em 0, acorda o MPU6050
  Wire.endTransmission(true);

  mpu.initialize();
  //verifica a conexão e informa a versão
  Serial.println("CODE INCLINACAO MPU6050_2.0.5_COM FUNCAO MAP E
LCD_TFT updated 02_05");
  Serial.println("Testando conexão do dispositivo...");

  Serial.println(mpu.testConnection() ? "MPU6050 conectado com sucesso!" :
"MPU6050 conexão falhou.");
  mpu.setXAccelOffset(0);

  //inicializa parametros iniciais do display TFT 128x160 1.8"
```

```
  ucg.begin(UCG_FONT_MODE_SOLID); //inicia o objeto criado e coloca fundo
  ucg.clearScreen(); //limpa a tela
  ucg.setRotate270(); //rotaciona a tela 270° p/ posicionar os dados em flat
  ucg.setFont(ucg_font_profont12_mr); //seta a fonte e tamanho (15)
  ucg.setPrintPos(0, 10);// seta a pos. inicial do cursor para o print
  ucg.print("Dados:");//printa o titulo das informações

  //inicializa os pinos digitais dos leds como saída
  pinMode(pinoLedRed1, OUTPUT);
  pinMode(pinoLedRed2, OUTPUT);
  pinMode(pinoLedGreen, OUTPUT);
  //inicia em nivel baixo os vermelhos e o verde em nivel lógico alto
  digitalWrite(pinoLedRed1, LOW);
  digitalWrite(pinoLedRed2, LOW);
  digitalWrite(pinoLedGreen, HIGH);
}

void loop() {

  mpu.getMotion6(&acelX, &acelY, &acelZ, &giroX, &giroY, &giroZ);

  // remap das leituras função map()
  eixoX = map(acelX, 0, 1023, 0, 255);
  eixoY = map(acelY, 0, 1023, 0, 255);
  eixoZ = map(acelZ, 0, 1023, 0, 255);

  // chamada das funções modularizadas
  printaSerial();
  filtro();
  printaDisplay();
  verificaPos();

}// fim de loop

void printaSerial(){ //printa a cada 0.5 segundo usando a função millis()
  if(millis() - timer > 500){
   //Não é necessário imprimir todas as informações no Monitor Serial

  //Serial.print("AcX = "); Serial.print(acelX);
  //Serial.print(" | AcY = "); Serial.print(acelY);
  //Serial.print(" | AcZ = "); Serial.print(acelZ);
  //Serial.print(" | Tmp = "); Serial.print(temperatura / 340.00 + 36.53); // converte a
temperatura em graus C
  //Serial.print(" | GyX = "); Serial.print(giroX);
  //Serial.print(" | GyY = "); Serial.print(giroY);
  //Serial.print(" | GyZ = "); Serial.print(giroZ);
  Serial.print(" | Media X = ");Serial.print(mediaX);
  //Serial.print(" | Eixo X = ");Serial.println(eixoX);

  //envio de info. por radio frequencia
```

```
  data = mediaX;
  radio.write(&data, sizeof(data));
  //radio.write(&text, sizeof(text));

  timer = millis();
  }
}//fim de printaSerial

void filtro(){
  //faz uma media para filtro passa baixa
  double amostras[64];
  double amostras2[64];
  double soma = 0;
  double soma2 = 0;
  int i,o;
  for(int i=0; i<=64; i++){ //preenche os arrays
    amostras[i] = eixoX;
    amostras2[i] = eixoY;
  }
  soma = soma += amostras[i];
  mediaX = soma / 64;

  soma2 = soma2 += amostras2[i];
  mediaY = soma2 / 64;

}//fim de filtro

//Função para imprimir os dados no Display
void printaDisplay(){
  ucg.setColor(0, 255, 255, 255);
  ucg.setColor(1, 0, 0, 0);
  ucg.setPrintPos(0, 28);
  ucg.print("X:"); ucg.print(mediaX);
  ucg.print(" Y:"); ucg.print(mediaY);
  ucg.setPrintPos(0, 40);
  ucg.print("Z:"); ucg.print(eixoZ);
  ucg.setPrintPos(0, 52);
  ucg.setColor(255, 0, 0);
  ucg.drawHLine(0, 50, 160);

  if(mediaX > 215.80 || mediaX < 206.20){
    ucg.setPrintPos(0, 65);
    ucg.print("Fora do Limite  ");
    ucg.drawDisc(25,85, 12, UCG_DRAW_ALL);
  }
  else{
    ucg.setColor(0,255,0);
    ucg.setPrintPos(0, 65);
    ucg.print("Dentro do Limite");
    ucg.drawDisc(25,85, 12, UCG_DRAW_ALL);
```

```
  }

}//fim de printaDisplay

//função para verificar a posição do Acelerometro no eixo X e acusar nos LEDs
// e/ou pode ser implementado n funcionalidades futuras
void verificaPos(){
  if(mediaX > 215.80){
    digitalWrite(pinoLedRed1, HIGH);
    digitalWrite(pinoLedGreen, LOW);
    Serial.println("Fora do Limite estabelecido!");
  }
  else if(mediaX < 206.20){
    digitalWrite(pinoLedRed2, HIGH);
    digitalWrite(pinoLedGreen, LOW);
    Serial.println("Fora do Limite estabelecido!");
  }
  else{
    digitalWrite(pinoLedRed1, LOW);
    digitalWrite(pinoLedRed2, LOW);
    digitalWrite(pinoLedGreen, HIGH);
  }
}// fim de verificaPos
```

A.4 Código Fonte do Protótipo Receptor RF

```
/*
    * Author/Coder : Bruno A. Patrocinio
    * Version : 1.0.0 Receptor de Radiofrequência com Arduino nano
    */
   #include <SPI.h>
   #include <nRF24L01.h>
   #include <RF24.h>

   RF24 radio(7, 8); //Cria o objeto radio, passando os pinos 7 e 8 (CE, CSN)

   const byte address[6] = "00002"; //CRIA UM ENDEREÇO PARA ENVIO DOS
   //DADOS (O TRANSMISSOR E O RECEPTOR DEVEM SER CONFIGURADOS
COM O MESMO ENDEREÇO)
   long timer = 0;

   void setup() {
     Serial.begin(38400);
     radio.begin(); //Inicia a comunicação sem fio RF
     radio.openReadingPipe(0, address); //Define o endereço para recebimento
```

```
  radio.setPALevel(RF24_PA_HIGH); //Coloca o modulo RF em alta potência
  radio.startListening(); //Define o modulo para apenas recebimento
}

void loop() {
  if (radio.available()) {
    float data;
    char text[32] = ""; //Var. que armazena os dados recebidos do emissor

    radio.read(&data, sizeof(data)); // Lê os dados

    if(millis() - timer > 500){
      Serial.println(data); // Imprime os dados no Monitor Serial

      timer = millis();
    }

  }

}
```

A.5 Código Fonte da Interface em Python

```
from tkinter import *
from tkinter import messagebox
from tkinter import ttk
import serial
import cv2 as cv
import numpy as np

#Lê a imagem do arquivo
src = cv.imread('d:/disp.png')

srcTri = np.array( [[0, 0], [src.shape[1] - 1, 0], [0, src.shape[0] - 1]] ).astype(np.float32)
dstTri = np.array( [[0, src.shape[1]*0.33], [src.shape[1]*0.85, src.shape[0]*0.25], [src.shape[1]*0.15,
src.shape[0]*0.7]] ).astype(np.float32)
warp_mat = cv.getAffineTransform(srcTri, dstTri)
warp_dst = cv.warpAffine(src, warp_mat, (src.shape[1], src.shape[0]))
#Rotaciona a imagem após empena-la levemente
center = (warp_dst.shape[1]//2, warp_dst.shape[0]//2)
angle = -17
scale = 1
rot_mat = cv.getRotationMatrix2D( center, angle, scale )
warp_rotate_dst = cv.warpAffine(warp_dst, rot_mat, (warp_dst.shape[1], warp_dst.shape[0]))

#Mostra a imagem do objeto na tela
cv.imshow('Warp + Rotate', warp_rotate_dst)

# Config. da Janela da interface
```

```
janela = Tk()
janela.title("Inclinação GUI")
janela['bg'] = ('lightgray')
janela.geometry('1024x450')

# Coordenadas
x = 130
y = 150
x2 = 450
y2 = 150

# Cria o Canvas
tela = Canvas(janela, height=400, width=600, bg="gray")
tela.place(x=400, y=10)

# Variaveis de Conexão do arduino
global numPorta
global baudRate

# Funções
def sair():
    #arduino.close()
    janela.destroy()

def desconectar():
    try:
        arduino.close()
        arduino.flush()
    except:
        messagebox.showinfo("Mensagem", "Nada Conectado.")

def reading():
    mensagem = arduino.read_until()
    # dado = int.from_bytes(mensagem, byteorder=sys.byteorder, signed=True)
    stringona = mensagem.decode('utf-8')

    # label1['text'] = stringona
    print(label1.cget('text'))
    label1.config(text='Dado: ' + str(stringona))
    janela.after(20, reading)
    # label1.after(200, reading)
    # tela.update_idletasks()
    # print(stringona)
    valor = [stringona]

    for n in valor:
        if n > "208.90":
            direita()
        elif n < "207.15":
            esquerda()
        elif n > "215.80":
            print("Risco de IA")
        elif n < "206.20":
            print("Risco de IA")
        else:
            neutro()

def esquerda():
    y = -10
```

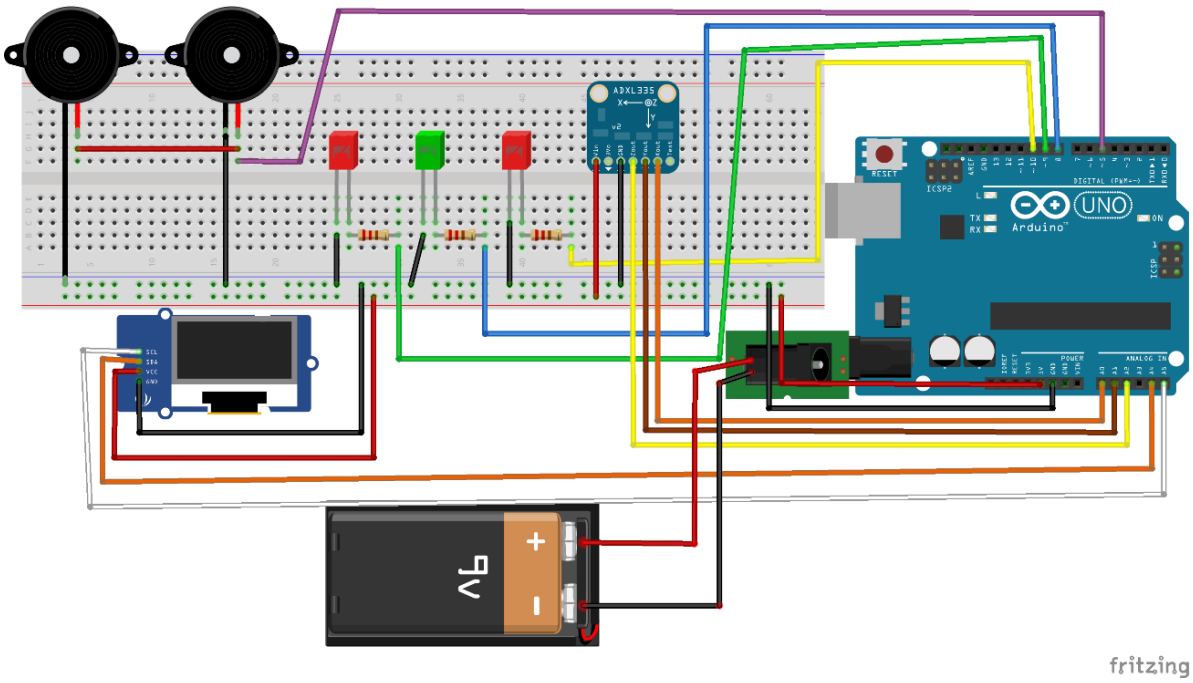
```
    x = 0
    x2 = 0
    y2 = 0
    tela.move(linha, x, y)
    # tela.coords(linha, 130, y - 10, 450, 150)
    rot_mat = cv.getRotationMatrix2D(center, angle-10, scale)
    warp_rotate_dst = cv.warpAffine(warp_dst, rot_mat, (warp_dst.shape[1], warp_dst.shape[0]))
    cv.imshow('Warp + Rotate', warp_rotate_dst)

def direita():
    y = +10
    x = 0
    x2 = 0
    y2 = 0
    tela.move(linha, x, y)
    # tela.coords(linha,130,y+10,450,150)
    rot_mat = cv.getRotationMatrix2D(center, angle+10, scale)
    warp_rotate_dst = cv.warpAffine(warp_dst, rot_mat, (warp_dst.shape[1], warp_dst.shape[0]))
    cv.imshow('Warp + Rotate', warp_rotate_dst)

def neutro():
    rot_mat = cv.getRotationMatrix2D(center, angle, scale)
    warp_rotate_dst = cv.warpAffine(warp_dst, rot_mat, (warp_dst.shape[1], warp_dst.shape[0]))
    cv.imshow('Warp + Rotate', warp_rotate_dst)

def conectar():
    try:
        comboSelecao.current()
        numPorta = comboSelecao.get()
        baudRate = comboSelecao2.get()
        global arduino
        arduino = serial.Serial(numPorta, baudRate, 8)
        reading()
    except:
        messagebox.showinfo("Erro", "Nenhuma porta Selecionada ou não encontrada.")
        print("Nenhuma porta selecionada ou não encontrada.")

# Componentes da interface

# Titulo
labelTitulo = Label(janela, text='Painel de Informações', bg='yellow', padx='50', pady='10', cursor='dot',
                    relief='groove', font=12)
labelTitulo.place(x=20, y=10)

# Representação do Dispositivo de Montagem
linha = tela.create_line(x, y, x2, y2, width=50, fill="blue")

# Sub titulo1
labelTitulo2 = Label(janela, text='Inclinação em X:', bg='lightblue', padx='50', pady='10', cursor='dot',
                     relief='groove', font=12)
labelTitulo2.place(x=20, y=60)

# Label de vis. Inclinação eixo X
label1 = Label(janela, text='Dado', bg='lightblue', padx='20', pady='10', cursor='dot', relief='groove',
font=8)
label1.place(x=20, y=115)

#Comobox Seleção de portas.
label2 = Label(janela, text='Porta USB:      ', bg='lightblue', padx='12', pady='10', cursor='dot',
```

```
relief='groove', font=8)
label2.place(x=20, y=164)

comboSelecao = ttk.Combobox(janela, values=["com0", "com1", "com2", "com3", "com4", "com5", "com6",
"com7",
                            "com8", "com9", "com10", "com11"])
comboSelecao.place(x=20, y=212)

#Label do Combobox Seleção do Baud Rate
label3 = Label(janela, text='Baud Rate:  ', bg='lightblue', padx='12', pady='10', cursor='dot', relief='groove',
font=8)
label3.place(x=20, y=240)

#Combobox Seleção de Baud Rate
comboSelecao2 = ttk.Combobox(janela, values=["9600", "19200", "38400", "57600", "74880", "115200"])
comboSelecao2.place(x=20, y=290)

#Botão conectar
botaoSelecao = Button(janela, text='Conectar', bg='blue', fg='yellow', padx='20', pady='10', font=10,
command=conectar)
botaoSelecao.place(x=120, y=320)

#Botão desconectar
botaoDesconect = Button(janela, text='Desconectar', bg='blue', fg='yellow', padx='10', pady='10', font=10,
command=desconectar)
botaoDesconect.place(x=250, y=320)

# botão sair
botao1 = Button(janela, text='Sair', bg='blue', fg='yellow', padx='30', pady='10', font=10, command=sair)
botao1.place(x=20, y=320)

# janela.after(200, reading)
# janela.update_idletasks()

janela.mainloop()
```

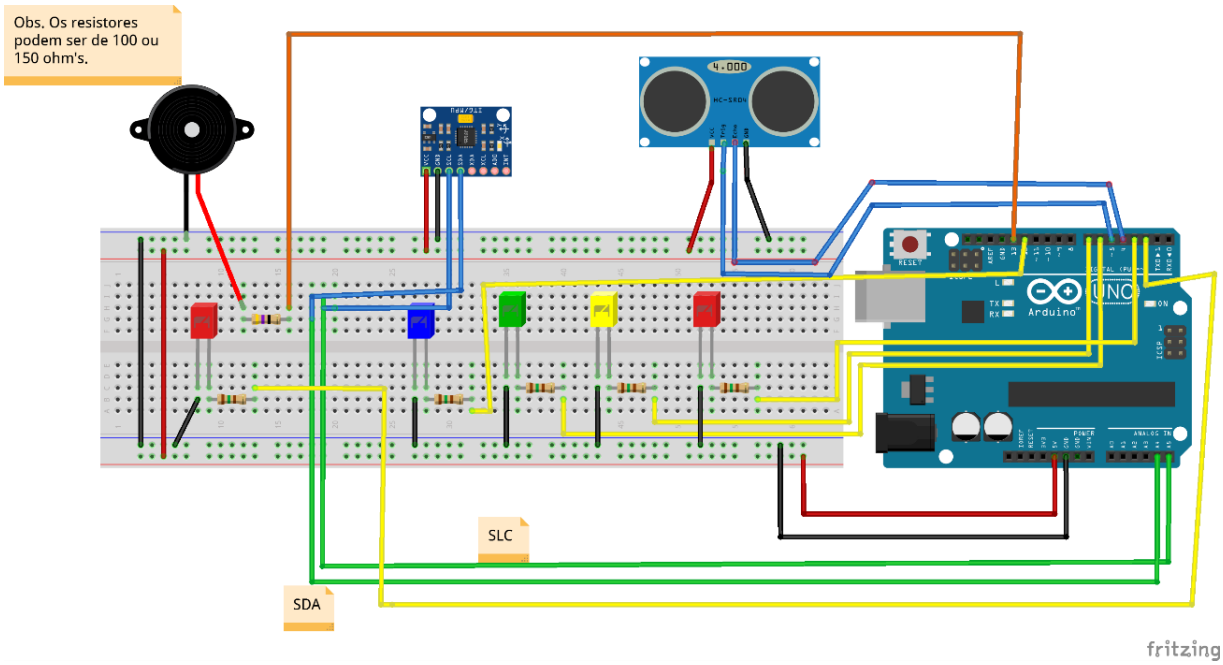
APÊNDICE B – ESQUEMAS DE MONTAGENS DOS PROTÓTIPOS

Nessa parte do trabalho é apresentado os modelos esquemáticos de montagem dos protótipos, com o *layout* das ligações feitas com os cabos jumper e os componentes utilizados.

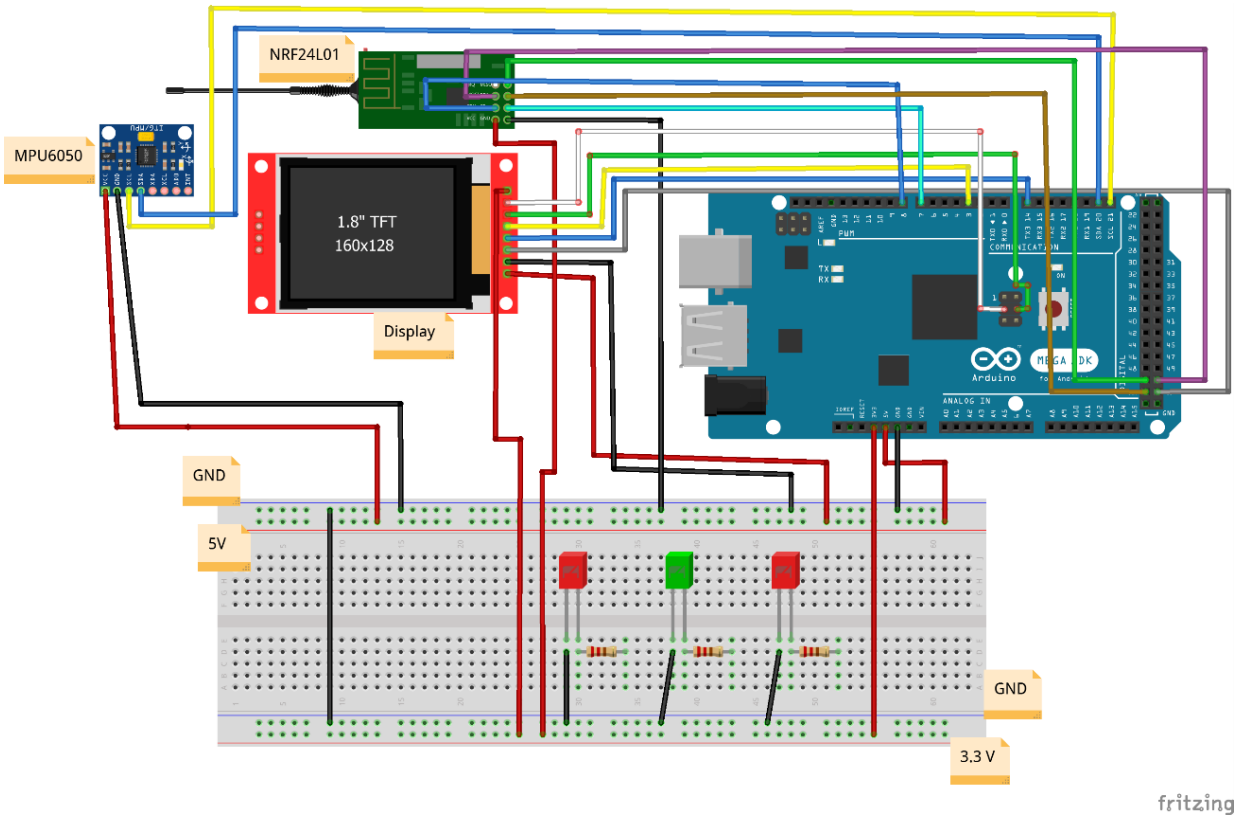
B.1 Esquema do Protótipo I



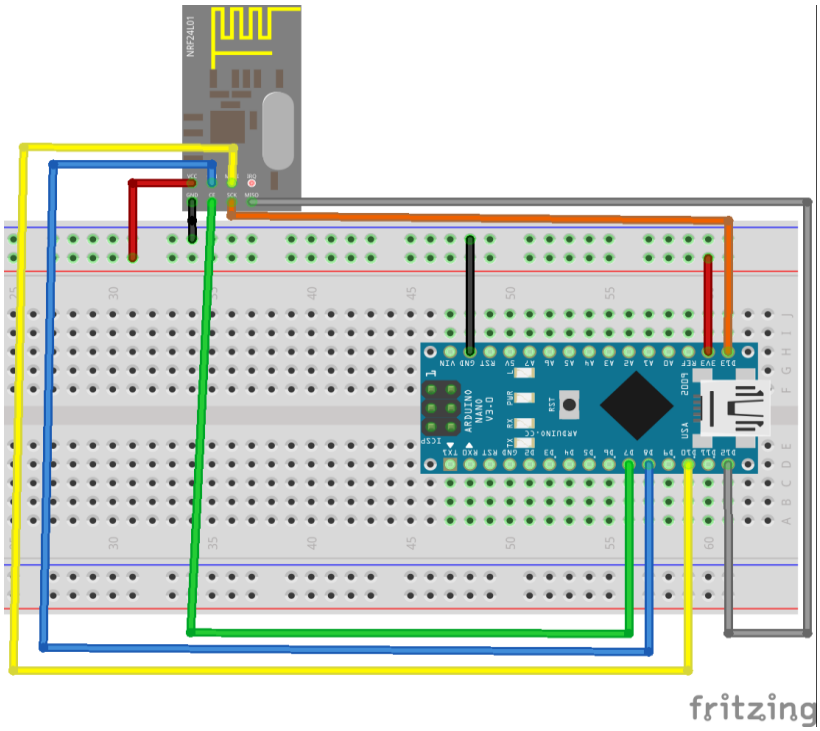
B.2 Esquema do Protótipo II



B.3 Esquema do Protótipo III



B.4 Esquema do Protótipo do Receptor de Radiofrequência



B.5 Tabela de Conexão dos Pinos do Módulo NRF24L01 no Arduino

NRF24L01	Arduino UNO, Nano	Arduino Mega
1: GND	pin GND	pin GND
2: VCC	pin 3V3	pin 3.3V
3: CE	pin 9	pin 9
4: CSN	pin 10	pin 10
5: SCK	pin 13	pin 52
6: MOSI	pin 11	pin 51
7: MISO	pin 12	pin 50