UNIVERSIDADE DO ESTADO DO AMAZONAS - UEA

ESCOLA SUPERIOR DE TECNOLOGIA

ENGENHARIA DE COMPUTAÇÃO

ABDA MYRRIA DE ALBUQUERQUE

UMA PROPOSTA DE AUTOMAÇÃO DE TESTES ANDROID MOBILE PARA A ESTRUTURA DE TESTES DA GOOGLE

Manaus

2022

ABDA MYRRIA DE ALBUQUERQUE

UMA PROPOSTA DE AUTOMAÇÃO DE TESTES ANDROID MOBILE PARA A ESTRUTURA DE TESTES DA GOOGLE

Trabalho de Conclusão de Curso apresentado à banca avaliadora do Curso de Engenharia de Computação, da Escola Superior de Tecnologia, da Universidade do Estado do Amazonas, como pré-requisito para obtenção do título de Engenheiro de Computação.

Orientador: Prof. MSc. Ricardo Rios Monteiro do Carmo

Coorientador: Prof. Eng. Esp. Ricardo de Albuquerque Pinto

Manaus

2022

Universidade do Estado do Amazonas - UEA

Escola Superior de Tecnologia - EST

Reitor:

André Luiz Nunes Zogahib

Vice-Reitor:

Katia do Nascimento Couceiro

Diretor da Escola Superior de Tecnologia:

Ingrid Sammyne Gadelha

Coordenador do Curso de Engenharia de Computação:

Áurea Hileia da Silva Melo

Coordenador da Disciplina Projeto Final:

Áurea Hileia da Silva Melo

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).
Sistema Integrado de Bibliotecas da Universidade do Estado do Amazonas.

A345up Albuquerque, Abda Myrria de
Uma Proposta de Automação de Testes Android Mobile para a Estrutura de Testes da Google / Abda Myrria de Albuquerque. Manaus : [s.n], 2022.
46 f.: color.; 31 cm.

TCC - Engenharia de Computação - Universidade do Estado do Amazonas, Manaus, 2022.
Inclui bibliografia
Orientador: Ricardo Rios Monteiro do Carmo, MSc.
Coorientador: Ricardo de Albuquerque Pinto, Eng. Esp.

1. Automação. 2. Teste de Software . 3. Arduino. 4. Braço Mecânico. 5. Automação de Teste. I. Ricardo Rios Monteiro do Carmo, MSc. (Orient.). II. Ricardo de Albuquerque Pinto, Eng. Esp. (Coorient.). III. Universidade do Estado do Amazonas. IV. Uma Proposta de Automação de Testes Android Mobile para a Estrutura de Testes da Google

Elaborado por Jeane Macelino Galves - CRB-11/463

FOLHA DE APROVAÇÃO

“Uma proposta de automação de testes Android Mobile para a estrutura de testes da Google”

ABDA MYRRIA DE ALBUQUERQUE

Trabalho de Conclusão do Curso de Engenharia da Computação defendido e aprovado pela banca avaliadora constituída pelos professores:

Presidente

Prof. Ricardo Rios Monteiro do Carmo, Mestre.

UNIVERSIDADE DO ESTADO DO AMAZONAS

Arguidor

Prof. Fábio Santos da Silva, Doutor

UNIVERSIDADE DO ESTADO DO AMAZONAS

Arguidor

Prof. André Luiz Printes, Mestre

UNIVERSIDADE DO ESTADO DO AMAZONAS

Manaus, 30 de maio de 2022.

Agradecimentos

Gostaria de agradecer primeiramente a Deus que tem me sustentado e abençoado durante essa jornada.

Aos meus pais, começando pela minha mãe Aleida Pinto por toda gentileza, conselhos, amor, pelos seus lanches maravilhosos e pelas orações feitas para minha proteção e segurança durante minhas idas a faculdade. Também quero agradecer ao meu pai Ricardo Pinto, o qual também é engenheiro, que me orientou e me auxiliou na construção da automação e em todas as adaptações de dispositivos. Tendo toda a paciência de ouvir minhas explicações sobre o funcionamento do braço e a implementação do código, além de dar sugestões para o melhor funcionamento. E agora posso orgulhá-lo com a minha formação em engenharia assim como ele.

A minha irmã mais nova Sofia, pela paciência e carinho em ter que aguentar a luz ligada do quarto durantes minhas noites de estudos.

Ao meu amado noivo, Matheus Lobato, que esteve comigo durante toda a escrita desse trabalho. Com toda a paciência, dedicação, amor e com seu bom humor, dedicou horas de seu tempo para revisar minha escrita e me auxiliar no ensaio da minha apresentação.

Ao meu orientador, Prof. MSc. Ricardo Rios, pela perseverança e sabedoria ao me guiar no desenvolvimento desse trabalho.

Ao Prof. MSc. Otoniel Mendes pela ajuda nos conceitos físicos desse trabalho.

Aos meus amigos da faculdade, Douglas, Julliana, Myllena, Cássio, Thierry e Orlando, que sempre estiveram me apoiando, e em especial ao Eduardo Morais, que já se encontra com o Senhor, mas que foi muito importante para mim durante o todo do curso, me apresentou o mundo do Linux e do Arduino.

Aos meus amigos do trabalho, que me deram todo o apoio e incentivo para a conclusão desse curso, em especial, a Alice Castro e João Cabral.

Resumo

Teste de *Software* vem sendo requisitado durante o desenvolvimento do projeto. A automação de testes tem sido utilizada para atender demandas em curto prazo porém garantindo a entrega com qualidade. Esse trabalho tem como finalidade propor uma automação do caso de teste do Pacote de Compatibilidade da Google®, chamado Verificação do Vetor de Rotação, cujo objetivo é reduzir as falhas operacionais e otimizar o tempo de entrega de *smartphones* Android®. Também será apresentado o conceito de testes e sua aplicação, assim como o funcionamento dos sensores em *smartphones* e, por fim, descrito o objetivo e o funcionamento do caso de teste VCVR do pacote de compatibilidade em Android® da Google®.

Palavras Chave: Automação, Teste de Software, Automação de Teste, Braço Mecânico, Arduino.

Abstract

Software testing has been required during the development of the project. Test automation has been used to meet short-term demands while ensuring quality delivery. This work aims to propose an automation of the Google® Compatibility Pack test case, called Rotation Vector Verification, whose objective is to reduce operational failures and optimize delivery time of Android® *smartphones*. The concept of tests and its application will be presented, as well as the operation of sensors in *smartphones* and, finally, the objective and operation of the VCVR test case of the Google® Android® compatibility package will be described.

Key-words: Automation, Software Testing, Test Automation, Mechanical arm, Arduino.

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Uma das fases essenciais para o desenvolvimento de um sistema é o Teste de Software. Nessa etapa, a busca por erros ou falhas é realizada com o objetivo de não entregar ao cliente um produto sem que sejam realizadas validações e verificações na sua funcionalidade. Para isso, são empregados processos manuais de teste, nos quais é necessário muito empenho por parte do testador. Quando somente os processos manuais de teste são realizados, há risco de que erros não sejam encontrados.

A automação do processo de teste viabiliza uma qualidade melhor do resultado, devido o teste conseguir realizar uma grande quantidade de comparações entre resultados esperados e resultados obtidos, favorecendo maiores chances de encontrar falhas e erros em menos tempo.

A utilização de recursos automatizados tem sido muito eficaz para garantir qualidade e entrega em atualizações em menos tempo por conta da reprodutibilidade desses recursos (BERNARDO; KON, 2008). Não seria diferente para a utilização de automações de testes em sistemas operacionais de *smartphones*, como, por exemplo, Android®, pois poderia garantir mais segurança no sistema.

Quando se lida diariamente com o desenvolvimento de uma quantidade grande de dispositivos móveis de diferentes modelos, manter a eficiência nos resultados tem se tornado um desafio. Alguns problemas que têm sido observados são os desgastes da equipe e a lentidão para concluir os testes. Esse é o caso do teste de Verificação do Vetor de Rotação (VCVR) de *smartphones* Android® que é apresentado no Capítulo 2. Testes de VCVR exigem estabilidade e precisão.

Atualmente, o teste VCVR é feito manualmente e atender esses requisitos, bem como alcançar eficiência com redução de erros, com testes não automatizados não é uma tarefa fácil.

Neste trabalho, é proposto um mecanismo de automação de teste do VCVR de *smartphones* Android®. Esse mecanismo deve ser capaz de atender aos requisitos de estabilidade e precisão, bem como eficiência e redução de erros.

1.1 Objetivo Geral

Apresentar uma proposta de automação de teste do VCVR para dispositivos móveis Android®, com o intuito de garantir estabilidade e precisão do processo de avaliação do *software* nesse tipo de teste.

1.1.1 Objetivos Específicos

- Aumentar a precisão no resultado do teste;
- Reduzir as falhas operacionais durante os testes.
- Determinar os possíveis ganhos no tempo de execução;

1.2 Justificativa

O Teste de *Software* é uma das mais antigas e fundamentais ferramentas para a entrega de um produto de qualidade. Ao adotar um processo de Ciclo de Vida de Teste de Software (CVTS) para certificação da qualidade do produto, uma empresa de desenvolvimento deve garantir que todas as atividades de cada etapa do processo sejam realizadas de forma planejada e sistemática.

O processo de CVTS procura aumentar a produtividade na fase de desenvolvimento de *software*, tornando os processos dessa etapa mais ágeis e eficazes. Para isso, recorre-se à Automação de Testes, que consiste na adesão de estratégias e ferramentas (*software* e *hardware*) capazes de controlar e gerenciar testes. A automação permite, por exemplo, realizar a configuração de

pré-condições para cada teste, definir funções de controle e apresentação de relatórios, dentre outras tarefas.

No que se refere a dispositivos móveis, especificamente os que adotam o sistema Android®, desenvolvidos pela empresa Google®, testes automatizados são oferecidos por essa empresa por meio de uma Interface de Programação de Aplicativos (API, em inglês) chamada de "Pacote de Teste de Compatibilidade" – *Compatibility Test Suite* (CTS). Nessa API, apesar de vários testes automatizados estarem definidos, alguns não possuem automação. Esse é o caso do teste de Verificação do Vetor de Rotação (VCVR) – *Rotation Vector Crosscheck*. O VCVR valida a orientação do dispositivo em relação às coordenadas do eixo da Terra.

O teste de VCVR requer estabilidade e precisão durante o manuseio do dispositivo. Devido à imprecisão na realização do teste manual, são necessárias várias re-execuções desse caso de teste até a obtenção do sucesso. Nesse contexto, esse trabalho tem por objetivo apresentar uma proposta de automação de teste do VCVR para dispositivos móveis Android®.

1.3 Metodologia

Para o desenvolvimento desse trabalho, assim como a consequente elaboração de um trabalho monográfico, foi necessário inicialmente uma investigação sobre a necessidade de automação nos testes de VCVR diante do atual cenário de testes funcionais realizados manualmente.

O primeiro método adota uma abordagem qualitativa-quantitativa, ou mista, onde foi abordado a percepção do testador ao realizar o caso de teste. O segundo método a ser utilizado é o exploratório. A partir da percepção do testador, foram investigadas as condições e resultados atuais dos processos de testagem com o objetivo de identificar dificuldades e alternativas ainda não consideradas. Em sequência, foi elaborada uma proposta de solução de automação de teste através do referencial teórico, que permeia os campos de automação, sistemas embarcados e conceitos de testes de *software*.

Concomitantemente foi realizado a modelagem do diagrama de ligação do protótipo de rotação, utilizando um *software* livre de simulação a ser definido. Como plataforma de prototipagem foi utilizado o Arduino® e o ambiente de desenvolvimento Arduino IDE.

Por fim, com o Pacote de Teste de Compatibilidade – *Compatibility Test Suite* (CTS, em inglês) instalado em um dispositivo móvel foi verificado se o protótipo atende aos ângulos de rotação necessários e se atende aos requisitos do teste.

1.4 Materiais

- Computador;
- Motor de passo;
- Arduino UNO Atmega 328;
- Botão Liga e Desliga;
- Driver do motor de passo;
- Gancho para suporte do celular;
- Linguagem de Programação C;
- Fonte de alimentação DC;
- Software Arduino IDE;
- CTS Verifier – Verificador do CTS programa disponibilizado pela Android Developers;
- Base de madeira para o suporte;
- Haste de Alumínio.

1.5 Estrutura da Monografia

Este trabalho está estruturado da seguinte forma. O Capítulo 2 trata dos conceitos de teste de *software*, abrangendo os tipos de abordagens e estratégias. Ressalta, de maneira geral, a aplicação da automação de testes para *smartphones* e apresenta o pacote de compatibilidade

oferecido pela Android Developers, especificamente o funcionamento do Caso de Teste VCVR e a utilização de seus sensores. O Capítulo 3 são apresentados os trabalhos relacionados. O Capítulo 4 apresenta uma propostada de automação do Caso de Teste VCVR. O Capítulo 5 apresenta os experimentos e resultados obtidos. E por fim no Capítulo 6 apresenta a conclusão desse trabalho e trabalhos futuros.

Capítulo 2

Fundamentação Teórica

2.1 Teste de *Software* em *Smartphones*

O teste de *software* visa checar se as funcionalidades definidas no projeto correspondem ao que foi desenvolvido. Sommerville (2016, p. 227) define que o resultado obtido pela execução do teste pode informar se há erros, anomalias ou informações sobre os atributos não funcionais do *software*.

Um *software* pode ser testado tanto manualmente, quanto automaticamente. Testes manuais servem para comparar o comportamento do sistema a cada entrada, determinada pelo testador, com o resultado que ele espera. Testes automatizados são roteiros que contém determinadas ações e entradas que podem ser executadas diversas vezes em diferentes versões do *software*. Com isso, o testador é capaz de verificar se o *software* atende os requisitos funcionais e não funcionais, assim como validar se corresponde com as expectativas do cliente (SOMMERVILLE, 2016, p. 228).

Apesar de testes automatizados serem mais rápidos e de conseguirem aplicar diversas entradas e ações, eles não podem ser completamente automatizados. s testes apenas verificam se o *software* faz o que deve fazer. Existem alguns testes que só podem ser feitos manualmente, tal como os baseados na experiência do usuário, pois buscam saber se o *software* é intuitivo.

Atualmente, muitas empresas optam por investir em pesquisa e desenvolvimento de testes de *software* para dispositivos móveis, como, por exemplo, *smartphones*. Os dispositivos são

como minicomputadores com muitas capacidades e funcionalidades em termos de *hardware* e *software* (KNOTT, 2015, p. 23).

Considerando que há uma grande variedade de perfis do sistema operacional Android®, e ainda, uma expressiva diversidade de modelos, tal variedade possibilita, mesmo que minimamente, que o usuário se depare com eventuais falhas, o que poderia levar a uma desconfiança e consequente insegurança por parte do usuário.

Baseado nessa complexidade, a Google® oferece um pacote de testes para *smartphones* Android® com o qual valida suas exigências para todas as personalizações oficiais lançadas no mercado, buscando, no primeiro momento, eliminar as incidências de falhas devido ao uso cotidiano por parte dos usuários, ou, subsidiariamente, diminuir as chances de serem percebidas.

2.1.1 Tipos de Abordagens de Testes

Uma estratégia para teste de *software* é um planejamento das etapas de teste pelas quais o produto deve passar. Uma equipe de testadores, munida de um planejamento, consegue aplicar os métodos de testagem de forma eficiente, diminuir o tempo de espera, administrar recursos (testadores e/ou equipamento de teste) e obter informações a partir dos resultados. Dentre as várias abordagens ou estratégias de teste *software* que, segundo Pressman (2015), são as mais comuns estão:

- Teste Unitário;
- Teste de Integridade;
- Teste de Validação;
- Teste de Sistema.

O Teste Unitário tem por finalidade testar o código do sistema. No código, são testados componentes, funcionalidades, módulos do sistema, entre outros. Na Figura 2.1, o teste unitário começa no vórtice do espiral. O Teste de Integração, por sua vez, tem por foco o projeto e a construção da arquitetura do sistema. teste verifica como os componentes trabalham quando

integrados a outros componentes. O Teste de Validação valida os requisitos estabelecidos no início do projeto. E o Teste de Sistema tem por foco o sistema como um todo, como produto.

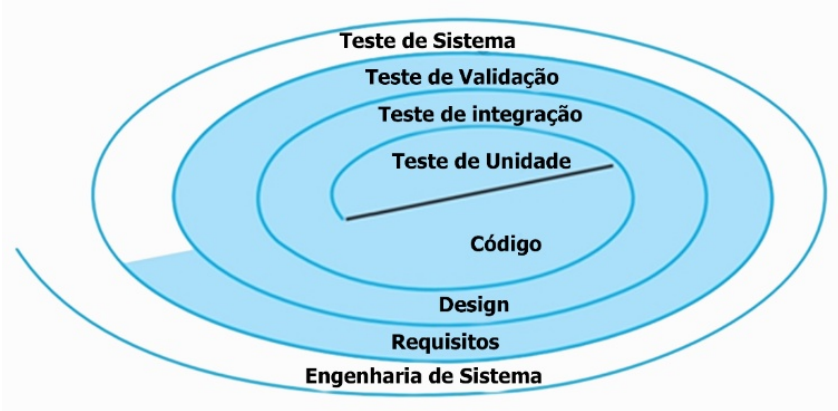

Figura 2.1: Estratégia de Teste.(PRESSMAN, 2015)

O *Compatibility Test Suite* ou CTS, por exemplo, é um conjunto de testes unitários projetados e aplicados para revelar incompatibilidades, desde as etapas iniciais do desenvolvimento do *software*. pacote tem por objetivo garantir que o *software* permaneça compatível ao longo de todo o processo. Dentro dele encontra-se o teste que se busca automatizar com o desenvolvimento do trabalho.

2.2 Automação de Teste

A independência da intervenção humana permite o aproveitamento dos benefícios de um computador, como, por exemplo, velocidade de execução, reprodutibilidade exata de um conjunto de ações, possibilidade de execução paralela de testes, flexibilidade na quantidade e momento das execuções dos testes e facilidade de criação de casos completos (BERNARDO; KON, 2008, p. 2).

Cada falha encontrada exige uma nova solução e um novo caso de teste para validar. E com essa repetição manual é possível que os testadores deixem passar algum erro devido à exaustão.

Ao permitir que a automação exerça essa atividade, pode-se exigir maior precisão, reprodução e qualidade. Segundo Bernardo e Kon (2008), a reprodutibilidade de testes permite simular identicamente e inúmeras vezes situações específicas, garantindo que passos importantes não sejam ignorados por falha humana e facilitando a identificação de um possível comportamento não desejado. Os autores Nagowah e Sowamber (2012) também afirmam que uma ferramenta de automação de teste automatiza as etapas usuais que são envolvidos em um teste.

De acordo com Haller (2013) existem duas razões para automatizar os casos de teste de dispositivos móveis: para garantir o mínimo de cobertura funcional e para obter cobertura de configuração de teste escalável.

O CVTS dos projetos é apresentado na estrutura ilustrada na Figura 2.2. Inicialmente, é necessário que sejam definidos os requisitos funcionais e não funcionais com o cliente. Deve-se avaliar s requisitos minuciosamente para que possa ser definido o "Plano de Testes". No seguimento, identifica-se quais tipos de abordagens de testes serão utilizados para poder desenvolver os casos de testes, para em seguida executá-los e obter um relatório sobre o que atendeu aos requisitos e o que não atendeu.

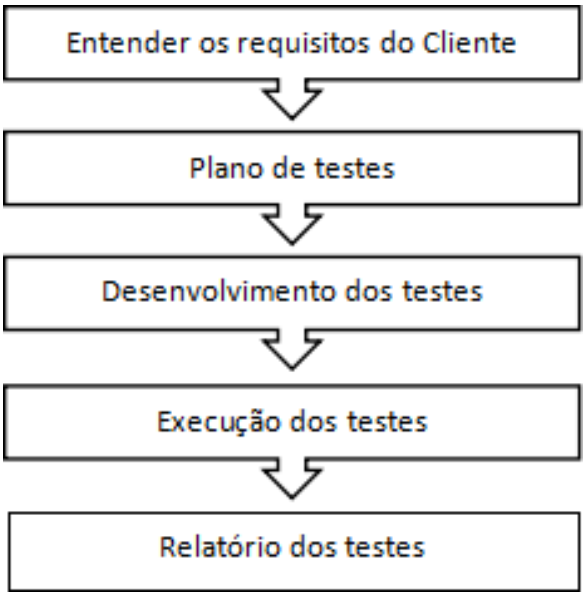

Figura 2.2: Ciclo de Vida dos Testes.

Na fase de Requisitos do Cliente, é necessário fazer o levantamento de as todas as características que o produto deverá ter para atender a necessidade do cliente, extraindo o máximo

de detalhes para que possa prosseguir para a próxima fase. No Plano de Testes é definida a estratégia que será melhor aplicada para cada requisito. No Desenvolvimento dos Testes é onde inicia a programação de roteiros (*scripts*, em inglês) com a descrição os casos de testes. Na fase de Execução de Testes, o testador, cuja sua função é realizar a homologação dos resultados dos testes elabora o Relatório dos Testes que será repassado para a equipe de desenvolvimento ou para o analista de requisitos.

Entende-se, portanto, que a automação beneficia a aplicação dos métodos de testagem, por aplicar-lhes maior precisão e qualidade nos resultados.

2.3 Sensores em *Smartphones*

Smartphones possuem vários sensores, que são integrados ao sistema e geram dados "brutos" com alta precisão. São úteis para monitorar o movimento, ou o posicionamento tridimensional do dispositivo, ou ainda para acompanhar as mudanças no ambiente ao redor, (DEVELOPERS, 2020b). O sistema Android® é compatível com três categorias de sensores:

- Sensores de Movimento: possuem a capacidade de medir as forças rotacionais e de aceleração nos três eixos. Os sensores mais conhecidos nessa categoria são os acelerômetros, os sensores de gravidade, os giroscópios e os sensores vetoriais de rotação;
- Sensores Ambientais: medem os vários parâmetros encontrados em determinado ambiente, tal como temperatura, pressão do ar, iluminação e umidade. Nessa categoria estão os barômetros, fotômetros e termômetros; e
- Sensores de Posição: responsáveis por medir a posição física de um aparelho. Nessa categoria estão os sensores de orientação e os magnetômetros.

Para o atingir o objetivo deste trabalho, a análise focou em duas categorias, os sensores de movimento e os de posição.

2.3.1 Sensor de Movimento: Acelerômetro

Um acelerômetro tem como função medir a aceleração do dispositivo. O sensor é composto por outros menores, os quais incluem estruturas de cristais de tamanho microscópio que são alvo do estresse resultante da influência da força de aceleração. Diante do fenômeno, o sensor calcula o nível de tensão por estresse, que é originado nos cristais, e informa ao sistema a velocidade que o *smartphone* se movimenta em um determinado momento e ainda a direção de sua trajetória. São duas forças conhecidas como aceleração estática e aceleração dinâmica.

A medição da aceleração estática em relação à gravidade possibilita encontrar qual o ângulo no qual o dispositivo se encontra em relação ao eixo da Terra. Ao registrar uma variação na aceleração em relação a eixo, pode-se identificar se o dispositivo está em movimento ou não. A partir disso, o funcionamento do acelerômetro pode ser evidenciado e compreendido de diversas formas, pois existem inúmeros tipos de aplicações com este sensor. Porém, para a melhor compreensão e adequação ao objeto de estudo, trabalho atem-se a aplicação em *smartphones*.

Em *smartphones* há uma placa de circuito integrado (Figura 2.3) capaz de detetar as variações na capacitância de duas microestruturas eletrônicas, identificadas em amarelo e azul, que se encontram próximas umas das outras. Havendo aplicação de força para movimentar as estruturas, a capacitância é modificada. Essa informação é processada e infomada por meio dos software do *smartphone*.

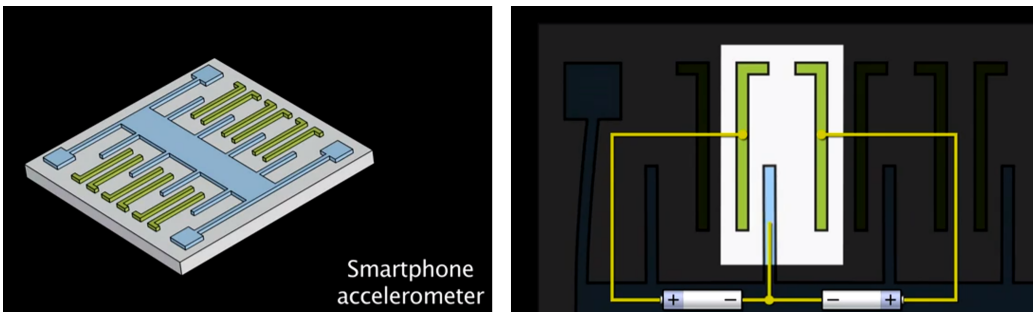

Figura 2.3: Circuito de um acelerômetro.(HAMMACK, 2012)

A placa, ilustrada na Figura 2.3, é composta pela *"Seismic Mass"* (Massa Sísmica, em tradução livre) em formato de "H". Ao sofrer algum estímulo de movimento em massa, vibra sobre a base da placa, a qual é denominada *"Housing"*. A cada vibração, as pontas no meio,

em azul, aproximam-se dos capacitores, em amarelo, fazendo com que a capacitância sofra alterações. O circuito capta essas vibrações e calcula o diferencial da tensão em cada capacitor, retornando um valor para a saída do acelerômetro. Cada valor é proporcional à movimentação do mesmo (HAMMACK, 2012).

Além disso, de acordo com Developers (2020a), é comum aos sensores de movimento retornarem matrizes multidimensionais de valores sensoriais para cada *SensorEvent* traduzido como Evento no Sensor, que é uma função no código que captura qualquer ação de movimento realizada pelo usuário. Tem-se assim que, ao longo de um único evento de sensor, o acelerômetro retorna os dados de força da aceleração para os eixos euclidianos, bem como o giroscópio retorna a taxa dos dados de rotação para aqueles eixos.

2.3.2 Sensor de Movimento: Giroscópio

Um giroscópio pode ser comumente definido como um corpo sólido capaz de girar em alta velocidade angular sobre um eixo que sempre passa por um ponto fixo. O ponto fixo pode ser o centro de gravidade do sólido ou pode ser qualquer outro ponto no espaço. Mais especificamente, a forma usual de giroscópio é um dispositivo mecânico cuja parte inicial é um volante com um aro pesado, montado de forma que, enquanto gira em alta velocidade, seu eixo de rotação pode girar em qualquer direção sobre um ponto fixo daquele mesmo eixo, conforme pode ser observado na Figura 2.4.

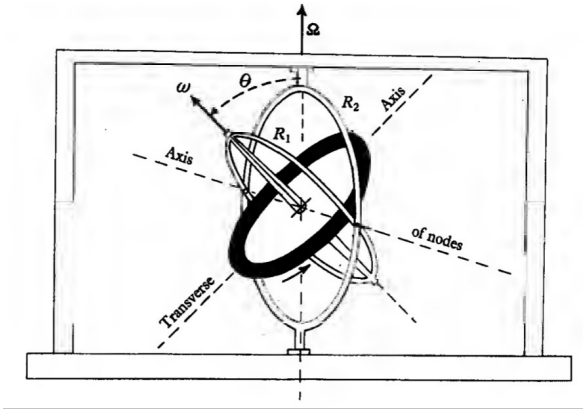

Figura 2.4: Representação do conceito de um Giroscópio. (SCARBOROUGH, 1958)

Em um *smartphone*, segundo Developers (2020a), o giroscópio informa a taxa de rotação do dispositivo em torno dos três eixos do sensor. E não pode ser emulado com base em magnetômetros e acelerômetros, pois isso faria com que ele tivesse consistência e capacidade de resposta locais reduzidas. É necessário que ele esteja baseado em um circuito integrado de giroscópio usual.

2.3.3 Sensor de Posição: Magnetômetro

O magnetômetro é um instrumento científico usado para medir a intensidade e a direção dos campos magnéticos. Uma de suas aplicações mais evidentes é a utilização em observações científicas do campo magnético da Terra, que podem sofrer variações em diferentes condições. Os *smartphones* conseguem encontrar o Norte na bússola utilizando o sensor magnetômetro (Figura 2.5).

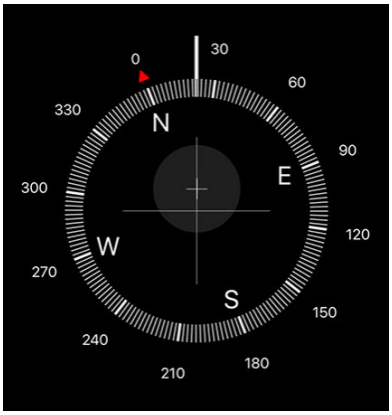

Figura 2.5: Bússola do *smartphone* usando Magnetômetro.(NIELD, 2017)

Além disso, tais dispositivos são capazes de mensurar o impacto de diversas atividades humanas que interferem no campo magnético terrestre. N ponto, pesquisas de longo curso mantêm os sensores em uma execução contínua com o objetivo de coletar dados suficientes para comparar as variações ao longo do tempo. Conforme Guilherme (2019) tais análises são possíveis, pois, além de realizar medições nos três eixos do plano cartesiano, o magnetômetro é muito sensível e assim muito preciso.

De acordo com Developers (2020a), o VCVR utiliza o acelerômetro e o magnetômetro para compensar o desvio do giroscópio, e caso de teste não pode ser implementado usando apenas o

acelerômetro e o magnetômetro.

2.4 Plataforma Arduino

De acordo com McRoberts (2018), um Arduino (Figura 2.6) é um pequeno computador que pode ser programado para processar entradas e saídas entre o dispositivo e os componentes externos conectados a ele.

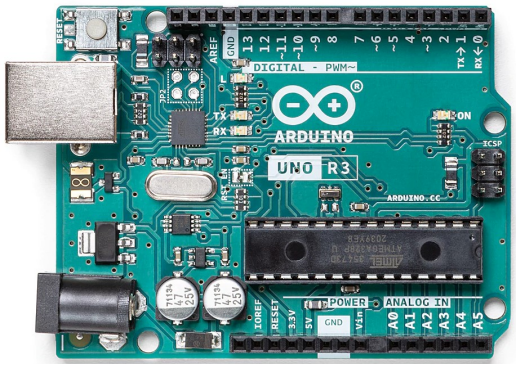

Figura 2.6: Arduino (ARDUINO, 2018)

O Arduino também pode ser definido como uma plataforma de computação física ou embarcada, ou ainda, um sistema que se relaciona com o ambiente em que foi aplicado por meio de *hardware* e *software* (MCROBERTS, 2018).

O Arduino pode ser utilizado em diversas áreas, como, por exemplo, automações residenciais, fazê-lo conectar-se à Internet e enviar dados para sensores ou capturar dados da rede e informar algum dispositivo, como, por exemplo, um *smartphone*. Também pode ser usado didaticamente para que estudantes tenham contato com microcontroladores, sensores e atuadores.

Isso é possível pois o *hardware* e *software* são de fonte aberta. Por isso, existem diversas plataformas semelhantes ao Arduino e há vários códigos disponíveis nas comunidades dessas plataformas e que estão disponíveis para a utilização livre para qualquer propósito. Assim, o Arduino tem uma vantagem, além de seu custo benefício, possui a facilidade de qualquer pessoa aprender a desenvolver com a ajuda da comunidade, não tendo como pré-requisito ser um programador, pois ocorre uma evolução de conhecimento desde os projetos básicos até os mais complexos.

2.5 Motores de Passo

Os motores de passo, segundo Santos (2008), são dispositivos eletro-mecânicos que convertem pulsos elétricos em movimentos mecânicos que geram variações angulares discretas. McRoberts (2018, p.238) também define que os motores de passo são um motor de corrente contínua que possuem sua rotação dividida em uma série de passos, permitindo que ele rotacione em um número definido de momentos, levando a um controle mais preciso de velocidade e giros, o que o difere dos motores padrão.

Um motor de passo possui um rotor e um estator. O rotor é construído com um ou mais imãs caracterizados como permanentes, pois não perdem seu campo magnético e o estator é composto por conjunto de fios enrolados, ou enrolamento, no qual sua polaridade sofrerá alterações entre pólo positivo e o negativo. O motor de passo pode existir nos mais variados formatos e tamanhos, dependendo da necessidade e complexidade do mecanismo. Os unipolares e bipolares são os utilizados mais frequentemente.

Observando a Figura 2.7A, constata-se que existem quatro bobinas no motor, sendo que existe um fio, indicado pela letra C, em cada bobina que vai para a fonte de alimentação. Os demais fios são ligados ao pólo negativo um por vez enquanto o fio C recebe o pólo positivo fazendo com que o rotor rotacione, como ilustrado na Figura 2.7 no sentido A, B, C e D.

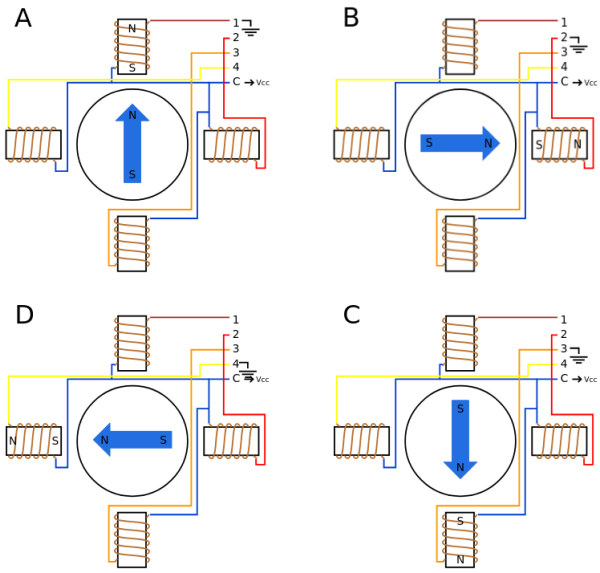

Figura 2.7: Motor de Passo Unipolar (TUBES, 2004)

Em motores bipolares não existe este fio C em cada bobina ligado à fonte de alimentação. Cada bobina é conectada por um fio a seu par, como ilustra a Figura 2.8. Enquanto uma bobina recebe a alimentação positiva ($HIGH$), ou "pulsos", atravessando para seu par, outra bobina recebe o pólo negativo (LOW) que vale também para o seu par conectado, e alternando bobinas para rotacionar o rotor, de A,B,C,D como na Figura 2.8

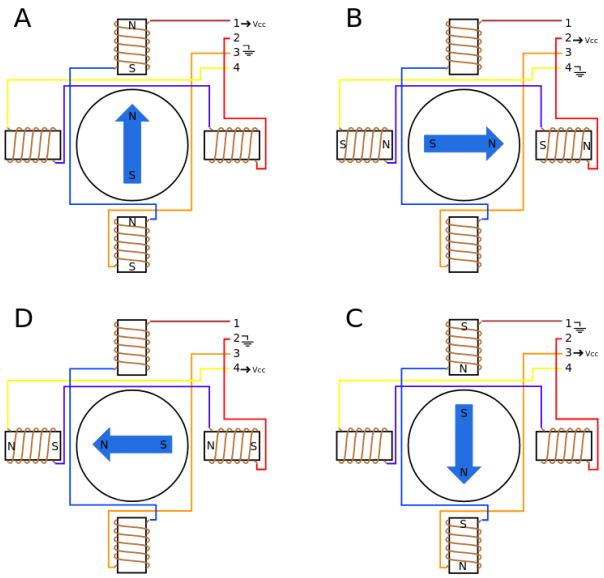

Figura 2.8: Motor de Passo Bipolar (TUBES, 2004)

O trabalho do Arduino é aplicar os comandos $HIGH$ e LOW apropriados às bobinas, na sequência e velocidade corretas, para permitir que o eixo rotacione.

A velocidade do motor está diretamente relacionada a s pulsos. Essa velocidade é definida pela frequência em que os pulsos são enviados e o número de revoluções, ou voltas, do eixo é definido pelo número de pulsos. O dispositivo que controla os pulsos elétricos enviados ao motor é chamado de *driver* (MOTION, 2017).

2.5.1 Tipos de Motores

Os motores de passo podem ser de quatro fios, seis fios ou oito fios. Os motores de quatro fios só podem ser bipolares em série, os motores de seis fios podem ser bipolares em série ou unipolares. Os motores de oito fios podem ser bipolares em série, bipolares paralelos ou unipolares. Conexões paralelas podem atingir velocidades mais altas devido à menor indutância

nas bobinas. Quando conectado em série, o torque (*Seção 2.7*) de baixa velocidade é igual ou ligeiramente maior do que em paralelo, e menos corrente é consumida (MOTION, 2017).

NEMA	MODELO	CONEXÃO		HOLDING TORQUE (kgf.cm)	CORRENTE (A/fase)	TENSÃO (V/fase)	RESISTÊNCIA (Ω/fase)	INDUTÂNCIA (mH/fase)
17	AK17/1.1F6LN1.8	Bipolar	Série	1,1	0,07	0,017	140	148
		Unipolar		0,77	0,1	0,012	37	37
23	AK23/4.6F6FL1.8	Bipolar	Série	4,6	0,7	7	10	24,8
		Unipolar		3,2	1	5	5	6,2
	AK23/7.0F8FN1.8	Bipolar	Série	7	1	5	2,4	9,2
			Paralelo		2	2,5	0,6	2,3
		Unipolar		4,9	1,4	3,5	1,2	2,3
	AK23/15F6FN1.8	Bipolar	Série	15	2,1	4,2	2	8
		Unipolar		10,5	3	3	1	2
	AK23/21F8FN1.8	Bipolar	Série	21	2,8	3,36	1,2	11,2
			Paralelo		5,6	1,68	0,3	2,6
		Unipolar		14,7	4	2,4	0,6	2,8
34	AK34/32F6BB1.8	Bipolar	Série	32	2,4	8,2	3,36	20,96
		Unipolar		22,4	3,5	5,88	1,68	5,24
	AK34/42F8FN1.8	Bipolar	Série	42	2,94	4,7	0,4	14
			Paralelo		5,88	2,35	1,6	3,5
		Unipolar		29,4	4,2	3,36	0,8	3,5
	AK34/52F4CN1.8	Bipolar	Série	52	5	3,75	0,75	6,4
	AK34/100F8FN1.8	Bipolar	Série	100	2,1	11,2	5,34	55,2
			Paralelo		4,2	5,6	1,36	13,8
		Unipolar		70	3	8	2,67	13,8

Figura 2.9: Tabela de Motores (MOTION, 2017)

Abaixo lista-se as informações técnicas de alguns dos motores citados na Figura 2.9.

- Nema 17 — AK17/1.10F6LN1.8 Observa-se a velocidade de rotações por minuto (*RPM*) d motor em relação ao torque (*Seção 2.7*) e sua ligação bipolar série ou unipolar.

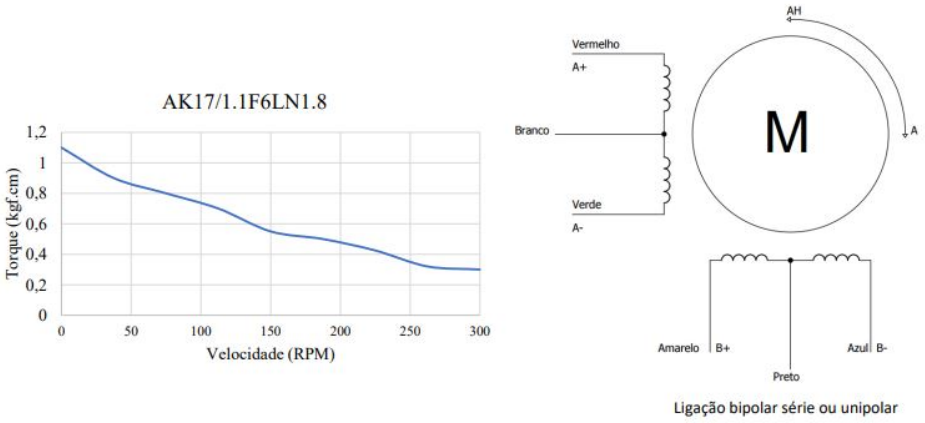

Figura 2.10: Informações Técnicas do Nema 17. (MOTION, 2017).

- Nema 23 — AK23/4.6F6FL1.8 O mesmo também pode-se observar a relação entre a velocidade de rotações por minuto (*RPM*) d motor e o torque (*Seção 2.7*) e sua ligação série ou unipolar.

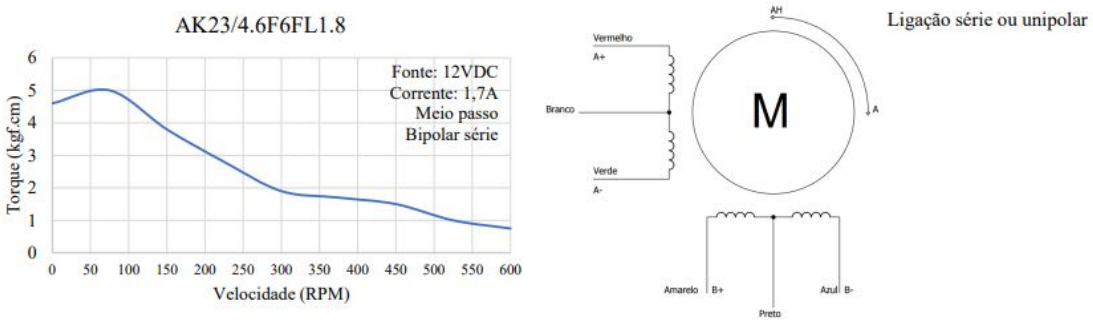

Figura 2.11: Informações Técnicas do Nema 23.(MOTION, 2017).

2.5.2 Controlador de Motores de Passo

Um controlador ou *driver* (Figura 2.12) de motor de passo funciona através de uma placa geradora que controla os pulsos utilizados pelos motores. Um deles é o *driver* com o chip A4988. O A4988 é responsável pelo micropasso completo com conversor embutido para fácil operação. Ele é projetado para operar motores de passo bipolares nos modos de passo completo, meio, um quarto, um oitavo e um dezesseis avos de passos com capacidade de acionamento de saída de até 35V e ±2A. O A4988 inclui um regulador de corrente de tempo fixo capaz de operar em modo de decaimento lento ou decaimento misto (MICROSYSTEMS, 2009).

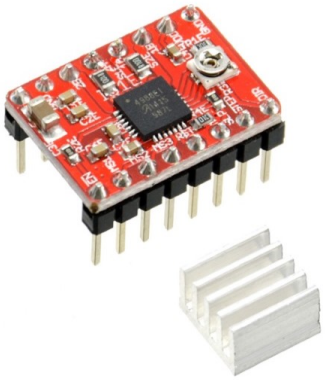

Figura 2.12: *Driver* de motor de passo A4988 e um dissipador de calor autoadesivo. (COELHO, 2021)

2.6 Conceitos Físicos — Momento e Centro de Massa

Para o desenvolvimento deste trabalho são necessários alguns conceitos físicos a respeito de Momento e Centro de Massa para condições de equilíbrio. De acordo com Meriam, Kraige e Bolton (2020), a força **F** é uma grandeza vetorial, porque seu efeito depende da direção e do módulo da ação (Figura 2.13).

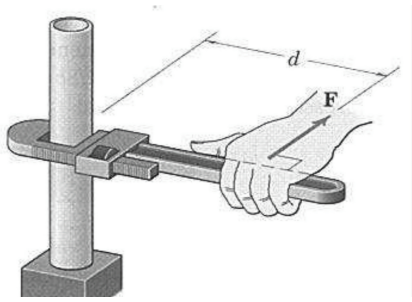

Figura 2.13: Momento *Torque*. (MERIAM; KRAIGE; BOLTON, 2020)

O Momento **M**, também conhecido como torque τ, é a força exercida sobre um objeto. Essa força tende a fazer com que os objetos girem em torno de um eixo. O eixo pode ser qualquer reta que não cruze ou não seja paralela ao vetor de ação da força. A magnitude do momento é proporcional à magnitude da força e ao braço de alavanca d, que é a distância vertical do eixo ao vetor de ação da força (MERIAM; KRAIGE; BOLTON, 2020).

A fórmula do Torque é:

$$\tau = F \times d \tag{2.1}$$

onde d é o comprimento do braço de alavanca e F é a força.

Segundo Hibbeler (2017), para que haja equilíbrio, a soma de todas as forças externas deve ser igual a zero e a soma de todos o Momentos deve ser também igual a zero.

A fórmula do centro de massa é:

$$CM_x = \frac{x_1 m_1 + x_2 m_2 + ... + x_n m_n}{m_1 + m_2 + ... + m_n} \tag{2.2}$$

$$CM_y = \frac{y_1 m_1 + y_2 m_2 + ... + y_n m_n}{m_1 + m_2 + ... + m_n} \tag{2.3}$$

$$CM = (CM_x, CM_{y)} \tag{2.4}$$

CM_x = Posição do centro de massa em x; CM_y = Posição do centro de massa em y; x_1, x_2 = Posição em x; y_1, y_2 = Posição em y; m = massa; CM = Ponto de centro de massa.

2.7 Caso de teste *Rotation Crosscheck Vector*

O caso de teste "Verificação Cruzada da Visão Computacional" (*Rotation Crosscheck Vector*, em inglês), também conhecido como "Verificação do Vetor de Rotação", ou simplesmente VCVR, está disponibilizado no sítio de Desenvolvedores Android® da Google® (DEVELOPERS, 2020b). Ele é utilizado para verificar e validar se o *smartphone* pode detetar sua própria orientação no espaço, obtendo inicialmente uma posição de referência e, após o deslocamento, sua posição final (DEVELOPERS, 2020b).

A tela inicial do caso de teste VCVR indica como preparar o *smartphone* antes de iniciar o procedimento de validação dos sensores incorporados. Primeiramente, deve-se ativar o "Modo Avião", em seguida, desativar o "Brilho Automático", a "Rotação Automática", o "Permanecer Ativo", encontrado no modo desenvolvedor do Android®, e a "Localização", como pode ser observado na descrição dos testes inicias, conforme identificado em verde na Figura 2.14. Também é necessário instalar a APK do *Open Source Computer Vision Library* (*OpenCV*), que fornece uma biblioteca de visão computacional otimizados em tempo real, de acordo com a arquitetura do *smartphone*.

Após a preparação do *smartphone*, é importante observar a posição de referência indicando o lugar exato para iniciar o procedimento de teste, com utilização do papel chamado "Padrão de Imagem", representado na Figura 2.15.

padrão, que deve ser impresso em papel e deve estar sobre uma superfície plana, serve de ponto de referência para quando o testador movimentar o *smartphone* sobre ele, seja mantido o foco da câmera na área determinada no "Padrão de Imagem".

No *smartphone*, tem uma indicação em amarelo que deverá estar alinhada com o marcador

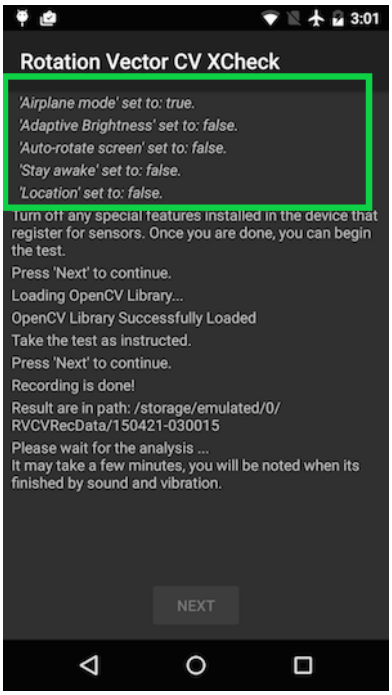

Figura 2.14: Tela inicial do VCVR. (DEVELOPERS, 2020b)

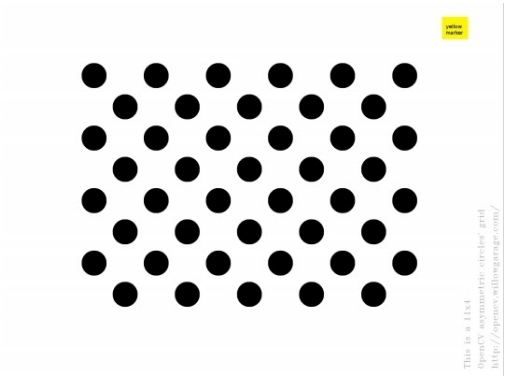

Figura 2.15: Padrão de imagem para teste. (DEVELOPERS, 2020b)

amarelo do "Padrão de Imagem". Recomenda-se manter uma certa distância, para que a imagem fique enquadrada na área estabelecida na tela, como representado na Figura 2.16. Com isso, obtém-se a posição de referência. Na tela do *smartphone* é exibida a "Faixa de Inclinação", nos cantos em azul. Essa faixa limita a inclinação e avisa sonoramente quando chegou ao seu limite.

Enquanto os pontos de referências estiverem alinhados na mesma direção com o "Padrão de Imagem", e o papel estiver completamente dentro dos limites da tela do *smartphone*, deve-se movimentá-lo no espaço euclidiano, que é um espaço vetorial real de dimensão finita, conforme pode ser observado nas etapas de teste (Figura 2.17), obedecendo aos comandos solicitados

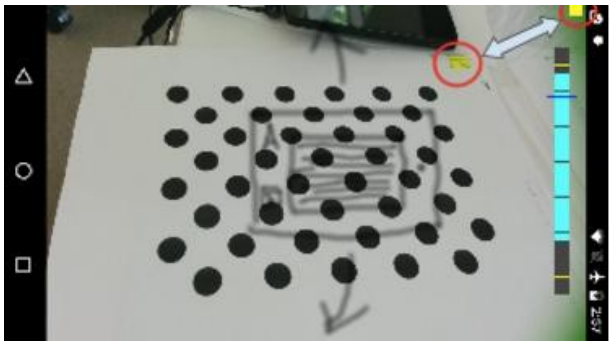

Figura 2.16: Alinhando o teste do *smartphone* com o papel padrão. (DEVELOPERS, 2020b)

pelo indicador de faixa de rotação em azul. Os mesmos movimentos nos eixos usados no VCVR também são aplicados ao controle dos eixos de aeronaves.

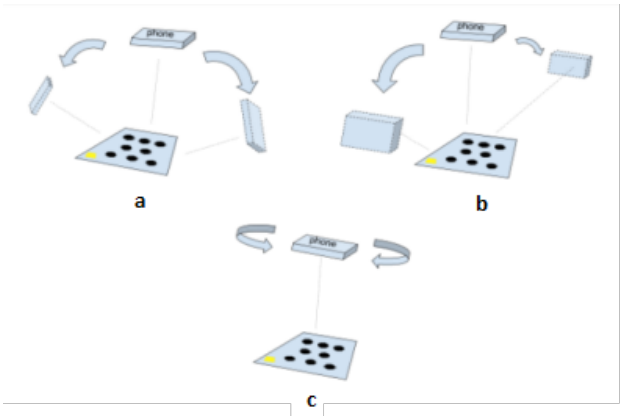

Figura 2.17: Manipulação do *smartphone* durante o teste. (DEVELOPERS, 2020b)

Uma aeronave para permanecer estabilizada no espaço aéreo, usa o sistema de controle que identifica o ponto onde os três eixos (eixo x, eixo y e eixo z) se cruzam, conhecido pelo o nome de Centro de Gravidade (CG), onde está concentrado todo o peso do avião. s três eixos são:

- Eixo longitudinal; em torno d eixo, o avião realiza os movimentos conhecidos como rolagem ou inclinação lateral, e pode ser efetuado para esquerda ou para a direita.
- Eixo transversal ou lateral: em torno d eixo, o avião realiza os movimentos de “arfagem” ou “tangagem”, que são os movimentos levantar a fronte do avião, e baixar a fronte do avião.
- Eixo vertical: em torno do qual, a aeronave realiza o movimento conhecido por “guinada”, que faz com que o avião gire a fronte para a direita ou para esquerda.

Pode-se observar na Figura 2.18, que ambos utilizam os eixos para determinar suas coordenadas.

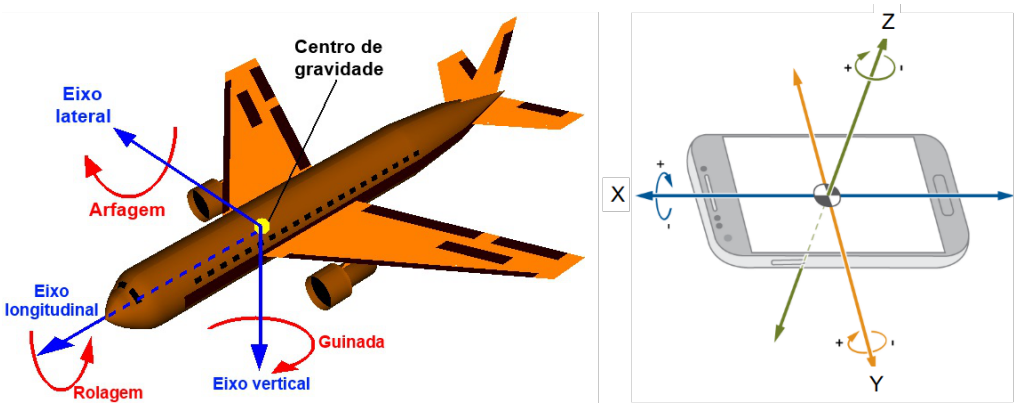

Figura 2.18: Sistema de coordenadas em um avião (COBEL,) e de um *smartphone* (DEVELOPERS, 2020a)

No eixo longitudinal, é realizada a inclinação para cima e para baixo das asas do avião, em torno do seu próprio eixo. O sensor, presente no avião, faz a medição determinando o ângulo de inclinação que a aeronave se deslocou em relação ao ponto zero, conforme ilustrado na Figura 2.19.

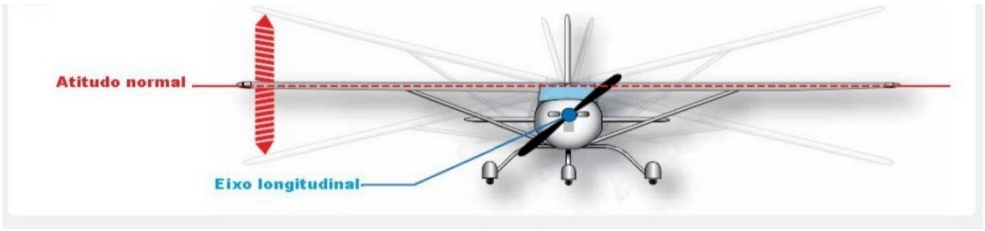

Figura 2.19: Eixo longitudinal. (FAA, 2018).

mesmo movimento (Figura 2.20) é realizado no *smartphone* inclinando para esquerda e para a direita, ou vice-versa.

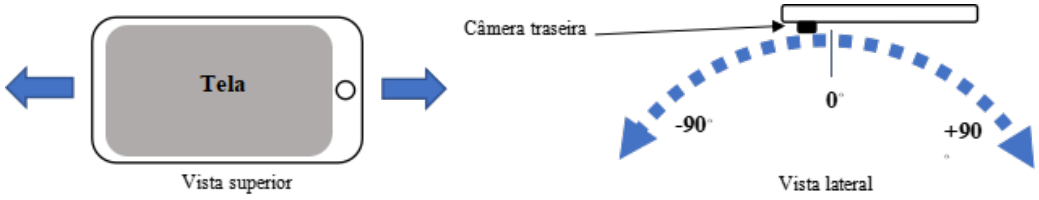

Figura 2.20: Inclinação no Eixo X

No eixo lateral, a arfagem é responsável pelo movimento no eixo Y, como a inclinação para

cima e para baixo da ponta do nariz do avião. Com o movimento para cima, a aeronave está no modo ascendente e da mesma forma se a inclinação da ponta frontal for para baixo, a aeronave está no modo descendente, conforme ilustrado na na Figura 2.21.

No *smartphone* o eixo Y, o movimento de inclinação é para frente e para trás, com um ângulo com aproximadamente 160° de abertura. Tal como no eixo X, a faixa limita a inclinação e avisa sonoramente quando chegou ao seu limite.

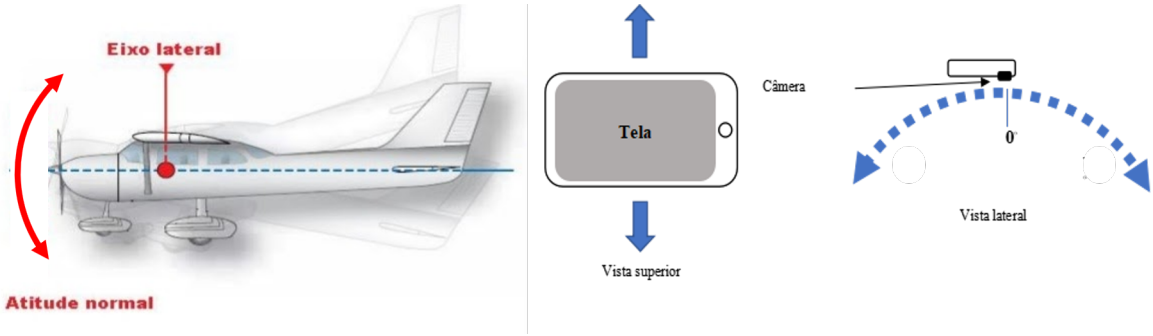

Figura 2.21: Eixo lateral (FAA, 2018) e inclinação do eixo Y.

No último eixo, conhecido como vertical, a guinada é responsável pelo movimento frontal do avião, em relação o eixo Z, fazendo com que a aeronave faça movimentos rotacionais perpendiculares em seu eixo no sentido horário ou anti-horário, conforme Figura 2.22. No eixo Z o movimento de rotação deve ter um ângulo de 360º contínuo. A faixa de rotação avisa sonoramente quando chegou ao seu limite e conclui esta parte do teste iniciando análise do vídeo feito, desde o início da inclinação no eixo X.

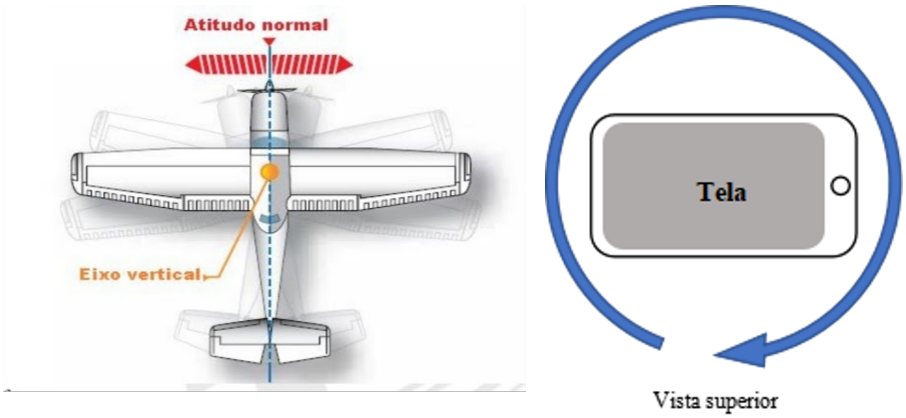

Figura 2.22: Eixo vertical (FAA, 2018) e a rotação no eixo Z.

O centro de gravidade é muito importante para uma aeronave porque sua posição tem

uma forte relação com a estabilidade. No centro onde os três eixos se encontram, cada eixo é perpendicular aos outros dois.

Depois dos movimentos serem executados, iniciasse a análise do vídeo capturado desde o início da execução do teste, por meio da APK OpenCV, que tem uma duração de cinco minutos, podendo variar com desempenho do processador do *smartphone*.

Ao final da análise, o *smartphone* avisará sonoramente e aparecerá um resultado numérico na tela, indicado em verde (Figura 2.23), se a análise for bem-sucedida. E após clicar em *Next*, o VCVR prossegue para a tela de aprovação/reprovação e para revisão do resultado.

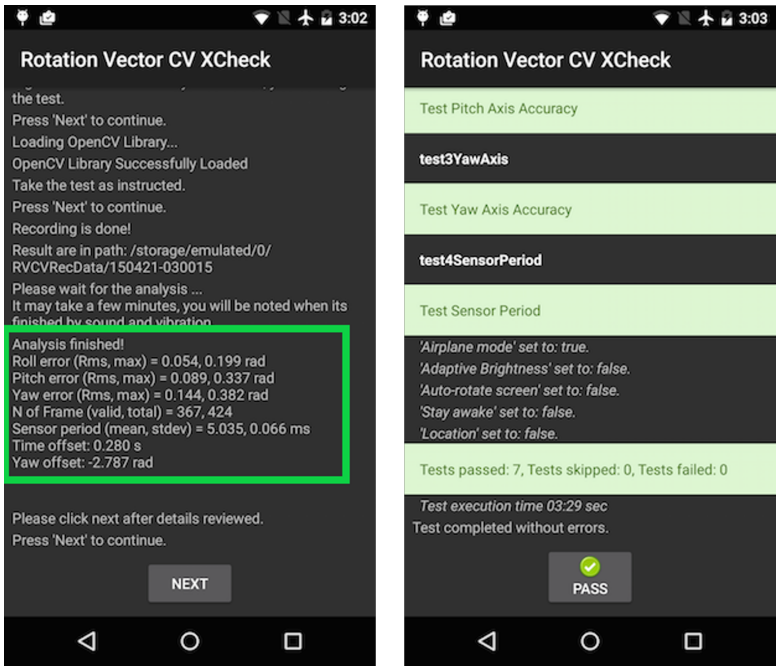

Figura 2.23: Resultado numérico e resultado bem-sucedido. (DEVELOPERS, 2020b)

2.8 Considerações

A proposta de automação do VCVR, citado no trabalho, visa reproduzir os movimentos nos eixos das coordenadas, de acordo com o que é exigido no caso de teste, alcançando este mesmo resultado com um número menor de tentativas. Obtendo valores mais precisos, obedecendo os conceitos físicos citados neste capítulo, com uma proposta de automação bem desenvolvida que constam nos próximos capítulos.

Capítulo 3

Trabalhos Relacionados

Nesse capítulo serão abordadas algumas formas de desenvolver um braço mecânico utilizando a plataforma Arduino.

3.1 Braço Robótico para Testes de Retrato para *Smartphones*

O *"AR(m)obo Test: Braço Robótico para Suporte à Testes automáticos de Retrato e Paisagem para Smartphones"* (BARBOZA, 2016), tem por objeto o desenvolvimento e utilização de uma estrutura que remonta a um braço robótico articulado. Apresenta também um aplicativo para o sistema Android® para controle e testagem do comportamento do *software* e do *hardware* durante as mudanças de orientação do modo retrato para o de paisagem (visão vertical para horizontal) e vice versa.

Assim, para tratar o problema evidenciado, o autor do trabalho desenvolveu uma automação dos testes por meio de um braço robótico. Não apenas isso, mas ele soma ao desenvolvimento do braço a criação de um aplicativo que permita o controle e análise dos resultados. O *AR(m)obo Test*, como é chamado o braço robótico, foi construído a partir de servomotores conectados a uma placa Arduino, modelo MEGA 2560, que é controlada via *Bluetooth* (HC-06) por um aparelho de comunicação celular.

O mecanismo foi construído de modo que sua movimentação e articulações se assemelhassem a de um braço humano, operando com baixa quantidade de potência elétrica, que varia de 10

Volts até 12 Volts, sendo capaz de carregar um peso de até 160 gramas e construído a partir de materiais leves e preferencialmente moldáveis. Por questões de segurança, o braço robótico foi programado para interromper suas atividades diante de eventuais obstáculos. Além disso, para uma melhor execução de sua função, o braço atua em intervalos programados, entrando em modo de repouso após vinte minutos de funcionamento.

Figura 3.1: Braço Robótico para Testes. (BARBOZA, 2016)

Quanto ao *software* desenvolvido, *AR(m)obo Test App*, para exercer sua função de controle remoto do braço mecânico e definir os movimentos necessários para execução dos casos de teste, foi implementado para que o testador tivesse uma interação com o braço, podendo escolher o tipo de movimento pré-definido, ou escolher os ângulos, ou acompanhar o estado da conexão entre o dispositivo e o braço robótico, ou que pudesse realizar a verificação de calibragem antes de dar início ao teste. Além disso, é possível verificar o último teste realizado, a temperatura do braço, os resultados resumidos do teste e um resultado completo com informações sobre posição do braço e dos sensores.

Considerando os padrões necessários para o funcionamento do *hardware* e do *software*, vê-se

que apresenta várias vantagens para uma aplicação ampla na realização das etapas de testagens dos sistemas operacionais, pois o *AR(m)obo Test* pode executar inúmeros testes de sensores com o diferencial de possuir um aplicativo de controle que exibe os resultados. Ainda mais, o aplicativo permite a criação de novas suítes de testes a partir da função *"add suite"* o que o torna passível de ser ajustado a novas hipóteses de testes para necessidades futuras.

3.2 Braço Manipulador Robótico Simples

A elaboração do trabalho "*Desenvolvimento de um braço manipulador simples, didático e de baixo custo utilizando Arduino*" (MEGDA; MOREIRA; FASSBINDER, 2012), teve como objetivo desenvolver um braço manipulador de baixo custo que fosse fácil de programar e de ser reproduzido.

É interessante que o braço manipulador (Figura 3.2) é acionado por três servomotores, que são dispositivos constituídos por motores de corrente contínua, com cerca de 180° graus de abertura. Esses servomotores possuem redutores, que são engrenagens que reduzem a velocidade das rotações do motor. Essas engrenagens proporcionam maior desempenho do motor quando é necessário mais torque

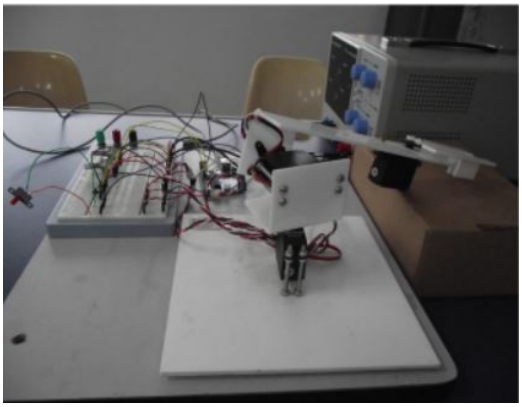

Figura 3.2: Braço Manipulador Robótico. (MEGDA; MOREIRA; FASSBINDER, 2012)

Segundo os autores, após ter sido submetido a vários testes com objetos de diversas formas e pesos de até 80 gramas, foi possível verificar que o manipulador robótico é ágil e preciso. Mostrou-se de fácil construção e pode ser replicado a baixo custo. O projeto elimina obstáculos

para o desenvolvimento de novos protótipos, como a construção de bancadas de teste que simulam linhas de produção industrial.

3.3 Braço robótico para demonstração de uso para as indústrias

O artigo *Braço Robótico Articulado controlado por Arduino para demonstração do Emprego da Robótica nas Indústrias*, elaborador por (SILVA et al., 2017), afirma que desde os momentos iniciais da aplicação dos braços robóticos é possível visualizar significativas melhorias no processo produtivo, as quais podem ser sintetizadas em uma linha automatizada mais eficiente, com redução de custo de mão de obra e mais rápidas.

Nesse artigo utilizou-se o Arduino, porque se apresenta como uma plataforma de desenvolvimento que integra componentes facilmente conectáveis a um computador, alcançando diversas finalidades com baixo custo de implementação. Em seguida, foram integrados quatro micro servomotores de amperagem baixa e um módulo *joystick* movimentar as articulações do braço robótico, produzindo movimento proporcional ao comando recebido, até 180° graus, além de abrir a garra até 55mm.

Por fim, mesmo que o movimento do protótipo desenvolvido ainda não tenha funcionado como planejado, tendo em alguns momentos movimentos abruptos de algumas articulações, para aplicações em manipulação ou carga e descarga de materiais, o protótipo atendeu satisfatoriamente às necessidades de demonstrar uma possível funcionalidade dos braços robóticos para as indústrias.

3.4 Braço robótico microcontrolável

O trabalho “Braço robótico microcontrolável” (LASMAR, 2014), tem como objetivo apresentar um braço robótico para fins didáticos, microcontrolável, com os princípios de funcionamento semelhantes a um manipulador industrial. O autor relata que o braço robótico surgiu para suprir as necessidades das indústrias para substituir o trabalho humano por uma mão de obra automatizada e mais eficiente. Esse robô deveria desempenhar atividades como soldagem,

manipulação de peças, pintura e outras tarefas perigosas ou que requisitassem precisão.

Para o desenvolvimento desse trabalho, o mecanismo, ilustrado pela Figura 3.3, foi idealizado de forma a se assemelhar à estrutura humana, composto por um braço, um antebraço e o pulso, todos conectados por juntas de movimento relativo. Nas juntas são acoplados acionadores que permitem a realização de movimentos individuais a partir das informações enviadas por um sistema de controle. Conectado ao pulso está o órgão terminal, que no caso se trata de uma garra que serve para a manipulação de objetos de tamanhos distintos, variando a partir da destinação de sua aplicação.

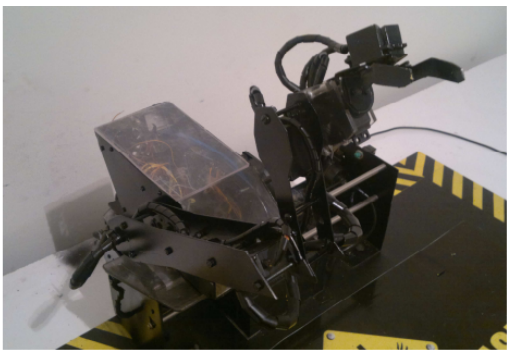

Figura 3.3: Braço Microcontrolável (LASMAR, 2014)

Quanto ao controle do mecanismo, o autor utilizou-se de um Arduino Mega 2560, pois fornece os circuitos adicionais necessários para que o módulo processador possa funcionar. Além disso, no decorrer do processo houveram aprimoramentos considerando as necessidades do projetor, ocasionando a substituição de dois motores, localizados no braço e no antebraço que suportassem a carga de 13kg e 8kg, respectivamente. Não apenas isso, mas os elos e eixos foram substituídos e surgiu a necessidade de colocar um sistema de ventilação, posto que os componentes passaram a apresentar um aquecimento após com o decorrer do tempo de uso que comprometia o seu funcionamento correto.

3.5 Considerações

O autor, BARBOZA (2016), não deixa claro pra quais tipos de testes ele foi desenvolvido, visto que podem variar para necessidades mais específicas como o teste VCVR. Do mesmo analisa-se

que os autores Megda, Moreira e Fassbinder (2012), conseguiram desenvolver um braço com o uso de servomotores com redutores para alcançar um torque maior, sem precisar de um contrapeso, tendo um limite de até 80 gramas. O autor Silva et al. (2017) obteve dificuldades na implementação do código e devido a escolha dos micro servomotores utilizados, estes não suportariam um peso grande.

Considerando que a soma do braço a um aplicativo pode atribuir novas tarefas e métodos e ampliar sua aplicabilidade a diversos sistemas e testes, integrando-o a necessidade de várias empresas, e o uso de *joysticks* para também manipular o braço, se tornaram pontos positivos a se pensar na implementação da automação do VCVR.

Por fim, o trabalho desenvolvido por Lasmar (2014) possui algumas características mais detalhadas que o tornam de grande vantagem para além do meio didático. A quantidade de articulações permite uma variação maior de movimentos, bem como diferentes usos a partir da variação do tipo do seu órgão terminal, que não precisa se limitar a uma garra. A inclusão de um sistema de ventilação também se mostra um diferencial, dado que permite ao mecanismo ser submetido a rotinas maiores de trabalho que não comprometam ou diminuam a eficácia dos testes ou outras finalidades que for aplicado.

Capítulo 4

Projeto de Automação do VCVR

Neste capítulo, foi descrito a implementação de automação do VCVR. Foi apresentado um esboço da automação, uma descrição das etapas da execução da inclinação nos eixos do *smartphone* fixado no braço. E tal como foi desenvolvida esta automação.

4.1 Proposta de Solução

A proposta de automação apresentada neste trabalho é um braço mecânico que tem a função de executar movimentos nos eixos X, Y e Z em relação ao espaço euclidiano, da mesma maneira que o testador de um *smartphone* Android® executaria manualmente conforme os procedimentos de rotina estabelecido para o teste VCVR.

Para executar essas atividades, o braço deverá ser composto de motores de passo para auxiliar nos movimentos angulares e um suporte para manter o *smartphone* firme enquanto faz os movimentos angulares. A base servirá de apoio para manter o braço estável.

Os movimentos realizados pelos motores devem ter uma velocidade constante para que a análise do vídeo feito pelo caso de teste não gere erros por falta de foco nos quadros.

4.2 Descrição de funcionamento

Acompanhando o fluxograma, ilustrado pela Figura 4.1, o Motor 2, inicialmente, inclinará o braço com o *smartphone* do ponto 0° grau no sentido horário, atingindo os 60°, e retornará 120° graus para atingir toda a faixa de inclinação azul do eixo X, identificada na tela do *smartphone* e por fim, posicionará o braço no sentido 0° grau.

Após isso, o Motor 1 deverá girar 90° graus, no sentido horário, posicionando o *smartphone* para entrar na segunda etapa do VCVR no sentido transversal. Simultaneamente, Figura 4.1, ao Motor 1, o Motor 3 rotacionará o suporte em 90° graus, no sentindo anti-horário, para que não haja interferência no sensor quando chegar no último passo do VCVR. Com este movimento o giro do Motor 3 com o suporte é quase imperceptível.

Em seguida na Figura 4.1, o Motor 2 inclinará o braço em 60° graus e retornará 120° graus para atingir toda a faixa de inclinação azul do eixo Y, identificada na tela e posicionará o braço no sentido 0° grau. Por fim, na última etapa do VCVR, no eixo Z, o Motor 3, fará o giro de 360° graus no sentido horário, obedecendo todo o percurso indicado na faixa de rotação azul na tela do *smartphone*. O VCVR emitirá o sinal sonoro de finalização confirmando que todos as etapas de leitura dos eixos X, Y e Z foram concluídos normalmente.

Depois que a automação finaliza seus movimentos, o caso de teste VCVR deverá iniciar a validação do que fora gravado durante a execução. Com isso espera-se que esses movimentos no eixos X, Y e Z sejam os mais precisos, evitando ao máximo re-execuções por imprecisão do testador, por ter sido implementado o processo na automação. O resultado do VCVR será gravado na memória indicando se o sistema foi concluído com êxito ou com falhas em algum dos três eixos.

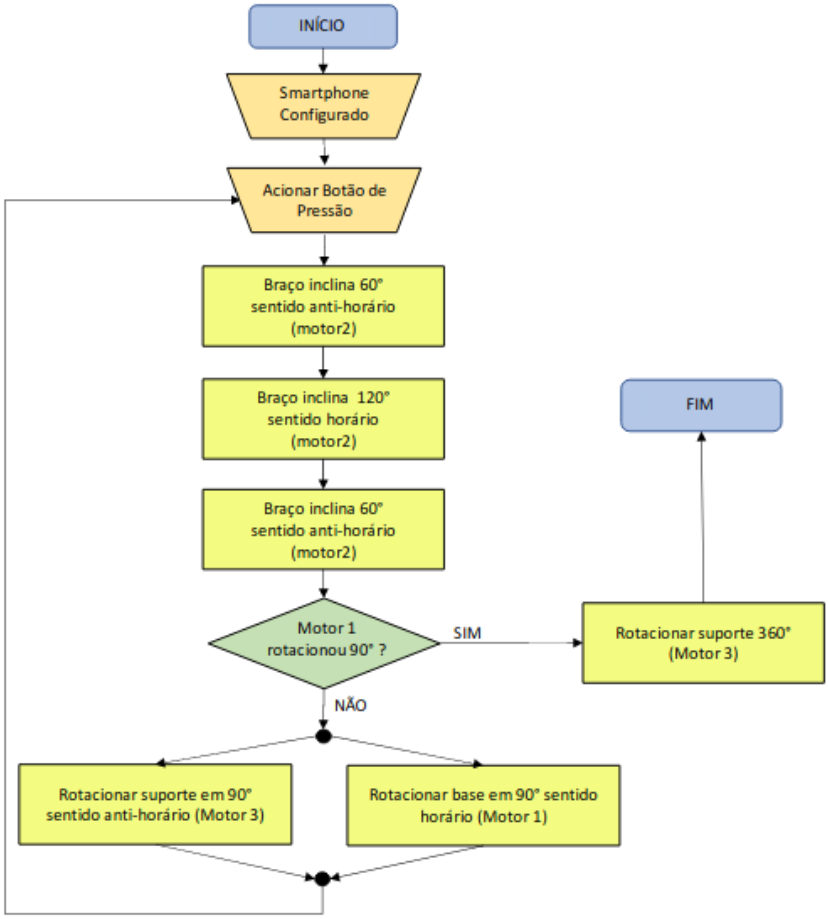

Figura 4.1: Fluxograma da Automação.

4.3 Implementação

Para o funcionamento dos motores fez-se necessário o uso dos *drivers* de motores que utilizam o chip A4988, pois foi desenvolvido para motores de passo bipolares.

4.3.1 Materiais Utilizados

Pode-se observar na Figura 4.2 uma representação das ligações elétricas presentes no protótipo. Neste protótipo, o circuito possui três motores, sendo eles (Tabela 4.1): motor de passo Nema 23, o qual se encontra fixado no braço; motor de passo Nema 17, o qual tem a função de rotacionar o disco em 90° graus, e o motor de fax adaptado que se encontra no topo do braço, onde é fixado no suporte de sustentação para o *smartphone*.

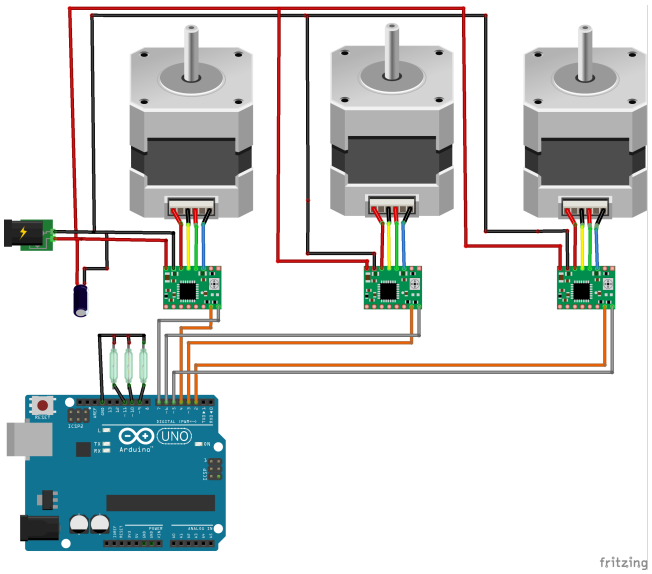

Figura 4.2: Protótipo - Circuito de ligação usando Fritzing.

Tabela 4.1: Lista de Motores

Motor	Nome	Angulação
Motor 1	Nema 17	90° graus
Motor 2	Nema 23	60° a 120° graus
Motor 3	Motor de Fax	360° graus

4.3.2 Estrutura Mecânica

Para a elaboração da automação do braço mecânico, arredondou-se um dos cantos da base para que o braço se desloca-se para outra lateral. Na Figura 4.3, tem-se um esboço da visão geral do braço mecânico, onde vê-se a base, o braço, o contrapeso, os motores, o disco, a cremalheira e o suporte em que fica preso o *smartphone*. Algumas das peças utilizadas para construção desse projeto, foram reutilizados a partir de aparelhos eletrônicos descartados.

Pode-se observar na Figura 4.4 o resultado final do circuito do protótipo. Com essa configuração do circuito, o protótipo conseguiu se aproximar do resultado final esperado, tendo torque o suficiente para realizar os movimentos do VCVR.

Na Figura 4.5a, tem-se o projeto em uma de suas laterais, onde vê-se a base, o braço e o suporte em que fica preso o aparelho. A fonte utilizada (Figura 4.5b) foi adaptada de um *desktop*, alimentado em 127VAC, com ajuste de tensão de saída para 16VDC, mantendo a corrente de trabalho próximo dos 8 Ampérs, bem como sua saída de ar e seu cooler (resfriador).

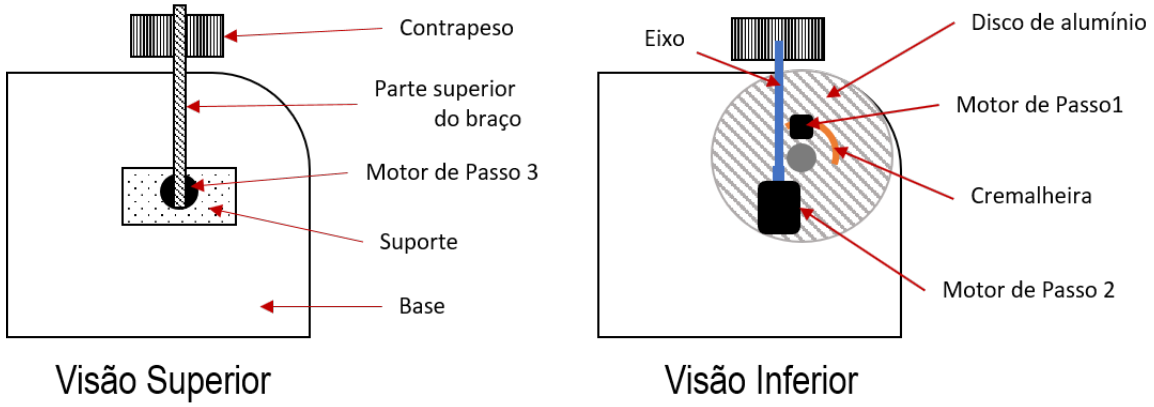

Figura 4.3: Visão Geral da arquitetura da Automação.

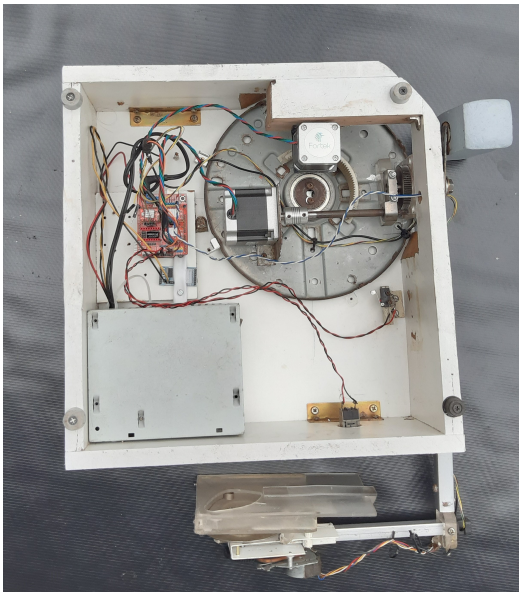

Figura 4.4: Protótipo - Circuito de ligação.

Foi incluído um botão de pressão amarelo (*PushButton*) onde se inicia o funcionamento do Braço Mecânico, levando a configuração e centralização da haste para que seja acoplado o *smartphone* no suporte (Figura 4.5b).

No suporte, feito de acrílico (Figura 4.6) fica acoplado o *smartphone* e acima está o motor de fax adaptado, que por sua vez realiza a rotação no eixo Z em 360° graus. E por não exigir muito esforço, este motor foi escolhido para realizar apenas esta função.

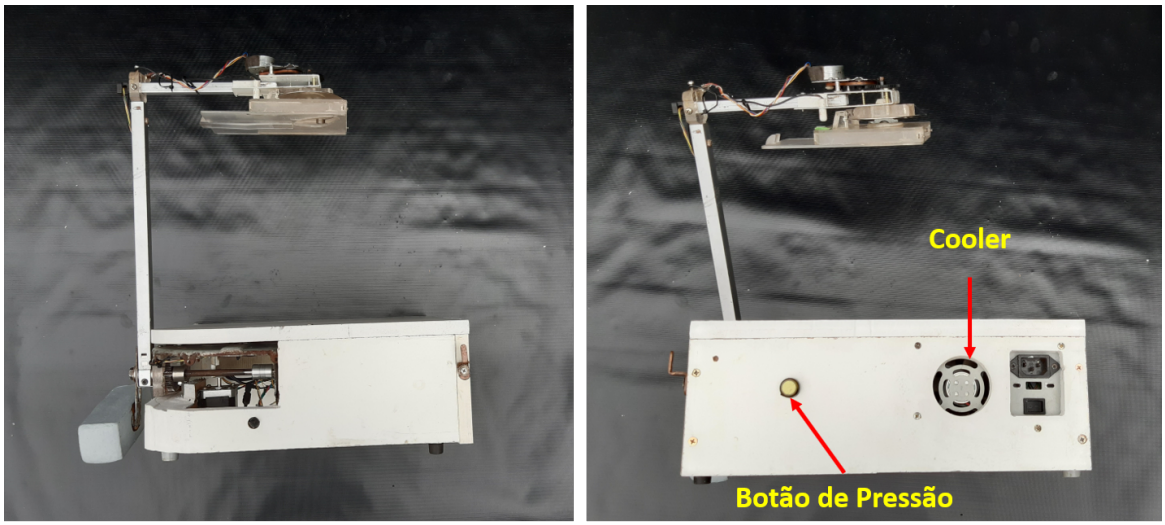

Figura 4.5: Protótipo montado.

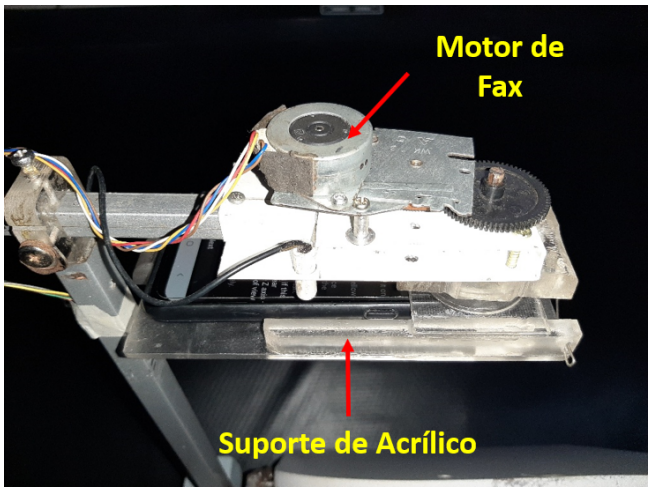

Figura 4.6: Protótipo - Motor do suporte do *smartphone*

4.3.3 Desenvolvimento do Código Fonte

Para implementação do código fonte da automação foi utilizado o ambiente de desenvolvimento *IDE Arduino* 1.8.13 *(The Arduino Integrated Development Environment)* com a linguagem de programação C. Utilizou-se também duas bibliotecas que permitem que as placas Arduino controlem vários de motores de passo:

- ***Stepper.h***: Essa biblioteca permite controlar motores de passo unipolares ou bipolares (ARDUINO, 2018).

- ***AccelStepper.h***: Fornece uma interface orientada a objetos para motores de passo de 2, 3 ou 4 pinos e drivers de motor (ARDUINO, 2018).

Abaixo seguem partes do código implementado no Arduino para a funcionalidade dos motores.

A Figura 4.7 corresponde à parte inicial do código fonte, onde pode-se ver as duas bibliotecas mencionadas previamente nas linhas "*include*". É possível observar também as pinagens da placa Arduino nas linhas 15 a 18. Os pinos 2 e 5 (linha 15) representam o motor 1, citado anteriormente. Assim como os pinos 3 e 6 (linha 16) representam o motor 2 e por fim, o motor 3 é representado com os pinos 4 e 7 (linhas 17 e 18) . Essa configuração foi feita para que fosse possível controlar os motores separadamente.

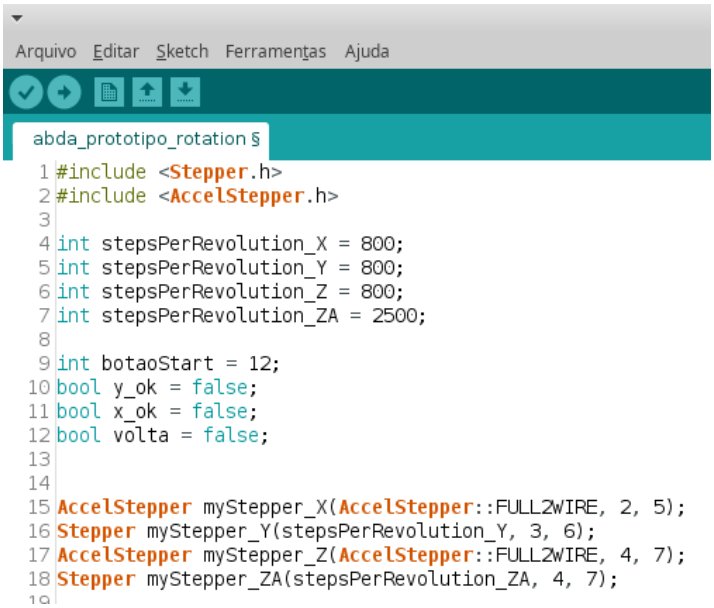

Figura 4.7: Início do código fonte implementado no Arduino.

Na sequência, a Figura 4.8 representa a função de configuração (*setup*) do código em relação ao Arduino, onde define-se para os motores, a velocidade e aceleração em passos por segundo em vez de radianos por segundo.

A Figura 4.9 apresenta a função principal (*loop*), onde há uma condição do estado no botão de pressão (*PushButton*). Se for o botão for pressionado ou estiver "*low*", incia-se a função "motor_Y()", cuja funcionalidade é realizar as inclinações em 60° e 120° graus, descritas na Figura 4.1. Na sequência, no laço de repetição "*while*", o motor 1 e 3 estão implementados na

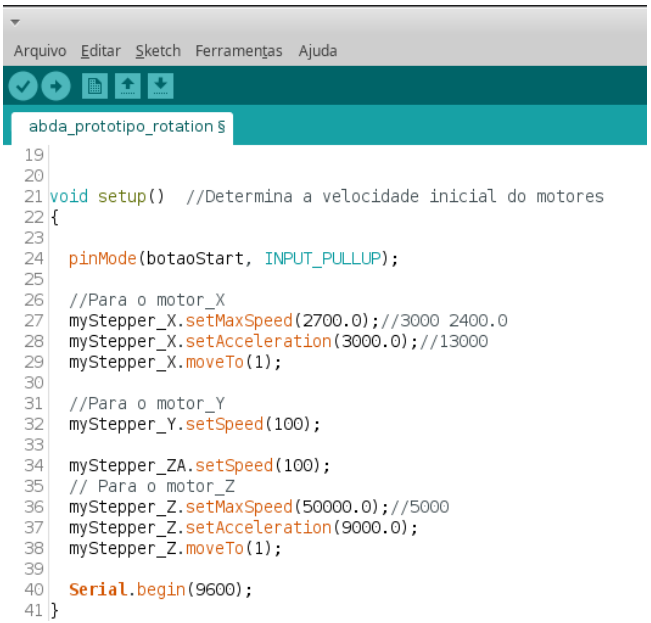

Figura 4.8: Continuação do código fonte implementado no Arduino.

função ”motor_X()”, realizam a rotação em 90° graus, motor 1 no sentido horário e o motor 3 no sentido anti-horário. Em seguida, após o botão ser pressionado novamente, o braço executa a função ”motor_Y()” para as inclinações. Finalizando na função ”motor_ZA()” onde o motor 3 rotaciona em 360° graus.

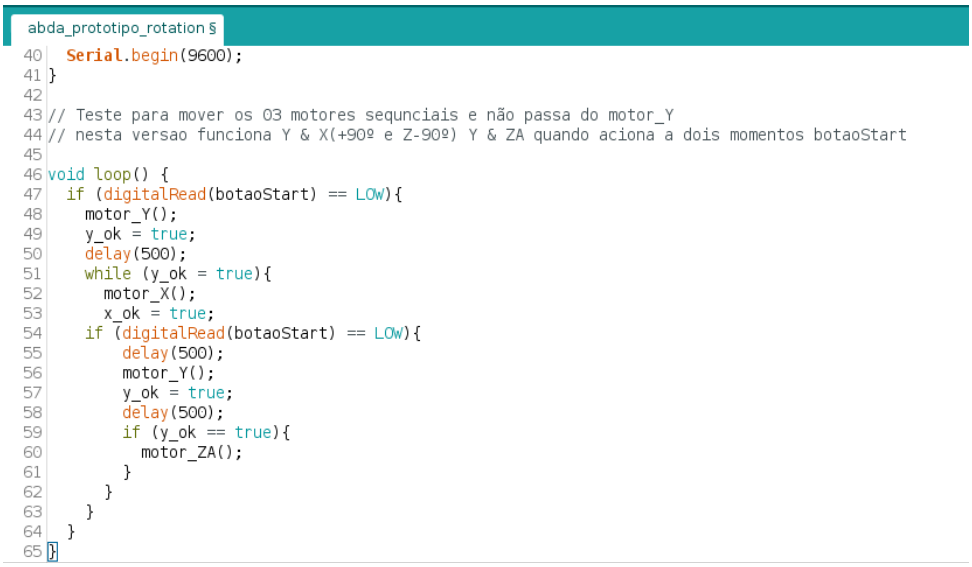

Figura 4.9: Final do código fonte implementado no Arduino.

4.4 Identificando o Momento Máximo

A partir de tudo que fora apresentado até o momento no presente trabalho, passa-se a apresentar os cálculos realizados para aplicar a automação, bem como as fórmulas utilizadas. Através da aplicação matemática é possível projetar o que o protótipo fará, inclusive por meio de gráficos que apontam os momentos e as variações do braço, esteja ele inclinado para a direta, esquerda ou centralizado.

Os cálculos fundamentais, do momento máximo é representado pelas Figuras 4.10 e 4.11.

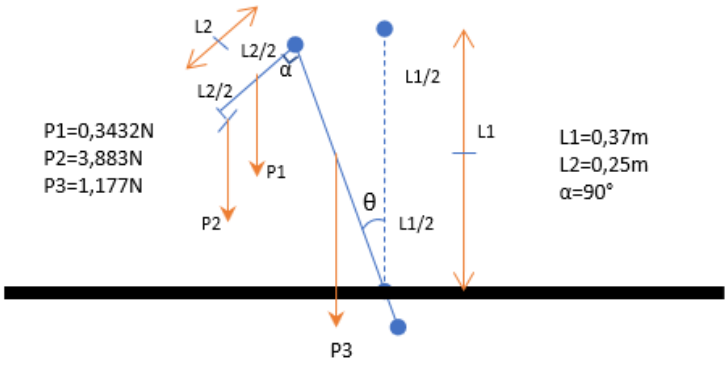

Figura 4.10: Cálculo do Momento de Torque.

Aplicando a fórmula do torque (Equação 2.1), encontra-se as equações das distâncias d1, d2, e d3 respectivamente:

$$d1 = \frac{L1}{cos\Theta}; \tag{4.1}$$

$$d2 = \frac{L1}{cos\Theta} + \frac{L2}{2} * cos(\frac{\pi}{2} - \Theta); \tag{4.2}$$

$$d3 = \frac{L1}{cos\Theta} + L2 * cos(\frac{\pi}{2} - \Theta); \tag{4.3}$$

Chegamos na equação do Torque do protótipo.

$$f(x) = 1.22 * cos(x) + 0.38 * cos(\frac{\pi}{2} - x), (-\frac{\pi}{3} < x < \frac{\pi}{3}) \tag{4.4}$$

Ressalta-se que os cálculos apresentados se assemelham aos exemplos presentes em (MERIAM; KRAIGE; BOLTON, 2020) .

Pormenorizando, foram realizados cálculos para cada parte do movimento, sendo o primeiro partindo de seu ponto de equilíbrio e inclinando-se para o eixo X, onde o braço retorna ao ponto zero, e depois onde se inclina para a direita. Com isso é possível combinar todos os cálculos e valores obtidos dos movimentos para encontrar o momento máximo e uma fórmula capaz de gerar um gráfico, Figura 4.11, onde se visualiza o momento máximo A.

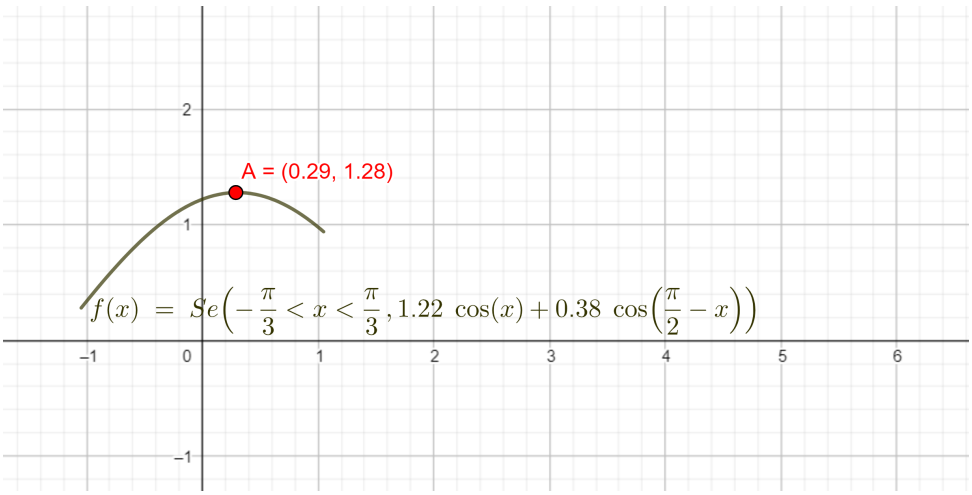

Figura 4.11: Gráfico do Momento máximo por *GeoGebra Classic*.

4.5 Considerações

A Automação VCVR pode ser considerada multidisciplinar, pois são necessárias múltiplas habilidades de diversas áreas de conhecimento, como, por exemplo automação (Engenharia Elétrica, Eletrônica, Controle e Automação e Engenharia Mecânica) e Computação.

Um problema foi identificado após a construção do protótipo. Esse problema está relacionado ao esforço despendido para que o braço levantasse o *smartphone* para o momento inicial. Verificou-se que nesse momento era exigido um esforço grande do motor, acarretando um aumento na energia necessária para que o braço retornasse ao ponto zero. Para contornar tal dificuldade, foi instalado um contrapeso ao final da haste do braço.

Capítulo 5

Experimentos e Resultados

Neste capítulo aborda os experimentos referentes aos testes realizados na automação do VCVR, com intuito de apresentar os resultados obtidos pela mesma.

5.1 Resultados

O protótipo desenvolvido realiza as inclinações nos eixos X e Y dentro dos limites da faixa, atendendo o que fora proposto pelo VCVR. Observa-se na Figura 5.1 a faixa de Inclinação em azul na tela do *smartphone* assim como no eixo X e no eixo Z (Figura 5.2) o protótipo atende a faixa de rotação.

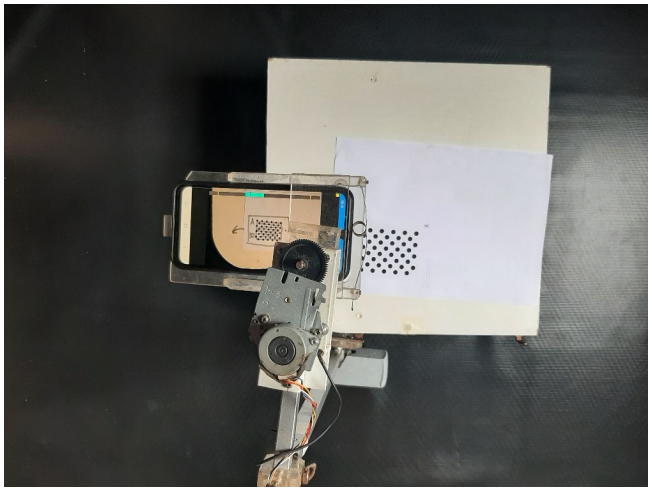

Figura 5.1: Protótipo com o Papel de Imagem no eixo Y.

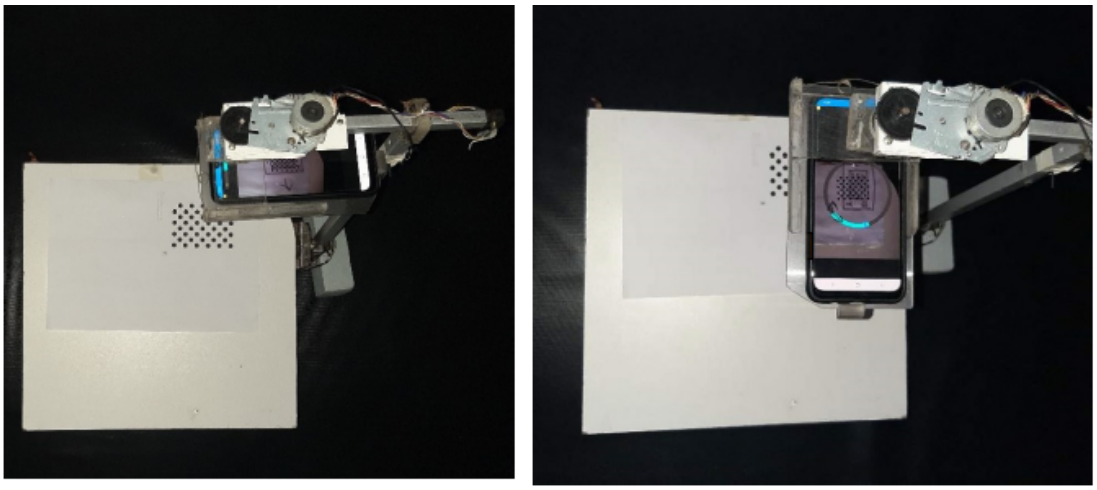

Figura 5.2: Protótipo com o Papel de Imagem no eixo X e no eixo Z.

Durante os experimentos do braço mecânico, o motor de passo Nema 23 obteve sucesso em sustentar o peso do braço e do *smartphone* acoplado no suporte durante a inclinaçao, com o auxílio do contra peso. E isso auxiliou para que o braço não avançasse além da inclinação necessária por conta do peso do suporte com o *smartphone* causando desgaste no eixo do motor. A execução dos motores em suas programações atenderam conforme desenvolvido.

Através de experimentos, pode-se concluir que os resultados finais foram bem-sucedidos, pois a automação atendeu aos requisitos do VCVR. A Tabela 5.1 apresenta as comparações do tempo e resultado de execução entre a automação e um testador humano.

Tabela 5.1: Tempo e resultado da execução na primeira tentativa

Nº	**Automação**	**Resultado**	**Testador**	**Resultado**
1	53”04	✓	45”05	×
2	52”17	✓	24”99	×
3	51”84	×	20”35	×
4	50”90	✓	29”09	×
5	50”22	✓	21”07	✓
6	55”07	✓	24”79	✓
7	50”03	✓	1’06”00	✓
8	52”07	×	1’15”00	×

Pode-se observar que, do total de oito execuções de testes automatizados, seis cumpriram com precisão os requisitos estabelecidos do teste do VCVR, alcançando 75% de acertos. Os dois outros casos (teste 3 e 8 da Tabela 5.1) falharam por leves erros no encaixe do *smartphone*

no suporte, sendo necessária uma nova execução representando os 25% de falhas.

O Tempo de execução na automação ainda poder ser reduzido no código fonte, aumentando o número de rotações por minuto dos motores. Mas pode-se observar também que o número de acertos pela automação é maior que a do testador.

Capítulo 6

Conclusão

Atendendo aos objetivos propostos neste trabalho, o tempo do braço mecânico conseguiu se manter padronizado em relação aos testes realizados manualmente, com uma variação mínima de tempo para cada tentativa.

Considerando atender o objetivo de precisão na leitura do VCVR, o projeto braço robótico atendeu aos requisitos definidos, pois na tela do *smartphone* ficou visível que o braço mantinha uma estabilidade durante a leitura do Papel de Imagem. Comparando o processo manual efetuado pelo testador, observa-se que a causa da instabilidade durante a execução do VCVR, é um dos motivos que requer a re-execução do teste.

Por fim, o objetivo de reduzir falhas operacionais foi alcançado por meio de resultados satisfatórios, tendo visto que ao executar os testes VCVR na automação, a incidência de falhas foi reduzida frente ao testador humano.

6.1 Trabalhos Futuros

Para trabalhos futuros propõe-se um acionamento remoto via rede, utilizando módulos de Wi-Fi no Arduino para que, à distância, o testador possa acompanhar o progresso da execução. O Arduino também poderá capturar os ângulos obtidos pelo braço e mostrar uma barra de progresso via Web, talvez utilizando uma plataforma para uso de Internet das Coisas, para exibir quais passos o braço executou e se houve algum obstáculo.

Referências Bibliográficas

ARDUINO. *Arduino Hardware*. 2018. Url = <https://https://www.arduino.cc/en/hardware>. "[Acessado em Maio/2022]".

BARBOZA, L. d. A. *AR (m) obo Test: um braço robótico para suporte à testes automáticos de retrato e paisagem para smartphones*. Dissertação (Mestrado) — Universidade Federal de Pernambuco, 2016.

BERNARDO, P. C.; KON, F. A importância dos testes automatizados. *Engenharia de Software Magazine*, v. 1, n. 3, 2008.

COBEL, R. *Teorias Rotativas 05*. Url = <http://canalpiloto.com.br/teorias-rotativas-05/>. "[Acessado em Maio/2022]".

COELHO Ítalo. *Como usar Motor de Passo com Driver A4988 e Arduino*. 2021. Url = <https://www.filipeflop.com/blog/como-usar-motor-de-passo-com-driver-a4988/>. "[Acessado em Maio/2022]".

DEVELOPERS, A. *Tipos de Sensores*. 2020. Url = <https://source.android.com/devices/sensors/sensor-types>. "[Acessado em Novembro/2021]".

DEVELOPERS, A. *Verificação Cruzada do Vetor de Rotação*. 2020. Url = <https://source.android.com/compatibility/cts/rotation-vector>. "[Acessado em Novembro/2021]".

FAA, F. A. A. *Aviation Maintenance Technician Handbook — Airframe (FAA-H-8083-31A, Vol.1)*. Oklahoma City, OK, United States: U.S. Department of Transportation, 2018. Disponível em: <https://www.faa.gov/sites/faa.gov/files/regulations_policies/handbooks_manuals/aviation/amt_airframe_hb_vol_1.pdf>.

GUILHERME, W. D. Contradições e desafios na educação brasileira. 2019.

HALLER, K. Mobile testing. *ACM SIGSOFT Software Engineering Notes*, ACM New York, NY, USA, v. 38, n. 6, p. 1–8, 2013.

HAMMACK, B. Eight amazing engineering stories. *Lexington: Articulate Noise Books*, 2012.

HIBBELER, R. *Estática e Dinâmica. Mecânica para Engenharia*. 14. ed. [S.l.]: Pearson Prentice Hall, 2017.

KNOTT, D. *Hands-on Mobile App Testing: A Guide for Mobile Testers and Anyone Involved in the Mobile App Business*. [S.l.]: Addison-Wesley, 2015.

LASMAR, A. W. R. Braço robótico microcontrolável. 2014.

MCROBERTS, M. *Arduino Básico*. [S.l.]: Novatec Editora, 2018.

MEGDA, O. A.; MOREIRA, H. R.; FASSBINDER, A. G. de O. Desenvolvimento de um braço manipulador robótico simples, didático e de baixo custo utilizando arduino. 2012.

MERIAM, J. L.; KRAIGE, L. G.; BOLTON, J. N. *Engineering mechanics: dynamics*. [S.l.]: John Wiley & Sons, 2020.

MICROSYSTEMS, A. *DMOS Microstepping Driver with Translator And Overcurrent Protection*. 2009. Url = <https://www.pololu.com/file/0J450/a4988_DMOS_microstepping_driver_with_translator.pdf>. "[Acessado em Maio/2022]".

MOTION, N. *Datasheet de Produto – Motores de Passo*. 2017. Url = <https://www.neomotion.com.br/wp-content/uploads/2017/07/CatÃąlogo-Datasheet-dos-motores-de-passo-R01.pdf>. "[Acessado em Maio/2022]".

NAGOWAH, L.; SOWAMBER, G. A novel approach of automation testing on mobile devices. In: IEEE. *2012 international conference on computer & information science (ICCIS)*. [S.l.], 2012. v. 2.

NIELD, D. *Conheça todos os sensores do seu smartphone e como eles funcionam*. 2017. Url = <https://gizmodo.uol.com.br/sensores-smartphones-guia/>.

PRESSMAN, R. S. *Software Engineering: a Practitioner's Approach*. 10. ed. [S.l.]: Raghu Srinivasan, 2015.

SANTOS, V. P. de A. Motor de passo. Niterói, RJ, Brasil, 2008. "[Acessado em Maio/2022]".

SCARBOROUGH, J. B. *The Gyroscope*. [S.l.]: Interscience Publ., 1958.

SILVA, C. C. et al. Braço robótico articulado controlado por arduíno para demonstração do emprego da robótica nas indústrias. In: *VI JORNACITEC-Jornada Científica e Tecnológica*. [S.l.: s.n.], 2017.

SOMMERVILLE, I. *Software Engineering*. 10. ed. Harlow, England: Addison-Wesley, 2016.

TUBES, A. *Tipos de motores de passo e modos de operação*. 2004. Url = <https://www.altanatubes.com.br/webstore/?c=282&t=Tipos-de-motores-de-passo-e-modos-de-operacao>. "[Acessado em Maio/2022]".