



**Miguel Filipe Batista Prego**

Licenciado em Engenharia Eletrotécnica e de Computadores

## **Deteção e registo automático de irregularidades no asfalto**

Dissertação para obtenção do Grau de Mestre em  
**Engenharia Eletrotécnica e de Computadores**

Orientador: José Manuel Matos Ribeiro da Fonseca,  
Professor Auxiliar com Agregação,  
Faculdade de Ciências e Tecnologia  
Universidade Nova de Lisboa



FACULDADE DE  
CIÊNCIAS E TECNOLOGIA  
UNIVERSIDADE NOVA DE LISBOA

**Março, 2017**



*Aos meus pais,  
por todo o apoio incondicional que me deram durante o  
decorrer deste curso.*



## AGRADECIMENTOS

Em primeiro lugar gostaria de agradecer ao Professor José Manuela Fonseca por me ter ajudado no desenvolvimento desta dissertação, pela sua disponibilidade e empenho. Gostaria também de agradecer à Faculdade de Ciências e Tecnologia por toda a aprendizagem que fui capaz de adquirir, bem como aos seus professores que me passaram o conhecimento importante para que fosse possível completar o curso. Quero também agradecer à minha família, pelo apoio e motivação excepcionais, em particular aos meus pais, que estiveram sempre ao meu lado e me apoiaram, tanto nos bons como nos maus momentos. Deixo um agradecimento especial à minha namorada e colega, Gisela Seixas, por todos os momentos de descontração e alegria que me deu e também por toda a ajuda e companheirismo que mostrou durante estes últimos três anos. Não poderia esquecer-me dos meus colegas de curso, em particular ao André Estevam, à Daniela Oliveira, ao David Mestre, ao Duarte Segurado, ao Fábio Carmo, ao Fábio Silva, ao Gonçalo Freitas e à Joana Barruncho por todos os momentos animados, divertidos e únicos que tivemos juntos. Todas as palhaçadas, risadas e brincadeiras com vocês tornaram este caminho muito mais fácil de fazer. Por último, queria agradecer ao Riscas, o meu gato, um membro da família que me acompanhou desde cedo nesta caminhada, que soube sempre quando mais precisava de um apoio quando algo corria mal.



## **R E S U M O**

---

O trabalho descrito nesta dissertação foi desenvolvido de maneira a poder ajudar a solucionar o problema das estradas irregulares. A metodologia proposta é apenas uma possibilidade entre muitas outras funcionais que tenta estabelecer uma ligação com a “Internet das Coisas” e com as “Cidades Inteligentes”, dois conceitos que têm apresentado um grande crescimento nos últimos anos. O trabalho desenvolvido está dividido em três partes, cada uma diretamente relacionada com um dispositivo específico. A primeira parte, em que é feita a recolha de dados, utiliza um microprocessador Arduino Uno, que guarda e analisa valores recolhidos por um acelerómetro e os junta a informação fornecida por um receptor GPS. Através de uma ligação Bluetooth, esta informação é passada para um telemóvel que representa a segunda fase do processo, onde os dados são armazenados para um posterior envio para o último passo do processo. Nesse último passo, os dados são recebidos num servidor, via Wi-Fi e são dispostos num *web site* para uma fácil consulta dos utilizadores. Todo este processo permitiu o estudo e utilização de várias tecnologias, bem como a interação entre as mesmas, tornando-se assim num projeto bastante diversificado com focos em diferentes áreas.



## **ABSTRACT**

---

The work described in this dissertation was developed in order to help solve the problem of irregular roads. The proposed methodology is just one possibility among many other that also work and tries to establish a connection with the “Internet of Things” and with “Intelligent Cities”, two concepts that have shown a great growth in recent years. The developed work is divided into three parts, each one directly related to a specific device. The first part, in which the data is collected, uses an Arduino Uno microprocessor, which stores and analyzes values collected by an accelerometer and joins the information provided by a GPS receiver. Through a Bluetooth connection, this information is passed to a mobile phone that represents the second phase of the process, where the data is stored for a subsequent send to the last step of the process. In this last step, the data is received on a server via Wi-Fi and is arranged in a web site for easy user usage. All this process allowed the study and use of several technologies, as well as the interaction between them, thus becoming a very diversified project with focuses in different areas.



# ÍNDICE

<b>Lista de Figuras</b>	<b>xiii</b>
<b>Lista de Tabelas</b>	<b>xv</b>
<b>1 Introdução</b>	<b>1</b>
1.1 Uma curta apresentação . . . . .	1
1.2 Montagem do sistema . . . . .	2
1.2.1 Componentes integrados no veículo . . . . .	2
1.2.2 Componentes do telemóvel . . . . .	2
<b>2 Estado da arte</b>	<b>3</b>
2.1 Principais metodologias . . . . .	3
2.1.1 Câmera vídeo . . . . .	3
2.1.2 Câmera vídeo com iluminação artificial . . . . .	5
2.1.3 Sensor ultrassónico . . . . .	7
2.1.4 Acelerómetro . . . . .	8
2.2 Comparação de resultados . . . . .	9
<b>3 Conceitos teóricos</b>	<b>11</b>
3.1 Sensor - GPS . . . . .	11
3.1.1 NMEA . . . . .	13
3.2 Sensor - Acelerómetro . . . . .	14
3.3 Comunicação - Bluetooth . . . . .	15
3.4 Comunicação - Wi-Fi . . . . .	16
3.5 Comunicação - Redes móveis (3G) . . . . .	17
3.6 Processamento - Arduino . . . . .	17
3.7 Processamento - Android . . . . .	18
3.7.1 Aplicações . . . . .	19
3.7.2 MIT App Inventor 2 . . . . .	20
3.8 Linguagem de programação - C . . . . .	21
3.9 Linguagem de programação - PHP . . . . .	21
3.10 Base de dados - MySQL . . . . .	22
3.11 Linguagem de marcação - HTML . . . . .	22

## ÍNDICE

---

<b>4 Trabalho desenvolvido</b>	<b>25</b>
4.1 Funcionamento do Arduino . . . . .	25
4.2 Funções adicionais . . . . .	27
4.2.1 SD_init . . . . .	27
4.2.2 WriteSD . . . . .	28
4.2.3 ReadSD . . . . .	28
4.2.4 AccelInit . . . . .	29
4.2.5 Accel . . . . .	29
4.2.6 displayGPS . . . . .	30
4.2.7 FTOA - Float to ASCII . . . . .	31
4.3 Funcionamento do Android . . . . .	31
4.4 Base de dados . . . . .	35
4.5 WebSite . . . . .	35
4.5.1 home.php . . . . .	35
4.5.2 store.php . . . . .	36
4.5.3 listLocations.php . . . . .	36
4.5.4 showLocation.php . . . . .	37
4.6 Montagem do sistema . . . . .	38
4.7 Validação de resultados . . . . .	38
4.8 Resultados obtidos . . . . .	40
<b>5 Conclusões</b>	<b>43</b>
5.1 Trabalho futuro . . . . .	43
5.2 Conferências . . . . .	44
<b>Bibliografia</b>	<b>47</b>
<b>A Código do Arduino</b>	<b>49</b>
<b>B Código da aplicação</b>	<b>53</b>
<b>C Código das páginas web</b>	<b>57</b>

## LISTA DE FIGURAS

2.1 Montagem com duas câmeras . . . . .	4
2.2 Sensor Kinect . . . . .	5
2.3 Matriz de deteção de um buraco . . . . .	6
2.4 Comparação da linha em superfície lisa e num buraco . . . . .	6
2.5 Sensor Ultrassom . . . . .	8
2.6 Métodos de deteção de Mednis et al. 2011 . . . . .	9
3.1 Sensor Venus638FLPx . . . . .	12
3.2 Formato de mensagem RMC . . . . .	12
3.3 Formato de mensagem de configuração do recetor GPS . . . . .	13
3.4 Ilustração de um acelerómetro capacitivo . . . . .	14
3.5 Módulo bluetooth . . . . .	16
3.6 Arduino Uno . . . . .	18
3.7 Código exemplo Android . . . . .	19
3.8 Código exemplo AI2 . . . . .	20
3.9 Exemplo de código SQL . . . . .	22
3.10 Exemplo de código HTML e o resultado num browser . . . . .	23
4.1 Arquitetura do sistema . . . . .	25
4.2 Fluxograma referente ao código Arduino . . . . .	26
4.3 Código da função SDInit . . . . .	27
4.4 Código da função WriteSD . . . . .	28
4.5 Código da função ReadSD . . . . .	29
4.6 Código da função AccelInit . . . . .	29
4.7 Gráfico de uma irregularidade detetada . . . . .	30
4.8 Código da função vetorGPS . . . . .	30
4.9 Formatos de apresentação de coordenadas . . . . .	31
4.10 Aspetto da aplicação Smart Shock . . . . .	32
4.11 Fluxograma referente ao código Android . . . . .	34
4.12 Página <i>web home.php</i> . . . . .	36
4.13 Informação tipo adicionada na base de dados . . . . .	37
4.14 Página <i>web listlocations.php</i> . . . . .	37
4.15 Esquemático do sistema . . . . .	38

---

**LISTA DE FIGURAS**

4.16 Montagem real do sistema . . . . .	39
4.17 Instalação do sensor no veículo de testes . . . . .	39
4.18 Percurso de validação de resultados . . . . .	40
5.1 Poster apresentado na Green Business Week . . . . .	45
A.1 Zona de inicialização e função setup . . . . .	49
A.2 Função loop . . . . .	50
A.3 Funções accelInit e SD_init . . . . .	50
A.4 Funções WriteSD e ReadSD . . . . .	51
A.5 Função Accel . . . . .	51
A.6 Funções getField e FTOA . . . . .	52
B.1 Ciclo principal da aplicação . . . . .	53
B.2 Gestão de mensagens de erro . . . . .	54
B.3 Inicialização de variáveis e ações dos botões “Desligar” e “Dispositivos” . . .	54
B.4 Ação do botão “Shock” e remoção do ficheiro do telemóvel . . . . .	55
B.5 Ação do botão “Enviar” . . . . .	55
B.6 Transformação de variáveis para envio para a base de dados . . . . .	56
C.1 Código da página home.php . . . . .	58
C.2 Código da página listlocationsphp.php . . . . .	59
C.3 Código da página storephp.php . . . . .	59
C.4 Código da página showlocationsphp.php . . . . .	60

## **L**ISTA **D**E **T**ABELAS

2.1	Comparação de resultados de artigos . . . . .	10
2.2	Comparação de resultados com utilização de acelerómetro . . . . .	10
3.1	Interpretação dos dados de uma mensagem NMEA do tipo RMC . . . . .	13



# INTRODUÇÃO

## 1.1 Uma curta apresentação

Ao longo dos anos, o número de defeitos e falhas nas estradas tem vindo a aumentar, mostrando-se um problema cada vez mais importante na vida de muitos cidadãos, especialmente os que fazem da condução a sua profissão. De modo a caminhar em direção à resolução deste problema, surgiu a oportunidade de realizar o projeto aqui apresentado, em que será sugerida uma das muitas soluções possíveis, utilizando uma abordagem que está diretamente ligada ao rápido aumento de utilizadores de telemóvel bem como ao grande número de condutores em Portugal, visto que existem cerca de 19 milhões de telemóveis em Portugal<sup>1</sup> para os cerca de 11 milhões de habitantes na mesma região<sup>2</sup>.

Surge então a necessidade de uma monitorização das vias de trânsito para a sua manutenção e reparação. Desta forma, tentando aliar a tecnologia com essa mesma necessidade, será desenvolvido um sistema que irá ser integrado em automóveis e que fará a deteção e comunicação de defeitos nas estradas com o telemóvel de cada automobilista.

Este documento irá fornecer, da forma mais detalhada possível, uma pequena introdução sobre quais os componentes necessários para construir um sistema de deteção de defeitos existentes nas estradas, utilizando um acelerómetro e um telemóvel, bem como comunicação *Bluetooth* e *wi-fi*, de modo a estabelecer a comunicação entre os vários componentes do sistema. Serão também apresentados os diversos pontos fortes e fracos da solução tomada, fazendo a comparação com projetos já desenvolvidos e as suas respetivas soluções.

---

<sup>1</sup><http://www.pordata.pt/Portugal/Assinantes+++equipamentos+de+utilizadores+do+servi%C3%A7o+m%C3%B3vel-1180>

<sup>2</sup><http://www.pordata.pt/Portugal/Popula%C3%A7%C3%A3o+residente+total+e+por+grupo+et%C3%A1rio-10>

## 1.2 Montagem do sistema

O sistema desenvolvido tem como principal objetivo a deteção de irregularidades no asfalto, mantendo sempre um custo de produção bastante reduzido, assim como o consumo de energia. Como tal, primeiro é necessário identificar quais os componentes essenciais para que seja possível o desenvolvimento do projeto, entre os quais se destacam o acelerómetro e o GPS, bem como módulos de comunicação *Bluetooth* e *wi-fi/3G*.

### 1.2.1 Componentes integrados no veículo

De modo a poder obter dados mais precisos, será considerada a utilização de um acelerómetro externo ao telemóvel, preso ao chassi do veículo, abaixo do amortecedor. Esta montagem torna possível obter diretamente as leituras quando um veículo passa numa irregularidade, fazendo com que a recolha de dados seja consistente uma vez que instalado, o acelerómetro não necessita de calibração posterior. Para que seja possível conhecer a localização da irregularidade, um recetor GPS está instalado na viatura e sempre que detetada uma irregularidade, as coordenadas são lidas e armazenadas. Os dados obtidos são enviados para o telemóvel assim que exista uma ligação através de um Arduino que seja capaz de estabelecer comunicação *Bluetooth*.

### 1.2.2 Componentes do telemóvel

No telemóvel apenas é necessário armazenar dados para um posterior envio para uma base de dados, através da antena Wi-Fi ou 3G existente no aparelho. É também possível a comunicação de irregularidades manualmente, sendo utilizado o GPS do telemóvel neste caso. Para poder realizar este procedimento, foi desenvolvida uma App utilizando o MIT App Inventor 2.



## ESTADO DA ARTE

### 2.1 Principais metodologias

Para que esta dissertação pudesse apresentar vantagens em relação a trabalhos já desenvolvidos, foi necessário fazer uma pesquisa sobre metodologias existentes que propusessem soluções para a deteção de irregularidades existentes no asfalto, sendo que foram apenas encontrados trabalhos que propõem a deteção de buracos no asfalto, deixando de parte outras irregularidades como juntas de dilatação ou lombas. Após a análise de vários trabalhos relacionados com este problema, foi possível agrupar os trabalhos mais relevantes em categorias referentes às metodologias utilizadas para possíveis soluções, sendo os principais:

1. Câmera vídeo
2. Câmera vídeo com iluminação artificial
3. Ultrassom
4. Acelerômetro

Nesta secção serão descritos os trabalhos e as tecnologias utilizadas em cada um deles, bem como as vantagens e desvantagens de cada uma das implementações apresentadas.

#### 2.1.1 Câmera vídeo

No que toca às câmeras de vídeo, os trabalhos aqui referidos não são diretamente comparáveis com o trabalho que é aqui apresentado, uma vez que utiliza uma tecnologia para a deteção e reconhecimento de buracos no asfalto bastante diferente da que foi utilizada nesta dissertação. Ainda assim, são trabalhos que se enquadram no tema e dos quais

foram retiradas ideias para metodologias de comunicação entre os vários componentes da montagem. De todos artigos estudados, três são aqueles que mostram abordagens mais interessantes, pois apresentam soluções diferentes entre si. Em Zhang et al. 2014 e Chan et al. 2014 são utilizadas duas câmeras e apresentados algoritmos muito semelhantes, com resultados bastante positivos, sendo possível verificar uma relação entre estes trabalhos devido às várias colaborações já existentes entre os autores. A solução proposta em Zhang et al. 2014 utiliza duas câmeras para criar uma imagem a três dimensões de modo a conseguir detetar um buraco no asfalto mas também calcular a sua profundidade. Cruzando a imagem de ambas as câmeras, é possível encontrar os pontos comuns no asfalto de modo a sobrepor as imagens para o cálculo do desvio que existe entre elas de forma a criar um mapa virtual da superfície e do buraco. Depois da criação do mapa é possível determinar qual o plano da estrada e também os pontos que se encontram abaixo desse plano, onde se encontram localizados os buracos. O sistema é bastante preciso, permitindo a deteção de buracos com profundidades maiores que 4cm mas apresenta problemas de transmissão em tempo real uma vez que é necessário bastante processamento para fazer a sobreposição das imagens e também o mapeamento da superfície.

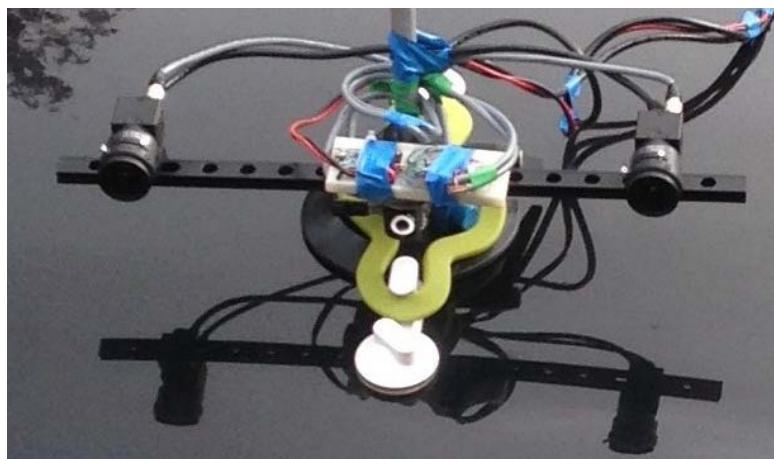


Figura 2.1: Montagem com duas câmeras <sup>1</sup>

Em Chan et al. 2014 é apresentada uma melhoria a este sistema, no que diz respeito ao processamento em tempo real, sendo que foram possíveis avaliar cinco imagens por segundo, uma a cada 200 milissegundos, ao invés de apenas três. Conjugando estes dois projetos é possível obter-se uma solução viável em termos de processamento e deteção mas que necessita de duas câmeras e um processador de gama superior, capaz de suportar todo o processamento necessário, tornando assim a sua montagem bastante dispendiosa.

Em Moazzam et al. 2013 é descrita uma solução diferente baseada num processo de deteção de buracos utilizando um sensor Kinect <sup>2</sup> que só tem uma câmera mas tem um sensor de infravermelhos para determinar a distância a que se encontra de um objeto.

---

<sup>1</sup>Zhang et al. 2014

<sup>2</sup><https://developer.microsoft.com/en-us/windows/kinect>

O sensor Kinect calcula a distância até um obstáculo e faz o mapeamento dessas distâncias, criando um mapa virtual do solo. De seguida, as coordenadas X e Y do mapa são convertidas para o mundo real, de modo a obter as coordenadas reais. São então feitos diversos cálculos para determinar as áreas de diferentes profundidades do buraco para que seja possível saber o volume que o buraco ocupa no asfalto. Por fim, os buracos são classificados com base nas curvas de relação área-profundidade, área ao nível do asfalto e profundidade máxima. Esta solução tem resultados semelhantes aos de Zhang et al. 2014 e Chan et al. 2014 e para a sua implementação é necessário menos processamento, uma vez que o software Kinect está otimizado. Além disso, embora não seja apresentado, é de esperar que esta opção seja menos dispendiosa que as anteriores graças ao preço do Kinect e que não seja necessário um processador tão potente. Ainda assim existe a grande desvantagem do sistema só ter sido testado de forma estacionária, detetando buracos num ambiente controlado, não existindo dados para a instalação em automóvel.

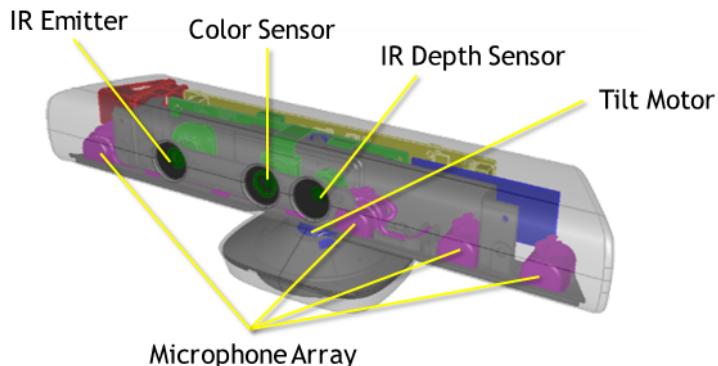


Figura 2.2: Sensor Kinect<sup>3</sup>

### 2.1.2 Câmera vídeo com iluminação artificial

Apesar dos artigos apresentados nesta secção utilizarem também câmeras, é interessante separá-los dos anteriores uma vez que a deteção dos buracos é feita de forma diferente, consoante a passagem pelo próprio buraco. Em Yu e Salari 2011 é utilizado um laser vermelho que projeta uma reta no asfalto, perpendicular ao sentido de deslocação do automóvel. Esta reta é então capturada por uma câmera que analisa a imagem e a filtra, através de um filtro de mediana a quatro máscaras, de modo a remover ruído, mantendo detalhe suficiente na imagem. Utilizando depois o método de Otsu, a imagem é binarizada para uma fácil deteção da linha laser. Essa imagem binarizada é então colocada numa matriz X Y, juntamente com as imagens sucessivas para que seja possível mapear um buraco e a sua área de superfície. Utilizando cada imagem e calculando a distância do desvio da linha é também possível calcular a profundidade do buraco, sendo esse um dos parâmetros utilizados nas classificações de todas as deteções. Apesar de mostrar bons resultados, são salientadas nas conclusões que existem algumas perturbações da

<sup>3</sup><http://developer.microsoft.com/en-us/windows/kinect>

linha aquando a passagem por zonas mais danificadas, diminuindo assim a qualidade da avaliação de uma deteção e é também referida a necessidade de um algoritmo de avaliação robusto que leva a que seja necessário um nível de processamento elevado.

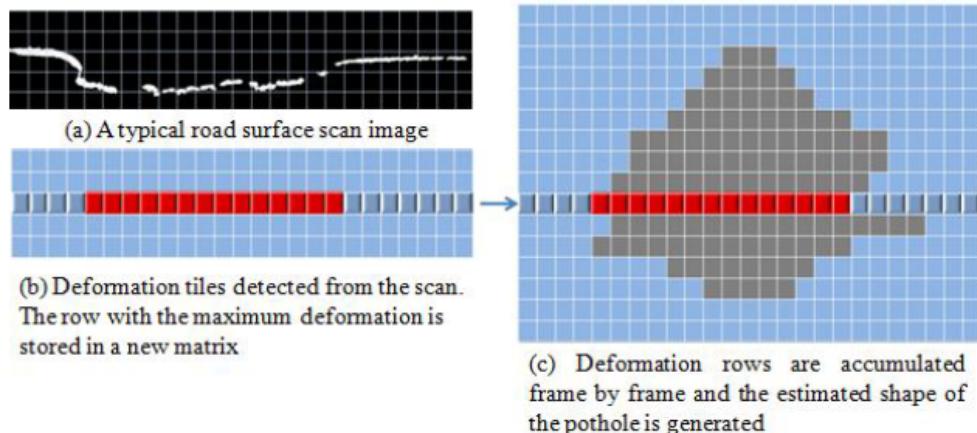


Figura 2.3: Matriz de deteção de um buraco <sup>4</sup>

Já em He et al. 2011 as linhas vermelhas projetadas no solo são provenientes de uma luz *led* de baixa potência, também perpendiculares à estrada. Este sistema utiliza duas câmeras para fazer a deteção da linha, aumentando a precisão da classificação do buraco quanto à sua profundidade. De modo a diminuir ruído existente na imagem, é utilizado o método de Otsu para deteção da linha, bem como uma avaliação dos pixels da vizinhança para que possam ser eliminados falsas deteções, uma vez que a linha é composta por vários pixels de cor semelhante. Por fim a coloração da imagem é invertida, e é feita uma dilatação e uma contração para uma melhor definição da linha detetada. Este sistema consegue apresentar erros inferiores a 2 mm em termos de profundidade quando comparado com instrumentação de alta precisão para medição de irregularidades.



Figure 2. LED light band on smooth-riding surface.

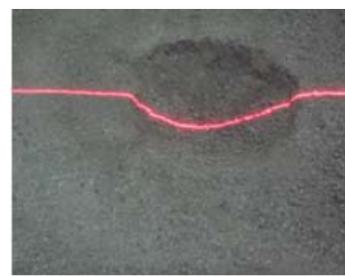


Figure 3. LED light band in pothole.

Figura 2.4: Comparação da linha em superfície lisa e num buraco <sup>5</sup>

---

<sup>4</sup>Yu e Salari 2011

<sup>5</sup>He et al. 2011

### 2.1.3 Sensor ultrassónico

Outra abordagem completamente diferente é a utilização de sensores ultrassom. Um sensor ultrassónico é um dispositivo utilizado para medir a distância a um objeto através de ondas sonoras. Este dispositivo mede a distância enviando uma onda sonora com uma frequência específica e aguarda que a onda sonora retorne. Ao gravar o tempo decorrido entre emissão da onda sonora e a onda sonora de retorno, é possível calcular a distância entre o sensor e o objeto. Uma vez que se sabe que o som viaja através do ar a cerca de  $344m/s$  (pois depende da temperatura do ar), é possível multiplicar o tempo de viagem da onda sonora por 344 metros para encontrar a distância total da viagem da onda sonora, sendo que este valor representa a distância de ida e volta da onda pelo que para encontrar a distância até o objeto, basta dividir esta distância por dois. Esta é uma opção semelhante à que foi tomada neste projeto na medida em que é quase obrigatório passar por um buraco para fazer a sua deteção, ao contrário da análise de imagens em que é possível evitá-los. Em Hegde et al. 2015 é mostrado um protótipo que permite a deteção de buracos através de um sensor ultrassónico testado num protótipo autónomo. Quando detetado um buraco, é estabelecida a comunicação com outros veículos de modo a que possam ajustar as suas velocidades dependendo da qualidade da estrada, diminuindo-a sempre que são detetados buracos. Para a comunicação foi utilizado um módulo ZigBee que permite estender a comunicação a mais veículos em trabalhos futuros. O sistema apresentou bons resultados em estradas simuladas mas foi necessário fazer uma calibração específica para este protótipo, sendo que nas conclusões é referida a necessidade de um ajuste aos valores de deteção, consoante o veículo onde o sistema se encontra montado. Este fator é um grande inconveniente pois depende de cada veículo e da qualidade das suspensões pois caso a suspensão seja muito sensível poderão ser detetados falsos positivos. Madli et al. 2015 é um projeto mais elaborado, em que uma montagem semelhante é utilizada num veículo e o conceito é testado em estradas reais. Este sistema tem também a vantagem de detetar elevações no asfalto, através de uma diminuição da distância entre o sensor e o asfalto, ao contrário do que acontece num buraco em que a distância aumenta. O sistema é também composto por um módulo GPS e outro GSM para que seja possível determinar as coordenadas de uma deteção e enviá-las para um telemóvel, através de uma mensagem de texto, onde é possível consultar e armazenar todas as deteções. Foi também desenvolvida uma aplicação Android que determina a localização do utilizador e, caso a distância até uma deteção anterior seja inferior a 100 m, um alerta é gerado para que o utilizador possa ajustar a sua velocidade em relação às condições do asfalto. Apesar dos bons resultados na deteção, o sistema tem algumas falhas ao nível da comunicação com o utilizador pois não é possível eliminar deteções antigas no caso delas serem reparadas, sendo enviados avisos mesmo que uma estrada se encontre em boas condições. Também existe o mesmo problema que em Hegde et al. 2015 quando avaliada a instalação do sistema. Uma vez que este não se encontra no quadro do veículo, o sistema pode detetar falsos positivos devido a oscilações no terreno que não sejam causadas por buracos nem lombas mas apenas pelo

asfalto defeituoso.



Figura 2.5: Sensor Ultrassom<sup>6</sup>

#### 2.1.4 Acelerómetro

Esta será a abordagem a ser utilizada e aquela em que os artigos analisados se mostram mais relevantes, devido à semelhança com o projeto que se pretende desenvolver. Em Mednis et al. 2011 é utilizado o acelerómetro do telefone e são apresentados vários algoritmos para a determinação do que é ou não um buraco na estrada. São também feitas algumas comparações sobre os resultados que diferentes telefones apresentam, dependendo do seu acelerómetro, tendo em conta a frequência de amostragem e o desvio padrão do eixo Z. Neste trabalho foi concluído que diferentes telemóveis recolhem diferentes informações no mesmo percurso, sendo crítico que o acelerómetro não esteja incluído no telemóvel para que exista consistência nos dados recolhidos por diferentes utilizadores. Foram também avaliados quatro métodos de análise de aceleração para determinar deteções de buracos, sendo que o método "Z-DIFF" foi o que apresentou melhores resultados (ver figura 2.6). Graças a esta análise, foi desenvolvido código que pudesse fazer uma análise semelhante à proposta neste trabalho pois este método apresentou resultados positivos em 92% dos casos.

No que diz respeito a Fouad et al. 2014, é um documento semelhante ao anterior mas adiciona um giroscópio para melhor deteção de buracos. Embora os resultados apresentados sejam bons, terá que ser tido em conta o processamento extra necessário para a utilização dos dados do giroscópio que não parece adicionar muito mais sensibilidade ao sistema. Neste trabalho, foi também desenvolvida uma aplicação em Android para que os dados recolhidos possam ser armazenados e consultados pelos utilizadores, apesar de não existir comunicação com outros elementos, como uma base de dados para cruzamento de dados com outros utilizadores. Tal como mencionado nas conclusões, este trabalho não apresentou resultados promissores uma vez que teve uma taxa de sucesso de apenas 75% graças à utilização do acelerómetro do telemóvel. Em Chen et al. 2011 foi criado

---

<sup>6</sup>[http://www.3bscientific.es/sensor-de-movimento-de-ultrassom-u11361,p\\_559\\_14224.html](http://www.3bscientific.es/sensor-de-movimento-de-ultrassom-u11361,p_559_14224.html)

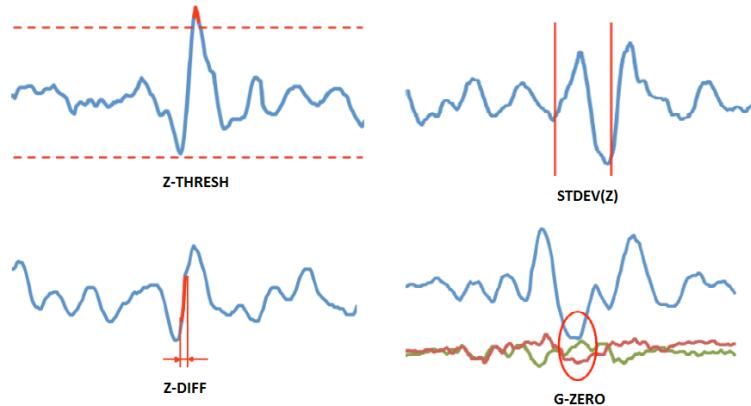


Figura 2.6: Métodos de deteção de Mednis et al. 2011

um elemento que contém GPS e acelerómetro externos ao telemóvel e ainda um microprocessador para processar os dados adquiridos. É um sistema muito bem construído e que apresenta vários resultados em estradas de diferentes condições mas poderá ser mais caro do que o pretendido desenvolver neste projeto. No artigo apresentado em Jang et al. 2015 é apresentada a comunicação entre vários veículos quanto à sua deteção de buracos. Esta ideia é bastante interessante uma vez que possibilita a comunicação em tempo real e caminha na direção das *Smart Cities* e da IoT (Internet of Things - Internet das Coisas em Português). O sistema é composto por um par de acelerómetros instalado no *tablier* do automóvel e uma antena GPS no seu exterior para a obtenção de coordenadas. Esta solução apresenta resultados bastante precisos mas tem um elevado nível de processamento, pelo que foi necessário um processador mais potente para a gestão dos dados detetados, levando a que o preço total da montagem fosse mais elevado do que o desejado. Por fim, é ainda de salientar o trabalho em Kattan e Aboalmaaly 2014 que apresenta uma metodologia semelhante à que foi tomada neste trabalho no que diz respeito à comunicação com uma base de dados mas com pouco desenvolvimento e uma curta fase de testes, sendo que as deteções de buracos tiveram uma taxa de sucesso de apenas 61% devido à colocação do telemóvel no *tablier* do veículo e à baixa qualidade do algoritmo de deteção. Foi também utilizado o GPS do telemóvel que apresentou algumas falhas devido à sua baixa precisão e potência, sendo que em alguns casos não era possível obter a localização do utilizador. Ainda assim é mencionado que para trabalho futuro é proposto um melhoramento deste algoritmo bem como uma nova solução para a deteção de coordenadas.

## 2.2 Comparação de resultados

Comparando todos os artigos apresentados, é de notar que cada um tem os seus pontos fortes e fracos, como seria de esperar. Para o trabalho que será desenvolvido as qualidades mais importantes a ter em conta são o preço do material utilizado e também o tempo de processamento que o método consome. A tabela 2.1 mostra as características que serão

tidas em conta bem como quais as metodologias que as apresentam.

Tabela 2.1: Comparação de resultados de artigos

	Qualidade de resultados	Preço de material	Tempo de processamento	Processador já incluído
Câmera vídeo	✓	✗	✗	✗
Câmera + iluminação	✓	✗	✗	✗
Ultrassom	✓	✓	✓	✗
Acelerómetro	✓	✓	✓	✓

Desta forma, a utilização de um acelerómetro é a mais indicada. Dentro desta metodologia, ainda é possível fazer uma comparação dos vários artigos analisados e tirar algumas conclusões, apresentadas na tabela 2.2.

Tabela 2.2: Comparação de resultados com utilização de acelerómetro

	Acelerómetro do telefone	GPS do telefone	Giroscópio do telefone	Processador do telefone	Custos além do telefone
Mednis et al. 2011	✓	✓	✗	✓	✗
Fouad et al. 2014	✓	✓	✓	✓	✗
Chen et al. 2011	✗	✗	✗	✗	✓

Embora os resultados dos trabalhos em que todos os componentes fazem parte do telefone sejam mais promissores em termos do preço dos materiais, a sua viabilidade é mais baixa, uma vez que um sistema que seja para o público em geral necessita de apresentar resultados consistentes, independentemente da situação e se o acelerómetro não estiver sempre no mesmo sítio (neste caso, o telefone) as leituras de cada irregularidade detetada são alteradas a cada passagem, dependendo do local em que o telefone se encontra, seja no bolso do casaco, no banco do veículo ou no seu *tablier*. A utilização de GPS é obrigatória para que os buracos detetados possam ser localizados, sendo que um GPS externo pode apresentar mais precisão do que o integrado no telemóvel e evita falhas de receção tal como é referido em Kattan e Aboalmaaly 2014. Desta forma, a solução apresentada neste projeto conta com um smartphone do utilizador, bem como um acelerómetro e GPS externos que estão fixados no veículo. O acelerómetro está colocado diretamente no quadro do veículo de testes de modo a que a qualidade das suspensões não interfira com as leituras. O GPS foi colocado no interior do veículo pois a sua antena é bastante potente, fazendo com que não seja necessário um local muito específico para a sua colocação.

## CONCEITOS TEÓRICOS

### 3.1 Sensor - GPS

O Sistema de Posicionamento Global (GPS – Global Positioning System) é uma rede utilizada mundialmente pela população para determinação da localização na superfície do globo terrestre. Este sistema tem o nome de Navstar e é financiado e gerido pelo Departamento da Defesa dos Estados Unidos da América. Este é um dos vários sistemas de posicionamento existentes mundialmente, pois existem também os sistemas GLONASS da Rússia, o COMPASS desenvolvido pela China e o GALILEO da União Europeia, cada um deles com diferentes níveis de desenvolvimento e testes. Apesar de ter sido desenvolvido para fins militares, em 1983 foi permitida a sua utilização por parte de civis, sendo que na época o sinal era degradado e continha um erro de +/- 100 metros, algo que foi eliminado no ano de 2010. Os satélites GPS transmitem sinais para os receptores GPS, os quais são elementos passivos no sistema pois não transmitem qualquer tipo de informação, tornando-se este um sistema de comunicação unilateral. Este serviço de localização necessita de uma visão desimpedida do céu e apenas no exterior consegue apresentar bons resultados, os quais dependem de relógios atómicos presentes nos satélites que proporcionam ao sistema a sua elevada precisão. Cada satélite GPS transmite informação relativa ao seu posicionamento relativo ao planeta, bem como a hora atual. Todas as operações realizadas pelos satélites encontram-se sincronizadas para que o seu sinal constante seja transmitido nos mesmos instantes de tempo. Os sinais emitidos pelos satélites são enviados à velocidade da luz para os receptores GPS e utilizando o tempo de percurso desse sinal é possível determinar a distância a que cada satélite se encontra do receptor. Assim que o receptor ler a informação de pelo menos quatro satélites é possível calcular a sua posição em três dimensões. A todo o momento existem pelo menos 24 satélites GPS em funcionamento que realizam duas órbitas por dia a uma altura de 18

500 km e a uma velocidade próxima de 14 000 km/h. Cada recetor recebe informação sobre o local em que um satélite se encontra num determinado momento do dia e com essa informação consegue determinar que se encontra algures numa superfície esférica em que o centro é o próprio satélite. Cruzando os pontos de pelo menos quatro satélites, o recetor determina o local em que todas as esferas se intersectam e desta forma conclui a sua própria posição com um erro de +/- 10 m.

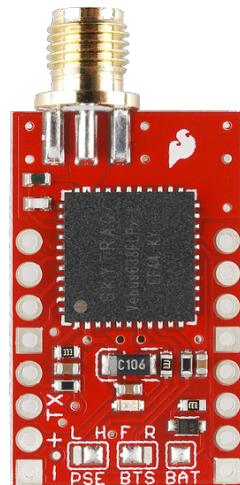


Figura 3.1: Sensor Venus638FLPx<sup>1</sup>

O recetor GPS utilizado foi o Venus638FLPx que recebe mensagens normalizadas NMEA (explicada em 3.1.1) a uma taxa padrão de 9600 bps que pode ser alterada para 115200 bps e é alimentado a uma tensão de 3.3 V. Por definição padrão, este recetor GPS recebe cinco tipos de mensagens NMEA, das quais apenas uma é utilizada neste projeto, pois apenas ela refere todos os valores desejados obter: latitude, longitude, data e hora. Uma mensagem do tipo RMC tem o aspeto da figura 3.2 sendo que a tabela 3.1 descreve o seu significado.

Estrutura:

\$GPRMC, hhmmss.sss, A, dddmm.mmmm, a, dddmm.mmmm, a, x.x, x.x, ddmmyy,, a\*hh<CR><LF>  
1 2 3 4 5 6 7 8 9 10 11

Exemplo:

\$GPRMC,111636.932,A,2447.0949,N,12100.5223,E,000.0,000.0,030416,,A\*61<CR><LF>

Figura 3.2: Formato de mensagem RMC

Foi também necessário fazer algumas configurações, utilizando um conjunto de mensagens que o recetor conseguisse entender. Todas as mensagens têm o formato apresentado na figura 3.3, em que o campo <CC> representa o código da mensagem e <BB> a informação a transmitir nessa mensagem. As mensagens enviadas tinham os códigos “05”

---

<sup>1</sup><http://www.sparkfun.com/products/11058>

e “09”, referentes à velocidade de comunicação do GPS e ao tipo de mensagem que o GPS deve ler, respetivamente. Toda esta informação foi retirada diretamente do datasheet do recetor Venus638FLPx.

```
<0xA0,0xA1>< PL><CC><BB><CS><0x0D,0x0A>
```

Figura 3.3: Formato de mensagem de configuração do recetor GPS

Tabela 3.1: Interpretação dos dados de uma mensagem NMEA do tipo RMC

Campo	Nome	Exemplo	Descrição
1	Tempo UTC	111636.932	Tempo UTC no formato hhmmss.sss (000000.000 ~235959.999)
2	Estado	A	Estado: 'V' - Aviso na receção de dados 'A' - Dados válidos
3	Latitude	2447.0949	Latitude no formato ddmm.mmm
4	Indicador N/S	N	Indicador de hemisfério: 'N' - Norte 'S' - Sul
5	Longitude	12100.5223	Longitude no formato dddmm.mmm
6	Indicador E/W	E	Indicador de hemisfério: 'E' - Este 'W' - Oeste
7	Velocidade no solo	000.0	Velocidade no solo em nós
8	Percorso no solo	000.0	Direção de deslocação em graus
9	Data UTC	030416	Data UTC no formato ddmmyy
10	Indicador de modo	A	Indicador de modo: 'N' - Data inválida 'A' - Modo autónomo 'D' - Modo diferencial 'E' - Modo de estimativa 'M' - Modo manual 'S' - Modo de simulação
11	Confirmação	61	Número de caracteres enviados pelo satélite

### 3.1.1 NMEA

NMEA é a sigla de National Marine Electronics Association (Associação Nacional de Eletrónica dos Marines) desenvolveu uma especificação que define a interface entre vários elementos do equipamento desta forma militar norte-americana. A receção de comunicação GPS está englobada nesta especificação e a maioria dos programas desenhados para fornecer dados sobre posicionamento em tempo real estão desenhados para obter informação neste formato. A informação fornecida por este serviço inclui posição, velocidade e tempo e esta informação é enviada em formato de uma frase, iniciada sempre com o carácter '\$', é procedida pelo prefixo GP (relativo a posicionamento global) e ainda por três letras que identificam o tipo de mensagem que é recebida. Cada frase é terminada

com um carácter de fim de linha e retorno e a informação contida em cada mensagem está separada por vírgulas, tal como indicado na figura 3.2.

### 3.2 Sensor - Acelerómetro

Um acelerómetro é um dispositivo que mede a aceleração a que está sujeito, no caso de se encontrar estacionário na superfície terrestre, este dispositivo sofre uma aceleração vertical de  $9,8 \text{ m/s}^2$  e caso se encontre em queda livre mede uma aceleração de  $0 \text{ m/s}^2$ . Por este valor ser o valor constante de aceleração gravítica, foi criada uma unidade de medida “g” para mais facilmente expressar valores de aceleração, sendo que 1 g equivale a  $9,8 \text{ m/s}^2$ . Os acelerómetros têm diversas aplicações na indústria e na ciência devido à sua elevada sensibilidade e são regularmente utilizados para detetar vibrações em diversas máquinas, em telemóveis e tablets e também em aeronaves, incluindo drones, para uma melhor estabilização de voo. Conceptualmente, um acelerómetro comporta-se como uma massa amortecida por uma mola, sendo possível determinar o valor de aceleração sabendo o peso da massa e a constante de elasticidade da mola. Em termos práticos um acelerómetro pode utilizar vários métodos para determinar as suas leituras mas os dois mais comuns são os acelerómetros piezoelétricos e os capacitivos. Acelerómetros piezoelétricos medem aceleração diretamente proporcional a uma força a partir de certos tipos de cristais.

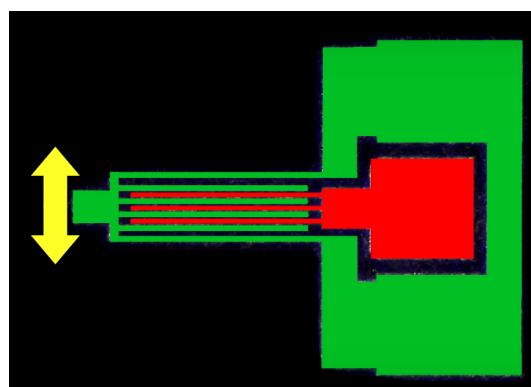


Figura 3.4: Ilustração de um acelerómetro capacitivo <sup>2</sup>

Estes cristais, quando comprimidos, deslocam as cargas existentes no seu interior e consequentemente alteram o seu valor resistivo que pode ser traduzido para uma variação de corrente ou tensão e que pode ser medida. Já os acelerómetros capacitivos (3.4) utilizam um sistema semelhante ao de um condensador. São compostos por duas zonas eletricamente condutoras e isoladas uma da outra que contêm alguma carga. A separação entre estas zonas é tão pequena, que quando existe algum movimento, a distância entre ambas é alterada, alterando assim o valor de capacitação do sistema, que se traduz facilmente para uma variação de tensão. Para este projeto foi utilizado um acelerómetro

---

<sup>2</sup><https://www.youtube.com/watch?v=i2U49usFo10>

ADXL345 capacitivo, capaz de medir variações de aceleração até  $\pm 16g$ . Para a sua utilização foi necessário determinar a velocidade de transmissão com o valor de 9600 bps, tal como os restantes dispositivos do sistema e também a frequência de amostragem de valores, nomeadamente 200 Hz. Para a comunicação foi utilizado o método I<sup>2</sup>C todavia, estava também disponível a opção de transmissão por SPI, sendo que a escolha foi tomada devido a serem necessárias menos ligações em I<sup>2</sup>C. Apesar da comunicação SPI suportar velocidades de transmissão mais elevadas, esta vantagem não se mostra relevante a 200 Hz. O acelerómetro em questão oferece uma biblioteca de funções de teste, entre as quais a deteção de queda livre, toque simples e toque duplo que, apesar de não serem utilizadas no projeto, foram bastante importantes aquando da aprendizagem de utilização do dispositivo. Para que fosse possível obter os valores de leitura do acelerómetro, foi necessário ler os valores entre os registos 50 e 55, sendo que cada par contém informação sobre um dos três eixos de deslocação. Os valores lidos foram então estudados utilizando um gráfico, sendo que a sua análise está descrita na secção 4.7.

### 3.3 Comunicação - Bluetooth

Bluetooth é um padrão de tecnologia sem fio utilizada para troca de dados a curtas distâncias que utiliza ondas rádio UHF, nomeadamente com a utilização de uma frequência que se encontra entre 2400 e 2480 MHz. Esta tecnologia permite a ligação entre vários dispositivos, até um máximo de sete, sendo uma das primeiras tecnologias a surgir que permitiu a ligação entre mais de dois dispositivos. A tecnologia Bluetooth é gerida pelo Grupo de Interesse Especial (SIG - Special Interest Group em inglês) e conta com a colaboração de mais de 30.000 empresas parceiras, que têm atividade nas áreas de telecomunicação, computação e redes. O sistema Bluetooth divide os dados que deseja enviar em pacotes e transmite cada um desses pacotes num dos 80 canais Bluetooth, cada um deles com uma largura de banda de 1 MHz, alternando a seleção de cada canal cerca de 800 vezes por segundo, utilizando a tecnologia AFH - Adaptive Frequency Hopping. Existe também um modo de baixo consumo que utiliza canais de 2 MHz, o que faz um total de 40 canais. Este sistema funciona assente numa estrutura master-slave, em que um mestre (ou dispositivo primário) pode comunicar com um máximo de sete escravos (ou dispositivos secundários), sendo que todos os escravos estão sincronizados com o mestre, através do seu relógio interno. O relógio do mestre conta intervalos de tempo de  $312,5\mu s$ , juntando dois intervalos de tempo para fazer intervalo de comunicação de  $625\mu s$ . A sequência destes intervalos de comunicação determina que o mestre transmite nos intervalos pares e recebe informação nos ímpares e por analogia o escravo recebe nos intervalos pares e envia nos ímpares. É também de salientar que um dispositivo dentro de um sistema não é sempre escravo ou mestre, pois pode mudar o seu estatuto, dependendo da funcionalidade que lhe é atribuída e que um escravo pode ter mais que um mestre. O padrão de Bluetooth tem uma ligação direta entre potência e alcance e esse alcance determina em que classe de dispositivo rádio o transmissor se encontra.

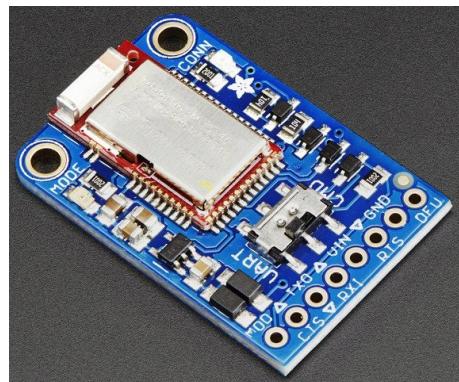


Figura 3.5: Módulo bluetooth<sup>3</sup>

Um transmissor de classe 3 tem um alcance até 1 metro e consome o máximo de 1 mW. Para transmissores de classe 2, o alcance vai até 10 metros, levando a um consumo máximo de 2,5 mW. Por último, a classe 1 tem transmissores com alcance máximo de 100 metros, sendo consumidos até 100 mW. O sistema Bluetooth encontra-se principalmente na classe 2 mas versões mais atuais conseguem por vezes alcançar a classe 1, com um alcance de 60 metros. No que diz respeito ao dispositivo utilizado, foi um HC-06 3.5 série que não necessitou de qualquer tipo de configuração uma vez que a frequência de transmissão pré-definida é de 9600 bps. Foi apenas necessário alimentar este dispositivo e quando era estabelecida a ligação, uma palavra-passe tinha que ser introduzida, neste caso, “1234”.

### 3.4 Comunicação - Wi-Fi

Wi-Fi é uma tecnologia de ligação local sem fios baseada nos padrões IEEE 802.11, uma norma que regula a comunicação sem fios em determinadas frequências rádio. Entre os dispositivos que utilizam esta tecnologia estão os computadores, consolas de jogos, tablets e telemóveis e também impressoras, mas outros dispositivos como televisões e automóveis começam também a integrar esta tecnologia. A ligação à Internet pode ser feita através de WLAN, uma rede local que usa ondas de rádio, através de um *router*, dispositivo que obtém ligação à Internet via cabo Ethernet. Como esta tecnologia opera em frequências que não necessitam de licença, o apelo à sua utilização é bastante elevado, sendo apenas necessário equipamento homologado pela entidade responsável do país vendedor do dispositivo, que usualmente segue normas internacionais. A tecnologia Wi-Fi utiliza habitualmente as frequências de 2,4 GHz e 5 GHz e tem um alcance médio de 20 metros para interiores, caso não existam obstáculos pelo caminho, como paredes, o que pode limitar o alcance a apenas uma sala. No que toca ao exterior, a cobertura Wi-Fi pode atingir vários quilómetros caso sejam utilizados repetidores de sinal, normalmente instalados com grandes antenas. A certificação dos dispositivos Wi-Fi é feita pela aliança

<sup>3</sup><https://www.adafruit.com/product/2479>

Wi-Fi, um conjunto de entidades espalhadas por todo o mundo que garantem a utilização do protocolo 802.11 definido pelo IEEE, bem como os padrões de segurança WPA e WPA2 e o padrão de autenticação EAP.

Existem também pontos de acesso espalhados por várias cidades, que fornecem uma rede gratuita para os cidadãos, nomeadamente em Lisboa e em Sintra, assim como em universidades, através da rede Eduroam, comum à maioria das universidades portuguesas, permitindo o acesso a qualquer estudante, independentemente da universidade em que esteja, mesmo não sendo estudante nesse estabelecimento de ensino. Quanto ao nível de interferências, existem bastantes, graças à existência de vários pontos de acesso que sobreponem as suas redes, assim como pela utilização das mesmas frequências por outras comunicações como o Bluetooth ou o ZigBee. Ainda assim, estas interferências são usualmente colmatadas com a rápida transmissão de dados, sendo que informação não recebida é rapidamente reenviada. Outro mecanismo de redução de interferências é o SSID, um identificador de serviço que permite identificar o emissor e receptor de determinada informação.

## 3.5 Comunicação - Redes móveis (3G)

3G, abreviatura de terceira geração, é a terceira geração de tecnologia de telecomunicações móveis sem fios e assenta num conjunto de normas e serviços de utilização de telecomunicações móveis, cumprindo normas IMT-2000. Esta tecnologia tem como principais focos a realização de chamadas por telemóveis, o acesso à Internet sem fios e também transmissão de televisão sem fios. Estas redes permitem a transferência de informação a velocidades sempre superiores a 200 Kbps e podem atingir velocidades de vários Mbps, uma velocidade bastante elevada na área de comunicação sem fios. Gerações anteriores a esta foram criadas no inicio dos anos 80 e cada nova geração surge com espaçamentos de cerca de 10 anos. Esta tecnologia, apesar de ter processos bastante diferentes da Wi-Fi, apresenta resultados semelhantes, pois permite a ligação à Internet através de um telemóvel, tornando possível o envio dos dados recolhidos pelo acelerómetro para uma página web.

## 3.6 Processamento - Arduino

O Arduino é uma plataforma *open-source* utilizada para projetos de eletrónica que contém um micro-controlador programável através de um *software* dedicado. Esta plataforma é muito popular devido à sua facilidade de programação, tanto pela linguagem utilizada (C++) como pelo *hardware* em si, uma vez que é apenas necessário ligar a placa ao computador via USB. Um dos seus pontos fortes está na capacidade de poder interagir com botões, motores, câmeras, LEDs, telemóveis, entre tantas outras coisas. Esta diversidade, aliada ao baixo custo do sistema, tem levado ao crescimento da comunidade que, contribui com código e instruções de utilização para diversos projetos desenvolvidos. A placa mais comum é o Arduino Uno mostrado na figura 3.6, pois contém um número de pins

suficiente para construir a maior parte dos projetos iniciais, sendo que para projetos mais avançados são normalmente utilizados o Arduino Nano, devido ao seu pequeno tamanho, e o Arduino Mega que tem mais pins e um maior poder de processamento. Todas as placas têm pins em comum entre si, alterando a quantidade desses pins dependendo do tamanho da placa. O Arduino pode ser alimentado via USB ou através de um conector que forneça uma tensão dentro de limites específicos, normalmente entre 7 e 12 V. Existem também os pins GND para ligação à massa, 5V e 3.3V, que fornecem alimentação com esses valores, Analog e Digital, utilizados para ligação de sensores analógicos e digitais, respetivamente. Existe também um botão de reset que faz o código reiniciar, para que seja possível fazer múltiplos testes sem que seja necessário desconectar a alimentação. Por fim, existe um conjunto de LEDs de controlo para ajudar o utilizador a saber o estado do seu projeto. O LED “power” indica se existe uma alimentação correta ao Arduino e os LEDs “Rx” e “Tx” fornecem informação relativa à comunicação série do Arduino com outros equipamentos, nomeadamente “Rx” acende quando a placa recebe informação e “Tx” acende quando é enviada informação.



Figura 3.6: Arduino Uno <sup>4</sup>

### 3.7 Processamento - Android

O sistema Android é um sistema operativo para telemóveis desenvolvido pela Google, que utiliza Linux na sua base e tem como principal objetivo o desenvolvimento de aplicações para telemóveis e tablets que disponham de ecrã táctil. Nestes ecrãs é possível detetar diversos gestos como deslizar, bater e apertar, de modo a manipular objetos que estejam visíveis no ecrã, dando também a possibilidade de utilizar o teclado existente no ecrã. É também possível adicionar outros componentes de hardware para controlar as aplicações, através de ligação USB ou Bluetooth, de modo a aumentar as possibilidades de interação com o utilizador. O sistema Android dispõe também de um sistema televisivo, automóvel e de relógios, que ainda se encontram numa fase de desenvolvimento inicial mas apresentam resultados promissores. De modo a fazer uma melhor leitura dos gestos

---

<sup>4</sup><http://www.arduino.cc/en/Main/ArduinoBoardUno>

que o utilizador realiza, o sistema Android é também capaz de ler informação de acelerómetros, giroscópios ou receptores GPS que façam parte do telemóvel, dando respostas visuais pelo ecrã mas também sensoriais ao nível do som e vibração do dispositivo. O ecrã inicial do Android utiliza atalhos e *widgets*, sendo que um atalho inicia a aplicação referente enquanto que um *widget* fornece informação em tempo real, como o estado do tempo ou emails que sejam recebidos pelo utilizador. No topo deste ecrã inicial existe uma barra que fornece informação sobre o dispositivo e as suas conexões, que pode ser arrastada para baixo, expondo informação relevante sobre as aplicações, atualizações e até algumas definições que sejam habitualmente alteradas como o brilho do ecrã ou a ligação à Internet.

### 3.7.1 Aplicações

As aplicações são o que possibilitam uma utilização mais complexa de um dispositivo com o sistema operativo Android instalado. Estas são normalmente desenvolvidas em Java mas podem ser combinadas com código C ou C++, como é possível ver na figura 3.7.



Figura 3.7: Código exemplo Android

A programação destas aplicações é feita num SDK (software de desenvolvimento) que normalmente inclui ferramentas de desenvolvimento, um detetor de erros, algumas bibliotecas, um compilador e um emulador, para que seja possível simular a utilização de um telemóvel no computador. Entre os mais comuns estão o Eclipse, o Android Studio e o MIT App Inventor. O ambiente de programação Eclipse necessita da instalação de um extra para que seja possível o desenvolvimento de código para Android e era o ambiente de desenvolvimento oficial até 2014. No final de 2014, a Google apresentou o Android Studio: um programa dedicado ao desenvolvimento Android que contém todas as opções existentes no universo Android, sendo este o principal ambiente de desenvolvimento.

Existe também o MIT App Inventor 2, que corre num *browser web* e permite o desenvolvimento de aplicações sem que seja escrito código, utilizando a programação por módulos que se ligam entre si, não exigindo uma capacidade de processamento tão alta como o Android Studio. Para que uma aplicação possa ser executada no telemóvel, é necessário instalar o ficheiro APK da aplicação, através da loja “Google Play” ou diretamente por ligação USB a um computador, permitindo a qualquer utilizador o desenvolvimento da sua própria aplicação, o seu teste e partilha. O sistema operativo Android é baseado em Linux, com bibliotecas, API's e algum software escrito em C e ainda algumas traduções de bibliotecas para Java que torna possível a execução das aplicações. Estas bibliotecas em Java são originárias e programadas a partir do OpenJDK, uma ferramenta que contém uma máquina virtual, as suas próprias bibliotecas e também um compilador. Em termos específicos desta dissertação, a escolha do sistema operativo Android foi simples, uma vez que existem cerca de 86% de utilizadores do sistema Android no universo dos telemóveis. Além disso, as ferramentas para desenvolvimento de aplicações Android são mais acessíveis do que para iOS, o segundo sistema operativo mais frequente em telemóveis.

### 3.7.2 MIT App Inventor 2

O MIT App Inventor 2 (AI2) é uma plataforma de *open source* criada pela Google e mantida atualmente pelo Massachusetts Institute of Technology. Esta ferramenta permite que qualquer pessoa crie aplicações para o sistema operativo Android, mesmo que não tenha um conhecimento muito aprofundado de linguagens de programação. Graças à sua programação por blocos, o desenvolvimento de aplicações torna-se muito mais rápido e intuitivo, promovendo a utilização de tecnologia junto daqueles que são usualmente utilizadores, motivando-os a ser também criadores.



Figura 3.8: Código exemplo AI2

Tal como é possível ver na figura 3.8, não existe código relacionado com a criação da imagem, do botão ou da label. O único código que existe está relacionado com as ações que cada um dos elementos, o que torna o desenvolvimento de uma aplicação significativamente mais simples.

## 3.8 Linguagem de programação - C

A linguagem de programação C foi uma das primeiras a surgir e uma das mais utilizadas no ramo das linguagens de baixo nível. Em C, todo o código executável está contido em sub-rotinas denominadas funções, que podem ter parâmetros de entrada e de saída, de forma a comunicar os seus resultados com outras funções. Existe um número limitado de comandos pré-definidos: for, if/else, do/while, switch, entre outros, que formam a base desta linguagem. É também possível fazer as operações matemáticas mais elementares, utilizando os operadores lógicos normalmente utilizados em matemática, assim como operações mais complexas através da utilização de funções já existentes. Existem também sequências de elementos do mesmo tipo chamados “arrays”, que funcionam como um bloco e podem ser manipulados como uma única variável. Devido à sua baixa utilização de memória, é uma das linguagens mais rápida a processar código, sendo uma das principais escolhas para a criação de programas. Para que o código possa ser criada é necessário um editor de texto, sem qualquer tipo de especificações e para que possa ser criado um ficheiro executável é indispensável um compilador, que gera um ficheiro executável a partir do código escrito pelo programador. As possibilidades de programação vão desde a criação de uma máquina de calcular até ao processamento de imagem, passando pela criação de jogos ou a simples gestão de um ficheiro de clientes de uma loja.

## 3.9 Linguagem de programação - PHP

PHP é uma linguagem *script* utilizada no desenvolvimento *web*, que contém código executável e que é normalmente contido em código HTML, utilizando os delimitadores “<php” e “?>”, permitindo que exista código executável numa página juntamente com código estático. Uma das suas vantagens vem ser executada no servidor, permitindo um processamento mais seguro do código fonte, uma vez que é impossível descobrir quais as ações executadas, pois o navegador apenas recebe o resultado do processamento do código. Além disso, o código PHP é bastante simples, promovendo a utilização de novo programadores, oferecendo muitas funcionalidades de modo a satisfazer programadores mais experientes. Esta linguagem é em muitos aspectos semelhante a C, tanto no que toca às variáveis, bem como às suas funcionalidades, possibilitando a criação de funções, vetores e variáveis que podem ter os seus valores alterados tantas vezes quantas desejadas. Desta forma, PHP é uma solução bastante boa para desenvolver código dinâmico em páginas *web*, algo que não é possível fazer com C.

### 3.10 Base de dados - MySQL

MySQL é um gestor de base de dados *open-source* desenvolvido em C que é compatível com diversos sistemas, incluindo Linux, macOS e Windows. Esta base de dados permite o armazenamento de variáveis e os seus valores em tabelas, em que cada linha contém um identificador único, de modo a identificar cada uma das entradas na tabela. Desta forma, é possível determinar quais as linhas da tabela que se pretendem consultar, facilitando a leitura de dados que estejam armazenados. As suas funcionalidades principais incluem a inserção de dados, a sua pesquisa, atualização e remoção, tornando possível uma fácil gestão da informação que aí se encontra armazenada. Para que seja possível fazer pesquisa na base de dados, são necessários operadores, matemáticos que limitam a pesquisa, assim como um conjunto de instruções que determinam o objetivo da pesquisa. Na figura 3.9 é possível ver duas instruções, sendo que a primeira retorna as dez primeiras instâncias em que o valor “força” é inferior ao valor “3” e na segunda são inseridos novos parâmetros na base de dados com o nome “coordenadas”. Neste trabalho foi bastante útil para que fosse possível fazer a gestão e armazenamento de todas as irregularidades detetadas.

```
SELECT forca, latitude, longitude, data FROM coordenadas WHERE forca < 3 LIMIT 10;  
INSERT INTO coordenadas (forca, latitude, longitude, data) VALUES (3, 39.2391, -8.3451, 15/02/2017 13:57);
```

Figura 3.9: Exemplo de código SQL

### 3.11 Linguagem de marcação - HTML

HTML é uma linguagem de marcação utilizada para a construção de páginas *web*, que são interpretadas pelo navegador. O navegador pode ter acesso ao ficheiro se ele estiver alojado num servidor ou num ficheiro local, transformando-o numa página *web* totalmente navegável. Esta linguagem torna possível a utilização de imagens, texto e vídeos numa página *web*, bem como formulários interativos. As marcações de HTML são maioritariamente denominadas *tags*, um tipo de dados que recebe parâmetros de entrada e os molda, consoante a sua função. Estas *tags* costumam ser apresentadas em pares, como “*<h1>*” e “*</h1>*” que afetam todo o código que se encontre entre elas, neste caso, tornando o texto normal em cabeçalho de nível 1, mas também podem ser individuais, como “*<img>*”, que determina uma imagem e todos os seus atributos. Outro aspeto importante em HTML é a declaração da linguagem a utilizar. Apesar da página ter a extensão HTML, ela suporta outras linguagens como PHP, JavaScript e também SQL, tornando esta linguagem muito mais completa e diversificada, sendo assim possível utilizar determinadas linguagens nos pontos fortes de cada uma.

Na figura 3.10 é possível distinguir várias secções do código: O texto entre “*<html>*” e “*</html>*” descreve a página e a linguagem em que deve ser interpretado. Entre “*<title>*” e “*</title>*” vem o título da página que apenas aparece como título do navegador. Já em “*<body>*” e “*</body>*” está o texto visível, que aparece na execução da página. “*<h2>*”

```
<!DOCTYPE html>
<html>
  <title>
    The cube
  </title>
  <body>
    <h2>Uma bela fotografia</h2>
    
  </body>
</html>
```

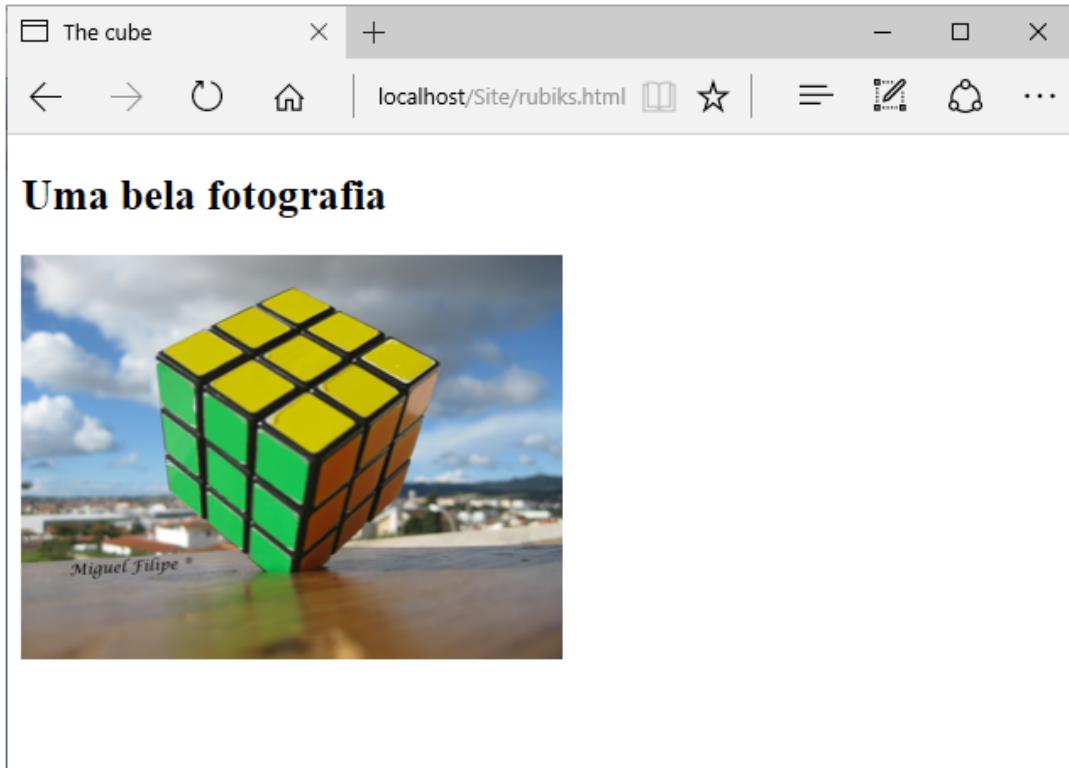


Figura 3.10: Exemplo de código HTML e o resultado num browser

e “</h2>” representam um cabeçalho de nível 2 e <img ... > é uma imagem, com os seus atributos declarados.



## TRABALHO DESENVOLVIDO

A solução proposta para a resolução do problema apresentado no capítulo 1 é um sistema separado em três fases, cada uma delas ligada a um elemento físico: o Arduino, o telemóvel e a base de dados (alocada num computador). Na figura 4.1 é possível ver um esquema da arquitetura do sistema desenvolvido. Assim as seguintes secções são referentes às três fases do sistema.

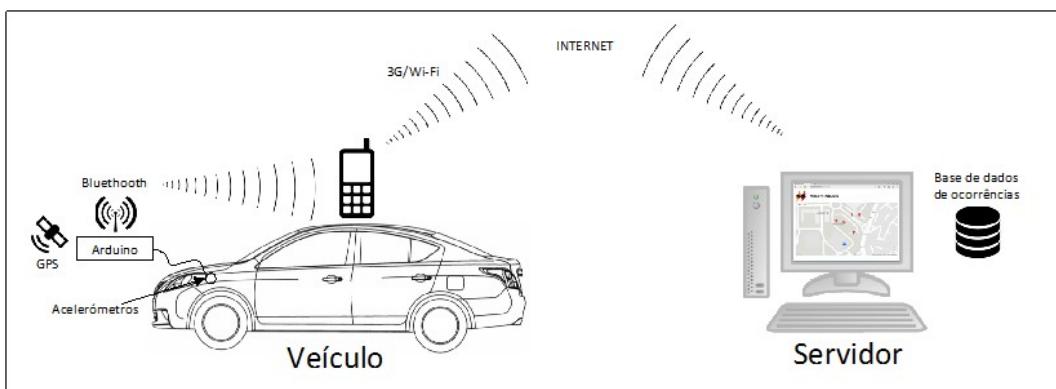


Figura 4.1: Arquitetura do sistema

### 4.1 Funcionamento do Arduino

O código Arduino está dividido em três partes: inicialização, setup e loop, sendo que a inicialização e setup são apenas executadas uma vez enquanto que loop é executada até que o Arduino seja desligado ou alguma parte do código force uma paragem. No apêndice A é possível ver o código desenvolvido para o Arduino, o qual é explicado de seguida.

- Na zona de declarações são incluídas as bibliotecas referentes ao acelerómetro, ao

cartão de memória e ao GPS. São também definidos os modos de escrita no cartão de memória, os portos de ligação de comunicação do GPS e criadas variáveis globais referentes ao GPS e ao acelerómetro. Além disso, são criadas variáveis globais de controlo do código, variáveis de passagem de parâmetros entre vários ciclos da função *loop* e declaração de constantes que serão utilizadas diversas vezes ao longo do código, facilitando a alteração do seu valor em todo o código, caso tal se mostre necessário.

- Na função *setup* são declaradas as frequências de comunicação entre o Arduino e o telemóvel e entre o GPS e o Arduino. São também inicializados dois *leds* como elementos de saída de informação e um botão como entrada de informação ao qual foi atribuído o estado de “desligado” a ambos os *leds*. Por fim, são executadas as funções de inicialização do leitor do cartão de memória e do acelerómetro, *SD\_init* e *AccelInit* respectivamente, ambas explicadas mais á frente.

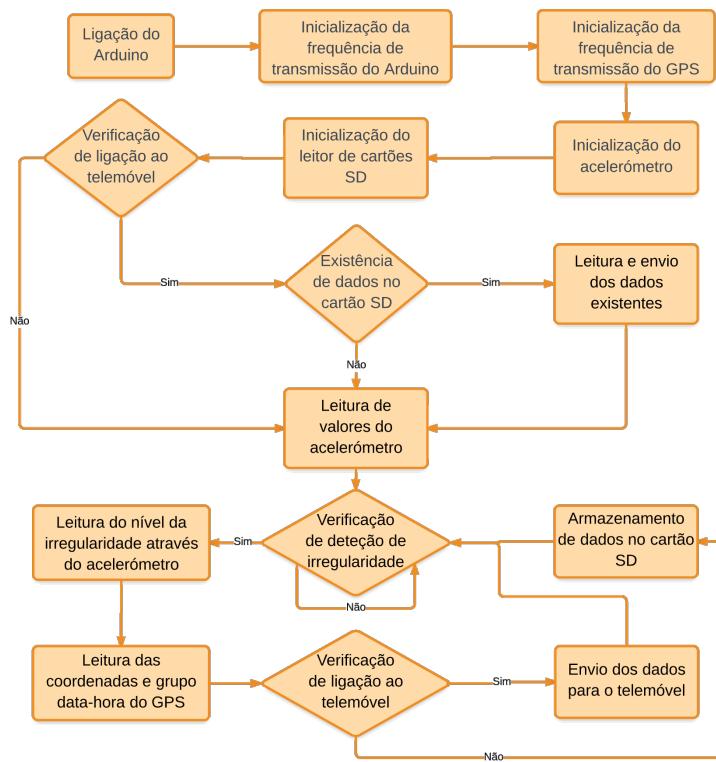


Figura 4.2: Fluxograma referente ao código Arduino

- A função *loop*, embora executada de uma forma constante, contém um código bastante simples mas importante. A função começa por verificar se existe informação disponível para leitura no módulo Bluetooth, relativa à ligação. Caso tal se verifique, essa informação é lida e comparada com a leitura anterior, para que seja possível determinar se a ligação ao módulo Bluetooth do Arduino sofreu alguma alteração, nomeadamente, quanto à ligação com a aplicação desenvolvida para esta dissertação. Na hipótese desta alteração se verificar, é avaliada se a ligação foi estabelecida

ou cortada entre o Arduino e o telemóvel, sendo que, no caso de ser estabelecida, é executada a função ReadSD para leitura do cartão de memória e um *led* de controlo é aceso. Caso contrário, o mesmo *led* de controlo é desligado. De seguida são lidos os valores de aceleração dos eixos X, Y e Z, provenientes do acelerómetro, armazenados e comparados com a leitura anterior para verificação da existência de alguma irregularidade no asfalto. A maneira como esta comparação é realizada será explicada mais à frente. Caso seja detectada alguma irregularidade, um dos *leds* de controlo pisca e são armazenados, numa variável **S**, os valores do nível da irregularidade, bem como o grupo data-hora proveniente do GPS. Seguidamente é verificada a ligação entre o Arduino e a aplicação e caso esta exista, os valores de **S** são enviados directamente para o telemóvel. Se a ligação não existir, os valores de **S** são transferidos para o cartão de memória para que possam ser transmitidos para o telemóvel assim que exista uma ligação estável com o mesmo. Por fim, os valores actuais do acelerómetro são armazenados para que possam ser comparados na próxima execução da função *loop*.

## 4.2 Funções adicionais

### 4.2.1 SD\_init

Esta função serve para inicializar o leitor de cartões SD. Nesta é chamada a função *begin* da biblioteca *SD* e verificada a resposta dessa mesma função. Caso a resposta seja o valor 10, um *led* de controlo pisca para que o utilizador saiba que a inicialização ocorreu com sucesso e é verificada a existência de um ficheiro com informação previamente armazenada sobre irregularidades detetadas anteriormente. Caso exista esse ficheiro, este é aberto de modo a que se possam juntar novos valores de irregularidades detetadas, caso contrário, o ficheiro é criado para futuros armazenamentos.

```

91 void SD_init() {
92   if (SD.begin(10)) {
93     digitalWrite(3, HIGH);
94     delay(500);
95     digitalWrite(3, LOW);
96     delay(500);
97   }
98 }
```

Figura 4.3: Código da função SDInit

### 4.2.2 WriteSD

Para que o cartão de memória possa armazenar dados é apenas necessário abrir um ficheiro em modo escrita. A função *open* da biblioteca do SD está desenhada de forma a que, quando é pedido para abrir um ficheiro, este seja procurado no cartão de memória e caso não exista nenhum ficheiro com esse nome, então é criado um novo ficheiro vazio, com o nome inserido e posteriormente aberto. Para que possa ser mais fácil ao utilizador saber se a abertura e escrita do ficheiro foi feita com sucesso, um *led* pisca duas vezes. No final resta apenas fechar o ficheiro para que tudo fique guardado devidamente.

```
100 * void WriteSD (String aux) {
101   File fich = SD.open("datalog.txt", FILE_WRITE);
102   if (fich) {
103     digitalWrite(3, HIGH);
104     delay(500);
105     digitalWrite(3, LOW);
106     fich.print(aux);
107     fich.close();
108   }
109   else {
110     digitalWrite(3, HIGH);
111     delay(100);
112     digitalWrite(3, LOW);
113     delay(100);
114     digitalWrite(3, HIGH);
115     delay(100);
116     digitalWrite(3, LOW);
117     delay(100);
118   }
119 }
```

Figura 4.4: Código da função WriteSD

### 4.2.3 ReadSD

Semelhante à função de escrita, a função de leitura tenta abrir um ficheiro com o nome pretendido, embora neste caso, se o ficheiro desejado não existir, não é criado um novo pois se este não existir significa que não existem dados pendentes para envio. Quando um ficheiro é lido com sucesso, os seus valores são enviados pela porta série do Arduino para o telemóvel, o ficheiro é fechado e eliminado, de modo a que não sejam enviados dados duplicados. O código está protegido de modo a que os dados sejam apenas enviados para um telemóvel com a aplicação desenvolvida pois previamente foi recebida uma mensagem de controlo enviada pela própria aplicação, informando o estado da ligação, evitando que os dados sejam enviados para ligações desconhecidas.

```

121 void ReadSD () {
122     File myFile = SD.open("datalog.txt");
123     if (myFile) {
124         char readChar;
125         while (myFile.available()) {
126             readChar = myFile.read();
127             Serial.print(readChar);
128         }
129         myFile.close();
130         SD.remove("datalog.txt");
131     }
132 }
```

Figura 4.5: Código da função ReadSD

#### 4.2.4 AccelInit

Nesta função o acelerómetro é ligado e são determinados os valores de sensibilidade do mesmo. É também feita a inicialização de valores para queda livre e batida que não são utilizados na dissertação. Esta funcionalidade foi mantida para possíveis aplicações futuras e uma deteção mais pormenorizada das irregularidades. Seguidamente, um *led* de controlo pisca para o utilizador saber que a inicialização foi bem sucedida.

```

74 void AccelInit() {
75     adxl.powerOn();
76     adxl.setRangeSetting(4);
77     adxl.setSpiBit(0);
78     adxl.setTapDetectionOnXYZ(0, 0, 1);
79     adxl.setTapThreshold(250);
80     adxl.setTapDuration(40);
81     adxl.setFreeFallThreshold(7);
82     adxl.setFreeFallDuration(30);
83     adxl.FreeFallINT(1);
84     adxl.singleTapINT(1);
85     digitalWrite(3, HIGH);
86     delay(500);
87     digitalWrite(3, LOW);
88     delay(500);
89 }
```

Figura 4.6: Código da função AccelInit

#### 4.2.5 Accel

Na função *Accel* é realizado o processamento dos valores de aceleração dos três eixos do acelerómetro para determinar a classificação da irregularidade detetada. Embora o

processo seja simples, é eficaz e semelhante ao método Z-DIFF apresentado em Mednis et al. 2011, mas aplicado aos três eixos de aceleração em simultâneo. O método baseia-se em detetar a existência de desvios superiores a um limite previamente determinado, tendo sido considerados valores múltiplos de 100 nesta dissertação. Assim, é feita uma subtração de valores sucessivos de aceleração dos eixos e avaliado o valor absoluto dessa diferença. O valor atribuído à irregularidade é o mais baixo dos três desvios. Na figura 4.7 é possível ver o gráfico de uma deteção, em que o nível atribuído foi 4.

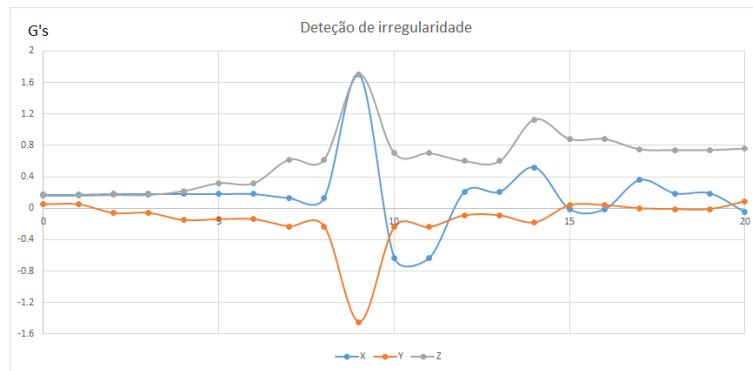


Figura 4.7: Gráfico de uma irregularidade detetada

#### 4.2.6 displayGPS

A função *displayGPS* é sem dúvida a mais elaborada nesta secção do trabalho devido aos dados enviados pelo GPS, muito extensos e, neste caso, não necessários, além de se encontrarem num formato não muito fácil de separar. A função é chamada dentro da função *loop* de modo a registar tanto o conjunto data-hora como as coordenadas em que a irregularidade foi detetada. Quando é chamada, faz uma leitura dos valores que o sensor GPS recebe dos satélites e percorre-os até encontrar o texto “GPRMC”, a partir do qual vem toda a informação necessária e armazena todos esses valores num vetor de strings (sequências de caracteres). O passo seguinte consiste em percorrer o vetor e retirar apenas os segmentos relevantes, pela ordem desejada, neste caso, começando pela latitude e longitude, seguindo-se a data e por fim a hora. Quando estes quatro parâmetros são armazenados, a função muda o valor de uma variável de controlo e o código prossegue.

```
$GPRMC,111636.932,A,2447.0949,N,12100.5223,E,000.0,000.0,030407,,,A*61<CR><LF>
          Hora      Latitude    Longitude      Data
```

Figura 4.8: Código da função vetorGPS

#### 4.2.7 FTOA - Float to ASCII

Dada a forma em que os valores das coordenadas do GPS são recebidos, é necessária fazer uma conversão para um formato mais simples de transmitir e como o processo de conversão é executado diversas vezes, foi criada uma função que evita repetição do código. Os dados enviados pelo GPS, relativos às coordenadas vêm no formato DDS – Degree Decimal Minutes (Graus e Minutos Decimais em Português) e apresentam o formato (1) da figura 4.9. Tal como se pode verificar, existem seis parâmetros neste formato, nomeadamente graus, minutos e hemisfério, três para a latitude e outros três para a longitude. Um formato mais simples é o DD – Decimal Degree (Graus Decimais em Português), também apresentado na figura 4.9, no ponto (2). Neste formato existem apenas quatro parâmetros (graus e hemisfério) que podem ser reduzidos a dois, se forem considerados graus negativos, representando o hemisfério a que a latitude ou longitude se referem, sendo assim necessário enviar menos dados para representar a mesma quantidade de informação, tornando o processo mais rápido. A conversão de DDS para DD é simples, sendo apenas necessário dividir a parte dos minutos decimais por 60 (convertendo minutos para graus) e somar esse valor aos graus já existentes. Como os valores que são recebidos do GPS vêm em *strings*, é necessário convertê-los para formato numérico, através da função `atoi` (ASCII to Integer, ASCCI para inteiro em português) que já existe nas bibliotecas do Arduino e posteriormente voltar a converter para formato de *string*. Apesar de ser possível escrever um valor numérico numa *string*, quando esta operação é feita, apenas duas casas decimais são utilizadas e para que as coordenadas possam apresentar um elevado grau de precisão são necessárias quatro casas decimais, sendo assim necessária a criação desta nova função de modo a reter tantas casas decimais quanto as desejadas.

**DDS - DD° MM.MM** (1)

**DD - DD.DD** (2)

Figura 4.9: Formatos de apresentação de coordenadas

### 4.3 Funcionamento do Android

Tal como explicado na secção 3.7, o método escolhido para a programação Android foi a MIT App Inventor 2, no website <http://ai2.appinventor.mit.edu>. Por correr num web browser, todo o processamento e compilação de código é feito no servidor e não no computador local, tornando o processo de criação mais rápido. Além disso, a sua programação por blocos torna o processo muito mais comprehensível uma vez que os blocos existentes mostram quais as funcionalidades de cada tipo de dados, desde números a vetores. Este modo de programação mostrou-se muito prático dada a inexperiência de desenvolvimento de aplicações para telemóvel. A existência de apenas quatro botões torna

esta aplicação bastante fácil de utilizar, pois cada botão tem a sua função devidamente identificada, tal como é possível ver na figura 4.10.



Figura 4.10: Aspetto da aplicação Smart Shock

A base do código da aplicação é um ciclo *while* que corre desde que a aplicação seja executada, semelhante à função *loop* do Arduino. Dentro deste ciclo existem duas condições: uma que verifica se o GPS do telemóvel está a detetar coordenadas e outra que verifica a ligação ao módulo *Bluetooth* do Arduino, bem como um procedimento que verifica a existência de dados armazenados no telemóvel para futuro envio para a base de dados. O GPS existente no telemóvel é utilizado para que seja possível reportar irregularidades existentes no asfalto mesmo que não se esteja a conduzir, possibilitando todos os transeunte adicionar informações sobre o local em que se encontram. Sempre que são detetadas coordenadas disponíveis, um botão com o texto “Shock” fica ativo e sempre que é pressionado são armazenadas as coordenadas atuais, bem como a data e hora presentes para um posterior envio para a base de dados. Para que estas ocorrências possam ser diferenciadas das detetadas pelo acelerómetro montado no Arduino, o código que lhe é atribuído tem um valor único, facilitando assim a diferenciação entre os dois tipos de irregularidades.

No que toca à ligação a um módulo *Bluetooth*, sempre que esta existe, é enviado um código para o dispositivo ao qual foi feita a ligação. Este código serve para o Arduino saber que o dispositivo que se encontra ligado tem a aplicação a correr, evitando assim que os dados recolhidos por este sejam enviados para um dispositivo desconhecido. Depois de ser feita a ligação, o código fica a “escutar” o Arduino até que o mesmo envie alguma informação para o telemóvel e quando tal acontece, os vários conjuntos de dados recebidos são escritos e organizados numa lista temporária. Após a divisão ser feita, a lista é percorrida e os dados são enviados para o cartão de memória do telemóvel ou para a base de dados, dependendo da existência de ligação Wi-Fi. Esta ligação é verificada a partir de um erro produzido automaticamente pelo sistema Android, sendo emitida uma mensagem de erro com um código específico. O envio de dados para a base de dados é

feito através do método *POST*, tornando possível ocultar os dados enviados, sendo esta uma vantagem no caso de no futuro serem desenvolvidas contas de utilizador em que certas informações devem permanecer confidenciais. Se a ligação à Internet não existir, os dados são armazenados no telemóvel para um futuro envio, sendo necessário pressionar um botão existente na aplicação. Este botão serve também para informar o utilizador que tem dados armazenados no dispositivo, uma vez que o botão nem sempre se encontra ativo.

Quanto à seleção de dispositivos *Bluetooth*, é mostrada uma lista dos dispositivos com que o telemóvel está emparelhado. No caso de ser a primeira vez que um determinado telemóvel se liga ao Arduino, é necessário fazer o emparelhamento fora da aplicação, como para qualquer outro dispositivo. Quando a aplicação é terminada, é enviada uma mensagem para o Arduino de modo a que este saiba que não é possível enviar mais informação sobre irregularidades e armazene todas as deteções nesse cartão de memória. Na figura 4.11 é possível compreender qual o ciclo de funcionamento da aplicação, sendo que no lado esquerdo se encontra o ramo principal e o ciclo que está sempre em funcionamento enquanto a aplicação estiver aberta. Já do lado direito é possível ver quais os procedimentos realizados quando os botões “Shock” e “Enviar” são pressionados.

No apêndice B são apresentados alguns dos blocos construídos para o funcionamento da aplicação para uma melhor compreensão do funcionamento do App Inventor.

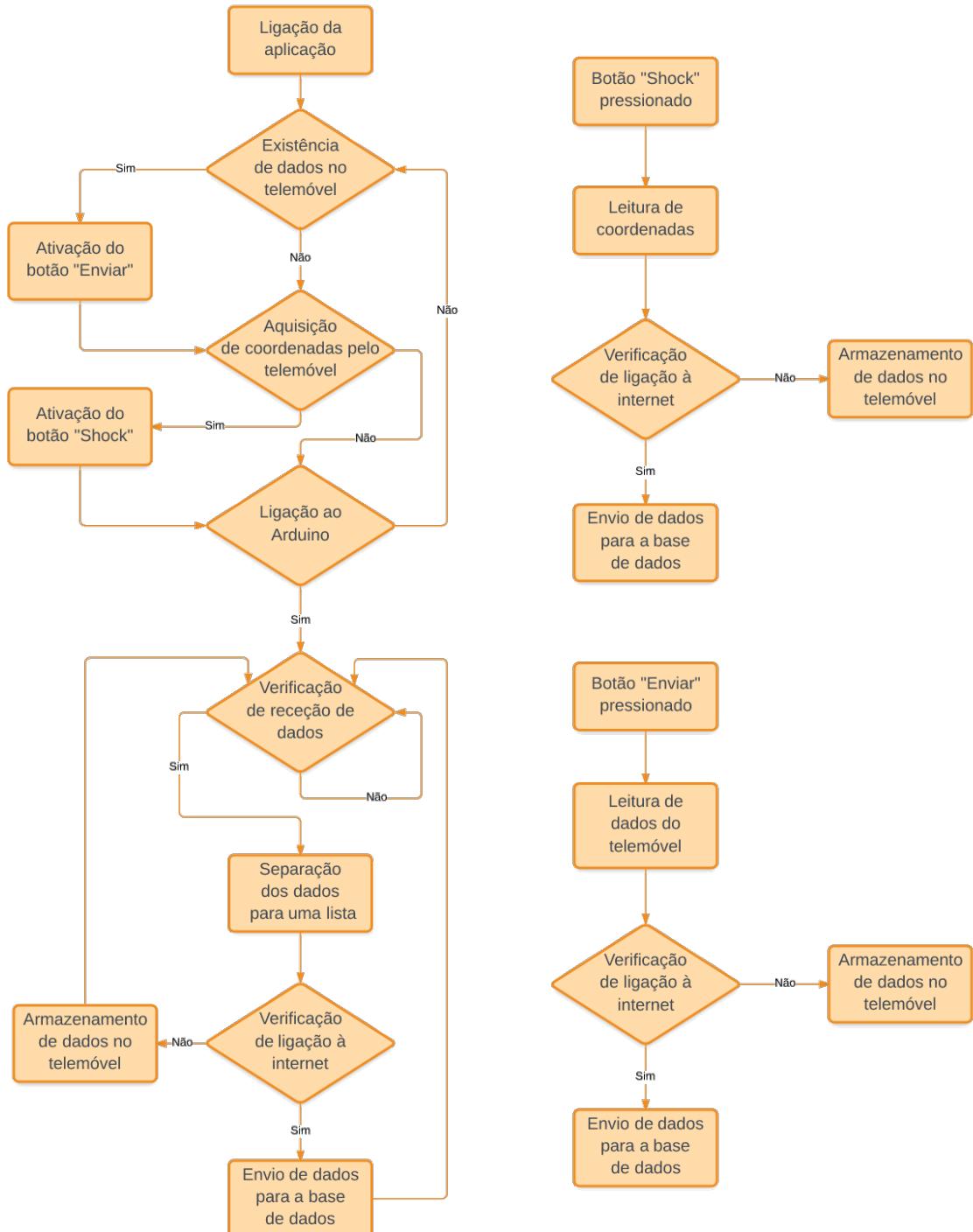


Figura 4.11: Fluxograma referente ao código Android

## 4.4 Base de dados

De modo a que seja possível consultar as irregularidades, foi necessário desenvolver uma base de dados onde ficassem armazenadas as coordenadas, data, hora e intensidade da irregularidade detectadas. Esta base de dados foi desenvolvida em MySQL e contém cinco campos, sendo eles o ID (campo de identificação de cada entrada), a intensidade da irregularidade, a latitude e a longitude da irregularidade e também o conjunto data-hora, utilizando apenas uma variável. A pode ainda ser consultada e alterada por um administrador da mesma, caso este ache necessário de modo a inserir novas funcionalidades no projeto. Para este projeto foi utilizado o xampp, uma plataforma que permite criar um servidor apache local, hospedado na máquina do utilizador. Deste modo, tanto a base de dados como o servidor estão a correr localmente, facilitando os acessos e testes, visto que estes se podem realizar sem que seja necessária qualquer tipo de ligação à internet. Esta plataforma permite também ao utilizador criar o seu próprio servidor, uma vez que pode permitir o acesso à sua máquina, desde que torne o acesso público, abrindo um acesso na rede em que está ligado, a partir de uma exceção da firewall.

## 4.5 WebSite

No apêndice C é possível ver o código das páginas referidas no texto que se segue. Através desse texto é possível recriar todo o sistema desenvolvido.

### 4.5.1 home.php

Esta é a página principal da parte *web* desta dissertação. Nela é possível ver-se o logótipo e o nome utilizados neste projeto e também um mapa que contém todas as irregularidades detetadas ou inseridas manualmente, referindo a intensidade das mesmas. Para que essas irregularidades possam ser mostradas, é percorrida toda a base de dados através de uma *query* (pesquisa, em português) de MySQL em que são devolvidos os valores de latitude, longitude e intensidade. Depois, utilizando a API Google Maps, todos os pontos são marcados no mapa, sendo que no seu sinalizador está indicada a intensidade da irregularidade, como se pode ver na figura 4.12. A integração com a API é bastante simples, uma vez que é apenas necessário conter um array com as coordenadas dos pontos, bem como a etiqueta que se quer atribuir a cada ponto marcado. Por fim, basta aceder à API utilizando uma chave de acesso que pode ser obtida gratuitamente através do Google Maps. No caso de existirem várias irregularidades muito próximas fisicamente, a interface agrupa-as automaticamente, informando quantas se encontram nessa zona. A cor destes agrupamentos é alterada consoante o número de itens aglomerados, tornando assim mais fácil a deteção de locais com elevado número de ocorrências.

#### 4.5.2 store.php

A página *store* não contém nenhuma componente gráfica para consulta de informação pois destina-se apenas ao armazenamento de dados na base de dados. Quando o telemóvel envia dados, é este o destino e como tal, tem que existir algum processamento dos dados recebidos. A informação recebida é exemplificada na figura 4.13 e vem dividida em quatro *strings*: força, latitude, longitude e data-hora, sendo portanto necessário fazer uma conversão para o formato correto e consequente armazenamento em variáveis locais. De seguida é feita uma *query MySQL* para que seja possível fazer o armazenamento na base de dados, sendo emitida uma mensagem de controlo para o telemóvel de modo a dar informação sobre o sucesso ou falha deste armazenamento.

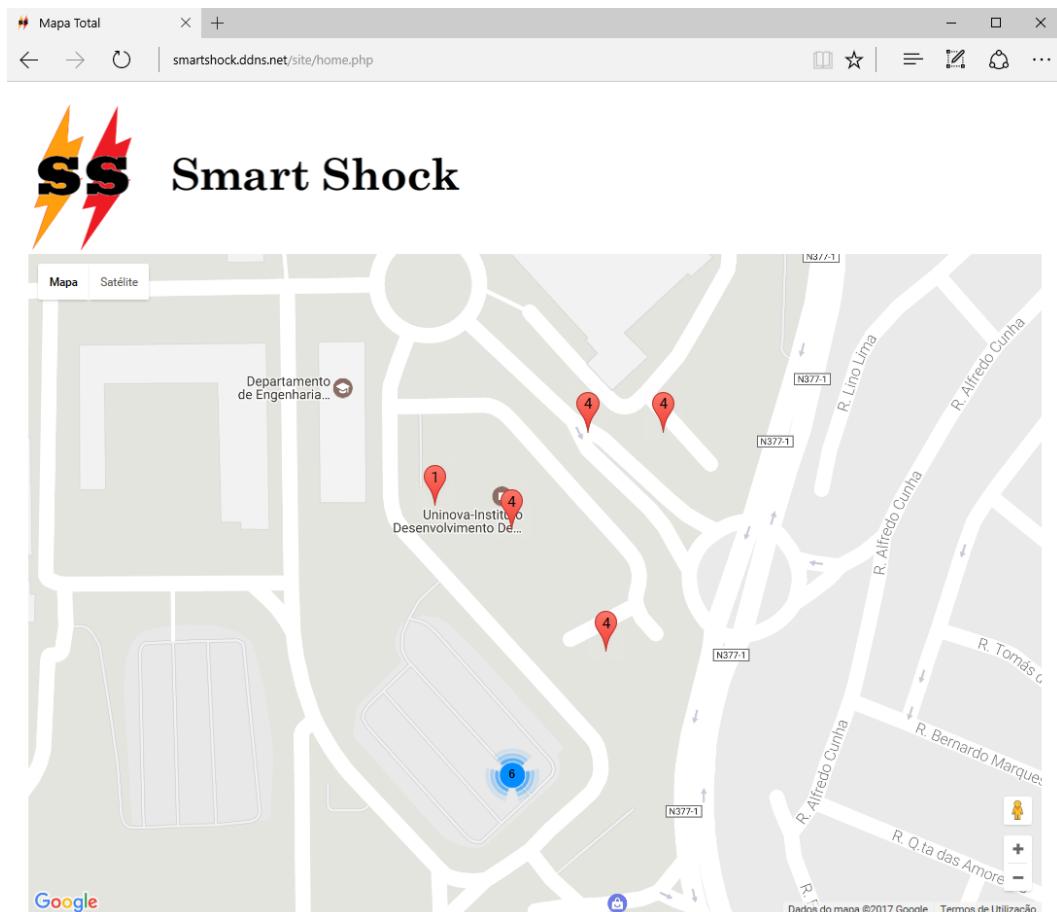


Figura 4.12: Página web *home.php*

#### 4.5.3 listLocations.php

Esta página web é bastante semelhante à *home.php*, sendo que aqui apenas é mostrada a localização da irregularidade selecionada na página *listlocations.php* de modo a ser mais

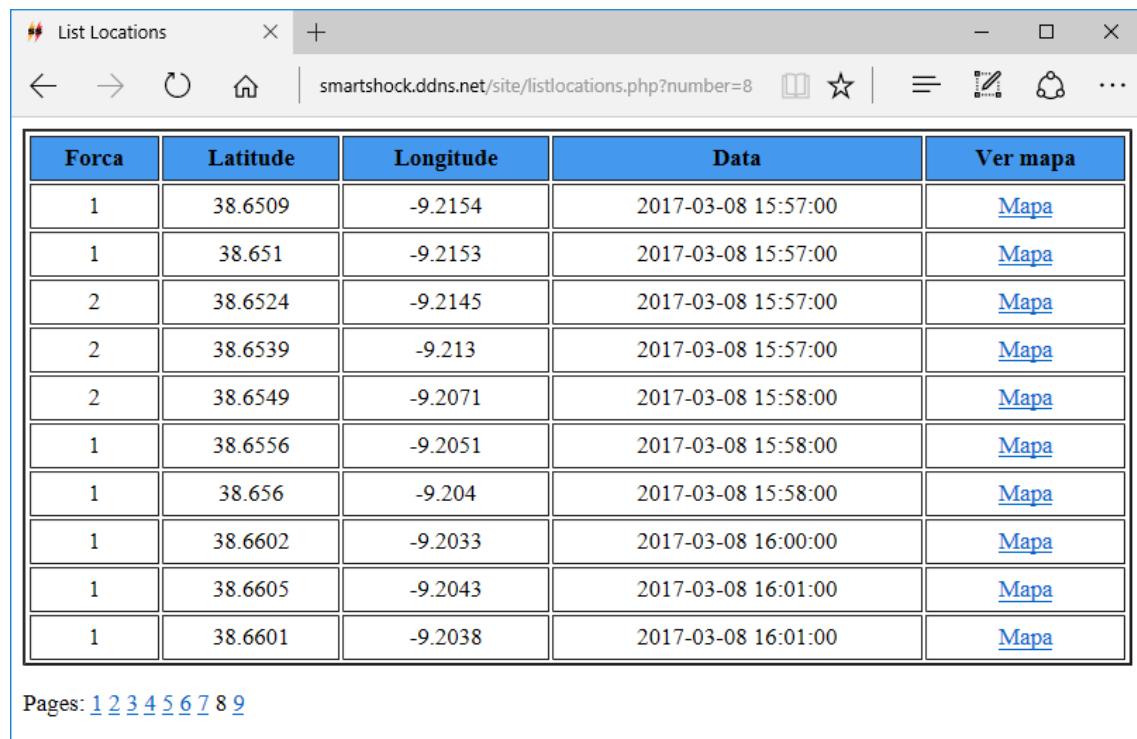
fácil descobrir a localização da irregularidade quando esta se encontra agrupada com outras irregularidades existentes nas proximidades, como acontece na figura 4.12.

**3 +38.6601 -09.2047 2017-02-15 14:37**

Figura 4.13: Informação tipo adicionada na base de dados

#### 4.5.4 showLocation.php

Esta página tem a finalidade de consultar todas as irregularidades num formato numérico, ao invés da página *home.php*, em que as irregularidades são apresentadas num mapa. Tal como apresentado na figura 4.14, nesta página é possível consultar toda a informação referente a uma irregularidade, incluindo a data e hora em que foi detetada. Para que seja necessário visualizar a lista de irregularidades, é necessário colocar no endereço o número da página que se deseja visualizar, sendo o valor pré-definido “1”. Cada uma destas páginas mostra dez entradas da lista de irregularidades e no fundo da página é mostrado um navegador para as diferentes páginas. Alinhado com cada irregularidade, existe uma hiperligação para a página *showLocation.php* em que é mostrado no mapa a localização da irregularidade selecionada para uma melhor compreensão dos valores de latitude e longitude.



The screenshot shows a web browser window titled "List Locations". The address bar displays "smartshock.ddns.net/site/listlocations.php?number=8". The main content is a table with the following data:

Forca	Latitude	Longitude	Data	Ver mapa
1	38.6509	-9.2154	2017-03-08 15:57:00	<a href="#">Mapa</a>
1	38.651	-9.2153	2017-03-08 15:57:00	<a href="#">Mapa</a>
2	38.6524	-9.2145	2017-03-08 15:57:00	<a href="#">Mapa</a>
2	38.6539	-9.213	2017-03-08 15:57:00	<a href="#">Mapa</a>
2	38.6549	-9.2071	2017-03-08 15:58:00	<a href="#">Mapa</a>
1	38.6556	-9.2051	2017-03-08 15:58:00	<a href="#">Mapa</a>
1	38.656	-9.204	2017-03-08 15:58:00	<a href="#">Mapa</a>
1	38.6602	-9.2033	2017-03-08 16:00:00	<a href="#">Mapa</a>
1	38.6605	-9.2043	2017-03-08 16:01:00	<a href="#">Mapa</a>
1	38.6601	-9.2038	2017-03-08 16:01:00	<a href="#">Mapa</a>

Pages: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#)

Figura 4.14: Página web *listlocations.php*

## 4.6 Montagem do sistema

O sistema final foi montado numa PCB desenhada para o Arduino Mega, apesar de ter sido utilizado um Arduino Uno, graças à semelhança dos equipamentos. A utilização desta PCB deve-se ao tamanho da mesma, sendo assim possível espaçar os componentes mais facilmente, evitando possíveis interferências entre eles e que possibilita uma melhor gestão das ligações. Foi também necessário utilizar dois reguladores de tensão para converter a tensão de entrada de 12V do automóvel para 5.0V e 3.3V. Apesar do Arduino já conter reguladores semelhantes, existe uma limitação de 200mA na corrente de alimentação, a qual não é suficiente para alimentar o sistema. São também necessários dois reguladores visto existirem componentes que necessitam de ser alimentados a 5.0V e outros a 3.3V pelo que o regulador LM7805 faz a conversão de 12V para 5.0V e o regulador LM3940 faz a conversão de 5.0V para 3.3V. As figuras 4.15 e 4.16 representam um esquema da montagem e a montagem real do sistema, respetivamente. A montagem da figura 4.16 foi então instalada num veículo de testes para validação dos resultados, abaixo do amortecedor para evitar a deterioração dos dados recolhidos, como mostra a figura 4.17.

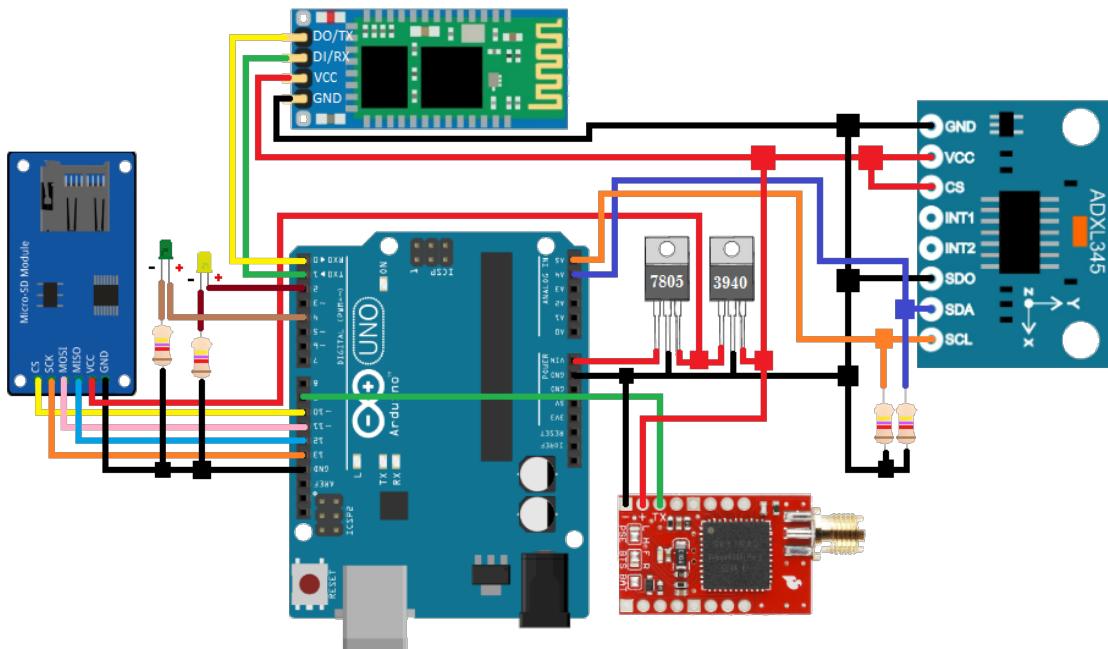


Figura 4.15: Esquemático do sistema

## 4.7 Validação de resultados

De modo a fazer uma validação do sistema e dos seus resultados, o sistema foi parcialmente montado num veículo de testes, mantendo o acelerómetro no seu interior para que fossem inseridas irregularidades manualmente, apesar do bom estado do pavimento. Este

#### 4.7. VALIDAÇÃO DE RESULTADOS

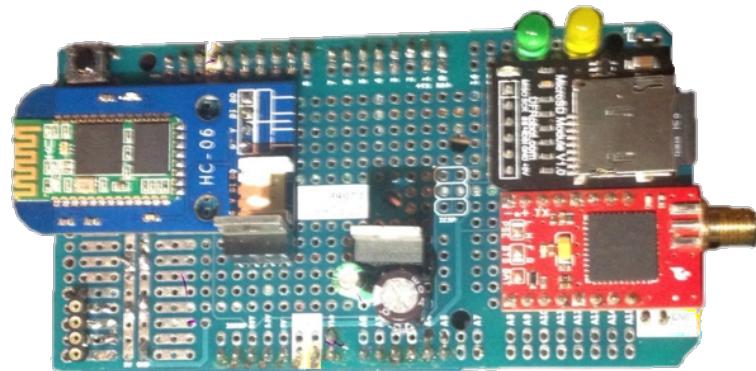


Figura 4.16: Montagem real do sistema



Figura 4.17: Instalação do sensor no veículo de testes

este serviu para confirmar o bom funcionamento de todo o sistema, ou seja, o acelerómetro e o GPS, a comunicação Bluetooth e 3G e também a base de dados e web site para consulta de dados inseridos. Na figura 4.18 é possível consultar o percurso deste teste.

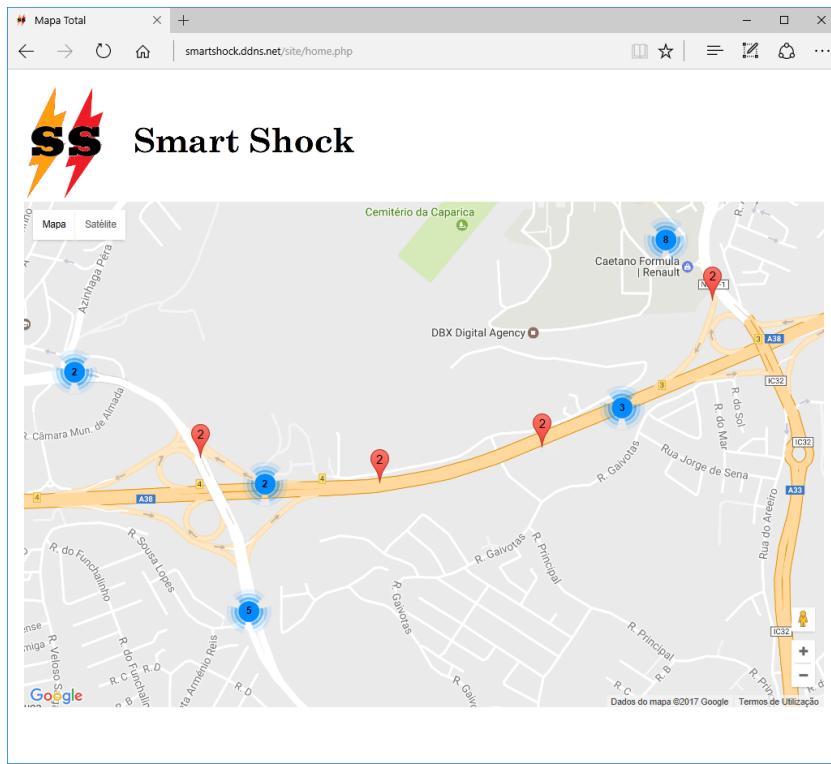


Figura 4.18: Percurso de validação de resultados

Após concluir que todo o sistema estava a funcionar corretamente, o acelerómetro foi montado no automóvel e foram adquiridos dados ao longo de vários dias para que a base de dados ficasse mais completa. Apesar de não existir uma análise destes dados, o sistema mostrou funcionar em perfeitas condições, fazendo deteções regularmente sempre que o veículo passava por alguma irregularidade. Na análise do mapa, é possível verificar que algumas deteções se encontram fora das estradas, algo impossível de acontecer no cenário real. Ainda assim, o afastamento nunca é superior a 5 metros, a escala menor do mapa, sendo esse um valor aceitável neste tipo de sistemas pois à escala humana facilmente se identificam todas irregularidades que estejam a 5 metros de distância de um determinado ponto.

## 4.8 Resultados obtidos

Tal como esperado, o sistema apresentou resultados positivos quanto à deteção de irregularidades no asfalto. O acelerómetro é um sistema fiável no que diz respeito a este tipo de deteção e a sua calibração é praticamente inexistente tendo em conta a montagem proposta, tornando este sistema bastante fácil de usar e apelativo. O recetor GPS também

#### **4.8. RESULTADOS OBTIDOS**

---

apresentou resultados bastante fiáveis e consistentes ao longo de todos os testes feitos, mesmo aqueles em que o sistema não se encontrava totalmente montado.

A aplicação também mostrou ser útil como ponto intermédio de comunicação. O seu desenho simples torna muito intuitiva a utilização sendo apenas necessário selecionar o dispositivo com que se deseja comunicar. Para a comunicação à base de dados é apenas necessário que exista ligação à Internet, não existindo qualquer diferença quanto ao modo de ligação, seja ele Wi-Fi ou 3G. É ainda possível determinar que, através do App Inventor, a construção da aplicação foi bastante rápida e intuitiva, reduzindo significativamente o tempo de desenvolvimento e de teste da mesma.

Por fim, a base de dados fez um armazenamento de dados sem qualquer tipo de problemas. A configuração da mesma foi bastante fácil e a sua consulta em SQL é totalmente adequada aos requisitos deste projeto. A construção das páginas *web* à volta dos dados da aplicação serviu para que a consulta de dados fosse mais prática. A utilização do Google Maps e das suas funcionalidades possibilitou a utilização de um mapa interativo para que fosse mais direta a identificação da localização das irregularidades detetadas.



## CONCLUSÕES

Tal como esperado, o sistema que foi desenvolvido funciona e apresenta resultados bastante positivos. O sistema foi capaz de detetar irregularidades existentes no asfalto e classificá-las consoante a sua intensidade, recorrendo ao acelerómetro. Esta recolha de dados mostrou-se bastante precisa, graças ao local onde o acelerómetro foi instalado, fazendo com que o sistema possa ser instalado em diferentes veículos sem que seja necessário qualquer tipo de calibração, dado que as vibrações sentidas pelas rodas são diretamente transmitidas para o acelerómetro. Outro fator importante é o custo do sistema, que é bastante reduzido e que pode ser ainda mais conveniente, caso seja feita a sua construção em série. Como o sistema é compacto e tem um tamanho reduzido, a sua instalação é muito transparente e rápida na maioria dos casos e não causa qualquer tipo de alteração visual no veículo. Apesar da aplicação não se encontrar disponível nas principais lojas do género, a sua instalação é bastante acessível graças à possibilidade de transferência pelo browser do telemóvel. A sua utilização é também bastante simples, com um número reduzido de botões que estão devidamente identificados quanto à sua função. No que diz respeito às páginas *web*, a página de consulta de irregularidades é bastante simples, para uma navegação natural entre irregularidades, sendo bastante explícita e acessível a todos os utilizadores. A página principal do sistema mostra a informação essencial das deteções num mapa, sendo muito intuitiva a sua utilização, bem como a possibilidade de concluir quais as áreas que afetam mais utilizadores.

### 5.1 Trabalho futuro

Existem sempre alguns detalhes a melhorar nos sistemas existentes e este não falha à regra. Apesar de detetar irregularidades, a avaliação das irregularidades detetadas não permite a diferenciação entre uma depressão no asfalto ou uma protuberância. Ao nível da

aplicação, existem algumas falhas no armazenamento, maioritariamente devido a falhas na comunicação Bluetooth, uma vez que não é possível determinar quais pacotes que são enviados e não são recebidos. O armazenamento de dados no telemóvel, embora não ocupe muita memória, pode sempre ser otimizado, alterando o tipo de variáveis a guardar, bem como fazendo a sua compressão. Relativamente às páginas *web*, são bastante funcionais e práticas mas não mostram grande apelo visual, sendo apenas uma ferramenta de consulta dos dados registados. Idealmente será também interessante que seja feita uma parceria com uma empresa que tenha uma grande frota de veículos, uma vez que a validação de dados e o seu grau de relevância se torna mais facilmente detetável se vários utilizadores fizerem a deteção de irregularidades. Esta ideia poderá ser promovida junto de entidades responsáveis pela gestão das vias, através de incentivos aos seus utilizadores.

## 5.2 Conferências

Um ponto positivo a salientar neste projeto foi a sua apresentação na Green Business Week, sendo este o maior evento nacional de transferência de conhecimento, com especial foco nas área da eficiência energética, o aproveitamento de águas e as cidades inteligentes. Devido ao especial foco deste projeto nas cidades inteligentes, foram contactadas diversas entidades, em especial câmaras municipais, de forma a divulgar mais o trabalho desenvolvido para uma possível parceria futura de modo a melhorar o sistema. A receptividade da apresentação feita foi bastante positiva e é expectável que num futuro próximo surjam apoios para um maior desenvolvimento desta solução. Na figura 5.1 é possível ver o poster apresentado na Green Business Week.

## 5.2. CONFERÊNCIAS

**Smart Shock for better roads**

José Manuel Fonseca      Miguel Batista Prego

CA3 - Computational Intelligence Research Group – CTS/UNINOVA  
 Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa

**Introdução**

- O mau estado das vias rodoviárias causa desconforto na população, despesas indesejadas na manutenção dos veículos e aumenta o risco rodoviário
- A participação da comunidade na gestão dos recursos locais é um fator de satisfação social
- Existem já alguns esforços no sentido de envolver a comunidade na gestão dos seus problemas (exemplo: PRP<sup>1</sup>)
- A detecção e inventariação das irregularidades no asfalto é fastidiosa e demorada pelo que a sua atualização em tempo real é quase impossível sem recorrer a processos automáticos
- A criação de estruturas camarárias para este efeito é dispendiosa, demorada e com uma eficiência discutível
- A criação de um sistema de baixo custo que permita a participação dos cidadãos no registo das irregularidades no asfalto pode proporcionar a criação de uma base de dados em tempo "quase real" que permita uma gestão optimizada das estruturas camarárias e um elevado grau de satisfação aos munícipes
- A coleta de dados por uma "comunidade" de angariadores de informação pode ser a solução desde que corretamente gerida

<sup>1</sup><http://www.risco-rodovario.prp.pt/>






**Objetivos**

- Criação de um equipamento de baixo custo, não intrusivo, que instalado em viaturas normais permita registar e classificar as irregularidades no asfalto através das vibrações na suspensão do veículo
- Constituição de uma comunidade de utilizadores (particulares ou profissionais) que instalem nas suas viaturas este equipamento para participação no reencenseamento das irregularidades em troca de benefícios a definir (estacionamento grátis, redução no IUC, etc)
- Construção de uma base de dados alargada, com atualização contínua e permanente das irregularidades registadas que permita a gestão das mesmas e a extração de conhecimento para a otimização dos recursos envolvidos na manutenção das rodovias
- Oferta à comunidade de uma possibilidade de participação ativa na comunicação de deficiências nas rodovias aumentando o sentido de cidadania e aproximando o poder local da população

**Arquitectura**

- Desenho de um equipamento com custo reduzido, baixo consumo, sem despesas de comunicação (trando partido da comunicação Wi-Fi e/ou 3G normalmente existente nos telemóveis atuais)
- Portal de gestão de irregularidades detetadas acessível online com possibilidade de visão adequada ao gestor e ao utilizador normal




**Protótipo desenvolvido**



Aspecto do portal demonstrativo do sistema Smart Shock mostrando algumas das irregularidades detectadas

Piso vermelho com numero - irregularidade e grau de severidade  
 Círculo azul com numero - zona de concentração com número de deteções menor que 10  
 Círculo amarelo com numero - zona de concentração com número de deteções entre 11 e 100  
 Círculo vermelho com numero - zona de concentração com número de deteções superior a 100 (não exibe-se na figura)



Protótipo funcional do sistema de aquisição Smart Shock



Instalação do sensor no veículo de teste

**Conclusões**

- O sistema desenvolvido permite atingir os objetivos pretendidos – colectar informação sobre as irregularidades no asfalto
- A produção em série deste tipo de equipamentos pode tornar os custos da sua produção muito convenientes
- É viável a utilização de qualquer viatura sem adaptações ou modificações significativas
- A instalação do sistema é, na maioria dos casos, muito fácil e rápida
- O portal poderá suportar a gestão da informação de forma a optimizar as intervenções de reparação dos serviços responsáveis

**Agradecimentos**

Este trabalho foi parcialmente financiado pelo programa FCT Strategic Program UID/EEA/00066/203 do UNINOVA, CTS






Contactos:

José Manuel Fonseca jmnf@uninova.pt	Miguel Batista Prego m.prego@campus.fct.unl.pt
--	---

Figura 5.1: Poster apresentado na Green Business Week



## BIBLIOGRAFIA

- Android is enabling opportunity.* URL: <http://www.android.com/everyone/enabling-opportunity/> (acedido em 07/01/2017).
- Arduino - Introduction.* URL: <http://www.arduino.cc/en/Guide/Introduction> (acedido em 13/01/2017).
- Chan, C. K., Y. Gao, Z. Zhang e N. Dahnoun (2014). "Implementation and evaluation of a pothole detection system on TI C6678 digital signal processor". Em: *EDERC 2014 - Proceedings of the 6th European Embedded Design in Education and Research Conference*.
- Chen, K., M. Lu, X. Fan, M. Wei e J. Wu (2011). "Road condition monitoring using on-board three-axis accelerometer and GPS sensor". Em: *Proceedings of the 2011 6th International ICST Conference on Communications and Networking in China, CHINACOM 2011*.
- Fouad, M. M., M. A. Mahmood, H. Mahmoud, A. Mohamed e A. E. Hassanien (2014). "Intelligent road surface quality evaluation using rough mereology". Em: *2014 14th International Conference on Hybrid Intelligent Systems*.
- He, Y., J. Wang, H. Qiu, W. Zhang e J. Xie (2011). "A research of pavement potholes detection based on three-dimensional projection transformation". Em: *Proceedings - 4th International Congress on Image and Signal Processing, CISP 2011*.
- Hegde, S., H. Mekali e G. Varaprasad (2015). "Pothole detection and inter vehicular communication". Em: *2014 IEEE International Conference on Vehicular Electronics and Safety, ICVES 2014*.
- How does the Global Positioning System (GPS) work ?* URL: <http://www.pocketgpsworld.com/howgpsworks.php> (acedido em 10/01/2017).
- Jang, J., A. W. Smyth, Y. Yang e D. Cavalcanti (2015). "Road surface condition monitoring via multiple sensor-equipped vehicles". Em: *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*.
- Kattan, A. e M. F. Aboalmaaly (2014). "A smartphone-cloud application as an aid for street safety inventory". Em: *Proceedings of the 11th International Conference on Electronics, Computer and Computation, ICECCO 2014*.
- Madli, R., S. Hebbar, P. Pattar e V. Golla (2015). "Automatic Detection and Notification of Potholes and Humps on Roads to Aid Drivers". Em: *IEEE Sensors Journal*.

## BIBLIOGRAFIA

---

- Mednis, A., G. Strazdins, R. Zviedris, G. Kanonirs e L. Selavo (2011). "Real time pothole detection using Android smartphones with accelerometers". Em: *2011 International Conference on Distributed Computing in Sensor Systems and Workshops, DCOSS'11*.
- Moazzam, I., K. Kamal, S. Mathavan, S. Usman e M. Rahman (2013). "Metrology and visualization of potholes using the microsoft kinect sensor". Em: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*.
- PHP: O que é o PHP? - Manual.* URL: [http://secure.php.net/manual/pt\\_BR/intro-whatis.php](http://secure.php.net/manual/pt_BR/intro-whatis.php) (acedido em 09/01/2017).
- Wi-Fi Alliance.* URL: <http://www.wi-fi.org/> (acedido em 07/01/2017).
- Yu, X. e E. Salari (2011). "Pavement pothole detection and severity measurement using laser imaging". Em: *IEEE International Conference on Electro Information Technology*.
- Zhang, Z., X. Ai, C. K. Chan e N. Dahoun (2014). "An efficient algorithm for pothole detection using stereo vision". Em: *IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*.



## CÓDIGO DO ARDUINO

```
#include <SparkFun_ADXL345.h>
#include <SD.h>
#include <SoftwareSerial.h>

#define FILE_WRITE (O_READ | O_APPEND | O_WRITE | O_CREAT)
SoftwareSerial ss (9, 8);
ADXL345 adxl = ADXL345();

const int sentenceSize = 80;
char sentence[sentenceSize];
int ctrl = 0;
int TAFS[6] = {0};
int ligado = 79;

void setup()
{
    Serial.begin(9600);
    ss.begin(9600);
    pinMode(2, INPUT);
    pinMode(3, OUTPUT);
    pinMode(4, OUTPUT);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    SD_init();
    AccelInit();
}
```

Figura A.1: Zona de inicialização e função setup

## APÊNDICE A. CÓDIGO DO ARDUINO

---

```
void loop()
{
    int ret = 0;

    while (Serial.available() > 0) {
        ligado = Serial.read();
        if (ligado == 73) {
            digitalWrite(3, HIGH);
            ReadSD();
        }
        if (ligado == 79) {
            digitalWrite(3, LOW);
        }
    }

    adxl.readAccel(&TAPS[3], &TAPS[4], &TAPS[5]);
    ret = Accel(TAPS[0], TAPS[1], TAPS[2], TAPS[3], TAPS[4], TAPS[5]);

    if (ret && (ctrl == 0)) {
        digitalWrite(4, HIGH);
        delay(300);
        digitalWrite(4, LOW);

        String S;
        S = S + ret;
        S = S + ' ';
        while (ctrl == 0)
            displayGPS(S);
        ctrl = 0;

        if (ligado == 79)
            WriteSD(S);
        if (ligado == 73)
            Serial.print(S);

        TAPS[3] = 0;
        TAPS[4] = 0;
        TAPS[5] = 0;
    }

    TAPS[0] = TAPS[3];
    TAPS[1] = TAPS[4];
    TAPS[2] = TAPS[5];
}
```

Figura A.2: Função loop

```
void AccelInit() {
    adxl.powerOn();
    adxl.setRangeSetting(4);
    adxl.setSpiBit(0);
    adxl.setTapDetectionOnXYZ(0, 0, 1);
    adxl.setTapThreshold(250);
    adxl.setTapDuration(40);
    adxl.setFreeFallThreshold(7);
    adxl.setFreeFallDuration(30);
    adxl.FreeFallINT(1);
    adxl.singleTapINT(1);
    digitalWrite(3, HIGH);
    delay(500);
    digitalWrite(3, LOW);
    delay(500);
}

void SD_init() {
    if (SD.begin(10)) {
        digitalWrite(3, HIGH);
        delay(500);
        digitalWrite(3, LOW);
        delay(500);
    }
}
```

Figura A.3: Funções accelInit e SD\_init

---

```

void WriteSD (String aux) {
    File fich = SD.open("datalog.txt", FILE_WRITE);
    if (fich) {
        digitalWrite(3, HIGH);
        delay(500);
        digitalWrite(3, LOW);
        fich.print(aux);
        fich.close();
    }
    else {
        digitalWrite(3, HIGH);
        delay(100);
        digitalWrite(3, LOW);
        delay(100);
        digitalWrite(3, HIGH);
        delay(100);
        digitalWrite(3, LOW);
        delay(100);
    }
}

void ReadSD () {
    File myFile = SD.open("datalog.txt");
    if (myFile) {
        char readChar;
        while (myFile.available()) {
            readChar = myFile.read();
            Serial.print(readChar);
        }
        myFile.close();
        SD.remove("datalog.txt");
    }
}

```

Figura A.4: Funções WriteSD e ReadSD

```

int Accel(int oldX, int oldY, int oldZ, int newX, int newY, int newZ) {
    if ( abs(oldX - newX) > 500 && abs(oldY - newY) > 500 && abs(oldZ - newZ) > 500 )
        return 5;

    else if ( abs(oldX - newX) > 400 && abs(oldY - newY) > 400 && abs(oldZ - newZ) > 400 )
        return 4;

    else if ( abs(oldX - newX) > 300 && abs(oldY - newY) > 300 && abs(oldZ - newZ) > 300 )
        return 3;

    else if ( abs(oldX - newX) > 200 && abs(oldY - newY) > 200 && abs(oldZ - newZ) > 200 )
        return 2;

    else if ( abs(oldX - newX) > 100 && abs(oldY - newY) > 100 && abs(oldZ - newZ) > 100 )
        return 1;

    else
        return 0;
}

```

Figura A.5: Função Accel

```
void getField(char* buffer, int index)
{
    int sentencePos = 0;
    int fieldPos = 0;
    int commaCount = 0;
    while (sentencePos < sentenceSize)
    {
        if (sentence[sentencePos] == ',')
        {
            commaCount++;
            sentencePos++;
        }
        if (commaCount == index)
        {
            buffer[fieldPos] = sentence[sentencePos];
            fieldPos++;
        }
        sentencePos++;
    }
    buffer[fieldPos] = '\0';
}

String FTOA(double aF) {
    int i, a;
    String S;
    F = F / 10;
    for (a = 0; a < 7; a++) {
        if (a == 2)
            S = S + ".";
        else if (F > 0) {
            i = F;
            F = (F - i) * 10;
            S = S + i;
        }
    }
    return S;
}
```

Figura A.6: Funções getField e FTOA



## CÓDIGO DA APLICAÇÃO

```
when [Clock1] .Timer
do
  if [0] = [LocationSensor1] .Latitude
    then [set Shock .Enabled to false]
    else [set Shock .Enabled to true]

  call [File1] .ReadFrom
    fileName [/FicheiroTeste.txt]

  if [BT_client] .IsConnected
    then call [BT_client] .SendText
      text [0]

  if [call [BT_client] .BytesAvailableToReceive] > [0]
    then set [global Receiver] to [join [get [global Receiver]] [call [BT_client] .ReceiveText] [numberOfBytes] [call [BT_client] .BytesAvailableToReceive]]
      set [global Lista] to [split [text] [get [global Receiver]] [at [=]]]
      remove list item [list] [get [global Lista]] [index [length of list] [list] [get [global Lista]]]
      if [length of list] [list] [get [global Lista]] = [1]
        then [set [global Init] to [0]]
        else [set [global Init] to [1]]

      if [length of list] [list] [get [global Lista]] ≠ [get [global Tamanho]]
        then for each [Var1] from [get [global Tamanho] + [1]] to [length of list] [list] [get [global Lista]] by [1]
          do [set [global VarStore] to [get [Var1]]]
            call [CriarStringValores]
            call [Envio]
          end
        end
      end
      set [global Tamanho] to [length of list] [list] [get [global Lista]]
    end
  end
end
```

Figura B.1: Ciclo principal da aplicação

## APÊNDICE B. CÓDIGO DA APLICAÇÃO

---

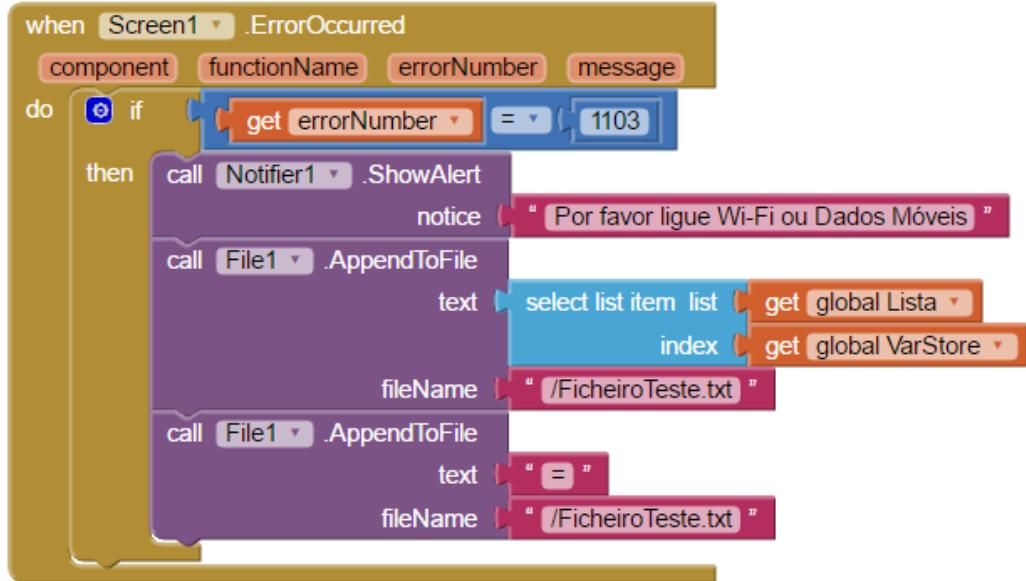


Figura B.2: Gestão de mensagens de erro

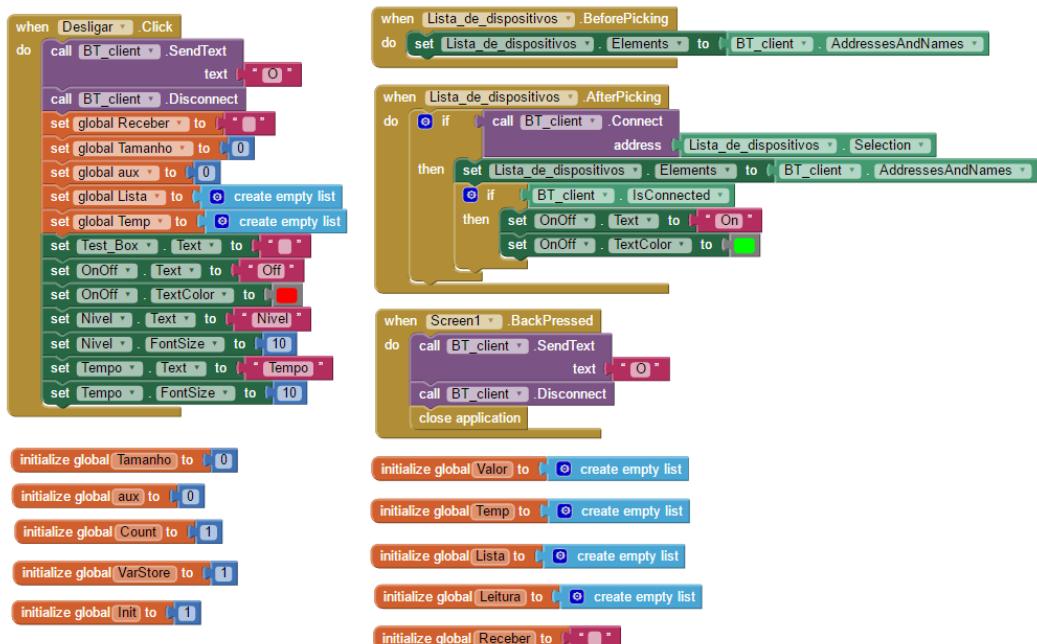


Figura B.3: Inicialização de variáveis e ações dos botões “Desligar” e “Dispositivos”

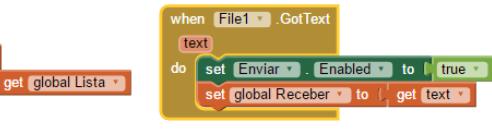
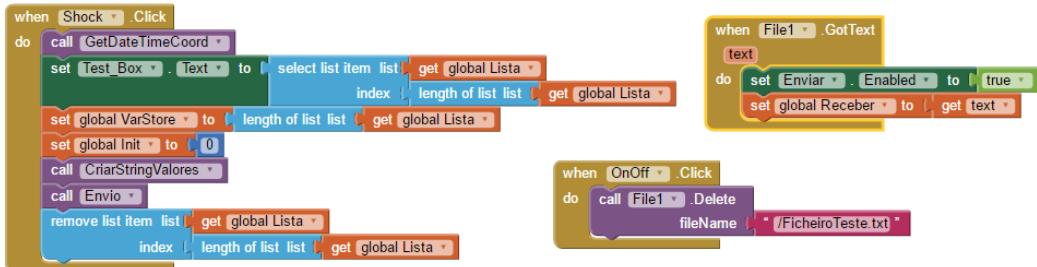


Figura B.4: Ação do botão “Shock” e remoção do ficheiro do telemóvel

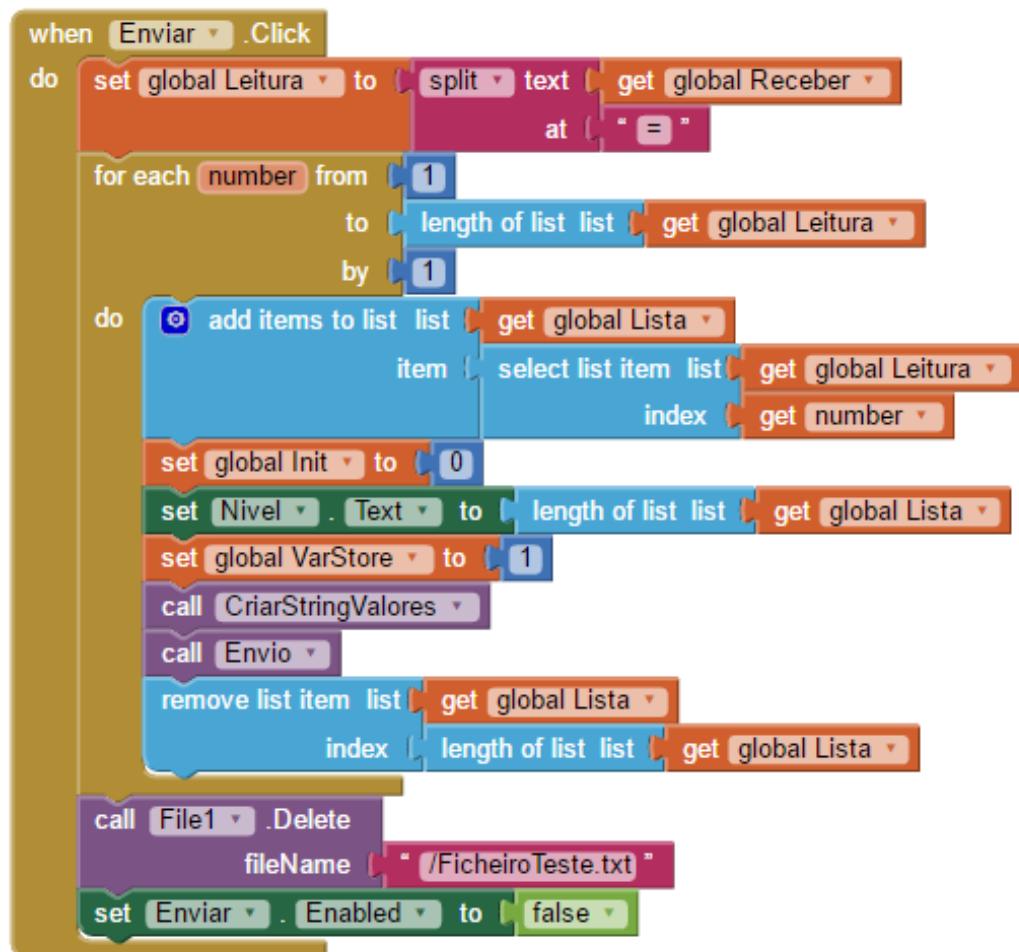


Figura B.5: Ação do botão “Enviar”

## APÊNDICE B. CÓDIGO DA APLICAÇÃO

---

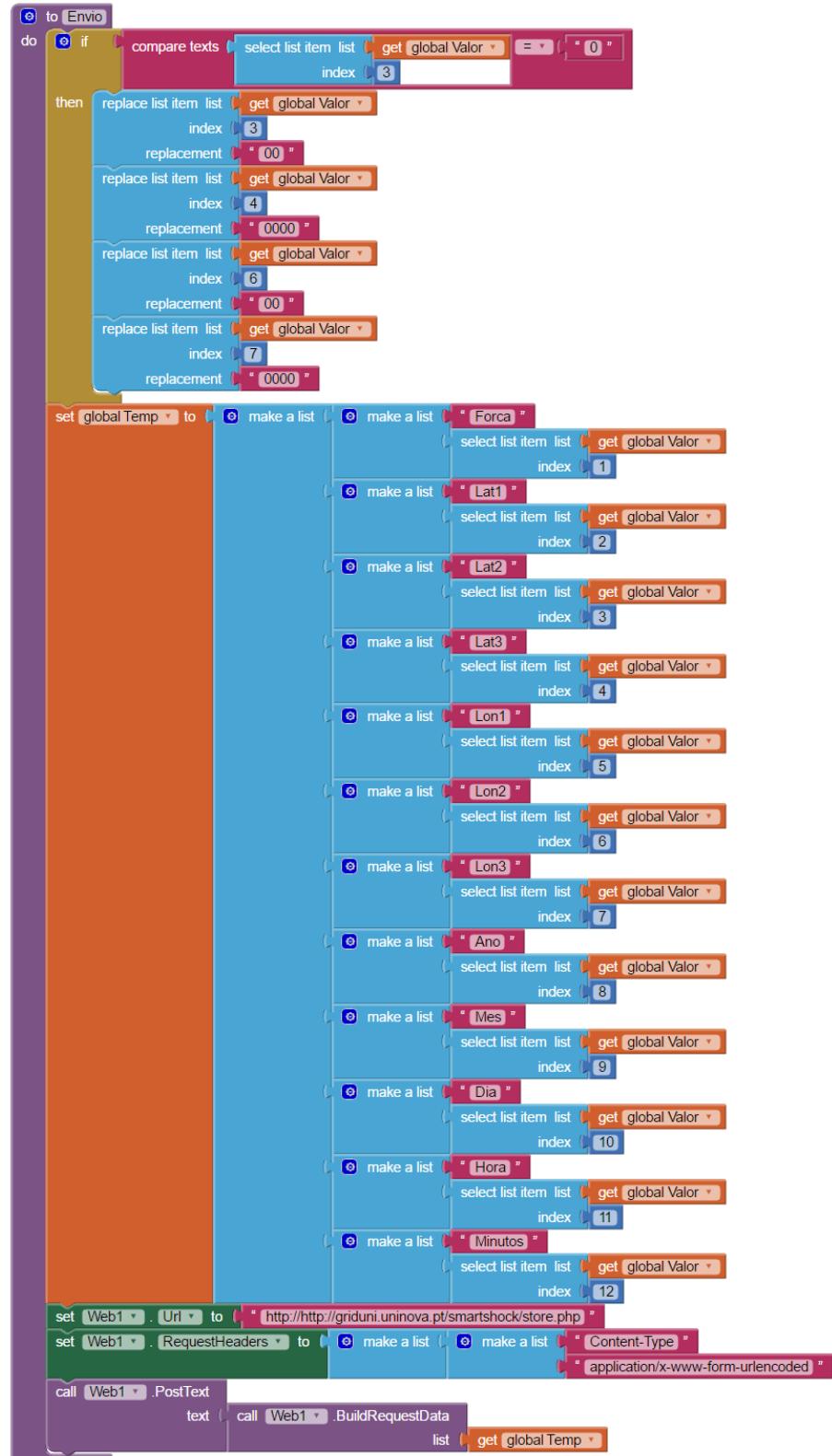


Figura B.6: Transformação de variáveis para envio para a base de dados

A P È N D I C E



## CÓDIGO DAS PÁGINAS WEB

```

<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no">
    <meta charset="utf-8">
    <link rel="icon" href="SSS.png" type="image/gif" sizes="16x16">
    <title>Mapa Total</title>
    <style>
        #map {height: 90%}
        html, body {height: 90%; margin: 1%; padding: 0;}
    </style>
</head>
<body>
    
    <script>
        var locations = [];
        var labels = [];
    </script>
    <?php
        $connect = mysqli_connect('localhost', 'root', '') or die('Não foi possível conectar: ' . mysql_error());
        mysqli_select_db($connect, 'tese') or die('Não foi possível selecionar o banco de dados');
        $query = 'SELECT FORCA, LATITUDE, LONGITUDE FROM COORDENADAS';
        $result = mysqli_query ($connect, $query);
        $number = mysqli_num_rows($result);
        $i=0;
        if(mysqli_num_rows($result)>0)
        while($rows = mysqli_fetch_array($result)){
            echo '<script>
                locations[".$i."]={lat:'.$rows["LATITUDE"].', lng: '.$rows["LONGITUDE"].'};
                labels[".$i."="'.$rows["FORCA"].'";
            </script>';
            $i = $i + 1;
        }
    ?>
    <div id="map"></div>
    <script>
        function initMap() {
            var map = new google.maps.Map(document.getElementById('map'), {
                zoom: 10,
                center: {lat: 38.6598, lng: -9.2037}
            });
            var markers = locations.map(function(location, i) {
                return new google.maps.Marker({
                    position: location,
                    label: labels[i]
                });
            });
            var markerCluster = new MarkerClusterer(map, markers,
                {imagePath: 'https://developers.google.com/maps/documentation/javascript/examples/markerclusterer/m'});
        }
    </script>
    <script src="https://developers.google.com/maps/documentation/javascript/examples/markerclusterer/markerclusterer.js">
    </script>
    <script async defer
        src="https://maps.googleapis.com/maps/api/js?key=AIzaSyAWSJAYNSvMH3f_65A-xMS2gbGipI90Ehg&callback=initMap">
    </script>
</body>
</html>

```

Figura C.1: Código da página home.php

```

<!DOCTYPE html>
<html>
    <head>
        <title>List Locations</title>
        <link rel="icon" href="SSS.png" type="image/gif" sizes="16x16">
    </head>
    <body>
        <?php
            $tablesize = 10;
            $number = ($_GET['number']-1)*$tablesize;

            $connect = mysqli_connect('localhost', 'root', '') or die('Não foi possível conectar: ' . mysql_error());
            mysqli_select_db($connect, 'tese') or die('Não foi possível seleccionar o banco de dados');

            $query = 'SELECT FORCA, LATITUDE, LONGITUDE, DATA FROM COORDENADAS LIMIT '.$tablesize.' OFFSET '.$number;

            $result = mysqli_query ($connect, $query) or die('A consulta falhou: ' . mysqli_error($connect));
            $number = mysqli_num_rows($result);

            echo '<table style="width:100%" border="2" cellpadding="5" align="center">
                . <tr bgcolor="#4499EE">
                . <th>Força</th>
                . <th>Latitude</th>
                . <th>Longitude</th>
                . <th>Data</th>
                . <th>Ver mapa</th>
                . </tr>';

            if(mysqli_num_rows($result)>0)
                while($rows = mysqli_fetch_array($result)){
                    echo '<tr align="center">
                        . <td>' . $rows['FORCA'] . '</td>
                        . <td>' . $rows['LATITUDE'] . '</td>
                        . <td>' . $rows['LONGITUDE'] . '</td>
                        . <td>' . $rows['DATA'] . '</td>
                        . <td><a href="showlocation.php?latitude=' . $rows['LATITUDE'] . '&longitude=' . $rows['LONGITUDE'] . '&forca=' . $rows['FORCA'] . '"> Mapa </a></td>
                    </tr>';
                }
            echo '</table>';

            $query = 'SELECT COUNT(ID) FROM COORDENADAS';
            $result = mysqli_query ($connect, $query) or die('A consulta falhou: ' . mysqli_error($connect));
            $counter = intval(mysqli_fetch_row($result)[0]);
            $Pages = ceil($counter / $tablesize);

            echo "<p>";
            echo "Pages: ";
            for($i=1; $i<=$Pages; $i++){
                if($_GET['number'] == $i)
                    echo $i . ' ';
                else
                    echo '<a href="listlocations.php?number=' . $i . '">' . $i . '</a> ';
            }
        >>
    </body>
</html>

```

Figura C.2: Código da página listlocationsphp.php

```

<!DOCTYPE html>
<html>
    <head>
        <title>Store Page</title>
        <link rel="icon" href="SSS.png" type="image/gif" sizes="16x16">
    </head>
    <body>
        <?php
            if(isset($_POST)){
                $Forca = $_POST['Forca'];
                $Lat1 = $_POST['Lat1'];
                $Lat2 = $_POST['Lat2'];
                $Lat3 = $_POST['Lat3'];
                $Lon1 = $_POST['Lon1'];
                $Lon2 = $_POST['Lon2'];
                $Lon3 = $_POST['Lon3'];
                $Ano = $_POST['Ano'];
                $Mes = $_POST['Mes'];
                $Dia = $_POST['Dia'];
                $Hora = $_POST['Hora'];
                $Minutos = $_POST['Minutos'];

                $Lat = $Lat2 + $Lat3/10000;
                $Lon = $Lon2 + $Lon3/10000;

                if ($Lat1 == '-')
                    $Lat = $Lat * -1;

                if ($Lon1 == '-')
                    $Lon = $Lon * -1;

                $connect = mysqli_connect('localhost', 'root', '') or die('Não foi possível conectar: ' . mysql_error());
                mysqli_select_db($connect, 'tese') or die('Não foi possível seleccionar o banco de dados');

                $date = date("$Ano-$Mes-$Dia $Hora:$Minutos");
                $query = 'INSERT INTO coordenadas (forca, latitude, longitude, data) VALUES (' . $Forca . ',' . $Lat . ',' . $Lon . ',' . $date . ')';
                $result = mysqli_query ($connect, $query) or die('A consulta falhou 2! ' . mysqli_error($connect));
                if($result === TRUE){
                    echo "Coordenada registada com sucesso!";
                } else {
                    echo "Ocorreu um erro no registo!";
                }
            }
        >>
    </body>
</html>

```

Figura C.3: Código da página storephp.php

## APÊNDICE C. CÓDIGO DAS PÁGINAS WEB

---

```
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no">
    <meta charset="utf-8">
    <link rel="icon" href="SSS.png" type="image/gif" sizes="16x16">
    <title>Mapa Total</title>
    <style>
        #map {height: 90%;}
        html, body {height: 90%; margin: 1%; padding: 0;}
    </style>
</head>
<body>
    <?php
        $latitude = $_GET['latitude'];
        $longitude = $_GET['longitude'];
        $forca = $_GET['forca'];
    ?>
    
    <script>
        var locations = [];
        var labels = [];
    </script>
    <?php
        $i=0;
        echo '<script>
            locations[".$i."]={lat: '.$latitude.' , lng: '.$longitude.'};
            labels[".$i."]="'.$forca.'";
        </script>';
    ?>
    <div id="map"></div>
    <script>
        var lat1 = <?php echo $latitude; ?>;
        var lon1 = <?php echo $longitude; ?>;
        function initMap() {
            var map = new google.maps.Map(document.getElementById('map'), {
                zoom: 15,
                center: {lat: lat1, lng: lon1}
            });
            var markers = locations.map(function(location, i) {
                return new google.maps.Marker({
                    position: location,
                    label: labels[i]
                });
            });
            var markerCluster = new MarkerClusterer(map, markers,
                {imagePath: 'https://developers.google.com/maps/documentation/javascript/examples/markerclusterer/m'});
        }
    </script>
    <script src="https://developers.google.com/maps/documentation/javascript/examples/markerclusterer/markerclusterer.js">
    </script>
    <script async defer
        src="https://maps.googleapis.com/maps/api/js?key=AIzaSyAisJAYNSvIH3f_65A-xMS2gbGipI90Ehg&callback=initMap">
    </script>
</body>
</html>
```

Figura C.4: Código da página showlocationsphp.php