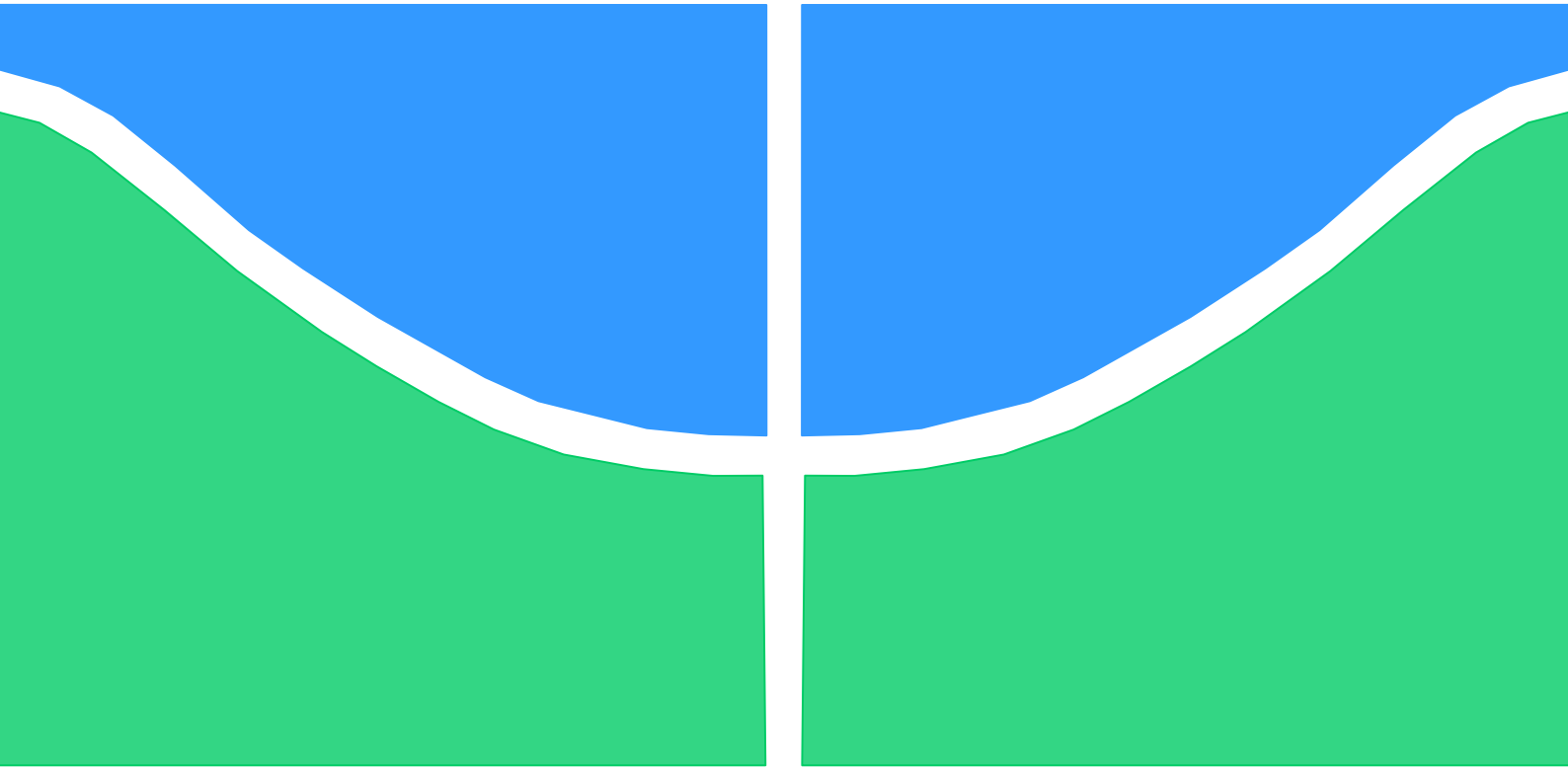

TRABALHO DE GRADUAÇÃO

SISTEMA DE DETECÇÃO E ALERTA DE QUEDAS PARA IDOSOS

João Paulo Botelho Guimarães Vieira
Matheus Henrique Mendonça Araújo

Brasília, julho de 2019

UNIVERSIDADE DE BRASILIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

TRABALHO DE GRADUAÇÃO

SISTEMA DE DETECÇÃO E ALERTA DE QUEDAS PARA IDOSOS

João Paulo Botelho Guimarães Vieira
Matheus Henrique Mendonça Araújo

Relatório submetido como requisito parcial para obtenção
do grau de Engenheiro de Redes de Comunicação

Banca Examinadora

Prof. Cláudia Barenco Abbas, UnB/ ENE (Orientador) _____

Prof. Daniel Chaves Café, UnB/ ENE _____

Prof. Ricardo Zelenovsky, UnB/ ENE _____

Dedicatórias

Dedico este trabalho a meus pais, que durante toda minha vida, do maternal ao fim desta graduação, sempre se esforçaram ao máximo para que eu tivesse a melhor formação que eles fossem capazes de me proporcionar.

João Paulo Botelho Guimarães Vieira

Dedico este trabalho ao meu filho, que me deu mais forças para continuar toda a trajetória acadêmica. Dedico também aos meus pais, por terem me apoiado ao longo da vida em todos os momentos que precisei.

Matheus Henrique Mendonça Araújo

Agradecimentos

Primeiramente gostaria de agradecer a Deus, a quem recorri diversas vezes durante minha caminhada nesta universidade, sem Ele nada seria possível. Agradeço a minha família, que é o pilar da minha vida, mas de uma maneira especial a minha mãe, que viveu comigo cada segundo de falhas e conquistas durante meu processo de graduação. Um agradecimento mais que especial a minha orientadora, Profª. Drª. Cláudia Jacy Barenco Abbas, que esteve sempre muito presente durante todas as etapas do TCC, sempre com muita atenção e disposição e uma simpatia ímpar. Agradeço a todos os excelentíssimos professores pelos quais tive o privilégio de ser aluno, vocês nunca serão esquecidos. Ao meu companheiro nesta longa e trabalhosa caminhada que foi a realização deste trabalho, meu colega e amigo Mateus Henrique, obrigado por topar se aventurar nessa comigo. A todos meus queridos e amados amigos, vocês aliviaram a barra que é passar por este curso, estaremos sempre juntos. Aos meus padrinhos Cláudia e Rinaldo e a minha amada avó Zelinda, agradeço do fundo do coração por ter me ajudado durante todo meu curso, foi muito importante para minha vida social e minhas cervejas do final de semana, que me fazia desestressar das centenas de semanas corridas. Por último, meus sinceros agradecimentos a Universidade de Brasília, onde me estressei várias vezes, mas me fez crescer como cidadão e futuro profissional, hoje posso dizer que saio outra pessoa desta instituição.

João Paulo Botelho Guimarães Vieira

Agradeço aos meus irmãos, João e Carol, por todo o apoio que me deram ao longo dessa trajetória de vida e muitos momentos memoráveis ao longo dela. Agradeço ao meu pai, João Bosco, por sempre se preocupar e buscar o melhor para os seus filhos. Agradeço a minha mãe, a senhora Adelina, por ser a melhor pessoa que o mundo já viu, uma mulher incrível e com o coração gigante. Agradeço a minha melhor amiga e namorada, Luiza, por ter me ajudado a continuar e me colocar para cima quando nem tudo parecia estar dando certo. Agradeço aos meus avós, Araujo e Neusa, por todo o carinho dado aos netos (e que saudade!). Agradeço ao meu filho, João Henrique, minha maior motivação para sempre buscar crescer. Agradeço os meus amigos, Lucas Marques, Lucas Crelier, Roberto Tramm, Gustavo Xudré e Pedro Holtz, a amizade de vocês foi muito importante ao longo da minha trajetória acadêmica. Agradeço ao meu companheiro de TCC, João Paulo, uma pessoa dedicada que se esforçou a todo momento por este projeto. Agradeço a minha orientadora, Professora Cláudia Jacy Barenco Abbas, por nos guiar nessa trajetória final e ter sido uma excelente orientadora ao longo desse projeto. Agradeço a Universidade de Brasília, por mais que a aventura de se graduar aqui tenha sido difícil, a UnB me tornou melhor, me fez amadurecer e me ensinou muito.

Matheus Henrique Mendonça Araújo

RESUMO

Este trabalho apresenta o desenvolvimento de um dispositivo de detecção de queda cujo público alvo são pessoas idosas. Para isso, utiliza-se um sensor acelerômetro que determina a aceleração e a inclinação em relação a gravidade do objeto acoplado ao sensor, que neste caso é um ser humano. Os dados gerados pelo acelerômetro são processados por um algoritmo de 4 etapas que determinará se ocorreu ou não uma queda. Caso tenha ocorrido uma queda, o dispositivo encaminhará a um e-mail pré-cadastrado por um aplicativo Android também desenvolvido neste trabalho, uma mensagem com informações sobre a queda: nome da vítima, o horário, e a posição estimada da queda.

ABSTRACT

This paper presents the development of a fall detection device whose target is the elderly. For this, an accelerometer sensor is used to determine the acceleration and inclination with respect to the gravity of the object coupled to the sensor, which in this case is a human being. The data generated by the accelerometer is processed by a 4-step algorithm that will determine whether or not a fall occurred. If a fall has been detected, the device will send to a pre-registered e-mail by an Android application also developed in this work, a message with information about the fall, such as the victim's name, the time, and the estimated position of the fall.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

Símbolos Latinos

G	Aceleração da gravidade	[m/s^2]
V	Tensão	[volt]
mAh	Capacidade de baterias	[mAh]
Hz	Frequência	[Hz]

Símbolos Gregos

Ω	Resistencia elétrica	[v/i]

Siglas

AP	Access Point
BLE	Bluetooth Low Energy
CAN	Controller Area Network
EEPROM	Electrically-Erasable Programmable Read-Only Memory
GMT	Greenwich Mean Time
GND	Ground
HTTP	Hypertext Transfer Protocol
ICT	Information and Communications Technology
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronic Engineers
IoC	Internet of Computers
IoP	Internet of People
IoT	Internet of Things
ISM	Industrial, Scientific and Medical
I2C	Inter-Integrated Circuit
LED	Light Emitting Diode
Li-Ion	Lithium-Ion
MCS	Modulation and Coding Scheme
MEMS	Micro-Electro-Mechanical Systems
MIT	Massachusetts Institute of Technology
PHP	Hypertext Preprocessor
RF	Radiofrequência
RSSF	Rede de Sensores Sem Fio
SCL	Serial Clock
SDA	Serial Data
SDIO	Secure Digital Input Output
SDO	Seria Data Output
SIG	Special Interest Group
SMTP	Simple Mail Transfer Protocol
SPI	Serial Peripheral Interface
STL	Stereolithography
TLS	Transport Layer Security
UART	Universal Asynchrounous Receiver/Transmiter
ULP	Ultra Low Power
URL	Uniform Resource Locator
USB	Universal Serial Bus
VCC	Common Collector Voltage
WHO	World Health Organization

LISTA DE EQUAÇÕES

$$|a| = \sqrt{a * a} = \sqrt{a_i^2 + \cdots + a_n^2} \quad (1)$$

$$Norm_{xyz} = |g| = \sqrt{x^2 + y^2 + z^2} \quad (2)$$

Sensitividade = $\frac{TP}{TP + FN}$ (3)

Especificidade = $\frac{TN}{FP + TN}$ (4)

Precisão = $\frac{TP}{TP + FP}$ (5)

Acurácia = $\frac{TP + TN}{TP + FP + TN + FN}$ (6)

1 INTRODUÇÃO

Com cerca de 646.000 acidentes fatais por ano, as quedas são um grande problema de saúde pública ao redor do globo, ocupando a segunda posição quando se trata de mortes acidentais. Além disso, as quedas possuem difícil prevenção: pessoas mais velhas tendem a perder um pouco o controle de seus movimentos (WORLD HEALTH ORGANIZATION, 2018).

Segundo a *World Health Organization* (2018), idosos com mais de 65 anos possuem mais chances de serem as principais vítimas fatais desses acidentes, enquanto a taxa de morte devido à queda é maior entre os idosos com mais de 60 anos.

A detecção de quedas pode ser significante para o tratamento e recuperação da vítima: o tempo para se prestar socorro é um fator importante para diminuir as consequências causadas por uma queda. Além disso, alguns fatores podem ser observados para se determinar a gravidade do acidente, como por exemplo, o tempo em que a pessoa permanece deitada no chão após a queda (XU; ZHOU; ZHU, 2018).

Uma definição aceita para queda é: uma pessoa descendo em direção ao chão ou algum outro terreno inferior, de maneira involuntária (WORLD HEALTH ORGANIZATION, 2018). O problema dessa definição é que uma queda pode ser confundida com outras ações semelhantes, como o ato de deitar-se, por exemplo (XU; ZHOU; ZHU, 2018).

Deve-se observar também que por ser um acidente e possuir alto risco para a vítima, é muito difícil que os pesquisadores consigam dados reais para adaptar o sistema de detecção de queda. Soma-se o fato de que: muitos sensores são utilizados para se detectar uma queda, não existindo uma solução universal para o problema.

O crescente avanço na tecnologia dos sensores de queda livre foi um dos principais motivos para os progressos desenvolvidos na área de detecção de quedas. Segundo Xu, Zhou e Zhu (2018) os sensores são divididos em três principais categorias: baseados em visão, baseados no uso de acelerômetro e baseado em radiofrequência (RF). A primeira categoria abriga os dispositivos que utilizam: câmeras, câmeras com sensor de profundidade e o *Kinect*. A segunda categoria inclui os dispositivos que se utilizam de acelerômetros e os acelerômetros presentes nos celulares. Por fim, última categoria inclui dispositivos com tecnologia baseadas em Wi-Fi e Radar.

Após 2014 é possível observar que 80% dos dispositivos passaram a utilizar *Kinect*, acelerômetros ou ambos, enquanto os dispositivos que utilizam câmera (5%) ou RF (15%) passaram a ser menos utilizados. Antes de 2014, os sensores que utilizam câmeras ocupavam 36% do total, mas alguns problemas como: necessidade de um algoritmo mais robusto, ser afetado pelas condições de luz, e não conseguir reconhecer profundidade dificultavam o uso desses sensores. Além disso, o surgimento do *Kinect*, que utiliza um sensor infravermelho, capaz de prover captura dos movimentos de todo o corpo, ser menos afetado pelas condições de luz e possui sensor de profundidade, foi outro fator que reduziu o uso de câmeras nos sensores de detecção de quedas. Com a terceira geração de *smartphones* os celulares também puderam ser usados como sensores de detecção, com a vantagem de que o celular também pode ser utilizado para alertar de que a queda ocorreu (XU; ZHOU; ZHU, 2018).

Os sensores que utilizam Wi-Fi ou Radar analisam as quedas por um novo ângulo: eles tentam identificar mudanças no ambiente a partir do comportamento humano. Permitindo assim que nenhum dispositivo precise ser utilizado (XU; ZHOU; ZHU, 2018).

Um sistema de detecção de quedas eficiente é aquele que possui boa precisão. Segundo Xu, Zhou e Zhu (2018), observa-se que a maior parte dos algoritmos de detecção de quedas possuem precisão acima de 90% de acertos. Mesmo assim, não é possível encontrar um sistema de detecção de quedas que tenha feito sucesso no mercado.

O primeiro fator que provoca essa falta de sucesso dos sistemas de detecção de queda no mercado é o fato de que não existem muitas bases de dados que levem em consideração quedas reais, sendo a maioria construída com quedas simuladas. Assim, por existirem muitos tipos de ações humanas diferentes é difícil determinar o que é ou não queda, podendo ocorrer falsos positivos. O que torna muitos dos algoritmos insatisfatórios quando se trata de uma implementação prática.

O segundo fator que dificulta é que os algoritmos de detecção podem ter dificuldade de definir o que é ou não uma queda, de acordo com o método de detecção utilizado. Por exemplo, algoritmos baseados em *threshold* utilizam a velocidade da queda para diferenciar de outras ações. Porém, esse método pode detectar um falso positivo para outras ações, como pular (XU; ZHOU; ZHU, 2018).

1.1 Descrição e solução do problema

Sabe-se que a possibilidade de queda é um grande problema na vida da população idosa, principalmente se essa pessoa mora sozinha ou fica sozinha em casa a maior parte do tempo. Muitas vezes, quando há uma queda, o idoso não é capaz de se levantar sem ajuda, devido à falta de força, portanto, esperar essa ajuda por muito tempo pode ter consequências fatais.

O socorro a essas vítimas o quanto antes é imprescindível. Pensando nisso, a solução proposta neste trabalho é a criação de um sistema de detecção e alerta de queda, com a intenção de informar ao responsável pelo idoso que ocorreu uma queda e que provavelmente uma ajuda rápida é necessária.

Primeiramente foi desenvolvido um dispositivo capaz de detectar quedas, que deve ser fixado a um cinto confortável, porém fixo e centralizado na cintura. Em seguida, desenvolveu-se um aplicativo para celulares Android que serve como instrumento de configurações do dispositivo detector de queda por meio de uma conexão *Bluetooth Low Energy* 4.2. Através deste aplicativo, configura-se a rede Wi-Fi à qual o dispositivo deve se conectar, o nome do paciente que utilizará o detector e o endereço de e-mail do responsável por esse paciente. Toda vez que uma queda for detectada, o dispositivo enviará uma mensagem para o e-mail cadastrado, informando o nome da vítima, o horário da queda e a posição estimada da queda.

1.2 Objetivo

Elaborar uma solução completa de um sistema de detecção e alerta de queda, com o desenvolvimento de um circuito, de um algoritmo de detecção, de um aplicativo de configurações necessárias para o dispositivo criado, de um recipiente para alocar os componentes utilizados e um mecanismo de envio de e-mail para alerta através de uma rede Wi-Fi.

1.3 Trabalhos Relacionados

Em Xu, Zhou e Zhu (2019) foram compiladas as mais relevantes soluções de detecção de quedas, que trazem diversos dispositivos e algoritmos sendo utilizados para detecção, entre eles: Mastorakis e Makris (2014) que utiliza o *Microsoft Kinect* e o algoritmo *Decision Tree (*DT*)*, Wang, Wu e Ni (2018) utilizando-se do Wi-Fi e do *Random Forest Algorithm (RFA)* e Kwolek e Kepski (2014) que utiliza o *Kinect* e acelerômetro para detectar a queda.

Em Wang et al. (2014) foi desenvolvido um sistema detector de quedas baseado em *threshold.* A abordagem é interessante por tratar de diferentes níveis de gravidade da queda, baseando-se em três *thresholds* diferentes. O sistema monitora a frequência cardíaca, a aceleração de queda e o ângulo do tronco após o impacto. Níveis de urgência diferentes acionam alvos diferentes para cuidar dos primeiros-socorros: o sistema decide se enviará uma mensagem para os parentes, cuidadores ou até mesmo para ambulância (quando os três valores de *threshold* forem maiores que o normal). O diagrama de blocos desta abordagem pode ser visto na Figura 1.1. Este algoritmo alcançou precisão de 97.5%. Segundo

Wang et al. (2014) é um dos melhores para algoritmos baseados em *threshold*, e um dos melhores entre todos os algoritmos.

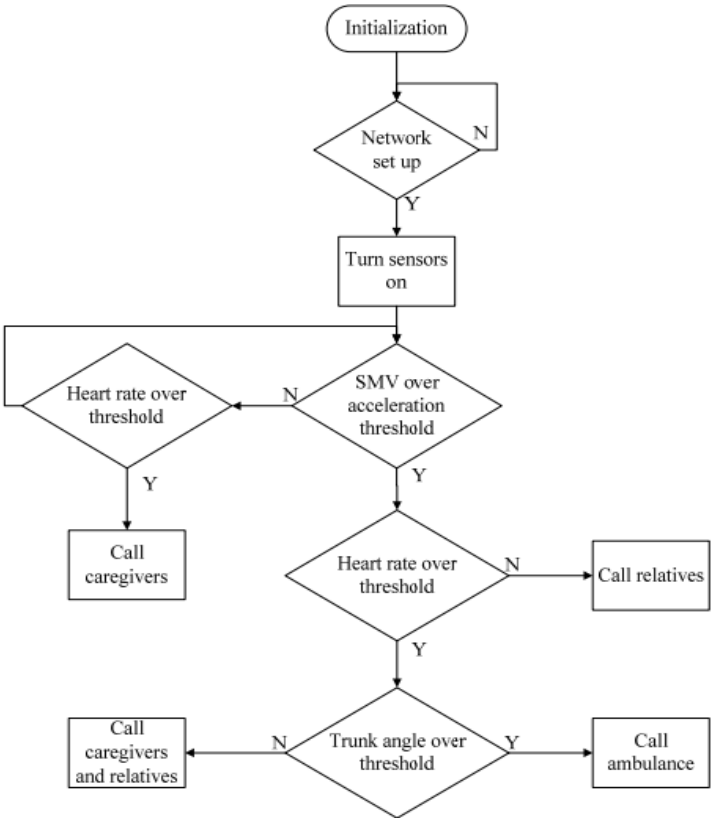

Figura 1.1. Algoritmo utilizado em Wang et al. (2014).

Em Kau e Chen (2014) é utilizado acelerômetro, porém, ao invés de construir um dispositivo específico para isso, os autores se aproveitem dos *smartphones*, item cada vez mais comum para todos. O artigo também propõe um sistema de alerta e primeiros-socorros em caso de ocorrência de quedas. O algoritmo se baseia na aceleração do celular e no ângulo de inclinação do celular, monitorando os estados anteriores e verificando se ocorreu alguma mudança em decorrência de uma possível queda. O sistema desenvolvido e o algoritmo de *SVM (Support Vector Machine)* são complexos e a precisão de detecção foi de apenas 92%.

A abordagem de Wu et al. (2014), apesar de não estar citada em Xu, Zhou e Zhu (2019), faz uso do mesmo dispositivo sensor que será utilizado nesse projeto, o ADXL345, que é capaz de obter parâmetros como ângulos e acelerações nos três eixos (x, y e z), e será melhor apresentado mais adiante. Nessa abordagem, a aceleração de cada eixo é obtida pelo sensor, calcula-se então o módulo da aceleração total do sistema, idêntico ao *threshold* de aceleração em Wang et al. (2014). Além disso, com o uso de quatérnios, um ângulo θ é calculado. Com base nesse ângulo, o algoritmo determina se ação foi confirmada como queda ou como ação normal, como deitar-se. A Figura 1.2 retrata a máquina de estados do algoritmo.

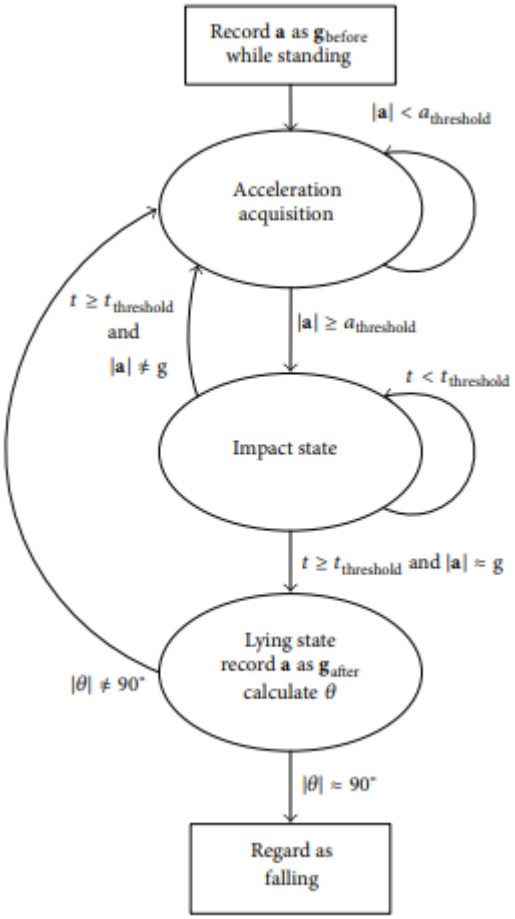

Figura 1.2. Máquina de estados do algoritmo de Wu et al. (2014).

Em Xu, Zhou e Zhu (2019) é possível ver que, dos algoritmos de detecção de quedas citados por ele, aqueles que utilizam somente acelerômetros, tiveram excelentes resultados com o emprego de algoritmos baseados em *threshold*. Em geral, os algoritmos citados possuem precisão acima de 94%, o que pode ser considerado uma boa precisão. A utilização de algoritmo baseado em *threshold* nesse projeto, se dá pelo fato dele ser menos complexo e ainda assim conseguir excelentes resultados. A tabela 1.1 realiza uma comparação entre alguns dos algoritmos citados até o momento.

Tabela 1.1. Comparativo entre os algoritmos de detecção de queda.

Referência	**Sensor**	**Algoritmo**	**Precisão**
Wang et al. (2014)	Acelerômetro	*Threshold*	97.5%
Wu et al. (2014)	Acelerômetro	*Threshold*	97.1%
Kau e Chen (2014)	Acelerômetro	DWT (*Discrete Wavelet Transformer*)	92%
Kwolek e Kepski (2014)	Acelerômetro e Kinect	SVM (*Support Vector Machine*)	98.33%
Wang, Wu e Ni (2018)	Wi-Fi	RFA (*Random Forest Algorithm*)	94%

As soluções mostradas na Tabela 1.1 foram citadas por Xu, Zhou e Zhu (2019), com exceção da solução de Wu et al. (2014). Como as soluções que utilizam o *Kinect* possuem um preço de mercado

elevado, o sensor de quedas desenvolvido em cima deste equipamento seria de alto custo, quando comparado a utilização de um acelerômetro. A Tabela 1.1 mostra que os algoritmos baseados em *threshold* com a utilização de acelerômetros, possuem uma precisão satisfatória. Também é possível verificar que isso ocorre em Xu, Zhou e Zhu (2019), onde os melhores resultados foram obtidos pelos algoritmos que utilizam o *Kinect*, entretanto, os outros algoritmos baseados em *threshold*, no geral, também possuem resultados satisfatórios.

Em Hsieh et al. (2017) foi utilizado um algoritmo hierárquico de detecção de queda, implementado tanto com abordagem *threshold* (limiar) quanto em *machine-learning*. O algoritmo baseado em *threshold* foi utilizado para nortear o desenvolvimento do presente projeto e do algoritmo de detecção de quedas. O diagrama de blocos pode ser visualizado na Figura 1.3 abaixo.

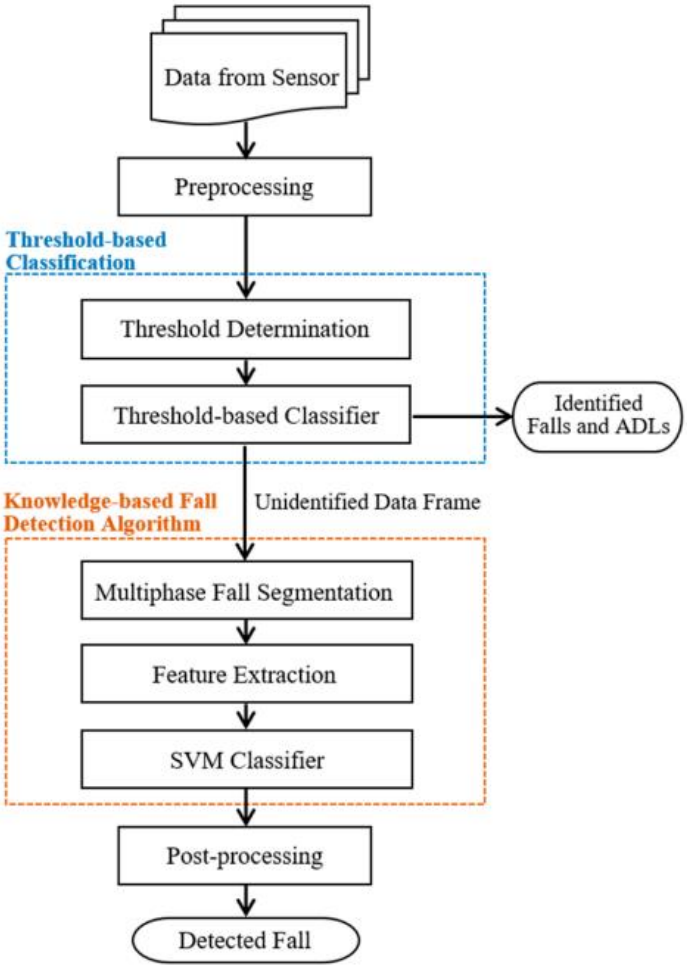

Figura 1.3. Diagrama de blocos do algoritmo proposto em Hsieh et al. (2017).

Em Hsieh et al. (2017), o algoritmo é dividido em quatro estágios: pré-processamento, classificação baseada em *threshold*, uma fase de algoritmo de detecção de queda baseado em conhecimento (*threshold*), e por fim, o pós-processamento. A fase de pré-processamento busca determinar um mesmo tamanho de janela de tempo para cada captura de dados. Os detalhes de cada fase serão tratados mais adiante neste trabalho na seção de desenvolvimento do algoritmo. A classificação baseada em *threshold* utiliza-se dos módulos da aceleração nos três eixos, que representa a variação especial da aceleração no intervalo de queda, e o módulo horizontal que é a norma euclidiana da aceleração no plano horizontal, para distinguir quedas de outras atividades do dia a dia.

O autor ainda divide a queda em 3 fases: queda livre, impacto e descanso. A fase de detecção de queda baseada em conhecimento faz uso dessa segmentação. O algoritmo usa amostras e valores máximos para tratar os dados das fases de queda livre, impacto e descanso. O autor também realizou testes com o algoritmo baseado em *machine learning*, e obteve menor precisão comparado ao

algoritmo baseado em *threshold* exibido na Figura 1.3. Ainda que a diferença de precisão seja menor que 1%, o algoritmo baseado em conhecimento é mais simples de ser implementado. Tanto a precisão quanto a acurácia (exatidão) foram maiores do que 99% (99.05% e 99.33%, respectivamente). Esses dados revelam que o algoritmo é eficiente para lidar com detecção de quedas e ainda obtém um bom desempenho. Por isso, o trabalho desenvolvido em Hsieh et al. (2017), será utilizado como base para o algoritmo desenvolvido neste projeto.

Considerando que o custo do *Kinect* é cerca de dez vezes maior que o custo do ADLX345, a escolha pelo acelerômetro possui um custo benefício maior. Além disso, soluções baseadas em Wi-Fi foram descartadas porque elas se restringem à um ambiente específico e possuem pouca mobilidade. Finalmente, por ser mais simples e possuir bons resultados, o algoritmo utilizado neste projeto será baseado em *threshold*.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo aborda os principais embasamentos teóricos necessários para moldar a proposta de desenvolvimento do dispositivo de detecção e alerta de queda do presente trabalho.

2.1 Internet of Things

Nos dias de hoje é difícil imaginar viver em um mundo sem *Internet*, principalmente para os nascidos a partir dos anos 2000. A cada dia que passa a *Internet* se torna mais parte de nossas vidas pessoais e profissionais. Diversos dispositivos como *smartphones*, sensores, computadores, *tablets* ou qualquer outro dispositivo "inteligente" são exemplos que nos cercam diariamente. Essas e outras tecnologias são consideradas como IoT (*Internet of Things*).

Inicialmente, este tipo de tecnologia era conhecido como *Internet of Computers*, sendo mais tarde adotado o termo *Internet of People*. Devido ao rápido desenvolvimento do chamado ICT (*Information and Communications Technology*), ou tecnologia da comunicação e informação, o conceito se difundiu de maneira tão ampla que hoje é conhecido como IoT (*Internet of Things*), pois quase tudo é passível de comunicação com a *Internet*. Na concepção do IoT, os mais diversos dispositivos com certo nível de inteligência de processamento, podem ser capazes de se tornar acessível e identificados, afim de extrair informações ou realizar atividades estabelecidas. (ABDUL-QAWY et al., 2015)

Hoje, nas inovações da chamada ICT (*Information and Communications Technology*), o IoT ocupa grande espaço, pois é amplamente reconhecida como uma das infraestruturas mais importantes e promissoras da área, pois permite de maneira descomplicada a interação e integração do mundo físico e do mundo cibernético. (ABDUL-QAWY et al., 2015)

O IoT pode ser considerada como a base do que pode vir a ser a *internet* no futuro, permitindo operações inteligentes e comunicações avançadas dentre os mais diversos dispositivos, sistemas ou serviços. Não há dúvidas de que é um progresso considerável no mundo das tecnologias das comunicações, pois o IoT será capaz de interconectar relógios a *smartphones*, geladeiras a computadores. No conceito do IoT a maior quantidade de objetos deverá ser capaz de trocar informações com outros objetos. (ABDUL-QAWY et al., 2015)

Ainda hoje, não há uma padronização das tecnologias IoT, mas segundo Ma (2011) pode ser definido como "baseado nos tradicionais portadores de informações, incluindo a *internet*, rede de telecomunicações e assim por diante, a Internet das Coisas (IoT) é uma rede que interconecta objetos físicos comuns com os endereços identificáveis para fornecer serviços inteligentes". Já segundo Botterman (2009), IoT foi definido como "uma rede mundial de objetos interconectados unicamente endereçáveis, baseada em protocolos de comunicação padrão".

Apesar de diversas definições atribuídas as tecnologias IoT, seu principal valor está na sua capacidade de interconectar os mais variados dispositivos, que se diferem em seus propósitos, *designs*, sistemas, inteligências e até mesmo protocolos de comunicação, para que possa haver troca de informações, para coleta, geração e processamento de dados, seja por meio de aplicativos ou qualquer outro sistema de gerenciamento, como a nuvem, por exemplo (ABDUL-QAWY et al., 2015).

2.2 IEEE 802.11 N

Em outubro de 2009 o 802.11n foi publicado pela IEEE, trazendo consigo a implementação de aprimoramentos de camada MAC e camada física em relação às versões anteriores do padrão 802.11 (KARMAKAR; CHATTOPADHYAY; CHAKRABORTY, 2017).

Dentre os aprimoramentos da camada física, pode-se citar a implementação do MIMO-OFDM *(Multiple-input, multiple-output orthogonal frequency-division multiplexing),* onde múltiplas antenas são utilizadas para realizar a comunicação, buscando otimizar a taxa de transmissão e minimizar os erros. A técnica MIMO pode ser implementada através de 3 mecanismos no 802.11n: multiplexação espacial (SM - *Spatial Multiplexing),* codificação espaço-tempo (STBC - *Space-time Block Coding)* e transmissão *Beamforming.*

No mecanismo SM, o sinal de saida é dividido em múltiplos "espaços" que são transmitidos simultaneamente por diferentes antenas, o que maximiza as taxas de transmissão. Esses espaços são chamados de fluxos espaciais (*spatial streams)* e chegam ao receptor com diferentes intensidades e atrasos. No STBC, usando diferentes fluxos espaciais, o sinal é transmitido de maneira redundante (e codificado de maneiras diferentes), aumentando a intensidade do sinal e reduzindo a taxa de erros para a SNR do sistema. A ideia da transmissão *Beaforming,* é utilizar um arranjo de múltiplas antenas, com foco em transmitir o sinal na correta direção onde se encontra o receptor, aumentando a taxa de transmissão e melhorando a SNR do receptor (KARMAKAR; CHATTOPADHYAY; CHAKRABORTY, 2017).

Outro aprimoramento do 802.11n é a ligação de canais (*channel bonding),* que consiste na combinação de dois canais adjacentes de 20 MHz para criar um canal de 40 MHz. Na faixa de frequência dos 5 GHz é onde a técnica realmente faz diferença, pois permite até 24 canais de 20 MHz, ou até 12 canais de 40 MHz, além de ser uma faixa de frequência que sofre menos interferência. A técnica de ligação de canais é capaz de dobrar a taxa física de transmissão de dados, porém outros dispositivos terão menos canais para operarem.

O 802.11n trouxe diversas opções de MCS (*modulation and coding scheme),* que determina o número de canais, modulação e taxa de codificação utilizada. O MCS influência diretamente na taxa de transmissão do sistema, sendo a maior possível quando se usa o MCS 31, com 4 canais, modulação 64-QAM e taxa de codificação 5/6, atingindo o máximo de 600 Mbps.

No 802.11n foram introduzidos outros aprimoramentos de camada MAC, como a agregação de quadros e o *block acknowledgement*, onde os ACK's para cada quadro transmitido, podem ser confirmados em blocos junto com a utilização da agregação de quadros.

A *frame aggregation (*agregação de quadros) pode ser através do mecanismo que agrega quadros de mesma origem e destino, reduzindo os eventos de contenção, esse mecanismo é denominado *Aggregated MSDU* (A-MSDU). Além desse mecanismo, é possível também a concatenação de MPDUs em um quadro MAC agregado, na técnica denominada *Aggregated MPDU (*A-MPDU).

Como cada MPDU é criptografada e descriptografada separadamente, é possível reconhecer os pacotes enviados em blocos de A-MPDUs que tenham o mesmo destino. Esse quadro pode conter até 64 MPDUs. E por fim, há ainda um mecanismo de retransmissão seletiva para possíveis quadros perdidos.

É pelo 802.11n que o sistema desenvolvido nesse projeto se conectará até um *access point* (AP), permitindo conexão com a *internet*. Uma vez conectado, será transmitida uma mensagem *HTTP GET* que contendo as informações de e-mail do responsável, o nome do paciente e a posição de queda, que será tratado mais à frente neste trabalho.

2.3 Bluetooth Low Energy

Em 1994 a Ericsson lançava o padrão Bluetooth, desde seu início, o Bluetooth tinha o objetivo de possibilitar a comunicação entre dispositivos portáteis, apresentando: segurança, confiabilidade e baixo consumo de energia. O SIG (*Special Interest Group*) publicou a versão mais recente em 2016, denominada: Bluetooth 5.0, que conta com a tecnologia BLE. A versão mais recente conta com otimizações para Internet das Coisas (IOT - *Internet of Things*), Redes de Sensores sem Fio (RSSF) e *Smart Cities* (Cidades inteligentes). Porém, as versões 4.x do BLE estão entre as mais populares do

mercado para comunicação entre dispositivos próximos (HOSNI, 2017). A versão 4.2 do BLE é a utilizada neste projeto.

Segundo os dados oficiais do SIG Bluetooth, mais de 33.000 empresas se aderiram ao SIG até o ano de 2017 (SIG BLUETOOTH, 2019). Segundo Armstrong e Helps (2007), o SIG Bluetooth possuía 6.000 membros em 2006, assim em 11 anos a empresa cresceu 550% em número de membros. Além disso, segundo ABI Research aponta que 30 bilhões de dispositivos que utilizam a tecnologia Bluetooth farão parte da grande rede formada pela *Internet* das Coisas (IoT - *Internet of Things*) até 2020.

O BLE foi desenvolvido como um aprimoramento compatível com o Bluetooth clássico, sendo assim uma solução para controle e monitoramento de aplicações com baixo consumo energético (HOSNI, 2017). Como citado em Veiga (2018) "Em especial, o sistema de baixa energia inclui características projetadas para produtos que requeiram menor consumo de corrente, menor complexidade e menor custo comparado com outros sistemas, e para casos de uso e aplicações com baixas taxas de dados e ciclos de trabalho reduzidos".

Operando na faixa de frequência ISM (*Industrial, Scientific and Medical*) não licenciada a 2,4GHz, com 40 canais de 2 MHz, o BLE possui dois tipos de canais diferentes: canais de anúncio (3 canais) e canais de dados (37 canais). Os canais de anúncio são utilizados na comunicação *broadcast*, e os canais de dados podem ser utilizados como canais de anúncio secundários.

Um dispositivo BLE possui dois modos de operação: modo de anúncio e modo de escuta. Ao entrar em modo de anúncio o dispositivo envia um pacote de anúncio, a cada certo intervalo de tempo, revezando entre os três canais reservados para anúncio, esse modo anuncia a presença do dispositivo. Um outro dispositivo em modo de escuta que desejasse iniciar uma conexão, só precisa transmitir uma solicitação no mesmo canal em que recebeu o pacote-anúncio. Um dispositivo também pode solicitar conexão com outro dispositivo já pareado, independente deles estarem ou não em modo de anúncio. Durante o paramento, os dispositivos trocam chaves criptográficas para garantir uma conexão segura (HOSNI, 2017).

Com a conexão estabelecida, o dispositivo solicitante assume o papel de mestre, enquanto o outro é o escravo da conexão. O mestre define se algum canal será evitado e a duração do intervalo de evento, podendo alterar esses dados ao longo da conexão. Por fim, a comunicação é feita utilizando os canais de dados disponíveis.

A tecnologia BLE surge em 2010 com o Bluetooth 4.0, trazendo grandes avanços na eficiência de consumo energético que permitiu novas possibilidades de aplicações, com avanços importantes para redes de sensores sem fio (RSSF) e IoT, além de uso para cuidados de saúde, que é o foco deste projeto.

Para reduzir o consumo de energia o BLE faz-se uso de mensagens de curta duração (DOS SANTOS; ALVES, 2018). Essas mensagens podem ser de dados ou mensagens denominadas *advertisement messages* (mensagens de anúncio) que carregam algum tipo de anúncio. As mensagens de anúncio são as únicas que podem ocorrer em modo *broadcast* e são controladas pelo GAP (*Generic Access Profile)*.

Um dos tipos de mensagem de anúncio mais comum são os *beacons*. Os *beacons* podem ser recebidos e tratados por dispositivos compatíveis e podem carregar informações, como por exemplo, dados que possam permitir ao dispositivo estimar a localização do dispositivo emissor. O padrão GATT (*Generic Attribute Profile)* é responsável por permitir que dispositivos periféricos estabeleçam conexão com um dispositivo central, permitindo a troca e o armazenamento de dados (DOS SANTOS; ALVES, 2018).

Figura 2.1. Topologia de conexão GATT (MEDIATEK LABS, 2019).

É definido no GATT que o dispositivo periférico só pode se conectar a um dispositivo central. Quando um dispositivo periférico se conecta ao dispositivo central, este para de enviar mensagens de anúncio, sendo impossível de se estabelecer uma conexão com o dispositivo central.

O *central device* (dispositivo central) da figura 2.1 é tratado como o GATT *Client* enquanto os dispositivos periféricos assumem o papel de GATT *Server*. As trocas de dados ocorrem por meio de solicitação do GATT *Client*, que são respondidas pelo GATT S*erver*.

O Bluetooth 4.1 trouxe novos recursos e trouxe como objetivo deixar as conexões mais rápidas e estáveis. As reconexões manuais se tornaram menos frequentes e a troca de dados passou a ser mais eficiente. Outro destaque é a flexibilidade para dispositivos que utilizam a topologia Bluetooth, através de duas topologias: *link-layer e dual-mode*.

O Bluetooth 4.2 é a versão utilizada no desenvolvimento desse projeto. Essa versão trouxe significativas mudanças na privacidade dos usuários e nas taxas de transmissão de dados das conexões. Como agora é necessário permissão para o rastreamento através de *beacons*, impede que dispositivos rastreiem outros sem permissão.

Na versão 4.2 a taxa de transmissão é até 2.5 vezes mais rápida que nas versões anteriores. O tamanho dos pacotes também sofreu aumento, o que reduziu a probabilidade de erros de transmissão e o consumo de bateria. Assim, o padrão Bluetooth 4.2 possui conexões muito mais eficientes que suas versões anteriores.

Por seu amplo uso comercial e simplicidade, neste projeto o Bluetooth 4.2 será o mecanismo de comunicação entre a aplicação do celular e o dispositivo de detecção de queda. Isso permite que os parâmetros de configuração da nossa solução sejam transmitidos através de um celular para o dispositivo detector de queda, tais como: SSID da rede Wi-Fi, senha da rede Wi-Fi, nome do usuário e e-mail do responsável pelos cuidados em casos de queda.

2.4 Acelerômetro

Segundo da Rocha (2013), “a aceleração é uma grandeza física cinemática que aponta quão rapidamente a velocidade de um corpo varia ao longo do tempo”. Os dispositivos acelerômetros são sensores capazes de medir essa variação em função do tempo. Seu uso hoje em dia é bastante difundido e é possível observá-lo em diversos dispositivos, como em celulares e tablets, para ajustar a orientação da tela e para jogos eletrônicos, em laptops para proteção do disco rígido e até drones para estabilização de vôos.

Esses sensores conseguem converter a aceleração da gravidade ou movimentos, em sinais elétricos analógicos ou digitais para mensurar a força aplicada ao sistema. Este sistema indica em tempo real, no caso analógicos ou em forma discreto, no caso digital, a aceleração instantânea do objeto no qual o acelerômetro é acoplado. A unidade utilizada pelos acelerômetros para medir a aceleração é o

“g”, que é definido a partir da força gravitacional exercida pela Terra a um objeto ou pessoa, um exemplo é a aceleração da gravidade no nível do mar, em que 1g equivale a aproximadamente 9,80665 m/s^2. Os acelerômetros são capazes de medir aceleração em apenas um eixo, ou seja, são direcionais. Caso seja necessário a medição nos três eixos, são utilizados acelerômetros multieixos (x, y e z), que nada mais são do que a união de três acelerômetros ortogonais entre si (PERNIA-MÁRQUEZ, 2004).

Os acelerômetros podem ser construídos de diversas maneiras, dentre as mais utilizadas estão os acelerômetros capacitivos, também conhecidos como acelerômetros ativos, os acelerômetros pieozelétricos, conhecidos como acelerômetros passivos, e os acelerômetros MEMS (*Micro-Electro-Mechanical Systems*).

Os acelerômetros ativos possuem circuitos internos que convertem a carga do acelerômetro em um sinal de tensão, seu ponto negativo é a exigência de uma fonte de tensão constante para o perfeito funcionamento (PERNIA-MÁRQUEZ, 2004). O tamanho também compromete a utilização em alguns casos.

Figura 2.2. Exemplo de um acelerômetro capacitivo (MECANICA INDUSTRIAL, 2019).

Já os acelerômetros passivos enviam a carga gerada pelo elemento sensor, que na maioria das vezes é um material piezoelétrico. Há uma massa presa em um cristal piezoelétrico, sendo que quando ocorre uma aceleração no sistema, a massa gera uma deformação no cristal, gerando um sinal. Este sinal gerado, tem amplitudes muito pequenas, o que faz com que esse tipo de acelerômetro necessite de um amplificador para elevar o sinal. Ao contrário dos acelerômetros ativos, os passivos não necessitam de fonte de tensão constante (PERNIA-MÁRQUEZ, 2004). Os acelerômetros piezoelétricos possuem dimensões ainda maiores do que os capacitivos, portanto para algumas aplicações, como o propósito deste trabalho, também se torna inviável sua utilização.

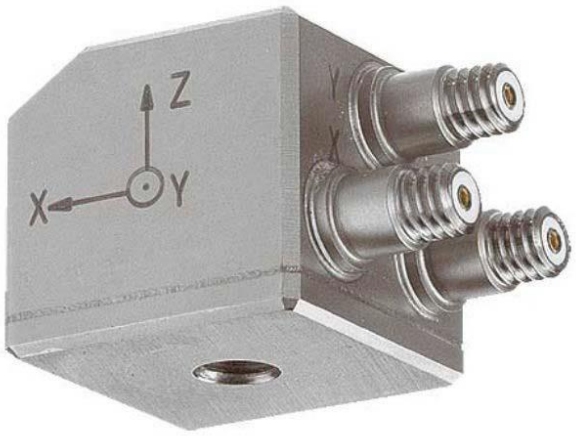

Figura 2.3. Exemplo de um acelerômetro piezoelétrico (VIBRODATA, 2019).

Segundo Bao (2005) a tecnologia MEMS (*Micro-Electro-Mechanical Systems*) é definida como "um microssistema que consiste de sensores micromecânicos, atuadores e circuitos microeletrônicos", ou seja, é uma tecnologia de dispositivos microscópicos.

Os acelerômetros MEMS se dividem em piezoresistivos e capacitivos. Os piezoresistivos consistem em um sistema de grau único de liberdade de uma massa de prova (quantidade conhecida de massa usada em um instrumento de medição como referência para a medição de uma quantidade desconhecida) suspensa na ponta de um cantiliver (estrutura em consola) que possui propriedades de uma mola e um material piezoresistivo no início do cantiliver, conforme a Figura 2.4. (ALBARBAR et al., 2007)

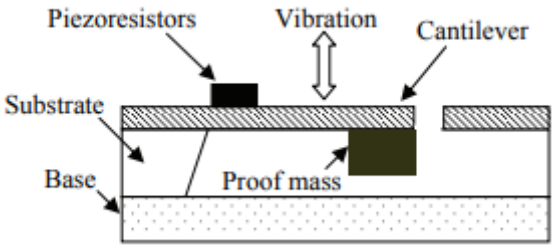

Figura 2.4. Esquema de um acelerômetro MEMS piezoresistivo (ALBARBAR et al., 2007).

Já os acelerômetros MEMS baseados em capacitores, medem as alterações da capacitância entre uma massa de prova e um eletrodo fixo, separados por uma abertura estreita, conforme pode-se observar na Figura 2.5 (ALBARBAR et al., 2007).

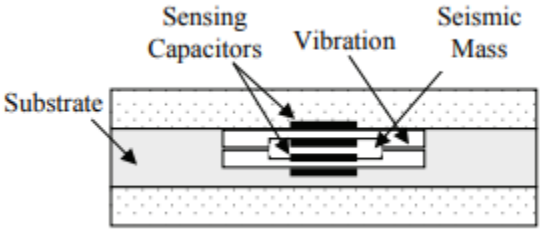

Figura 2.5. Exemplo de um acelerômetro MEMS capacitivo (ALBARBAR et al., 2007).

Os acelerômetros MEMS, portanto, são dentre os citados, os de menor dimensão, como é possível observar pela figura a comparação de seu tamanho médio com uma moeda de 25 centavos de dólar.

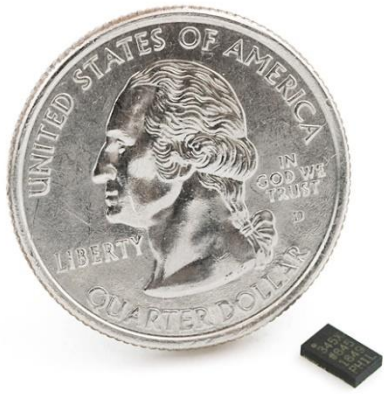

Figura 2.6. Comparação do tamanho de um acelerômetro MEM com uma moeda de 25 centavos de dólar (KARLSSON ROBOTICS, 2019).

Com a grande popularização do uso de acelerômetros em dispositivos variados, hoje se tem uma enorme variedade de oferta deste tipo de sensor, o que provoca uma queda dos preços em geral, de forma que se torna um sensor barato e poderoso, devido as suas diversas aplicabilidades (DA ROCHA, 2013).

O acelerômetro utilizado neste trabalho é um MEMS capacitivo (ANALOG DEVICES, 2009). Sua escolha se deu principalmente pelas suas dimensões reduzidas e ao baixo consumo, características essenciais para projetos de IoT.

3 COMPONENTES UTILIZADOS

Este capítulo descreve todos os componentes eletrônicos utilizados no sistema de detecção de queda desenvolvido, suas especificações e funcionalidades.

3.1 ESP32

O ESP32 é um microcontrolador com dois núcleos de processamento, possui 520 Kbytes de memória SRAM, permitindo a execução de programas complexos, interação com redes CAN, possui suporte UART, I2C, SPI, Infravermelho, SDIO (interface com cartão de memória) e outros recursos (ESPRESSIF SYSTEMS, 2019). Os componentes internos do ESP32 podem ser visualizados na Figura 3.1.

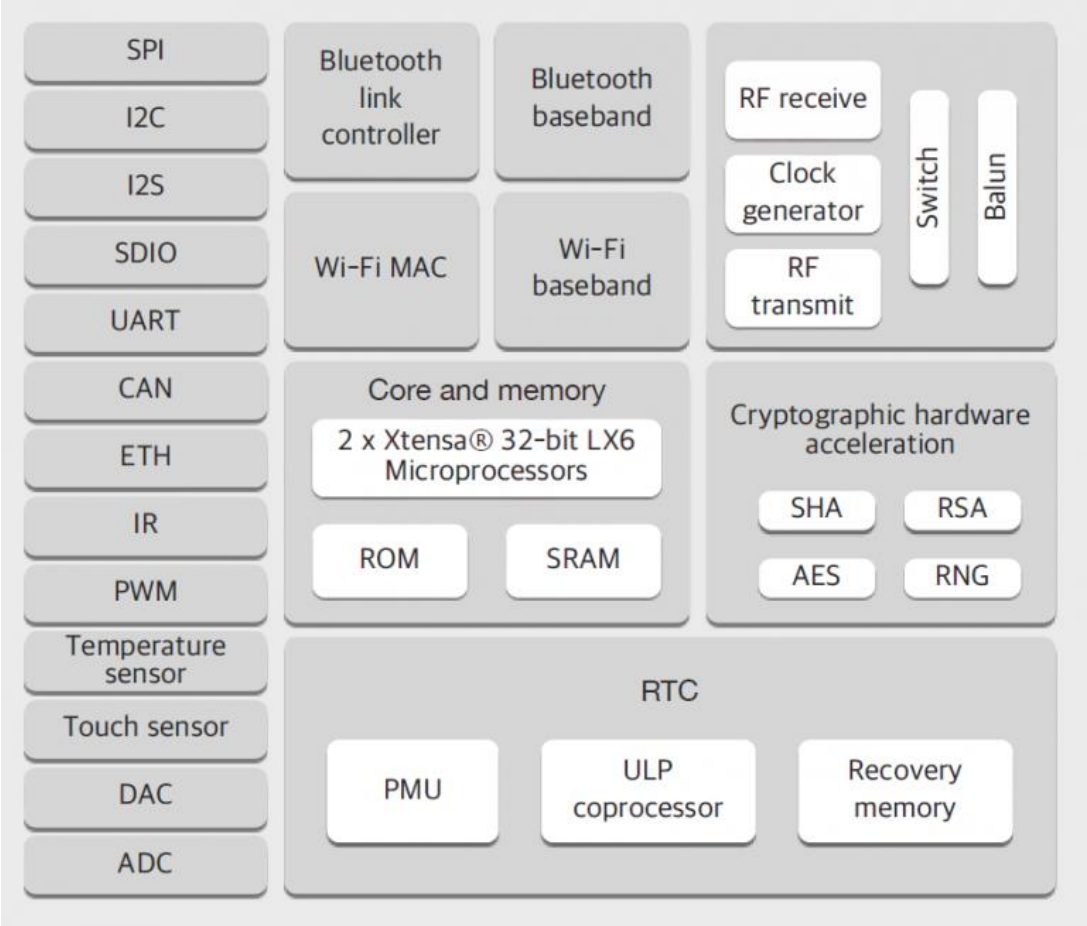

Figura 3.1. Diagrama de blocos dos componentes do ESP32 (ESPRESSIF SYSTEMS, 2019).

O ESP32 possui o modo "*sleep*" mais sofisticado que seus microcontroladores anteriores, com a presença de um coprocessador de "ultrabaixo" consumo (ULP *Coprocessor*), que permite a delegação de tarefas (mais simples) enquanto os outros processadores estão inativos. Além de possuir o Bluetooth 4.2, com a tecnologia BLE, permitindo compatibilidade com dispositivos que utilizem Bluetooth (ESPRESSIF SYSTEMS, 2019). A Tabela 3.1 detalha as especificações do microcontrolador utilizado.

Tabela 3.1. Especificações do ESP32 (ESPRESSIF SYSTEMS, 2019).

Processador:	Microprocessador Tensilica Xtensa 32-bit LX6
Núcleos:	2
Frequência do *Clock*:	Até 240 MHz
Performance:	até 600 DMIPS
Conectividade sem Fio:	
Wi-Fi:	802.11n @ 2.4GHz até 150Mbit/s
Bluetooth:	v4.2 BR/EDR e BLE
Memória Interna:	
ROM:	448 KB
SRAM:	520 KB
RTC slow SRAM:	8 KB
RTC fast SRAM:	8 KB
eFuse:	1 Kbit
Memória Flash Externa:	Até 16 MB de memória externa.

O ESP32 conta ainda com alguns recursos de segurança, como *Flash Encryption*, conectividade 802.11n com suporte aos protocolos de segurança WFA, WPA/WPA2 e WAPI, além de *boot* seguro e aceleração de criptografia em *hardware* utilizando AES, SHA-2, RSA, ECC e RNG (ESPRESSIF SYSTEMS, 2019).

Por fim, a Tabela 3.2 lista os periféricos de entrada e saída do ESP32, os pinos do microcontrolador podem ser visualizados na Figura 3.2.

Tabela 3.2. Periféricos do ESP32 (ESPRESSIF SYSTEMS, 2019).

Periféricos de Entrada/Saída do ESP32
10 GPIOS com suporte a toque capacitivo.
16 canais e conversor SAR ADCs de 12 Bits.
2 canais de 8 bits DACs
2 Interfaces I²C
2 Interfaces UART
Controlador CAN 2.0
4 Interfaces SPI

2 Interfaces I²S
RMII (Parte Ethernet)
16 canais de PWM

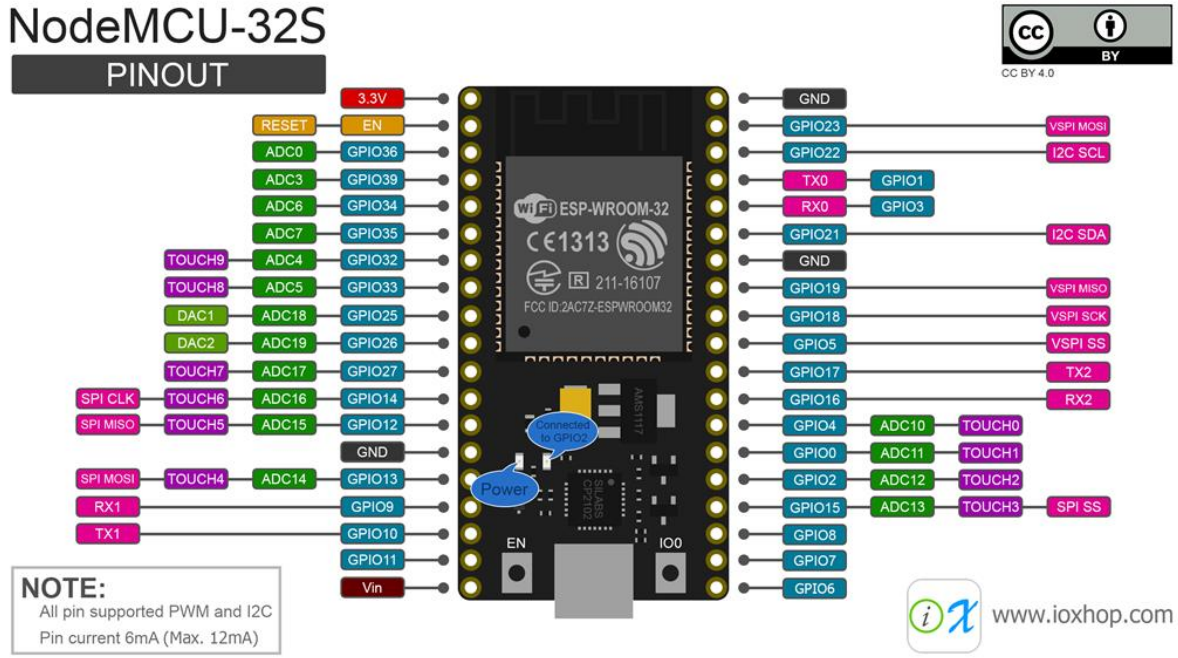

Figura 3.2. Microcontrolador ESP32 (INSTRUCTABLES CIRCUITS, 2019).

3.2 ADXL345

Um dos principais componentes do trabalho desenvolvido é um acelerômetro de 3 eixos, modelo ADXL345, um acelerômetro de alta resolução (13-bit), de medição de até 16g (positivo ou negativo), e acessível por conexão SPI ou I2C (ANALOG DEVICES, 2009).

O ADXL345 é adequado para se obter medidas de aceleração estática de gravidade, para aplicações que exijam medidas de inclinação. Além disso, consegue medir aceleração dinâmica resultante de movimento ou choque (ANALOG DEVICES, 2009). Por possuir detecção de choque, inatividade, inclinação e queda livre, o ADXL é adequado para o projeto. A Tabela 3.2 mapeia as especificações do ADXL345.

Tabela 3.3. Especificações do ADXL345 (ANALOG DEVICES, 2009).

Tensão de Alimentação	2-3.6V
Consumo de corrente em modo medição	40uA
Consumo de corrente em modo *standby*	0.1uA

O acelerômetro escolhido é baseado na tecnologia MEMS (*Micro Electro Mechanical Systems).* MEMS é uma tecnologia que busca fabricar elementos mecânicos em uma escala próxima a de circuitos

microeletrônicos. Apesar de serem muito pequenos, não deve ser confundido com circuitos microeletrônicos, pois possuem características divergentes: enquanto circuitos eletrônicos são estruturas sólidas e compactas, os MEMS possuem cavidades, canais, buracos, membranas e outras estruturas que tentam simular ou imitar partes mecânicas. A ideia principal por trás é buscar trazer dispositivos já existentes para uma escala menor de tamanho, o que não só reduz o custo de material, como permite aplicações que antes eram impossíveis. Como a maioria dos MEMS são feitos com silício, eles possuem fácil integração com circuitos microeletrônicos, permitindo aquisição, armazenamento e filtragem de dados, além de comunicação entre dispositivos. Todos esses fatores tornam a tecnologia MEMS adequada para o desenvolvimento do projeto (ANDREJAŠIC, 2008).

Figura 3.3. Sensor acelerômetro ADXL345 (SPARKFUN, 2019).

3.3 TP4056

O TP4056 é um módulo carregador para baterias de lítio, com conexão por cabo micro USB e LEDs indicadores de carga, o que possibilita o carregamento da bateria sem removê-la do dispositivo. Ele será utilizado neste projeto para recarregar a bateria que alimenta o sistema. Seu funcionamento é relativamente simples, bastando apenas conectar a bateria nos terminais positivo (B+) e negativo (B-). O LED vermelho, indica que a bateria está carregando, e o LED verde acende quando a carga estiver completa (NANJING TOP POWER ASIC CORP, 2019). A Figura 3.4 mostra o TP4056. Na parte superior é possível ver os terminais B+ e B-. A Tabela 3.4 traz as especificações relevantes do módulo TP4056.

Tabela 3.4. Especificações do módulo TP4056 (NANJING TOP POWER ASIC CORP, 2019).

Tensão de Operação:	5V
Capacidade Máxima de Carga:	1A (ajustável)
Tensão de Corte na Saída:	4.2V +/- 1%
Temperatura de Operação:	-10ºC à 85ºC
Dimensões:	26 x 17 x 5mm

Figura 3.4. Módulo TP4056 (FILIPEFLOP, 2019).

3.4 Buzzer

Para emitir um sinal sonoro caso o dispositivo tenha detectado uma queda, foi escolhido um *buzzer* que opera em 5V. Apesar da bateria escolhida fornecer apenas 3,8V, não prejudica o funcionamento do *buzzer*, apenas diminui a intensidade do som emitido (ver Figura 3.5).

Figura 3.5. Emissor de som, *buzzer*.

3.5 Chave botão com trava

Para ligar e desligar o dispositivo foi utilizado uma chave botão com trava de dimensão 8 mm x 8 mm. Sua escolha no projeto foi por ser a menor chave encontrada no mercado, o que ajuda a economizar espaço no dispositivo (ver Figura 3.6).

Figura 3.6. Chave botão com trava.

3.6 LED

Foram escolhidos LEDs difusos de luminosidade 800 mcd nas cores vermelho e amarelo para indicação do carregamento da bateria e do estado do dispositivo (ligado ou desligado). O diâmetro da cabeça do LED é de 3 mm, o que foi a causa da escolha deste tipo de LED devido a sua pequena dimensão. Sua tensão de operação é de 2V e corrente de 20 mA (ver Figura 3.7).

Figura 3.7. LED vermelho 3mm.

3.7 Bateria

Para alimentar o sistema, foi escolhida a bateria do celular LGE450F, o motivo de escolha foi por ela operar em 3.8V e possuir capacidade de 1700 mAh (suficiente para alimentar todos os componentes do sistema), além de ser uma bateria do tipo Li-Íon (capaz de ser carregada pelo modulo TP4056), é uma bateria pequena e leve (ver Figura 3.8).

Figura 3.8. Bateria utilizada no projeto para alimentar o sistema.

4 DESENVOLVIMENTO DO SISTEMA DE DETECÇÃO DE QUEDA

Este capítulo apresenta o desenvolvimento do sistema de detecção de queda que foi dividido em 5 etapas, dentre elas: o desenvolvimento do circuito, em que estabelece as conexões entre todos os componentes eletrônicos. O desenvolvimento do aplicativo para Android, interface por onde é configurado o dispositivo. Desenvolvimento do mecanismo de envio do e-mail, que é a solução proposta para que o responsável pelo usuário tome conhecimento em ocorrências de queda. Desenvolvimento do design do dispositivo para impressão 3D, onde se cria uma caixa que possibilite as menores dimensões possíveis e que caiba todos os componentes eletrônicos. E por fim o algoritmo de detecção de queda, onde se tenta maximizar as detecções de quedas verdadeiras e minimizar as detecções de quedas errôneas. Todo o desenvolvimento dos códigos para o microcontrolador ESP32 foi realizado na IDE do Arduino.

4.1 Desenvolvimento do circuito

O circuito de detecção e alerta de queda desenvolvido pode ser observado pela Figura 4.1. O microcontrolador ESP32 é o responsável por todo processamento de informação feito no circuito, sejam os dados enviados pelo acelerômetro ADXL345 ou gerados de trocas de informações realizadas nas comunicações Bluetooth e Wi-Fi.

O principal componente desse circuito para realizar a detecção da queda é o sensor acelerômetro ADXL345. Este sensor conta com 8 pinos, porém foram utilizados apenas 6, pois foi escolhido o protocolo de comunicação I2C (*Inter-Integrated Circuit*). Esse protocolo foi escolhido devido a sua simplicidade de funcionamento e a quantidade de pinos de comunicação exigidos, apenas o SDA, responsável pela transmissão dos dados e o SCL, responsável pelo *clock* do barramento. Foram necessários resistores de *pull-up* no valor de 4,7 KΩ para as conexões do SDA e SCL do ADXL345 nos pinos P21 e P22, respectivamente, no microcontrolador ESP32. O pino VCC do acelerômetro foi conectado a saída de 3.3 V do ESP32, pois sua tensão de funcionamento varia entre 2.0 V e 3.6 V (ANALOG DEVICES, 2009). O pino CS foi conectado ao VCC e o SDO ao GND do microcontrolador, seguindo instruções do fabricante.

A bateria utilizada foi retirada de um celular LG E450f. Foi escolhida essa bateria devida a suas dimensões, apropriadas para esse projeto, e pelo fato ser Li-Íon, pois assim é possível seu carregamento através de um modulo carregador. Seus polos positivos e negativos, foram então conectados ao módulo carregador TP4056, com entrada micro USB para carregamento externo.

No módulo carregador foram soldados dois LEDs para informar o estado de carregamento. O LED amarelo, para quando a bateria estiver completamente carregada e o LED vermelho, caso ainda esteja carregando. A saída positiva do módulo TP4056 foi conectada a entrada Vin do ESP32 e a negativa foi conectada a uma chave com trava. A outra perna da chave foi conectada ao GND do microcontrolador para abrir ou fechar a passagem de corrente, funcionando então como um botão para ligar e desligar o dispositivo.

O *buzzer*, responsável pela emissão de som em caso de queda detectada, foi conectado o polo positivo no pino P33 e o polo negativo no GND do circuito.

Por último, foi instalado um LED vermelho para informar o estado do dispositivo, ou seja, se está funcionando ou desligado. Foi adicionado um resistor de 220 Ω em série para limitar a corrente que passa pelo LED.

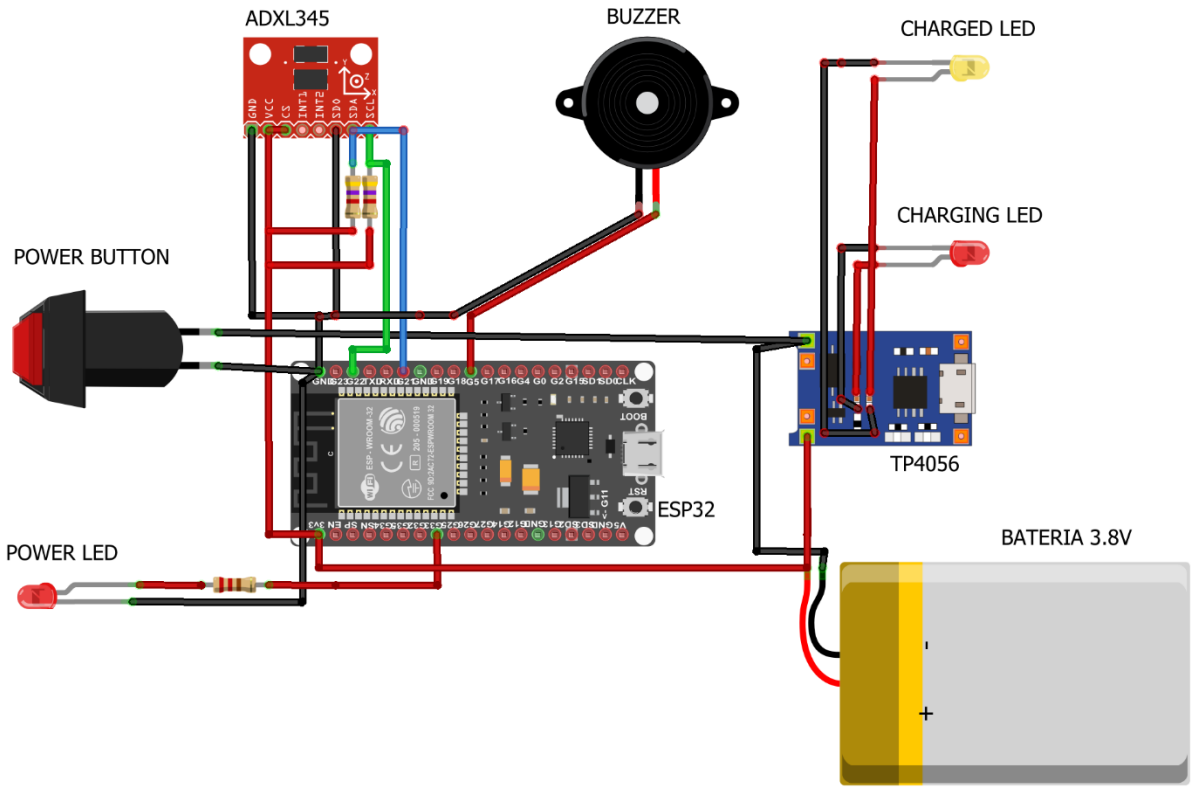

Figura 4.1. Esquemático do circuito de detecção e alerta de queda.

4.2 Desenvolvimento do aplicativo para Android

Para as configurações necessárias do dispositivo, foi desenvolvido um aplicativo para a plataforma Android. O MIT App Inventor foi escolhido como ambiente de desenvolvimento.

Esse ambiente de criação de aplicativos Android foi desenvolvido por integrantes do MIT, o Instituto de Tecnologia de Massachusetts. A intenção inicial dos desenvolvedores da plataforma era estritamente acadêmica, como uma forma mais amigável de inserir os alunos no mundo da programação para smartphones. Porém, devido à grande adesão, eles perceberam que poderia ser uma forma de popularizar a criação de aplicativos devido sua lógica de programação ser mais familiar para usuários comuns, pois contém elementos visuais (MIT APP INVENTOR, 2019).

Hoje, esta plataforma já está bem difundida, e auxilia na criação de aplicativos desde sistemas de automação residencial a ferramentas corporativas.

O conceito do MIT App Inventor é dividir a criação do aplicativo em duas etapas, a primeira é o design, que permite o usuário definir a identidade visual do que está desenvolvendo (ver Figura 4.2).

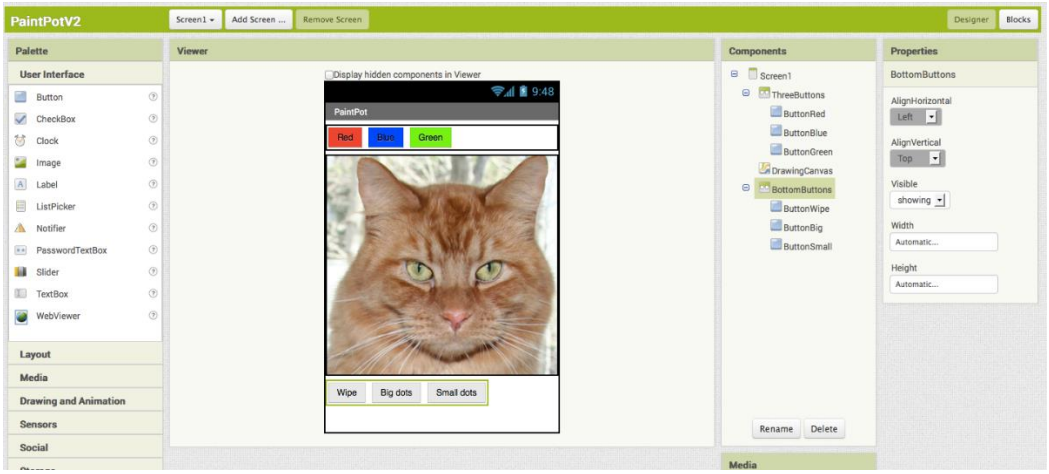

Figura 4.2. Pagina de desenvolvimento do design do aplicativo (MIT APP INVENTOR, 2019).

A segunda parte consiste no algoritmo propriamente dito, que é o diferencial do APP Inventor, pois toda lógica adicionada às funcionalidades do aplicativo, vem por meio de blocos lógicos, ou seja, elementos visuais (ver Figura 4.3).

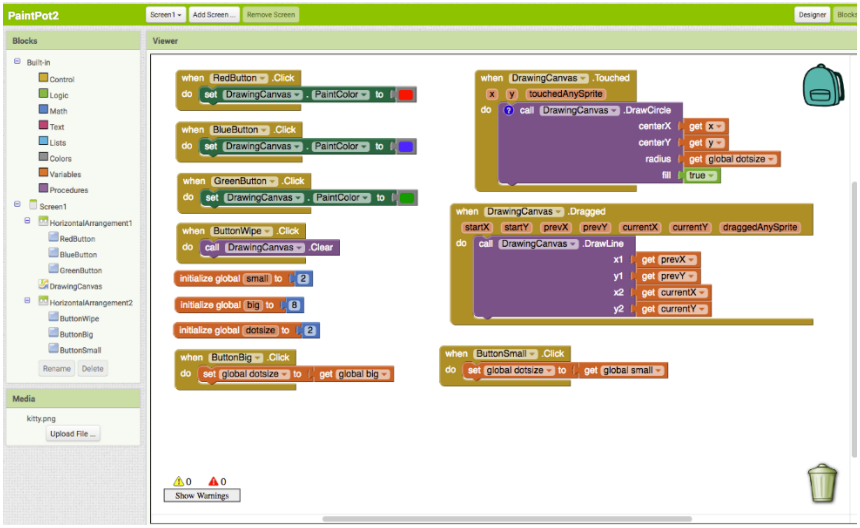

Figura 4.3. Pagina de desenvolvimento do código do aplicativo (MIT APP INVENTOR, 2019).

A plataforma conta com diversas bibliotecas nativas, o que já seria suficiente para desenvolver diversos aplicativos de complexidade média que utilizassem por exemplo mídias, mapas, sensores dos smartphones, redes sociais, banco de dados, conectividade Bluetooth clássica, dentre outros. Entretanto, algumas bibliotecas mais especificas não se encontram disponíveis nativamente, então se fez necessário a importação (ver Figura 4.4).

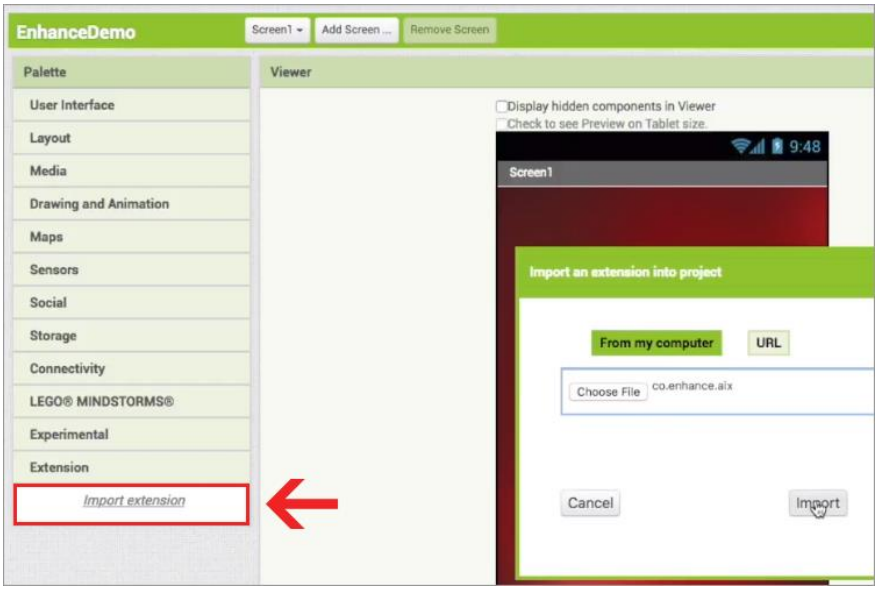

Figura 4.4. Local para importação de bibliotecas externas.

Para esse projeto, na etapa de design, foram utilizadas apenas duas bibliotecas nativas da plataforma, a "*User Interface*", para adicionar caixas de texto, etiquetas, imagens e botões e a "*Layout*", que organiza a apresentação visual. Na Figura 4.5 é possível visualizar a tela inicial do aplicativo.

Figura 4.5. Tela de início do aplicativo de configuração do dispositivo detector de queda.

Ao apertar o botão "Conectar", o usuário é levado a uma nova tela, onde seleciona o dispositivo "Detector de Queda" dentro de uma lista de opções de dispositivos *Bluetooth Low Energy*. Nesta etapa, foi necessário adicionar a biblioteca de funcionalidades do BLE, chamada "BluetoothLE1", pois a plataforma possui apenas suporte nativo para a versão clássica (ver Figura 4.6).

Figura 4.6. Tela de seleção do dispositivo por meio do *Bluetooth Low Energy*.

Após selecionado o dispositivo correto, o usuário é levado a uma tela para preencher as informações necessárias para o funcionamento do sistema de detecção e alerta de queda. As informações solicitadas são: o nome da rede Wi-Fi onde o *gadget* manterá sua conexão com a internet, a senha desta rede, o nome do usuário que utilizará o *gadget* e por fim o e-mail do responsável (ver Figura 4.7).

Figura 4.7. Tela de preenchimento de informações para configurar o dispositivo.

Por fim, ao se preencher os parâmetros solicitados pelo aplicativo, este apresenta uma mensagem de alerta ao usuário para verificar a caixa de entrada de seu e-mail, pois o *gadget* quando configurado corretamente, dispara uma mensagem de teste. Caso essa mensagem não chegue, o usuário deve selecionar “Não recebi o teste” para que o aplicativo volte para a tela inicial e reinicia o processo de configuração do *gadget*. Caso o usuário tenha recebido o e-mail de teste, deve selecionar “Recebi o teste”, assim o usuário é encaminhado para uma tela de encerramento (ver Figura 4.8).

Figura 4.8. Tela de verificação do recebimento da mensagem de teste.

Na última tela do aplicativo no processo de configuração, é exibida uma mensagem de êxito e a conexão Bluetooth com o *gadget* é encerrada, pois a partir desse momento apenas a conexão bluetooth é necessária para o funcionamento do sistema (ver Figura 4.9).

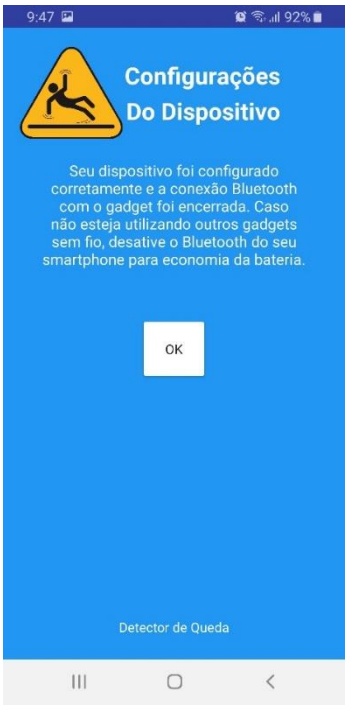

Figura 4.9. Tela de encerramento das configurações.

A plataforma do MIT App Inventor, também permite a customização do ícone do aplicativo, e a identidade visual escolhida foi conforme a Figuras 4.10 e 4.11. Este logotipo também está presente no recipiente do *gadget*. A inspiração artística veio das placas de alertas de piso molhado.

Figura 4.10. Identidade visual escolhida para o projeto.

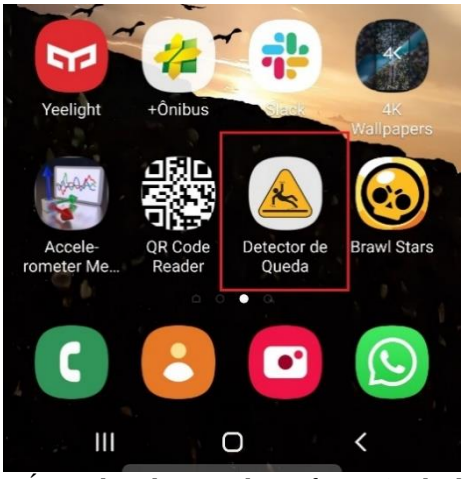

Figura 4.11. Ícone do aplicativo de configuração do dispositivo.

4.3 Desenvolvimento do mecanismo de envio do e-mail

Para disparar o e-mail quando o *gadget* detecta uma queda, de acordo com o algoritmo proposto, foi necessário a criação de um servidor *web* e de um *script* em PHP.

Ainda no microcontrolador, ao se detectar que houve uma queda, a função “sendDataToServer” é chamada, conforme Figura 4.12.

```
void sendDataToServer()
{
  if (client.connect(servername, 80)) {  //Inicia conexão, verifica conexão
    Serial.println("Connectado");
    client.println(String("GET /enviar_email.php?enderecoemail=") + emailresponsavel + String("@")
    + dominioemail + String("&nomepaciente=") + nomepaciente + String("&lado=") + lado_queda + " HTTP/1.1");
    client.println("Host: quedatcc.000webhostapp.com");
    client.println("Conexão: encerrada");  //Encerra a conexão 1.1 persistente
    client.println(); //Fim da requisição GET
    digitalWrite (5, LOW); //Ativa o buzzer
    delay(5000);
    client.stop(); //stop client
    digitalWrite (5, HIGH); //Desativa o buzzer
    temporizador.attach(periodo, funcao_interrupcao); //Reiniciar a função interrupção
    Serial.println("Reinicia Interrupção");
```

Figura 4.12. Função responsável pelo método HTTP GET para envio do e-mail.

Na parte em destaque do código da Figura 4.12, foi utilizado o método HTTP GET, em que o e-mail do responsável pelo usuário, armazenado na variável “emailresponsavel”, o domínio do e-mail,

armazenado em “dominioemail”, o nome do usuário do dispositivo, armazenado em “nomepaciente” e a posição aproximada da queda, armazenada na variável “lado_queda”, serão passados como parâmetro para um *script* em PHP chamado “enviar_email.php” que está armazenado dentro de um servidor *web*, que neste caso tem como URL : “quedatcc.000webhostapp.com”.

Quando a requisição HTTP GET chega ao servidor, as informações contidas nas variáveis que foram enviadas como parâmetros, serão processadas pelo script PHP citado anteriormente que está sendo executado dentro do servidor.

```php
<?php
$enderecoemail = $_GET["enderecoemail"];
$nomepaciente = $_GET["nomepaciente"];
$lado = $_GET["lado"];
```

Figura 4.13. Variáveis no código PHP responsaveis por receber os parametros enviados pelo método HTTP GET.

A biblioteca PHPMailer foi utilizada para poder enviar um e-mail a partir de um servidor *web*. Utiliza a porta 587, reservada para o protocolo de envio de e-mail SMTP e utiliza o TLS como protocolo de segurança.

```php
require("phpmailer/class.phpmailer.php");

$mail = new PHPMailer();
$mail->SetLanguage("br", "libs/");
$mail->SMTP_PORT = "587";
$mail->SMTPSecure = "tls";
```

Figura 4.14. Configurações dos parâmetros da biblioteca PHPMailer.

Para poder enviar um e-mail seguro através de um servidor *web*, é necessária uma autenticação do remetente. Portanto, foi criado um e-mail no Outlook, cujo servidor é “smtp.live.com” com endereço “detectorquedatcc@outlook.com” e ativou-se a autenticação SMTP, com comando “SMTPAuth = true”.

```php
$mail->IsSMTP();
$mail->Host = "smtp.live.com";
$mail->SMTPAuth = true;
$mail->Username = "detectorquedatcc@outlook.com";
$mail->Password = "[redacted]";
```

Figura 4.15. Configurações de autenticação do remetente.

A biblioteca PHPMailer permite que se insira o assunto e o corpo das mensagens através dos atributos “Subject” e “Body”. Caso seja um e-mail de teste, enviado logo depois das configurações feitas através do aplicativo Android, a mensagem enviada será a que se pode observar na Figura 4.16 no “Body” da primeira condição “if”. Para maior exatidão de informação do evento queda, foi adicionado no corpo da mensagem a posição da queda, “$lado_queda”, o nome do usuário, “$nomepaciente”, o horário, “$time” e a data, “$date”, sendo os dois últimos configurados previamente para operarem no horário oficial do Brasil (GMT-3).

```php
$mail->Subject = "Alerta de Queda de Paciente";

if ($lado == 0){
    $mail->Body = "Mensagem de teste do detector de queda. Seu dispositivo foi configurado corretamente!";
}

else{
    $mail->Body = "Foi detectado a queda para $lado_queda do(a) paciente $nomepaciente aproximadamente $time do dia
        $date";
}
```

Figura 4.16. Mensagens pré-configuradas para o assunto e o corpo do e-mail.

No momento em que a mensagem de queda for enviada, o e-mail do responsável cadastrado receberá em sua caixa de entrada as informações conforme as figuras 4.17 e 4.18. Em que será

alterado apenas a posição de queda, o nome do paciente cadastrado, o horário e a data em que uma queda for detectada.

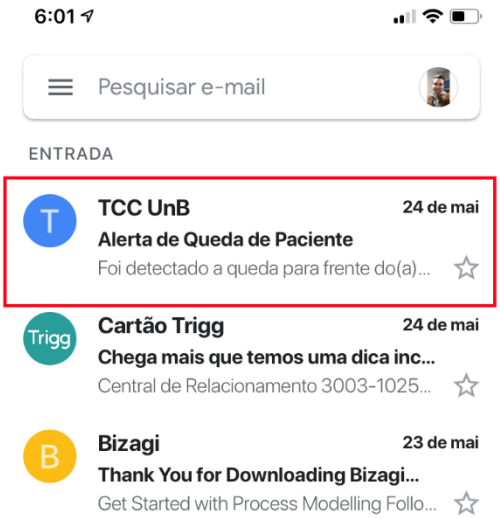

Figura 4.17. Caixa de entrada do e-mail cadastrado do responsavel.

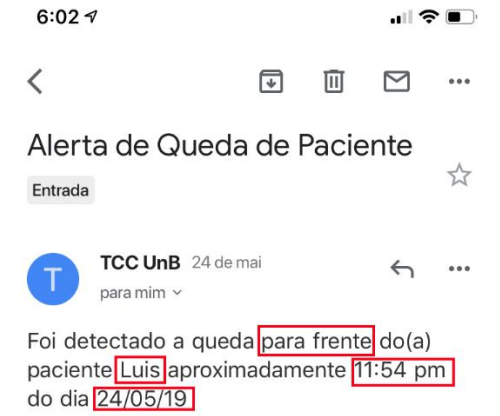

Figura 4.18. Mensagem de detecção de queda.

4.4 Desenvolvimento do design do dispositivo

Para desenvolver o invólucro do *gadget*, ou seja, o *design* do produto, foi utilizado a ferramenta SketchUp, desenvolvida pela At Last Software, que hoje pertence a Google (TRIMBLE, 2019). Essa ferramenta está disponível nas versões profissional e gratuita nas plataformas Windows e Mac. O programa é extremamente versátil, pois pode ser utilizado por engenheiros civis, engenheiros mecânicos, escultores, *designer* de games, *designer* de móveis ou qualquer outro profissional que necessite realizar esboços e até produtos finais tridimensionais, pois deste modo elimina a necessidade de utilizar modelos físicos feitos de papel, cartolina, massa de modelagem ou acrílico.

Diversas ferramentas de uso básico se encontram presentes nativamente no programa para realização dos desenhos em 3D, porém muitas vezes o esboço necessita de técnicas um pouco mais avançadas para se representar com mais exatidão a modelagem em mente, por isso o SketchUp permite adicionar plugins com ferramentas específicas.

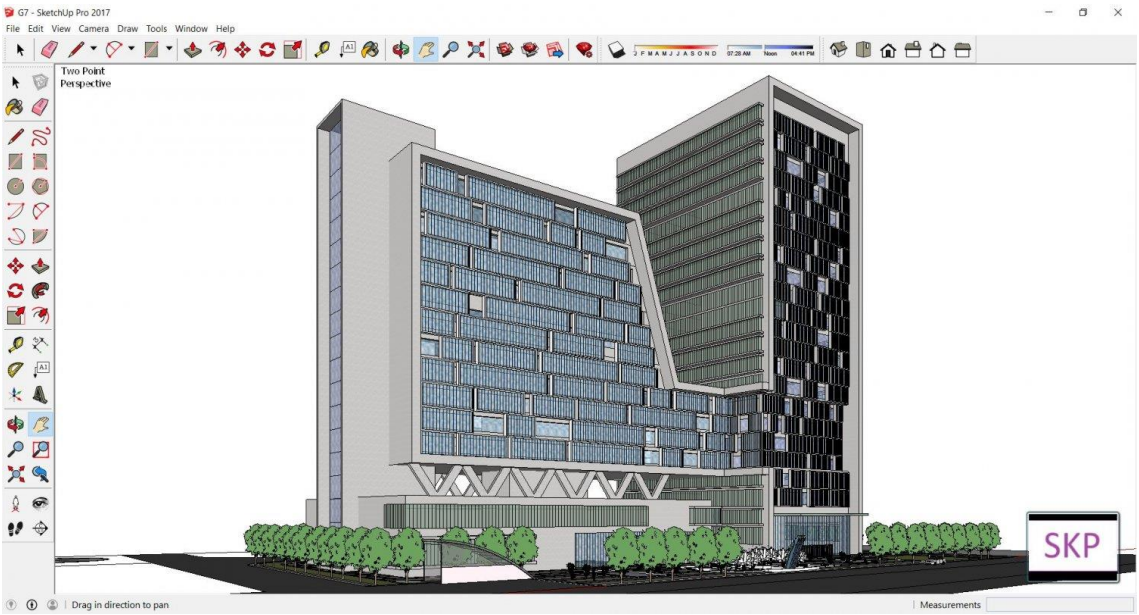

Figura 4.19. Janela do programa de modelagem 3D SketchUp com as principais ferramentas (3DEXPORT, 2019).

A finalidade desse programa no projeto foi desenhar um modelo em 3D para impressão de um invólucro para armazenar todas as peças e componentes utilizados no circuito de detecção de queda desenvolvido. Inicialmente, foi pensando para ser utilizado em outras regiões do corpo, como pulso (em formato de relógio ou pulseira) ou o tórax (junto a um cordão preso ao pescoço), porém como o pulso é uma região de muita movimentação, assim como um cordão preso ao pescoço, estas soluções foram descartadas. A escolha, então, foi por uma caixa que deverá ser presa a um cinto e alocada na região posterior da cintura, pois essa região, por ser centralizada ao corpo, não realiza muitos movimentos, o que ajuda o algoritmo a eliminar falsos positivos durante o funcionamento.

Para a realização deste projeto, foi utilizado a versão gratuita para Windows e além das ferramentas nativas, foi necessário adicionar mais dois plugins, o “RoundCorner” para arredondar as arestas do recipiente e o “SketchUp STL”, que converte o formato original do programa “.skp” em “.stl”, que é uma extensão de figuras 3D amplamente compatível com a maioria das impressoras 3D.

O *design* projetado para o *gadget* foi pensado para ser o menor possível sem prejudicar o funcionamento dos componentes, como por exemplo, o acelerômetro localizado no centro do recipiente, o que se posicionado corretamente na cintura, ficará alinhado com o corpo do usuário, melhorando os resultados.

As dimensões estabelecidas foram 54 mm x 100 mm x 25 mm, ou seja, 54 mm de altura, 10 mm de largura e 25 mm de profundidade. Na lateral direita foi alocado uma abertura para entrada micro USB para o carregamento do *gadget*, duas aberturas para os LEDs de “carga completa” e “carregando” e uma abertura para a saída sonora do *buzzer* (ver Figura 4.20).

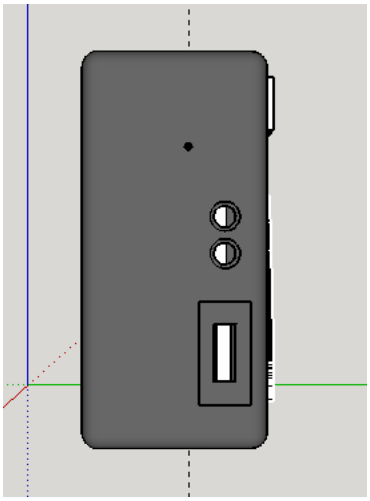

Figura 4.20. Lateral direita do recipiente.

Na lateral esquerda foi feito uma abertura para o botão *power* e para um LED que indica se o gadget está ligado ou desligado. Foram desenhados também os logotipos das tecnologias Wi-Fi e Bluetooth em alto relevo (ver Figura 4.21).

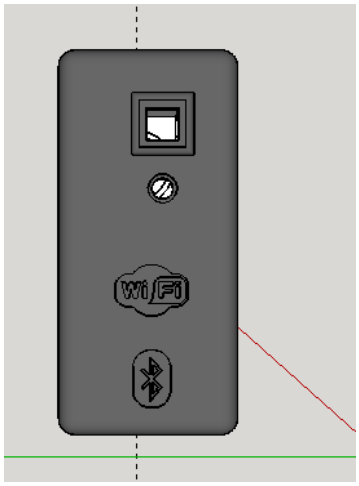

Figura 4.21. Lateral esquerda do recipiente.

Na face frontal, foram desenhados também em alto relevo, o logotipo escolhido para o projeto (o mesmo que se encontra no ícone do aplicativo) e o nome do projeto, "*Fall Detector*" (ver Figura 4.22).

Figura 4.22. Visão frontal do recipiente.

Na região posterior do recipiente foi feita uma abertura para alocar os suportes que dão sustento aos dispositivos internos (ver Figuras 4.23 e 4.24). Também foram esboçados cinco furos circulares para entrada dos parafusos que unirão o recipiente com a tampa traseira (ver Figuras 4.25 e 4.26).

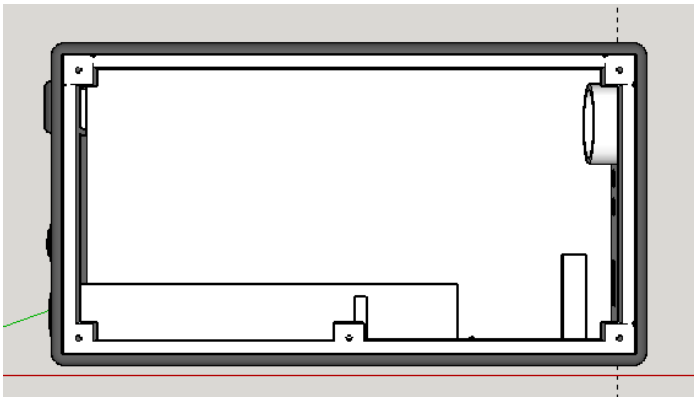

Figura 4.23. Visão frontal interna do recipiente.

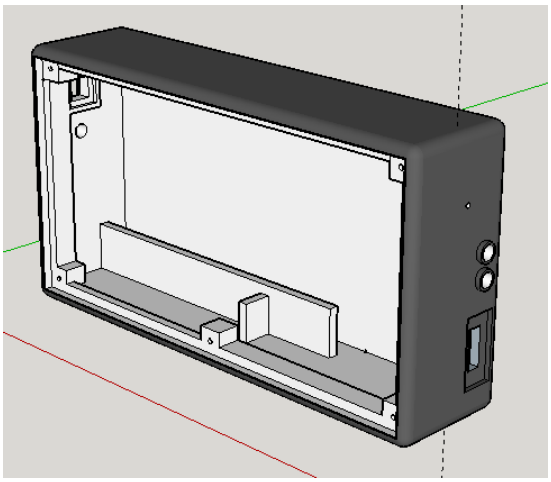

Figura 4.24. Visão superior interna do recipiente.

Por último, foi desenvolvido a tampa do recipiente. Foram esboçadas duas alças para prender o *gadget* ao cinto, de tamanho máximo de 4 cm. Cinco furos para os parafusos e um nivelamento na parte interna para encaixe mais preciso ao recipiente (ver Figura 4.25).

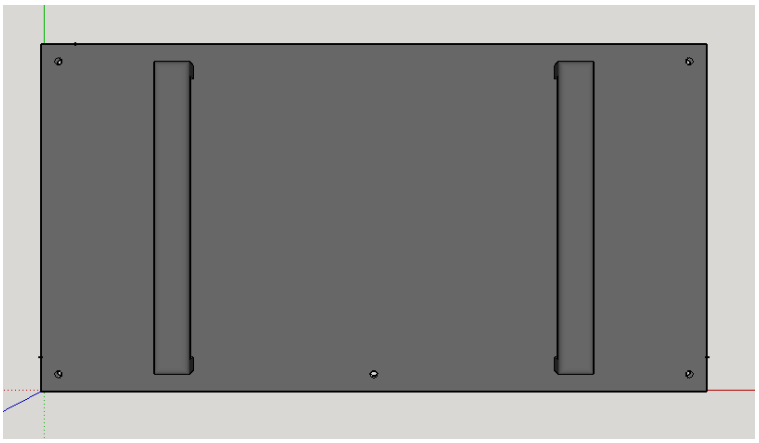

Figura 4.25. Visão traseira do recipiente.

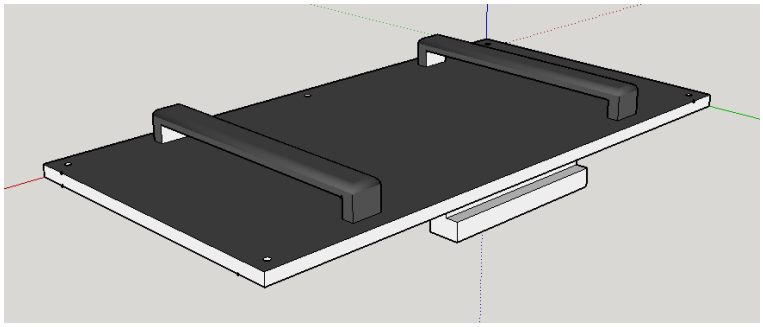

Figura 4.26. Visão superior traseira do recipiente.

4.5 Desenvolvimento do algoritmo de detecção de queda

Com a parte física do dispositivo montada, foi necessário desenvolver um algoritmo para a detecção de queda. O algoritmo desenvolvido teve como base o trabalho de Hsieh et al. (2017), em que foi desenvolvido um algoritmo de três fases de detecção, entre elas "*Free Fall Phase*", "*Impact Phase*" e "*Rest Phase*", conforme Figura 4.27.

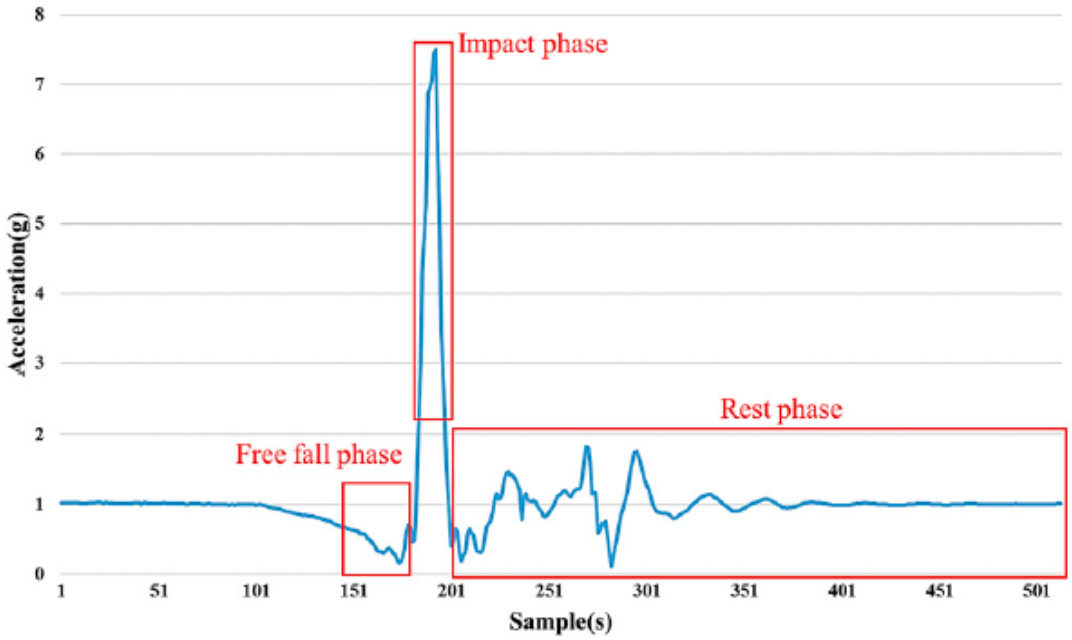

Figura 4.27. Gráfico da aceleração em função da quantidade de amostra, com indicação das três fases para detecção de queda utilizado em Hsieh et al. (2017).

Para obtenção da aceleração da gravidade (g), foi utilizada a norma euclidiana.
Dado $a = (a_1, \dots, a_n) \in \mathbb{R}^n$ definimos a norma euclidiana da seguinte forma:

$$|a| = \sqrt{a * a} = \sqrt{a_i^2 + \cdots + a_n^2} \qquad \text{(1) por Lomonaco (2017)}$$

A norma euclidiana associa um número real a cada vetor $a \in \mathbb{R}^n$ e podemos interpretar geometricamente $|a|$ como o 'comprimento' do vetor a (LOMONACO, 2017).

Adaptando a equação aos valores das acelerações fornecidas pelo acelerômetro nos eixos x, y e z, obtemos g tal que:

$$Norm_{xyz} = |g| = \sqrt{x^2 + y^2 + z^2} \qquad \text{(2) por Hsieh et al. (2017)}$$

Desta forma, $Norm_{xyz}$ é utilizada para descrever o módulo da variação espacial da aceleração durante a janela de queda (HSIEH et al., 2017).

Os valores das acelerações nos eixos x, y e z foram obtidos a uma taxa de amostragem de 128 Hz, analogamente $Norm_{xyz}$ também é obtido nessa mesma taxa. Segundo Hsieh et al. (2017) é mais que suficiente para a média de 100 Hz necessária para se amostrar os movimento do corpo humano.

Para analisar o comportamento do movimento, Hsieh et al. (2017) definiu uma janela, que pode ser observada na Figura 4.27, que conta com 513 amostras. Sabendo que a taxa de amostragem é de 128 Hz, então cada amostra tem duração de 7.812,5 µs. Sendo assim, o algoritmo irá processar 7.812,5 µs x 513 amostras, o que dá aproximadamente 4 s.

Dentro dessas 513 amostras, Hsieh et al. (2017) divide a janela da seguinte forma "a janela é determinada coletando 1,5 s (128 Hz x 1,5 s = 192 amostras) antes do ponto crítico e 2,5 s (128 Hz x 2,5 s = 320) amostras depois do ponto crítico". O ponto crítico que o autor se refere é a fase "*Impact Phase*", portanto antes do ponto crítico se refere a "*Free Fall Phase*" e depois do ponto crítico a "*Rest Phase*".

A primeira fase, que funciona como um gatilho para a verificação das outras etapas, é a "*Impact Phase*", ou fase de impacto, que é o momento em que o corpo se choca com o chão. Nesta fase é onde se encontra o pico do gráfico da Figura 4.27, ou seja, é onde a aceleração da gravidade alcança os maiores valores. Segundo os experimentos realizados por Hsieh et al. (2017), para casos de queda, a fase de impacto variou aproximadamente entre $3g$ e $10g$, dependendo da característica da pessoa e do tipo de queda. O autor não informou o *thereshold* utilizado nesta fase.

Em seguida temos a "*Free Fall Phase*", ou fase de queda livre. Esta fase ocorre imediatamente antes do corpo se chocar no chão, ou seja, é a queda propriamente dita. Durante esta etapa a aceleração da gravidade deve assumir valores inferiores a $1\ g$, tendendo a $0\ g$ por um determinado período de tempo, que para Hsieh et al. (2017) deve ser durante o 1,5 s que precedem a fase de impacto.

Por último, verifica-se a "*Rest Phase*", ou fase de descanso. Esta fase ocorre logo após a fase de impacto em que o corpo tende a ficar imóvel por um determinado período de tempo. Durante esta fase a aceleração da gravidade tende a se manter em aproximadamente $1g$.

O autor não apresenta muitas informações sobre as métricas e as técnicas utilizadas em cada fase para determinar se ocorreu ou não a queda, portanto, o algoritmo desenvolvido no presente trabalho não se fideliza, nas três etapas apresentadas, ao trabalho realizado por Hsieh et al. (2017).

A proposta de algoritmo realizada neste trabalho utiliza as três etapas propostas por Hsieh et al. (2017), porém foi acrescido uma quarta etapa. Esta quarta etapa mede a inclinação do corpo em relação a gravidade. A taxa de amostragem adotada também foi de 128 Hz, através de uma função interrupção que coleta as informações do eixo disponibilizadas pelo acelerômetro a cada 7.812,5 µs ($\tau = \frac{1}{f}$). O fluxograma completo do algoritmo desenvolvido neste trabalho pode ser observado na Figura 4.28. Todas as métricas e limites definidos nesse algoritmo foram desenvolvidos empiricamente, através de diversas simulações e testes que serão abordados mais à frente.

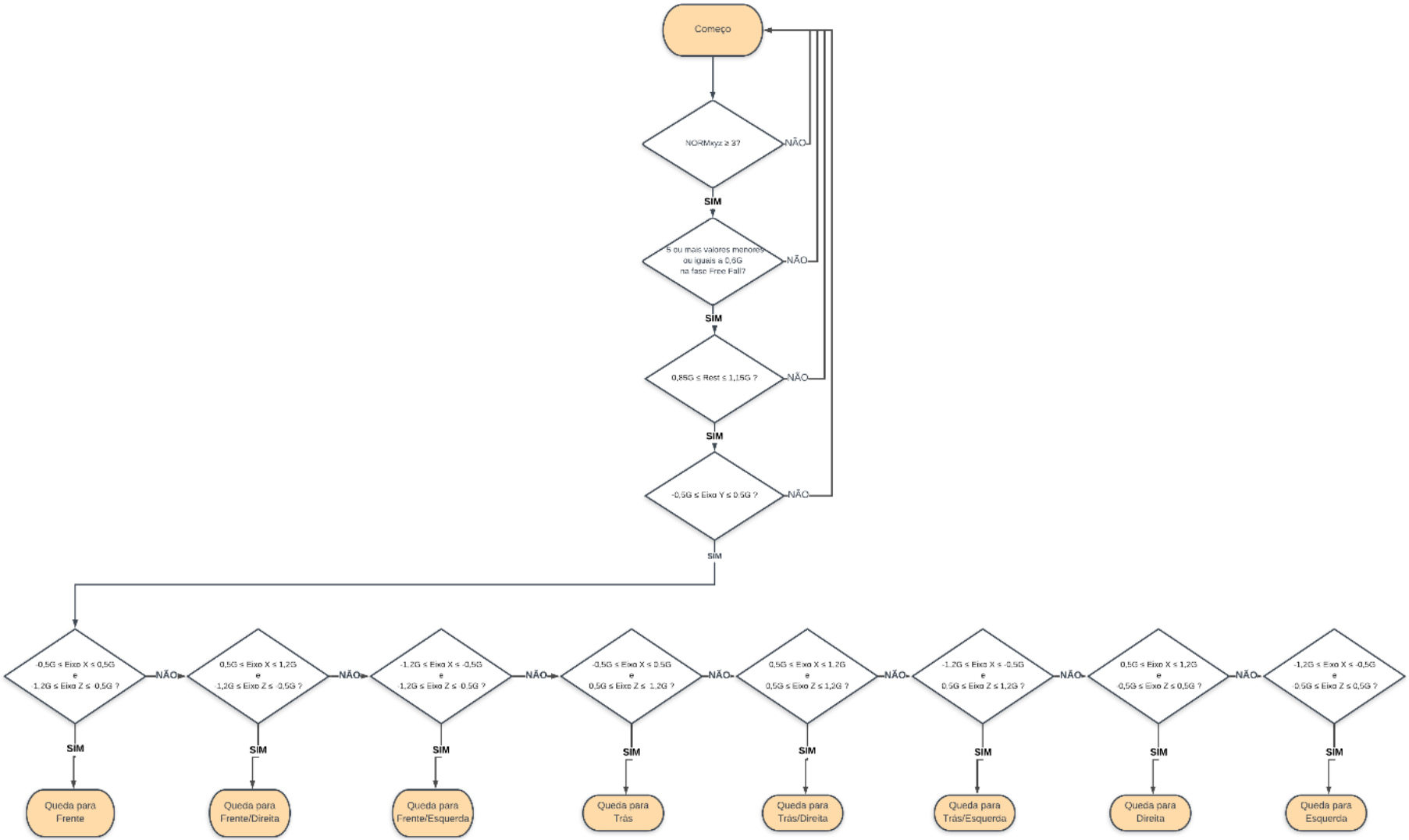

Figura 4.28. Fluxograma do algoritmo de detecção de queda desenvolvido

Assim como no trabalho de Hsieh et al. (2017), a primeira etapa e gatilho para o restante do algoritmo, é a fase de impacto, em que se definiu o *thereshold* do $Norm_{xyz}$ como maior ou igual a 3 g.

```
if ((NORMxyz >= 3) || (threshold == true)) {
  posicao = i;
  threshold = true;
  threshold_rest[k] = NORMxyz; // Recebe o valor do threshold de impacto e mais 320 valores NORMxyz apos o impacto
  eixoX[k] = xg; // Recebe os valores do eixo X após a queda, para verificar o angulo
  eixoY[k] = yg; // Recebe os valores do eixo y após a queda, para verificar o angulo
  eixoZ[k] = zg; // Recebe os valores do eixo Z após a queda, para verificar o angulo
  k++;
```

Figura 4.29. Trecho do código que verifica se a aceleração da gravidade é maior do que 3 *g*.

Sendo a aceleração maior ou igual a 3 g, a primeira fase informa que houve um indício de impacto, que pode ser uma queda, e deve ser confirmado pelas fases seguintes.

A próxima etapa é a fase de queda livre, em que se deve ter pelo menos 5 amostras, menores do que 0,6 g, dentre as amostras 91 até 192.

```
void verificar_fase_freefall() {
  int j;
  for (i = 91; i < 192; i++) {
    if (freefall[i] <= 0.6) {
      j++;
      if (j = 5) {
        verificar_rest_fase(); //Fase FreeFall OK, chama fase Rest
      }
    }
    else
      j = 0;
  }
  temporizador.attach(periodo, funcao_interrupcao); //Inicia a função interrupção

}
```

Figura 4.30. Trecho do código que verifica as condições da fase de queda livre.

Após a função "verificar_fase_free_fall()" detectar uma queda livre, deve-se analisar se o corpo da vítima se manteve em repouso. Para isso, tira-se a média das últimos 50 amostras e verifica se esta média está entre 1,15 g e 0,85 g.

```
void verificar_rest_fase() {
  for (i = 270; i < 321; i++) {// Somando todos os valores da posição 270 a 320 para a fase de descanço
    mediaRest_Fase = mediaRest_Fase + threshold_rest[i];
  }
  mediaRest_Fase = mediaRest_Fase / 50; // Obtendo a media em G do descanço

  if (((mediaRest_Fase) <= 1.15) && ((mediaRest_Fase) >= 0.85)) { // Se a média estiver entre 0.85G e 1.1G,
    verificar_eixos(); //Fase Rest OK, chama fase Eixos
  }

  else {
    temporizador.attach(periodo, funcao_interrupcao); //Inicia a função interrupção
  }

}
```

Figura 4.31. Trecho do código que verifica se as condições da fase de descanso.

Por último, analísa-se a inclinação do corpo em relação a gravidade. Esta fase é o principal diferencial do presente trabalho, pois não foram encontradas bibliografias de detecção de queda com acelerômetro utilizando desta abordagem.

A posição de vestimenta em relação a cintura do usuário e a direção dos eixos do sensor podem ser observadas na Fígura 4.32.

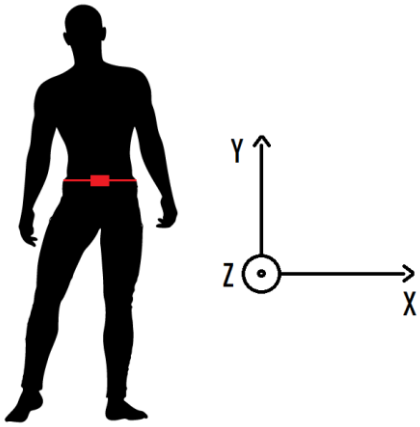

Figura 4.32. Localização do dispositivo no corpo e as direções dos eixos do sensor acelerômetro.

Desta forma, uma pessoa em pé, pulando, correndo ou sentada, mantem a aceleração da gravidade no eixo y em aproximadamente 1 g, ou seja, o eixo y está ortogonal ao plano do chão. Quando uma queda real ocorre, o corpo da vítima fica estendido ao chão, portanto o eixo y fica paralelo ao chão e a aceleração da gravidade deste eixo tende a 0 g.

Sendo assim, foram calculadas as médias dos últimos 50 valores dos eixos x, y e z. Caso a média de y ficar entre -0,5 g e 0,5 g, então finalmente o algoritmo afirmaria que ocorreu uma queda.

```
void verificar_eixos() {
  for (i = 270; i < 321; i++) { // Somando todos os valores da posição 130 a 321 dos eixos X, Y e Z
     mediaX = mediaX + eixoX[i];
     mediaY = mediaY + eixoY[i];
     mediaZ = mediaZ + eixoZ[i];
  }
  mediaX = mediaX / 50; // Obtendo a media em G dos eixo X, Y e Z
  mediaY = mediaY / 50;
  mediaZ = mediaZ / 50;

  if (((mediaY) <= 0.5) && ((mediaY) >= -0.5)) {
```

Figura 4.33. Trecho do código que verifica se o eixo *y* está paralelo ao plano do chão.

A inclinação do corpo em relação a gravidade, também foi capaz de proporcionar a informação do lado aproximado de queda analisando as médias dos eixos x e z conforme Tabela 4.1.

Tabela 4.1. Intervalo das médias dos eixos *x* e *z* para cada lado de queda.

Lado aproximado da queda	**Eixo X**	**Eixo Z**
Queda para Frente	-0,5 ≤ Média X ≤ 0,5	-1,2 ≤ Média Z ≤ -0,5
Queda para Frente/Direita	0,5 ≤ Média X ≤ 1,2	-1,2 ≤ Média Z ≤ -0,5
Queda para Frente/Esquerda	-1,2 ≤ Média X ≤ -0,5	-1,2 ≤ Média Z ≤ -0,5
Queda para Trás	-0,5 ≤ Média X ≤ 0,5	0,5 ≤ Média Z ≤ 1,2
Queda para Trás/Direita	0,5 ≤ Média X ≤ 1,2	0,5 ≤ Média Z ≤ 1,2
Queda para Trás/Esquerda	-1,2 ≤ Média X ≤ -0,5	0,5 ≤ Média Z ≤ 1,2
Queda para Direita	0,5 ≤ Média X ≤ 1,2	-0,5 ≤ Média Z ≤ 0,5
Queda para Esquerda	-1,2 ≤ Média X ≤ -0,5	-0,5 ≤ Média Z ≤ 0,5

O algoritmo então, após verificar que o eixo y está próximo de 0 g (entre -0,5 g e 0,5 g), ou seja, passou da última etapa de detecção de queda, ainda é capaz de informar para que lado a vítima sofreu a queda.

```
if ((((mediaX) <= 0.5) && ((mediaX) >= -0.5)) && (((mediaZ) <= -0.5) && ((mediaZ) >= -1.2))) {
  lado_queda = 1; // Queda para Frente
  imprimir();
}
else if ((((mediaX) <= 1.2) && ((mediaX) >= 0.5)) && (((mediaZ) <= -0.5) && ((mediaZ) >= -1.2))) {
  lado_queda = 2; // Queda para Frente/Direita
  imprimir();
}
else if ((((mediaX) <= -0.5) && ((mediaX) >= -1.2)) && (((mediaZ) <= -0.5) && ((mediaZ) >= -1.2))) {
  lado_queda = 3; // Queda para Frente/Esquerda
  imprimir();
}
else if ((((mediaX) <= 0.5) && ((mediaX) >= -0.5)) && (((mediaZ) <= 1.2) && ((mediaZ) >= 0.5))) {
  lado_queda = 4; // Queda para Tras
  imprimir();
}
else if ((((mediaX) <= 1.2) && ((mediaX) >= 0.5)) && (((mediaZ) <= 1.2) && ((mediaZ) >= 0.5))) {
  lado_queda = 5; // Queda para Tras/Direita
  imprimir();
}
else if ((((mediaX) <= -0.5) && ((mediaX) >= -1.2)) && (((mediaZ) <= 1.2) && ((mediaZ) >= 0.5))) {
  lado_queda = 6; // Queda para Tras/Esquerda
  imprimir();
}
else if ((((mediaX) <= 1.2) && ((mediaX) >= 0.5)) && (((mediaZ) <= 0.5) && ((mediaZ) >= -0.5))) {
  lado_queda = 7; // Queda para Direita
  imprimir();
}
else if ((((mediaX) <= -0.5) && ((mediaX) >= -1.2)) && (((mediaZ) <= 0.5) && ((mediaZ) >= -0.5))) {
  lado_queda = 8; // Queda para Esquerda
  imprimir();
```

Figura 4.34. Trecho do código com o intervalo das médias dos eixos *x* e *z* para cada lado de queda.

Durante todas as etapas, nos casos em que os valores de $Norm_{xyz}$ e das médias de x, y e z não cumprirem os limiares de cada fase, então o algoritmo retorna a verificação do impacto, ou seja, retorna a primeira fase.

A Figura 4.35 mostra uma janela que foi processada pelo algoritmo, nela está representado a aceleração da gravidade de $Norm_{xyz}$ em g (eixo x do gráfico) em função da quantidade de amostras coletadas (eixo y do gráfico), bem como as três primeiras fases da detecção destacadas de vermelho. Este foi um exemplo de queda para frente ao tentar se levantar de uma cadeira, isso pode-se confirmar pelas médias dos eixos x, y e z, que foram 0,14 g, 0 g e -1,054 g respectivamente.

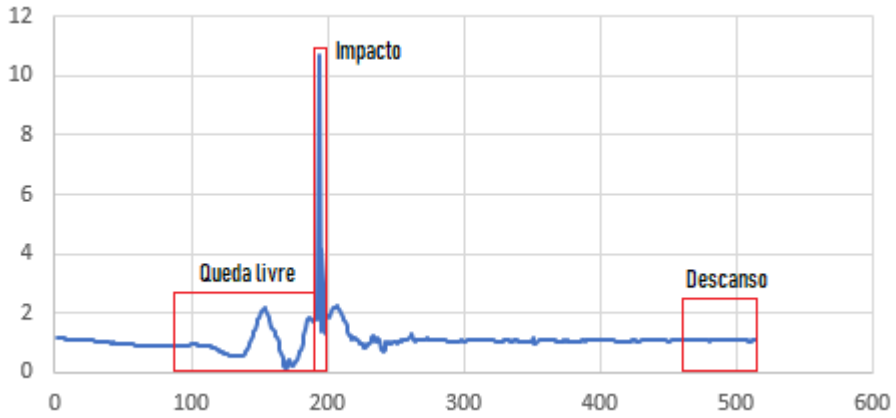

Figura 4.35. Exemplo de uma janela processada pelo algoritmo (g x amostras).

5 METODOLOGIA PARA AJUSTES DE PARÂMETROS E MÉTRICAS

Este capítulo apresenta os métodos utilizados para testar a eficiência do algoritmo de detecção de queda proposto bem como a análise dos testes para encontrar os melhores parâmetros para as 4 fases do algoritmo, a fim de maximizar a detecção de quedas verdadeiras e minimizar a detecção de queda em atividades do dia a dia.

5.1 Simulações

Para encontrar as melhores parâmetros a serem utilizadas nas 4 fases do algoritmo de detecção de queda (impacto com o chão, queda livre, descanso e a inclinação do corpo em relação a gravidade) foram realizadas 9 atividades que finalizavam em quedas conforme Tabela 5.1 e 11 atividades do dia a dia que não envolviam queda conforme Tabela 5.2. Convocou-se 8 pessoas para a realização dos testes. Para fins de segurança, foi convocado apenas pessoas jovens e saudáveis, de 18 a 25 anos, 7 homens e 1 mulher. Desta forma, foram simulados 72 quedas e 88 atividades do dia a dia.

Tabela 5.1. Atividade que geram de queda.

Queda	Quantidade de Testes
Cair para trás tentando sentar	8
Ficar parado e cair para a direita	8
Ficar parado e cair para a esquerda	8
Ficar parado e cair para frente	8
Ficar parado e cair para trás	8
Levantar da cadeira e cair para frente	8
Inclinar o corpo e cair	8
Andar e cair para frente	8
Andar e cair para trás	8
Total	72

Tabela 5.2. Atividades do dia a dia.

Atividades do Dia a Dia	Quantidade de Testes
Pular no chão	8
Deitar rápido	8
Deitar normal	8

Sentar rápido	8
Sentar normal	8
Andar rápido	8
Andar normal	8
Levantar depois de agachado	8
Levantar depois de sentado	8
Subir escadas	8
Descer escadas	8
Total	88

O dispositivo detector de queda foi preso à cintura utilizando um cinto de tal forma que o eixo y do sensor estivesse ortogonal ao chão, conforme a Figura 5.1.

Figura 5.1. Dispositivo encaixado no cinto e alocado na cintura.

Para que não houvesse perigo de acidentes durante as simulações de queda, foi necessário a utilização de um tatame de 2 cm de espessura em cima de um colchonete. Os dados foram coletados através de uma conexão USB do microcontrolador com o computador a fim de recebê-los pelo “Monitor Serial” da IDE do Arduino.

Figura 5.2. Cenário de simulação de queda em tatame e colchonete.

Foram montados gráficos para todas as simulações realizadas, bem como exemplos das quedas e atividades do dia a dia disponíveis no banco de dados em Vieira e Araújo (2019). Dois exemplos de gráficos, um para queda e um para atividade do dia a dia podem ser observados nas Figuras 5.3 e 5.4.

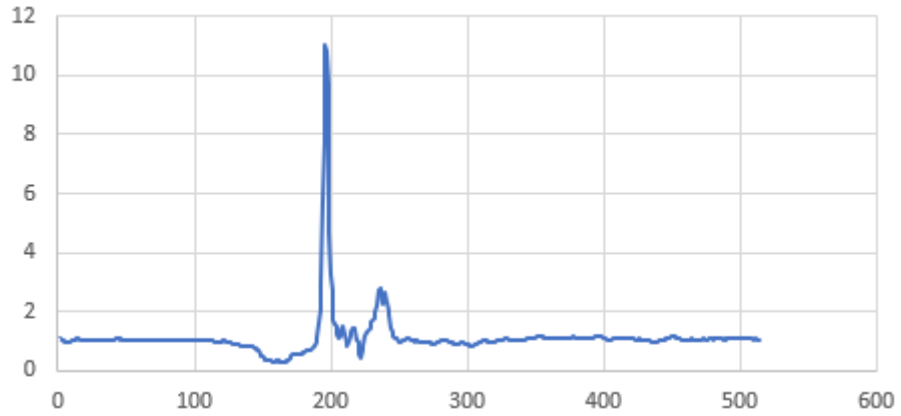

Figura 5.3. Exemplo de gráfico de queda na modalidade "Ficar parado e cair para a direita" ($Norm_{xyz}\,(g)$ x Amostras)

Figura 5.4. Exemplo de gráfico de atividade do dia a dia na modalidade "Sentar rápido" ($Norm_{xyz}$ x Amostras).

A partir dos gráficos gerados, deu-se início a análise com o intuito de encontrar os parâmetros que mais detectassem quedas para as 4 fases do algoritmo.

5.2 Análise das simulações

Após a realização de todas as simulações com os 8 colaboradores, foram coletados os dados das acelerações da gravidade $Norm_{xyz}$ e dos eixos x, y e z a fim de se plotar gráficos e a partir desses gráficos apontar as melhores métricas a serem utilizadas no algoritmo. Com isso, deseja-se a maximização da detecção de quedas verdadeiras e minimização de falsos positivos.

A primeira etapa do algoritmo é a fase de impacto do corpo com o chão, que tende a gerar valores de $Norm_{xyz}$ mais elevados. Nos testes realizados conforme Tabela 5.1, a modalidade "Cair para trás tentando sentar" foi a que forneceu menores valores para $Norm_{xyz}$. O menor valor encontrado para essa modalidade foi 3,18 g, como é possível observar na Figura 5.2.

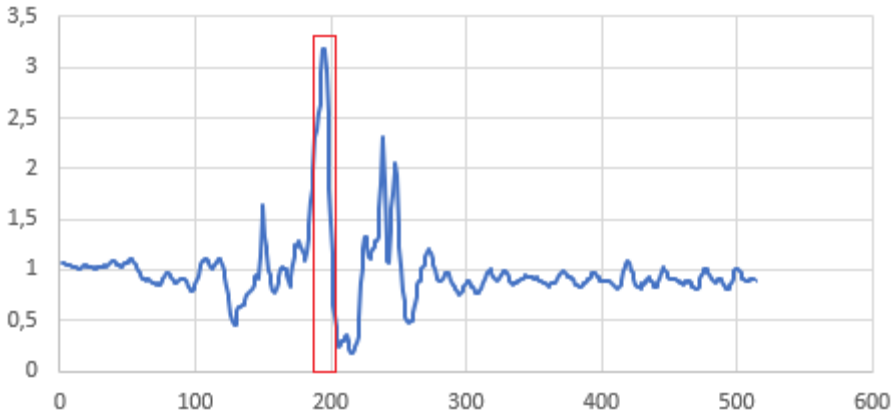

Figura 5.5. $Norm_{xyz}$ (g) em função da quantidade de amostras para a modalidade "Cair para trás tentando sentar" de um colaborador.

Este tipo de queda (Cair para trás tentando sentar) tende a gerar valores de aceleração da gravidade baixos pois a pessoa que sofreu a queda estará mais próxima do chão antes do impacto. Segundo Watabe et al. (2015) este é um tipo de queda bastante recorrente em pessoas idosas, portanto sua detecção correta é crucial para um sistema de detecção de queda para o público idoso. Sendo assim, foi definido um *thereshold* de 3 g para $Norm_{xyz}$ na fase de impacto. A Figura 5.6 apresenta um gráfico com os máximos e mínimos obtidos para cada modalidade de queda.

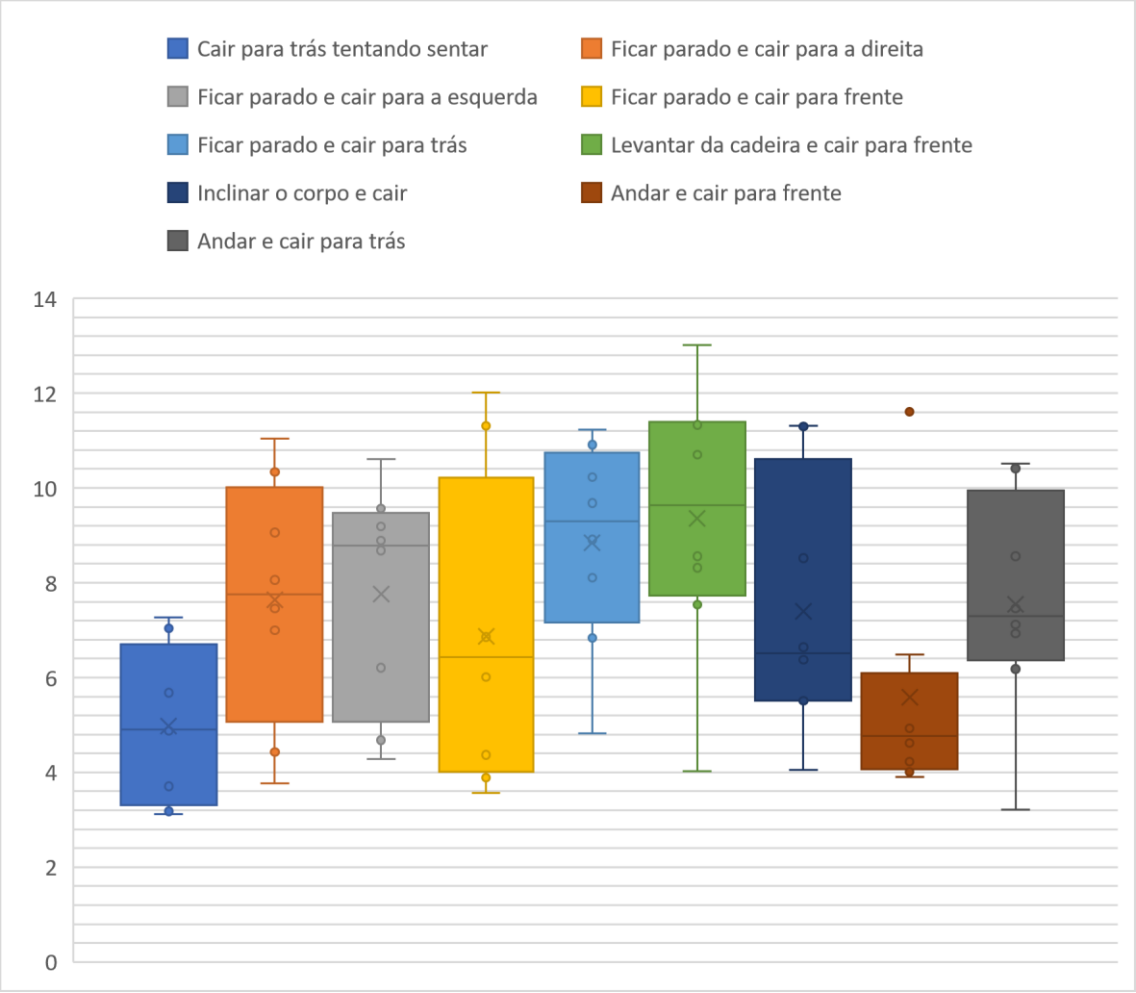

Figura 5.6. Máximos e minimos para Normxyz (g) por modalidade de queda.

A segunda etapa do algoritmo é a fase de queda livre, cronologicamente anterior ao impacto do corpo no chão. Foi observado pelos gráficos que sempre que uma queda ocorria, o valor de $Norm_{xyz}$ ficava por um determinado período de tempo abaixo de 0,6 g. A amplitude temporal variou entre 5 a 34 amostras, sabendo que a taxa de amostragem é de 128 Hz e que $\tau = \frac{1}{f}$, então cada amostra tem um tempo de duração de aproximadamente 7.812,5 µs, sendo assim, foram detectado quedas com 39.062,5 µs até 273,4375 ms (aproximadamente 0,27 s) de queda livre abaixo de 0,6 g.

Analisando o gráfico de setores da Figura 5.3 é fácil observar que a maioria das quedas (53%) tem um tempo de queda livre menor que 10 amostras, ou 78.125 µs. Vale notar também que em nenhuma das quedas simuladas, observou-se um tempo de queda livre com $Norm_{xyz} \leq 0{,}6\ g$ menor que 5 amostras (39.062,5 µs).

Desta forma, as métricas que maximizam a detecção de queda na fase de queda livre são os *thereshold*: $Norm_{xyz} \leq 0{,}6\ g$ e tempo de queda maior que 5 amostras (39.062,5 µs).

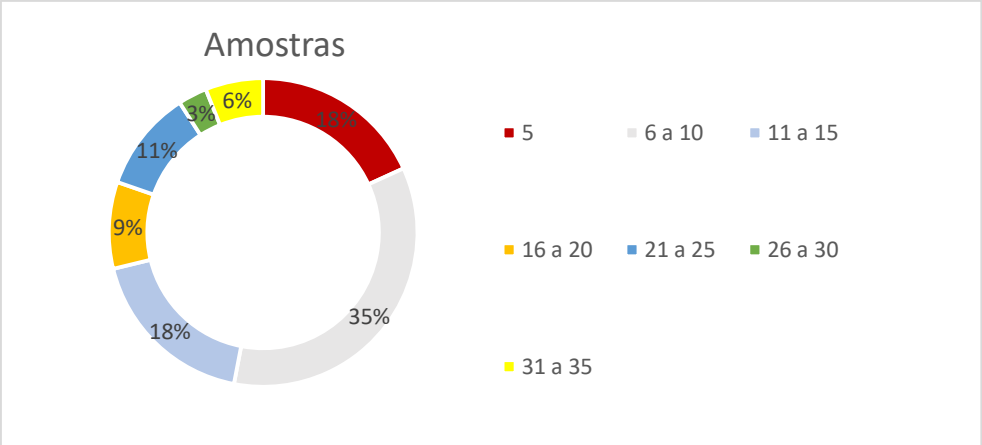

Figura 5.7. Gráfico de setores que informa a quantidade de quedas de acordo com o tamanho do intervalo da fase de queda livre.

A terceira etapa do algoritmo é a fase de descanso, cronologicamente posterior ao impacto do corpo no chão. Na análise dos gráficos dessa fase, foi observado que em alguns casos o corpo demora a se estabilizar no chão e gera flutuações aleatórias na aceleração da gravidade em $Norm_{xyz}$, como pode ser observado na Figura 5.8 na marcação em vermelho.

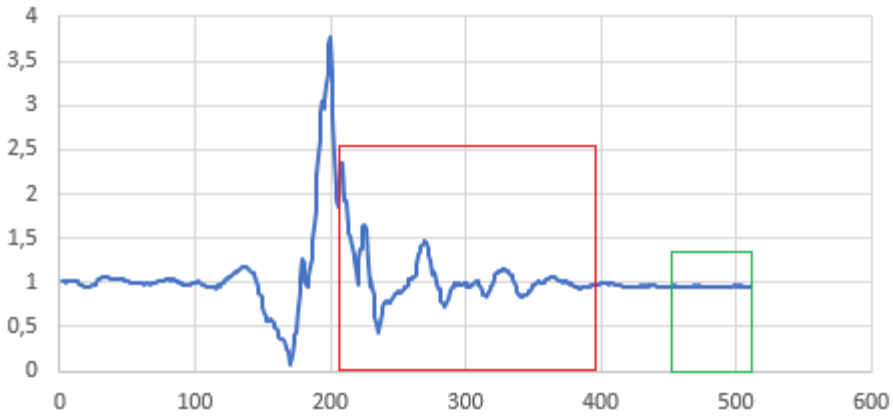

Figura 5.8. $Norm_{xyz}\,(g)$ em função da quantidade de amostras para a modalidade "Ficar parado e cair para a direita" de um colaborador.

Sendo assim, o processamento apenas dos valores finais, destacados em verde na Figura 5.8 garante que o corpo já tenha se estabilizado no chão e não gere grandes flutuações, que comprometeriam o funcionamento do algoritmo.

Portanto, em todas as simulações realizadas, as últimas 50 amostras de $Norm_{xyz}$ se mostraram estáveis e, então, qualificadas para serem processadas pelo algoritmo. Para se ter o valor de $Norm_{xyz}$ aproximado para essa etapa, foi coletada a média dos últimos 50 valores, bem como os valores máximos e mínimos encontrados nas médias de cada queda simulada, conforme Tabela 5.3.

Tabela 5.3. Média, maior valor e menor valor da $Norm_{xyz}$ dentre todas as quedas para a fase de descanso.

Média (g)	**Maior valor (g)**	**Menor valor (g)**
0,987175	1,1374	0,8522

.

Por fim, foi definido o *threshold* para a fase de queda livre, em que: $0{,}85\ g \leq Norm_{xyz} \leq 1{,}15\ g$.

A última fase do algoritmo mede a inclinação do corpo em relação a gravidade por meio das acelerações da gravidade individuais dos eixos x, y e z.

Assim como no caso da fase de descanso, foram utilizados apenas as médias das últimas 50 amostras de cada eixo, pois é necessário que o corpo se estabilize no chão para diminuir os ruídos. Como apenas o eixo y está ortogonal ao chão quando a cintura está ereta, conforme Figura 4.32, este eixo é suficiente para informar, nesta etapa, se ocorreu ou não uma queda.

Sabe-se que quando o corpo está ereto, ou seja, o eixo y ortogonal ao chão, a aceleração da gravidade nesse eixo é matematicamente próxima de 1 g. Desta forma, quando ocorre uma queda e o eixo y se torna paralelo ao chão a aceleração da gravidade se aproxima de 0 g.

Analisando os gráficos para as detecções de queda com sucesso, obteve-se as seguintes informações contidas na Tabela 5.4.

Tabela 5.4. Média das médias (50 últimos valores do eixo *y*), maior valor dentre as médias de *y* e menor valor dentre as médias de *y* para todas as quedas simuladas.

Média das médias de *y* (g)	**Maior valor dentre as médias de *y* (g)**	**Menor valor dentre as médias de *y* (g)**
0,01621	0,435	-0,478

Desta forma, foi definido o *threshold* para a fase de inclinação do corpo em relação à média do eixo *y*, em que: $-0{,}5\ g \leq Média\ y \leq\ 0{,}5\ g$.

Ainda na fase de inclinação do corpo em relação a gravidade, os eixos x e z são capazes de informar o lado em que o usuário do dispositivo sofreu a queda, conforme Tabela 4.1. O *threshold* para cada posição de queda (através dos eixos *x* e *z*) foi obtido através da mesma metodologia em que se encontrou o *threshold* para a média do eixo *y*, observado os maiores e menores valores para cada caso.

6 RESULTADOS

Este capítulo apresenta os resultados da qualidade do sistema de detecção de queda obtidos através dos testes e simulações realizados.

Para o resultado final do dispositivo, foi realizada uma bateria de testes, com o total de 8 pessoas de diferentes tamanhos e pesos, realizando diversas atividades conforme já descrito anteriormente e mostradas na Tabela 6.1 e na Tabela 6.2. Os testes foram separados em dois grupos: quedas e atividades do dia a dia. Os testes de quedas foram realizados em 9 atividades diferentes, visualizadas na Tabela 6.1. Para o grupo de atividades do dia a dia os testes foram realizados em 11 tipos de atividades diferentes, visualizadas na Tabela 6.2. As tabelas a seguir informam a quantidade de testes corretos, falsos negativos e falsos positivos.

Tabela 6.1. Atividade que geram de queda.

Queda	Quantidade de Testes	Corretos	Falsos Negativos
Cair para trás tentando sentar	8	7	1
Ficar parado e cair para a direita	8	7	1
Ficar parado e cair para a esquerda	8	7	1
Ficar parado e cair para frente	8	8	0
Ficar parado e cair para trás	8	8	0
Levantar da cadeira e cair para frente	8	8	0
Inclinar o corpo e cair	8	7	1
Andar e cair para frente	8	8	0
Andar e cair para trás	8	8	1
Total	72	68	5

Tabela 6.2. Atividades do dia a dia.

Atividades do Dia a Dia	Quantidade de Testes	Corretos	Falsos Positivos
Pular no chão	8	8	0
Deitar rápido	8	5	3
Deitar normal	8	8	0
Sentar rápido	8	8	0
Sentar normal	8	8	0
Andar rápido	8	8	0

Andar normal	8	8	0
Levantar depois de agachado	8	8	0
Levantar depois de sentado	8	8	0
Subir escadas	8	8	0
Descer escadas	8	8	0
Total	88	85	3

Com os resultados obtidos dos testes, foram calculados os seguintes parâmetros do sistema de detecção de queda: a sensitividade (capacidade de detectar queda), especificidade (taxa de verdadeiros negativo), precisão (qualidade na detecção de quedas exatas) e a acurácia (proporção de testes corretos dentro do total).

Segundo Hsieh et al. (2017) muitos estudos adotam esses critérios para exibir os resultados de técnicas de detecção de quedas. Esses parâmetros podem ser visualizados na Tabela 6.3 e seus cálculos se dão através das seguintes equações:

- Sensitividade = $\frac{TP}{TP + FN}$ (3) por Hsieh et al. (2017)
- Especificidade = $\frac{TN}{FP + TN}$ (4) por Hsieh et al. (2017)
- Precisão = $\frac{TP}{TP + FP}$ (5) por Hsieh et al. (2017)
- Acurácia = $\frac{TP + TN}{TP + FP+TN+FN}$ (6) por Hsieh et al. (2017)

O *TP* é o total de detecção correta de queda, *FN* é o total de vezes em que se caiu e não identificou queda (falsos negativo), *TN* é o total de atividades do dia a dia que não foram detectado quedas e *FP* é o total de atividades do dia a dia em que se detectou queda (falsos positivos).

Tabela 6.3. Resultados finais do sistema de detecção de queda desenvolvido.

Parâmetro	**Resultado**
Sensitividade	93.15%
Especificidade	96.59%
Precisão	95.77%
Acurácia	95.03%

Como observado na Tabela 6.3, o sistema possui todos os parâmetros superiores a 93%, sendo que a acurácia e a precisão são maiores que 95%. Como os parâmetros indicam, os resultados foram bastante satisfatórios. A grande melhoria percebida no sistema desenvolvido neste trabalho é em relação a taxa de falsos positivos, que neste caso foi 3,41% contra 5,52% quando comparado ao proposto por Hsieh et al. (2017), que foi utilizado como base. Esta melhora é explicada principalmente pela etapa de inclinação do corpo em relação a gravidade, que se mostrou bastante eficaz para algoritmo concluir que não houve quedas em atividades do dia a dia, com exceção atividades de deitar rápido.

Apesar da abordagem utilizando a inclinação do corpo em relação a gravidade ter eliminado a maioria dos falsos positivos nas atividades do dia a dia, a atividade de deitar rápido ainda foi confundida

com queda em 3 dos 5 testes devido as ações de cair e deitar-se rápido gerarem janelas de aceleração da gravidade para $Norm_{xyz}$ similares e média do eixo y próxima de 0 g, pois em ambos os casos o corpo estará paralelo ao chão. Podemos observar a semelhança nos gráficos de aceleração da gravidade em função da quantidade de amostras de ambos os casos nas Figuras 6.1 e 6.2.

Figura 6.1. Gráfico da aceleração da gravidade (g) em função da quantidade de amostras para o caso de queda para trás.

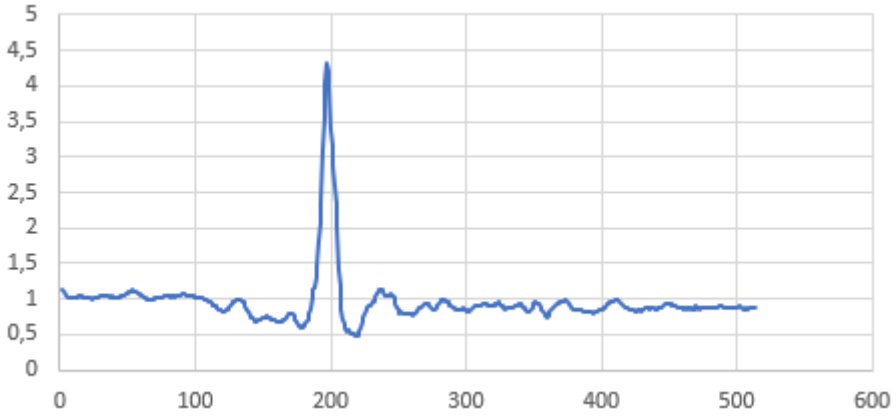

Figura 6.2. Gráfico da aceleração da gravidade (g) em função da quantidade de amostras para o caso de deitar-se rápido.

Não foram encontradas, nas bibliografias, soluções capazes de diferenciar efetivamente quedas reais de atividades como deitar-se rapidamente. Entretanto, as simulações realizadas ao deitar-se rápido foram testadas exagerando a ação, a fim de se testar os limites do sistema. Este tipo de ação não é comumente praticável pela população alvo deste dispositivo, as pessoas idosas, que em geral tem seus movimentos suaves e limitados, ou seja, tendem a deitar-se mais lentamente, sendo que neste caso, nenhum falso positivo foi detectado.

7 CONCLUSÃO

O projeto aqui desenvolvido, teve como principal meta a elaboração de um dispositivo capaz de detectar e alertar quedas a fim de oferecer maior liberdade e mobilidade ao idoso, bem como um maior conforto aos familiares por mantê-los informados em casos de queda, melhorando assim a qualidade de vida das pessoas envolvidas.

O dispositivo desenvolvido alcançou excelentes resultados nas simulações analisadas dentro desse trabalho. Sobre os resultados, é importante ressaltar o seguinte fato: os 3 únicos erros encontrados durante os testes das atividades do dia a dia, foram contabilizados como falsos positivos, mas a análise dos gráficos (e do experimento) mostra que de fato, as 3 pessoas sofreram uma queda, menos intensa e atenuada pelo sofá, e ainda sim o sensor foi capaz detectá-las, isso se deve ao fato de que a simulação da modalidade "deitar rápido" foi realizada pelas cobaias com intensidades exageradas se comparadas aos movimentos limitados do público alvo, os idosos.

Também foram realizados alguns testes alterando-se rapidamente os eixos do sensor: um dos participantes do experimento realizou um movimento de rotação no ar, "mortal para frente", na linguagem popular, e mesmo com essa brusca mudança nos eixos do acelerômetro, o dispositivo ainda foi capaz de detectar a queda.

A proposta inicial era seguir fielmente o algoritmo de 3 fases proposto por Hsieh et al. (2017), verificar se os resultados informados eram verídicos e a partir disso propor ajustes e contribuições de melhoria. Todavia, o trabalho de Hsieh et al. (2017) não disponibiliza detalhes dos parâmetros e métricas utilizados no algoritmo, o que inviabilizou a reprodução exata da proposta. Desta forma, foi aproveitado apenas o conceito das três fases: queda livre, impacto e descanso, sendo as métricas e parâmetros definidos empiricamente. Além disso, neste trabalho, foi adicionado mais uma fase como forma de contribuição: a inclinação do corpo em relação a gravidade, que ajudou a diminuir a taxa de falsos positivos.

O sistema desenvolvido ainda tem pontos a evoluir para viabilizar uma versão comercial. O primeiro ponto está no aplicativo, desenvolvido apenas para plataforma Android, necessita de aprimoramentos para tratar erros do usuário durante as configurações. O segundo ponto é no que diz respeito ao armazenamento das configurações realizadas pelo aplicativo, como e-mail, nome do usuário e rede, que ficam salvas como variável em uma memória volátil, portanto são apagadas ao reiniciar o dispositivo. Uma solução é o armazenamento dessas informações em uma memória EEPROM não-volatil. O último ponto, como os componentes foram conectados manualmente por meio de soldas, faz com que o circuito tome dimensões maiores, o ideal seria a elaboração do circuito em uma placa de circuito impresso, que deixaria o dispositivo mais compacto e ergonômico.

Por fim, uma ideia de aprimoramento, seria integrar um GPS ao dispositivo para este ser capaz de informar também a localização da pessoa que caiu, tornando o socorro mais efetivo e ampliando a cobertura de segurança.

REFERÊNCIAS BIBLIOGRÁFICAS

ABDUL-QAWY, Antar Shaddad *et al*. **The Internet of Things (IoT): An Overview**. Journal of Engineering Research and Applications, Kakatiya University, Warangal, India, p. 71-82, 1 dez. 2015.

ALBARBAR, Alhussein *et al*. **Suitability of MEMS Accelerometers for Condition Monitoring: An experimental study**. Sensors, University of Manchester, Manchester, p. 784-799, 30 nov. 2007.

ANALOG DEVICES. **ADXL345**. [S. l.: s. n.], 2009. Disponível em: https://www.analog.com/media/en/technical-documentation/data-sheets/ADXL345.pdf. Acesso em: 11 jun. 2019.

ANDREJAŠIC, Matej. **Mems Accelerometers**. 2008. Seminário (Física) - Estudante, University of Ljubljana, 2008. Disponível em: http://mafija.fmf.uni-lj.si/seminar/files/2007_2008/MEMS_accelerometers-koncna.pdf. Acesso em: 11 jun. 2019.

ARMSTRONG, Janell; HELPS, C. Richard. **Comparative evaluation of ZigBee and Bluetooth: Embedded wireless network technologies for students and designers**. 2007. Artigo (Graduação) - Estudante, Brigham Young University, 2007.

BAO, Minhang. **Analysis and Design Principles of MEMS Devices**. Estados Unidos: Elsevier Science, 2005.

BOTTERMAN, M. **Internet of Things: an early reality of the Future Internet**. *In*: WORKSHOP REPORT, EUROPEAN COMMISSION, 2009, Praga, República Checa.

DA ROCHA, Fábio Saraiva. **Acelerômetro eletrônico e a placa Arduino para ensino de Física em tempo real.** Cad. Bras. Ens. Fís., Universidade Federal do Pampa, Bagé, Rio Grande do Sul, 1 abr. 2013.

DOS SANTOS, Eduardo de Oliveira; ALVES, Pedro Souza. **Estudo e desenvolvimento de aplicativo para detecção de proximidade para dispositivos BLE**. 2018. Projeto Final (Graduação) - Estudante, Brasília, 2018.

ESPRESSIF SYSTEMS. **ESP32 Series Datasheet**. [S. l.: s. n.], 2019. Disponível em: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf. Acesso em: 12 jun. 2019.

FILIPEFLOP. **Módulo Carregador de Bateria de Lítio TP4056**. [S. l.]. Disponível em: https://www.filipeflop.com/produto/modulo-carregador-de-bateria-de-litio-tp4056/. Acesso em: 16 jun. 2019.

HOSNI, Shamsaa Hilal Al. **Bluetooth Low Energy: A Survey**. International Journal of Computer Applications , Sohar University, 1 mar. 2017.

HSIEH, Chia-Yeh et al. **Novel Hierarchical Fall Detection Algorithm Using a Multiphase Fall Model**. Sensors, Taipei, Taiwan, p. 1-11, 8 fev. 2017.

INSTRUCTABLES CIRCUITS. **NodeMCU-32S - Pinout**. [S. l.]. Disponível em: https://www.instructables.com/id/ESP32-Remote-Control-With-Sockets/. Acesso em: 16 jun. 2019.

KARLSSON ROBOTICS. **Triple Axis Accelerometer - ADXL345**. [S. l.], 18 jun. 2019. Disponível em: https://www.kr4.us/triple-axis-accelerometer-adxl345.html. Acesso em: 16 jun. 2019. Citações:

KARMAKAR, Raja; CHATTOPADHYAY, Samiran; CHAKRABORTY, Sandip. **Impact of IEEE 802.11n/ac PHY/MAC High Throughput Enhancements on Transport and Application Protocols—A Survey**. IEEE Communications Surveys & Tutorials, [S. l.], p. 2050 - 2091, 25 ago. 2017.

KAU, Lih-Jen; CHEN, Chih-Sheng. **A Smart Phone-Based Pocket Fall Accident Detection, Positioning, and Rescue System**. IEEE Journal of Biomedical and Health Informatics, [*S. l.*], p. 44 - 56, 4 jun. 2014.

KWOLEK, Bogdan; KEPSKI, Michal. **Human fall detection on embedded platform using depth maps and wireless accelerometer**. Comput. Methods Programs Biomed, Krakow, Polonia, p. 489-501, 3 dez. 2014.

LOMONACO, Luciana Luna Anna. **Cálculo Diferencial e Integral V / Cálculo Diferencial**. [S. l.], 4 ago. 2017. Disponível em: https://www.ime.usp.br/~thiagoap/monitorias/2017_2_calculo5luna/aula02.pdf. Acesso em: 13 jun. 2019.

MA, Hua-Dong. **Internet of Things: Objectives and Scientific Challenges**. Journal of Computer Science and Technology**,** Beijing University of Posts and Telecommunications, Beijing, China, p. 919–924, 5 set. 2011.

MASTORAKIS, G.; MAKRIS, D. **Fall detection system using Kinect's infrared sensor**. Journal of Real-Time Image Processing, Surrey, Reino Unido, p. 635–646, 4 dez. 2014.

MECANICA INDUSTRIAL. **O que é um acelerômetro capacitivo**. [S. l.]. Disponível em: https://www.mecanicaindustrial.com.br/136-o-que-e-um-acelerometro-capacitivo/. Acesso em: 16 jun. 2019.

MEDIATEK LABS. **Using GATT in Bluetooth Low Energy on the LinkIt ONE development board**. [S. l.]. Disponível em: https://docs.labs.mediatek.com/resource/linkit-one/en/tutorials/using-gatt-in-bluetooth-low-energy-on-the-linkit-one-development-board. Acesso em: 16 jun. 2019.

MIT App Inventor. **About us**. [*S. l.*]. Disponível em: https://appinventor.mit.edu/explore/about-us.htmls. Acesso em: 5 jun. 2019.

MIT App Imventor. **PaintPot (Part 2) for App Inventor 2**. [S. l.]. Disponível em: http://appinventor.mit.edu/explore/ai2/paintpot-part2.html. Acesso em: 16 jun. 2019.

NANJING TOP POWER ASIC CORP. **TP4056 1A Standalone Linear Li-lon Battery Charger with Thermal Regulation in SOP-8**. [*S. l.*: *s. n.*], 2018. Disponível em: https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Prototyping/TP4056.pdf. Acesso em: 11 jun. 2019.

PERNIA-MÁRQUEZ, Ing. Daniel A. **Introducción a la Medición de Vibración**. 2004. Artigo (Pós-Graduação) - Estudante, Mérida, Venezuela, 2004.
SPARKFUN. SparkFun **Triple Axis Accelerometer Breakout - ADXL345**. [S. l.]. Disponível em: https://www.sparkfun.com/products/9836. Acesso em: 16 jun. 2019.

SIG BLUETOOTH. **Our History**. [*S. l.*]. Disponível em: https://www.bluetooth.com/about-us/our-history/. Acesso em: 12 jun. 2019.

TRIMBLE. **Company History**. [S. l.], 18 jun. 2019. Disponível em: https://www.trimble.com/Corporate/About_History.aspx. Acesso em: 16 jun. 2019.

VEIGA, Adonay Aum. **Estudo e implementação de uma solução para cidade inteligente baseada em redes mesh sobre Bluetooth de baixa energia**. 2018. Projeto Final (Graduação) - Estudante, Brasília, 2018.

VIBRODATA. **Acelerometro Triaxial**. [S. l. Disponível em: www.vibrodata.com.br%2Facelerometro-triaxial.html&psig=AOvVaw2WeR5YPp0NfdXsI0LFFJfB&ust=1560801473576063. Acesso em: 16 jun. 2019.

VIEIRA, Joao P. B G.; ARAUJO, Matheus Henrique Mendonça. **Base de Dados das Simulações**. Brasília, 13 jun. 2019. Disponível em: https://drive.google.com/drive/folders/1H_SlUTkuo33pAp5kIAxZaR0FrEWhjMMb?usp=sharing. Acesso em: 13 jun. 2019.

WANG, Jin *et al*. **An enhanced fall detection system for elderly person monitoring using consumer home networks**. IEEE Transactions on Consumer Electronics, Nanquim, Jiangsu, China, p. 23 - 29, 2 abr. 2014.

WANG, Yuxi; WU, Kaishun; NI, Lionel M. **WiFall: Device-Free Fall Detection by Wireless Networks**. IEEE Transactions on Mobile Computing, Shenzhen, Guangdong, China, p. 581 - 594, 22 abr. 2018.

WATABE, Takayuki et al. **Falls in the sitting position - Characteristics and efficacy of preventive measures**. Japanese Journal of Comprehensive Rehabilitation Science, Tóquio, Japão, p. 151 - 157, 2015.

WORLD HEALTH ORGANIZATION. **Falls**. [*S. l.*], 16 jan. 2018. Disponível em: https://www.who.int/en/news-room/fact-sheets/detail/falls. Acesso em: 5 jun. 2019.

WU, Falin *et al*. **Development of a Wearable-Sensor-Based Fall Detection System**. International Journal of Telemedicine and Applications, Beijing, China, p. 1-11, 30 dez. 2014.

XU, Tao; ZHOU, Yun; ZHU, Jing. **New Advances and Challenges of Fall Detection Systems: A Survey**. Applied Sciences, Shaanxi, China, 12 mar. 2018.

3DEXPORT. **Sketchup Hotel Complex**. [S. l.]. Disponível em: https://3dexport.com/items/2018/08/28/542204/210303/sketchup_hotel_complex_g7_3d_model_c4d_max_obj_fbx_ma_lwo_3ds_3dm_stl_2216844_o.jpg. Acesso em: 16 jun. 2019.