INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área Departamental de Engenharia Mecânica

ISEL

Utilização de Plataformas de Fonte Aberta no Controlo de Condição

João Pedro Ferreira de Sousa

Licenciado em Engenharia Mecânica

Trabalho Final de Mestrado para obtenção do grau de Mestre em Engenharia Mecânica

Orientador: Doutor Rui Pedro Chedas de Sampaio

Júri:

Presidente: Doutor Joaquim Infante Barbosa

Vogal: Doutor José Augusto da Silva Sobral

Vogal: Doutor Rui Pedro Chedas de Sampaio

INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área Departamental de Engenharia Mecânica

ISEL

Utilização de Plataformas de Fonte Aberta no Controlo de Condição

João Pedro Ferreira de Sousa

Licenciado em Engenharia Mecânica

Trabalho Final de Mestrado para obtenção do grau de Mestre em Engenharia Mecânica

Orientador: Doutor Rui Pedro Chedas de Sampaio

Júri:

Presidente: Doutor Joaquim Infante Barbosa

Vogal: Doutor José Augusto da Silva Sobral

Vogal: Doutor Rui Pedro Chedas de Sampaio

Resumo

Neste projeto pretendeu-se utilizar plataformas de fonte aberta, de baixo custo, para desenvolver um equipamento de aquisição e processamento de vibrações para o controlo de condição fazendo uso de sensores de vibração MEMS (*micro electro mechanical systems*), também estes de baixo de baixo custo, por meio de uma aplicação para detecção de avarias em máquinas rotativas e para determinação do risco em estruturas sujeitas a excitações impulsivas.

Através da plataforma de fonte aberta Android projetou-se uma aplicação para um dispositivo móvel sob a forma de um analisador de vibrações. Este é capaz de caracterizar sinais no domínio do tempo e da frequência, tem pré-definidos os limites das normas consideradas, que o utilizador poderá personalizar todas as opções. Tem incluído, para isso, diferentes métodos de aquisição de dados, transformada rápida de Fourier, filtros passa-alto e passa-baixo, alertas sonoros e visuais tanto locais como não locais.

Assim provou-se que é possível, mesmo sem conhecimentos profundos de programação, fazer uso destas plataformas e concretizar ferramentas para um uso efetivo e prático no controlo de condição de máquinas e estruturas. A utilização deste género de plataformas, como *tablets*, telemóveis e portáteis, poderá facilmente alargar-se a outro tipo de tecnologia de controlo de condição, como sejam a termografia, os parâmetros de processo e análises aos fluidos.

Palavras-chave

Manutenção, Controlo de Condição, Vibrações, Plataformas de Fonte Aberta, Android

Abstract

This project highlights the use of low-cost open-source platforms in condition monitoring with the intention of using vibration sensors, such as MEMS (*micro electro mechanical systems*), and therefore the development of data displays and ways of warning for use in condition monitoring of both machines and risk assessment of structures.

A vibration analyzer was projected using the Android open source software. It possess a variety of characteristics such as display data in time and frequency domain, ability of setting warnings following standards or personalized ones. Therefore, a group of methods like the fast Fourier transform, low and high-pass filters, local or non-local audible and visible warnings was developed.

Even though, there was no profound knowledge in programing, a vibrations analyzer application for mobile devices was developed. The use of this kind of platforms, like tablets, mobile phones and laptops, could easily spread to other type of technologies in condition monitoring such as, thermography, process control and fluid analysis.

Key words

Maintenance, Condition Monitoring, Vibrations, Open Source Platform, Android

Agradecimentos

Este projeto só foi possível realizar com muita dedicação pessoal e com importantes apoios e incentivos sem os quais seria-me difícil torná-lo realidade.

Quero agradecer à minha família, especialmente aos meus pais, sem eles não teria chegado aqui, como também aos meus tios e ao Eládio por todo o apoio. À Mariana, pelo apoio e motivação nesta etapa da minha vida, que foi muito importante para o meu sucesso!

Ao professor Rui Pedro Chedas Sampaio e Paulo Pereira, pelo apoio e pela disponibilidade para me ajudar, ao longo do desenvolvimento deste trabalho.

A todos os meus amigos, em especial ao Tiago e ao Filipe, que me acompanharam nesta jornada e sempre me deram força para continuar.

A todos, obrigado!

ÍNDICE

Página

ÍNDICE DE FIGURAS

Página

ÍNDICE DE TABELAS

Página

ÍNDICE DE GRÁFICOS

Página

Lista de siglas e abreviaturas

DIY	Faça Você Mesmo (do inglês *do it yourself*)
FFT	Transformada Rápida de Fourier (do inglês *fast fourier transform)*
GPS	Sistema de Posicionamento Global (do inglês *global positioning system*)
IOT	Internet das Coisas (do inglês *internet of things*)
LCD	Ecrã de Cristais Líquidos (do inglês *liquid crystal display*)
MEMS	Sistemas Electromecânicos de Pequena Dimensão (do inglês *micro electro mechanical systems*)
OTG	Cabo *on the go*
RMS	Média Quadrática (do inglês *root mean square*)
S.O.	Sistema Operativo
USB	Ligação Universal
XML	Extensible Markup Language

1. INTRODUÇÃO

O controlo de condição é uma importante estratégia da manutenção preventiva e é amplamente utilizada na indústria. Esta abordagem traz, sem dúvidas, benefícios à indústria como a redução de custos diretos de manutenção mas permite também, à função produção, ser informada quando terá de parar e realizar algumas tarefas de manutenção. É reconhecida como a estratégia mais eficiente numa variedade de indústrias sendo um pilar da função manutenção [1].

A manutenção, segundo a norma portuguesa NP EN 13306 [2], *é a combinação de todas as ações técnicas, administrativas e de gestão durante o ciclo de vida de um bem, destinadas a mantê-lo ou repô-lo num estado em que possa cumprir a função requerida.* Esta também define a manutenção preventiva como *a manutenção realizada com uma frequência pré-definida, com o objetivo de reduzir a probabilidade de avaria e deterioração do equipamento.* Os objetivos da manutenção preventiva centram-se em aumentar a duração de vida dos materiais, diminuir a probabilidade das avarias e também os tempos de imobilização, permitir tomar decisões sobre a melhor ação corretiva a tomar e, sobretudo, em diminuir os custos de manutenção.

O controlo de condição encaixa-se nesta filosofia sendo a ação de *determinação da condição das máquinas e estruturas, enquanto em funcionamento, para prever e programar a reparação mais eficiente antes da ocorrência da falha catastrófica* levando a ações de reparação ou substituição de forma condicionada [2].

Contudo são necessários equipamentos que permitam a aquisição de dados, condição essencial à manutenção preventiva para a determinação da condição das máquinas e estruturas. São medidos vários parâmetros processuais, temperaturas (termografia), propriedades físico-químicas (análise aos fluídos), desgastes (análises espectrométricas e ferrográficas), entre outras, de forma a determinar o estado do sistema. Esta aquisição de dados é normalmente feita com sistemas de aquisição e processamento de sinal dispendiosos. Destas técnicas, a análise vibracional é a mais utilizada na indústria pois é a mais eficaz e eficiente na detecção de dano e avaliação do estado [1].

1.1. ENQUADRAMENTO DO TEMA

Mesmo em boas condições uma máquina produz vibrações que poderão estar relacionadas com o seu funcionamento como a rotação de veios e o acoplamento de engrenagens. A frequência

com que estas ocorrem dão informações sobre o seu estado. Nas máquinas rotativas as variações no sinal vibracional poderão indicar mudanças na condição do equipamento sempre que a velocidade e carga deste não tenham variado.

No controlo de condição as vibrações são primordiais sendo usadas na determinação da condição de máquinas e estruturas [3]. Para isso recorre-se à aquisição de sinal e ao seu processamento que será realizado nos analisadores de vibrações.

A vibração é definida como o movimento de um dado ponto, ou sistema, em torno de uma posição que pode ser medida, instante a instante, relativamente a uma posição de referência. Poderá ser lida em aceleração, velocidade ou deslocamento. Pode ser classificada em livre, quando um sistema vibra após o fim da excitação, ou forçada quando a excitação está presente. Também poderá ser não amortecida, como num caso teórico em que o amortecimento seja nulo, representada no Gráfico 1.1, ou amortecida onde a amplitude tenderá a decrescer até zero, como no Gráfico 1.2.

Medir a vibração em unidades de deslocamento é onde se obtém a melhor aproximação da ideia de oscilação, ou seja, maior oscilação maior vibração. No entanto maior frequência também implica maior vibração o que nos leva a precisar de medições em velocidade ou aceleração na maior parte das aplicações.

$$x(t) = X.\cos(2\pi.f.t + \alpha) \tag{1.1}$$

$$\dot{x}(t) = 2\pi.f.X.\cos(2\pi.f.t + \alpha + \pi/2) \tag{1.2}$$

$$\ddot{x}(t) = (2\pi.f)^2.X.\cos(2\pi.f.t + \alpha + \pi) \tag{1.3}$$

Todas as formas de vibração são compostas pela soma de vibrações harmónicas e têm como propriedades a amplitude (X), a frequência (f), que representa o número de ciclos ou repetições do movimento por unidade de tempo e também a fase, α, que nos diz a posição inicial do ponto aquando do início da medição tal como está representado na equação (1.1.

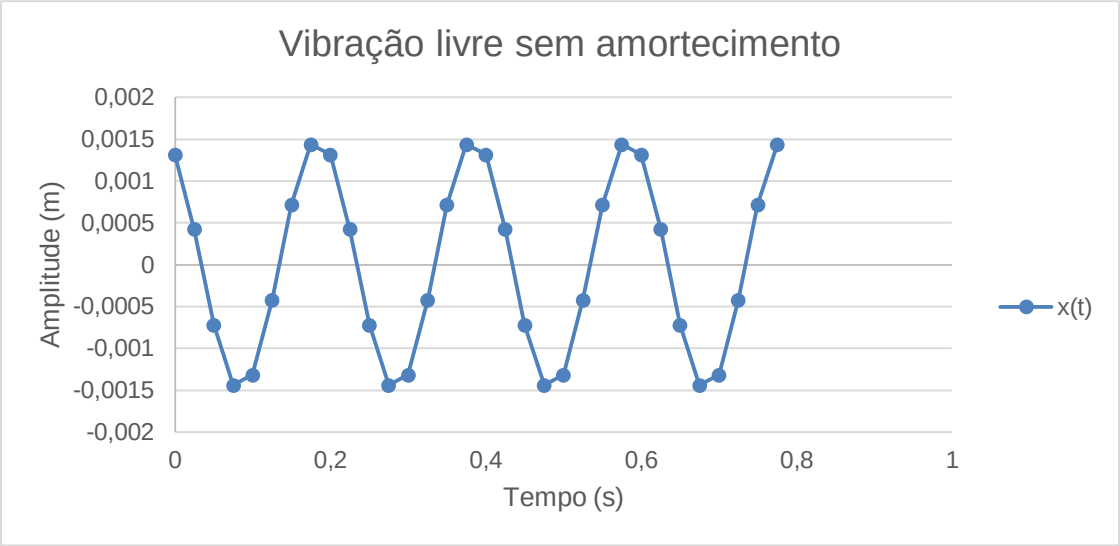

Gráfico 1.1 - Representação gráfica de um sinal harmónico.

Esta trata-se da representação mais simples de uma onda. No mundo real os sinais são compostos por várias harmónicas e, dada a existência de amortecimento (c), os sinais são mais complexos e representados pela equação (1.4.

$$x(t) = e^{-\zeta * \omega_n * t} * X * sen(\omega_d * t + \alpha) \quad (1.4)$$

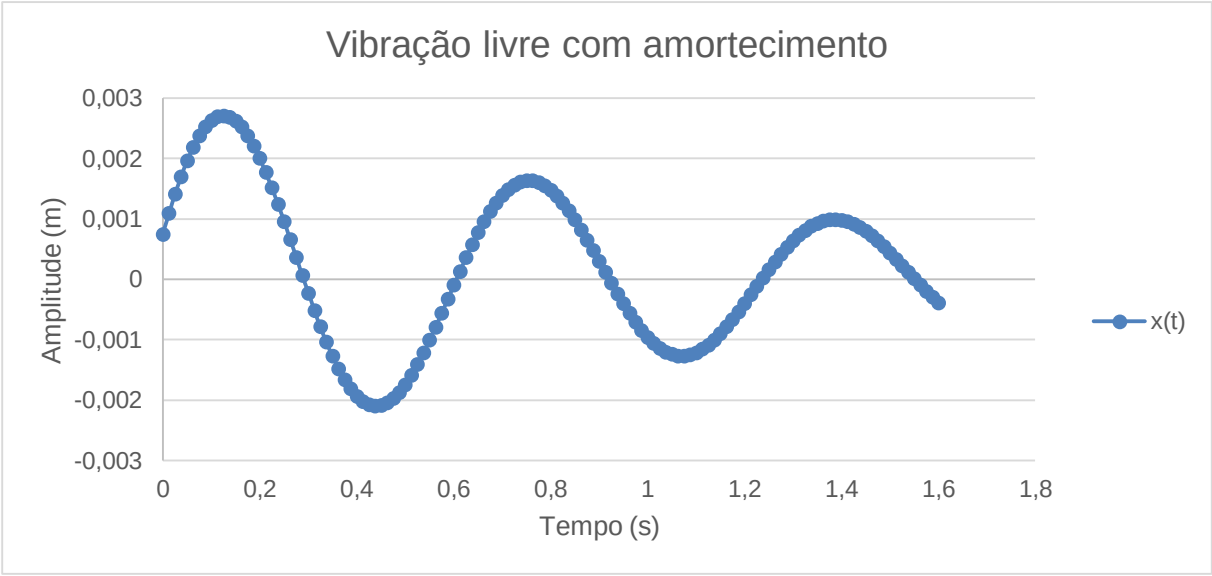

Gráfico 1.2 - Representação gráfica de vibração livre com amortecimento.

A determinação da condição é fundamentada na interpretação dos sinais adquiridos e suas características como a direção da vibração, amplitude e outros. É um processo complexo que se baseia na relação entre as frequências presentes no sinal adquirido.

Randall, R. B. *in Vibration Based Condition Monitoring* apresenta vários exemplos destes fenómenos. O desalinhamento de veios, fenómeno comum, poderá ser paralelo quando os veios estão desfasados lateralmente, mas mantêm o paralelismo, ou poderá ser angular quando um dos eixos de um dos veios apresenta algum ângulo de desfasamento em relação ao outro. Este facto provocará cargas adicionais nos apoios que alterarão a vibração, como representa a Figura 1.1.

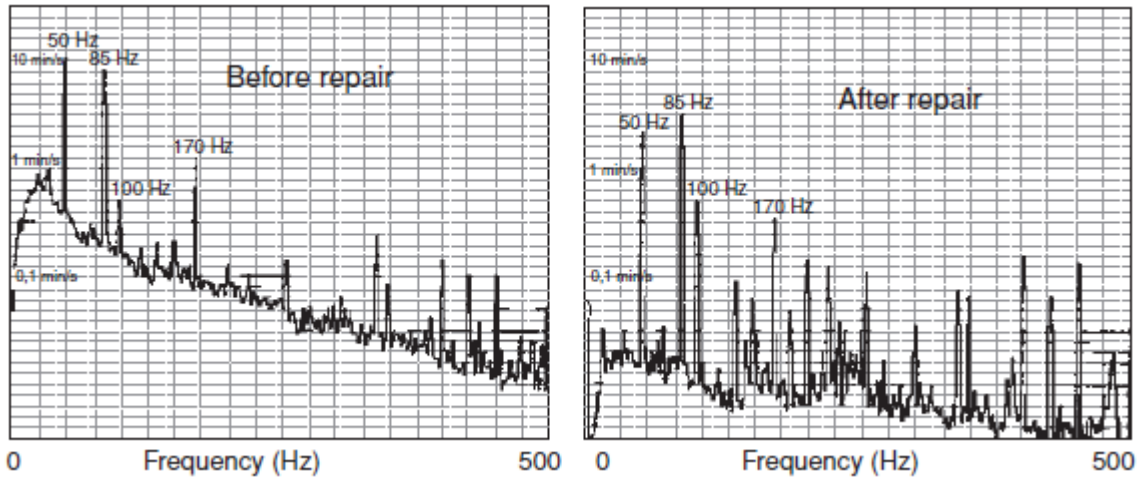

Figura 1.1 – Exemplo do efeito do desalinhamento no espectro de frequênica; [1]

O controlo de condição é também pratica corrente na manutenção de rolamentos. Estes dispositivos presentes em muitos órgãos de máquinas apresentam componentes no espectro de frequência relativos à sua operação. Na Figura 1.2 representa-se o efeito da folga da montagem das pistas do rolamento.

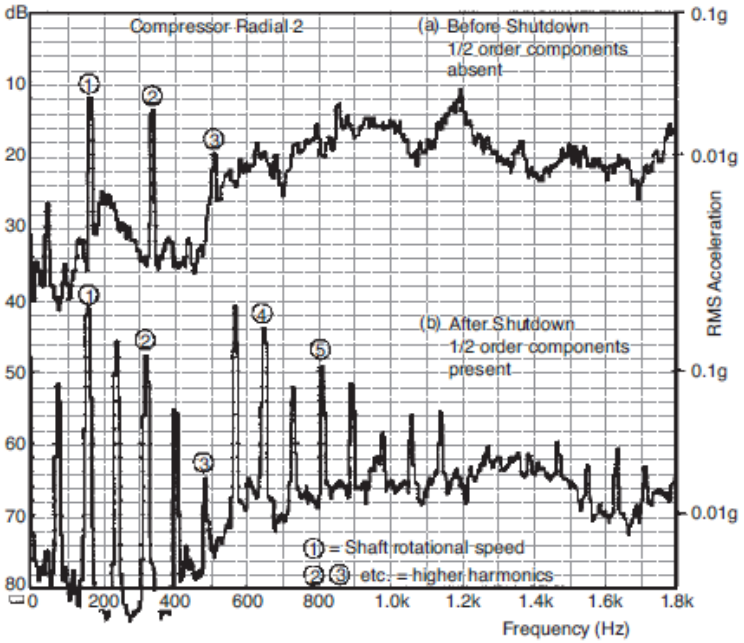

Figura 1.2 – Representação do espectro de frequência do aparecimento de um defeito na pista de um rolamento; [1]

A medição de vibrações também é usada no controlo de condição de estruturas. *In Vibration Based Structural Health Monitoring of an Arch Bridge*, Magalhães, F. *et al.* (2012) é apresentado um sistema de monitorização de uma ponte, pela medição das suas frequências naturais, e é demonstrado que é possível a detecção de dano associado à variação destas.

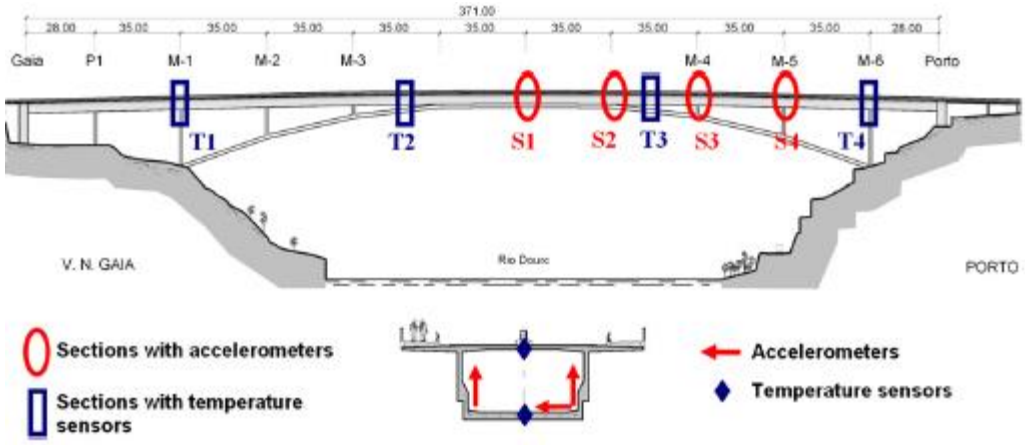

Figura 1.3 – Representação da estrutura e colação dos sensores; [4]

A variação da frequência natural implica a existência de variação de características intrínsecas à estrutura como a rigidez e a massa. A existência de dano na estrutura será então avaliada pela

medição das vibrações recorrendo a acelerómetros como exemplifica a Figura 1.3. Este género de análise também pode ser aplicada a outro género de estruturas [4].

1.2. OBJETIVO DO PROJETO

Neste projeto pretende-se utilizar plataformas de fonte aberta, de baixo custo, para desenvolver um equipamento de aquisição e processamento de vibrações para o controlo de condição fazendo uso de sensores de vibração MEMS, também estes de baixo de baixo custo, por meio de uma aplicação para detecção de avarias em máquinas rotativas e para determinação do risco em estruturas sujeitas a excitações impulsivas.

O termo controlador de fonte aberta implica algo acessível, passível de modificação e no fundo projetado para todos e para tudo. Juntamente com um controlador tem-se na generalidade dos casos acesso ao código de desenvolvimento e plataformas para podermos aplicar as nossas próprias mudanças. Os controladores fechados, ou proprietários, não oferecem tanta liberdade e são usualmente mais dispendiosos de adquirir e o utilizador terá de aceitar os termos de uso e também que não fará alterações ao programa [5].

Declarada a importância da análise de vibrações no controlo de condição e que a forma que é feita atualmente a aquisição de sinais de vibração é por meio de equipamentos dispendiosos e pouco adaptáveis a cada realidade industrial surge o objetivo de criar, numa plataforma de fonte aberta, que recaiu sobre um dispositivo Android, uma ferramenta que possa ser usada como um analisador de vibrações versátil e de baixo custo.

Assim, neste projeto, demonstrar-se-á que, por muito menos, consegue-se criar ferramentas para o auxílio ao controlo de condição mais adaptadas à nossa realidade sem serem precisos muitos recursos. A oferta de controladores de fonte aberta é cada vez maior e com maior capacidade de processamento abrindo portas a novos projetos.

1.3. METODOLOGIA

A criação de um analisador de vibrações numa plataforma de fonte aberta pressupõe a aquisição de conhecimentos sobre a sua natureza e especificidades e uma compreensão das necessidades de um futuro utilizador.

Começou-se por estudar conceitos e tecnologias importantes do controlo de condição por análise de vibrações como a aquisição de sinal e seu processamento de forma a se conseguir escolher qual a plataforma que mais se adequaria.

Esta escolha, como supramencionado, recaiu sobre os dispositivos Android. Tal facto levou à necessidade de estudar a sua linguagem de programação, Java, que não é objeto de estudo no presente plano curricular do Mestrado em Engenharia Mecânica, levando a uma aprendizagem autónoma. O desenvolvimento do projeto seguiu as seguintes etapas:

i. Revisão dos conhecimentos de controlo de condição por análise de vibrações
    a. Aquisição de sinal
    b. Processamento de sinal
    c. Detecção de avarias em máquinas
    d. Normas ISO 10816 e NP2074
ii. Estado da arte dos controladores de fonte aberta
    a. Plataformas existentes
    b. Possibilidade de projetos
iii. Introdução à linguagem de programação Java
    a. Compreensão da estrutura de programação
    b. Aprendizagem de métodos e metodologias de programação
iv. Introdução ao sistema operativo (S.O.) Android
    a. Primeiro contato com a plataforma de desenvolvimento
    b. Criação de aplicações básicas
v. Realização da aplicação
    a. Levantamento do problema (objetivos)
    b. Análise do problema
        i. Definição das saídas
        ii. Definição das entradas
        iii. Desenho da interface da aplicação
    c. Programação
    d. Concretização de um primeiro protótipo
    e. Testes

1.4. ESTRUTURA

O capítulo de abertura enquadrará o estado da arte do controlo de condição por medição de vibrações, aquisição e processamento de sinal bem como a ilustração dos vários equipamentos de fonte aberta que foram considerados para a realização deste projeto. Também se consultou vários equipamentos de aquisição de sinal, de fonte fechada, para perceber quais as características necessárias para uma boa experiência de utilização.

Em seguida introduz-se, de forma mais criteriosa, a teoria por detrás do projeto. A compreensão do controlo de condição por medição de vibrações é fundamental para os passos dados nos capítulos seguintes. O foco estará na aquisição e no processamento de sinal proveniente do sensor.

O S.O. Android, capítulo 4, será a plataforma que receberá este projeto. Neste sistema operativo, que gere os recursos do equipamento e a interface, serão desenvolvidos métodos e classes que visarão o cumprimento dos objetivos de um coletor de dados. A linguagem de programação usada, características dos métodos, capacidades do *hardware* e *software* serão abordados no mesmo capítulo.

O capítulo cinco tratará o projeto de uma forma mais concreta. Será dada uma explicação dos métodos usados na programação, relacionando sempre, com a teoria do controlo de condição por vibrações e exposição do meio como o utilizador irá interagir com estes métodos através da interface gráfica.

2. ESTADO DA ARTE

O uso de dispositivos móveis para efetuar outras atividades que não a comunicação é um tema muito atual. Estes dispositivos possuem caraterísticas de processamento e de memória aceitáveis para realizar cálculo e aquisição de dados através dos sensores internos que possui. Atualmente os dispositivos adquirem informações de vários sensores, que podem ser na ordem das dezenas, para dar uma melhor experiência de utilização ao portador, desde acelerómetro (do tipo MEMS), o giroscópio, o sensor de proximidade sob a forma de infravermelhos, o sensor de luz, o barómetro, o termómetro, entre outros.

Kotsakos, Dimitros *et al.* (2013) e De Dominics *et al.* (2014) apresentam trabalhos sobre a utilização dos sensores internos dos dispositivos móveis inteligentes também chamados de *smartphones*. No primeiro é estudada a utilização de vários *smartphones,* em rede, em que os seus acelerómetros internos são usados para o controlo de condição da estrutura em que estão assentes enquanto estes não estão a ser usados [6]. É criada uma aplicação que, quando o utilizador não está a usar o seu dispositivo, esta pode recolher dados do acelerómetro e efetuar o seu processamento que, após a recolha da informação dos vários utilizadores, servirá para fazer uma correta localização de dano ao longo da estrutura. Outra iniciativa similar a esta é a Quake-Catcher Network.

Esta iniciativa pretende desenvolver a maior rede mundial de sismógrafos, usando acelerómetros de baixo custo, ligados por rede onde indivíduos ou instituições podem participar [7]. Deste modo o projeto quer aumentar o número de estações sísmicas e melhorar a caracterização de sismos de moderada ou grande escala. Funciona com o uso de MEMS que estejam já presentes nos dispositivos ou que sejam adquiridos para uso como periférico que, quando estes não estão em uso, adquirem dados e aplicam o processamento para então enviar para um servidor central para análise posterior. Este projeto conta já com algumas centenas de participantes e consegue alertar para a ocorrência de um sismo entre 3 a 6 segundos depois da sua ocorrência [8].

No segundo trabalho é estudada a viabilidade destes mesmos sensores internos. Foi criada uma banca de testes recorrendo a um *shaker*, equipamento que aplicará oscilações para simular ondas vibracionais, um dispositivo móvel e um acelerómetro de grande capacidade com o intuito de perceber a real capacidade do acelerómetro interno do *smartphone*. Os resultados apresentados indicam que o acelerómetro interno só é fiável para baixas frequências [9].

Person, Tony (2012) usou a mesma abordagem aqui apresentada em que o acelerómetro é externo. Com o auxílio de um fabricante acelerómetros o autor criou uma aplicação que é capaz de receber informação, proveniente da ligação entre os dois, e apresentá-la graficamente [10].

Figura 2.1 – Dispositivo Android com acelerómetro externo ligado e representação do sinal adquirido; [10]

Este projeto é aquele que mais se enquadra no que aqui é apresentado, contudo, não faz uso dos acelerómetros do tipo MEMS nem apresenta características para além da aquisição de sinal, representado na Figura 2.1, focando-se mais na criação da comunicação entre os dois dispositivos.

2.1. AQUISIÇÃO E PROCESSAMENTO DE SINAL NO CONTROLO DE CONDIÇÃO

O estudo do sinal é fortemente baseado na análise em frequência, ou seja, no domínio da frequência. O algoritmo que é usado para a transformação do sinal no tempo para sinal em frequência é a transformada rápida de Fourier (FFT do inglês *Fast Fourier Transform*).

Na realidade, o sinal adquirido é discreto o que levanta dificuldades também. A primeira restrição obriga o número de pontos a ser uma potência de 2 e que a análise seja feita a todo o bloco adquirido. Apesar disso metade do sinal anterior fornece a informação que se pretende: amplitude e fase. Haverá ainda necessidade de usar filtros e janelas adequadas a uma determinada situação.

O conhecimento de tais factos influenciará a escolha do equipamento para efetuar a aquisição e o processamento do sinal. No caso concreto das vibrações será necessário o uso de um sensor de aceleração que transformará o estímulo físico num sinal eléctrico, mais concretamente por meio de acelerómetros.

Os acelerómetros com mais uso recorrem a um sensor piezoeléctrico, como o da Figura 2.2 [1]. Este material, geralmente quartzo, produz um potencial eléctrico, quando sujeito a pressão mecânica, criado pelo deslocamento de iões [11].

Figura 2.2 - Acelerómetro piezoeléctrico SKF; [12]

Mais recentemente tem-se usado pequenas placas de aquisição que já incluem o sensor e transformação de sinal denominadas de MEMS [13]. Estas pequenas placas podem ser piezoresistivas ou capacitivas. No primeiro caso o impulso eléctrico é gerado pelo movimento de uma massa de teste, por vibrações por exemplo, que alterará a resistência do material piezoresistivo. O capacitivo, descrito na Figura 2.3, mede mudanças da capacitância entre a massa teste e um eléctrodo que estão separados por uma pequena margem [14].

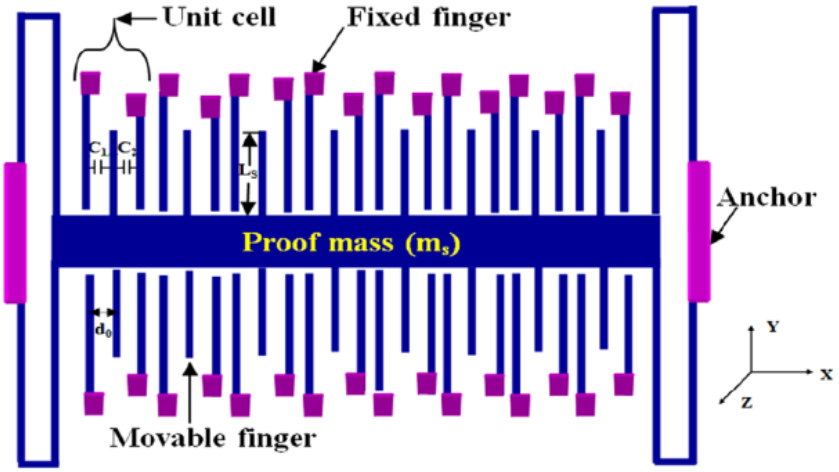

Figura 2.3 – Estrutura de acelerómetro MEMS capacitivo; [15]

Apesar da sua pequena dimensão e baixo custo são aplicados em diversas situações como, controlo de condição de edifícios [16], de pás eólicas [17], de compósitos laminados [18], e é muito utilizado na indústria automóvel. O seu uso ainda não é consensual dado a existência de algumas discrepâncias no seu desempenho quando comparado com outro género de acelerómetros [14]. Kavitha *et al.* [15] concluíram que o uso destes aparelhos para a detecção

de danos estruturais (SHM do inglês *structural health monitoring*), em edifícios de grandes dimensões, não é fiável dada a pouco sensibilidade que alguns MEMS apresentam.

O MEMS 1041 da Phidget foi o escolhido para este projeto dado o seu razoável preço e disponibilidade de apoio da marca em termos de *software* complementar. Esta disponibiliza na sua página conteúdo que permitirá a placa de aquisição passar a informação para o dispositivo final [19]. Esta companhia especializa-se na manufatura de sensores e controladores de baixo custo e fáceis de usar.

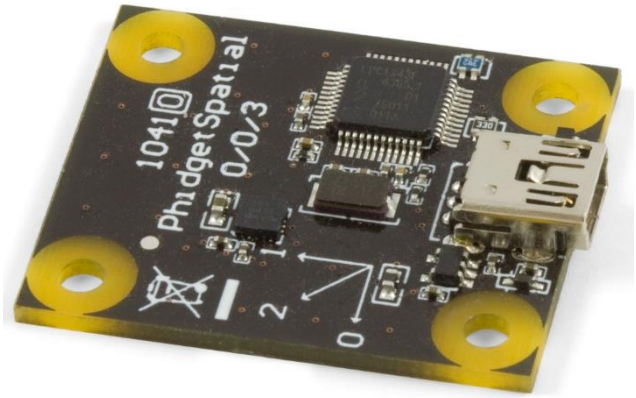

Figura 2.4 – Placa de aquisição 1041 PhidgetSpatial 0/0/3 da Phidget; [20]

A placa 1041, Figura 2.4, é capaz de adquirir valores em 3 eixos e reporta os valores medidos de aceleração gravítica, vulgo g´s. Estes serão afetados por ruído devido a oscilações de temperatura e variações mecânicas dentro do próprio sensor. Estes valores poderão variar para cada eixo de leitura dada a construção dissimilar.

O ruído pode ser *branco* ou por desfasamento. O *white noise* é quase sempre consistente e pode ser anulado pela média de várias medições. Dado isto, para uma aplicação que necessita de baixas frequências de amostragem, é preciso ter em conta este fenómeno. O desfasamento, ou *drift*, como o nome indica, dá-se quando os valores começam a variar cada vez mais do original. É menos importante em aplicações com movimento constante e baixa frequência de amostragem [21].

2.2. PLATAFORMAS DE FONTE ABERTA

No mercado existem muitas ofertas de controladores *open source*, que cada dia são mais diversificados, com uma disponibilidade de acessórios também ela muito grande.

Neste projeto abordam-se as plataformas Arduino, Rasperry Pi, Intel e o sistema operativo Android. A plataforma Arduino, como as contrapartes da fundação Rasperry e da Intel, é um controlador de prototipagem electrónica assente na filosofia de livre acesso e livre modificação. Tanto os componentes físicos e programação são de livre acesso e modificação [22].

O projeto Arduino foi criado em Itália, 2005, com o objetivo de fornecer às escolas formas mais práticas de poderem criar protótipos a um custo mais baixo. Novamente, o Rasperry Pi e o Intel Galileo também foram criados com o mesmo intuito sendo que a fundação Rasperry tem como missão a transmissão de conhecimentos de programação de adultos e crianças.

O Rasperry Pi é mais que um controlador, é na verdade um microcomputador, do tamanho de um cartão de crédito, que pode ser usado, diretamente, num monitor de televisão, permite o uso de teclado e rato e é muito capaz em termos de prestação [23].

A placa Galileo é projetada especialmente para a educação, mas também para fabricantes [24]. Esta placa está mais na linha do Arduino e vem complementar e ampliar as possibilidades deste. Num outro espectro a Intel também oferece o Edison que é mais capaz e enquadra-se mais com o Rasperry Pi [25].

O S.O. Android foi criado para permitir dar novas funcionalidades aos aparelhos do nosso dia-a-dia. As primeiras iterações foram pensadas para câmaras fotográficas digitais, mas, uma vez que o mercado não era escalável, decidiram atacar o mercado móvel de comunicações fazendo frente ao Symbiam e Windows Mobile, não obstante do facto de já existirem televisões e câmaras fotográficas, na mesma altura, que faziam usufruto deste [26].

Outro fator a realçar é o facto do ambiente Android poder ser facilmente modificado e que leva a que seja possível criar um ambiente de trabalho propício a cada situação ou ambiente industrial.

2.1.1. PROJETOS COM CONTROLADORES DE FONTE ABERTA

No meio industrial muitas vezes surge a necessidade de projetar uma solução de controlo ou monitorização de forma rápida e de baixo custo. As plataformas acima referidas são ideais para se realizar um protótipo do que se pretende e, porventura, ser a solução definitiva.

Na página da Arduino encontram-se acessórios como acelerómetros, atuadores, detectores de presença por ultrassons, termómetros e voltímetros abrindo portas a que seja possível "prototipar*"* uma grande variedade de controladores. Anexando a esta placa uma plataforma de computação como o Edison pode-se obter um sistema de controlo eficaz e confiável.

Através de uma rápida pesquisa encontram-se inúmeras utilizações possíveis com as supramencionadas plataformas. O *hub* preferencial deste género de protótipos, muitas vezes denominados de projetos *DIY*, do inglês "*do it yourself*", é a página Instructables.

Focando em projetos de utilidade no controlo de condição encontraram-se vários controladores de temperatura. A destacar o termóstato inteligente [27]. Este projeto inclui o uso da plataforma Arduino e o desenvolvimento para um dispositivo móvel de comunicações Android visto haver a possibilidade de controlo remoto. Outro exemplo, de Patrick S. [28], mais simples é apresentado como uma solução para controlo automático para um sistema de ventilação que ajusta a posição da potência do aparelho conforme a temperatura ambiente sentida num quarto. Também se podem criar detectores de fumo, metais e de presença.

Sem dúvida que estes exemplos não retratam todas as possibilidades mas servem essencialmente para demonstrar que, dado ao elevado número de pessoas que usam estas plataformas, encontram-se disponíveis imensos exemplos de aplicações que poderão ser aplicados em prol da função manutenção.

A iniciação a uma nova plataforma trará sempre algumas dificuldades de aprendizagem. No caso destas plataformas tudo é cedido gratuitamente para o auxílio à aprendizagem e, em alguns casos, após a aquisição das placas e dos respectivos sensores, pode-se descarregar o código pronto a usar e ajustar os parâmetros para as necessidades de um determinado projeto. Nas respectivas páginas de internet encontram-se fóruns, ideias para projetos, e bibliotecas educacionais. Na sequência deste projeto muitas dificuldades foram ultrapassadas em plataformas online, dedicadas a programadores, que são acessíveis a todos.

2.3. ANALISADORES DE VIBRAÇÕES

A ferramenta essencial para recolher dados para o controlo de condição são os analisadores de vibrações, e o mercado oferece muitas soluções. Estes equipamentos são normalmente robustos, à prova de água e têm uma interface simplista.

Os analisadores, como o da Figura 2.5, podem receber dados de forma dinâmica ou estática, recebem informação dos acelerómetros por ou sem fio, e dispõe de acessórios complementares. As leituras podem ser feitas em três eixos, e também efetuam análise espectral [29].

Figura 2.5 – Analisador de vibrações SKF; [29]

Este dispositivo recorre ao sistema operativo Windows Embedded e conta com um processador de 806 MHz, 128 MB de memória e 120 MB de armazenamento interno. Estas características não são impressionantes e ficam muito abaixo de um dispositivo móvel recente. Este modelo particular conta ainda com uma porta de comunicação, indicador visuais e leitor de cartão de memória. A bateria pode durar até 8 horas de contínua aquisição de dados.

Na Tabela 2.1 apresenta-se as características do acelerómetro CMSS 2200 que é compatível com o analisador acima descrito.

Tabela 2.1 – Caraterísticas do Acelerómetro SKF CMSS 2200; [29]

Aceleração máxima	±80 g
Sensibilidade	100 mg
Frequência máxima	10 000 Hz
Temperatura de Funcionamento	-50 a 120 °C

3. FUNDAMENTOS DO CONTROLO DE CONDIÇÃO

Dada a importância já referida das vibrações no controlo de condição desenvolveram-se técnicas de medição e análise de vibrações para a identificação de dano. Esta divide-se na detecção antecipada da avaria, através de medições periódicas e no diagnóstico da sua causa. A avaria é designada, *in* NP EN 13306:2007, por "*cessação da aptidão de um bem para cumprir uma função requerida"* sendo um acontecimento que pode levar ao estado de *em falha* ou *avariado* [2].

Num sinal adquirido sabe-se que a frequência indica o tipo de avaria e/ou o componente avariado, a amplitude diz-nos a gravidade da avaria enquanto a fase permite distinguir avarias com sintomas semelhantes. No entanto, para se fazer a quantificação da vibração, é necessário recorrer a ferramentas matemáticas como a raiz média dos quadrados e transformada de Fourier.

A ferramenta mais básica é a raiz média dos quadrados ou RMS (do inglês *root mean square*) que permite a comparação efetiva de dois sinais adquiridos. O sinal também pode ser adquirido em várias unidades de medida consoante a preferência e também dimensões: deslocamento, velocidade e aceleração quer seja no domínio do tempo seja no domínio da frequência.

A análise em frequência tem como objetivo a determinação das harmónicas que compõem o sinal levando ao conhecimento das frequências mais importantes presentes no sinal em análise. A ferramenta que permitirá conseguir tal objetivo é a transformada de Fourier [30] que, após uma correta aquisição de dados, transformará um sinal no domínio do tempo num espectro de frequência.

3.1. TRANSFORMADA DE FOURIER

Com o algoritmo rápido da transformada de Fourier, FFT (do inglês *Fast Fourier Transform*), é possível transformar um bloco de sinal no domínio do tempo (Gráfico 3.1) para o domínio da frequência (Gráfico 3.2) sem exigir muita capacidade de processamento.

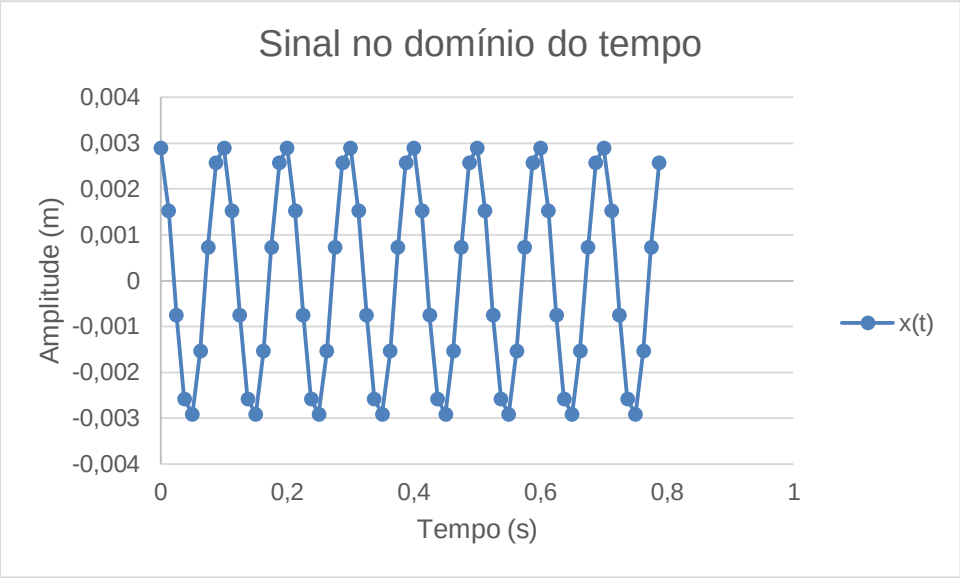

Gráfico 3.1 - Representação gráfica de um sinal no domínio do tempo

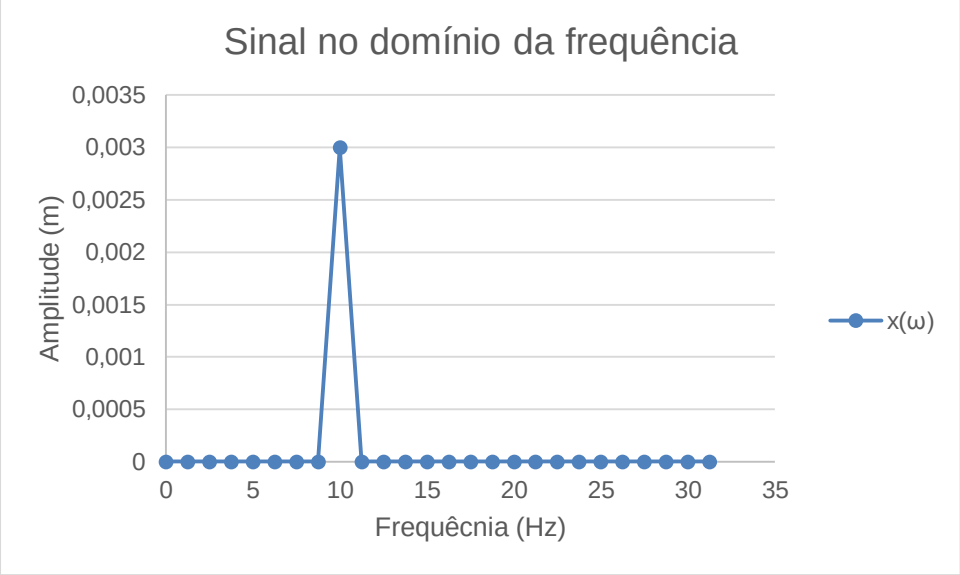

Gráfico 3.2 - Representação gráfica de um sinal no domínio da frequência

Contudo, esta operação matemática, acarreta algumas condições. Tendo em conta que a amostra temporal é discreta e não contínua é preciso ter em conta que os pontos, N, adquiridos estão restringidos a uma potência de 2, como 1024 por exemplo [31]. Outra propriedade a ter em conta é que a transformação só apresentará N/2 linhas igualmente espaçadas. Tal facto deve-se a que cada linha de frequência contém na verdade duas informações: amplitude e fase.

As limitações também afetam a frequência máxima ($F_{máx.}$) que será $N/2.T$. Com estes parâmetros definidos é de esperar que o espectro terá $N/2 + 1$ frequências, onde a 1º, excluindo o zero, será 1/T Hz [32].

As amplitudes das componentes harmónicas que existirem no sinal coincidentes com N/T aparecerão no espectro, enquanto que as não coincidentes se dividirão pelas frequências do espectro mais próximas. Isto levará a que apareçam falsas componentes e amplitude reduzida da frequência verdadeira no espectro de frequência. É como se cada unidade liquefizesse para as vizinhanças dando origem ao fenómeno do *leakage*.

Como na prática nada nos garante que, quando o utilizador efetua a medição da vibração, o período de amostragem escolhido coincida com um número inteiro de ciclos, ou seja o início da aquisição não coincide com o fim ao nível da amplitude, será necessário aplicar ferramentas de correção, que irão fazer coincidir o início da aquisição com o início do ciclo. Este erro pode ser reduzido se escolhermos a janela adequada. Uma das janelas mais usadas na medição de vibração periódicas em manutenção é a janela Hanning.

$$w(t) = 2.sin^2\left(2\pi.\frac{t}{T}\right) \tag{3.1}$$

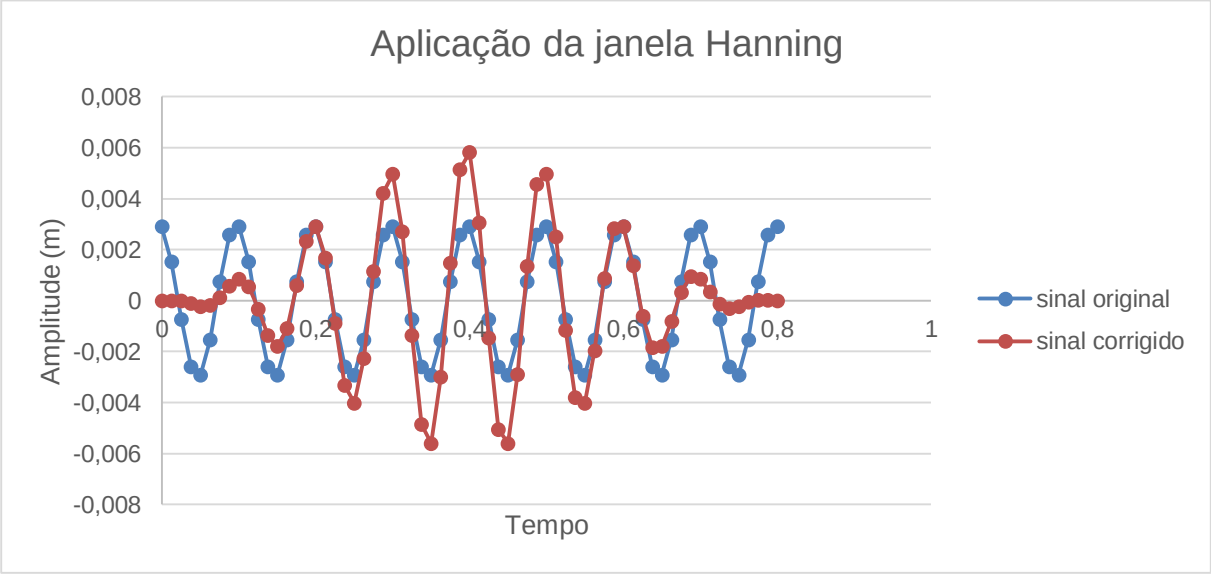

Gráfico 3.3 – Representação gráfica do efeito da janela hanning

A janela Hanning, exemplificada no Gráfico 3.3, de uso mais geral, tem como objetivo corrigir as amplitudes e reduzir o número de componentes falsas do espectro. Neste projeto abordar-se-

á outra janela, a exponencial. Esta última, representada pela equação (3.1) e explicada no Gráfico 3.4 é usada na análise de sinais transientes, que exibam uma queda exponencial, característicos de impactos. τ representa o fator de correção e t o tempo.

$$w(t) = e^{-t/\tau} \tag{3.1}$$

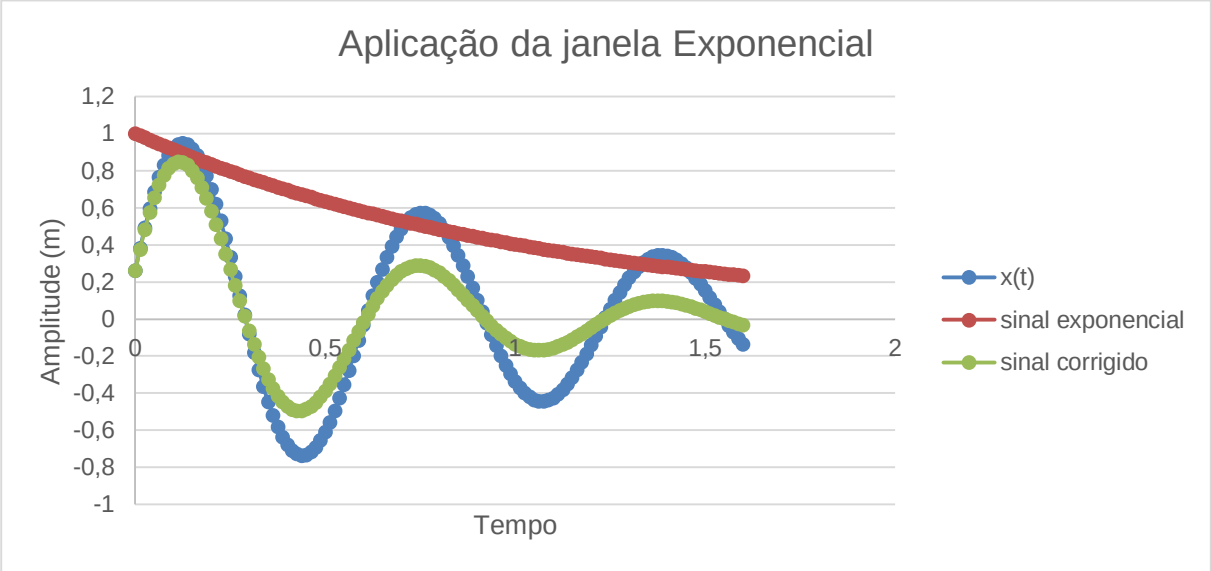

Gráfico 3.4 – Representação gráfica do efeito da janela exponencial

Após a realização da transformada há a possibilidade de se dar outros erros. O efeito de cerca, ou *picket fence effect* explicado na Figura 3.1, tem que ver com a amostragem na frequência. Este erro pode ser atenuado aumentando o tempo de aquisição [1].

Um outro erro é o *aliasing*, representado no Gráfico 3.5, que significa o erro de confundir uma frequência mais alta por outra mais pequena (no domínio do tempo). Para não induzir o erro de *aliasing* é necessário que a frequência de amostragem (fa), seja pelo menos o dobro da maior frequência de interesse (3.2).

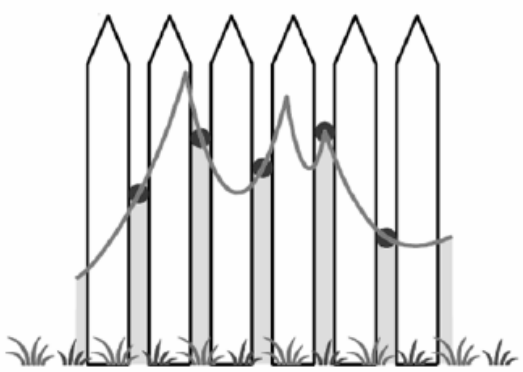

Figura 3.1 – Representação do efeito de cerca; [33]

Também se poderá aplicar um filtro, *anti-aliasing*, que atenuará as altas frequências. Mas devido ao efeito *roll-off*, em que a atenuação afeta o sinal para lá da frequência alvo, é necessária fazer mais uma correção dando origem à nova equação (**3.3**).

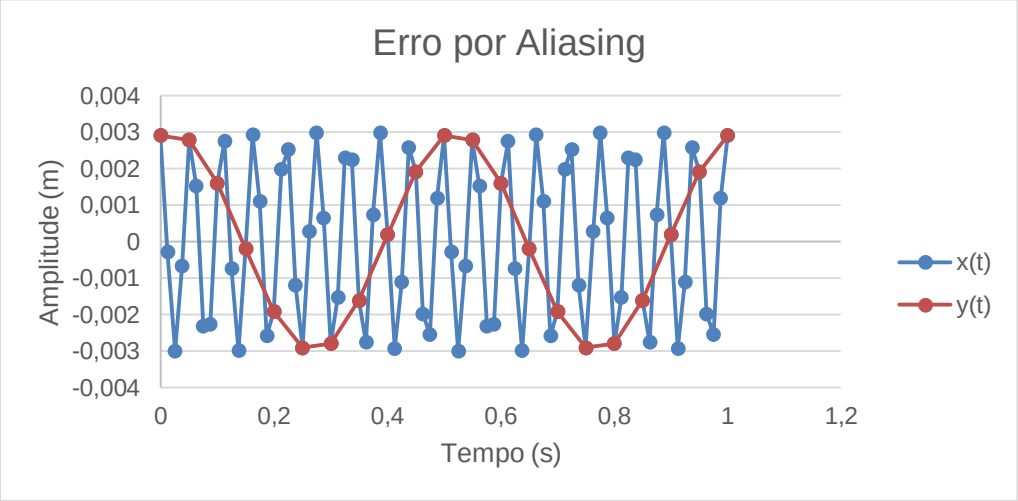

Gráfico 3.5 – Representação gráfica do erro por aliasing.

$$f_a = N/T \geq 2.f_{máx.} \tag{3.2}$$

$$f_a = N/T \geq 2{,}56.f_{máx.} \tag{3.3}$$

3.2. AQUISIÇÃO DE SINAL

Estes erros devem-se ao facto da aquisição de sinal ser discreta e irão deturpar a verdade dos valores. Estes acontecem ou por erro de utilização ou pela natureza da transformada, por isso, é importante conhecer como se procede à aquisição de sinal.

O sensor, neste projeto um acelerómetro, mede uma dimensão física e transforma-a num sinal eléctrico, normalmente em voltagem, que passará depois por um condicionador de sinal que

converterá este sinal analógico em digital. No acelerómetro Phidgets o condicionador faz parte da placa de aquisição de sinal necessitando apenas o utilizador de um computador para ler e interpretar através da normal entrada USB (*universal serial bus*) com uma alimentação de 5 volts.

É importante conhecer as caraterísticas do sensor adequando à realidade que se quer aplicar. Frequência de aquisição e nível de saturação são de grande relevância, mas não se pode descurar outros parâmetros como a resolução, ganho e precisão.

A resolução indica o mínimo número lido pelo sensor dada a alteração da dimensão física estando relacionada com os bits de precisão. Por exemplo, com 5 volt de *input* levará a uma resolução de 76,29 µV numa placa de 16bit. Pode-se aumentar a sensibilidade do conversor através do ganho, ou amplificação [33].

Denota-se que existem muitas variáveis que tornam a aquisição de dados numa tarefa por vezes difícil. O valor medido será sempre uma aproximação ao real, e tanto melhor quanto for a precisão do sensor.

Muitas vezes recorre-se ao uso de filtros para remover algumas irregularidades no sinal, como descrito na Figura 3.2. Os filtros podem ser analógicos ou digitais, discretos ou contínuos, lineares ou não lineares. Neste projeto, dado o requerimento da normalização aqui usada, precisou-se de recorrer ao uso de filtros, como o filtro passa-baixo e o filtro passa-alto.

O filtro passa-baixo permite atenuar a amplitude de frequências mais altas que o valor alvo, definido pelo utilizador, e permite obter um sinal no tempo e em frequência mais nítido. A combinação de ambos os filtros forma o filtro passa-banda [1]. De notar que devido ao efeito *roll-off* criado pela utilização de filtros serão apresentadas, numa fase seguinte, somente N/2,56 linhas no espectro de frequência,

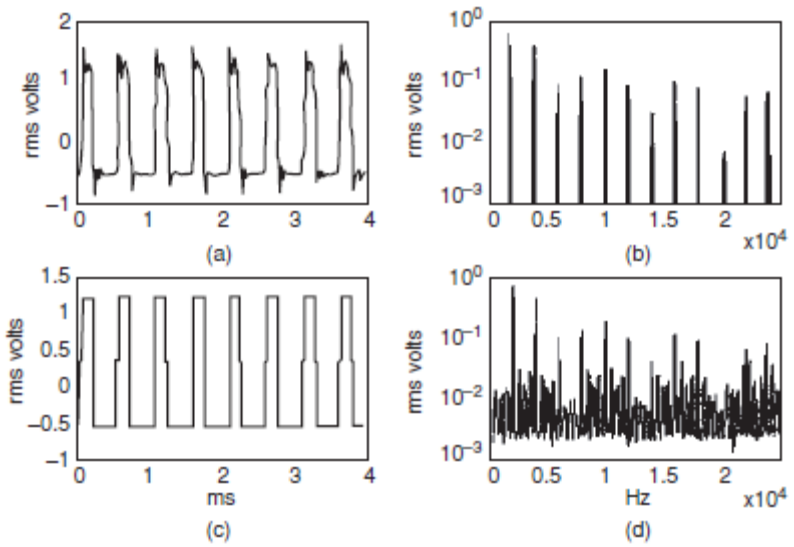

Figura 3.2 – a) Sinal filtrado com passa-alto; b) Espectro de frequência de a); c) Sinal sem filtragem; d) Espectro de frequência de c); [1]

3.3. NORMALIZAÇÃO

Dada a panóplia de fatores que poderão afetar a aquisição de dados houve a necessidade de consolidar a forma como esta é feita. Assim criou-se normalização que definisse procedimentos e metodologias. Neste projeto ir-se-á abordar normas com foco na detecção de dano estrutural e em equipamentos.

A norma ISO 10816, da International Standards Organization, para a detecção de dano por meio de vibrações mecânicas, é a mais usada na indústria. Esta estabelece procedimentos para a medição e classificação da vibração mecânica em máquinas rotativas. Os valores de alerta são definidos consoante o tipo de máquina que por sua vez ditará onde deverá ser fixado o sensor. A aquisição de dados terá de ser feita sempre da mesma forma controlada.

Após a aquisição de um intervalo de pontos, em velocidade (mm/s), é calculado o valor global, pela equação (3.4). Previamente a esta aquisição é necessário usar um filtro passa-banda entre os 10Hz e os 1000Hz. Como já referido, estes valores adquiridos, serão comparados com valores tabelados pela norma e despoletarão diferentes níveis de alerta [34].

$$\mathrm{RMS} = \sqrt{\frac{\sum_{i=0}^{N-1} x^2}{N}} \tag{3.4}$$

Estes valores de aquisição são depois comparados com a Figura 3.3 e, assim, definidos os alertas consoante a gravidade.

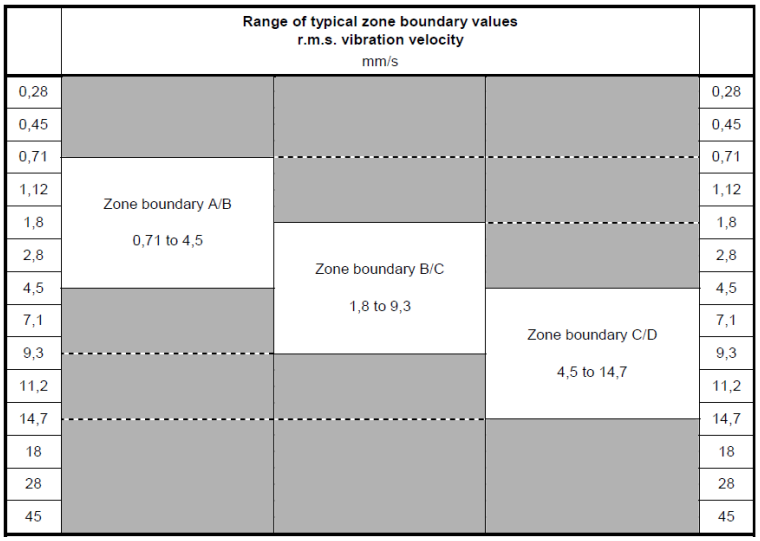

Figura 3.3 – Quadro da norma ISO 10816 para os valores fronteira típicos de vibração (mm/s) em RMS; [34]

Neste projeto também se teve em conta o controlo de condição de estruturas e introduziu-se a possibilidade do utilizador poder usar também normas como a norma portuguesa NP 2074.

A NP tem como objetivo estabelecer limites de vibrações impulsivas com o intuito de evitar danos que possam ocorrer nas estruturas. Os requisitos de instrumentação indicam que, para além da obrigatória calibração segundo o normativo NP EN ISO 17025:2005, o acelerómetro deve ser triaxial, ter capacidade de registar frequências de 2Hz a 80Hz e de velocidades de pico entre 0,5 mm/s a 100 mm/s.

Os valores limites são definidos na norma dentro de três tipos de estruturas em ligação com três intervalos de frequências dominantes, f. Na Tabela 3.1 estão representados os valores limites recomendados para a velocidade de vibração, de pico, em mm/s.

Tabela 3.1 – Valores limites recomendados para a velocidade de vibração (mm/s) de pico; [35]

Tipo de estruturas	**Frequência dominante, f**		
	f ≤ 10 Hz	10 Hz < f ≤ 40 Hz	f > 40 Hz
Sensíveis	1,5	3,0	6,0
Correntes	3,0	6,0	12,0
Reforçadas	6,0	12,0	40,0

Classifica-se as estruturas em sensíveis, correntes ou reforçadas consoante o estado de conservação, esbeltez e o seu valor patrimonial. A medição é feita com a fixação do sensor à estrutura, seguindo a Norma ISO 5348:1998. A velocidade de vibração calculada pela equação (3.5), num determinado intervalo de tempo nos três eixos de leitura do sensor, onde se determinará o máximo valor dessa vibração em velocidade. Em seguida determina-se qual a frequência dominante, sabendo qual das frequências presentes tem maior amplitude, e assim, sabe-se se o valor foi excedido ou não [35].

$$v_{máx} = \left| \sqrt{v_L^2(t) + v_V^2(t) + v_T^2(t)} \right| \tag{3.5}$$

4. PLATAFORMA ANDROID

A plataforma Android faz uso exclusivo do seu sistema operativo com o mesmo nome. Este baseia-se numa versão modificada do Linux e é, neste momento, desenvolvido pela Google. Cada nova versão do sistema recebe um nome de uma sobremesa e são representadas pela variação do logotipo verde reconhecido como android, na Figura 4.1 [36].

Figura 4.1 – Logotipo representativo so sistema operativo Android; [36]

Neste momento a plataforma Android está presente em dispositivos móveis de comunicação, relógios, televisões, nos automóveis e nos *tablets*. Hoje, será tão importante saber programar para os dispositivos móveis como é para computador. É um sistema operativo de fácil uso e de rápida aprendizagem em que as janelas seguem sempre o mesmo princípio.

A janela é o que o utilizador vê ao interagir com a aplicação que, normalmente, ocupa todo o ecrã do dispositivo. Estas janelas são governadas pelas atividades que contêm todas as classes programáticas, típicas da linguagem de programação, Java. Nesta aplicação, a cada janela, corresponde uma atividade, excetuando, a atividade de definições que contém várias.

Em todas as janelas podem haver mais de que uma maneira de navegação. Neste projeto a barra de ação está sempre presente bem como os botões de ação normais ao S.O.

4.1. A ESCOLHA DESTA PLATAFORMA

A escolha desta plataforma para este projeto prendeu-se com duas razões primordiais: acessibilidade das plataformas de desenvolvimento e a capacidade dos dispositivos.

Neste contexto de plataformas de desenvolvimento de fonte aberta geralmente os *softwares* de desenvolvimento são gratuitos e de fácil aprendizagem. Nesta experiência com o Android Studio, plataforma de desenvolvimento oficial, surgiram algumas dificuldades com a

comunicação entre dispositivo Android e computador, aplicação de bibliotecas e ferramentas para o desenvolvimento e alguns percalços visto esta plataforma ainda ser muito recente.

Tendo em conta o que já foi referido no capítulo anterior sabe-se que os dispositivos móveis que suportam o S.O. Android são cada vez mais evoluídos. A evolução dos seus processadores e de memória levam a que estes tenham sido uma opção fácil de considerar.

Podia-se argumentar que um projeto baseado nesta plataforma seria dispendioso dado o valor de aquisição do dispositivo a ser usado para a aquisição de dados, contudo estes trazem já maior valor acrescentado. Fazendo a comparação com plataforma Arduino, e tendo em conta os objetivos deste projeto, denota-se que o Arduino, de baixo custo de aquisição, implicaria um adicional custo para se poder dar ao utilizador a representação visual dos valores adquiridos pelo acelerómetro dada a necessidade de adquirir um ecrã LCD externo. Este facto revela que, apesar de um custo objetivamente mais alto de aquisição, a plataforma Android já traz consigo um módulo GPS, módulo de comunicação, câmaras, e um ecrã obviamente. A visão para este projeto é de um dispositivo com o S.O. Android, como um vulgo *tablet*, possa ser instalado de forma permanente numa estrutura a ser controlada e daí todo este valor, trazido pela plataforma, possa ser explorado.

4.2. SISTEMA OPERATIVO

Os dispositivos móveis estão em constante evolução e nos últimos anos essa foi exponencial. De realçar as cada vez maiores capacidades do *hardware* destes dispositivos que já cotam com 4 GB de memória virtual e processadores capazes de 2.1 GHz.

O S.O. é usado por milhões de utilizadores em todo o mundo e é continuamente alvo de melhorias. Daí, após consulta dos dados da Google, representados no Gráfico 4.1, verificou-se que já não faz sentido constringir o código da aplicação para versões mais antigas do sistema e assim só suportar as versões superiores, ou iguais, ao 4.0 também conhecido como *ice cream sandwich*.

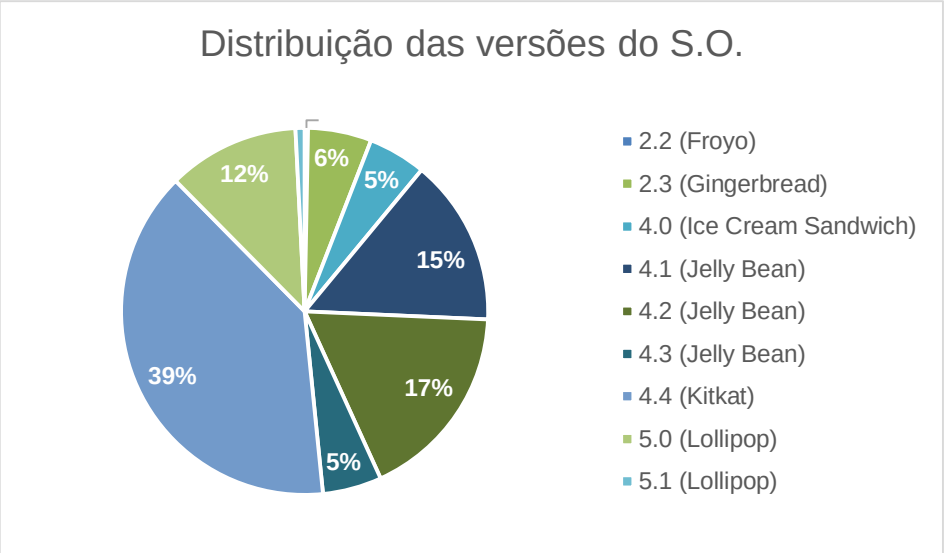

Gráfico 4.1 - Representação gráfica da distribuição das versões do sistema operativo; [37]

Esta restrição, ficando-se pelo 4.0, abrange mesmo assim um vasto mercado de utilizadores. Os dispositivos que apenas suportam esta versão, já antiquada, são de baixo custo, mas a aplicação, poderá ser usada em todas as versões superiores de S.O. Como exemplo, consegue-se adquirir um dispositivo com o aspeto de forma de um *tablet* por poucas dezenas de euros.

A distribuição deste sistema é muito eficaz. Na Tabela 4.1 verifica-se a imensidão do mercado que este abrange.

Tabela 4.1 - Estatísticas e factos sobre o sistema operativo; [38]

	Valores
Cota de mercado global	78,4%
Número diário de ativações	1 500 000
Número de aplicações descarregadas	50 Mil Milhões
Preço médio de uma aplicação	0.05€

4.3. PLATAFORMA DE DESENVOLVIMENTO

O programa de desenvolvedor para Android é bastante abrangente e de fácil acesso. Chamado de *developer.android* consiste em documentação de apoio ao desenvolvimento, vídeos de

aprendizagem e o acesso à plataforma informática de desenvolvimento Android Studio. Tudo isto é disponibilizado gratuitamente.

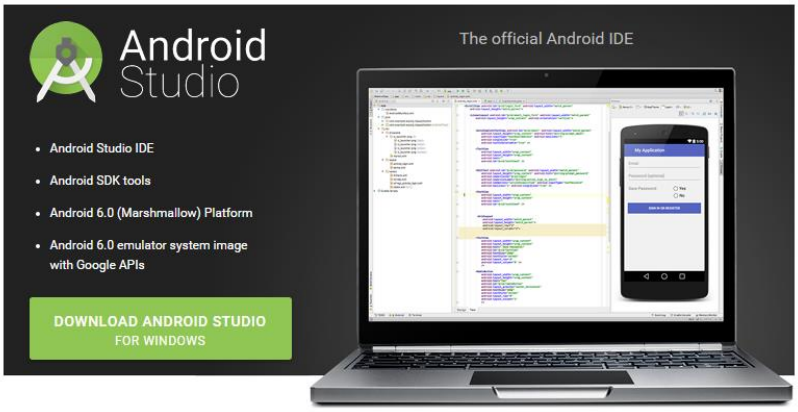

Figura 4.2 – Representação do Android Studio; [36]

Esta é a plataforma de desenvolvimento oficial para este sistema operativo, descrita na Figura 4.2. Contém várias ferramentas que auxiliam a criação e o teste das aplicações. O Android Studio conta com meios de personalização, configuração, variadas bibliotecas com diferentes atributos e de monitorização. Isto dá a possibilidade de criar um projeto flexível e que facilmente poderá ser evoluído.

Para o teste das aplicações pode-se recorrer ao emulador de dispositivos móveis, chamado de Android Virtual Device Manager, que irá permitir ao utilizador verificar se a aplicação corre num variado número de dispositivos. Contém também ferramentas de depuração e de monitorização dos recursos a serem usados pelo dispositivo receptor da aplicação.

4.4. ESPECIFICIDADES DA PROGRAMAÇÃO EM ANDROID

A criação de uma aplicação para esta plataforma inicia-se na definição do alvo de desenvolvimento, em termos de dispositivo e geração do S.O. Após este passo é apresentado o ambiente de desenvolvimento que tem duas partes fundamentais: o ficheiro *.java* onde se programa é o núcleo da aplicação e o ficheiro *.xml* contém as características da interface. Naturalmente existem muitas facetas, mas que não serão aqui abordadas.

O ficheiro *AndroidManifest.xml*, criado automaticamente, é essencial para a execução da aplicação já que, descreve as atividades da aplicação, dá nomes às aplicações, declara permissões e lista as bibliotecas que a aplicação irá usar [39].

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="(…)" >
   <uses-permission android:name="android.permission.SEND_SMS"/>
   <uses-permission android:name="android.permission.internet"/>
   <application
      android:allowBackup="true"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >
      <activity
         android:name=".MainActivity"
         android:label="@string/app_name" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
      <activity
         android:name=".SettingsActivity"
         android:label="@string/title_activity_settings"
         android:parentActivityName=".MainActivity" >
         <meta-data
            (…)

   </application>

</manifest>
```

Figura 4.3 – Exemplo de programação de um *manifest.xml*

Os recursos precisos para o desenvolvimento de arquiteturas em Java socorrem-se muitas vezes de bibliotecas e ferramentas externas para atingir um certo objetivo. No caso deste projeto foi necessário recorrer a várias dessas ferramentas. A placa de aquisição tem a sua própria biblioteca, tanto para gerir a comunicação por USB como para receber a informação, por exemplo.

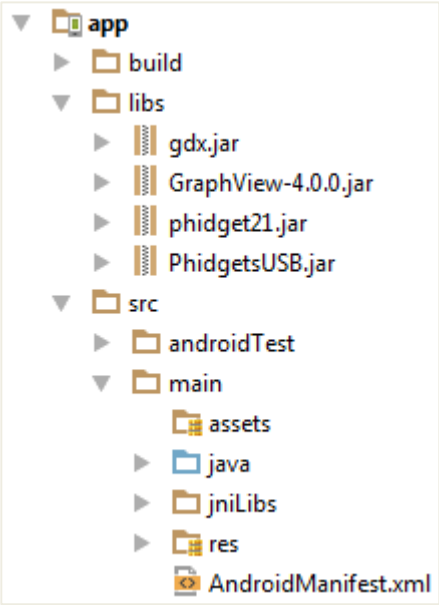

Figura 4.4 – Demonstração da presença de bibliotecas no projeto

```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    android:padding="10dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Sensor:"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:id="@+id/sensor_name"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Version:"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:id="@+id/sensor_version"
        android:layout_below="@+id/sensor_name"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true" />
```

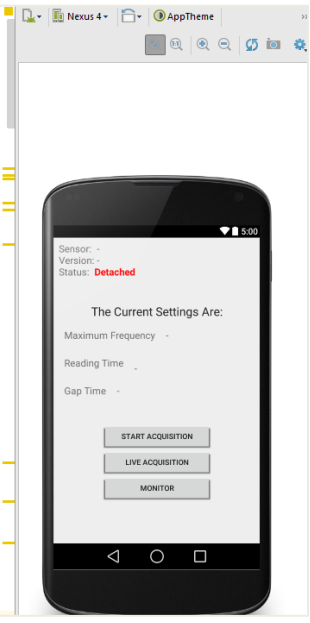

Figura 4.5 – Exemplo de programação de uma interface e pré-visualização.

A criação da tão importante interface gráfica será feita no editor de XML. Pode-se criar diferentes ambientes de desenho, com diferentes orientações, botões, caixas de texto, cores e muitas outras formas de comunicar com o utilizador, como na Figura 4.5.

Para interligar tudo usam-se os ficheiros Java com a programação que irá fazer funcionar todo o projeto. Num primeiro momento instanciam-se as variáveis que serão usadas. Dentro de onCreate() criam-se todos os métodos necessários para exibir a atividade, exemplificado na Figura 4.6.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

(…)

}
```

Figura 4.6 – Exemplo do método onCreate() para a atividade principal.

4.5. DISPOSITIVOS DE TESTE

O dispositivo aconselhado para esta aplicação seria um dispositivo com ecrã de 8'' (20 cm aproximadamente), com sistema operativo 4.0 ou superior, presença de leitor de cartões, módulo de comunicação de terceira geração e, preferencialmente, com resistência a líquidos e pó. No desenvolvimento deste projeto usou-se um dispositivo móvel de pequenas dimensões e outro de 8''.

Na Tabela 4.2 pode-se verificar as características destes aparelhos. De notar que os seus atributos não são elevados, comparando com os dispositivos atuais, mas que demonstraram serem suficientes para a aquisição de sinal e posterior processamento.

Tabela 4.2 – Características dos aparelhos de teste; [40], [41]

Sony Sola	
Dimensões do ecrã	3,7’’
Sistema operativo	4.0
Processador	1 GHz
Memória RAM	512 MB
Capacidade de armazenamento	8 GB (expansível até 32GB)
Acer Iconia A1-810	
Dimensões do ecrã	8’’
Sistema operativo	4.2.2
Processador	1,2 GHz
Memória RAM	1 GB
Capacidade de armazenamento	8 GB (expansível até 32 GB)

5. REALIZAÇÃO DO PROJETO

Após o estudo do controlo de condição por análise de vibrações e suas ferramentas passou-se à concretização deste projeto sob a forma da realização de um analisador de vibrações, com opções de alerta e várias formas de visualização e comunicação, numa plataforma de fonte aberta.

Numa aplicação a comunicação visual é muito importante e, nesta fase inicial, adoptou-se uma abordagem simplista ao longo das diferentes atividades que constituem o programa. Após adquirir o acelerómetro, parte integrante do projeto, criaram-se vários protótipos e introduziram-se mais funcionalidades a cada passo. Por fim demonstrou-se que as diferentes ferramentas externas se adequam à realidade do processamento de sinal e consequentes formas de alerta.

5.1. INTERFACE E O SEU PORQUÊ

O *design* de uma aplicação deverá seguir um conjunto de boas práticas em prol da melhor utilização do portador. Segundo Ricardo Queirós, *in* Desenvolvimento de Aplicações Profissionais em Android, pode-se organizar estas práticas em cinco partes: a adaptabilidade, a navegação, a personalização, a acessibilidade e o uso de animações e gráficos, representado na Figura 5.1 [42].

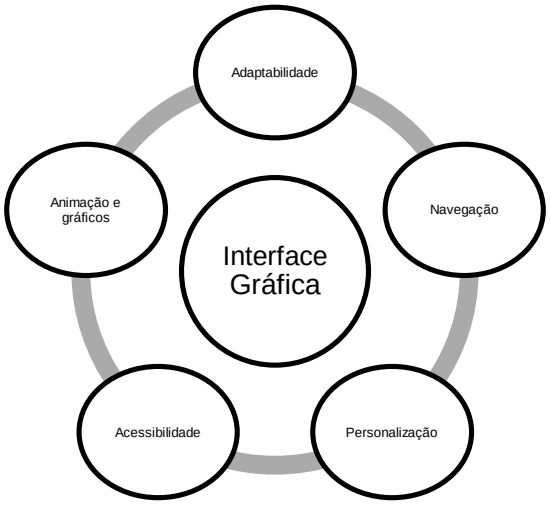

Figura 5.1 - Boas prácticas para o bom desenvolvimento de uma interface gráfica; adaptado de [42]

Uma aplicação deve ser modular e adaptar-se aos variados dispositivos, fazendo uso correto da navegação, que é fundamental para a experiência de utilização. A personalização é uma característica distintiva do Android e deve ser aliada da acessibilidade, evitando o erro de criar uma *app.* muito confusa e pesada, degradando a utilização.

Neste projeto a hierarquia das janelas é simples e com poucos níveis. Na Figura 5.2 estão representados os três níveis da aplicação e os caminhos possíveis para lá chegar. De notar que se exclui a janela de definições.

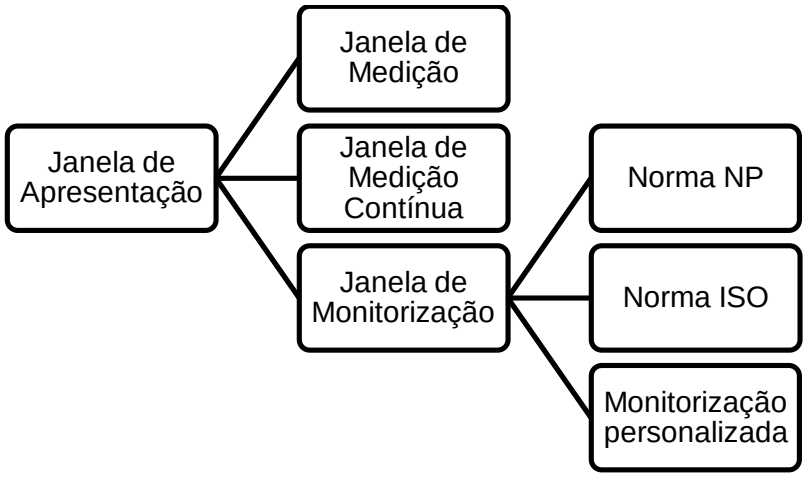

Figura 5.2 – Representação da hierarquia da aplicação

Na janela de apresentação o utilizador poderá ser direcionado à janela de medição, de medição contínua, de monitorização e, desta última, chegar à monitorização personalizada, segundo norma NP ou segundo norma ISO através dos botões de ação presentes nesta.

5.1.1 JANELA DE APRESENTAÇÃO

A primeira janela, que será a atividade principal, tem como função primordial dar início à comunicação entre a placa de aquisição e o dispositivo. A parte central será ocupada com um indicador de presença e informações fundamentais sobre o sensor. Neste poder-se-á também ver as características da aquisição de dados que posteriormente será feita noutras atividades.

A frequência máxima, tempo de leitura e intervalo de aquisição são mostradas na zona central enquanto que o nome, versão e estado da ligação do sensor se situam no topo.

Figura 5.3 - Representação da janela de apresentação e suas características.

Sendo esta a janela de apresentação é o único sítio onde se poderão alterar as definições da leitura por meio do ícone que se encontra mais à direita na barra de tarefas. Nesta barra também há a possibilidade de se ir ao encontro de mais informação sobre o sensor através do ícone de informação.

5.1.2 JANELA DE MEDIÇÃO

A segunda e terceira janela têm como função apresentar a variação do sinal adquirido. Na zona central deverá ser representado o gráfico. O tipo de gráfico apresentado representa a escolha previamente feita pelo utilizador na atividade de definições.

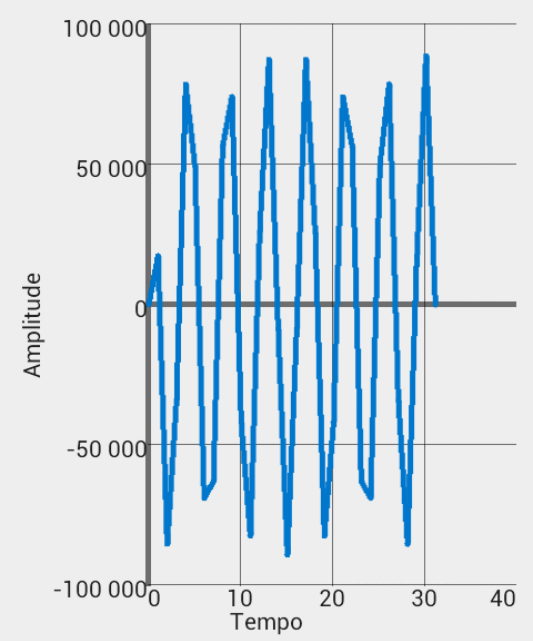

Figura 5.4 – Exemplo da representação gráfica da aplicação

Este grafismo disponibiliza a representação dos pontos recolhidos, mas não só. Cada ponto, que foi recolhido para a representação, está marcado e, após ser pressionado, mostra ao utilizador o seu valor e posição.

5.1.3 JANELA DE DEFINIÇÕES

A janela de definições é ativada pelo ícone habitual desta plataforma. Esta janela contém três patamares mais um primário para definições gerais da aquisição de dados, como representado na Figura 5.5.

Nas definições gerais, representadas na Figura 5.6, pode-se consultar o nome do aparelho que virá a ser útil no caso da existência de vários em uso na unidade industrial. Os seguintes referem-se à leitura propriamente dita como a frequência máxima, o tempo de leitura e a sua dimensão física. No degrau seguinte o utilizador é direcionado para as opções avançadas que dispõe a aplicação.

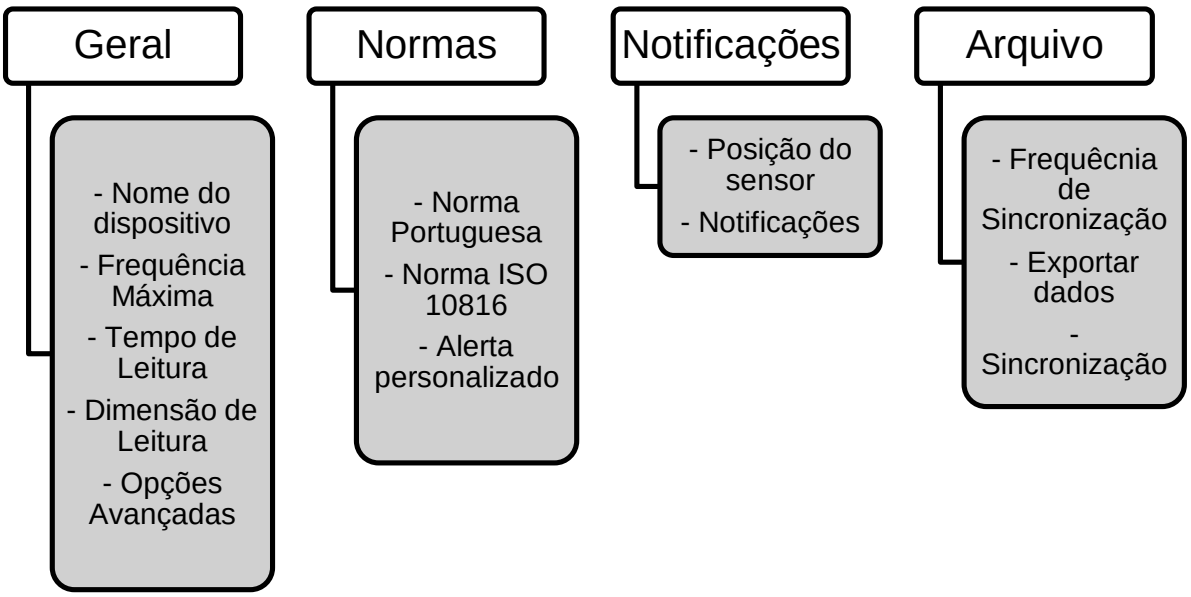

Figura 5.5 – Representação dos patamares da janela de definições

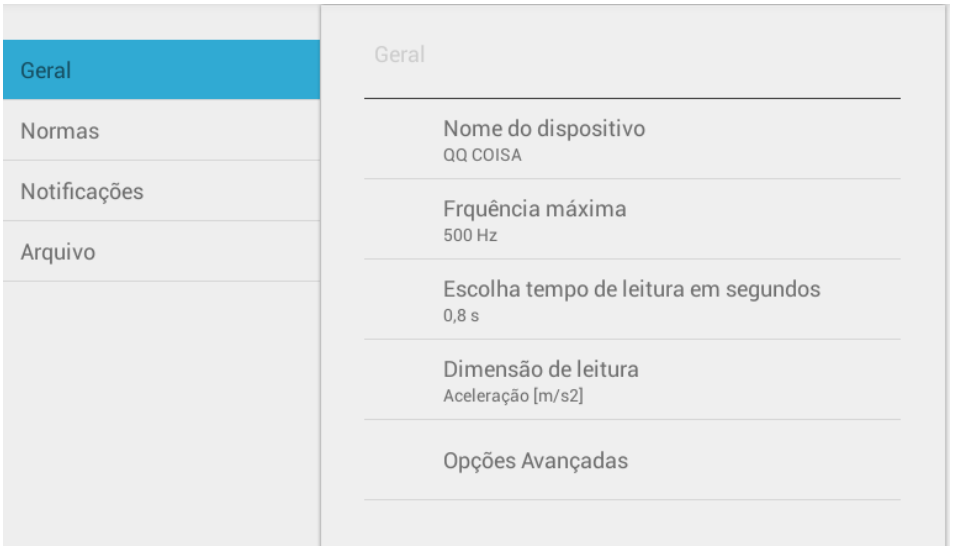

Figura 5.6 - Representação das janela de definições, geral.

Neste patamar, representado na Figura 5.7, encontram-se as opções de utilização de filtros e seus parâmetros, aplicação de janelas, como a janela de Hanning, e por fim o eixo de leitura em que se quer efetuar a aquisição.

Figura 5.7 - Representação da janela de definições avançadas

No patamar seguinte, exemplificado na Figura 5.8, definem-se as particularidades das normas. Em primeiro surge a norma portuguesa e dá a possibilidade de escolha do tipo de estruturas, que estão presentes na norma, em análise. Em seguida surge a norma internacional ISO 10816 com a possibilidade de definir os valores de alarme, dentro dos intervalos para cada zona, como determina a norma.

A principal razão para este patamar é mesmo a definição de alertas seguindo as recomendações das normas indicadas. Por isso foi criada também a opção de definir alertas personalizáveis.

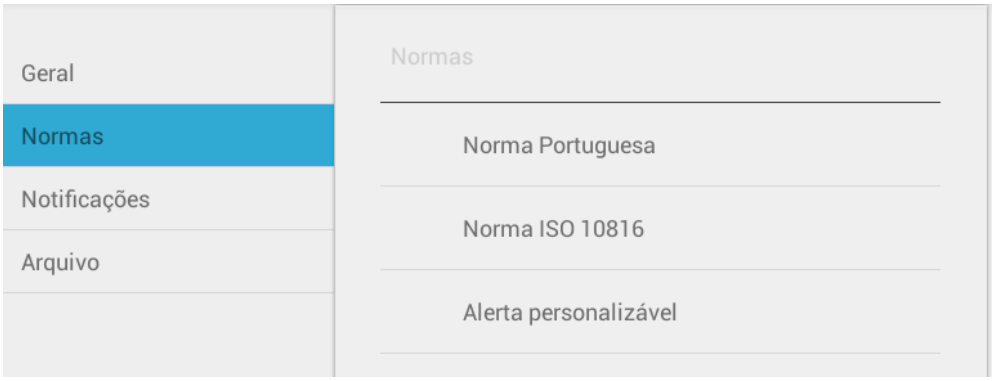

Figura 5.8 - Representação da janela de definições, janela de normas

A terceira divisória lida com os alertas que serão ativados consoante a gravidade ditada pelas normas, como na Figura 5.9. É dada a possibilidade de ativar alarmes sonoros e o envio de avisos escritos para um receptor determinado para cada utilizador.

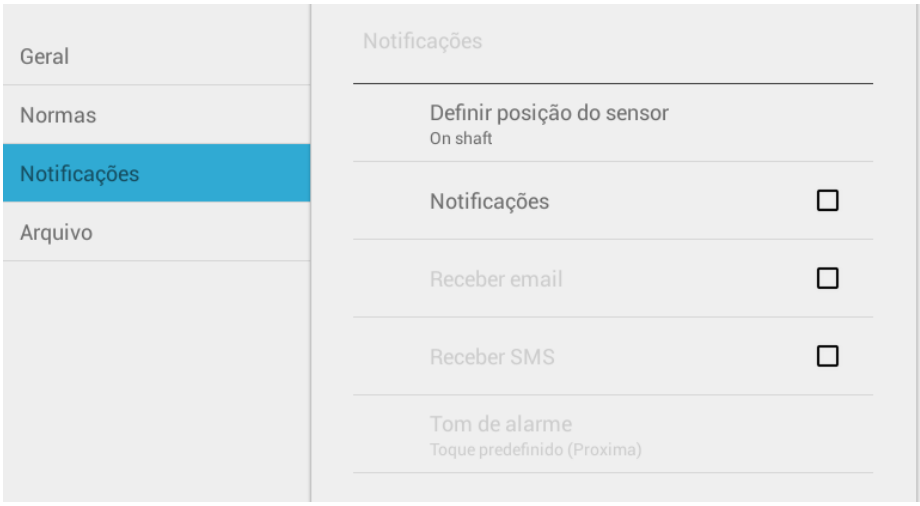

Figura 5.9 – Representação da janela de definições, patamar das notificações

Por fim tem-se o patamar que será desenvolvido a pensar num futuro de comunicação entre vários dispositivos inseridos na unidade. Recorrendo à partilha de um ficheiro texto com características essenciais à boa operação do equipamento que depois será enviado para um painel de controlo central facilitando assim a observação de vários equipamentos.

5.1.4. JANELA DE MONITORIZAÇÃO

A monitorização é iniciada após o utilizador entrar na atividade que permite escolher qual a norma que quer seguir. Esta, numa primeira fase, faz uma avaliação das condições iniciais escolhidas para verificar se cumprem os pressupostos das normas, tal como os valores balizados das zonas da norma ISO 10816, e inicia a atividade que efetivamente faz a monitorização.

É constituída por um cabeçalho que indica a norma, antes escolhida, e logo abaixo o valor global limite mais próximo àquele que se está a adquirir, representado na Figura 5.10.

Figura 5.10 – Representação da janela que apresenta as informasções sobre a monitorização

5.2. ACELERÓMETRO USADO

Como já referenciado nos capítulos anteriores este projeto será baseado na utilização de um acelerómetro externo ao dispositivo móvel por intermédio de uma ligação USB. Na realização do projeto utilizou-se, maioritariamente, o modelo 1041 da marca Phidget, com as caraterísticas na Tabela 5.1, e Tabela 5.2.

Este sensor tem capacidade de adquirir acelerações em três eixos distintos em unidades de aceleração gravítica, g, até um máximo de 8.1g sendo que até 8g as leituras são precisas. O intervalo de aquisição mais pequeno é de 1 milissegundo até um máximo de 1 segundo. É preciso também ter em conta que o *software* deste só aceita entradas de *data rate* de 1,2,4,8 e múltiplos de 8 milissegundos [43].

Tabela 5.1 – Características do sensor Phidget 1041; [43]

Aceleração máxima	±8 g
Resolução	976,7 µg
Ruído branco	2.5 mg
Desfasamento mínimo	23.1 µg

Tabela 5.2 – Características da placa de aquisição Phidget Spatial; [43]

Consumo máximo de corrente	30 mA
Intervalo de aquisição mínimo	1 ms
Intervalo de aquisição máximo	1 s
Conversão de digital para analógico	16 bits
Voltagem mínima	4.4 V
Voltagem máxima	5.3 V
Temperatura mínima de operação	-40 ºC
Temperatura máxima de operação	85 ºC

Existe a possibilidade de usar este sensor por intermédio de servidor mas não foi essa a abordagem aqui usada. Dos manuais do fabricante lê-se "*tablets com porta usb e versões de android 3.1, ou superiores, podem controlar a placa Phidgets através de ligação direta*" [44]. Contudo é necessário usar um cabo USB intermédio chamado *on the go* (OTG) para permitir a comunicação como o que está representado na Figura 5.11.

Figura 5.11 – Imagem de dispositivo com sensor ligado com respectivo cabo *OTG*.

5.3 EVOLUÇÃO DOS PROTÓTIPOS

O primeiro protótipo, realizado para a pré-apresentação, só se propunha adquirir sinal do sensor e transmiti-lo em seguida ao utilizador. Esta primeira tentativa foi baseada nos exemplos da Phidget, exposto na Figura 5.12, para a linguagem de programação Java que torna possível este projeto [19].

Neste exemplo é de destacar a primeira linha: com.phidgets.usb.Manager.*Initialize*(**this**). Esta dá início à comunicação entre a placa de aquisição e o dispositivo móvel que, na fase de encerramento do programa, em onDestroy() (chamado imediatamente antes da aplicação ser encerrada), será parada pela sua inversa - com.phidgets.usb.Manager.*Uninitialize*(). Seguidamente cria-se o objeto SpatialPhidget().

Após estar criado o objeto pode-se dar início aos métodos responsáveis por receber informação e transmitir dados. Em addAttachListener e em addDetachListener, como o nome indica, será onde se receberá a indicação de que o sensor está ligado ou desligado. De maior preponderância para o projeto é o método addSpatialDataListener que tratará da recepção da informação, neste caso, dos valores de aceleração para os três eixos.

```
com.phidgets.usb.Manager.Initialize(this);
   spatial = new SpatialPhidget();
   spatial.addAttachListener(new AttachListener() {
      public void attached(final AttachEvent ae) {
         AttachDetachRunnable handler = new AttachDetachRunnable(ae.getSource(), true);
         synchronized(handler)
         (…)
      }
   });
   spatial.addDetachListener(new DetachListener() {
      public void detached(final DetachEvent ae) {
         AttachDetachRunnable handler = new AttachDetachRunnable(ae.getSource(), false);
         synchronized(handler)
         (…)
      }
   });
   spatial.addSpatialDataListener(new SpatialDataListener() {
      public void data(SpatialDataEvent sde) {
         runOnUiThread(new SpatialDataRunnable(
               sde.getData()[0].getAcceleration()[0],
               sde.getData()[0].getAcceleration()[1],
               sde.getData()[0].getAcceleration()[2]));
      }
   });
```

Figura 5.12 – Código exemplo fornecido pela Phigdet

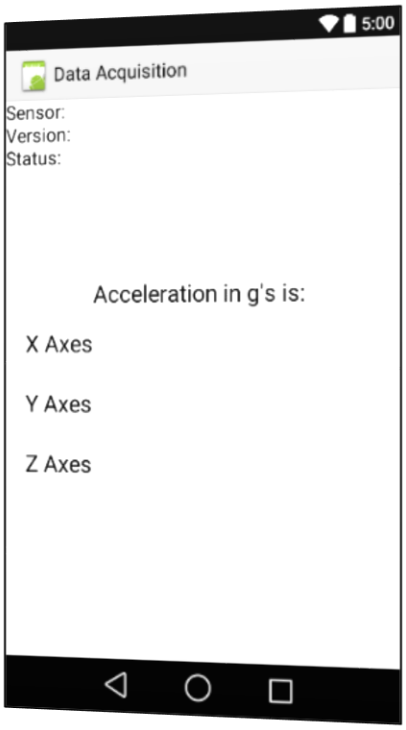

Figura 5.13 – Primeira aplicação realizada

Na Figura 5.13, pode-se ver o ecrã de apresentação da primeira experiência. Esta imagem foi retirada do emulador da plataforma Android Studio na versão 5.0 do S.O. e, dado a limitação desta, não é possível replicar a ação do sensor.

Nesta demonstração já se tentou cumprir alguns requisitos subjacentes ao projeto. Na secção superior, onde se lê sensor, versão e estado, é dado *feedback* ao utilizador sobre o sensor como o seu nome, a sua versão e se se encontra ligado ou desligado.

Dominando a janela de apresentação, na zona central, tem-se as leituras obtidas pelo acelerómetro em 3 eixos distintos, o máximo possível neste sensor. Os dados são nativamente adquiridos em g's.

5.3.1. INTRODUÇÃO DA REPRESENTAÇÃO GRÁFICA

Na nova iteração introduziu-se a possibilidade do utilizador iniciar uma segunda atividade. Esta, seguindo a premissa anterior de cumprir os requisitos de projeto, apresenta na parte central os gráficos. Começando pelo gráfico estático da Figura 5.14 e depois pelo gráfico dinâmico.

As legendas adaptar-se-ão automaticamente a cada um deles assim como as escalas. Este objetivo é fulcral à realização do analisador e foi cumprido usando uma biblioteca não nativa ao Android de nome GraphView [45].

O recurso a bibliotecas externas é muito comum no desenvolvimento de aplicações e esta não é exceção. Como é de fonte aberta, mantendo a mesma filosofia do S.O., pode-se modificar e usar desde que se cumpra com as exigências das licenças de uso definidas pelo autor.

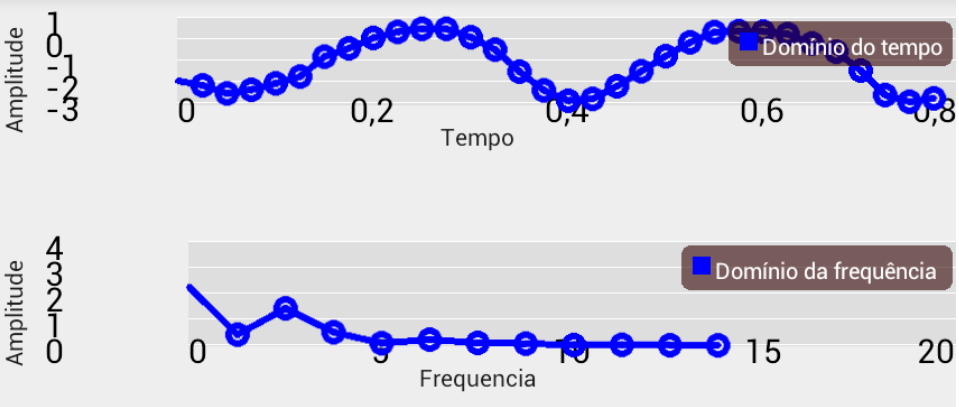

Figura 5.14 – Exemplo da representação gráfica da aplicação

A adoção da biblioteca foi realizada recorrendo aos exemplos que são disponibilizados na página do criador [46]. A escolha recaiu nesta biblioteca, em detrimento de outras, devido à facilidade de obtenção de documentação e apoio encontrado. A capacidade de criar gráficos de linhas ou barras, várias series ao mesmo tempo, criar legendas personalizáveis e adicionar a capacidade de poder saber qual o valor marcado naquele ponto fazem, desta biblioteca, versátil e eficaz.

Iniciado o objeto GraphView é dado início do processo de construção do gráfico. Primeiramente começa-se por transportar os dados, no método generateData(), do vector dataArray para um vector próprio que a biblioteca do Graph View compreenda. Posteriormente cria-se o ambiente gráfico desde as legendas, fundos e cores a funções como marcação de pontos e a possibilidade de haver ampliação.

A apresentação gráfica é importante numa aplicação e considerou-se que esta ferramenta cumpre essa tarefa de forma excelente e empresta mais funcionalidades que poderão ser úteis ao utilizador quando analisa os dados recolhidos.

5.3.2. GRÁFICO DINÂMICO

Na seguinte etapa criou-se o gráfico dinâmico. A diferença fundamental está na programação da atualização da imagem que o utilizador vê consoante os dados recolhidos. Este método recolhe dados, de dimensão predefinida, e cria o gráfico para depois esperar alguns milissegundos, enquanto recolhe novos dados, para mostrar novo gráfico, Figura 5.15.

```
protected void onResume() {
   super.onResume();
   mTimer1 = new Runnable() {
      @Override
      public void run() {
         mSeries.resetData(generateData());
         mHandler.postDelayed(this, (DR*1.15));}
   };
   mHandler.postDelayed(mTimer1, (DR*1.15));
}
```

Figura 5.15 – Representação do método onResume() da terceira atividade.

Essa diferença é feita com os métodos dentro de onResume() que se dá quando a atividade passa a interagir com o utilizador. Pode-se ver que é criado um temporizador que esperará pelo fim do intervalo, calculado através do intervalo de aquisição, para agrupar uma nova vaga de pontos que será transportada para o gráfico.

5.3.3. CONCRETIZAÇÃO DA TRANSFORMADA DE FOURIER

A transformada de Fourier, outro objetivo pilar, foi conseguida também recorrendo a uma biblioteca externa, a LibGDX [47]. Esta biblioteca é utilizada para a criação de jogos em múltiplas plataformas e dispõe de muitas ferramentas, contudo, para este caso, interessa o facto de fornecer o algoritmo da transformada rápida de Fourier. Novamente recorrendo aos exemplos fornecidos desta biblioteca *open-source* criou-se a possibilidade de haver a análise em frequência.

O uso é muito simples tratando-se de chamar a classe FFT(n, fs) onde n é o número de pontos e fs a frequência de amostragem. Dentro desta classe existem ferramentas para obter a transformada - fft.forward() - a inversa - fft.inverse() – ou obter só a parte real ou imaginária da resultante.

5.3.4. INTRODUÇÃO DAS OPÇÕES E NORMAS

No intuito de dar mais valor ao programa decidiu-se adicionar a possibilidade de realizar monitorizações, seguindo normas ou personalizáveis, e de definir alertas. Nesta fase inicial de desenvolvimento, designada de alfa, estão programadas a norma internacional ISO 10816 e a NP 2074.

Os alertas, visuais e sonoros, foram feitos recorrendo somente à biblioteca nativa do sistema operativo Android. De salientar que a extensiva lista de definições é onde será feito o controlo das características dos alarmes.

5.4 PROGRAMAÇÃO DOS MÉTODOS

De forma a se obter uma boa experiência e uma aplicação de confiança foi preciso coordenar os métodos programáticos com as características da aquisição de dados.

A atividade de apresentação, já anteriormente descrita, é importante para se dar a conhecer a frequência máxima, tempo de leitura e intervalo de aquisição de pontos que são necessárias para que o utilizador faça uma análise dos dados adquiridos bem informada.

As características advêm da escolha na janela das definições. Os valores presentes para escolha foram pré-calculados de modo a que se consiga que o número de pontos seja uma potência de dois, na Tabela 5.3, como já justificado anteriormente, equação (5.2, satisfazendo a exigência da transformada rápida de Fourier, tal como, o intervalo de tempo entre aquisição de pontos para a placa de aquisição, equação (5.1.

$$F_{máx} = 1/2.\Delta t \tag{5.1}$$

$$N = F_{máx}.2 * T \tag{5.2}$$

Tabela 5.3 – Números de pontos gerados para cada tempo de leitura escolhido

Intervalo de Aquisição (Δt)						
		0,001	**0,002**	**0,004**	**0,008**	**0,016**
Tempo de Leitura (T)	**1**	1024	512	256	128	64
	2	2048	1024	512	256	128
	4	4096	2048	1024	512	256
	8,2	8192	4096	2048	1024	512
	16,4	16384	8192	4096	2048	1024

De notar que a cada ciclo é gravado um novo valor de aceleração, que depois são guardados num vector de dimensão *n*. Estas informações serão levadas para a atividade de grafismo, quer estática quer dinâmica, pelo mesmo veículo que na atividade principal, proveniente das definições.

A próxima etapa é perceber qual o eixo escolhido pelo utilizador e também a dimensão da leitura. Esta é feita *à priori* da aquisição de dados onde se fará a passagem de g's para unidades métricas de aceleração e seus múltiplos para numa fase seguinte ser feita a transformação na dimensão de velocidade e deslocamento.

5.4.1. AQUISIÇÃO DE PONTOS

A variável axesID e dimension são responsáveis por essa função e irão desplotar as ferramentas necessárias para as concretizar. A escolha do eixo é, simplesmente, transportar a variável das definições, que carrega o eixo pedido, e associar o valor proveniente da placa a essa escolha. A escolha da dimensão é já um processo mais complicado.

Em unidades de velocidade e deslocamento criou-se a ferramenta FFTutils.*fftV* e FFTutils.*fftD*. Serão responsáveis por transformar a aceleração em velocidade e deslocamento respectivamente.

Seguindo as equações regentes de um sinal harmónico em cada dimensão pode-se transformar um sinal em aceleração para velocidade pela sua integração. Contudo este método, em termos programáticos, seria muito difícil de implementar e implicariam erros que teriam de ser debelados. Optou-se por fazer esta transformação em frequência.

Com a ferramenta fft.getImaginaryPart() e fft.getRealPart consegue-se partir a transformada em valores reais e imaginários, em aceleração que, aplicando a equação (5.3) se tornam em unidades de velocidade e a equação (5.4) em unidades de deslocamento. Depois é necessário efetuar a correção da fase.

$$\dot{X} = \frac{\ddot{X}}{2\pi . f} \tag{5.3}$$

$$X = \frac{\ddot{X}}{(2\pi . f)^2} \tag{5.4}$$

Da equação (1.1) para (**1.2**) sabe-se que a fase roda 90º o que, no plano de Argan, significa multiplicar por j cada parcela do resultado da transformada de Fourier. Esta operação irá implicar as seguintes mudanças:

$$(a + bj).j = -b + aj \tag{5.5}$$

$$(a + bj).j^2 = -a - bj \tag{5.6}$$

A fase seguinte, após a aquisição de pontos num vector com dimensão *n*, é a aplicação de filtros consoante a escolha previamente feita. Os filtros usados provêm de uma plataforma para ferramentas musicais. A filosofia segue aquela descrita no capítulo anterior.

5.4.2. FILTROS

É chamado o filtro através de FILTERutils.Filter(**fc**,**fa**,**"filterType"**,r,*dataArray*) onde **fc** representa a frequência de corte, **fa** a frequência de amostragem, **filterType** indica o tipo de gráfico, passa-baixo ou passa-alto, r representa o parâmetro de ressonância e a última secção indica o vector alvo.

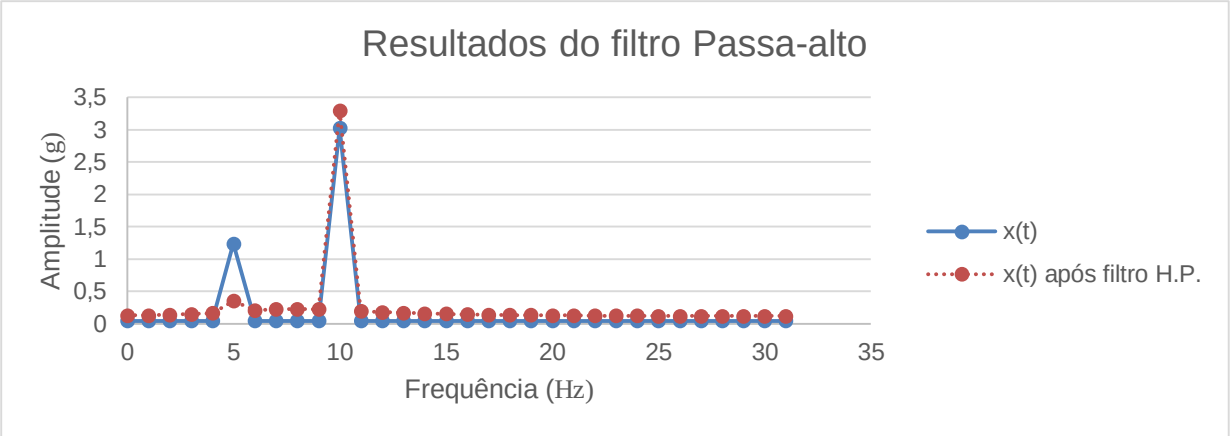

Gráfico 5.1 – Representação gráfica do efeito do filtro passa-alto

Seguidamente surge a análise em frequência evocada pelo método *rundata()* que irá efetuar a transformada de Fourier pura e simplesmente com o intuito de se obter a informação no domínio da frequência. Os resultados desta transformada, representados Gráfico 5.2, foram comparados com as do programa de cálculo Excel e considerados muito satisfatórios, descritos no Anexo A.

5.4.3. TRANSFORMADA DE FOURIER

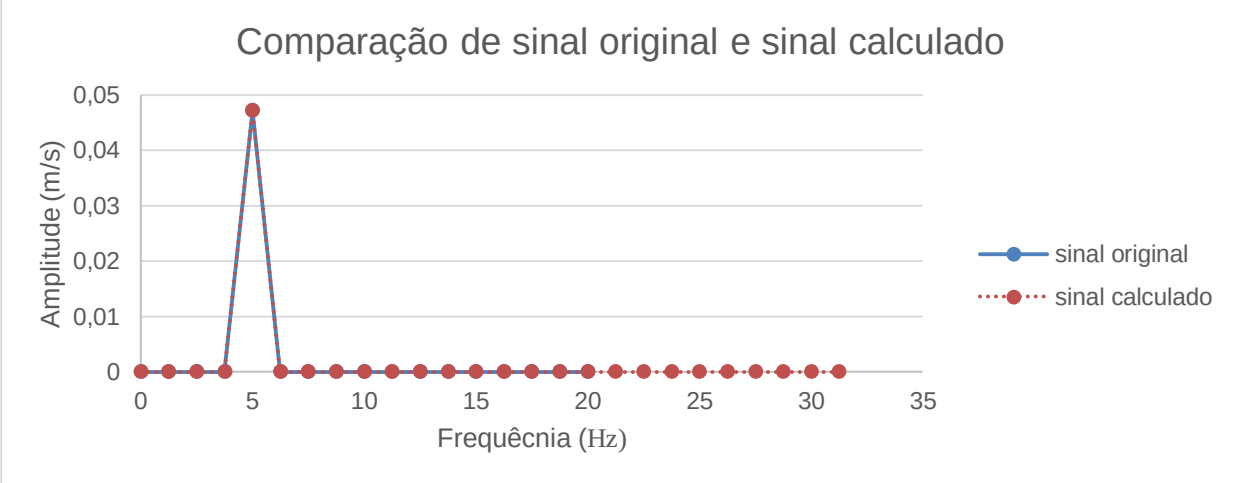

Gráfico 5.2 – Representação gráfica da comparação de um sinal em velocidade e a sua contraparte calculada a partir de um sinal em aceleração

A transformação em frequência é feito pela chamada de FFTutils.*fft*(**N**,**Fa**,*dataArray*), como na Figura 5.16, com N a significar o número de pontos do *array*. De realçar o número de pontos devolvido respeitar a premissa anterior, do capítulo 3.1 (pág. 21): res=n/2.56.

```
FFT fft = new FFT(n, fs);
fft.forward(new_array);
fft_cpx = fft.getSpectrum();
tmpi = fft.getImaginaryPart();
tmpr = fft.getRealPart();

for (int i = 0; i < res.length; i++) {
    real[i] = (double) tmpr[i];
    imag[i] = (double) tmpi[i];
    mag[i] = Math.sqrt((real[i] * real[i]) + (imag[i] * imag[i]));
    res[i] = (float) (mag[i]*2/n);
}
```

Figura 5.16 – Formulação do cálculo para a transformação de um sinal no domínio do tempo em domínio da frequência

Existe a particularidade de a aplicação fazer uso, ou não, das correções por janela, para debelar erros, já aqui referidos, consoante a escolha do utilizador. Em caso afirmativo o sistema procura saber qual a janela escolhida seguindo a ordem descrita na apresentação da atividade de definições.

5.4.4. MONITORIZAÇÃO

A monitorização desempenha um papel importante na utilidade deste projeto. Dado o seu início o utilizador vai para uma página onde lhe é dado a escolher qual a monitorização quer seguir, contudo, o seu papel não se esgota aqui. Também é feita a verificação dos valores de alerta, escolhidos pelo utilizador, para ver se cumprem os tramites da norma. Em seguida passa-se para a monitorização propriamente dita.

Esta enclausura todas as ferramentas necessárias para a concretização dos pressupostos da norma. É chamada por STANDARDSutils.*npStandard*(**N**, **fa**, **T**, *dataArray1*, *dataArray2*, *dataArray3*), no caso da norma portuguesa e STANDARDSutils.*isoStandard*(**N**, **fa**, **T**, *dataArray1*), para a norma internacional. Em ambos os casos, nas três primeiras posições, transportam-se as características da leitura e nas restantes os vectores necessários para o cálculo dos valores.

Após a chamada do método que irá efetuar os cálculos é feita a comparação dos valores na quarta atividade da aplicação, tanto para a norma internacional como para a norma portuguesa.

Os valores, devolvidos anteriormente, serão comparados com os valores quer tabelados quer definidos pelo utilizador, e depois desencadearam uma ação correspondente com o grau de alerta e severidade.

Dada a necessidade de se obter dados dos três eixos de leitura em simultâneo é necessário transportar três vectores para *npStandard* onde se irá efetuar a sua transformação de dimensão. Posteriormente tem-se a sua soma e é retirado o valor máximo de velocidade, como indica a equação (3.5. Seguidamente é preciso perceber qual a frequência dominante.

Na Figura 5.17 representa-se o cálculo feito para se obter este parâmetro. Numa primeira fase é realizada a transformada de Fourier do sinal conjunto e daí se retira qual a posição da frequência com maior amplitude, logo a dominante, e assim sabe-se o seu valor.

```
dataArray1= FFTutils.fftV(N, (float)fs,T, array1);
dataArray2= FFTutils.fftV(N, (float)fs,T, array2);
dataArray3= FFTutils.fftV(N, (float)fs,T, array3);
int maxIndex=0;
for (int i = 0; i < dataArray1.length; i++){
   dataArray[i] =dataArray1[i]*dataArray1[i] + dataArray2[i]*dataArray2[i] + dataArray3[i]*dataArray3[i];
   Vmax[i] =Math.abs((float)(Math.sqrt((double)dataArray[i])));
   if (Vmax[i] > Vmax[maxIndex]) {
      maxIndex = i;
   }
}
res[0]=Vmax[maxIndex]; maxIndex=0;
data = FFTutils.fft(N, (float)fs, Vmax);
for (int i = 0; i < data.length; i++){
   if (data[i] > data[maxIndex]) {
      maxIndex = i;
   }
}
float fD = maxIndex*(float)(1/T);res[1] = fD;
```

Figura 5.17 – Representação do cáculo para a normalização NP 2074 *in npStandard*

Na secção da norma portuguesa começa-se por transformar os valores de aceleração em velocidade, nas unidades de mm/s, para em seguida elevar à potência de dois, somar e realizar a raiz quadrada. É devolvido um vector com duas posições, com a frequência dominante e com a velocidade máxima. Em isoStandard efetua-se também a transformação em unidades de velocidade, mm/s, Figura 5.18, e depois aplicam-se os filtros para se calcular o valor global.

```
dataArray3 = FFTutils.fftV(N, (float)fs,T, array);
new FILTERutils.Filter(1000, fa,"high",1,dataArray3);
new FILTERutils.Filter(10, fa,"low",1,dataArray3);

for (int i = 0; i < dataArray3.length; i++) {
   dataArray2[i] = dataArray3[i]*dataArray3[i];
}
float sum =0;
for (int i = 0; i < dataArray2.length; i++){
   dataArray1[i] = dataArray2[i]/ fa;
   sum += dataArray1[i];
}
res[0]=((float)Math.sqrt((double)sum));
```

Figura 5.18 – Representação do cáculo para a normalização ISSO 10816 *in isoStandard*

Obtidos esses valores, já na quarta atividade, é realizada a verdadeira monitorização. Para ambas as normas os valores de velocidade são comparados com os recomendados que, consoante o grau de gravidade, desencadearão uma ação diferente, mas, só e só se, o utilizador assim o desejar. Os alertas podem ser só visuais ou também sonoros e são acionados dentro de runAlarm(*value*), como na Figura 5.19.

```
if (value==0&&NOTIFICATION.equals(true)&&SMS.equals(true)&&noRepeat==0){
   SmsManager sendSMS = SmsManager.getDefault();
   sendSMS.sendTextMessage("contacto",null,"The alarm value has been exceed",
         null,null);
   noRepeat=1;
}else if (value==1&&NOTIFICATION.equals(true)&&SMS.equals(true)&&noRepeat==0){
   SmsManager sendSMS = SmsManager.getDefault();
   sendSMS.sendTextMessage("contacto",null,"The alarm value has been exceed",
         null,null);
   noRepeat=1;    playSound(getApplicationContext(), uri);
}else if (value==1&&NOTIFICATION.equals(true)&&SMS.equals(false)){
   playSound(getApplicationContext(), uri);
}
```

Figura 5.19 – Representação do código inserido em runAlarm

Value definirá a gravidade da ação a executar. Como mostrado na interface desta atividade existe a possibilidade de definir vários alertas. O mais básico, e sempre presente, trata de colocar blocos de imagem consoante a gravidade - ImageView alarmViewR = (ImageView) findViewById(R.id.***image***) -, sendo verde estar tudo bem e vermelho grave. Usando a ISO como exemplo se o valor mais grave for excedido o fundo da aplicação torna-se vermelho, dada a norma indicar que ultrapassado este valor, há grande possibilidade de existir dano, ficando assim até ser reiniciada (background.setBackgroundColor(Color.***RED***)).

No caso do aviso grave poderá soar um alarme - playSound(getApplicationContext(), uri) – ou o sistema poderá enviar uma mensagem de texto - sendSMS.sendTextMessage(**"número de contacto"**,**null**,**"The alarm value has been exceeded"**, **null**,**null**), explicado Figura 5.19. Dado o facto de haver outras hipóteses de comunicação neste S.O. os alarmes poderão ser modificados.

5.5. TESTES EFETUADOS

Para cimentar todos os parâmetros da aplicação foram efetuados alguns testes:

1. Agitar o acelerómetro
2. Medição de impacto
3. Avaliação das frequências de rotação de um ventilador
4. Determinar as frequências pressentes numa onda sonora
5. Verificação da possibilidade e haver dano numa estrutura segundo a norma portuguesa NP 2074

O primeiro teste serviu para testar o comportamento básico da aplicação e perceber o que tinha de ser feito para corrigir pequenas anomalias que qualquer nova aplicação apresenta. Para isso agitou-se o acelerómetro algumas vezes e registou-se o seguinte gráfico, representado na Figura 5.20.

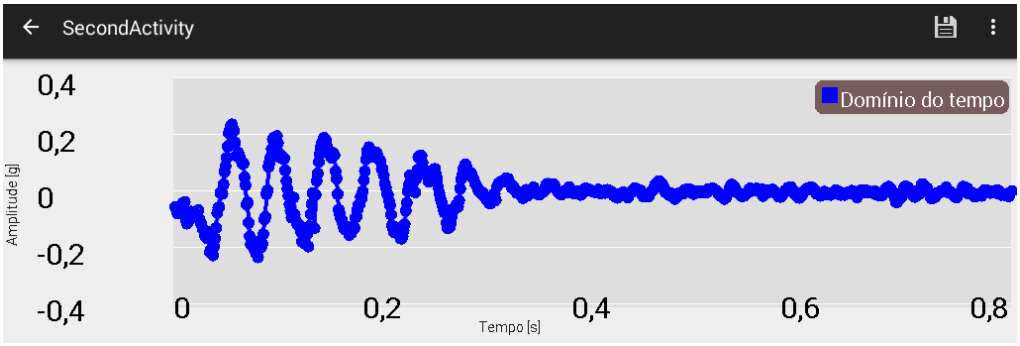

Figura 5.20 – Representação dos resultados do teste básico

Da análise da figura é possível observar-se que o gráfico obtido representa as oscilações a que o sensor foi sujeito. Deste teste verificou-se que a interface cumpre os seus objetivos mostrando ambos os gráficos de forma nítida e que as legendas são legíveis.

No segundo teste, o de impacto, foi realizado com o intuito de testar o comportamento da escala automática do gráfico. Na Figura 5.21 observa-se que a ferramenta GraphView, referida no subcapítulo 5.3.1, cumpre as premissas expostas. A escala adaptou-se, em ambas as figuras, ao género de excitação, mantendo ao centro as linhas, apresentando cada ponto individualmente e com a opção de o utilizador poder saber qual o ponto adquirido. Contudo denotou-se alguma dificuldade em fazer uma aproximação da imagem (*zoom*).

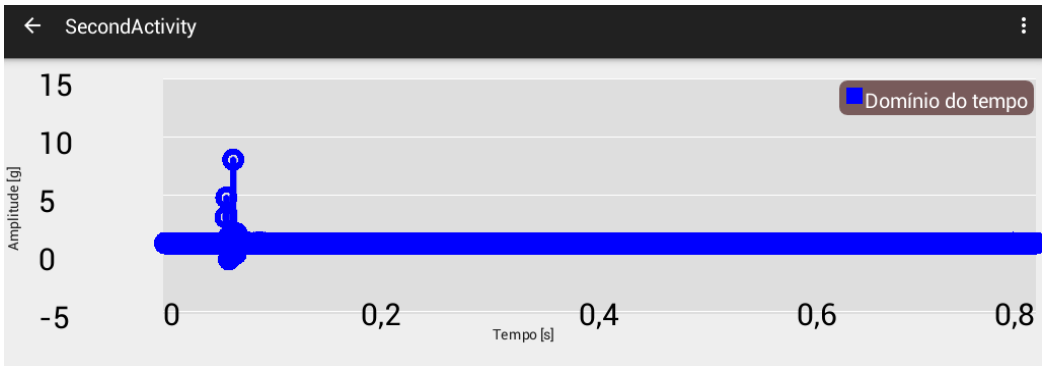

Figura 5.21 - Representação do resultado do teste de impacto

5.5.1. TESTE DE MONITORIZAÇÃO DE VELOCIDADE

Para o terceiro teste já se quis comparar a leitura efetuada pela aplicação com um outro software criado em Labview. Este último já foi verificado e servirá para comparar e acreditar as leituras do primeiro. Usou-se um ventilador comum com dois níveis de velocidade para este teste de modo a testar a análise em frequência.

Para se garantir uma boa comparação, ambas as aplicações foram usadas com as mesmas características de leitura, representados na Tabela 5.4.

Tabela 5.4 – Caracteísticas de leitura de ambas as aplicações

Características de Leitura	
Frequência Máxima	500 Hz
Tempo de Leitura	0,8 s
Intervalo de Aquisição	1 ms
Eixo de Leitura	Eixo X

Na Figura 5.22 apresenta-se a representação gráfica da excitação provocada por um ventilador na sua velocidade mais baixa enquanto que, na Figura 5.23, apresenta-se para a sua velocidade mais alta.

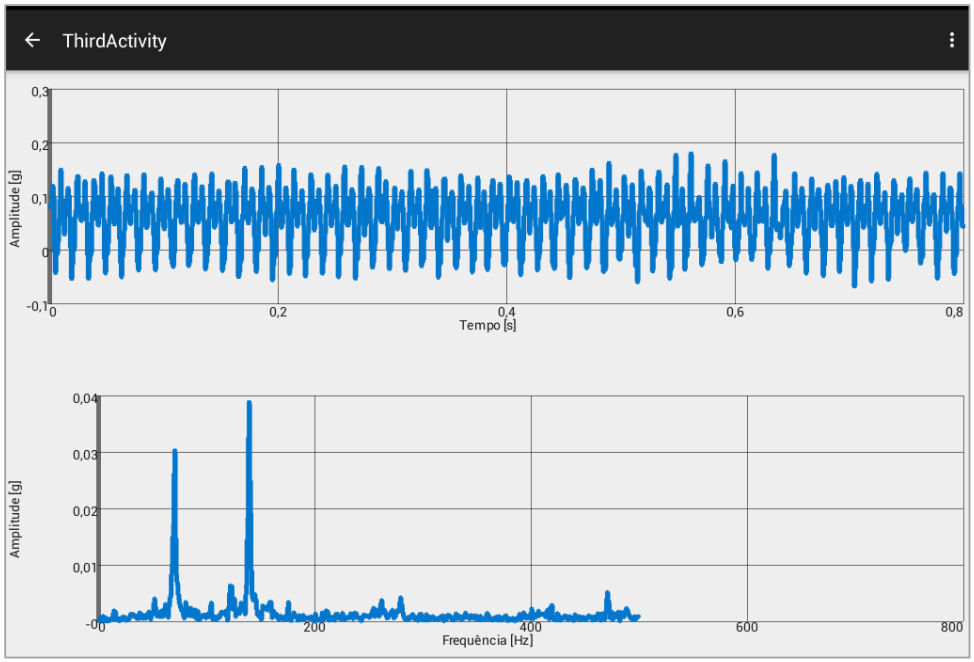

Figura 5.22 – Representação dos dados adquiridos, em modo contínuo, do ventilador na sua velocidade mais baixa

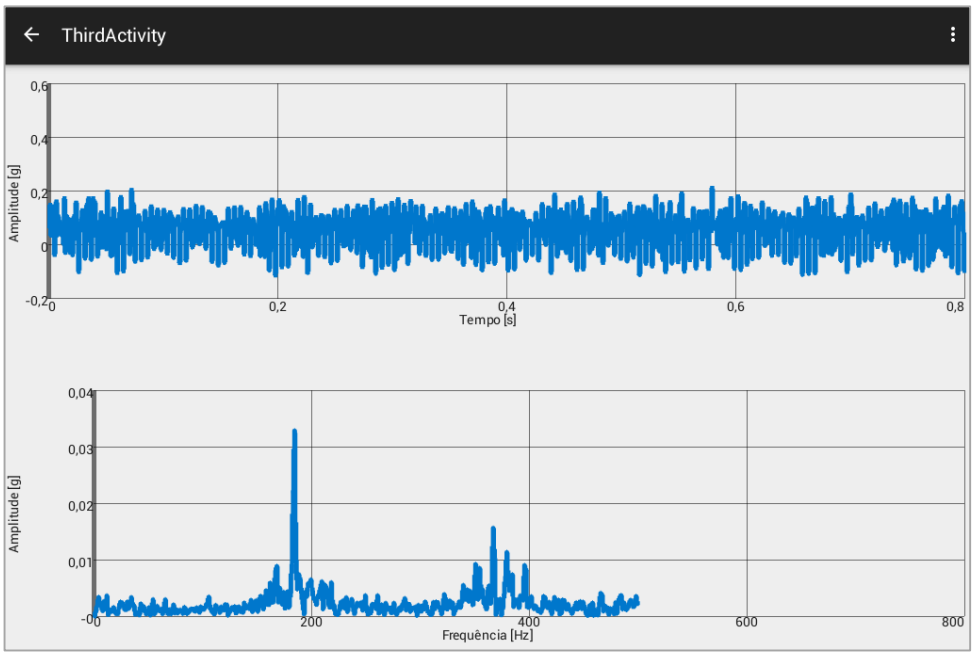

Figura 5.23 - Representação dos dados adquiridos, em modo contínuo, do ventilador na sua velocidade mais alta

Pela comparação de ambas as figuras observa-se a variação das frequências registadas em que, como seria de esperar, a mais alta regista-se para a velocidade mais alta do ventilador. Contudo, ao comparar com a Figura 5.24, retirada do Labview verifica-se que a leitura da aplicação é deficiente para as baixas frequências. Enquanto a aplicação só apresenta um pico para os 180 Hz com a uma amplitude de, aproximadamente, 0,03 g o Labview apresenta 0,008.

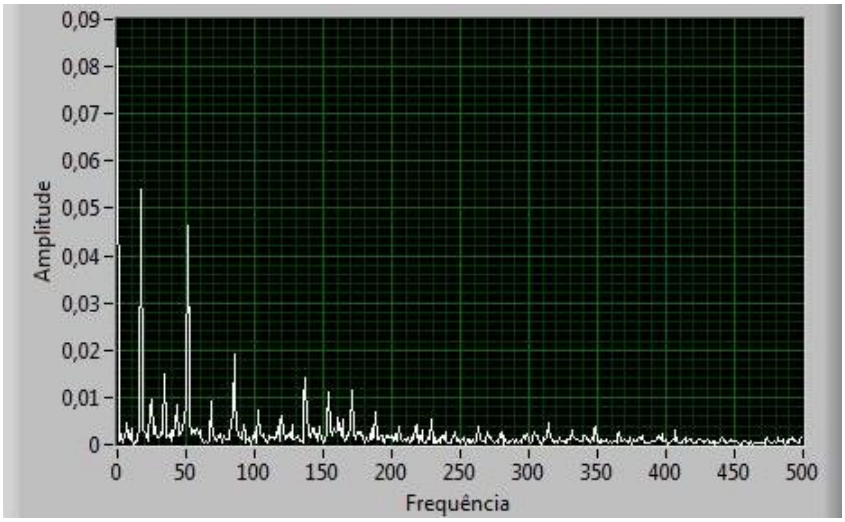

Figura 5.24 - Representação gráfica do espectro da excitação causada pela velocidade máxima do ventilador, usando o Labview

Tal facto revela que a aplicação necessitará de trabalho extra para debelar esta falha. Visto se ter comprovado que os métodos de processamento de sinal estão bem aplicados, como prova o Anexo A, o problema deverá estar na aquisição de dados. Verificou-se que a plataforma Android, para intervalos de aquisição baixos (inferiores a 16 ms), não consegue acompanhar a placa de aquisição distorcendo a posição temporal do ponto.

5.5.2. TESTE DE FREQUÊNCIA ADQUIRIDA

Fazendo uso de uma ferramenta acessível como a placa de som de um computador e de um emissor tentou-se criar uma onda sonora com diferentes frequências. Recorreu-se a um ficheiro de som e a um *subwoofer* de pequenas dimensões para este teste, como descrito na Figura 5.25.

Figura 5.25 – Representação da montagem de experiênica, usando ondas sonoras

A onda criada apresenta as seguintes frequências no espectro na Figura 5.26. Esta análise foi feita usando o Labview e o mesmo sensor, em condições passíveis de serem repetidas na aplicação com os dados da Tabela 5.4. O som emitido apresenta duas frequências distintas a 50 e 100 Hz.

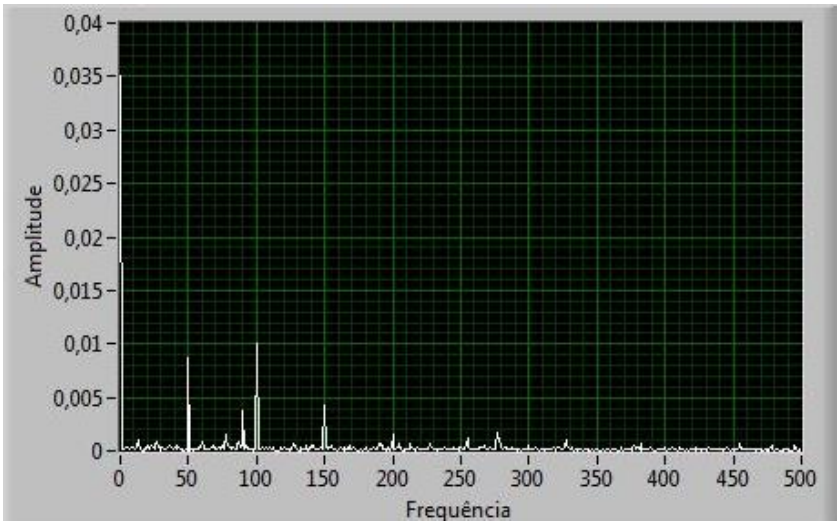

Figura 5.26 - Representação gráfica do espectro do som criado usando o Labview

Usando a aplicação, no modo de medição única, replicou-se o mesmo teste. Os resultados, apresentados na Figura 5.27, são pouco satisfatórios uma vez que, as frequências supramencionadas, não são detectadas. Ampliando a imagem e carregando no ponto de maior amplitude, o pico, tem-se um valor de 0,004, aproximadamente, para 255 Hz.

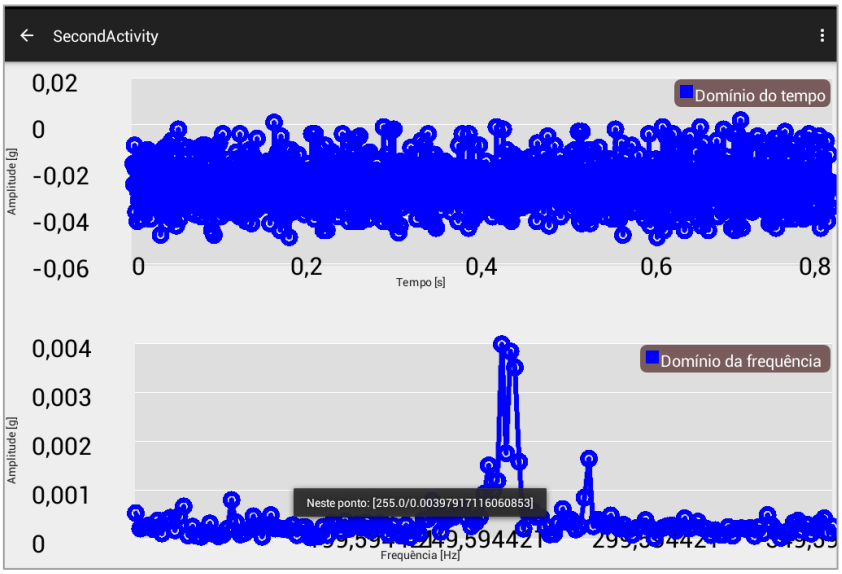

Figura 5.27 - Representação gráfica do espectro do som criado usando a aplicação

Comparando com ambas as aquisições denota-se que a aplicação não conseguiu obter as frequências que se pretendia, de 50 e 100Hz com, aproximadamente, 0,01 g de amplitude.

5.5.3. TESTE DA MONITORIZAÇÃO DA NORMA NP 2074

Para testar a atividade de monitorização aproveitou-se o facto de haver obras na proximidade de um edifício. Assim montou-se o sensor na parede, como indica a Figura 5.28, e fez-se uso da norma portuguesa. Tendo em conta as características da estrutura considerou-se ser uma estrutura corrente, parâmetro a ser escolhido nas definições, visto tratar-se de um edifico de habitação pouco esbelto.

Figura 5.28 – Representação da montagem da experiência, norma NP 2074

Pela observação da Figura 5.28 e Figura 5.29 vê-se que o valor limite de vibração de pico, 3 mm/s, não foi atingido dada a obtenção de um valor de leitura de 0,96 mm/s. De notar também que o alerta visual manteve-se verde indicando que os limites não foram excedidos.

Figura 5.29 – Resultado obtido na experiência

6. CONCLUSÃO

Neste projeto abordou-se as plataformas de fonte aberta com o intuito de demonstrar como a sua utilização é possível como também benéfica para o controlo de condição. Focou-se no controlo por vibrações por ser este o que mais consenso reúne.

Foi abordada a parte teórica do controlo de condição por vibrações que foi fundamental para o correto desenvolvimento de todos os métodos da aplicação Android aqui exposta. Falou-se de algumas facetas do sistema operativo e do seu desenvolvimento, nesta que será, porventura, a maior plataforma de fonte aberta.

Um analisador de vibrações foi projetado, na plataforma Android, capaz de adquirir dados em vários intervalos de tempo ou em contínuo, numa gama de frequências e com a possibilidade de se criar alertas consoante determinadas normas ou personalizados pelo utilizador.

Adquiriu-se uma importante experiência com sensores do tipo MEMS neste projeto, muito atual, ao nível das tecnologias usadas. Tantos os sensores referidos como a plataforma Android chegarão a cada vez mais utilizadores onde desempenharão funções mais relevantes.

6.1. IDEIAS A RETIRAR

O desenvolvimento deste projeto incorreu em algumas dificuldades dada a inexperiência em programação, especialmente numa linguagem de alto nível como o Java, e também por se tratar da comunicação entre a placa de aquisição e um dispositivo móvel. Foram ultrapassadas muitas vezes com o auxílio de fóruns de comunidades de programação disponíveis na internet.

Provou-se também que, mesmo sem conhecimentos profundos de programação, é possível concretizar projetos em plataformas de fonte aberta que sejam úteis no controlo de condição e que auxiliem a função manutenção sempre com baixo custo de implementação.

Infelizmente a aplicação ainda não garante resultados satisfatórios não sendo, assim, um viável substituto ao dispendioso analisador comercial. Contudo, não obstante deste facto, fica o importante desenvolvimento em duas tecnologias de futuro que são as plataformas de fonte aberta e os sensores do tipo MEMS como também o maior conhecimento do processamento de sinal e ferramentas matemáticas que o compõem.

6.1.1. PLATAFORMAS DE FONTE ABERTA

A prova maior da capacidade das plataformas de fonte aberta está num mercado cada vez maior onde as companhias investem mais e apostam. A nova geração do Rasperry Pi, por exemplo, terá disponível o novo sistema operativo da Windows, construído especialmente para a internet das coisas, do inglês *internet of things* (IOT).

Esta aproximação de uma grande empresa levará a que também os fabricantes de acessórios ganhem mais interesse levando assim a uma maior panóplia de periféricos. Também é de denotar que sensores de ultrassom e de infravermelhos já são comercializados para dispositivos móveis com o Android e o iOS.

Quer seja, para realizar um simples protótipo quer seja para uso concreto, estas plataformas serão sempre a abordagem de menor custo e com maior liberdade e adaptabilidade que auxiliará a função de manutenção nas suas tarefas.

6.1.2. ANDROID E PLATAFORMA DE DESENVOLVIMENTO

A plataforma Android cresceu muito e agora está disponível noutros dispositivos o que expande também o leque de possíveis projetos. A linguagem de programação será sempre a maior barreira para a realização dos projetos, mas, cada vez mais, caminha-se para uma maior simplificação.

Existem projetos que pretendem tornar a programação mais acessível a todos. De destacar o App Inventor desenvolvido no Instituto Tecnológico de Massachusetts. Nesta plataforma o desenvolvimento é feito por meio de peças de puzzle em que as ligações vão formando o código de programação, contudo tem algumas limitações. Neste projeto, por exemplo, não se conseguiria introduzir a informação para tornar possível a comunicação entre a placa de aquisição e dispositivo móvel. Não obstante deste facto já é possível criar aplicações com algum grau de complexidade e também a posterior conversão em linguagem escrita para ser usada no Android Studio.

O Android Studio é uma ferramenta de desenvolvimento muito acessível que oferece muitas dicas e auxilio ao utilizador. A informação disponível nos mais diversos canais digitais é muito vasta tal como é a oferta de livros sobre a criação de aplicações para dispositivos móveis.

6.1.3. POSSÍVEIS APLICAÇÕES

A substituição do analisador de vibração comercial por esta aplicação, após validação, é uma das principais aplicações sendo que é sem dúvida muito mais acessível, de fácil adaptação e capaz. Contudo, na mesma plataforma e com o mesmo periférico, é possível projetar outras soluções.

A empresa Phidgets oferece outros produtos com ligação USB como sensores de temperatura. Isto vem permitir que, a mesma plataforma, seja um analisador de vibrações e também um analisador de temperatura. A combinação de ambos ou uso de mais do que um sensor no mesmo dispositivo também é possível.

6.2. DESENVOLVIMENTOS FUTUROS

Para se aumentar o espectro de utilidade da aplicação, no futuro, prevê-se o desenvolvimento de um painel informativo, que receberá toda a informação necessária dos possíveis vários dispositivos na unidade industrial, recorrendo a *software* de fonte aberta, como o My Open Lab, que funcione através dos serviços de nuvem de dados. Outra etapa a desenvolver seria a criação de alertas para os novos dispositivos denominados de *smartwatches.*

O primeiro permitirá, por exemplo, recorrendo a um microcomputador, criar uma interface que dê ao utilizador, num só local, toda a informação dos vários dispositivos consoante as definições de cada um. A utilização de um relógio inteligente permitiria que, o colaborador responsável pela monitorização do controlo de condição, tenha no seu pulso toda a informação sobre o estado dos vários equipamentos e também receber alertas em tempo real.

REFERÊNCIAS

[1] R. B. Randall, Vibration-based Condition Monitiring, Ilinois, E.U.A.: Wiley, 210.

[2] IPQ, "Terminologia de Manutenção," Instituto Português da Qualidade, Lisboa, 2007.

[3] J. K. Sinha e K. Elbhbah, "A future possibility of vibration based condition monitoring," *Mechanical Systems and Signal Processing*, vol. 34, pp. 231-240, 2013.

[4] F. Magalhães, A. Cunha e E. Caetano, "Vibration based structural health monitoring of an arch bridge: From automated OMA to damage detection," *Mechanical Systems and Signal Processing*, pp. 212-228, 2012.

[5] "Discover an Open Source World," Red Hat, 2015. [Online]. Available: http://opensource.com/resources/what-open-source. [Acesso em Maio 2015].

[6] D. Kotsakos, P. Sakkos, V. Kalogeraki e D. Gunopulos, "SmartMonitor: using smart devices to perform structural health monitoring," *Proceedings of the VLDB Endowment*, pp. 1282-1285, Agosto 2013.

[7] N. Parker, "Earthquake Sensors 101," WeatherBug, Fevereiro 2010. [Online]. Available: http://backyard.weatherbug.com/group/earthquakes/forum/topics/earthquake-sensors-101. [Acesso em Setembro 2015].

[8] E. S. Cochran, J. F. Lawewnce, C. Christensen e R. S. Jakka, "The Quake-Catcher Network: Citizen Science Expanding Seismic Horizons," Seismological Research Letters, California, 2009.

[9] C. De Dominicis, A. F. Depari e E. Sisinmi, "Performance assessment of vibration sensing using smartdevices," em *Instrumentation and Measurement Technology Conference*, Montevideu, 2014.

[10] T. Persson, "Handheld Vibration Logger for Android Platforms," Luleå, 2012.

[11] N. Jalili, Piezoelectric-Based Vibration Control, Nova Iorque: Springer, 2010.

[12] SKF, “Sistemas para a indústria de mineração/processamento de minério,” SKF, Julho 2015. [Online]. Available: http://www.skf.com/pt/products/condition-monitoring/surveillance-systems/transmitter-based-systems/systems-for-the-mining-mineral-industry/index.html?switch=y. [Acesso em 2015].

[13] N. Maluf e K. Williams, An Introdustion to Microelectromechanical Systems Engineering, Londres: Artech House, 2004.

[14] A. Albarbar, A. Badri, J. K. Sinha e A. Starr, “Performance evaluation of MEMS accelerometers,” *Measurement,* vol. 42, pp. 790-795, 2009.

[15] S. Kavitha, R. Joseph Daniel e K. Sumanagala, “High performance MEMS accelerometers for concrete SHM applications and comparison with COTS accelerometers,” *Mechanical Systems and Signal Processing,* Vols. %1 de %266-67, pp. 410-424, 2014.

[16] R. Guidorzi, R. Diversi, L. Vincenzi, C. Mazzotti e V. Simioli, “Structural monitoring of a tower by means of MEMS-based sensing and enhanced autoregressive models,” *European Journal of Control,* vol. 20, pp. 4-13, 2014.

[17] O. Esu, J. A. Flint e S. J. Warson, “Condition monitoring of wind turbine blades using accelerometers,” Vienna, 2013.

[18] S. Mariani, A. Corigliano, F. Caimmi e M. Bruggi, “MEMS-based surface mounted health monitoring system for composite laminates,” *Microelectronics Journal,* vol. 44, pp. 598-605, 2013.

[19] P. Inc., “Programming Resources,” Phidgets Inc., [Online]. Available: http://www.phidgets.com/docs/Programming_Resources. [Acesso em Junho 2015].

[20] “Sobre o sensor: 1041_0 - PhidgetSpatial 0/0/3 Basic,” Phigets, Inc., 2012. [Online]. Available: http://www.phidgets.com/products.php?category=5&product_id=1041_0. [Acesso em Outubro 2014].

[21] P. Inc., “Accelerometer Primer,” Novembro 2014. [Online]. Available: http://www.phidgets.com/docs/Accelerometer_Primer. [Acesso em Junho 2015].

[22] “Arduino,” Arduino LLC, 2015. [Online]. Available: https://www.arduino.cc/. [Acesso em Fevereiro 2015].

[23] “Rasperry Pi Foundation,” Rasperry Pi Foundation, 2015. [Online]. Available: https://www.raspberrypi.org/help/what-is-a-raspberry-pi/. [Acesso em Fevereiro 2015].

[24] Intel, “Intel Maker - Galileo,” 2015. [Online]. Available: http://www.intel.com.br/content/www/br/pt/do-it-yourself. [Acesso em Fevereiro 2015].

[25] Intel, “Intel Maker - Edison,” 2015. [Online]. Available: http://www.intel.com.br/content/www/br/pt/do-it-yourself/edison.html. [Acesso em Fevereiro 2015].

[26] C. Yackulic, “Infographic: The Complete History of Android,” 15 Junho 2015. [Online]. Available: http://www.androidheadlines.com/2015/06/infographic-the-complete-history-of-android-cupcake-to-android-m.html. [Acesso em Julho 2015].

[27] F. Tronics, “Instructables,” Autodesk, 2015. [Online]. Available: http://www.instructables.com/id/Smart-Thermostat/. [Acesso em Julho 2015].

[28] P. S., “Instructables,” Autodesk, 2015. [Online]. Available: http://www.instructables.com/id/Arduino-Thermostat-Mechanical/. [Acesso em Junho 2015].

[29] SKF, “The SKF Microlog series catalogue,” Grupo SKF, [Online]. Available: http://www.skf.com/binary/149-144316/CM-P1-14285-1-EN-SKF-Microlog-Product-Catalog.pdf. [Acesso em Fevereiro 2014].

[30] R. N. Bracewell, The Fourier Transform and Its Applications, Singapore: McGraw-Hill, 2010.

[31] P. L. Gatti e V. Ferrari, Applied Structural and Mechanical Vibrations, New York: Taylor & Francis, 2003.

[32] D. H. Shreve, “Signal Processing for Effective Vibration Analysis,” 1995.

[33] C. Sampaio, “Introdução às vibrações,” U.C. de Vibrações Mecânicas, 2015.

[34] I. Standards, “Mechanical ibration - Evaluation of machine vibration by measurements on non-roatating parts,” ISO, Geneva, 2009.

[35] I. P. d. Qualidade, “Avaliação da influência de vibrações impulsivas em estruturas,” IPQ, Lisboa, 2014.

[36] Google, “Developers, Design,” Novembro 2014. [Online]. Available: https://developer.android.com/design/index.html.

[37] Google, “Sobre estatísticas: Dashboards,” Google, Inc., 2014. [Online]. Available: https://developer.android.com/about/dashboards/index.html?utm_source=suzunone. [Acesso em Dezembro 2014].

[38] S. Dossier, “Sobre: Statistics and facts about Android,” Statista, 2013. [Online]. Available: http://www.statista.com/topics/876/android/. [Acesso em Dezembro 2014].

[39] R. Queirós, Introdução ao Desenvolvimento de Aplicações, Venda do Pinheiro: FCA - Editora de Informática, 2013.

[40] gsmarena.com, “Sobre o Sony Sola,” maxcdn, 2015. [Online]. Available: http://www.gsmarena.com/sony_xperia_sola-4408.php. [Acesso em Julho 2015].

[41] gsmarena.com, “Sobre o Acer Iconia A1-810,” maxcdn, 2015. [Online]. Available: http://www.gsmarena.com/acer_iconia_tab_a1_810-5399.php. [Acesso em Julho 2015].

[42] R. Queirós, Desenvolvimento de Aplicações Profissionais em Android, Vendo do Pinheiro: FCA - Editora de Informática, 2014.

[43] I. Phidgets, “1041_0 - PhidgetSpatial 0/0/3 Basic,” Novembro 2014. [Online]. Available: http://www.phidgets.com/products.php?category=5&product_id=1041_0.

[44] I. Phidgets, “OS - Android,” Novembro 2014. [Online]. Available: http://www.phidgets.com/docs/OS_-_Android#Quick_Downloads.

[45] J. Gehring, “Android-GraphView, Summary and Features,” Jonas Gehring, 2015. [Online]. Available: http://www.android-graphview.org/. [Acesso em Fevereiro 2015].

[46] J. Gehring, “Android-GraphView, Documentaion,” Jonas Gehring, 2015. [Online]. Available: http://www.android-graphview.org/documentation. [Acesso em Fevereiro 2015].

[47] M. Zechner, “libGDX, Cross-platform Game Development,” XKlibur, 2013. [Online]. Available: https://libgdx.badlogicgames.com/. [Acesso em Março 2015].

Anexos

Anexo A	Cálculos Auxiliares
Anexo B	Métodos Programáticos