Dário Filipe Romana Pedro

Licenciado em Ciências da Engenharia Eletrotécnica e de Computadores

Posicionamento Cooperativo usando Dispositivos Android

Dissertação para obtenção do Grau de Mestre em

Engenharia Electrotécnica e de Computadores

Orientador: Luís Filipe Lourenço Bernardo, Professor Auxiliar com Agregação, Universidade Nova de Lisboa

Co-orientador: Marko Beko, Professor Auxiliar, Universidade Lusófona

Júri

Presidente: Luís Augusto Bica Gomes de Oliveira, FCT-UNL

Arguente: João Pedro Castilho Pereira Santos Gomes, IST-UL

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2017

Posicionamento Cooperativo usando Dispositivos Android

A todos que me apoiaram...

Agradecimentos

"Talent wins games, but teamwork and intelligence win championships." - Michael Jordan

Individualmente somos capazes de realizar grandes conquistas, mas os maiores sucessos são sempre realizados em união. A grandeza pode ser facilmente alcançada com trabalho de equipa e com o apoio dos que nos rodeiam. Por esse motivo, gostaria de salvar um lugar especial neste trabalho para agradecer às pessoas que estiveram presentes e me apoiaram durante todo esse processo de aprendizagem.

Em primeiro lugar quero agradecer ao meu orientador, Professor Luís Bernardo que despertou o meu interesse na área das Telecomunicações e em Android, desde o 2º ano de Faculdade. Tenho também que lhe agradecer todo apoio, intelecto, paciência e tempo disponibilizado ao longo desde último ano. Mesmo em períodos de picos de trabalho foi incansável, disponibilizando-se sempre para me ajudar. Adicionalmente gostaria de agradecer a todos os docentes da secção de Telecomunicações, por todo o conhecimento transmitido. Além disso, agradeço sinceramente ao Doutor Slavisa Tomic e ao Professor Marko Beko, com quem aprendi muito sobre a otimização não-linear, e com quem desenvolvi alguns algoritmos presentes neste documento.

De seguida, quero agradecer à FCT-UNL e ao DEE pelas condições que me proporcionaram, sendo para mim uma segunda casa que foi determinante para o meu crescimento intelectual e profissional.

Aos meus pais e irmã quero agradecer por me proporcionarem todas as condições para realizar e concluir esta etapa da minha vida. Sempre deram mais do que vos era pedido e são um exemplo para mim. Sei o esforço e dedicação que foram necessários para tal, e por isso estou-vos eternamente grato. Quero também agradecer aos familiares mais próximos, que me ensinaram tanto e tornaram possível ser quem sou hoje. Aos meus amigos de infância, que são praticamente família, tenho à agradecer os bons momentos passados e experiências partilhadas, que considero importantíssimos para finalizar esta etapa da minha vida.

Quero ainda agradecer a todos os meus colegas e amigos da FCT-UNL, que me acompanharam em todo este percurso. Agradeço ao Luís Gonçalves, à Mariana Borges e ao Pedro Prates todo o apoio prestado. Especialmente tenho que dar o meu obrigado ao meu grupo (3D++), sem o qual nunca teria alcançado tanto sucesso: André Cardoso, Diogo Correia, Diogo Matos e João Carvalho. Pelas horas de convívio e de trabalho intenso também. De brincadeiras e de dores de cabeça em estudo continuo. Tiveram a meu ver, o

papel mais importante no meu crescimento e aprendizagem ao longo de todo o curso.

Um agradecimento especial aos meus colegas do laboratório de robótica, que me acolheram no seu espaço de trabalho: André Caves, Carlos Simões, Eduardo Pinto, Francisco Marques, Lino Quaresma e Ricardo Mendonça. Nunca me recusaram a ajuda e estivam sempre dispostos a contribuir sem pedir nada em troca.

Agradeço o apoio financeiro dado pela FCT/MEC através de fundos nacionais e quando aplicável cofinanciado pelo FEDER, no âmbito do Acordo de Parceria PT2020 no âmbito do projeto UID/EEA/50008/2013.

Finalmente, um obrigado muito especial à Rita Pascoal. Por seres a mulher que mais marcou o meu percurso académico, apoiando-me sempre a 100%. Por todo o amor, carinho e preocupação, muito obrigado.

Resumo

Recentemente foram propostas várias soluções de localização *indoor* baseadas em WiFi, Bluetooth e UWB. Os ambientes *indoor* são espaços complexos que apresentam bastante diversidade, permanecendo aberta a solução para conseguir um sistema de posicionamento barato e preciso. Embora algumas destas soluções consigam bons resultados, muitas vezes requerem um trabalho de reconhecimento da localização exaustivo ou hardware especializado.

Nesta dissertação é estudado o posicionamento cooperativo de smartphones Android, explorando os sensores presentes nestes dispositivos e a infra-estrutura sem fios existente, usando a potência do sinal recebido. Inicialmente, os problemas de localização são formulados como a trilateração de um conjunto de medições para estimar a posição relativa dos emissores face ao smartphone.

Para a realização de testes, foi desenvolvida uma aplicação que implementa a odometria do dispositivo usando o acelerómetro, giroscópio, vetor de rotação entre outros. Tendo uma localização relativa à posição no momento em que se ligou a aplicação (0,0,0), é possível calcular a posição relativa do outro dispositivo. Essas informações são então compartilhadas entre os utilizadores do grupo usando um servidor. No servidor vai ser corrido um algoritmo de localização cooperativa, que permite minimizar o erro da estimação de localização.

A análise teórica e os resultados dos testes realizados com a aplicação demonstram que esta é uma boa abordagem. Tanto quanto se sabe, esta é a primeira implementação que aborda o problema de localização em dispositivos móveis numa perspetiva relativa, sem necessitar de informação *a priori*.

Palavras-chave: Posicionamento, Estimação, Localização, Android, Cooperação

Abstract

Recently, several indoor localization solutions based on WiFi, Bluetooth, and UWB have been proposed. Due to the limitation, complexity and diversity of the indoor environments, the solution to achieve a low-cost and accurate positioning system remains open. Although some of this solutions achieve good results, they often require a lot of data about of the site or specialized hardware.

This dissertation investigates Android's smartphones localization, exploiting the sensor already build in the devices and the wireless infrastructure, using the received signal strength. Initially the localization problems are formulated as the computation of the intersection of a group of sensing points.

An app was developed that implements device odometry using the smartphone accelerometer, gyroscppe and rotation vector, among others. Having the device current position, is possible to calculate the other device's relative position. This information is then shared among users of the group using a server that will be implemented in order to allow the use of a cooperative location algorithm.

Both theoretical analysis and some comparative simulations reveal that the proposed approach has a good estimation. To the best of our knowledge, this is the first implementation that approaches the locatization problem on mobile devices in a relative prespective.

Keywords: Positioning, Estimation, Localization, Android, Cooperative

Índice

Lista de Figuras

Lista de Tabelas

Siglas

A-GPS *Assisted GPS*.

AoA *Angle of Arrival*.

AP *Access Point*.

API *Application Programming Interface*.

APs *Access Points*.

BSSID *Basic Service Set IDentifier*.

CRF *Conditional Random Field*.

FCT Faculdade de Ciências e Tecnologia.

GMM *Gaussian MixModel*.

GN *Gauss-Newton*.

GNSS *Global Navigation Satellite System*.

GPS *Global Positioning System*.

GSM *Global System for Mobile Communications*.

GUI *Graphical User Interface*.

HMM *Hiddden Markov Model*.

IDE *Integrated Development Environment*.

IMU *Inertial Measurement Unit*.

IPS *Indoor Position System*.

JVM *Java Virtual Machine*.

LAN *Local Area Network*.

LBS *Location-Based Service*.

LM *Levenberg-Marquardt*.

LSS *Least Squares Solutions*.

MAC *Medium Access Control*.

MD5 *Message-Digest algorithm 5*.

MEMS *Micro-Electro-Mechanical Systems*.

ML *Maximum Likelihood*.

NFC *Near Field Communication*.

PDAs *Personal Digital Assistants*.

PLE *Path Loss Exponent*.

PLM *Path Loss Model*.

PT *Pedro-Tomic*.

PWF *Peak-based WiFi Fingerprinting*.

RFID *Radio-Frequency IDentification*.

RMSE *Root Mean Square Error*.

RPY *Roll, Pitch e Yaw*.

RSS *Received Signal Strength*.

RSSI *Received Signal Strength Indicator*.

RTLS *Real Time Location System*.

RV *Rotation Vector*.

SC *Step Counter*.

SD *Step Detector*.

SLAM *Simultaneous Localization and Mapping*.

SOCC *Second Order Cone Constraint*.

SOCP *Second Order Cone Programming*.

SPP *Serial Port Profile*.

SSID *Service Set IDentifier*.

TDOA *Time Difference of Arrival*.

ToA *Time of Arrival*.

UAV *Unmanned Autonomous Vehicle*.

UI *User Interface*.

UNL Universidade Nova de Lisboa.

UWB *Ultra-wideband*.

WAN *Wide Area Network*.

WLAN *Wireless LAN*.

WLANs *Wireless LANs*.

WLS *Weighted Iterative Least Squares*.

WPS *WiFi Position System*.

ZCL *ZigBee-based collaborative localization*.

Nomenclatura

Alguns dos símbolos e constantes mais comuns utilizados ao longo da tese estão listados abaixo. Ao longo da tese, letras maiúsculas representam matrizes enquanto letras minúsculas caracterizam escalares.

Constantes Físicas

π	Proporção entre o perímetro de uma circunferência e o seu diâmetro	$3,141\,593$
c	Velocidade da luz num sistema de inércia a vácuo	$2,997\,9 \times 10^8 m/s$
g	Aceleração gravitacional	$9,806\,65 m/s$

Unidades

dBm	Decibel Miliwatt
dB	Decibel
Hz	Hertz
m	Metro
rad	Radiano
s	Segundo

Símbolos Matemáticos

$\mathbf{0}_{m \times n}$	Matriz $m \times n$ com todas as entradas a 0
$\approx$	Aproximadamente igual a
$\mathbf{log}_a(x)$	Logaritmo de base a de x; quando a é omitido, representa um logaritmo natural
$\mathcal{N}\left(\mu, \sigma^2\right)$	Distribuição gaussiana de valor real com valor médio μ e variância σ^2
$\sim$	Distribuído de acordo com
$\widehat{x}$	Valor estimado de x
A^+	Matriz Pseudo-Inversa de Moore-Penrose de A

$\boldsymbol{A^{-1}}$ Matriz inversa de A

$\boldsymbol{A^{T}}$ Matriz transposta de A

$\boldsymbol{A_{ij}}$ O ij-ésimo elemento de A

$\boldsymbol{I_n}$ Matriz de Identidade $n \times n$

$\Sigma^{-\frac{1}{2}}$ Operador inverso da raíz quadrada da matriz

J Jacobiano

Outros Símbolos

$\boldsymbol{\gamma}$ Influência do Meio de Propagação

$\boldsymbol{\lambda}$ Perdas de Propagação

$\boldsymbol{\varepsilon}$ Erro Absoluto

$\boldsymbol{\vartheta}$ Fator de Amortecimento

$\boldsymbol{G_r}$ Ganho da antena receptora

$\boldsymbol{G_t}$ Ganho da antena de transmissão

$\boldsymbol{P_r}$ Potência Recebida

$\boldsymbol{P_t}$ Potência Transmitida

$\boldsymbol{w_i}$ Peso do elemento i

D Vetor de Distâncias

d Distância

f Frequência

N Número de medições

S Coordenadas do Smartphone

t *Target* (Alvo)

Capítulo 1

Introdução

1.1 Contexto e Motivação

O planeta terra é gigante, cheio de locais diferentes, onde é impossível para os mortais conhecê-los todos. A informação do meio em nosso redor e do lugar onde nos encontramos transmite segurança e permite sonhar mais além.

Hoje em dia a tecnologia é um recurso presente em cada minuto da nossa vida. Tudo está cada vez mais interligado. Os Smartphones tornaram-se um objeto indispensável, com muitas funcionalidades usadas para ajudar em inúmeras tarefas diárias. As aplicações que utilizam a localização do utilizador são uma área na moda para muitos programadores, que beneficiando desta informação, conseguem tornar as aplicações muito mais contextuais.

Existem diversas áreas que beneficiam da existência de aplicações de posicionamento *indoor* (interior), como o retalho inteligente, transportes públicos, entre outras. A necessidade de um novo algoritmo *indoor* que funcione autonomamente sem requerer ações externas é clara. Caso se dirija a um centro comercial e corra o Google Maps no exterior, a sua posição é muito precisa, com 1 ou 2 metros de erro. Contudo caso comece a caminhar para dentro do centro comercial, no piso 0, o erro rapidamente dispara para valores de aproximadamente 30 metros. Esta situação é ainda pior nos parques de estacionamento subterrâneos, onde podemos deparar-nos com erros de sensivelmente 100 metros.

O objetivo final é a criação de um sistema que realize a odometria de um utilizador em ambientes *indoor*, que localize em ambientes *outdoor* (exteriores) a partir da localização Android e que permita com a esta informação localizar entidades que transmitam sinal rádio.

1.2 Objetivos

O objetivo deste trabalho é permitir que as pessoas consigam obter melhores informações sobre o seu posicionamento e/ou o de elementos do grupo em que estejam inseridos. Para isso, foi implementada uma aplicação que recolhe dados do meio envolvente, utilizando-os para gerar informações que permitem estimar a localização dos emissores. Com isto em mente, pretende-se alcançar três metas:

- Avaliar algoritmos que permitam a um terminal móvel obter a posição relativa numa zona sem cobertura do serviço de localização.
- Avaliar e propor algoritmos que permitam realizar a localização de *Access Points* (APs) (Pontos de Acesso) remotos.
- Desenvolver um protótipo funcional do sistema.

1.3 Contribuições

A dissertação pretendeu analisar as técnicas de recolha de dados e o seu processamento em tempo real, sendo que todos os objetivos propostos foram alcançados.

Inicialmente foi realizado um levantamento bibliográfico sobre as principais técnicas e tecnologias utilizados em posicionamento. Nesta etapa é realizada uma análise e comparação entre os diversos métodos e sistemas de localização.

Foi desenvolvida uma aplicação[1] pode ser utilizada em diversos dispositivos. Os dispositivos Android modernos incorporam um conjunto vasto de sensores como acelerómetros, bússola, *Inertial Measurement Unit* (IMU) (unidade de medição inercial), recetores rádio diversos, entre outros. Com a utilização desta aplicação, foi estudado o desempenho de cinco métodos inerciais para localização *indoor* em dois terminais móveis.

Posteriormente, foram estudados dez algoritmos de localização e foi desenvolvido um algorimo original (o Pedro-Tomic), sendo que se testou com dados reais recolhidos *indoor* e num cenário veicular.

Por fim, criou-se um algoritmo capaz de juntar as informações de todos os utilizadores, possibilitando o sistema capaz de funcionar num modo cooperativo.

Com base nesta dissertação, foram elaborados dois artigos: o **Localization of static remote devices using smartphones** para a *IEEE VTC 2018 Spring*; e um segundo artigo de revista que se encontra em preparação, e que será entregue antes da discussão ao arguente.

1.4 Estrutura da Dissertação

Esta dissertação está organizado em sete capítulos:

[1] O código da aplicação pode ser encontrado em `https://github.com/dario-pedro/wifi_finder`.

- **Capítulo 1: Introdução** apresenta o trabalho e propõe a abordagem de implementação. As motivações são delineadas e a arquitetura é explicada.

- **Capítulo 2: Trabalho Relacionado** sumariza as tecnologias por trás das diversas técnicas de localização. Para tal, são estudadas as tecnologias sem fios, as métricas que permitem estimar a distância dos emissores de sinal, os métodos que utilizam *fingerprinting* (assinatura rádio) e os sistemas de localização modernos.

- **Capítulo 3: Sistema Desenvolvido** detalha a arquitetura do sistema implementado nesta dissertação. São especificadas algumas precauções relacionadas com processamento do sinal WiFi e dos sensores utilizados. Além disso, é descrito o funcionamento do módulo de Odometria, sendo introduzidos os conceitos necessários de detecção de movimento e orientação, sendo realizados vários testes com diversos tipos de sensores nos quais é feita uma análise crítica aos resultados, de forma a perceber os motivos dos erros associados às estimações.

- **Capítulo 4: Estimação da localização de Emissores** estuda a recepção de sinal WiFi em dispositivos Android e como estimar a distância a partir desta informação. Em seguida, analisa dez métodos tradicionais de localização que utilizam a informação relativa da Odometria. Numa fase posterior, é apresentado um novo algoritmo, o Pedro-Tomic. No final do capítulo, os algoritmos são testados com dados obtidos pela aplicação desenvolvida, sendo os resultados comparados.

- **Capítulo 5: Cooperação entre dispositivos** complementa o capítulo anterior, conseguindo aproveitar os resultados da estimação para a implementação de um esquema cooperativo onde a precisão é melhorada por métodos de cooperação. Foi criado um novo algoritmo cooperativo com base no estudo de otimização convexa, que permite minimizar as diferenças nas medições produzidas pelos diversos tipos de *hardware* (equipamento) presente em diferentes terminais móveis.

- **Capítulo 6: Conclusão e Trabalho Futuro** resume o estudo e o trabalho desenvolvido. Nesta secção são ainda realizados alguns comentários, críticas proposta de planos para a evolução do trabalho desenvolvido.

Capítulo 2

Trabalho Relacionado

A localização é um tema bastante antigo, que no caso dos portugueses começou a demonstrar grande relevância no século XV, com os descobrimentos. Desde o dia que o homem decidiu guiar-se pelas estrelas, até ao *Global Positioning System* (GPS), a evolução foi imensa, sempre com o objetivo de aumentar a precisão do posicionamento.

Um serviço de localização pode oferecer dois tipos tipos distintos de localização:

1. Localização *Outdoor*.
2. Localização *Indoor*.

Habitualmente, quando existe necessidade de obter a localização *outdoor*, é utilizado um serviço de navegação global por satélites, *Global Navigation Satellite System* (GNSS) obtendo-se uma precisão de localização da ordem de grandeza de metros (na versão civil do GPS). Embora este seja o modo preferido de localização *outdoor*, as técnicas baseadas em *GNSS* muitas vezes não funcionam bem em cidades ou com clima desfavorável, como por exemplo, em dias nublados. Quando os sinais de satélite são atrasados devido a múltiplos caminhos ou bloqueados por obstáculos, a qualidade do serviço de localização pode-se deteriorar. Além disso, o *GNSS* pode apresentar problemas em termos de consumo de bateria.

Devido à fraca capacidade de penetração dos sinais *GNSS* no ambiente *indoor*, vários outros tipos de sinais têm sido estudados para realizar o posicionamento. Estes sinais incluem a rede celular, com *Assisted GPS* (A-GPS) [Zan09], WiFi [BP00], Bluetooth [Liu+13; Zha+14], rádio FM [Liu+13; Zha+14], radiofrequência [Kuo+14; YS15], campo magnético [Chu+11; Yan+13], etc.

Vários serviços de posicionamento *indoor* têm sido propostos, tendo atraído muita atenção nos últimos anos devido ao seu impacto social e comercial. Até 2020, o valor de mercado previsto ascende a 10 biliões de dólares [HC16b].

Em contraste com os sinais *GNSS*, os sinais WiFi beneficiam do ecossistema de redes *Wireless LANs* (WLANs) instaladas, que podem ser usadas para localização.

Figura 2.1: Tecnologias utilizadas no posicionamento *indoor* [Sen]

O *Indoor Position System* (IPS) pode ser usado para localizar pessoas ou objetos dentro de edifícios, geralmente com o uso de um dispositivo móvel, como um *smartphone* ou *tablet* (terminais móveis com capacidade para correr aplicações) [Sen]. Algumas técnicas contam com tecnologias como sensores montados na parede ou no teto que trabalham em conjunto na deteção e/ou localização de um utilizador ou objeto, obtendo uma posição com um erro na ordem de grandeza dos metros. Como o GNSS, os sistemas IPS podem detetar a direção na qual o dispositivo se está a movimentar e estimar o caminho percorrido pelo utilizador. Com esta informação é possível ir minimizando o erro à medida que o espaço é percorrido. Na figura 2.1 é possível visualizar uma representação do agrupamento de tecnologias a trabalhar cooperativamente para realizar posicionamento *indoor*.

Tipicamente, as tecnologias IPS, baseiam-se na combinação de resultados de diversos sensores e na comparação dos mesmos usando algoritmos matemáticos que trabalham em segundo plano, determinando o caminho de um utilizador através de um extenso teste de hipóteses.

Na secção abaixo é descrita brevemente uma amostra de algoritmos e estimadores de localização, que estão implementados nas aplicações existentes.

2.1 Tecnologias Wireless

Para uma mais fácil compreensão dos algoritmos de localização *indoor* utilizados nos dias de hoje, é importante compreender as tecnologias sem fios envolvidas nos mesmos. Muitos sistemas usam a infraestrutura sem fios existente na maioria dos edifícios (rede WiFi) para obter ou melhorar a precisão de posicionamento dos *smartphones*.

Nesta secção são descritas as principais tecnologias que são utilizadas cooperativamente nos sistemas descritos em 2.4.

2.1.1 Sistema de posicionamento baseado em WiFi (WPS)

O *WiFi Position System* (WPS) (Sistema de posicionamento WiFi) , utilizado para o posicionamento em relação aos APs sem fios, baseia-se na medição da *Received Signal Strength* (RSS) (Intensidade do Sinal Recebido) e no método de *fingerprinting* [BP00; CK02; YA08]. Os parâmetros mais úteis para identificar um hotspot WiFi ou o *Access Point* (AP) sem fios incluem o *Service Set IDentifier* (SSID) e o endereço *Medium Access Control* (MAC). A precisão da localização de um cliente WiFi depende do número de posições de APs que foram inseridas na base de dados. Estas bases de mapas, ou mapas de localização, têm que ser criados *a priori*.

As flutuações do sinal recebido de cada AP, fruto de efeitos relacionados com várias fontes de desvanecimento e interferência, introduzem erro na estimação e tornam o sistema menos preciso. Desta forma, são necessários métodos de processamento de sinal que minimizem o erro. Como exemplo de um sistema de posicionamento baseado em WiFi temos o Anyplace [Geo+15]. O Anyplace [Geo+15] é um sistema de posicionamento de código aberto que permite a qualquer pessoa rapidamente mapear espaços interiores, tendo ganho vários prémios pela sua precisão de localização [Lao+12; Lym+15].

2.1.2 Bluetooth

De acordo com o Bluetooth Special Interest Group, o Bluetooth é principalmente utilizado para descobrir a proximidade a um alvo, e não para a localização exata do mesmo [RG09]. A tecnologia Bluetooth não foi concebida para descobrir coordenadas como nos GNSS, no entanto é utilizada para ter uma estimativa da distância a que se encontra o objeto. É assim, uma solução de proximidade *indoor*, e não uma solução de posicionamento *indoor*.

O serviço iBeacon, promovido pela Apple Inc., suporta o micro mapeamento e mapeamento *indoor* [Zho+11] utilizando *beacons* Bluetooth [SK10]. Mais recentemente a Google lançou um produto muito semelhante ao iBeacon, o *Eddystone Beacons*.

Existem vários sistemas de posicionamento interior que utilizam este tipo de Beacons [MT13], como por exemplo, os vários projectos da *Insoft* [Men+16], que conseguem realizar posicionamento com erros inferiores a 3m.

2.1.3 Ultra wideband

A *Ultra-wideband* (UWB) é uma tecnologia de rádio que usa um nível de energia muito baixo para um curto alcance, usando sinais que cobrem uma banda muito larga do espectro de rádio. Por essa razão, há uma interferência reduzida com outros dispositivos. Tanto o transmissor como o recetor devem ser sincronizados para enviar e receber pulsos com uma precisão de $1ps$. Em qualquer banda de frequência que possa já estar em utilização,

o sinal de banda ultra larga tem menos potência do que o ruído de fundo normal previsto, pelo que teoricamente não sofre interferência [Dea+12].

Normalmente, as aplicações que usam esta tecnologia são projetadas e testadas usando sinais WiFi e complementados com sinais de banda ultra-larga para uma maior precisão de localização. A maioria das diferenças está a nível do hardware, tornando a transição entre tecnologias relativamente simples. A *decaWave* tem vários produtos baseados nesta tecnologia que permitem ser instalados de formas diversas criando várias topologias, como por exemplo:

- *Secure Bubble* - Utiliza 1 âncora e 1 ou mais *tags* (etiquetas). É uma configuração que permite aos utilizadores monitorizar se as *tags* estão dentro de determinados perímetros pré-definidos em torno da âncora (a bolha) .
- Posicionamento - Utiliza 3 âncoras e 1 *tag*. O utilizador deve manter na sua posse a *tag* e caminhar dentro do alcance das três âncoras para ser possível obter a estimação da sua posição.

2.2 Métricas e Algoritmos

2.2.1 Conceitos de ponto de passagem

O posicionamento por ponto de passagem é um conceito simples de localização por zonas e de relatórios de presença de objetos *tagged* (com etiquetas), utilizando apenas a identificação perante um sensor conhecido [RG09]. Este é geralmente o caso dos sistemas de identificação por radiofrequência passiva *Radio-Frequency IDentification* (RFID), embora também possa ser usado com outras tecnologias. O funcionamento de tais abordagens requer que os dispositivos passem por alguma passagem estreita, que por sua vez está equipada com sensores que detetam os dispositivos. Num sistema passivo RFID, caso a passagem seja larga existe a possibilidade de passar num ponto que se encontra fora do alcance.

2.2.2 Ângulo de chegada

O *Angle of Arrival* (AoA)(ângulo de chegada) é o ângulo a partir do qual um sinal chega a um recetor. O AoA geralmente é determinado medindo a diferença de *Time Difference of Arrival* (TDOA) (tempo de chegada) entre várias antenas numa matriz de antenas. A direção de um sinal de entrada pode ser calculada explorando e detetando a diferença de fase entre antenas [SW10]. O AoA é normalmente utilizado juntamente com triangulação para encontrar a localização relativa a dois *beacons*. Um exemplo de uma solução comercial que usa AoA é o sistema HAIP da Quuppa, onde pode ser alcançada uma precisão de posicionamento de 0,5 a 1 m [Bel+09]. Esta solução requer um dispositivo de hardware específico, que contém 16 antenas dispostas em matriz, e uma *tag* especial que usa sinais Bluetooth.

A estimativa do AoA pode ser obtida usando pelo menos um par de antenas. Quando o número de antenas é superior a dois, o AoA pode ser estimado conjuntamente obtendo-se um melhor desempenho [YS15].

2.2.3 Tempo de Chegada

O *Time of Arrival* (ToA)(tempo de chegada) é a quantidade de tempo que um sinal leva para se propagar do transmissor ao recetor. Como a velocidade de propagação do sinal é aproximadamente constante e conhecida (ignorando as diferenças introduzidas pelos meios de propagação), o tempo de viagem de um sinal pode ser usado para calcular uma estimativa da distância. Múltiplas medições podem ser combinadas com trilateração e multilateração para estimar a localização do recetor, tal como é feito no sistema GPS. Os sistemas que utilizam ToA, geralmente requerem um mecanismo de sincronização para manter os relógios dos sensores sincronizados (e.g. usando repetidores para estabelecer emparelhamento) [BM14].

A precisão dos métodos baseados em ToA sofre frequentemente pelos efeitos de múltiplo caminho, que são causados pela reflexão e difração do sinal em objetos (por exemplo, paredes interiores, portas ou mobiliário) no ambiente. No entanto, é possível reduzir o efeito do multicaminho aplicando técnicas de dispersão temporal ou espacial [Com+11; Pou+12].

A distância pode ser calculada usando o tempo de viagem multiplicado pela velocidade da luz. Para isso requerem-se pelo menos três âncoras para ter a localização no plano-domínio 2D e quatro âncoras para localização 3D. A precisão do posicionamento é definida pela largura de banda de um sinal, e pela taxa de amostragem [AP03]. Tal como está ilustrado na figura 2.2, quando o sinal é amostrado no recetor, existe um erro de estimação do ToA que é tanto menor quanto menor for o período de amostragem e a duração do impulso de sinal (i.e. maior largura de banda). Por exemplo, num sistema sem fios com 10 MHz de largura de banda é amostrado com uma frequência de amostragem de pelo menos 20 MHz e só pode medir a duração de tempo com uma resolução de até $1 * 10^{-7}$ s. Portanto, o erro máximo de distância é de até $3 * 10^{8} * 10^{-7} = 30m$. Quando o sistema possui uma largura de banda de 1 GHz, o recetor pode medir até uma resolução de $1 * 10^{-9}$ s, de modo que o erro máximo na distância é inferior a 30 cm.

As soluções populares atuais aplicadas a ToA são sistemas de UWB [Dar+09]. A precisão que um sistema UWB pode alcançar é de até 1 cm [Liu+07]. No entanto, requer uma largura de banda muito grande, bem como o desenho de hardware específico para suportar a localização, o que resulta num custo de hardware elevado. [YS15].

2.2.4 RSSI - Indicação da intensidade do sinal recebido

A *Received Signal Strength Indicator* (RSSI)(Indicador de Potência do Sinal Recebido) é uma medição do nível de potência recebida. Como as ondas de rádio se propagam de acordo com leis da física, sabendo o meio de propagação, a distância pode ser aproximada

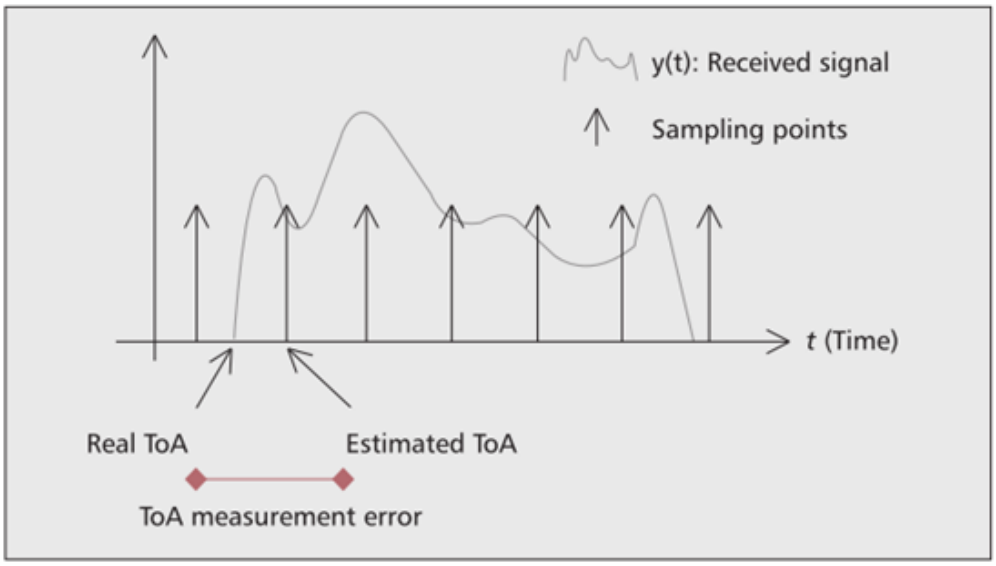


Figura 2.2: Ilustração do impacto da taxa de amostragem face às oscilações do sinal recebido [YS15]

pela equação (2.1) [Red], onde P_r é a Potência recebida, P_t é a Potência transmitida, G_r é o Ganho da antena recetora, G_t é o Ganho da antena de transmissão, λ representa o comprimento de onda e γ representa o *Path Loss Exponent* (PLE) (Expoente de Perdas ao longo do percurso) que modela a influência do meio de propagação.

$$P_r = P_t G_t G_r \left(\frac{\lambda}{4\pi}\right)^2 \frac{1}{d^\gamma} \text{ onde } \begin{cases} \gamma < 2 \text{ em corredores} \\ \gamma = 2 \text{ em espaço aberto} \\ \gamma > 3.5 \vee \gamma < 4 \text{ em ambientes urbanos} \\ \gamma = 5 \text{ com forte atenuação} \end{cases}, \tag{2.1}$$

O interior dos edifícios não é um ambiente de espaço livre, de modo a que a precisão é significativamente afetada pela reflexão e absorção das paredes. Os objetos não-estacionários como portas, móveis e pessoas podem representar um problema ainda maior, pois podem afetar a potência do sinal de forma dinâmica e imprevisível.

Muitos sistemas usam uma infraestrutura WiFi existente para fornecer informações de localização [Cha+10; Chi+10; Lim+10]. Infelizmente, as medições da potência do sinal WiFi apresentam bastante ruído, por isso existe bastante investigação em curso que se foca em fazer sistemas mais precisos usando estatística e métodos para filtrar os dados [Zho+15]. Esta dissertação revê no capítulo 4 vários algoritmos e propõe um novo, precisamente com este objetivo. Alguns sistemas de posicionamento WiFi por vezes são usados *outdoor* juntamente com GNSS nos dispositivos móveis, onde permitem melhorar a precisão de localização.

2.3 Localização avançada utilizando *fingerprint*

Os algoritmos de localização *indoor* que usam *fingerprint* (padrões) [Kja07] baseiam-se em métodos determinísticos [BP00] ou probabilísticos [Mad+05; YA08]. Os algoritmos determinísticos usam uma métrica de similaridade para diferenciar a medição de sinal online e dados pré-armazenados que caraterizam zonas, mais conhecidos como *fingerprint data*. O alvo é considerado como localizado na localização mais próxima numa determinada *fingerprint* no espaço de sinal [Han+14], considerando-se métricas como a distância Euclidiana [Au+13; Fen+12; HC16a], similaridade de coseno [HC14] e a similaridade de Tanimato [Jia+12]. A principal vantagem dos métodos determinísticos é a sua facilidade de implementação. Os métodos determinísticos tradicionais podem ser facilmente implementados com base nos k vizinhos mais próximos (k-NN) e a complexidade computacional é frequentemente baixa. Outros algoritmos determinísticos mais avançados, como as máquinas de vetores de suporte [Hon+09], permitem melhor precisão de localização com maior custo computacional.

Os algoritmos probabilísticos baseiam-se na inferência estatística entre a medição do sinal alvo e a *fingerprint* armazenada [Mir+14]. Usando um conjunto de treino, esses algoritmos encontram a localização do alvo no local com a maior semelhança. O Horus [YA08] estima a localização do alvo usando um modelo probabilístico que reflete a distribuição do sinal no local. Dado um vetor de potências de um sinal $s = (s_1, ..., s_L)$ e L APs, o Horus seleciona a posição x que maximiza a probabilidade

$$\arg\max_x [P(x|s)]. \tag{2.2}$$

Na equação (2.2) considera-se que $P(x|s)$ é a probabilidade do alvo estar na posição x dados os sinais s. Pode ser ainda transformado em

$$\arg\max_x [P(x|s)] = \arg\max_x \left[\prod_{l=1}^{L} P(s_l|x) \right], \tag{2.3}$$

onde $P(s_l|x)$ (probabilidade do sinal s_l aparecer dada a localização x) pode ser aproximada por algumas distribuições paramétricas incluindo a distribuição Gaussiana.

Também podem ser usados outros algoritmos por interferência probabilística, como as redes Bayesianas [Mad+05; Nan+12], a maximização da expectativa [Ouy+12], a divergência de Kullback-Leibler [Mir+12], o processo Gaussiano [Fer+07] e o campo aleatório condicional [Xia+14].

Nos algoritmos probabilísticos, a cada estimativa de localização pode estar associado um intervalo de confiança [Jun+13]. Estes algoritmos também podem fundir a informação de diferentes sensores, como o movimento [Sei+10] e o som [Nan+12]. Contudo, estes algoritmos geralmente requerem algumas suposições probabilísticas, como o ruído gaussiano ou a independência probabilística. Além disso, os modelos probabilísticos de treino podem ser complicados e exigirem mais conjuntos de dados do que os algoritmos determinísticos tradicionais [Sei+10] .

Os algoritmos determinísticos e probabilísticos são usados em alguns sistemas de localização. As subsecções seguintes ilustram alguns exemplos.

2.3.1 Relação de Padrões de Sinal Espacial e Temporal

A localização das *fingerprints* tradicionais baseia-se em vetores de RSS [BP00]. Devido ao ruído nas medições (efeitos multi-caminho [Xia+15b]), o alvo pode ser mapeado para uma posição distante de vetores de sinal similares [Liu+12; Tsu+13]. A precisão pode ser melhorada se a localização for estimada considerando conjuntamente observações temporais e espaciais:

- Os padrões temporais denotam os padrões de sequência de sinal WiFi medidos durante o movimento no ambiente *indoor*. Os padrões de sinal medidos ao longo da trajetória percorrida podem ser usados para inferir os locais. Em comparação com um vetor de sinal único em um local fixo, o padrão carrega informações temporais que podem ser usadas para restringir e corrigir a localização baseada em *fingerprints* WiFi.

- Os padrões espaciais estão relacionados com a distribuição geográfica dos sinais. Os padrões temporais WiFi muitas vezes requerem o conhecimento do movimento do utilizador (caminho percorrido ou direção), que pode não estar disponível ou ter a precisão necessária. Os padrões de sinais geográficos podem, portanto, ser usados para restringir a localização do utilizador.

A Tabela 2.1 mostra uma amostra com trabalhos que utilizam padrões de localização.

Tabela 2.1: Métodos que utilizam padrões temporais e espaciais

Categoria	Esquema	Sinais	Local Indoor	Informação Adicional	Precisão	Limitações e Robustez
Padrões temporais	PWF [Kim+12]	Picos RSSI duma sequência temporal.	Corredores; Zonas com muitos APs.	Direção da caminhada.	<2 m	O erro aumenta se o utilizador se movimentar rápidamente.
	Walkie-Markie [She+13]	Sequência RSSI.	Corredores	Direção da caminhada.	1.8 m	Funciona melhor quando o utilizador se mantém numa direção num corredor.
Padrões espaciais	UnLoc [Wan+12]	Pontos de referência WiFi.	Corredores; Zonas com muitos APs.	Direção da trajetória.	<2 m	A cobertura dos pontos de referência WiFi não podem abranger uma grande zona.
	HALLWAY [Jia+13]	Níveis dos valores RSSI.	Zonas de quartos/divisões.	N/A	~90%	As diferenças de nível podem não ser significativas afetando a precisão.
	Intersecção/Div de Cobertura de Sinal Wifi [HC14; KK12]	Valores semelhantes dentro de uma localização.	Espaços abertos e amplos; Necessita de múltiplos APs.	N/A	<6 m	Com os APs a cobrir a mesma região pode não haver restrições rigorosas.

Relativamente à categoria de padrões temporais, o método *Peak-based WiFi Fingerprinting* (PWF) [Kim+12] considera o pico numa sequência de valores de sinal para obter a localização. O pico (com um valor de sinal alto predefinido) da sequência de sinal no local indica que um AP WiFi está próximo do ponto de medição. Assim, o sinal com um

valor elevado mostra maior confiança do que outro mais fraco, para indicar a posição alvo. Para encontrar o pico, o utilizador necessita de recolher a sequência de potências WiFi enquanto se está a mover. O sistema deteta o pico, e a partir dele, encontra a localização correspondente. Este algoritmo consegue diminuir o erro, especialmente quando os APs são instalados nos tetos das divisões. No entanto, as informações de movimento do utilizador precisam de ser conhecidas durante a medição. Além disso, se o utilizador está a movimentar-se muito rapidamente, o pico da medição pode ser detetado com um erro significativo devido à falta de sinais de AP.

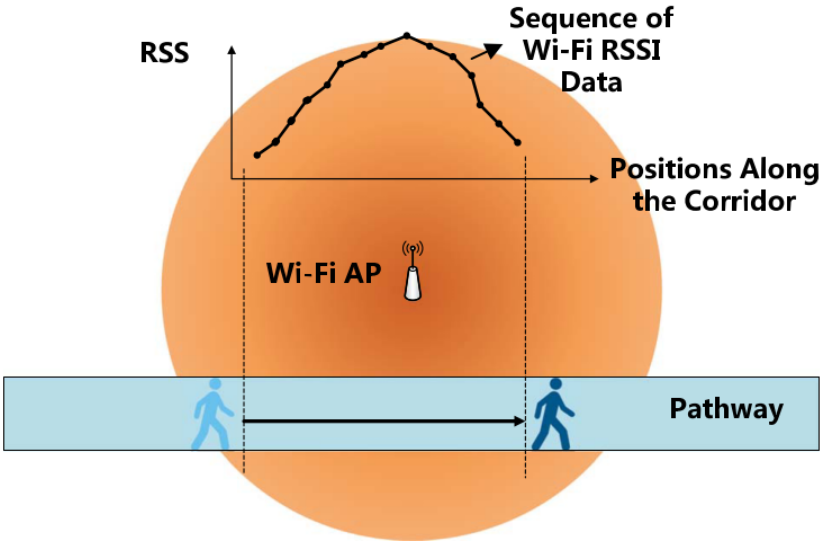


Figura 2.3: Ilustração dos padrões temporais de sinal no sistema *Walkie-Markie* [She+13]

Em ambientes com ruído, é mais robusto considerar uma sequência completa de dados do que olhar apenas para um único valor de pico. O Walkie-Markie [She+13] considera uma sequência de sinal completa para a classificação de locais. O Walkie-Markie regista primeiro os vetores RSSI do WiFi para usar como padrões para os diferentes corredores. Conforme está ilustrado na figura 2.3, um utilizador a caminhar ao longo do corredor pode "sentir" o aumento e diminuição da intensidade do sinal de um AP próximo. A sequência de dados RSSI pode formar o padrão para um determinado corredor. Ao combinar a sequência RSSI do utilizador durante a caminhada, o Walkie-Markie conhece a localização e as informações do mapa do alvo.

No entanto, a localização por sequência de sinal é mais adequada para espaços como corredores estreitos do que para zonas abertas, como as existentes nos aeroportos ou estações de metropolitano ou comboio. Além disso, as informações de movimento do utilizador precisam de ser tidas em conta para comparar as sequências de sinal com os padrões guardados, devido ao utilizador geralmente não caminhar com um padrão fixo.

Relativamente aos padrões espaciais, os APs WiFi podem ser instalados em posições internas específicas, como corredores ou escritórios. Assim, podemos identificar valores

de sinal notáveis apenas medidos numa área específica, que podem formar WiFi *landmarks* (ponto de referência) e serem usados para classificar exclusivamente as regiões correspondentes. As *landmarks* são recolhidos e armazenados numa base dados, para poderem ser utilizadas como comparação quando se utiliza a aplicação. Sistemas como o UnLoc [Wan+12] e o MapCraft [Xia+14] usam as medições individuais de APs e a informação da base de dados para corrigir os resultados de localização.

Conforme está ilustrado na figura 2.4, dada a informação online sobre os pontos de referência (região sombreada à direita), a trajetória do utilizador pode ser corrigida para a área correta, que se encontra à direita do valor estimado por outro método. Assim, podemos melhorar a precisão da estimativa de localização sem correções usando *landmarks* representada à esquerda. Como os APs WiFi podem não estar densamente instalados em alguns edifícios, as correções com *landmarks* podem não estar disponíveis. Portanto, também podem ser usados outros sinais, como campo magnético, para realizar uma correção de localização adicional [Azi+09], o que aumenta a complexidade da implementação do sistema.

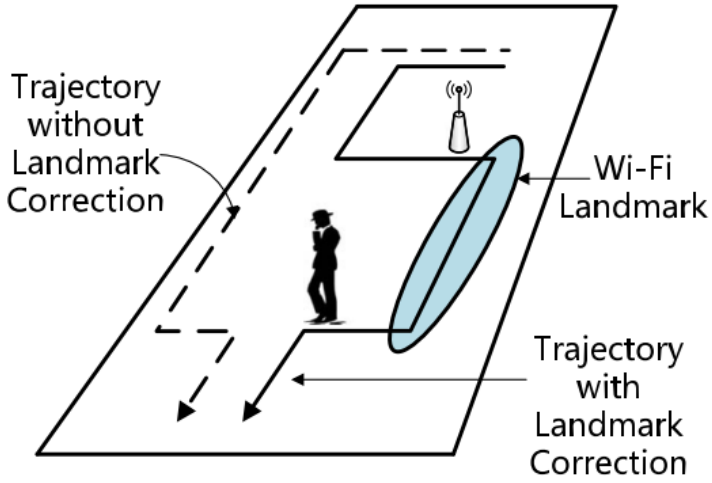


Figura 2.4: Utilização de padrões espaciais [Wan+12].

As abordagens apresentadas acima consideram o uso de APs individualmente como pontos de referência, o que na implementação pode não ser robusto e apresentar elevados erros de medição. Se a potência de um AP for definida como forte, então a cobertura do AP pode formar um grande ponto de referência. Nesse caso, podem-se considerar conjuntamente vários APs para melhorar a precisão. O sistema HALLWAY [Jia+13] baseia-se na observação do RSSI de diferentes APs, por depender da localização. Por exemplo, se se denotar a potência de sinal para AP_1, AP_2 e AP_3 como s_1, s_2 e s_3, respetivamente, diferentes salas poderão ter ordens distintas, em função da proximidade com os APs (na sala A, podemos considerar uma ordem de $s_1 < s_2 < s_3$ enquanto na sala B, podemos encontrar $s_1 < s_3 < s_2$). O sistema HALLWAY utiliza essa diferença para classificar os

quartos. Usando diferentes níveis de RSSI é possível reduzir a influência do dispositivo e pequenas flutuações do sinal. Portanto, este algoritmo pode localizar um terminal na zona de uma sala ou escritório.

Outra alternativa baseia-se na informação sobre a zona de cobertura de diferentes APs. Para um dado AP, a intensidade do sinal atenua-se em função da distância e restrições geográficas sobre a localização do alvo. Com base nessa observação, o trabalho em [HC14] restringe a estimativa de destino usando a intersecção da área de cobertura de sinal de vários APs. Para cada AP detetado, ele primeiro divide a sua área de cobertura de acordo com níveis de sinal discretos. Similarmente, usando uma abordagem probabilística, o trabalho em [KK12] também considera o uso da cobertura do sinal para reduzir a área de busca da localização do alvo. No modo online, o alvo é mapeado pela primeira vez para a interseção de vários setores. Então, dentro da restrição, o sistema encontra os pontos de referência com os padrões de sinais mais semelhantes como estimativa de localização através de mapeamento determinístico ou probabilístico. Desta forma, as restrições de cobertura eliminam os vizinhos mais próximos dispersos no espaço de sinal com alta similaridade. Estes métodos de interseção e esquemas de divisão também têm sido utilizados em alguns sistemas de localização de redes de sensores [Cha08; Zha+15].

Se os APs estiverem todos localizados dentro de uma região pequena, a interseção dos APs pode tornar-se demasiado grande e pode não restringir suficientemente a estimativa final. Portanto, a filtragem AP virtual [Mar+10] (filtragem dos endereços MAC gerados a partir do mesmo router físico WiFi) pode ser realizada antes de encontrar a cobertura do sinal e a decisão de localização final.

Em suma, os padrões temporais e espaciais ajudam a discriminar os locais e melhorar significativamente a precisão de posicionamento. No entanto, esses padrões de sinal geralmente funcionam melhor em determinados locais internos, como corredores estreitos [Kim+12; Wan+12] e áreas com boa cobertura de APs [KK12]. Por estas razões, quem desenhar um sistema utilizando estas técnicas precisa conjuntamente das propriedades dos locais e de selecionar algoritmos adequados para os mesmos. Além disso, pode ser necessária uma análise detalhada da zona e um pré-processamento dos dados para extrair e utilizar esses padrões.

2.3.2 Localização Colaborativa entre Dispositivos Móveis

A maioria dos trabalhos atuais na localização utilizando *fingerprints* WiFi baseiam-se em estimativas independentes, isto é, o sistema localiza cada alvo de forma independente [BP00] sem considerar as localizações relativas dos outros. Devido a erros de estimação independentes, dois alvos fisicamente próximos podem ter locais marcadamente diferentes estimados [Liu+12]. Assim, se a informação de posições relativas puder ser obtida e utilizada, os resultados da estimativa podem ser melhorados a partir da localização individual. Trabalhos recentes começaram a investigar as possibilidades de localização colaborativa. A principal motivação advém basicamente das seguintes tendências em

computação móvel:

- Contexto de localização da interação social: No ambiente interior, as pessoas podem reunir-se em cenários sociais típicos [Li-06]. Nos museus, as pessoas muitas vezes navegam através das exposições com a sua família e amigos. A interação social entre eles, portanto, mostra o padrão de localização. Com base nesse contexto, foi proposta uma aplicação *Location-Based Service* (LBS)(serviço baseado em localização) *indoor* [Con+10].

- Dispositivos móveis e sensores avançados: Atualmente, os *smartphones* implementam várias técnicas de deteção dos vizinhos. Um dispositivo móvel pode facilmente descobrir outros na sua vizinhança utilizando uma das várias interfaces rádio disponíveis (como Bluetooth [Liu+13], WiFi [All15; Ban+10] ou *Near Field Communication* (NFC) [Jun+13]). Portanto, os *smartphones* têm a possibilidade de detetar a existência de utilizadores vizinhos.

Na Tabela 2.2 é apresentada uma lista com trabalhos sobre localização colaborativa. Com base na precisão da medição de distância mútua, podemos repartir os trabalhos relacionados nas duas categorias seguintes: algoritmos baseados na distância e na proximidade:

- Baseado na distância: Tem havido trabalhos focados no uso de sensores [Pen+07; UN13] em *smartphones* para medir distâncias entre utilizadores. Essas distâncias entre pares podem ser utilizadas para formar o gráfico de rede (topologia) de diferentes utilizadores [Liu+12].

- Baseado na proximidade: Devido à diversidade de comportamentos humanos dinâmicos e interação social, a distância precisa entre pares de utilizadores pode não estar sempre disponível. Como a informação de proximidade pode não ser muito precisa, a inferência probabilística é frequentemente usada.

O sistema Virtual Compass (VC) [Ban+10] permite localizar os vizinhos nas proximidades com base em WiFi direct e Bluetooth. O VC propõe um algoritmo baseado no algoritmo Vivaldi [Dab+04] para estimar as coordenadas dos utilizadores. Todos os utilizadores formam uma rede, e o algoritmo de Vivaldi obtém a topologia de rede. Para diminuir o erro na medição de distância baseada em rádio, o VC funde o Bluetooth e o WiFi direto juntos. Ao modelar o intervalo de confiança de ambos os sensores, o VC obtém uma melhor precisão de distância do que utilizando um único sensor. No entanto, o uso de sinais de RF para a medição de distância é vulnerável ao ruído no meio. O VC não foi especificamente construído para um posicionamento absoluto.

O sistema Centaur [Nan+12] implementa uma rede Bayesiana para localização colaborativa. O Centaur não usa um gráfico rígido, que é vulnerável a erros de medição

Tabela 2.2: Esquemas de Localização Colaborativa

Categoria	Esquema	Informação	Precisão	Robustez das medições	Mobilidade do Utilizador	Limitações
Distância	VC [Ban+10]	WiFi Direct Bluetooth	Média	Afetado pelo efeito de múltiplos caminhos	Estático	Não é utilizado para posicionamento absoluto. A medição de distância pode não ser muito precisa
	Centaur [Nan+12]	Som	Alta	Alta	Estático	Sincronização entre utilizadores necessária; Concebido para dispositivos estáticos
Proximidade	ZCL [Li-06]	ZigBee	Baixa	Afetado pelo efeito de múltiplos caminhos	Dentro de um grupo pequeno ou estático	Os utilizadores precisam de estar perto uns dos outros; Não podem existir utilizadores que se movem aleatoriamente.
	Social-Loc [Jun+13]	WiFi Direct Bluetooth	Baixa	Afetado pelo efeito de múltiplos caminhos	Pedestre	Os limiares de deteção de encontro podem sofrer com ruído; Informações de encontro e não-encontro usando RSSI podem não ser precisas
Ambientes NLOS	Usando SDP [MB15]	Radio	Média	Alta	Estático	Apenas testado em simulações no MatLab.

para distâncias grandes. Em vez disso, o Centaur encontra os locais com a máxima probabilidade para todos os dispositivos envolvidos, mostrando mais robustez em ambientes ruidosos. No entanto, o protótipo do Centaur centra-se na localização de dispositivos estáticos, não tendo sido explorado o posicionamento dinâmico de utilizadores.

Alguns trabalhos recentes em esquemas baseados em proximidade [Hig+11; Hig+14] utilizaram as paragens momentâneas dos utilizadores para medir a distância exata entre si. Para melhorar a robustez na implementação prática, é também necessário considerar o movimento dos utilizadores e a imperfeição das medições de distância mútua. A proximidade, em vez de valores de distância precisos, pode ser utilizada para a medição dinâmica. No sistema [Li-06] é proposta a *ZigBee-based collaborative localization* (ZCL)(localização colaborativa baseada em ZigBee) para estimar a localização absoluta em cenários de ambiente similar. O ZCL usa primeiro o rádio ZigBee como método de deteção de vizinhos. Em seguida, calcula uma pontuação de confiança para cada alvo dentro duma região, de acordo com a combinação do modelo de movimento e estimativa da localização WiFi. Com base na diferença entre as pontuações de confiança, o sistema corrige conjuntamente as estimativas vizinhas através de um algoritmo distribuído proposto. Este esquema funciona como a "atração magnética", filtrando os locais candidatos com baixa confiança.

Embora o ZCL esteja adaptado a cenários de aplicações interessantes em museus e exposições, o ZigBee usado no sistema também pode sofrer com a transmissão multicaminho e, portanto, a informação de proximidade pode ter um erro associado. Para reduzir este problema, podem ser aplicados filtros de média sobre várias amostras de ZCL, com um custo de tempo de espera mais longo. Além disso, o ZCL funciona melhor quando os utilizadores estão dentro de um grupo numa região relativamente pequena (estática ou movendo-se na mesma direção), o que não é o comportamento habitual dos utilizadores comuns (tende a ser mais dinâmico).

Se os utilizadores estiverem a andar em edifícios, a informação de proximidade útil

entre os utilizadores é muitas vezes temporária. Alavancando a interação acima, o Social-Loc [Jun+13] propõe a utilização do facto dos utilizadores se estarem a aproximar ou afastar para a sua localização. Um utilizador pode encontrar outro, que é definido como "encounter". Se ele não se encontrar com outro utilizador específico durante o processo de localização, tal evento é definido como "non-encounter". O Social-Loc utiliza esses eventos de encontro e não encontro para corrigir cooperativamente os erros de localização.

No Social-Loc, as localizações dos utilizadores são inicializadas usando uma localização tradicional baseada em *fingerprints* WiFi [YA08]. Cada estimativa de um dado utilizador tem vários candidatos (pontos de referência), com probabilidade prévia diferente. Na figura 2.5, ilustra-se o cenário em que dois utilizadores se encontram, e nesse caso, os seus locais estimados devem entrar em sobreposição. Assim, as estimativas de local que não satisfaçam as informações de encontro de dois utilizadores seriam filtradas. A estimativa de localização corresponderá a um local onde o utilizador A e B se encontram. Se o utilizador A e B não se encontrarem numa posição candidata, a confiança para essa posição na estimativa final de WiFi irá diminuir. Dada a probabilidade posterior acima, as estimativas de localização final WiFi podem ser atualizadas para todos os utilizadores a usarem o Social-Loc.

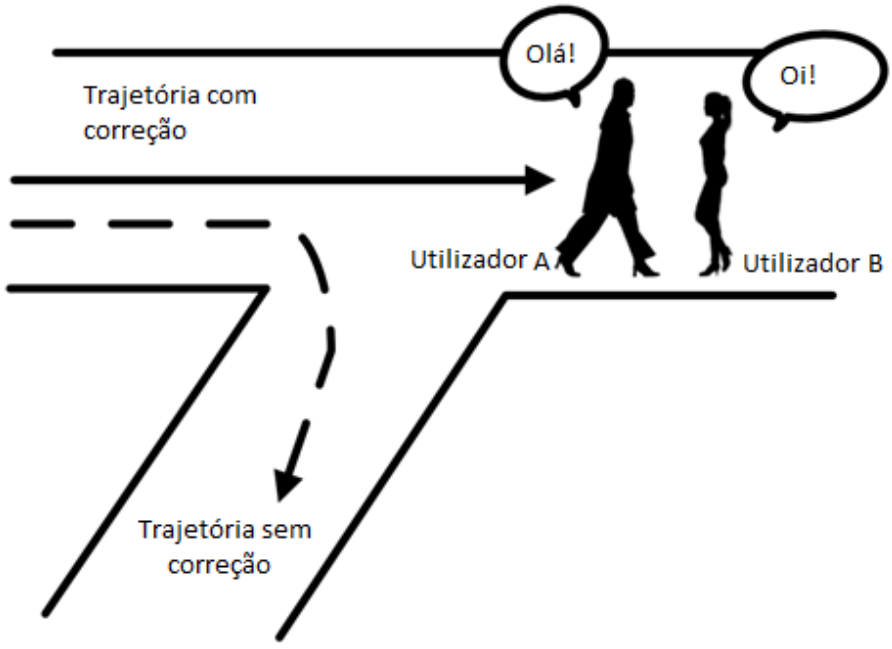


Figura 2.5: Representação da localização de dois utilizadores com ajuda da informação do seu encontro (adaptado de [HC16b]).

Em resumo, a localização colaborativa nos trabalhos acima permitiu melhorar a precisão da localização. No entanto, na implementação prática, várias questões precisam ser consideradas. A complexidade computacional é alta para a localização colaborativa

devido à comunicação entre dispositivos [Yan+14] e aos requisitos de sincronização [Liu+12]. A mobilidade dos utilizadores também torna a colaboração desafiadora, uma vez que as posições relativas dos mesmos mudam frequentemente, especialmente num aeroporto ou em estações de comboios [Liu+14]. Durante a interação social, a colaboração entre nós pode tornar pública informação sobre os proprietários dos dispositivos. Portanto, para resolver os problemas de privacidade, a localização colaborativa requer ainda que no futuro seja criado um protocolo seguro para compartilhar a informação referente aos locais [Ban+10; Sho+12; Tsa+10].

2.3.3 Localização Assistida por movimento

A localização WiFi assistida por movimento é uma técnica híbrida clássica para localização *indoor*. Tem avançado rapidamente nos últimos anos devido à aplicação generalizada de sensores de movimento nos dispositivos móveis. Neste capítulo, são apenas abordados os mais significativos avanços recentes na localização assistida por movimento:

- Avanços na medição do movimento: a monitorização exata dos comportamentos de movimento do utilizador é importante para a precisão da localização. O principal desafio na obtenção de informações de movimento é que os sensores inerciais nos *smartphones* comerciais sofrem com frequência de problemas de calibração e ruído nas medições. A contagem de passos é atualmente a principal aproximação utilizada para capturar o movimento e o trajeto dos pedestres [Har13]. Consequentemente, os trabalhos recentes tentam melhorar a deteção da caminhada, a contagem dos passos e a medição do seu comprimento. Além disso, a forma de se adaptar aos diferentes perfis dos utilizadores também é um desafio.

- Modelos de fusão eficientes: A maneira como se junta o movimento e a deteção WiFi é essencial para a precisão da localização. O modelo utilizado na fusão precisa de capturar a correlação (temporal ou espacial) entre os sinais medidos. Além disso, se o modelo for muito complicado, o gasto energético em computação será tendencialmente elevado. Portanto, um algoritmo que realize a fusão tem que ser preciso e eficiente [Sun+13; Xia+14], sendo a análise do custo/benefício importante.

A figura 2.6 mostra um sistema típico de localização assistido por movimento. Os sensores de movimento medem a distância caminhada pelo utilizador (entre duas medições WiFi sequenciais) e a direção. Combinando estas informações e a medição do sinal WiFi, é possível com um algoritmo de fusão obter a estimativa de localização.

Com os acelerómetros, giroscópios e magnetómetros, os *smartphones* recentes podem capturar a direção do percurso, distância e informações dos gestos. Estes sensores são geralmente chamados de sistemas *Micro-Electro-Mechanical Systems* (MEMS) (Sistemas Microeletromecânicos). Os acelerómetros medem a aceleração linear a três dimensões $\left(\frac{m}{s^2}\right)$ ou a força da gravidade. Os giroscópios dão a velocidade angular $\left(\frac{rad}{s}\right)$ e medem a

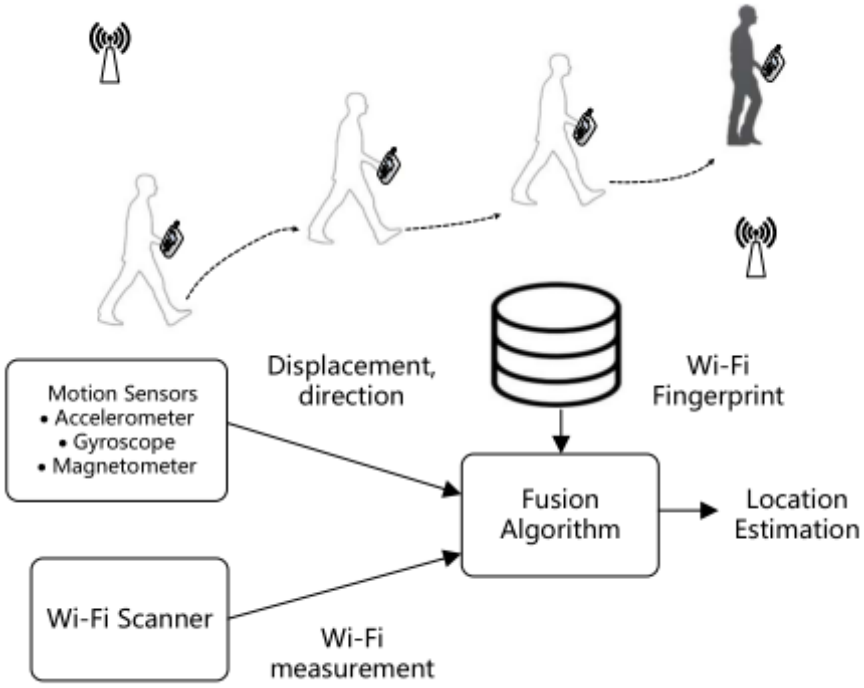


Figura 2.6: Ilustração da fusão das informações relativas ao sinal WiFi e dos sensores de movimento para uma localização mais precisa [HC16b].

orientação a partir do momento angular. Os magnetómetros fornecem a força e a direção dos campos magnéticos [Roy+14]. Com giroscópios e magnetómetros, pode-se saber a direção do utilizador. Com base nos sensores e nas técnicas de filtragem de sinal, pode-se realizar uma deteção de movimento mais avançada, como a rotação de dispositivos, contagem de passos e reconhecimento de gestos. Para medir a distância percorrida pelo utilizador, o pedómetro (contador de passos) é um esquema adequado para a localização [BH13]. Em termos de obtenção do deslocamento de posição, o pedômetro tem melhor desempenho do que a integração direta da aceleração, uma vez que a derivação do sinal leva a grandes erros após a dupla integração (necessária para se obter a posição). Na figura 2.7 é possível observar as leituras de um acelerómetro enquanto o utilizador se desloca, sendo estas leituras convertidas para o número de passos. Nos trabalhos recentes que utilizam o pedômetro [BH13], tipicamente são destacadas as seguintes três componentes:

- Deteção de Caminhada: Esta componente é utilizada para classificar o estado de movimento atual do alvo (como usar limites nas leituras do acelerómetro ou algoritmos avançados de aprendizagem [BH13]). Se um utilizador é identificado como "em movimento", a contagem de passos começa a funcionar.

- Contagem de passos: A deteção de passos pode basear-se na deteção de picos (como no caso da figura

- Medição do comprimento da passada: Através da deteção do início da caminhada e da contagem dos passos, o dispositivo móvel pode medir a distância de cada passo multiplicando o comprimento da passada pela contagem dos passos. O comprimento de cada passo depende da frequência do passo, da altura do utilizador e de outros fatores [Zij04]. Alguns trabalhos utilizam uma distribuição Gaussiana para modelar o ruído no comprimento do passo. Trabalhos mais avançados implementam alguns modelos de passos lineares para calibrar a relação entre o comprimento da passada e a frequência do passo. Em [LS13] é modelada a relação entre o comprimento do passo e a mudança de altura do dispositivo durante a caminhada.

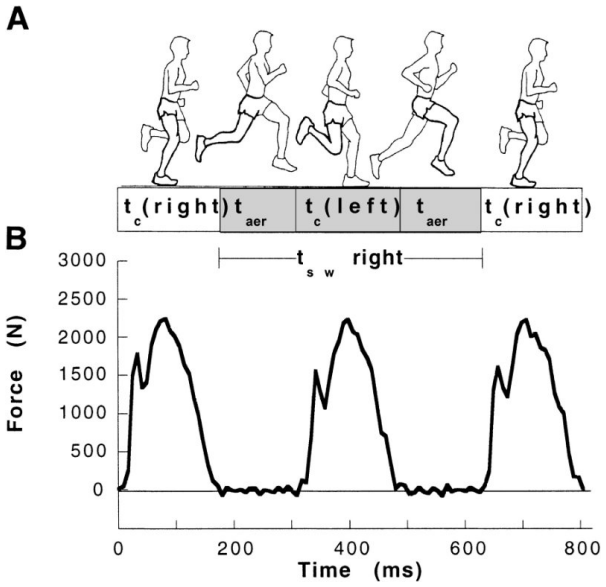


Figura 2.7: Leituras do acelerómetro de um *smartphone*. Cada pico das medições representa um passo [BJ14].

Em aplicações reais é bastante complicado descobrir estes parâmetros, pois o utilizador pode caminhar de formas diferentes e apresentar comportamentos diversificados. Existe ainda a possibilidade de o *smartphone* trocar de utilizador o que dificulta ainda mais o processo. Alguns trabalhos recentes propõem a calibração do contador de passos [Li+12], que mede o comprimento do passo quando o utilizador está a andar em corredores. Com restrições, usando um mapa *indoor*, o sistema pode estimar a distância de cada passado com mais precisão e aprender o seu comprimento. Trabalhos mais avançados, como em [Xia+15c], calibram o comprimento do passo através da aprendizagem utilizando um filtro de partículas ou a maximização da expectativa.

A maioria dos trabalhos geralmente supõem que o *smartphone* está numa posição estática relativamente ao corpo do utilizador durante todo o movimento. No entanto, na realidade, a posição relativa do *smartphone* muda e, portanto, pode influenciar o contador de passos e a orientação. Portanto, é necessária alguma compensação no ângulo de rotação para corrigir estes desvios, que são introduzidos pela mudança relativa da posição do

dispositivo em relação ao corpo do utilizador [RM13]. Outra abordagem é usar dispositivos wearable (ex: smartwatch), que podem ser anexados ao corpo humano, e capturar informações mais precisas de movimento para a localização [Har+13; LS13; Sou+13].

Tipicamente existem bastantes problemas na fusão de sensores, e por essa razão são realizadas algumas simplificações nos algoritmos. Na tabela 2.3 estão presentes as principais abordagens utilizadas para tentar resolver este tipo de problemas.

Tabela 2.3: Abordagens em sistemas ajudados pela informação do movimento do utilizador em localização WiFi

Esquema	Algoritmo de Localização	Informação de Movimento	Complexidade	Robustez	Precisão	Limitações
Zee [Rai+12]	Filtro de Partículas	Contador de passos utilizando a autocorrelação; Orientação.	Alta	Utiliza informações de mapa para filtrar partículas incorretas; Robusto em corredores estreitos.	2 m	Os dados do sinal *crowdsourced* podem conter ruído.
XINS [Gao+11]	Filtro de Partículas	Contador de passos utilizando a autocorrelação; Orientação.	Alta	Utiliza um filtro de partículas para fundir sinais diferentes.	N/A	Funciona melhor quando todos os sinais (célula , GNSS e WiFi) estão disponíveis.
Graph-Fusion [Hil+14]	Filtro de Partículas e gráfico de discretizaçãodo mapa indoor	Contador de passos utilizando a deteção de picos; Orientação; Estimativa do comprimento da passada em linha reta.	Média	Afetado pelo efeito de múltiplos caminhos.	2 m	Os espaços indoor mais amplos são difíceis de discretizar.
HMM Fusion [Sei+10]	HMM	Contador de passos; Orientação.	Baixa	Simplifica os modelos de mapas indoor.	6 m	Treinar o HMM requer um grande conjuntos de dados de treino e é um processo muito dispendioso.
Moloc [Sun+13]	Fingerprint com probabilidade mais elevada	Contador de passos utilizando a autocorrelação; Orientação; Perfil de movimento.	Baixa	Exige independência entre sinais WiFi e as leituras de movimento; Sofre bastante com o ruído.	1 m	É necessário o perfil de movimento do utilizador.
MapCraft [Xia+14]	Campo aleatório condicional	Contador de passos; Orientação.	Baixa	Trata os sinais WiFi e leituras de movimento em conjunto; Nenhuma suposição de dependência necessária; Elevada Robustez.	2 m	Requer um grande conjuntos de dados de treino e é um processo muito complicado.
SmartSLAM [Shi+11; Shi+12]	Filtro de Partículas e Filtro de Kalmen	Contador de passos utilizando o acelerómetro; Orientação usando a bússola digital.	Muito Alta	Não requer informação prévia; Melhora com a rosbustez utilização.	4 m	Complicado produzir o mapa e localizar-se no mesmo de imediato.

A fusão tradicional concentra-se frequentemente nos problemas de localização [Hil+14] reduzindo os erros de localização produzidos nas medições dos sinais sem fios [Sun+13]. As informações de movimento em fusão filtram as posições incorretas retornadas da localização WiFi, especialmente sob flutuações de sinal ou *fingerprints* ambíguas. As informações presentes nos mapas estão frequentemente disponíveis e podem contribuir para a filtragem de zonas [Rai+12].

O *Simultaneous Localization and Mapping* (SLAM) baseado em WiFi [BDW06a] é um algoritmo que pretende realizar a localização simultânea do alvo e a construção dos mapas *indoor*. Para se aproximar disso, o SLAM utiliza fusão de WiFi e medição de movimento para minimizar em conjunto a diferença de localização com a estrutura interna que pretende inferir no mapa. Ele também tem sido amplamente aplicado em cenários de aplicação de localização de robôs sem informações de mapa explícitas [BDW06b].

A localização em robótica [Thr00] é um dos principais alvos de estudo sendo a telerobótica e os sistemas autónomos as principais áreas que utilizam a informação sensorial para melhorar a localização sem fios. Algumas abordagens iniciais, como a SLAM de intensidade do sinal, utilizam o processo Gaussiano para estimar a posição dos robôs. No

entanto, os robôs são diferentes dos padrões humanos em movimento. É mais difícil associar os dados do sensor à informação de movimento humano devido à maior aleatoriedade no movimento pedestre. Com base na ideia de SLAM [AR12], alguns trabalhos recentes como WiFi GraphSLAM [Hua+11] e WiSLAM [BR11] propõem a fusão de WiFi e do movimento do utilizador para a localização *indoor* pedestre. O WiFi GraphSLAM formula um problema de otimização para encontrar o mapeamento da posição alvo para o mapa *indoor*. Para facilitar a convergência de SLAM, WiSLAM usa, em vez disso, a inferência Bayesiana na localização do alvo.

Um dos trabalho mais recentes que utiliza SLAM para a localização em *smartphones* é o SmartSLAM [Shi+11; Shi+12]. O SmartSLAM apresenta um modelo de mobilidade que realiza a odometria através da detecção de passos e da orientação do *smartphone*. Também fornece um modelo de observação, baseado em *fingerprints* WiFi, que regista as características físicas do ambiente em termos de força de sinal de rádio. Integrando estes dois modelos como é mostrado na figura 2.8, o SmartSLAM analisa o movimento de um utilizador num prédio construindo um plano da zona, sendo possível obter o *layout* (mapa) estrutural dos corredores através dos quais um utilizador passa. A principal dificuldade desta abordagem é lidar com situações em que a planta não é conhecida ou os sensores disponíveis são imprecisos.

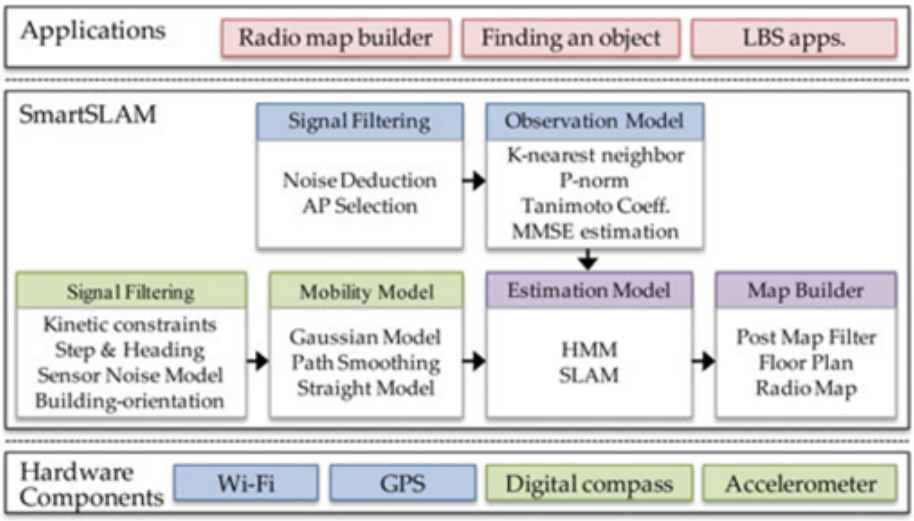


Figura 2.8: Arquitetura do SmartSLAM [Shi+12].

Para solucionar os problemas de SLAM, são usados algoritmos que pretendem encontrar o melhor gráfico de correspondência e resolver problemas de otimização [Hua+11]. Para abordar a localização tradicional ou problemas de SLAM, os algoritmos de fusão utilizam:

- Filtro de Kalman: O filtro de Kalman é uma formulação tipicamente usada para

descrever um sistema usando tempo discreto. A fusão de *fingerprints* WiFi e sensores inerciais baseados num filtro de Kalman tem sido estudada extensivamente em trabalhos como [AL+09; Che+15; EM06; Han+14; Yan+14]. Comparado com outras técnicas, como esquemas de *Mean root square* [Hua+11], o filtro de Kalman obtém melhores resultados em ambientes com ruído Gaussiano linear [Yan+14]. Assumindo que se conhece o modelo de movimento do utilizador e que este tem ruído Gaussiano, o filtro convencional de Kalman pode efetivamente resolver o problema de fusão de localização, especialmente para o modelo de movimento linear. Algumas extensões adicionais, incluindo o filtro de Kalman estendido e o filtro de Kalman unscented, foram propostas para abordar alguns problemas não-lineares.

- Filtro de partículas: Comparado com o filtro de Kalman tradicional, o filtro de partículas é mais geral e adequado para problemas de monitorização mais sofisticados com base no modelo de movimento não-linear. O filtro de partículas [Aru+02] começa por espalhar "partículas" na área *indoor* onde o utilizador se pode deslocar. Em seguida, as partículas que têm menor probabilidade de ser a posição atual do utilizador tendem a ser filtradas. Portanto, as partículas que se aproximam melhor da posição correta ganham uma maior confiança e são selecionadas como sendo trajetórias potenciais. As zonas mais prováveis ficam assim com mais partículas sobreviventes, obtendo as partículas nessa região um peso mais elevado na estimativa final. Na implementação prática, nem sempre são utilizados filtros de partículas porque os esquemas baseados em filtros de Kalman conseguem produzir resultados satisfatórios com custo computacional muito menor e maior facilidade de implementação.

Outros modelos de fusão tentam tornar o processo de filtragem mais eficiente [Hil+14; Sun+13; Xia+14]. Usando modelos probabilísticos eficientes ou simplificados, que implementam uma *Hiddden Markov Model* (HMM) (cadeia de Markov escondida) [Sei+10] ou um *Conditional Random Field* (CRF) [Xia+14] para simplificar o cálculo da localização enquanto obtêm estimativas satisfatórias.

O filtro Kalman de Fingerprint (FKF) [AL+09] utiliza um estimador que combina todas as medições de sinal atuais e passadas. No entanto, na prática, o ruído do sensor pode não ser Gaussiano e o modelo de movimento é bastante complicado, o que pode degradar o desempenho do filtro de Kalman tradicional. Além disso, o filtro de Kalman pode não ser adequado para alguns cenários com apenas informações do mapa e informações de movimento do utilizador [Xia+15a]. Para melhorar o desempenho do filtro de Kalman, o conhecimento (ou algumas heurísticas) da covariância de ruído pode ser integrada na implementação [Xia+15a].

No sistema de localização tradicional para pedestres, a informação do mapa está frequentemente disponível e, portanto, a localização precisa é mais importante. Em [Aru+02], o método Sequential Monte Carlo (SMC) foi implementado para a fusão de informação de WiFi e de sensores de movimento. No método SMC é utilizado um filtro de partículas.

Neste, as estimativas de WiFi e deteção de movimento são fundidas com base em partículas ponderadas. Com as restrições do mapa e resampling (reamostragem) de pesos, as partículas com localizações incorretas são filtradas e a estimativa de localização converge para uma posição mais precisa. Zee [Rai+12] e XINS [Gao+11] são dois esquemas que usam um filtro de partículas. O Zee utiliza as restrições do mapa para filtrar as partículas e diminui a região de busca da localização de destino. Portanto, as direções de orientação incorretas e trajetórias são filtradas. O XINS usa diversos sinais como GNSS ou *Global System for Mobile Communications* (GSM) para aumentar a possibilidade de correções de localização.

Uma representação em 2D de uma grande superfície aberta precisa de muitas partículas. Se discretizarmos o mapa e representarmos o caminho do corredor com um segmento de linha 1-D, o número de partículas pode ser reduzido. Usando esta técnica, o Graph-Fusion [Hil+14] propõe um sistema que simplifica a computação do filtro de partículas. Para reduzir a complexidade de localização online, durante o pré-processamento de mapas offline, o Graph-Fusion discretiza o mapa interno num gráfico simplificado removendo os graus de liberdade sem importância. Apesar de ser necessário muito preprocessamento de gráficos sobre os mapas, este algoritmo é eficiente em ambientes *indoor*. No entanto, para grandes espaços abertos *indoor*, como zonas de um aeroporto, a redução acentuada de partículas pode levar a grandes erros de estimação. A aleatoriedade dos peões no aeroporto é maior do que nos corredores e por esta razão é necessário um grande número de partículas nesses locais. Como o uso de muitas partículas aumenta significativamente a complexidade, alguns outros trabalhos focalizam o uso de modelos de fusão mais eficientes para substituir o filtro de partículas. O HMMFusion [Sei+10] propõe usar HMM para fundir os sensores e simplificar o processo de fusão. Como na formulação da HMM o estado de movimento depende apenas do estado anterior, a complexidade computacional é baixa. No entanto, a HMM requer uma grande sequência de dados de treino e o processo de treino offline é computacionalmente pesado [Sun+13].

Além do modelo HMM, MoLoc [Sun+13] modela a transição probabilística entre diferentes localizações na zona com base no comprimento da passada e orientação do utilizador. Em simultâneo, considera a probabilidade de se encontrar em diferentes zonas a partir de estimativas WiFi. Para estimar a localização, é utilizado em conjunto a informação WiFi e o movimento do utilizador. Através da correspondência de movimento, as *fingerprints* WiFi ambíguas em locais distantes também podem ser filtradas. Com base na suposição de independência entre sensores, o MoLoc encontra a localização com a máxima semelhança, o que simplifica o modelo de localização e aumenta a eficiência computacional. Na realidade, os sensores de movimento e os registos WiFi podem ser correlacionados [Xia+14; Xia+15a].

A fim de aumentar a precisão de localização e eficiência computacional, alguns trabalhos recentes consideraram conjuntamente a correlação entre medição WiFi e movimento. MapCraft [Xia+14; Xia+15a] propõe um sistema baseado num CRF [KT07]. Ao modelar as medições dos sensores internos (WiFi, campo magnético ou contador de passos) como

sequências de dados, o MapCraft utiliza CRF para mapeá-los no mapa interno e localizar o utilizador. Sem espalhar muitas partículas, o CRF no MapCraft considera conjuntamente as observações dos sinais e as medições de movimento. O treino offline do CRF é computacionalmente mais exigente [KT07] que o da HMM [Coh07]. Para garantir uma alta precisão de localização, geralmente é necessário um conjunto de dados mais abrangente com informações sobre peões, o que dificulta o treino do CRF. Todas as sequências devem ser rotuladas previamente para manter a precisão do treino.

Para reduzir o custo da recolhas dos dados, uma abordagem potencial é modelar a propagação do sinal para a predição do sinal em diferentes locais *indoor*. O WiGEM [Gos+11] utiliza *Gaussian MixModel* (GMM) e extrapola os máximos para estimar a localização do alvo, aprendendo assim os parâmetros de propagação do sinal. Em seguida, a intensidade do sinal em locais diferentes pode ser prevista de forma correspondente, o que faz com que o processo de levantamento de dados seja reduzido. No entanto, para que isto seja possível, é necessário o conhecimento explícito dos APs do local. Além disso, a computação inicial dos parâmetros é bastante pesada e deve, por essa razão, ser conduzido num servidor [Hyd].

Para resumir, melhorar a localização usando *fingerprints* WiFi com medições inerciais de sensores é uma direção interessante para implantação de sistemas LBS em ambientes *indoor*. Em comparação com a localização colaborativa, o esquema assistido por movimento não depende de utilizadores vizinhos e, portanto, consegue maior adaptabilidade e escalabilidade. Alguns *smartphones* e sensores emergentes, incluindo dispositivos no Google Project Tango [Gooi], têm estabelecido uma base de hardware para o esquema assistido por movimento, como capturar com precisão o movimento do utilizador. Além disso, também é importante para estudos futuros propor soluções para ter eficiência energética nas medições dos sensores e na computação da estimação da localização. Este assunto é abordado no capítulo 3 desta dissertação, onde é feito um estudo prático do desempenho de um conjunto de algoritmos.

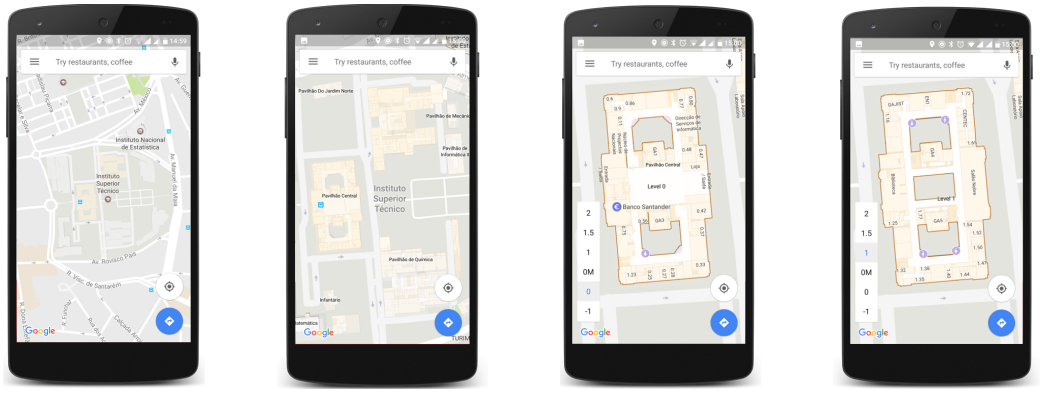
2.4 Sistemas de Localização

A grande maioria dos sistemas de estimativa de localização em ambientes *indoor* requer que o utilizador possua um dispositivo eletrónico ou *tag* que, em alguns casos, possa processar informações e enviar os resultados para um servidor que executa os algoritmos de posicionamento para avaliação e/ou processamento. Esta secção é focada na comparação entre vários sistemas de Localização Ativa, sendo apresentado o sistema que mais se assemelha com o apresentado nesta dissertação, o Google Maps. Os restantes sistemas estudados encontram-se no apêndice A.

2.4.1 Google Maps

O Google Maps é provavelmente o sistema de localização mais conhecido e utilizado atualmente. É o resultado de um conjunto de serviços como o Google Earth e Google Street View, desenvolvidos pela Google.

Tipicamente os utilizadores utilizam o Google Maps em ambiente *outdoor*. Contudo existe em [Gooj] a possibilidade de criar o mapa *indoor* de uma determinada localização. Para isto, é necessário que se submeta um mapa bastante detalhado do interior da localização. Posteriormente, quando o utilizador navegar no interior de um edifício dessa zona, existe a possibilidade de fazer zoom no mapa fazendo a transição entre ambiente *indoor-outdoor* de uma forma simples, como é possível observar na figura A.5. No modo *indoor* existe uma opção de selecionar o piso para se visualizar os diferentes níveis do edifício.



a Outdoor

b Outdoor com mais informação que (a).

c Indoor Piso 0

d Indoor Piso 1

Figura 2.9: Transição *indoor-outdoor* de visualização do Pavilhão Central do Instituto Superior Técnico.

É fornecida a possibilidade de calcular a rota dentro do edifício entre vários andares, navegar, ver fotos e panoramas em 360 graus. Há também a possibilidade de permitir notificações sobre locais próximos e o contexto envolvente. Em funcionamento *indoor*, são utilizadas as redes móveis e a rede WiFi para a orientação, obtendo-se uma precisão de posicionamento que varia de 5 a 10 metros [KS12] .

Em [Que16] são descritas as principais quatro funcionalidades da *Application Programming Interface* (API)(Interface de Programação de Aplicações) integrada no pacote *com.google.android.gms.location*, que permitem desenvolver aplicações sobre o Google Maps sem que o programador tenha que saber informações sobre as tecnologias de localização. Estas funcionalidades são:

- API Location – periodicamente obtém a localização do utilizador, gastando o mínimo possível de energia. Pode ser utilizada para realizar o tracking do dispositivo, facilitando a receção e tratamento da localização;

- API Geofencing – é possível denotar zonas, sendo que sobre essas zonas o programador pode definir ações específicas como a entrada e saída de utilizadores. Desta forma, é possível monitorizar determinadas regiões geográficas, denominadas de geofences;
- API Activity Recognition – permitem obter a atividade do utilizador, como por exemplo, se este está andar a pé ou de carro;
- API Places – permite aceder a mais informação da Google, sendo assim possível os utilizadores obterem uma experiência mais contextual, utilizando informações de lugares e nomes dos mesmos para coordenadas específicas.

2.4.2 Comparação de sistemas

A Tabela A.1 mostra uma comparação sobre as características dos Sistemas de Localização ativos analisados anteriormente. A tecnologia de posicionamento mais utilizada para áreas *indoor* é a *Wireless LAN* (WLAN) devido à infraestrutura já existente que é tipicamente aproveitada, tornando assim o sistema financeiramente mais económico. O RFID é mais adequado para ambientes densos. Tecnologias como: UWB, infravermelho, ultra-som, Bluetooth e também sistemas híbridos foram apresentados nesta secção.

Os algoritmos que apresentam melhor precisão são o Active Bats e o MIT Cricket, sendo que os que manifestam pior precisão são o Intel Place Lab, o Skyhook e o Google Maps. Analisando o nível de complexidade de cada algoritmo (coluna de custo), concluise que existe um trade-off entre precisão e custo, na medida em que quanto mais elevada é a precisão, maior é a complexidade do sistema.

Visto que nesta dissertação pretendeu-se desenvolver um sistema que consiga realizar a localização sem informação prévia nem hardware não existente nos *smartphones* usuais, este assemelha-se mais com os algoritmos com baixo custo. Em particular, o sistema desenvolvido aproxima-se mais com o Google Maps, pois são utilizadas APIs do sistema Android em ambas aplicações. Contudo, o Google Maps necessita da planta e de um mapa rádio para funcionar em ambientes *indoor*, o que não é necessário no sistema apresentado nesta dissertação.

É importante salientar que cada sistema tem a sua particularidade, mas todos necessitam de *hardware* específico e/ou um estudo e análise da região onde será implementado o sistema.

Tabela 2.4: Comparação entre sistemas de localização ativa

Sistema	Tecnologia & Métricas	Algoritmo	Precisão	Dimensões	Custo
SpotON [Hig+00]	RFID RSSI	Ad-Hoc lateration	depende do tamanho do cluster	2D	Baixo
Ubisense [SG05]	UWB TDOA e AoA	Least square	~ 0.3 m	3D	Médio
Ekahau RTLS [Krz06]	WLAN RSSI	Algoritmo probabilístico	2 - 3 m	2D	Médio
Microsoft RADAR [Bah+00; BP00]	WLAN RSSI	kNN	2 - 4.3 m	3D	Médio
AeroScout [Aer]	WiFi ToA, TDOA e RSSI	Servidor AeroScout	1 - 5 m	2D	Baixo
Intel Place Lab [Cor11; LaM+05]	Triangulação WiFi	Triangulação baseada em mapas	20 - 30 m	N/A	Baixo
PinPoint 3D-iD [RFT11]	UHF TDOA e WiFi	Algoritmos de localização híbridos	1 - 10 m	3D	Alto
Active Bats [Wan+92]	Beacons infravermelhos	Bayesian	~ 0.09 m	3D	Médio
BLIP Systems [Sys11]	Bluetooth RSSI	Lateration	0.1 - 10 m	2D	Baixo
MIT Cricket [Krz06; Pri+00]	Beacons RF	Least square	~ 0.1 m	3D	Alto
Skyhook [Sky11]	WLAN, *GNSS* e redes móveis	Algoritmos de localização híbridos	10 - 30 m	N/A	Baixo
Google Maps [Gooj]	Redes móveis e WiFi	Least square	5 - 10 m	3D	Baixo

Capítulo 3

Sistema Desenvolvido

Este capítulo apresenta uma visão geral sobre a arquitetura do sistema, especifica os módulos desenvolvidos e explica a abordagem utilizada na realização do projeto. Começa por descrever a arquitetura salientado os procedimentos necessários para localizar um AP, sendo estes posteriormente explicados de acordo com o modo que foram implementados na aplicação desenvolvida. Os módulos necessários são apresentados de modo a perceber a sua funcionalidade e utilidade. A utilização destes é descrita mais detalhadamente nos capítulos seguintes juntamente com a avaliação de desempenho.

Na parte final do capítulo, estudam-se métodos para estimar o movimento do utilizador (Odometria), apresentando-se funcionalidades de hardware e software relacionadas.

3.1 Arquitetura do Sistema

O sistema foi projetado para localizar a partir de um terminal móvel entidades que emitem um sinal WiFi. Esta abordagem encontra-se ilustrada na figura 3.1. Resumidamente, os valores de RSS medidos são armazenados e associados à localização do terminal, à medida que o terminal se move. Como é possível observar, neste cenário é fácil descobrir a localização do AP com trilateração. Contudo as medições RSS incluem ruído, pelo que podem existir situações em que existem infinitas intersecções ou nenhuma. Para estes casos é necessário usar algoritmos mais sofisticadas de multilateração, que serão aprofundados no capítulo 4. A localização dos APs selecionados é obtida executando um algoritmo de multilateração usando os valores de RSS do terminal exclusivamente (modo não cooperativo) ou uma combinação de medições de vários terminais (modo cooperativo).

O componente principal é o software executado no terminal, que implementa os módulos representados na figura 3.2. Neste sistema as setas a amarelo representam os eventos e as setas a preto simbolizam informação que está permanentemente disponível para os

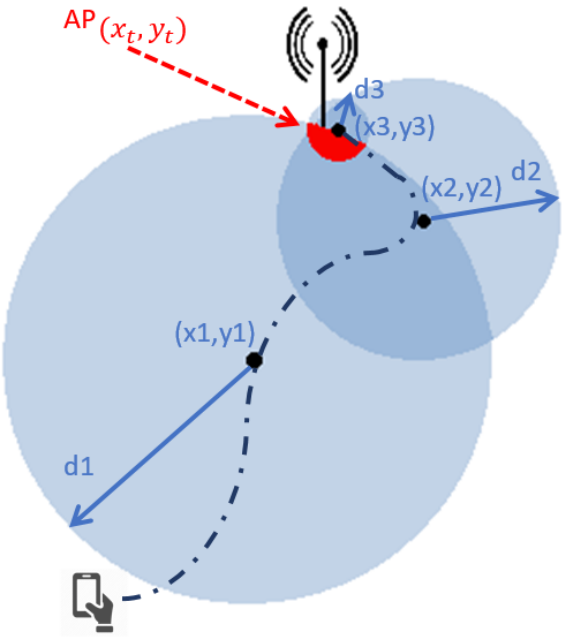


Figura 3.1: Esquema de localização.

módulos seguintes. O módulo de localização é capaz de alternar entre um modo *outdoor* e um modo *indoor*. No modo *outdoor*, a localização usa o serviço de localização do Android, que combina sinais GNSS, WiFi e celulares para estimar a localização. No modo *indoor*, é usado o módulo de odometria desenvolvido para rastrear a posição relativa do terminal a partir da última posição global conhecida. O módulo Wi-Fi *scanner* é executado em simultâneo, registando os valores de RSS usando a biblioteca *android.net.wifi* do Android. No modo não cooperativo, os vetores são alimentados ao algoritmo de multilateração, implementado no módulo FindAP, que retorna a localização do AP selecionado. Para o modo cooperativo, os vetores de RSS são enviados para um servidor, que realiza o cálculo e envia as atualizações para os terminais.

Resumidamente, o sistema é composto por 7 módulos:

- **Estimador de orientação:** Calcula a orientação do dispositivo e disponibiliza-a para o resto do sistema. Para isso, utiliza um grupo de sensores do *smartphone* (bússola, acelerómetro, giroscópio, entre outros) para conseguir detectar qualquer movimento que influencie a orientação. Com estes sensores é possível fazer algumas combinações, tornando assim o sistema mais robusto, pois torna possível que o *smartphone* não contenha um sensor e consiga correr a aplicação. As 5 combinações implementadas encontra-se detalhadas na secção 3.2.2.

- **Detector de passos:** O módulo de detecção de passos gera um evento sempre que considerar que o utilizador realizou um passo. Foram desenvolvidos 3 métodos de detecção de passos que são apresentados na secção 3.2.1.

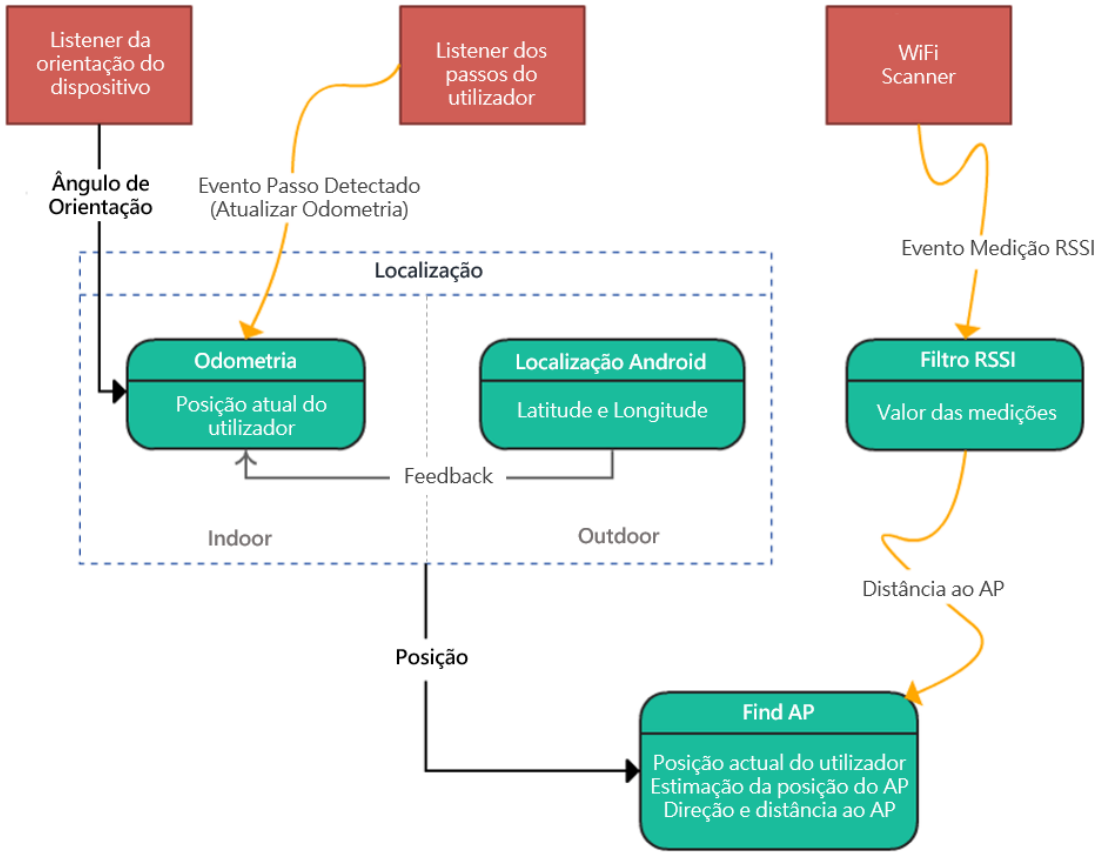


Figura 3.2: Esquema geral utilizado para estimar a localização de um emissor.

- **WiFi Scanner:** Analisa o meio WiFi para obter diversas informações sobre todos os emissores, como por exemplo o SSID, a frequência central e a RSS. Esta análise é realizada periodicamente, e o período é definido pelo utilizador.

- **Localização Android:** Em regime *outdoor*, é utilizado a API *FusedLocation* do Android para localizar o *smartphone*. Esta utiliza os sinais GNSS, WiFi e o Cell-ID. Podem ser utilizadas várias abordagens, sendo a decisão sobre quais tecnologias usar e confiar uma questão de *trade-offs* (troca) entre precisão, velocidade e eficiência de bateria. Não são realizados pedidos de atualização intermédios. No modo de melhor precisão apenas são actualizadas as coordenadas quando a interface *Location* gera eventos de mudança de localização no modo de melhor precisão [Gooe].

- **Odometria:** Este módulo utiliza a informação proveniente do estimador de orientação e do detector de passos. Sempre que o utilizador realiza um passo, o nó de Odometria atualiza as coordenadas do utilizador. O nó de Odometria é utilizado em ambientes indoor fornecendo a posição do dispositivo.

- **Filtro RSS:** Realiza uma filtragem do sinal recebido, obtido através do WiFi Scanner e caso seja necessário, converte potência num valor aproximado de distância.

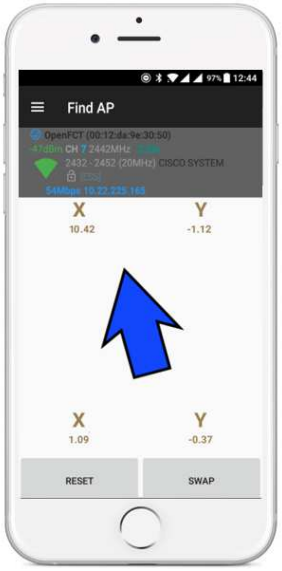


Figura 3.3: Interface Find AP

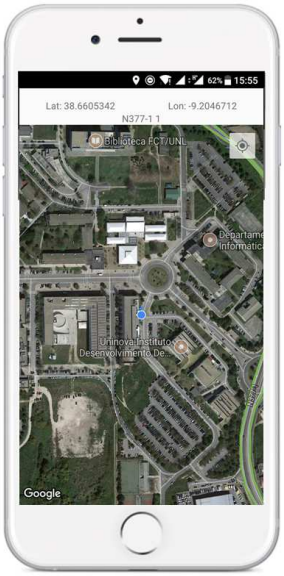


Figura 3.4: Interface Mapa

- **Find AP:** Neste módulo é realizada a estimação de emissores. Para que isto seja possível, sempre que o módulo de filtragem fornece informação ao Find AP, este gera um ponto para o qual tem informação da posição actual do utilizador através do nó de Odometria, e das distâncias a que todos os emissores se encontram. Caso existam três ou mais pontos, é aplicado um método de trilateração ou multilateração sobre esta informação, tornando possível estimar a localização de todos os emissores em redor do utilizador. No capítulo 4 analisam-se os métodos tradicionais e um novo método de estimação desenvolvido, o Pedro-Tomic.

A aplicação protótipo foi implementada em Android[1] para o modo não cooperativo. A aplicação oferece dois mecanismos de visualização para a localização do AP: ela pode fornecer uma seta que aponta em direção ao AP (figura 3.3); ou, pode fornecer a localização dos APs num mapa (figura 3.4). Esta aplicação foi desenvolvida para *smartphones*. No entanto, pode ser facilmente estendida para um cenário de *Unmanned Autonomous Vehicle* (UAV)s (veículos autônomos não tripulados): o módulo FindAP pode enviar as localizações dos APs diretamente para uma unidade de condução autónoma dentro do veículo e o módulo de localização pode receber a localização da unidade de odometria do veículo (que pode usar sensores ligados às rodas).

O esquema cooperativo apresenta vários desafios, como a integração, diversidade de sensores entre dispositivos, comunicação, entre outros. Os detalhes sobre o funcionamento dos dois modos cooperativos (centralizado e distribuído) são apresentados no capítulo 5.

Para fins de teste, foram criados métodos na aplicação que permitem exportar as medidas de RSS, tempo, odometria/localização do Android, sendo assim possível avaliar

[1] O código da aplicação pode ser encontrado em `https://github.com/dario-pedro/wifi_finder`.

os múltiplos algoritmos com dados reais.

Na secção 3.2 é apresentado o funcionamento detalhado do sistema de odometria, que fornece o posicionamento do utilizador em ambientes *indoor*. O cálculo da localização remota é descrito nos capítulos 4 e 5, respetivamente para sistemas centralizados e distribuídos.

3.2 Odometria num smartphone

A Odometria permite que se tenha uma estimação da posição relativa do utilizador. Tal como foi apresentado na secção 2.3.3, para a realização da odometria 2D num *smartphone* foram desenvolvidos dois módulos: o primeiro foi desenhado para informar o odómetro quando o utilizador dá um passo; o segundo tem como função determinar a orientação do dispositivo e disponibilizar essa informação ao nó que realiza a odometria.

Sempre que o utilizador dá um passo, o primeiro módulo faz o odómetro atualizar a posição do utilizador. O comprimento do passo do utilizador é parametrizável nas preferências da aplicação, ou pode ser estimado. A posição do utilizador é atualizada de acordo com a equação (3.1), onde s é o comprimento do passo e θ a orientação do *smartphone*.

$$\begin{cases} x_n = x_{n-1} + s * \cos(\theta) \\ y_n = y_{n-1} + s * \sin(\theta) \end{cases} . \tag{3.1}$$

A abordagem proposta dá origem a trajetórias como a presente na figura 3.5, que representa um percurso em que foram detetados 4 passos. No momento em que cada passo foi realizado, o dispositivo apresentava os ângulos de orientação $\theta_1 = 30°, \theta_2 = 60°, \theta_3 = 80°, \theta_4 = 110°$. Foi considerado que cada passo s tem $0.5m$ de comprimento e que o movimento se iniciou no ponto $(0, 0)$.

Neste capítulo são analisadas várias técnicas para realizar os dois módulos apresentados anteriormente. Cada técnica foi testada individualmente usando funcionalidades de teste criadas na aplicação desenvolvida. Os testes foram realizados utilizando dois terminais móveis: um *OnePlus 2* a correr o Android 6.0.1 na versão OxygenOS e um *Sony XPERIA Z1 Compact* a correr o Android 5.1.1 na versão standard da Sony.

Estes testes foram executados no terceiro piso do Departamento de Engenharia Electrótecnica da faculdade, representado na figura 3.6. A zona sombreada a cinzento representa o percurso com 16,5 m percorrido nos testes da aplicação. Na figura 3.7 está representado com mais detalhe o percurso. Durante os testes foram percorridas 34 lajes realizando 33 passos com exatamente 50 cm (o tamanho de cada laje). Desta forma, o erro medido nos testes não é devido à imprecisão no tamanho do passo, mas a erros na deteção de passos e/ou da orientação do terminal.

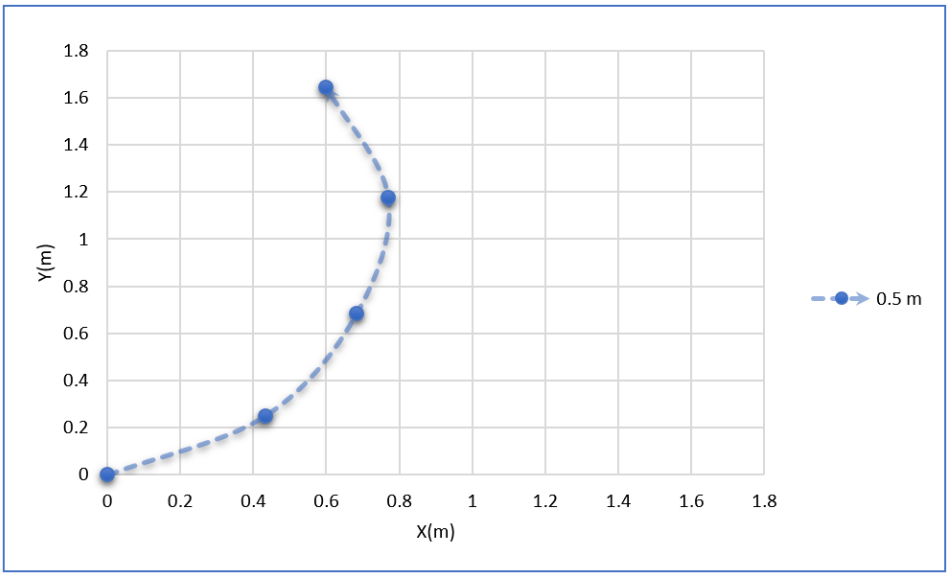


Figura 3.5: Gráfico da Odometria do *smartphone*. Exemplo de percurso com 4 passos de 5 cm , e orientações de 30º, 60º, 80º e 110º.

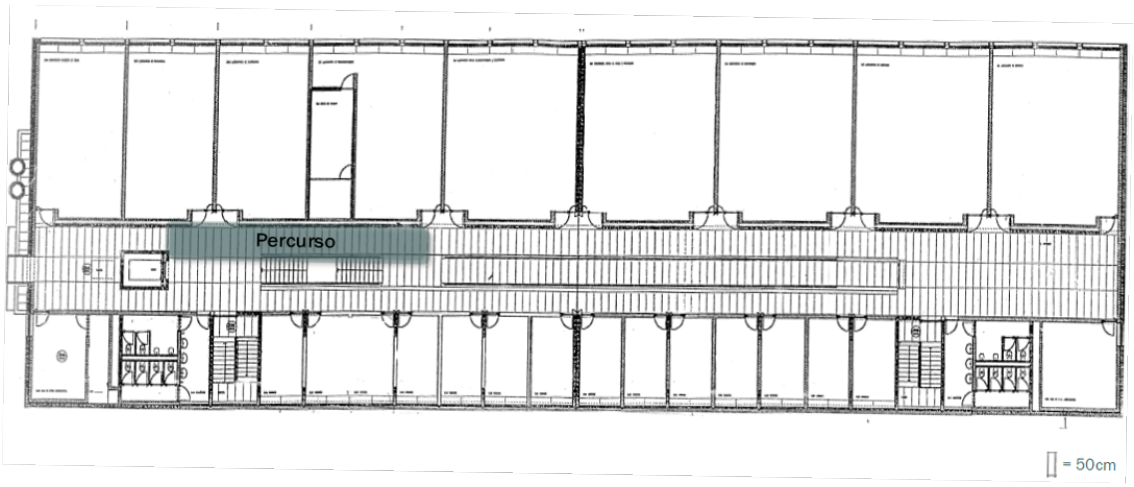


Figura 3.6: Planta do terceiro piso do Departamento de Engenharia Electrótecnica.

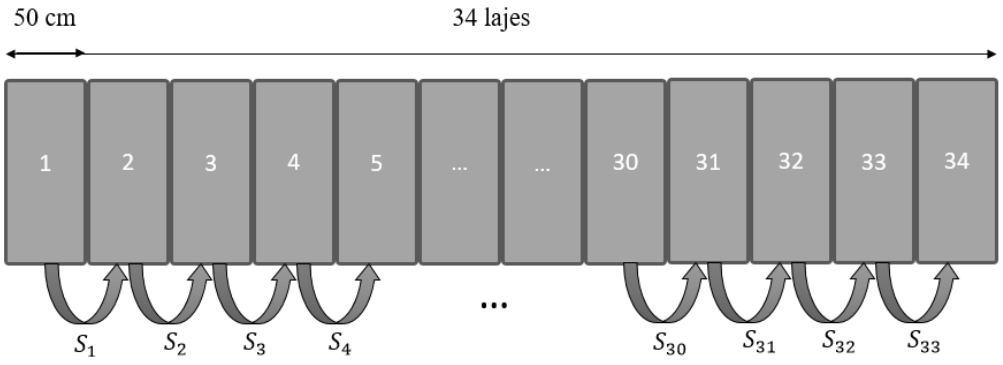


Figura 3.7: Percurso de 34 lajes utilizado para testes.

3.2.1 Deteção de Passos

Como estudado na secção 2.3.3, os MEMS presentes nos *smartphones* permitem que sejam criados pedómetros para a localização assistida por movimento.

Para a detecção do movimento do utilizador, tipicamente são desenvolvidos algoritmos com base na informação do acelerómetro. A partir do Android KitKat foram adicionados mais alguns módulos de software associados a componentes de hardware à lista de APIs que permitem a detecção de passos. Apesar de alguns *smartphones* ainda não conterem estes sensores de passos, no futuro, isso irá gradualmente tornar-se habitual. Para efeitos de teste foram implementados três métodos que permitem estimar o movimento do utilizador: baseados em acelerómetro, sensor SD e sensor SC, que são abordados nas subsecções abaixo.

3.2.1.1 Acelerómetro

O Acelerómetro é dos sensores mais antigos presentes na API Android. Quando o dispositivo sente uma alteração nas forças de aceleração é gerado um evento, que apresenta a aceleração ao longo dos eixos X, Y e Z (incluindo a gravidade), na orientação apresentada na figura 3.8. Na figura 3.9 está representado um conjunto de amostras de aceleração ao longo de 4 segundos nas três componentes, obtido num OnePlus 2. Ao longo destes 4 segundos foram percorridos 5 passos, para os quais o *smartphone* gerou 1281 eventos, ou seja, com uma frequência de aproximadamente $320Hz$. Os passos foram efetuados com o ecrã para cima, e por essa razão, a componente em Z foi claramente mais afetada pela gravidade e pelo movimento vertical durante os passos, sendo fácil distinguir os passos. Caso o utilizador se deslocasse com o telemóvel numa posição intermédia, as variações ficariam distribuídas entre as restantes componentes, o que tornaria o processo de selecção do eixo complexo.

Uma alternativa para detetar passos quando o utilizador se desloca com o telemóvel numa orientação desconhecida, é obter um único valor de aceleração do dispositivo A_d a partir das três componentes da aceleração F_s e retirar a influência da força da gravidade g. Nesta dissertação foi adoptada a soma linear das várias componentes, usando

$$A_d = -g + \sum F_s \ . \tag{3.2}$$

A partir dos pontos A_d é possível observar as variações de direção de aceleração (os máximos e mínimos locais). Consideremos p_{inf} um pico inferior, p_{sup} um pico superior, p_{dif} a diferença entre picos, isto é, $p_{dif} = \left|p_{sup} - p_{inf}\right|$, e u_p o valor da diferença entre picos no último passo detectado (p_{dif} anterior), inicializado a 0. Define-se que um utilizador dá um passo quando se verificam as quatro condições:

- Não ter sido detectado um passo nos últimos t_{min} ms. Para efeitos de teste foi considerado que $t_{min} = 500$ ms.

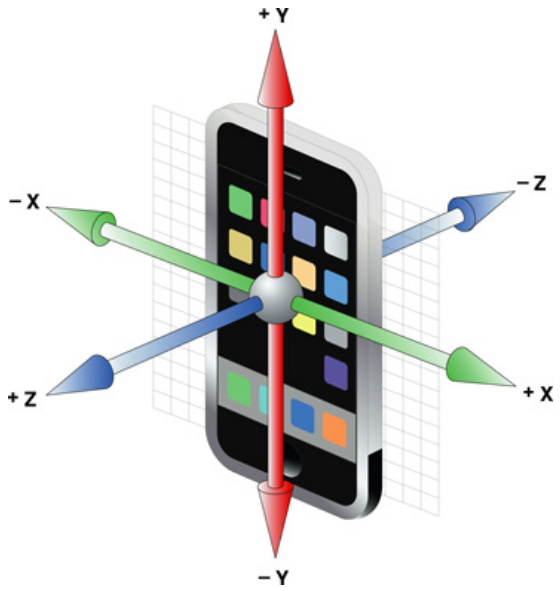


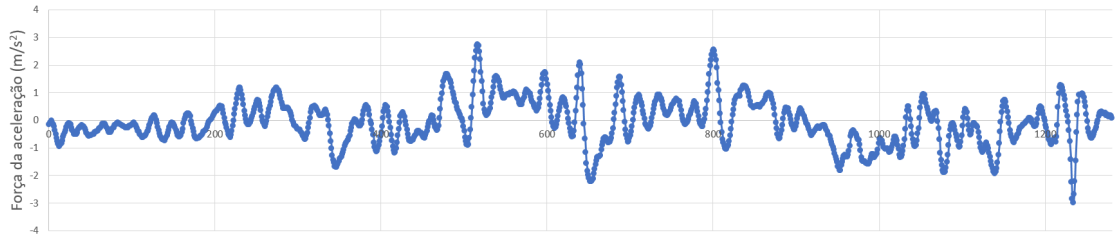
Figura 3.8: Sistema de coordenadas (relativo a um dispositivo) que é usado pela API do sensor [Goog] .

- Para se filtrar as baixas frequências, é verificado se $p_{dif} > \frac{2}{3}u_p$.
- De forma semelhante à condição anterior, para remover as altas frequências, é verificado se $u_p > \frac{1}{3}p_{dif}$.
- O último passo detectado não ter sido num pico exatamente anterior com a mesma direção, requerendo que exista uma alternação de direção para se detetar o primeiro passo.

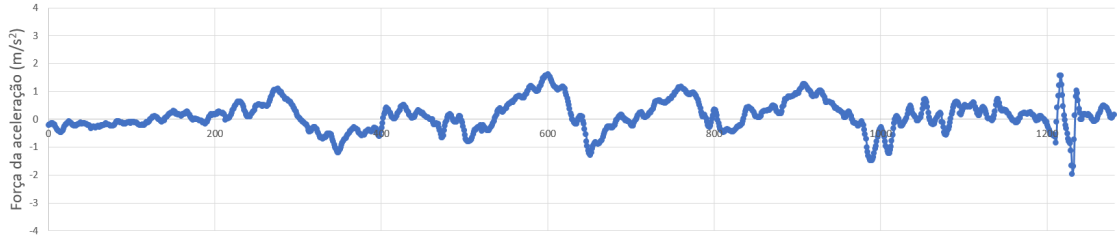
Este método de avaliação foi criado com o objectivo de detectar passos num sistema em tempo real, sendo assim realizadas operações com baixo custo computacional, mesmo que se prescinda de alguma precisão. Foi inspirado em trabalhos [DM; Goof] nos quais se atinge maior precisão com recurso à derivação dos dados obtidos e/ou filtros passa-alto e passa-baixo, mas introduzem algum atraso, o que faz com que se obtenha piores resultados em tempo real.

3.2.1.2 Sensor Step Detector

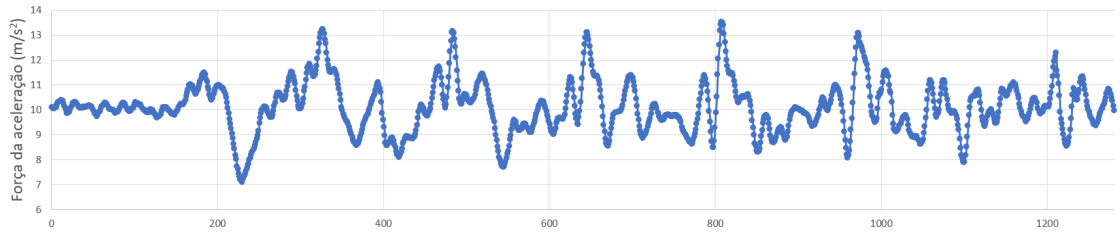
Um sensor do tipo SD serve para detectar cada passo individual assim que é capturado, por exemplo, para executar *Dead Reckoning*. O SD utiliza um filtro de Kalman [Goof] para comparar padrões de passos, o que faz com que exista algum atraso na detecção de cada passo, bem como uma necessidade de calibração. Caso não exista a calibração inicial, podem ocorrer erros significativos, como os ilustrados na figura 3.10b com um sensor do tipo SD não calibrado que não foi capaz de reconhecer a maioria dos passos.



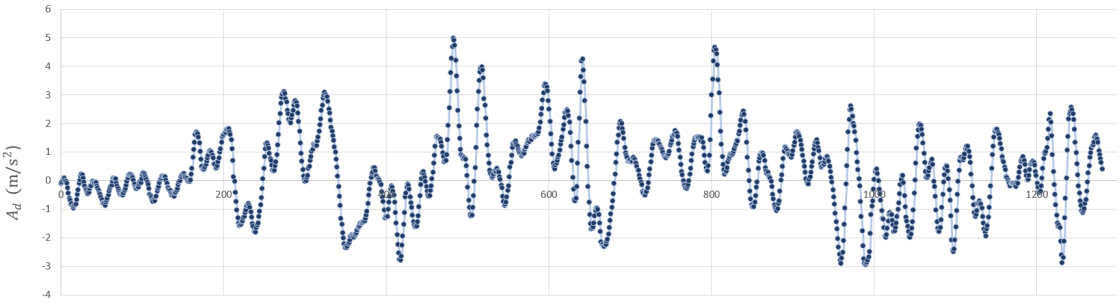
a Gráfico da aceleração ao longo do eixo X.



b Gráfico da aceleração ao longo do eixo Y.



c Gráfico da aceleração ao longo do eixo Z.



d Gráfico da aceleração A_d.

Figura 3.9: Fragmento contador de passos da aplicação WiFi Finder.

3.2.1.3 Sensor *Step Counter*

O sensor do tipo SC existe apenas nas versões mais recentes de Android. Este sensor é uma combinação de *hardware* e *software* [Goof] que sempre que gera um evento, retorna o número de passos caminhados pelo utilizador desde a última reinicialização da aplicação juntamente com o *timestamp* do último passo antes do evento ter sido disparado. Tendo em conta que grande parte do processamento deste sensor é realizado em hardware, não gasta muita energia [Nag]. Mesmo em *background*, este sensor continua a trabalhar e a contar o número de passos, e quando se volta à aplicação é disparado um evento com todos os passos realizados. Só pára quando é cancelado o registo no sensor.

De forma semelhante ao SD, é utilizado um filtro de Kalman, mas que compara grupos de passos, como se analisasse um padrão de caminhada do utilizador. Desta forma, é possível eliminar alguns movimentos que embora se assemelhem a um passo, não o são.

3.2.1.4 Comparação entre métodos

Para a validação da robustez dos três métodos de detecção de passos, foram desenvolvidas classes individuais que escutam cada um dos sensores, realizam o processamento devido, e no final geram eventos do tipo *passo*. Utilizando as classes desenvolvidas, foram realizados múltiplos testes no cenário descrito acima. Todos os métodos apresentaram resultados precisos na estimação do número de passos, sendo que em diversos testes, os três sensores detectaram que o utilizador realizou 33 passos.

As principais diferenças entre os métodos são:

1. Precisão : Todos os métodos mostraram resultados semelhantes neste ponto, contudo, em camainhadas com mais do que ≈ 100 passos o SD mostrou-se relativamente mais preciso que os restantes, sendo o acelerómetro o menos preciso. O SD e o SC também são mais robustos na filtragem de movimentos semelhantes a passos. O acelerómetro na figura 3.10a apresenta 1 passo a mais do que o realizado, pois considerou que o movimento de paragem foi um passo.

2. Tempo de Resposta : Este é o ponto mais diferenciador. Como é possível observar na figura 3.10a, foram realizados 33 passos e tanto o SD, como o SC não registaram o último passo .

3. *Delay* inicial: Os novos sensores Android usam filtros de Kalman, o que faz com que inicialmente tenham um regime transitório onde não enviam informações sobre passos. Este efeito é geralmente ultrapassado ao fim de cerca de 20 passos. Este efeito pode ser visualizado na figura 3.10b, onde o SD e o SC ainda não conseguiram descodificar o padrão do passo do utilizador, e encontram-se bastante atrasados face ao acelerómetro.

4. *Reset*: Para o SD teve que ser implementado um sistema para conseguir obter uma contagem parcial de passos, guardando o valor de passos até um momento, pois este sensor reporta sempre todos os passos, como está ilustrado na figura 3.10c.

Tendo em conta que para a aplicação em causa (Odometria), é necessário um tempo de resposta muito rápido (caso contrário a orientação do dispositivo θ pode ter mudado significativamente) e a precisão dos métodos ser similar, o acelerómetro foi o sensor escolhido para a realização da aplicação e é usado nos restantes testes ao longo da dissertação.

a Funcionamento Normal.

b Funcionamento sem treino inicial.

c Funcionamento normal sem reset do SC.

Figura 3.10: Testes de detecção de passos realizados na aplicação WiFi Finder num OnePlus2, sendo demonstrado os resultados do acelerómetro, do SD e do SC.

De um modo geral, o acelerómetro é um bom sensor para aplicações em que é necessário estar constantemente a observar o movimento do dispositivo. Quase todos os dispositivos Android têm um acelerómetro, e ele usa cerca de 10 vezes menos energia do que os outros sensores de movimento [Goof]. Uma desvantagem é que dependendo da utilização, é normalmente necessário implementar filtros passa-baixo e/ou passa-alto para eliminar as forças gravitacionais e reduzir o ruído. Em comparação com os métodos estudados no capítulo 2, o algoritmo desenvolvido para a deteção de passos utilizando o acelerómetro é computacionalmente mais leve, visto que os filtros de partículas requerem processamento adicional.

3.2.2 Deteção da Orientação

Os *smartphones* modernos apresentam um vasto número de sensores que podem ser utilizados para obter a orientação do dispositivo. A maioria dos dispositivos Android tem um acelerómetro, e muitos incluem um giroscópio. A disponibilidade de sensores baseados

em software é mais variável pois eles geralmente dependem de um ou mais sensores de hardware para gerarem dados. Dependendo do dispositivo, esses sensores baseados em software podem produzir dados do acelerómetro e bússola, ou do giroscópio.

Todos os sensores de movimento retornam matrizes multidimensionais de valores para cada evento do tipo *SensorEvent*. Por exemplo, durante um evento dum sensor, o acelerómetro retorna dados de força de aceleração para os eixos de coordenadas, e o giroscópio retorna os dados de velocidade de rotação para os três eixos de coordenadas. Esses valores são retornados numa matriz com valores do tipo *float* juntamente com outros parâmetros do *SensorEvent*.

Com base no estudo dos sensores MEMS apresentado no capítulo 2 foram testados 5 conjuntos de sensores (a partir da biblioteca em [Pac16]): acelerómetro e bússola, Sensor de Gravidade e bússola, Giroscópio com calibração, Sensor RV sozinho e o Sensor RV fundido com o Giroscópio com calibração.

Todos os esquemas de orientação foram testados com a aplicação no cenário descrito no início deste capítulo. Neste cenário foram realizados percursos de ida e volta repetidas vezes com ambos os *smartphones*. Para cada percurso foi exportado um conjunto de medições, que inclui: a posição relativa estimada, *timestamp* da realização de cada passo, e orientação do dispositivo no momento em que o passo foi detectado. As secções seguintes apresentam as medições efetuadas para cada tipo de sensor.

3.2.2.1 Acelerómetro e Bússola

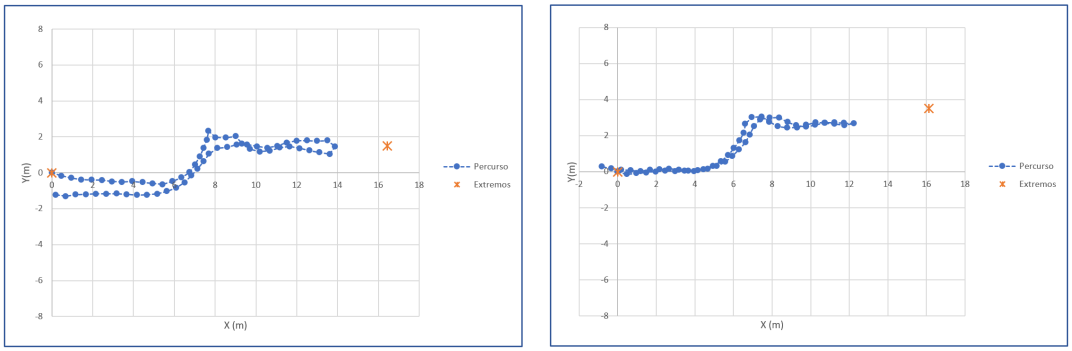
Este conjunto usa os sensores mais simples e eficientes a nível energético. Para a sua utilização basta apenas guardar os eventos de ambos os sensores, e fundir a informação mais recente de ambos utilizando a Matriz de Rotação do *Sensor Manager* do Android [Goof].

Na figura 3.11 é possível observar um dos testes realizados utilizando o Acelerómetro e a bússola para a obtenção da orientação de cada um dos dispositivos. É de notar que em $x \approx 8m$ a orientação mudou significativamente, devido à bússola ter sido afetada por variações do campo magnético.

3.2.2.2 Sensor de Gravidade e Bússola

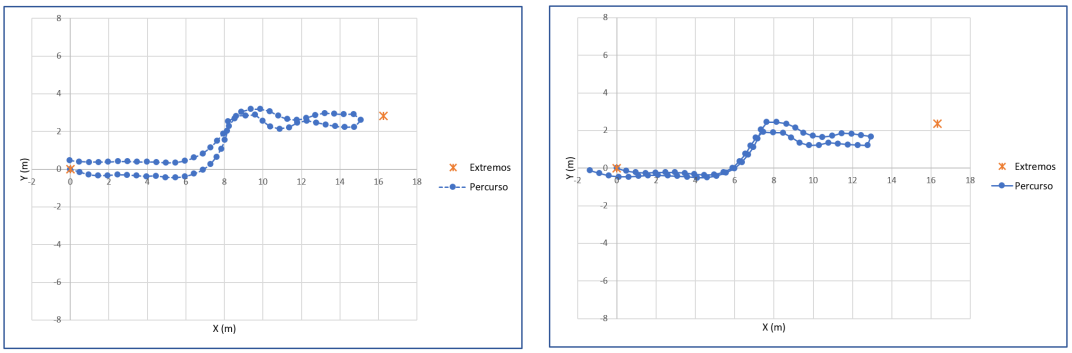
O Sensor de Gravidade e a bússola funcionam de forma muito semelhante ao conjunto do Acelerómetro e bússola [Gooh]. A vantagem de ter ambos os módulos implementados é que em diversos *smartphones*, os fabricantes optam por disponibilizar apenas o sensor de Gravidade ou o Acelerómetro.

Como seria de esperar, os resultados presentes na figura 3.12 são muito semelhantes aos representados na figura 3.11.



a Teste realizado no OnePlus2. b Teste realizado no Sony XPERIA.

Figura 3.11: Gráficos da odometria utilizando acelerómetro e bússola.



a Teste realizado no OnePlus2. b Teste realizado no Sony XPERIA.

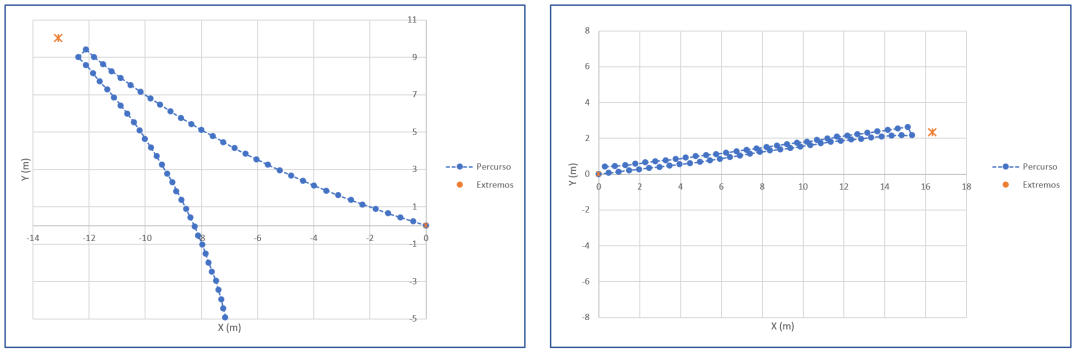
Figura 3.12: Gráficos da odometria utilizando o Sensor de Gravidade e bússola.

3.2.2.3 Giroscópio com calibração

O Giroscópio é um sensor que tem vindo a ganhar mais popularidade [Har]. É o sensor que apresenta mais estabilidade e não sofre de problemas de efeitos de campos magnéticos como a bússola. Estes factores têm levado a que a maioria dos produtores de jogos para *smartphones*, baseiem a orientação dos dispositivos nestes sensores, sendo um dos casos mais conhecidos o *Pokemon GO*.

O giroscópio funciona com base no princípio da inércia. O eixo em rotação tem um efeito de memória que guarda uma direção fixa em relação ao círculo máximo, sendo possível obter as coordenadas de rotação. Contudo, para ser possível comparar com os restantes conjuntos é necessário que o eixo de orientação absoluto seja o mesmo. Nestas medições usou-se a informação proveniente das primeiras leituras do acelerómetro e da bússola para ter o valor ponto inicial, ao qual são aplicadas as coordenadas de rotação obtidas pelo giroscópio. Este processo de referenciação é descrito na dissertação como calibração do giroscópio.

Os resultados observados na figura 3.13 demonstram que a qualidade do giroscópio influencia muito este algoritmo, sendo a precisão muito dependente do *hardware*. No caso



a Teste realizado no OnePlus2. b Teste realizado no Sony XPERIA.

Figura 3.13: Gráficos da odometria utilizando giroscópio com calibração.

das medições realizadas no Sony XPERIA, mostradas na figura 3.13b, os resultados foram muito mais precisos do que no caso do OnePlus 2, na figura 3.13a.

Em ambos os casos, foi possível observar uma muito menor variação do ângulo de orientação, o que torna a variação do gráfico de odometria muito mais suave.

3.2.2.4 Sensor Rotation Vector

O RV é um sensor de *software* [Goof] que usa a informação do acelerómetro, do giroscópio e da bússola. A orientação do dispositivo é representada como uma combinação de um ângulo e eixos, no qual o dispositivo está rodado de um ângulo θ em torno de um eixo (x, y ou z), como é possível observar na figura 3.14.

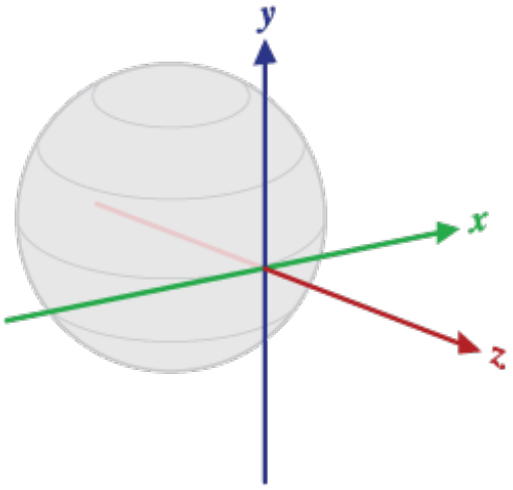


Figura 3.14: Sistema de coordenadas usado pelo sensor de vetor de rotação [Goog] .

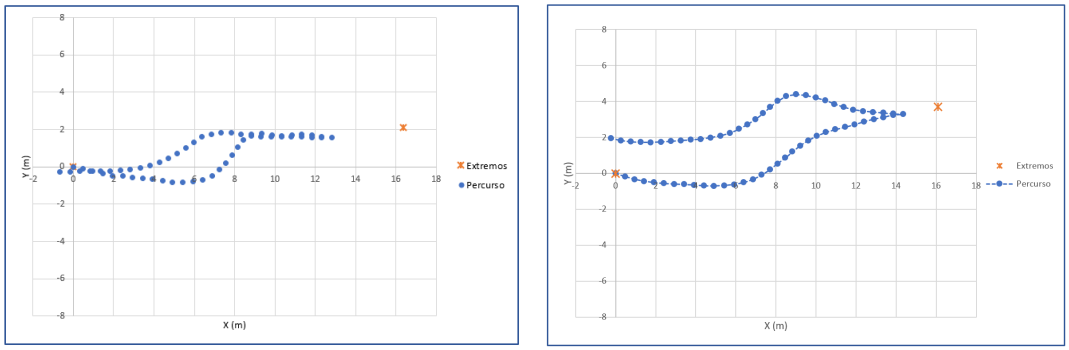
Por cada evento deste sensor, são gerados quatro valores:

$$\left[x * sin\left(\frac{\theta}{2}\right), y * sin\left(\frac{\theta}{2}\right), z * sin\left(\frac{\theta}{2}\right), cos\left(\frac{\theta}{2}\right)\right] \tag{3.3}$$

A partir destes valores, é possível obter a magnitude do vetor de rotação a partir de $sin\left(\frac{\theta}{2}\right)$, e a direção do vetor de rotação a partir dos eixos. A orientação dos eixos pode ser definida como:

1. **X** : Coordenada que é tangencial ao solo na localização atual do dispositivo e aponta aproximadamente para Este.
2. **Y** : Tangente ao solo na posição atual do dispositivo e aponta para o pólo Norte.
3. **Z** : Coordenada perpendicular ao plano da terra.

Utilizando esta informação e as funções do *SensorManager* é possível obter o *Roll*, *Pitch e Yaw* (RPY), de forma a ter as coordenadas no mesmo formato que os restantes grupos de sensores. Os resultados dos testes realizados estão ilustrados na figura 3.15. Podemos notar que mesmo com a combinação de múltiplos sensores, a orientação ainda sofre efeitos provenientes de campos magnéticos. Porém, consegue obter melhores resultados que os conjuntos que utilizam um sensor inercial e a bússola.



a Teste realizado no OnePlus2. b Teste realizado no Sony XPERIA.

Figura 3.15: Gráficos da odometria utilizando o Sensor RV.

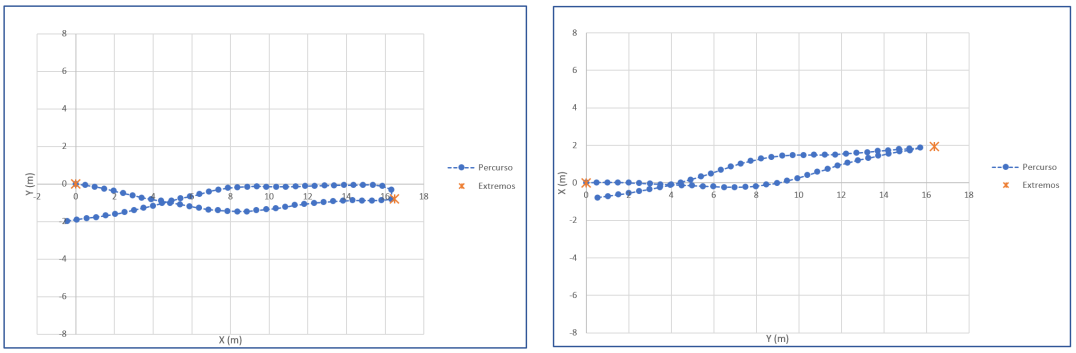
3.2.2.5 Sensor RV e Giroscópio com calibração

A junção da informação proveniente do Sensor RV e do Giroscópio foi realizada com recurso à biblioteca [Pac16]. Com estes dois módulos fundidos é possível melhorar a precisão face a utilizar o Sensor RV isoladamente. Visto que a biblioteca utiliza um filtro de Kalman no processo da fusão, existe um efeito transitório inicial que requer que o utilizador realize alguns movimentos até que o comportamento transitório do filtro seja ultrapassado.

O Giroscópio aumenta a estabilidade do conjunto e tendo em conta que são utilizados diversos sensores, também é mais fácil corrigir erros temporários provenientes de um sensor isolado, como os presentes na figura 3.13a.

Na figura 3.16 podemos observar os resultados dos testes realizados em ambos os *smartphones* utilizando esta combinação. Comparado as figuras 3.15 e 3.15 podemos

notar que combinando o RV com o giroscópio calibrado, o efeito dos campos magnéticos sobre a orientação é minimizado



a Teste realizado no OnePlus2. b Teste realizado no Sony XPERIA.

Figura 3.16: Gráficos da odometria utilizando o Sensor RV e Giroscópio com calibração.

Como se pode observar na figura 3.16, os resultados foram bastante positivos em ambos os *smartphones*. Tendo em conta que esta combinação de sensores agrega praticamente todos os sensores disponíveis para a obtenção da orientação, é mais provável que os resultados sejam menos sensíveis às especificidades de cada dispositivo.

É de notar que quando se utiliza um filtro de Kalman é importante esperar um tempo inicial para que o filtro estabilize. Caso esse tempo não seja cumprido podem ocorrer erros significativos, como é possível observar na figura 3.17, obtida com medições realizadas com um telemóvel OnePlus 2 acabado de iniciar.

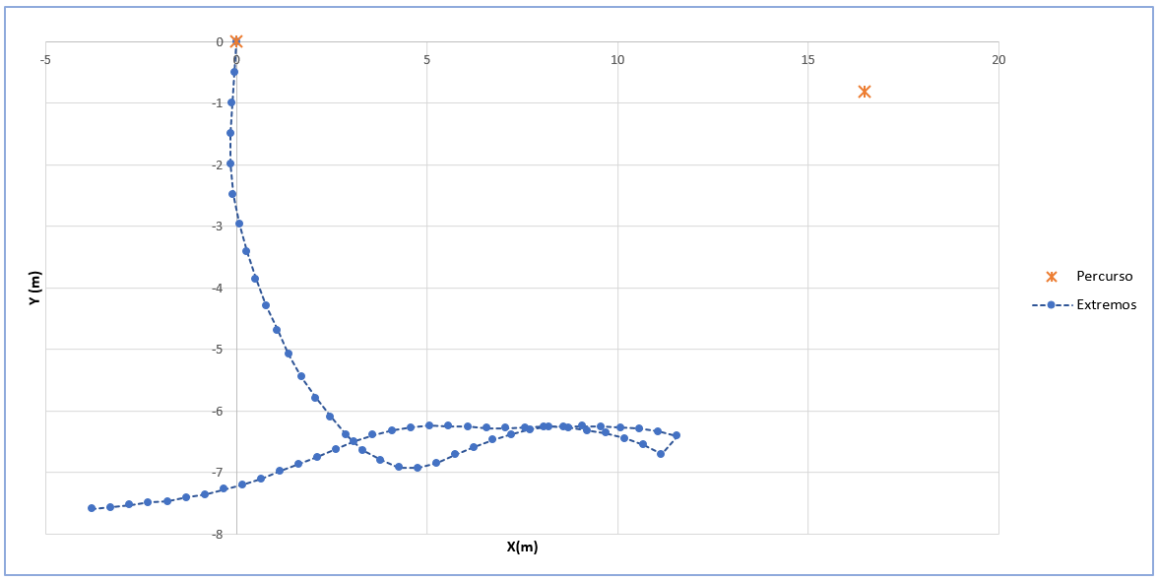


Figura 3.17: Teste de Odometria utilizando Sensor RV e Giroscópio com calibração sem realizar caminhada inicial para inicializar o filtro de Kalman.

3.2.2.6 Comparações entre métodos

Todos os métodos apresentados anteriormente têm pontos fracos e fortes. Quer seja a nível de simplicidade de implementação, gasto energético ou até mesmo compatibilidade com modelos de terminal diferentes, cada um é mais adaptável a uma determinada situação. Para o efeito desta dissertação, o fator mais importante é a precisão.

Para analisar a precisão dos métodos apresentados, foram medidas as distâncias entre o valor estimado por odometria e o valor real, no ponto final da caminhada e no ponto mais distante. Para esse efeito, consideremos que o ponto inicial e final é P_f e o ponto mais distante (na última laje) P_e. Quando a aplicação é testada com vários dispositivos recolhendo-se N amostras, para calcular o erro da distância num ponto, foi utilizada a distância Euclidiana:

$$Desvio_{ponto} = \frac{\sum_{i=0}^{N} \|P_{real} - P_{est}\|_i}{N} . \tag{3.4}$$

A métrica final utilizada para comparação dos algoritmos foi:

$$Desvio_{Extremidades} = \frac{\sum_{i=0}^{N} \left(\left\|P_{e_{real}} - P_{e_{est}}\right\|_i + \left\|P_{f_{real}} - P_{f_{est}}\right\|_i \right)}{2N} . \tag{3.5}$$

Na tabela 3.1 é possível observar os resultados dos diversos métodos analisados. Como se pode observar, os métodos que utilizam filtros de Kalman precisam de treino inicial (ex. caminhar alguns passos), antes que seja possível utilizar a informação proveniente do mesmo.

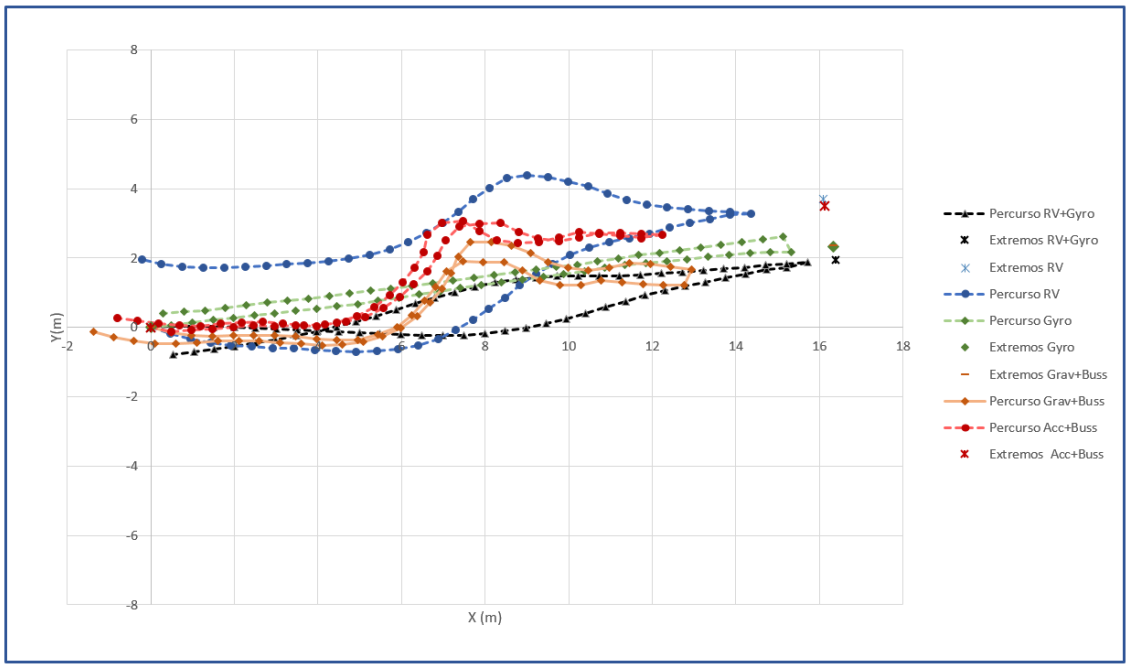
O método que apresentou melhores resultados foi o Sensor RV fundindo informações do Giroscópio, denotando um erro aproximado de 1m, o que neste cenário de testes representa 2 passos. Desta forma, para os restantes testes *indoor* da dissertação, a Odometria foi realizada utilizando o acelerómetro para obter o momento em que o utilizar deu um passo, e o Sensor RV com o Giroscópio calibrado ou o só Giroscópio calibrado (se o Sensor RV não estiver disponível) para obter a orientação do dispositivo.

Na figura 3.18 estão representados os desvios médios medidos para os vários métodos para os testes realizados com o Sony XPERIA e o OnePlus2. É possível verificar que foi viável a realização da odometria num dispositivo Android, pois para os dois modelos existiu sempre uma combinação de sensores que conseguiu resultados satisfatórios.

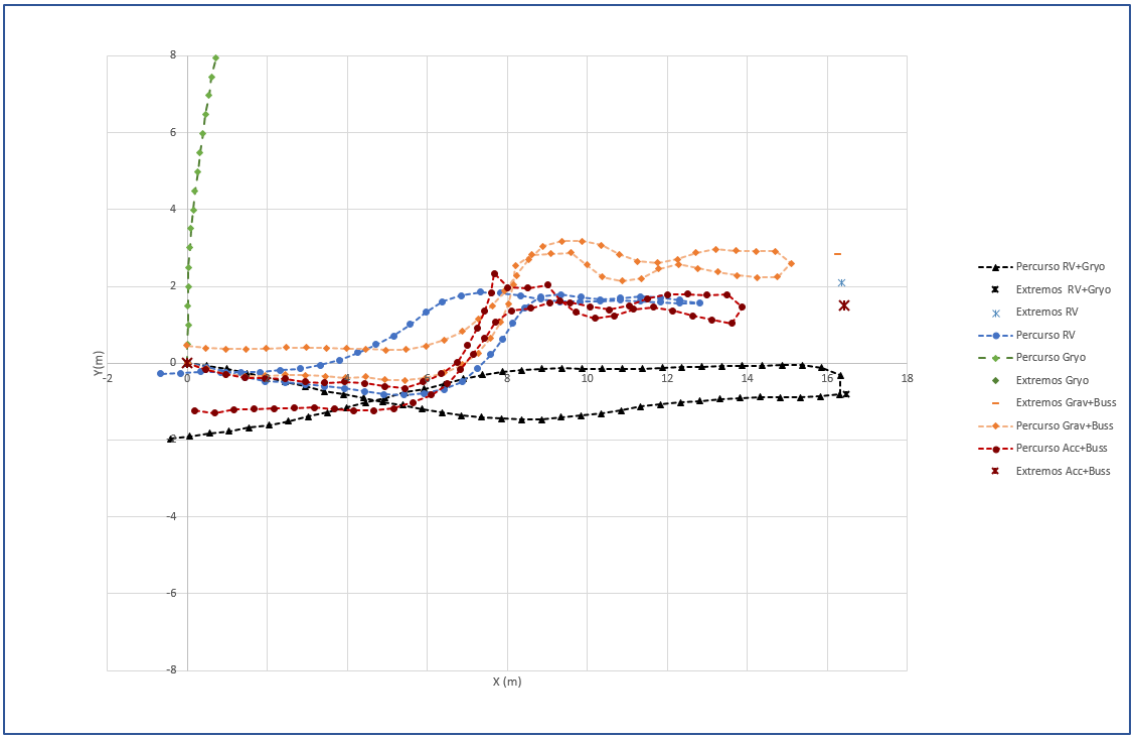
A partir destas figuras é também possível retirar algumas conclusões:

- A precisão do método utilizado é muito dependente da qualidade dos sensores. É possível observar que o giroscópio do OnePlus 2 apresenta resultados muito inferiores aos do Sony XPERIA Z1 Compact.

- O tamanho do passo é um factor muito determinante na qualidade da odometria. Para os testes apresentados nesta secção foi utilizado um comprimento de passo fixo conhecido, para este erro não influenciar os testes.



a Testes de Odometria no Sony XPERIA Z1 Compact



b Testes de Odometria no OnePlus 2.

Figura 3.18: Testes de Odometria usando diversos sensores de orientação (Rotation Vector, Giroscopio, bússola, Acelerómetro e Gravímetro).

Tabela 3.1: Comparação entre os diversos métodos de obtenção de orientação para a Odometria.

Método	Desvio no P_{es} (m)	Desvio no P_{fs} (m)	Desvio Extremidades (m)	Necessita Treino
Acelerómetro e Bússola	1.05	3.27	2.16	❌
Sensor de Gravidade e Bússola	2.32	0.91	1.62	❌
Giroscópio com calibração	0.88	4.59	2.74	✅
Sensor RV	2.92	1.34	2.13	✅
Sensor RV e Giroscópio com calibração	0.42	1.63	1.03	✅
Sensor RV e Giroscópio (sem treino)	7.45	8.49	7.97	❌

Contudo, para uma aplicação comercial, é necessário que a aplicação consiga estimar este valor. Para tal foi desenvolvido um algoritmo (apresentado na secção 3.2.4) que consegue realizar a estimação.

- Tendo em conta que os erros de odometria se propagam a cada medição, é necessário fazer *reset* (reiniciar) o odómetro periodicamente. Este *reset* pode ser realizado quando se passa por alguma posição conhecida. Para tal, basta guardar o desvio $(\Delta x, \Delta y)$, e subtrair estes desvios às medições futuras.

- Os métodos que utilizam a bússola têm a adversidade de serem influenciados por objetos magnéticos e por esta razão é recomendado evitar os mesmos, ou pelo menos fundir a bússola com outros sensores.

3.2.3 Integração dos módulos de Software

Nesta dissertação foram desenvolvidos 3 módulos de detecção de passos e 5 de orientação. Para permitir modificar e adicionar módulos, sem haver necessidade de mexer no resto do código mantendo a aplicação estável, optou-se por usar o esquema de integração representado na figura 3.19. Este apresenta apenas as principais classes, com as suas variáveis e métodos mais importantes.

Para isolar o bloco da detecção de passos criou-se uma interface que implementa o método *update()*. Este método é chamado sempre que for detectado um passo, requisitando ao módulo de Odometria que actualize as coordenadas.

É importante salientar que para ser possível obter informação dos sensores, quer os módulos de detecção da orientação, quer os módulos da detecção de passos, necessitam de implementar o método *onSensorChanged()* da classe abstrata *SensorEventListener* do Android.

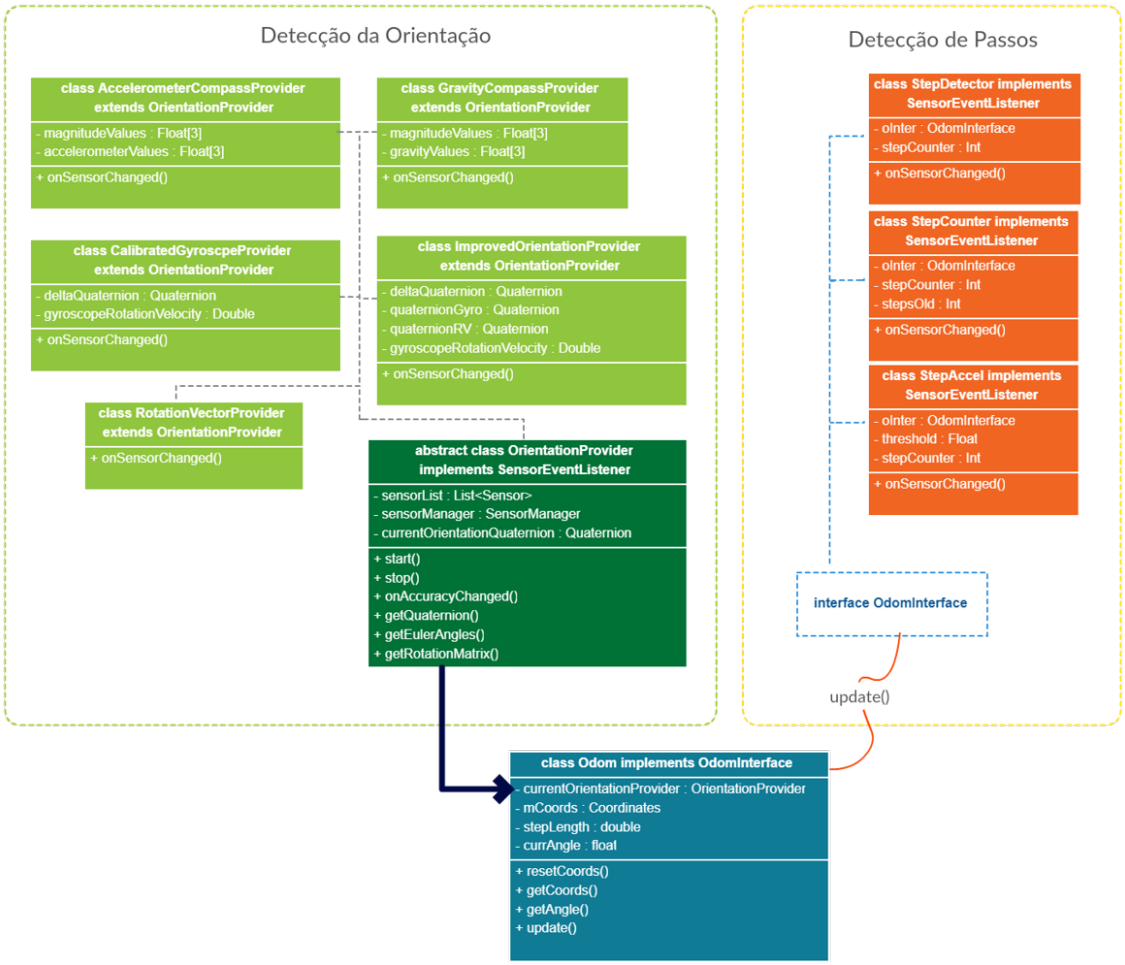


Figura 3.19: Integração dos módulos de software da odometria.

No caso do bloco da detecção da orientação, para criar uma camada de generalização entre os módulos de detecção e a classe de Odometria, decidiu-se utilizar uma classe abstrata, pois existem muitos métodos e procedimentos que são imutáveis e obrigatórios entre métodos, como por exemplo, a inicialização duma estrutura do tipo *Quaternion* e um método *getQuaternion()* que permita ao nó de Odometria obter este valor, ou mesmo estender a classe *SensorEventListener*.

A classe de Odometria implementa a interface *OdomInterface*, o que faz com que tenha implementado o método *update()*. Neste método é pedido à variável *currentOrientationProvider* o ângulo de orientação e a variável *mCoords* (posição actual) é atualizada com recurso ao *stepLength* (comprimento de passo). Esta classe apresenta ainda alguns métodos que são invocados pela parte gráfica, como é o caso do *resetCoords()* ou *getCoords()*.

3.2.4 Estimação do comprimento do passo

Para efeito de testes, a introdução manual do comprimento do passo nas definições é suficiente, mas para uma utilização normal da aplicação é mais cómodo o utilizador não

ter que se preocupar com quaisquer parâmetros. Por este motivo, foi desenvolvido um algoritmo que estima o comprimento do passo do utilizador.

Este algoritmo calcula o passo médio quando o utilizador se descola a pé numa zona com precisão de localização elevada. Foi utilizado o serviço *Activity Recognition* da Google Play [Que16] que permite estimar a actividade realizada pelo utilizador juntamente com seu nível de confiança. Quando se detecta que o utilizador está a caminhar, é analisada a precisão do sinal GNSS (com recurso ao método *Location.getAccuracy()*), sendo assim possível saber se as medidas podem ser usadas. Isto geralmente ocorre quando o utilizador se encontra num ambiente *outdoor*. Considerou-se um valor de precisão abaixo de 15m como suficiente, sendo o algoritmo corrido automaticamente. O algoritmo de estimação do passo considera o número de leituras da Localização Android e de Odometria realizadas em simultâneo. Em seguida é calculada a distância percorrida pelo mecanismo de Localização Android, isto é,

$$d_l = \sum_{i=1}^{N-1} \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2}\ , \tag{3.6}$$

onde N representa o número de amostras e (x_i, y_i) é a posição da i-ésima amostra. Através da Odometria conseguimos saber quantos passos foram andados pelo utilizador n_{passos}, o que torna possível estimar o comprimento médio do passo, $\widehat{p_0}$, através de:

$$\widehat{p_0} = \frac{d_l}{n_{passos}} \tag{3.7}$$

3.2.4.1 Teste Prático

Para testar o algoritmo de estimação do comprimento do passo do utilizador foram realizadas duas caminhadas dentro da FCT da UNL, sendo cada uma das caminhadas gravadas pelos dois dispositivos utilizados na secção anterior. As coordenadas obtidas pela aplicação estão representadas na figura 3.20.

No teste realizado no OnePlus 2 foram detectados $n_{passos} = 307$ e o $d_l = 243.799$m, pelo que se pode estimar um comprimento de passo de 0.7941cm. No XPERIA Compact $n_{passos} = 305$ e o $d_l = 252.707$m, o que nos permite estimar um comprimento de passo de 0.8285cm. A diferença entre as duas estimativas para o mesmo percurso foi de 4%, possivelmente função dos diferentes erros que afetaram as medições em cada terminal.

De forma a validar os resultados, foi realizado um teste sem a aplicação, na qual se caminhou ao longo de 50m em linha recta usando 42 passos, com um passo médio de 0.8065m. Este valor é muito semelhante aos obtidos através da aplicação, que têm uma variação inferior a 3% em ambos os dispositivos. Este valor é aceitável, pois o tamanho do passo de cada pessoa varia naturalmente ao longo do dia. Aumentando-se o número de amostras e recorrendo-se aos métodos de filtragem apresentados na secção 2.3.3, seria possível diminuir o erro na estimativa.

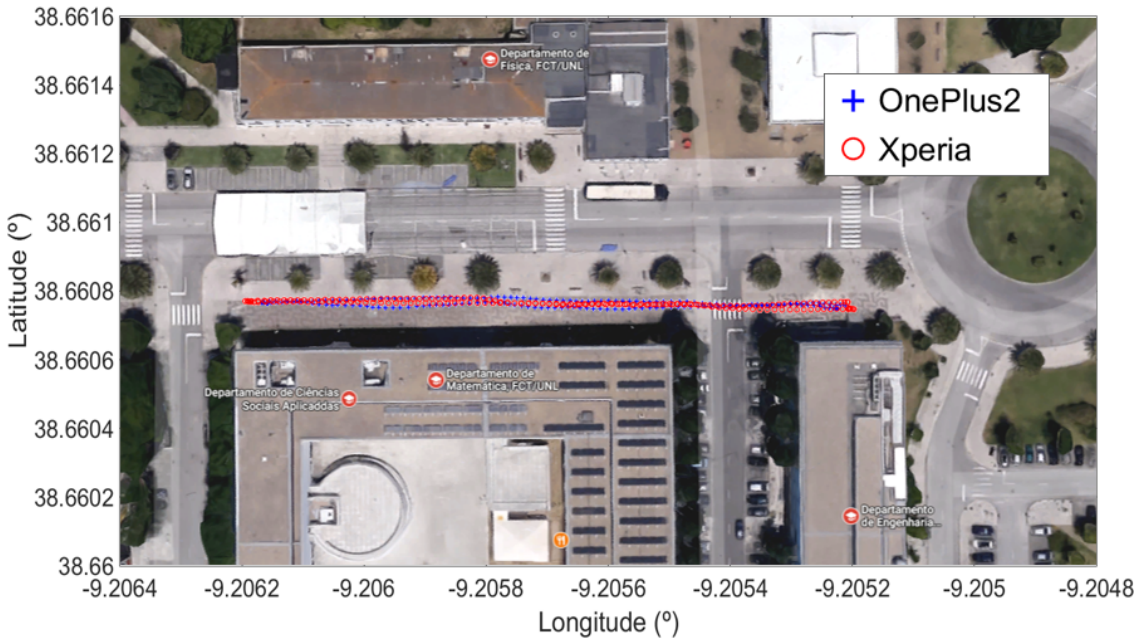


Figura 3.20: Caminhada Outdoor na FCT-UNL para estimação do comprimento do passo.

A inclusão do algoritmo de estimação de passo na aplicação Android, bem como o método em [LS13] apresentado no capítulo 2, foi deixado para trabalho futuro.

Capítulo 4

Estimação da localização de Emissores

A estimação da localização de emissores a partir de posições conhecidas de um ou mais terminais é um problema dual da localização de um terminal a partir do sinal de nós âncora conhecidos. O capítulo 4 estuda os vários processos envolvidos na estimação de APs e apresenta a aplicação WiFi Finder.

Numa primeira fase, analisa-se o processamento das medições de RSS. Numa segunda fase, são avaliados vários métodos de estimação de localização e é proposto um novo. Para cada método, é avaliada a precisão da estimação obtida recorrendo a medições realizadas com a aplicação.

4.1 Receção de RSS

A RSS WiFi é uma métrica da potência recebida num sinal WiFi. Todos os smarthphones Android conseguem, através do *WiFiManager* (biblioteca Android), obter informação sobre a RSS dos APs existentes na rede WiFi. Para além da RSS, o *WiFiManager* permite obter as informações mostradas na tabela 4.1. O SSID e o *Basic Service Set IDentifier* (BSSID) são identificadores a partir dos quais é possível criar um ID único a ser partilhado e utilizado pelos diversos dispositivos em modo de cooperação. Outra informação relevante são as métricas RSS e frequência, com as quais se podem aplicar diversos algortimos que estimam a distância ao respectivo emissor.

A utilização da RSS apresenta dois desafios:

- **Filtragem do sinal:** O valor de potência recebida tende a oscilar ao longo do tempo mesmo que o utilizador permaneça na mesma posição;
- **Conversão de potência para distância:** Não existe nenhuma fórmula direta e universal para fazer a conversão de potência para distância ou vice-versa.

Tabela 4.1: Leituras do WiFiManager do Android realizadas na aplicação WiFiFinder num terminal OnePlus 2.

SSID	BSSID	RSS	Canal Primário	Frequência Primária	Canal Central	Frequência Central	Largura de Banda	Segurança
NOS_WIFI_Fon	00:05:ca:93:7e:39	-86dBm	8	2447MHz	8	2447MHz	20MHz (2437 - 2457)	[ESS]
ZON-2920	00:05:ca:b0:29:28	-44dBm	6	2437MHz	6	2437MHz	20MHz (2427 - 2447)	[WPA-PSK-CCMP+TKIP][WPS] [WPA2-PSK-CCMP+TKIP][ESS]
NOS_WIFI_Fon	00:05:ca:b0:29:29	-42dBm	6	2437MHz	6	2437MHz	20MHz (2427 - 2447)	[ESS]
MEO-1B71D0	24:ec:99:1b:71:d0	-88dBm	11	2462MHz	11	2462MHz	20MHz (2452 - 2472)	[WPA-PSK-TKIP][WPS] [WPA2-PSK-CCMP][ESS]
MEO-WiFi	26:ec:99:1b:71:d1	-89dBm	11	2462MHz	11	2462MHz	20MHz (2452 - 2472)	[ESS]
Vodafone-BAC7EF	9c:97:26:ba:c7:ef	-80dBm	1	2412MHz	1	2412MHz	20MHz (2402 - 2422)	[WPA-PSK-TKIP][WPS] [WPA2-PSK-CCMP][ESS]
Vodafone-E51107	9c:97:26:e5:11:07	-59dBm	1	2412MHz	1	2412MHz	20MHz (2402 - 2422)	[WPA-PSK-TKIP][WPS] [WPA2-PSK-CCMP][ESS]
VodafoneHome	a4:b1:e9:ee:3e:76	-92dBm	1	2412MHz	1	2412MHz	20MHz (2402 - 2422)	[WPA-PSK-TKIP] [ESS]
Vodafone-EE3FB6	a4:b1:e9:ee:3f:b6	-81dBm	6	2437MHz	6	2437MHz	20MHz (2427 - 2447)	[WPA-PSK-TKIP][WPS] [WPA2-PSK-CCMP][ESS]
ZON-2FD0	bc:14:01:b0:2f:d8	-87dBm	2	2417MHz	2	2417MHz	20MHz (2407 - 2427)	[WPA-PSK-CCMP] [WPA2-PSK-CCMP][ESS]
NOS_WIFI_Fon	bc:14:01:b0:2f:d9	-86dBm	2	2417MHz	2	2417MHz	20MHz (2407 - 2427)	[ESS]
NOS-5330	bc:4d:fb:52:53:38	-82dBm	9	2452MHz	9	2452MHz	20MHz (2442 - 2462)	[WPA-PSK-CCMP+TKIP][WPS] [WPA2-PSK-CCMP+TKIP][ESS]
NOS_WIFI_Fon	bc:4d:fb:52:53:39	-85dBm	9	2452MHz	9	2452MHz	20MHz (2442 - 2462)	[ESS]

Uma das abordagens para realizar a filtragem do sinal é calcular a média do conjunto das últimas N medições reduzindo os efeitos da flutuação do sinal, de forma análoga ao esquema ZCL apresentado no capítulo 2. Esta filtragem introduz um atraso na atualização da potência vista pelos algoritmos, contribuindo para o erro final.

Na figura 4.1 estão apresentados testes realizados num OnePlus2 a azul e num Sony XPERIA Z1 Compact a vermelho, onde os dispositivos foram deixados em repouso a $2m$ de um AP. Foi realizada a média das últimas 15 amostras, sendo estes resultados apresentados a rosa e verde, para o OnePlus 2 e para o XPERIA respectivamente. Verifica-se que os dois dispositivos obtêm valores diferentes de potência, no mesmo instante, à mesma distância do AP. Este efeito deve-se principalmente às diferenças de hardware (antenas) dos *smartphones*. Como se pode observar, a média consegue minimizar as flutuações de sinal.

A figura 4.2 apresenta a leitura de potências de dois utilizadores em movimento e a média das últimas seis medições. Quando o utilizador se desloca, observa-se que a média de introduz um atraso na medida de RSS.

Podem ser considerados filtros mais complexos, como filtros de Kalman [Guv03] referidos no capítulo 2, mas ao longo dos testes deste capítulo optou-se por não aplicar filtragem, de forma a não introduzir atrasos, deixando-se o tratamento dos erros existentes na medida de RSS para os módulos de estimação de posição.

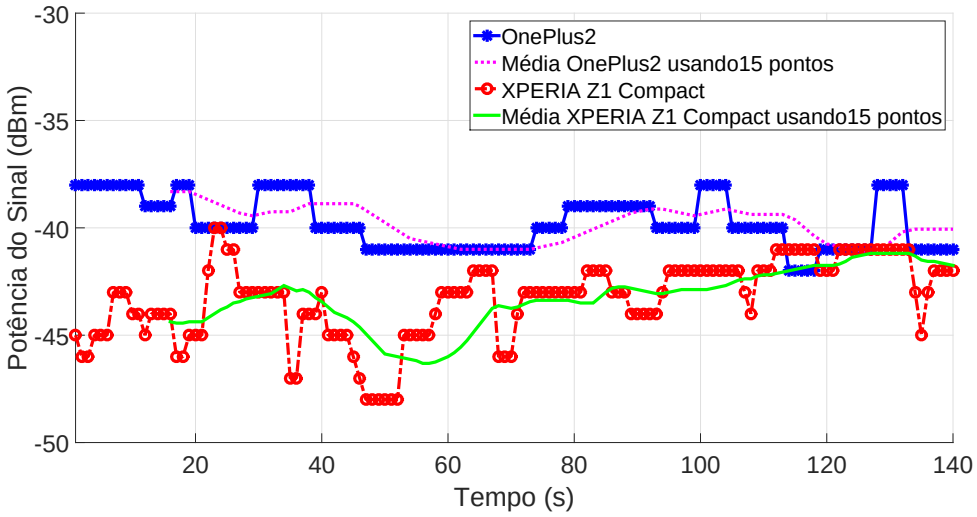
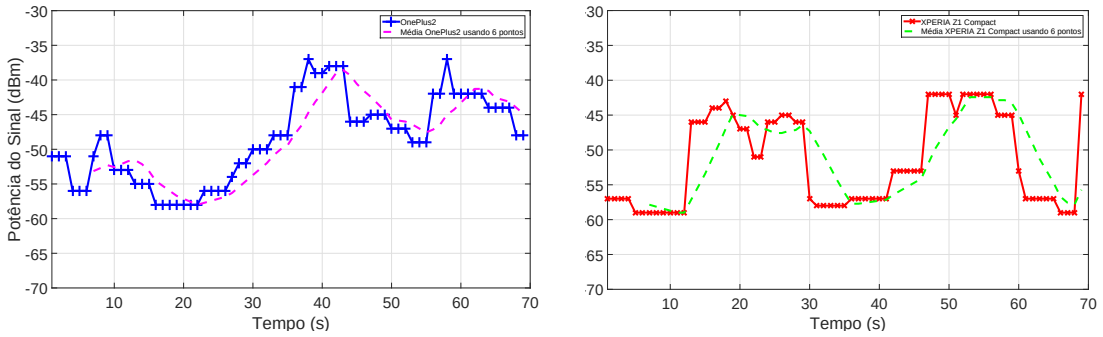


Figura 4.1: Potência do sinal recebida pelos dispositivos em repouso a 2 metros ao longo de 140 segundos.



a Teste realizado num OnePlus2

b Teste realizado num Sony XPERIA Z1 Compact

Figura 4.2: Potência do sinal recebida pelos dispositivos em movimento.

4.2 Estimação de distância

Na aplicação usou-se as bibliotecas do *WiFi Analyzer* da *VREM Software Development* [VRE], que realizam a transformação de potência P_r em distância d recorrendo à equação,

$$d = 10^{\frac{\alpha - P_r - 10\gamma \log_{10}(f)}{10\gamma}}, \tag{4.1}$$

onde é considerado que α é a constante que contém o efeito dos ganhos das antenas e da potência de transmissão (assumida uniforme)[Rap02], γ o PLE e f a frequência.

Relembrado o estudo apresentado na secção 2.2.4, podemos observar que esta fórmula fixa alguns parâmetros da equação (2.1), de Friis. Como é possível observar na figura 4.1, os dispositivos apresentam ganhos de antena G_r diferentes, o que faz com que para a mesma situação, a potência recebida por cada dispositivos seja diferente. Considerar o γ um valor fixo introduz algum erro, visto que existe uma variação do γ ao longo do tempo,

não se sabendo o valor exato em cada instante. Neste capítulo são introduzidos algoritmos que minimizam este erro, estimando α e γ.

Estas aproximações e generalizações introduzem erros no sistema, contudo a estimação relativa consegue diminuir estes efeitos de erro, como é demonstrado nas próximas secções.

4.3 Métodos de estimação

Nesta secção são estudados alguns algoritmos de estimação que fazem um trade-off entre complexidade e precisão. Este algoritmos são apresentados com complexidade crescente e precisão crescente, sendo que no final é explicado um algoritmo que permite estimar γ e P_t. Foram analisados os algoritmos: Trilateração, Trilateração Seletiva, Intersecção Simples, algoritmo de Range-Bancroft, algoritmo de Beck, algoritmo de Cheung, algoritmo de Gauss-Newton, algoritmo de Levenberg-Marquardt e a Relaxação de Tomic SOCP.

Inicialmente é descrito o cenário de testes no qual se avaliaram os diversos algoritmos. Posteriormente, são apresentados todos os algoritmos individualmente com os seus resultados.

Na secção seguinte é proposto um novo algoritmo que foi testado no mesmo cenário.

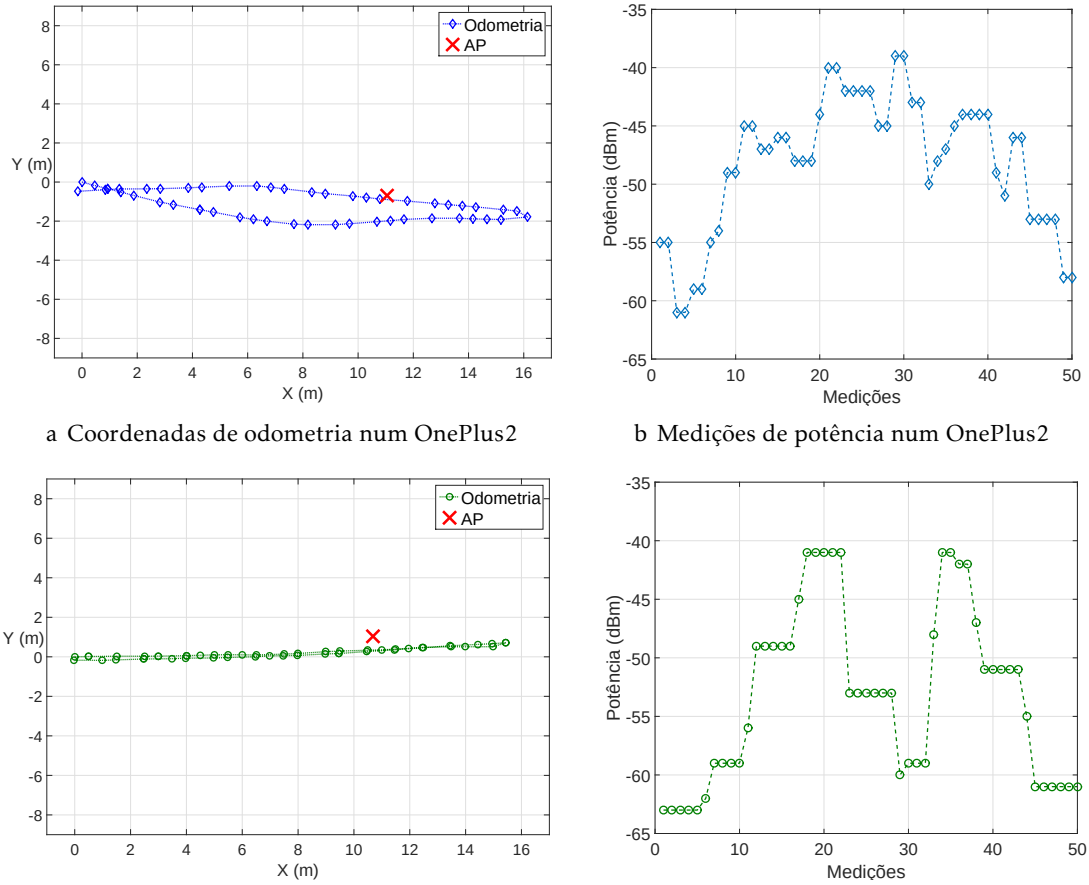
4.3.1 Cenário de Testes

De forma similar aos testes realizados no capítulo da Odometria, foram utilizados dois terminais móveis, OnePlus 2 e Sony XPERIA Z1 Compact, no terceiro piso do Departamento de Engenharia Electrótecnica da faculdade. Por motivos de simplificação e semelhança de resultados, apenas são apresentados os resultados obtidos nos ensaios realizados com o OnePlus 2 na apresentação individual de cada método. Posteriormente, na subsecção com a comparação final dos métodos, são comparados com as experiências realizadas com o Sony XPERIA.

Para estes testes, criou-se um mecanismo de exportação de dados que permite para cada ponto de odometria, ter informação da distância estimada ao AP, da RSS, da frequência, da posição estimada do AP (caso esta exista) e dos milisegundos em que esta estrutura de dados foi gerada. Com a aplicação a registar os dados, foi percorrido o percurso apresentado nas figuras 4.3a e 4.3c. Em cada coordenada de odometria foi registada a potência recebida, obtendo-se os conjuntos presentes nas figuras 4.3b e 4.3d. Nesta figura também se apresenta a posição do AP que se pretende localizar. Este teste foi realizado no OnePlus 2 usando o Acelerómetro para detecção de passos e o RV juntamente com o Giroscópio para detecção da orientação.

Na tabela 4.2 apresentam-se as características do AP utilizados nos testes.

Para além da avaliação do desempenho no cenário *indoor*, no apêndice H está disponível o resultado dos testes realizados num cenário veicular.



a Coordenadas de odometria num OnePlus2

b Medições de potência num OnePlus2

c Coordenadas de odometria num Sony XPERIA

d Medições de potência num Sony XPERIA

Figura 4.3: Percurso realizado onde se recolheu as coordenadas de odometria e as medições de potência.

Tabela 4.2: Características do router utilizado nos testes nesta secção.

SSID	BSSID	RSS	Canal Primário	Frequência Primária	Canal Central	Frequência Central	Largurade de Banda	Segurança
OpenFCT	00:12:da:9e:30:50	-43dBm	7	2442MHz	7	2442MHz	20MHz (2432 - 2452)	[ESS]

4.3.2 Algoritmos Tradicionais

A estimação da posição dos *smartphones* geralmente resume-se a resolver sistemas de equações não-lineares, muitas vezes com recurso aos mínimos quadrados. Os métodos estudados nesta secção tentam resolver estes sistemas [Sir10]. Consideram que existe um alvo t com coordenadas $2D$ ou $3D$ e existem vários pontos estacionários com posições conhecidas, $s_i, i = 1,...,N$, designados de âncoras. A estimativa da distância d_i, entre a âncora i e o ponto t, tem erro ε_i que tem origem no processo de medida, realizado através da RSS, ToA entre outros. É possível descrever a equação da distância como

$$d_i = \|s_i - t\| + \varepsilon_i \ . \tag{4.2}$$

No estudo que se segue, é usada uma simplificação da equação (4.2),

$$D = h(t) + \varepsilon \ , \tag{4.3}$$

onde D indica o vetor das medições de distância, $h(t)$ a função de diferenças entre o alvo e as âncoras com os valores em vetor e ε o vetor de erros. Esta notação permite avaliar vários métodos $h(t)$ com as medições consideradas. O objetivo de um método de posicionamento é encontrar um vetor de posição t tal que o termo de erro ε necessário para explicar a medida seja o menor possível.

As soluções que escolhem um ponto t que influencia t^* de forma a minimizar $\|D - h(t)\|$, ou de forma equivalente $\sum_i^N (D_i - h(t))^2$ são conhecidas como *Least Squares Solutions* (LSS)(Soluções de Mínimos Quadrados) e tendem otimizar a variância, isto é,

$$t^* = \arg\min_t \|D - h(t)\|^2 = \arg\min_t \sum_{i=1}^{N} (d_i - h_i(t))^2 \ . \tag{4.4}$$

Nesta dissertação as âncoras correspondem às posições conhecidas (através do módulo de localização) onde o terminal mede os valores de RSS do alvo. Visto que a odometria fornece coordenadas $2D$, os métodos são analisados com estimações $2D$. É de notar que neste esquema o erro ε não advém apenas das medições de distância, mas também dos erros gerados pelo módulo de localização.

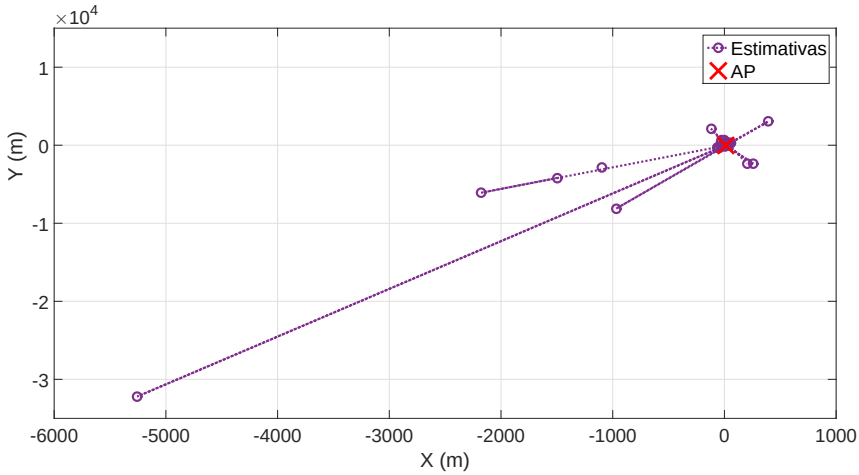
4.3.2.1 Trilateração

A Trilateração é o processo de determinar posições absolutas ou relativas de pontos por medição de distâncias, usando a geometria de círculos ($2D$) ou esferas($3D$).

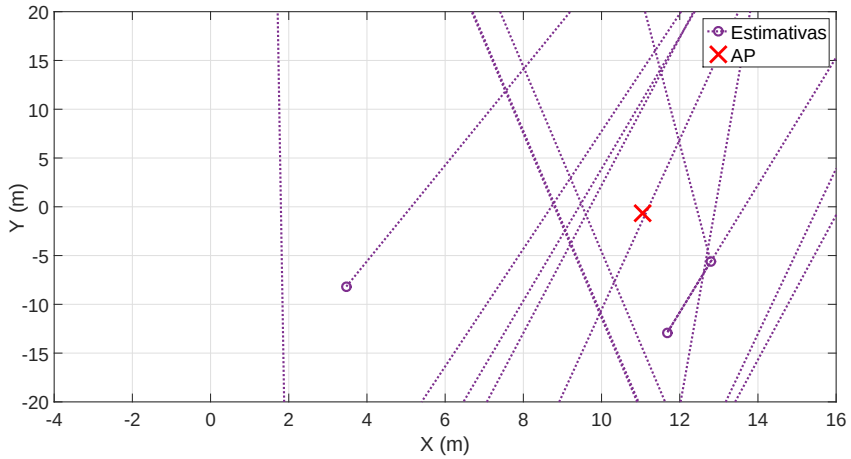
Utilizando a informação proveniente do WiFi Finder consegue-se estimar a posição relativa de um AP utilizando a posição do utilizador e a distância a que este se encontra do respetivo AP. Caso se pretenda estimar a sua localização num plano 2D basta considerar 3 posições conhecidas $s_i => (x_i, y_i), i = 1, 2, 3$, e que um AP se encontra na posição (x_t, y_t) a uma distância $d_i, i = 1, 2, 3$, e utilizar a equação (4.5) (demonstração no apêndice E).

$$\begin{cases} x_t = \frac{d_1^2-d_2^2+(x_2-x_1)^2+(y_2-y_1)^2}{2((x_2-x_1)-(y_2-y_1))} - \frac{(y_2-y_1)(d_1^2-d_3^2+(x_3-x_1)^2+(y_3-y_1)^2)}{2((x_2-x_1)(x_3-x_1)-(x_3-x_1)(y_2-y_1))} + x_1 \\ y_t = \frac{(x_2-x_1)(d_1^2-d_3^2+(x_3-x_1)^2+(y_3-y_1)^2)}{2((x_2-x_1)(x_3-x_1)-(x_3-x_1)(y_2-y_1))} - \frac{d_1^2-d_2^2+(x_2-x_1)^2+(y_2-y_1)^2}{2((x_2-x_1)-(y_2-y_1))} + y_1 \end{cases} \tag{4.5}$$

Nas figuras da secção 4.4 encontram-se representadas a roxo as diversas estimações realizadas ao longo do percurso, e a vermelho a localização real do emissor. Cada estimação representada pelo marcador 'o' utiliza os últimos três conjuntos de dados (s_i, d_i) . É possível observar que existe uma grande imprecisão, resultante da extrema sensibilidade aos erros presentes na localização e nas medidas de RSS.



a Resolução 7000x50000 (metros por unidade)



b Resolução 20x40 (metros por unidade)

Figura 4.4: Testes de estimação utilizando Trilateração.

4.3.2.2 Trilateração Seletiva

A seleção dos 3 pontos (s_i, d_i) é de extrema importância para o correcto funcionamento do algoritmo. É importante perceber que a capacidade de armazenamento de pontos em dispositivos móveis é limitada. Contudo caso existam pontos armazenados, é possível realizar uma seleção de pontos que diminua o erro do método de trilateração.

No caso do emissor estar estático é possível armazenar um número de pontos superior do que se este se encontrar em movimento, visto que alguns pontos em movimento podem conter informação desatualizada e grandes variações do ambiente em que o utilizador se encontra.

Para efeitos de teste, foi desenvolvido um algoritmo que considera os últimos 25 pontos s_{i,d_i} dos quais são utilizados os 3 que apresentem um valor de distância menor. Teoricamente, quanto mais perto do emissor, menor será o erro, tendo assim uma estimativa da localização com melhor precisão.

Nas figuras 4.5 são apresentados os resultados utilizando Trilateração Seletiva. Comparando com os resultados da Trilateração simples, observa-se que os erros máximos diminuíram e os valores estimados apresentam uma variância inferior. Mesmo assim, apresentam erros de estimação muito elevados, sendo por isso necessário recorrer a soluções de multilateração (com 3 ou mais pontos) que reduzam o efeito dos erros do módulo de localização e de RSS.

4.3.2.3 Intersecção Simples

A Intersecção simples é também conhecida como o método *line-of-position* [Caf00; Foy76]. O método é derivado do sistema de equações do quadrado das distância,

$$2s_i^T t = \|t\|^2 + \|s_i\|^2 - d_i^2, \tag{4.6}$$

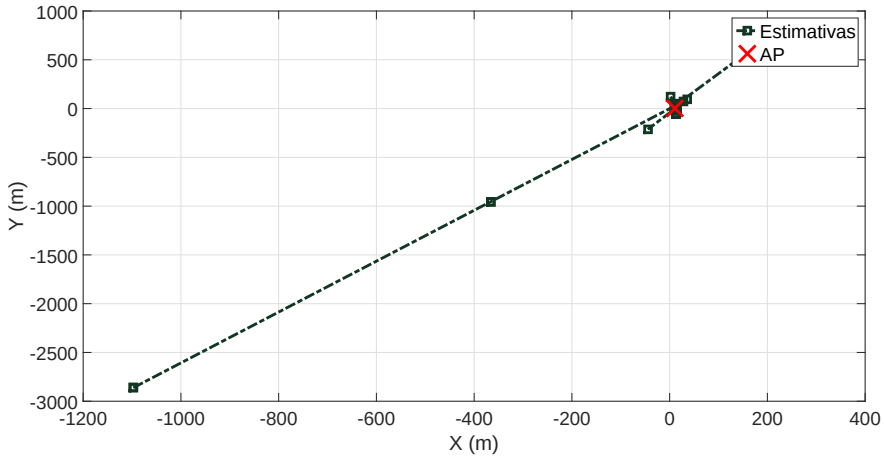
onde se pode subtrair uma das equações às restantes. Desta forma, é possível obter um sistema com $N-1$ equações lineares onde os termos não lineares $\|t\|^2$ foram eliminados. Do ponto de vista geométrico, esta subtração corresponde a considerar cada $N-1$ pares de círculos e substituir a intersecção dos mesmos por uma linha que representa essa intersecção. É de notar que por causa desta eliminação, este método é (assintoticamente) equivalente a resolver a posição com uma medição a menos.

Para a resolução do sistema, consideremos,

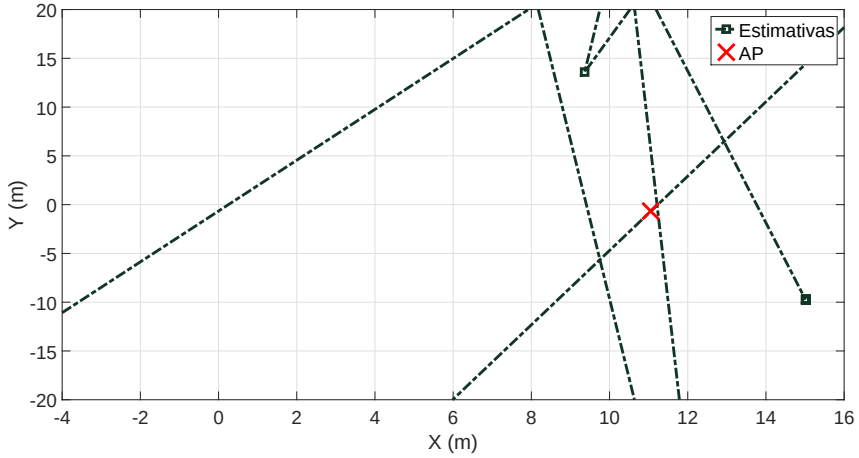
$$A = \begin{bmatrix} 2s_1^T \\ \vdots \\ 2s_N^T \end{bmatrix}, B = \begin{bmatrix} \|s_1\|^2 - d_1^2 \\ \vdots \\ \|s_N\|^2 - d_N^2 \end{bmatrix}, C = \begin{bmatrix} -1_{(N-1)\times 1} & I_{N-1} \end{bmatrix}. \tag{4.7}$$

Podemos estimar a posição do emissor, resolvendo

$$\widehat{t} = (A^T C^T C A)^{-1} A^T C^T C B. \tag{4.8}$$



a Resolução 1600x4000 (metros por unidade)



b Resolução 20x40 (metros por unidade)

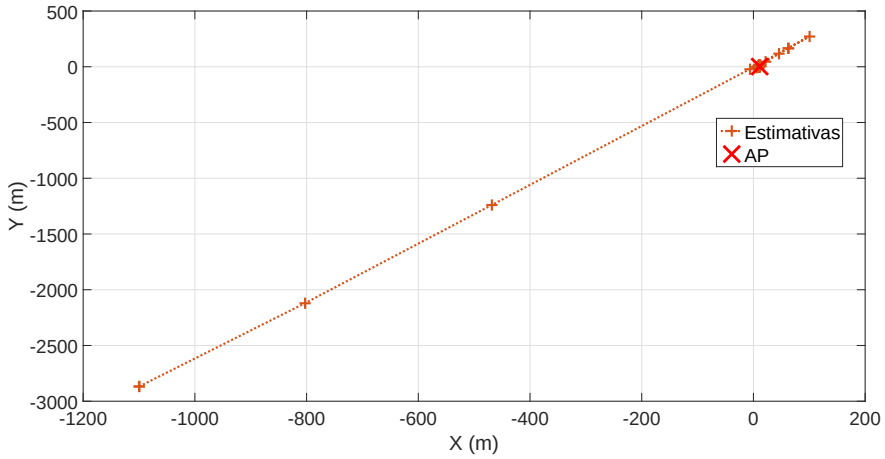
Figura 4.5: Testes de estimação utilizando Trilateração Selectiva.

Nas figuras em 4.6 podemos observar os resultados das estimações utilizando Intersecção Simples. Os resultados mostram que o algoritmo convergiu para um valor perto do real, apresentado menos de 3 metros de erro. Este erro é claramente inferior aos verificados com os dois métodos de Trilateração.

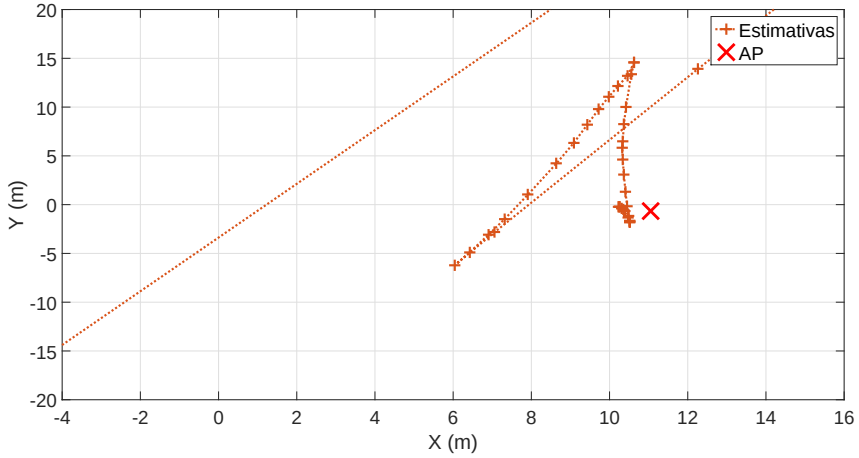
4.3.2.4 Range-Bancroft

O método de Bancroft é conhecido pelas suas aplicações usando GPS [BH15], mas pode ser utilizado com outras métricas de distâncias.

O ponto de partida do método de Bancroft também é baseado numa expansão das equações de intervalo quadrado apresentadas na interseção simples, que pode ser escrita usando matrizes como $At = \mathbf{1}\|t\|^2 + B$, onde $\mathbf{1}$ denota um vetor com todas as entradas iguais a um. Esta equação pode ser resolvida multiplicando ambos os lados pela matriz



a Resolução 1400x3500 (metros por unidade)



b Resolução 20x40 (metros por unidade)

Figura 4.6: Testes de estimação utilizando Intersecção Simples.

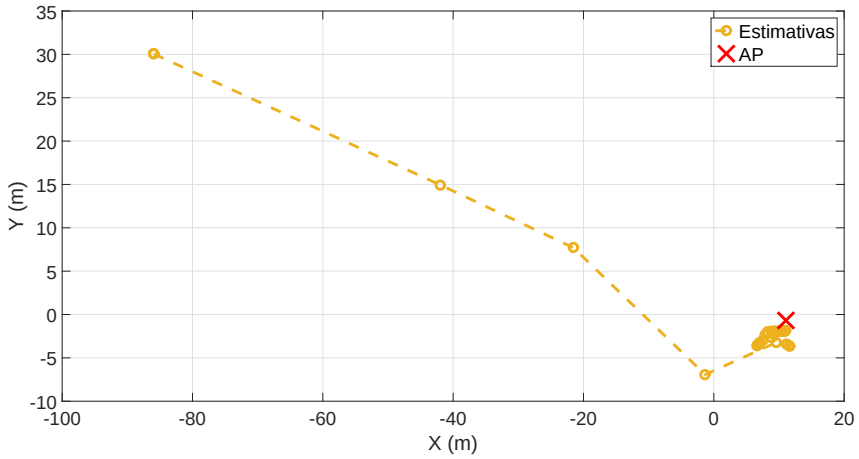
pseudo-inversa de Moore-Penrose de A e retirando a norma ao quadrado.

As soluções candidatas, x_i, são obtidas das duas raízes de (4.9) [Sir10], $x_i = \|t\|^2, i = 1...2$,

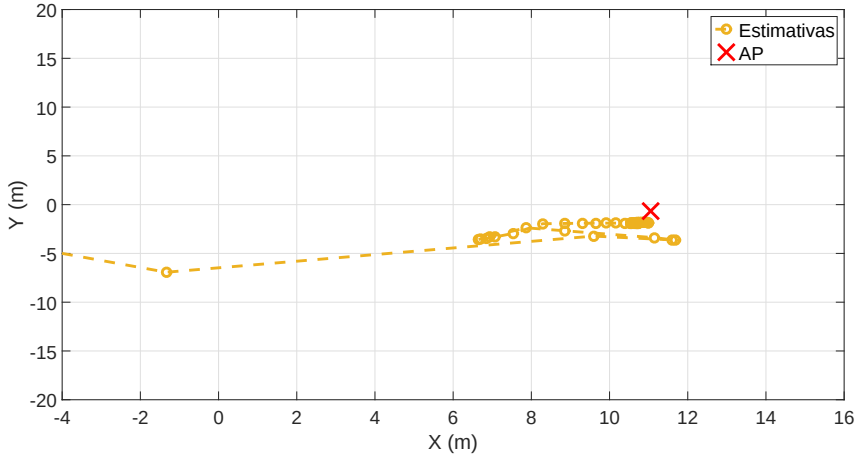
$$\|p\|^2 \|t\|^4 + (2p^T q - 1)\|t\|^2 + \|q\|^2 = 0, \tag{4.9}$$

onde $p = (A^T A)^{-1} A^T \mathbf{1}$ e $q = (A^T A)^{-1} A^T B$. A solução [Sir10], é o valor de $\widehat{t} = px_i + q$ com menor valor residual $\sum_{i=1}^{N} \left(d_i - \left\|s_i - \widehat{t}\right\|\right)^2$.

Nas figuras em 4.7 estão representados os resultados das estimações utilizando o método de Bancroft. Constata-se que este algoritmo produziu melhores resultados que o algoritmo de Intersecção Simples, encontrando-se as últimas sete estimações a menos de 1 metro do valor real.



a Resolução 120x45 (metros por unidade)



b Resolução 20x40 (metros por unidade)

Figura 4.7: Testes de estimação utilizando Range-Bancroft.

4.3.2.5 Beck

O método de Beck *et al* [Bec+08] baseia-se num procedimento capaz de calcular a solução mínima-quartica exata para as equações de distâncias. O procedimento requer o uso do método de bisecção para encontrar uma raíz de uma função estritamente monótona sem variação num intervalo de partida não infinito, ou seja, não é um método *closed-form*, mas ainda assim tem convergência garantida.

Realizando uma substituição de $\lambda = \|t\|^2$ e utilizando as matrizes

$$E = \begin{bmatrix} 2s_1^T & -1 \\ \vdots & \vdots \\ 2s_N^T & -1 \end{bmatrix}, B = \begin{bmatrix} \|s_1\|^2 - d_1^2 \\ \vdots \\ \|s_N\|^2 - d_N^2 \end{bmatrix}, P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, w = \begin{bmatrix} 0 \\ 0 \\ -\frac{1}{2} \end{bmatrix}, \tag{4.10}$$

a solução $\widehat{z}$ é obtida como

$$\widehat{z}(\lambda) = (E^T E + \lambda P)^{-1}(E^T B - \lambda w), \tag{4.11}$$

onde o procedimento de bisecção é usado para encontrar λ^*, o zero de

$$\phi(\lambda) = \widehat{z}(\lambda)^T P \widehat{z}(\lambda) + 2w^T \widehat{z}(\lambda)\,. \tag{4.12}$$

A posição estimada por este método $\widehat{t}$ tem como coordenadas os primeiros j componentes de $\widehat{z}(\lambda^*)$, onde j representa a dimensão do espaço considerado.

A figura 4.8 representa os resultados das estimações utilizando o algoritmo de Beck. Este algoritmo consegue alcançar precisões semelhantes ao de Bancroft, mas com uma convergência mais rápida.

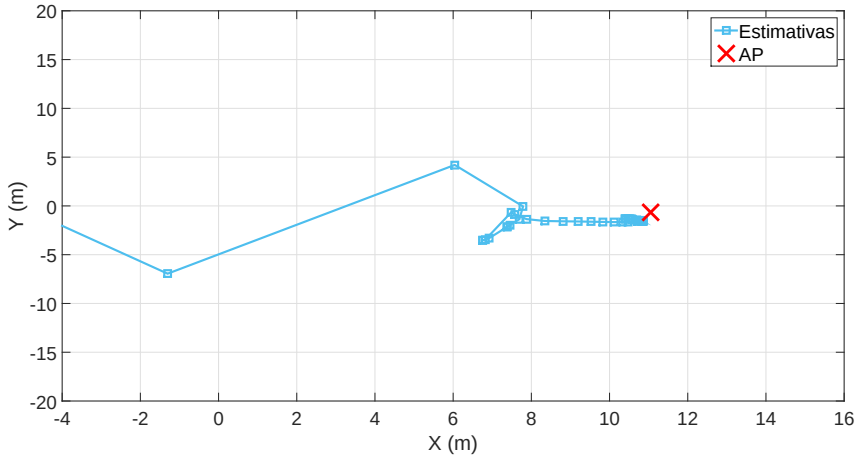


Figura 4.8: Estimação utilizando Beck com as N coordenadas anteriores.

4.3.2.6 Cheung

O método de Cheung *et al* [Che+04] fornece uma solução de mínimos quadrados ponderada restrita para medições de distâncias. As incógnitas a serem resolvidas são a posição e a distância ao quadrado da primeira posição de odometria s_1.

Em suma, para utilizar este método deve-se inicialmente calcular as matrizes,

$$\begin{gathered} E = \begin{bmatrix} 2s_1^T & -1 \\ \vdots & \vdots \\ 2s_N^T & -1 \end{bmatrix},\ B = \begin{bmatrix} \|s_1\|^2 - d_1^2 \\ \vdots \\ \|s_N\|^2 - d_N^2 \end{bmatrix},\ F = \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_N \end{bmatrix}, \\ \Psi = E\Sigma E,\ P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},\ w = \begin{bmatrix} 0 \\ 0 \\ -\frac{1}{2} \end{bmatrix}, \end{gathered} \tag{4.13}$$

calcular U e Λ a partir da decomposição de valores próprios

$$\left(E^T\Psi^{-1}E\right)^{-1}P = U\begin{bmatrix}\gamma_1 & & \\ & \gamma_2 & \\ & & 0\end{bmatrix}U^{-1}\ . \tag{4.14}$$

Utilizando estas variáveis é possível obter

$$c = 2U^T w, \tag{4.15}$$

$$g = 2U^T(E^T\Psi^{-1}E)^{-1}w, \tag{4.16}$$

$$e = (\Psi^{-1}EU)^T B, \tag{4.17}$$

$$f = U^{-1}(E^T\Psi^{-1}E)^{-1}E\Psi^{-1}B. \tag{4.18}$$

De seguida, deve encontrar-se a raiz de λ^* mais perto de zero a partir da equação de cinco raízes dada por :

$$\begin{aligned}c_3f_3 - \frac{\lambda}{2}c_3g_3 + \sum_{i=1}^{2}\frac{c_if_i}{1+\lambda\gamma_i} - \frac{\lambda}{2}\sum_{i=1}^{2}\frac{c_ig_i}{1+\lambda\gamma_i} + \sum_{i=1}^{2}\frac{e_if_i\gamma_i}{(1+\lambda\gamma_i)^2} - \\ \frac{\lambda}{2}\sum_{i=1}^{2}\frac{(e_ig_i+c_if_i)\gamma_i}{(1+\lambda\gamma_i)^2} + \frac{\lambda^2}{4}\sum_{i=1}^{2}\frac{c_ig_i\gamma_i}{(1+\lambda\gamma_i)^2} = 0\ .\end{aligned} \tag{4.19}$$

Cheung não especifica nenhuma maneira de resolver a equação de cinco raízes. Na implementação deste teste, a equação de cinco raízes foi manipulada de acordo com [Sir10], que transforma esta equação num polinômio de quinto grau $\lambda = \|t\|^2$ com coeficientes

$$p_5 = -\frac{1}{2}c_3g_3\gamma_1^2\gamma_2^2\ , \tag{4.20}$$

$$p_4 = c_3f_3\gamma_1^2\gamma_2^2 + (-\frac{1}{4}c_2g_2 - c_3g_3)\gamma_1^2\gamma_2 + (-\frac{1}{4}c_2g_2 - c_3g_3)\gamma_1\gamma_2^2\ , \tag{4.21}$$

$$\begin{aligned}p_3 = (\frac{1}{2}c_2f_2 + 2c_3f_3 - \frac{1}{2}e_2g_2)\gamma_1^2\gamma_2 + (\frac{1}{2}c_1f_1 - 2c_3f_3 - \frac{1}{2}e_1g_1)\gamma_1\gamma_2^2 + \\ (-\frac{1}{2}c_2g_2 - \frac{1}{2}c_3g_3)\gamma_1^2 + (-\frac{1}{2}c_1g_1 - \frac{1}{2}c_3g_3)\gamma_2^2 + (-\frac{1}{2}c_1g_1 - \frac{1}{2}c_2g_2 - 2c_3g_3)\gamma_1\gamma_2\end{aligned}\ , \tag{4.22}$$

$$\begin{aligned}p_2 = e_2f_2\gamma_1^2\gamma_2 + e_1f_1\gamma_1\gamma_2^2 + (c_2f_2 + c_3f_3)\gamma_1^2 + (c_1f_1 + c_3f_3)\gamma_2^2 + \\ (-e_1g_1 - e_2g_2 + c_1f_1 + c_2f_2 + 4c_3f_3)\gamma_1\gamma_2 + \ , \\ (-\frac{1}{4}c_1g_1 - c_2g_2 - c_3g_3)\gamma_1 + (-\frac{1}{4}c_2g_2 - c_1g_1 - c_3g_3)\gamma_2\end{aligned} \tag{4.23}$$

$$p_1 = (2e_1f_1 + 2e_2f_2)\gamma_1\gamma_2 + (\frac{1}{2}c_1f_1 + 2c_2f_2 + 2c_3f_3 - \frac{1}{2}e_1g_1)\gamma_1 + (2c_1f_1 + \frac{1}{2}c_2f_2 + 2c_3f_3 - \frac{1}{2}e_2g_2)\gamma_2 - \frac{1}{2}c_1g_1 - \frac{1}{2}c_2g_2 - \frac{1}{2}c_3g_3 \quad , \tag{4.24}$$

$$p_0 = \sum_{i=1}^{2} e_i f_i \gamma_i + \sum_{i=1}^{3} c_i f_i \ . \tag{4.25}$$

As raízes do polinómio podem ser encontradas com métodos numéricos, e é escolhido o que mais se aproximar de zero. O valor estimado de t, $\widehat{t}$, corresponde aos primeiros j elementos de

$$\widehat{z} = (E^T\Psi^{-1}E + \lambda^* P)^{-1}(E^T\Psi^{-1}B - \lambda^* w) \ . \tag{4.26}$$

Na figura 4.9 estão representados os resultados das estimações utilizando o método de Cheung. Como seria se esperar, os resultados do algoritmo de Cheung são praticamente idênticos aos do algoritmo de Beck, mas com um peso computacional menos elevado, como se confirma numericamente na secção 4.5.

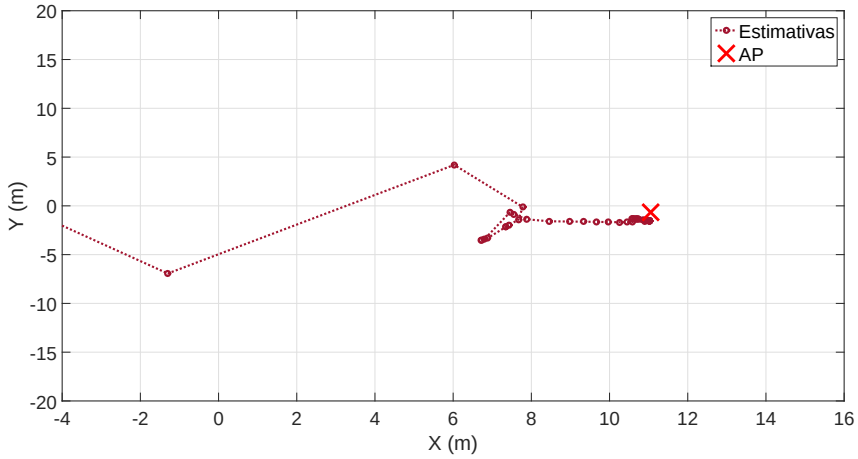


Figura 4.9: Estimação utilizando Cheung com as N coordenadas anteriores.

4.3.2.7 Gauss-Newton

O *Weighted Iterative Least Squares* (WLS)(Método iterativo com pesos mínimos quadrados) é também conhecido por método de *Gauss-Newton* [Sir10]. É um método numericamente robusto para resolver sistemas na presença de ruído aditivo com variância finita, permitindo atribuir uma ponderação às amostras. Uma desvantagem é necessitar de uma posição de partida inicial e, no caso de haver múltiplas soluções ambíguas, ser apenas encontrada uma solução.

O algoritmo é, no entanto, muito simples de implementar, desde que as operações básicas de álgebra linear estejam disponíveis e não exige por exemplo, a resolução de

polinómios de quinto grau. O único obstáculo com modelos de medição mais complexos é a necessidade de calcular explicitamente a matriz Jacobiana, ou a derivada da equação de medição. No caso de medições de distância, o método consiste apenas em vetores unitários que apontam para a estimativa de posição.

Para estimar a posição do emissor utilizando WLS devem ser realizados os seguintes passos:

1. Escolher uma estimativa inicial t_0 e tolerância de paragem δ A. Definir $k = 0$.

2. Calcular

$$J_k(t) = \begin{bmatrix} \frac{(s_1 - t)^T}{\|(s_1 - t)\|} \\ \vdots \\ \frac{(s_N - t)^T}{\|s_N - t\|} \end{bmatrix} \tag{4.27}$$

3. Definir $t_{k+1} = t_k + \Delta t_k$ com

$$\Delta t_k = -\frac{\Sigma^{-\frac{1}{2}} J_k}{\Sigma^{-\frac{1}{2}} (h(t_k) - d)}\ , \tag{4.28}$$

   onde $\Sigma^{-\frac{1}{2}}$ denota o inverso da raíz quadrada da matriz resultante.

4. Se a condição $\|\Delta t_k\| < \delta$ não for satisfeita e $k < K_{max}$, incrementar k e voltar a repetir o ponto 2.

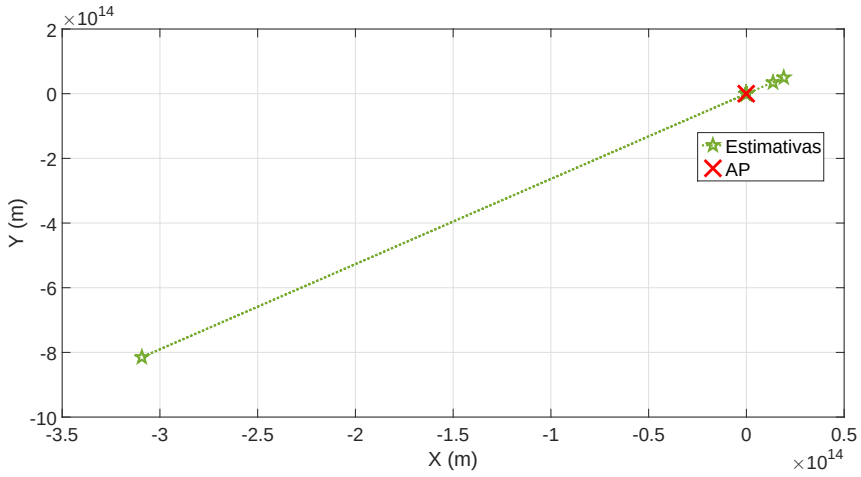
Nas figuras 4.10 estão representados os resultados das estimações utilizando o algoritmo de Gauss-Newton. Este demorou mais a convergir que os últimos três algoritmos estudados e apresenta um erro superior, nunca atingindo uma precisão abaixo de um metro.

Os erros podem ser minimizados com o uso de informação prévia. No apêndice F encontra-se uma outra versão do algoritmo de Gauss-Newton, o algoritmo Gauss-Newton-reg. Este algoritmo converge mais rapidamente numa fase inicial, isto é, até cerca de 15 pontos.

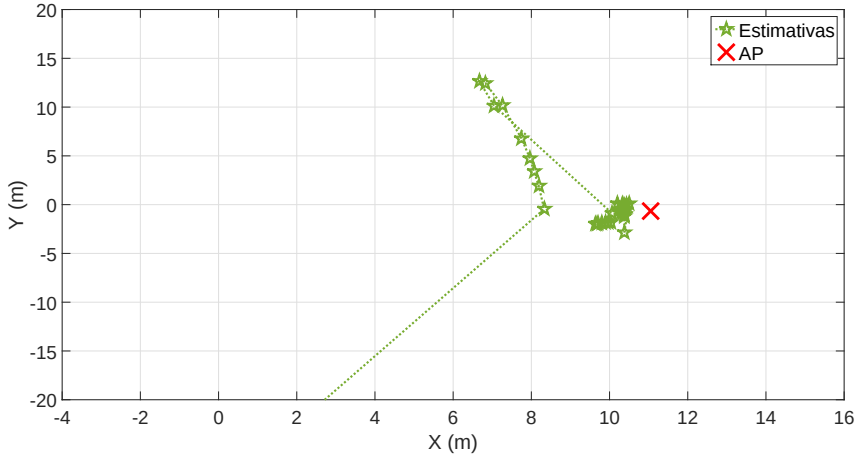
4.3.2.8 Levenberg-Marquardt

A otimização de Levenberg-Marquardt é um algoritmo não-linear de pseudo-segunda ordem o que significa que funciona apenas com avaliações de funções e informações de gradiente, mas estima a matriz de Hessian usando a soma dos produtos externos dos gradientes [Lou05; Row96].

De forma semelhante ao método Gauss-Newton, o algoritmo de Levenberg-Marquardt é um procedimento iterativo que tende para um mínimo local $t^{'}$. Para iniciar a minimização de $\sum_i^N (D_i - h(t))^2$, é necessário calcular uma estimativa inicial t_0, que nesta dissertação é calculada usando a média,



a Resolução 12^{14}x4^{14} (metros por unidade)



b Resolução 20x40 (metros por unidade)

Figura 4.10: Testes de estimação utilizando Gauss-Newton.

$$t_0 = \frac{\sum_{i=0}^{N} s_i}{N}\,. \tag{4.29}$$

Para evitar o sobredimensionamento das matrizes utilizadas, inicialmente é realizada uma balização das distâncias medidas com $d_i = max(d_i, 10^{-7})$. Cada par (s_i, d_i) tem um peso no algoritmo $w_i, i = 1,...,N$ que foi considerado como $w_i = 1/{d_i}^2$.

De forma semelhante aos algoritmos anteriores, o algoritmo começa por calcular a posição de partida inicial, $t_0 = t_d$ e define a tolerância de paragem δ (10^{-4} vezes a função de tolerância). Em cada iteração é calculado o Jacobiano J, e $t_{k+1} = t_k + \Delta t_k$, onde Δt_k é a LSS de

$$(J^T J + \vartheta \text{diag}(J^T J))\Delta t = J^T \varepsilon, \tag{4.30}$$

e o fator amortecimento ϑ é atualizado com base no sucesso da última atualização t_k. O ciclo acaba quando $\|\Delta t_k\| < \delta$, ou quando K_{max} é atingido.

O algoritmo LM foi inicialmente implementado usando a biblioteca Matlab, utilizando a função *lsqnonlin()*. Também foi selecionado para o protótipo Android não colaborativo, onde foi implementado utilizando a biblioteca *org.apache.commons.math3.fitting.leastsquares.LevenbergMarquardtOptimizer*, com um número máximo de iterações $K_{max} = 1000$.

Na figura 4.11 apresentam-se os resultados das estimações utilizando o método de Levenberg-Marquardt. Os resultados são muito semelhantes aos do algoritmo de Beck e de Cheung em termos de precisão.

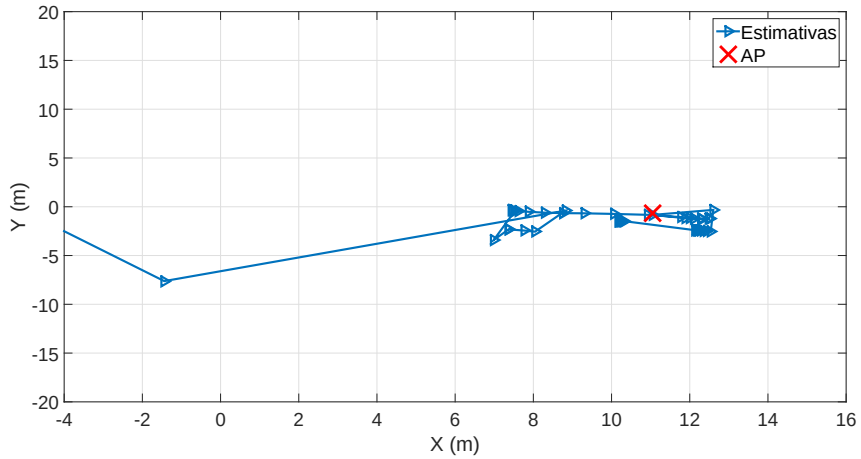


Figura 4.11: Estimação utilizando o método de Levenberg-Marquardt com N coordenadas anteriores.

4.3.2.9 Relaxação Tomic SOCP

O algoritmo Tomic [Tom+15] deriva um estimador convexo, que se aproxima fortemente do *Maximum Likelihood* (ML)(Máxima Probabilidade) para pouco ruído. Então, o novo estimador é relaxado aplicando relaxações convexas eficientes que são baseadas em *Second Order Cone Programming* (SOCP)(Programação de Cone de Segunda Ordem) .

Na sua formulação é considerado o seguinte modelo de propagação, *Path Loss Model* (PLM)(Modelo de Perdas ao longo do percurso) (em dB),

$$L_i = L_0 + 10\,\gamma\,\log_{10}\frac{\|t - s_i\|}{d_0} + v_i\,, i = 1,\ldots,N\,, \tag{4.31}$$

onde L_0 denota o valor das perdas ao longo do percurso a uma distância de referência mais curta d_0 (isto é $\|t - s_i\| \geq d_0$), γ é o PLE e v_i é o termo de sombreamento log-normal modelado como uma variável aleatória Gaussiana de média zero com variância σ_i^2, i.e. $v_i \sim \mathcal{N}\left(0, \sigma_i^2\right)$.

Para resolver este problema de localização é utilizado um relaxamento convexo para lidar com a não convexidade do problema.

A partir da relação $L_i(\text{dB}) = 10\log_{10}\frac{P_T}{P_i}$, onde P_i é o RSS medido ao longo das posições de odometria e P_T é a potência de transmissão desconhecida, o problema de localização pode ser formulado pela perda de percurso.

Com base nas medições em 4.31, o estimador ML é encontrado resolvendo o problema dos mínimos quadrados não-linear e não-convexo

$$\widehat{t} = \underset{t}{arg\ min} \sum_{i=1}^{N} \frac{1}{\sigma_i^2}\left[(L_i - L_0) - 10\ \gamma\ \log_{10}\frac{\|t - s_i\|}{d_0}\right]^2 . \tag{4.32}$$

A atenuação do sinal pode ser causada por muitos efeitos, como o desvanecimento de sinal pelo efeito de múltiplos caminhos, difração, reflexão, ambiente entre outras. Assim, é razoável supor que o PLE γ não é conhecido nos pontos de medição de odometria. Por esta razão, é assumido que P_T e γ são simultaneamente desconhecidos. A estimativa conjunta ML de t, L_0 e γ é escrita como

$$\widehat{\theta} = \underset{\theta=[t;L_0;\gamma]}{arg\ min} \sum_{i=1}^{N} \frac{1}{\sigma_i^2}\left[\left(L_i - h^T\theta\right) - 10\ g^T\theta\ log_{10}\frac{\left\|C^T\theta - s_i\right\|}{d_0}\right]^2 , \tag{4.33}$$

onde $h = [0_{2\times1}; 1; 0]$, $g = [0_{3\times1}; 1]$ e $C = [I_2; 0_{2\times2}]$. O problema definido em (4.33) é não convexo e não tem solução de forma fechada. Para o resolver pode ser utilizado o procedimento iterativo:

1. Selecionar valores empíricos [Rap02], e definir a estimativa inicial de $\gamma, \widehat{\gamma^0} \in [\gamma_{min}, \gamma_{max}]$.
2. Resolver a equação[1] (4.34) e encontrar a estimativa inicial de $t, \widehat{t^0}$.

$$\begin{aligned} &\underset{t,g,z,\eta,p}{\text{minimize}}\ p \\ &\text{subject to}\ \ \|[2z; p-1]\| \le p+1,\ \|t - s_i\| \le g_i,\ z_i = \psi_i g_i - \eta d_0,\ i = 1,\ldots,N. \end{aligned} \tag{4.34}$$

3. Com recurso a $\widehat{\gamma^0}$ e $\widehat{t^0}$ calcular a estimativa ML de L_0 , $\widehat{L_0}$.
4. Calcular o valor da função f_0 utilizando os valores calculados $\left\{\widehat{t^0},\ \widehat{L_0^0},\ \widehat{\gamma^0}\right\}$ na equação (4.33).
5. Considerar um contador de iterações k e iniciar o mesmo a 1.
6. Utilizar $\widehat{t^{k-1}}$ e $\widehat{L_0^{k-1}}$ para encontrar a estimativa ML para γ , $\widehat{\gamma}$, definido por

$$\widehat{\gamma^k} = \frac{\sum_{i=1}^{N} 10\log_{10}\ \frac{\left\|\widehat{t^{k-1}} - s_i\right\|}{d_0}\left(L_i - \widehat{L_0^{k-1}}\right)}{\sum_{i=1}^{N}\left[10\log_{10}\ \frac{\left\|\widehat{t^{k-1}} - s_i\right\|}{d_0}\left(L_i - \widehat{L_0^{k-1}}\right)\right]^2} . \tag{4.35}$$

[1] A demonstração de como chegar ao problema não convexo que permite encontrar simultaneamente t e L_0 é apresentado no Apêndice G.

Caso $\widehat{\gamma}^k \notin [\gamma_{min},\ \gamma_{max}]$ o processo deve ser parado neste ponto e utilizado $\widehat{t}^{k-1}$ como estimativa final.

7. Com recurso a $\widehat{\gamma}^k$ e $\widehat{L}_0^k$ deve resolver

$$\begin{aligned} &\underset{t,z,r,\beta}{\text{minimize}} \left(\widehat{\alpha}_i^2 z_i - 2d_0\widehat{\alpha}_i r_i\right) \\ &\text{subject to} \\ &\left\|[2t;\beta-1]\right\| \leq \beta+1,\ \ \|[2r_i;z_i-1]\| \leq z_i+1,\ \ z_i = \beta - 2s_i^T t + \|s_i\|^2,\ \ i=1,\ldots,N\ , \end{aligned} \tag{4.36}$$

onde $\widehat{\alpha}_i = 10^{\frac{L_i-\widehat{L}_0}{10\gamma}}$. Desta forma é possível obtermos uma estimativa de t, $\widehat{t}^k$ e L_0, $\widehat{L}_0^k$.

8. Calcular a função de custo $f_k = \sum_{i=1}^{N}\left[\left(L_i - \widehat{L}_0^k\right) - 10\ \widehat{\gamma}^k\ \log_{10}\frac{\|\widehat{t}_k - s_i\|}{d_0}\right]^2$. Caso $\frac{|f_k - f_{k-1}|}{f_{k-1}} < \varepsilon$ (ε é o limiar de paragem) ou $k > K_{max}$ (K_{max} representa o número máximo de iterações) deve parar e considerar como estimativa $\widehat{t}^k$; caso contrário repetir o passo 6 e incrementar k.

A figura 4.12 representa os resultados das estimações utilizando o método de Tomic, onde foi considerado que $L_0^0 = 30\ dB$, $\gamma_0 = 3$ e $d_0 = 1$. É possível observar que de todas as abordagens estudadas até aqui, esta é a que obtém estimativas que mais rapidamente e direcionalmente convergem para o ponto real.

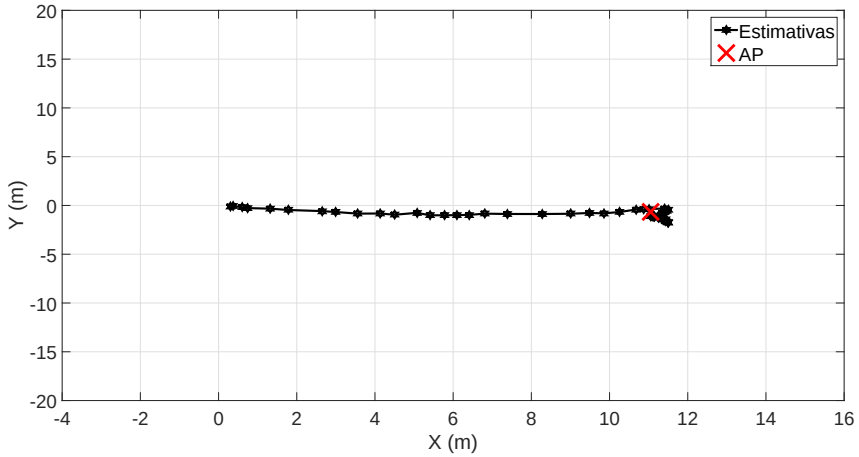


Figura 4.12: Estimação utilizando o método de Relaxação Tomic SOCP com N coordenadas anteriores.

Este tipo de algoritmo (que utiliza optimização convexa) é considerado a artilheria pesada dos algoritmos de multilateração, e por esta razão, com os melhores resultados vem associado um grande tempo de processamento que tipicamente mil vezes superior aos restantes . Esta análise é realizada com mais detalhe na secção 4.5 .

4.4 Algoritmo de localização proposto

Realizando uma abordagem semelhante à utilizada no algoritmo de Tomic, foi desenvolvido um algoritmo mais adaptado ao tipo de dados gerados pelo modelo estudado nesta tese, capaz de obter uma estimação com precisão semelhante à do algoritmo de Tomic mas sendo mais leve computacionalmente.

4.4.1 Relaxação Pedro-Tomic SOCP

O problema foi formulado segundo um modelo de propagação (em dBm) congénere a (4.31),

$$P_i = P_0 - 10\ \gamma\ \log_{10}\frac{\|t - s_i\|}{d_0} - \nu_i\ , i = 1, \ldots, N\ , \tag{4.37}$$

onde P_0 representa o valor de referência da potência a uma distância de referência d_0 mais curta (isto é $\|t - s_i\| \geq d_0$), γ é o PLE e ν_i é modelado com o uma distribuição normal de média μ_i e variância σ_i^2, i.e. $\nu_i \sim \mathcal{N}\left(\mu_i, \sigma_i^2\right)$.

Resolvendo a equação em ordem à distância entre o emissor e o ponto de odometria onde foi realizada a medição obtemos,

$$\|t - s_i\| = e^{\frac{P_0}{\rho}} e^{-\frac{P_i}{\rho}} e^{-\frac{\nu_i}{\rho}} \tag{4.38}$$

em que $\rho = \frac{10\gamma}{\ln 10}$. Aplicando uma aproximação a partir das séries de Taylor[2], ficamos com

$$\|t - s_i\| \approx e^{\frac{P_0}{\rho}} e^{-\frac{P_i}{\rho}} \left(1 - \frac{\nu_i}{\rho}\right) . \tag{4.39}$$

A expressão 4.39 pode ser desenvolvida para

$$\lambda_i \|t - s_i\| \approx e^{\frac{P_0}{\rho}} - \epsilon_i\ , \tag{4.40}$$

onde $\lambda_i = e^{\frac{P_i}{\rho}}$ e $\epsilon_i = e^{\frac{P_0}{\rho}} \frac{\nu_i}{\rho} \sim \mathcal{N}\left(e^{\frac{P_0}{\rho}} \frac{\mu_i}{\rho}, e^{\frac{2P_0}{\rho}} \frac{\sigma_i^2}{\rho^2}\right)$. Elevando ambas as parcelas ao quadrado e denominando $\theta = e^{\frac{2P_0}{\rho}}$, obtemos

$$\theta \approx \lambda_i^2 \|t - s_i\|^2 + 2\lambda_i \|t - s_i\| \epsilon_i + \epsilon_i^2\ . \tag{4.41}$$

Considerando $\epsilon_i \approx 0$ podemos remover o valor de segunda ordem do ruído. Desta forma, chegamos ao nosso problema de mínimos quadrados não-linear e não-convexo

$$(\widehat{t_i}, \widehat{\theta_i}) = \underset{t,\theta}{arg\ min} \sum_{i=1}^{N} \left(\frac{\theta - \lambda_i^2 \|t - s_i\|^2}{2\lambda_i \|t - s_i\|} \right)^2 . \tag{4.42}$$

Desenvolvendo $\|t - s_i\|^2$ e introduzindo uma variável $y = \|t\|^2$, ficamos com

[2] Uma série de Taylor é uma representação de uma função como uma soma infinita de termos que são calculados a partir dos valores das derivadas da função em um único ponto. A função exponencial e^x tem a série de Taylor $e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} \approx 1 + x + \ldots$.

$$\begin{aligned} \underset{t,\theta,y}{\text{minimize}} \ & \sum_{i=1}^{N}\left[\frac{\theta-\lambda_i^2\left(y-2s_i^T t+\|s_i\|^2\right)}{2\lambda_i\|t-s_i\|}\right]^2 . \\ \text{subject to} \ & \ y=\|t\|^2 \end{aligned} \tag{4.43}$$

Podemos ainda introduzir uma variável do tipo epígrafo ζ e aplicar uma relaxação [3] SOCP , obtendo-se

$$\begin{gathered} \underset{t,\theta,y,\zeta}{\text{minimize}} \ \zeta \\ \text{subject to} \\ \left\|\begin{bmatrix} 2\theta-2\lambda_i^2\left(y-2s_i^T t+\|s_i\|^2\right) \\ 4\lambda_i^2\left(y-2s_i^T t+\|s_i\|^2\right)-\zeta \end{bmatrix}\right\| \leq 4\lambda_i^2\left(y-2s_i^T t+\|s_i\|^2\right)+\zeta \ , \ \left\|\begin{bmatrix} 2t \\ y-1 \end{bmatrix}\right\| \leq y+1 \end{gathered} \tag{4.44}$$

Este problema de minimização pode ser resolvido utilizando o procedimento iterativo:

1. Eleger valores empíricos [Rap02], e selecionar a estimativa inicial de $\gamma, \widehat{\gamma}^0 \in [\gamma_{min}, \gamma_{max}]$.

2. Resolver a equação (4.44) e encontrar a estimativa inicial de $t, \widehat{t}^0$.

3. Com recurso a $\widehat{\gamma}^0$ e $\widehat{t}^0$ calcular a estimativa ML de P_0^0 , $\widehat{P}_0^0$, através de

$$\widehat{P}_0^0 = \frac{\sum_{i=1}^{N}\left(P_i - 10\gamma^0 \log_{10}\frac{\left\|\widehat{t}^0-s_i\right\|}{d_0}\right)}{N} . \tag{4.45}$$

4. Calcular o valor da função $f_0 = \sum_{i=1}^{N}\left[\left(\widehat{P}_0^0 - P_i\right) - 10\ \widehat{\gamma}^0\ \log_{10}\frac{\left\|\widehat{t}^0-s_i\right\|}{d_0}\right]^2$ utilizando os valores calculados $\left\{\widehat{t}^0,\ \widehat{P}_0^0,\ \widehat{\gamma}^0\right\}$.

5. Considerar um contador de iterações k e iniciar o mesmo a 1.

6. Utilizar $\widehat{t}^{k-1}$ e $\widehat{L}_0^{k-1}$ para encontrar a estimativa ML para γ , $\widehat{\gamma}$

$$\widehat{\gamma}^k = \frac{\sum_{i=1}^{N} 10\log_{10}\ \frac{\left\|\widehat{t}^{k-1}-s_i\right\|}{d_0}\left(\widehat{P}_0^{k-1} - P_i\right)}{\sum_{i=1}^{N}\left[10\log_{10}\ \frac{\left\|\widehat{t}^{k-1}-s_i\right\|}{d_0}\left(\widehat{P}_0^{k-1} - P_i\right)\right]^2} . \tag{4.46}$$

   Caso $\widehat{\gamma}^k \notin [\gamma_{min},\ \gamma_{max}]$ o processo deve ser parado neste ponto e utilizado $\widehat{t}^{k-1}$ como estimativa final. Caso contrário, estimar θ, calculando $\widehat{\theta}^k = \exp\left(\frac{2\widehat{P}_0^k - 1}{\widehat{\rho}^{k-1}}\right)$.

[3] Relaxação SOCP utilizada : $\left(\frac{x}{y}\right)^2 = z \Rightarrow \left\|\begin{bmatrix} 2x \\ y^2 - z \end{bmatrix}\right\| \leq y^2 + z$

7. Com recurso a $\widehat{\gamma}^k$, $\widehat{P}_0^{k-1}$ e $\widehat{\theta}^k$ resolver

$$\begin{aligned} &\underset{t,y,\zeta}{\text{minimize}}\ \zeta \\ &\text{subject to} \\ &\left\| \begin{bmatrix} 2\widehat{\theta}^k - 2\lambda_i^2\left(y - 2s_i^T t + \|s_i\|^2\right) \\ 4\lambda_i^2\left(y - 2s_i^T t + \|s_i\|^2\right) - \zeta \end{bmatrix} \right\| \leq 4\lambda_i^2\left(y - 2s_i^T t + \|s_i\|^2\right) + \zeta\ , \quad \left\| \begin{bmatrix} 2t \\ y-1 \end{bmatrix} \right\| \leq y+1 \end{aligned} \tag{4.47}$$

obtendo estimativas de t, $\widehat{t}^k$ e de P_0, $\widehat{P}_0^k$.

8. Obter o valor estimado de P_0, $\widehat{P}_0^k$, com os novos valores estimados, resolvendo

$$\widehat{P}_0^k = \frac{\sum_{i=1}^N \left(P_i - 10\gamma^k \log_{10} \frac{\|\widehat{t}^k - s_i\|}{d_0}\right)}{N}\ . \tag{4.48}$$

9. Calcular a função de custo $f_k = \sum_{i=1}^N \left[\left(\widehat{P}_0^k - P_i\right) - 10\ \widehat{\gamma}^k\ \log_{10} \frac{\|\widehat{t}_k - s_i\|}{d_0}\right]^2$. Caso $\frac{|f_k - f_{k-1}|}{f_{k-1}} < \varepsilon$ (ε é o limiar de paragem) ou $k > K_{max}$ (K_{max} representa o número máximo de iterações) deve parar e considerar como estimativa $\widehat{t}^k$; caso contrário repetir o passo 6 e incrementar k.

Na figura 4.13 encontram-se representados os resultados da estimação utilizando o método de Pedro-Tomic, para os quais foi considerado que $P_0^0 = -30$ dBm, $\gamma_0 = 3$ e $d_0 = 1$.

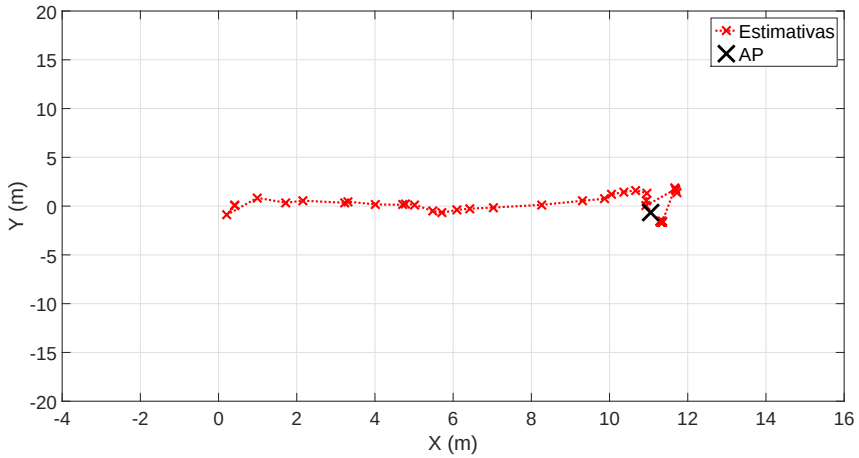


Figura 4.13: Estimação utilizando o método de Relaxação Pedro-Tomic SOCP com N coordenadas anteriores.

Embora os resultados não sejam tão precisos quanto o método de Tomic, este apresenta um peso computacional muito inferior, como é observado na secção 4.5.

4.5 Comparação entre métodos

Para uma melhor percepção dos resultados obtidos utilizando os diversos métodos, foi calculado o erro através da distância euclidiana entre o ponto real (x_t, y_t) e o ponto estimado $(\widehat{x}, \widehat{y})$, que é dado por

$$Erro = \sqrt{(x_t - \widehat{x})^2 + (y_t - \widehat{y})^2}\ . \tag{4.49}$$

Nas figuras 4.14 e 4.15 é possível observar os erros de estimação em função do número de amostras recolhidas fornecidas ao algoritmo. É possível comprovar que os algoritmos de multilateração conseguiram todos convergir a partir de ≈ 17 pontos. Em contraste, os algoritmos de triangulação não conseguiram convergir. Inicialmente, por o utilizador estar distante do emissor e não haver suficientes pontos, os algoritmos apresentaram erros elevados ou não conseguiram sequer calcular uma estimação.

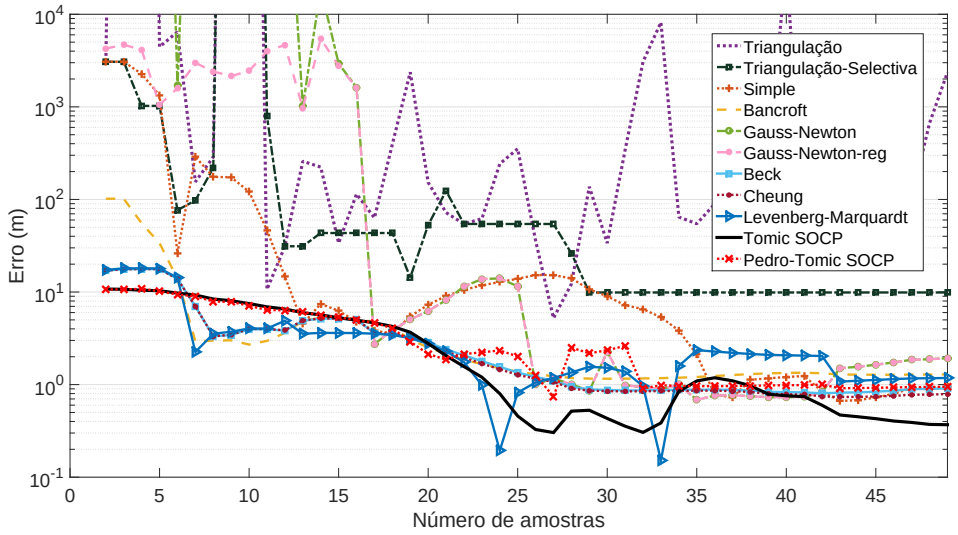


Figura 4.14: Comparação de erros dos diversos algoritmos com dados obtidos num One-Plus 2.

O algoritmo de Beck e Cheung apresentaram resultados muito semelhantes, sendo as únicas diferenças visíveis nos pontos iniciais, pois a partir do momento que os algoritmos utilizam mais do que 4 pontos as linhas vermelha e rosa encontram-se sobrepostas. O método de Levenberg-Marquardt demonstrou um erro semelhante ao Beck e Cheun mas obteve um peso computacional mais elevado, como é possível observar na tabela 4.3. No cenário ilustrado no anexo H obteve bons resultados, superando os outros dois algoritmos.

Utilizando algoritmos de otimização convexa, como o de Tomic, observam-se resultados mais precisos mas com custos computacionais mais elevados. A versão Pedro-Tomic apresenta-se como uma nova solução que atinge também resultados de elevada precisão (segundo mais preciso), com um desempenho a nível de processamento três vezes

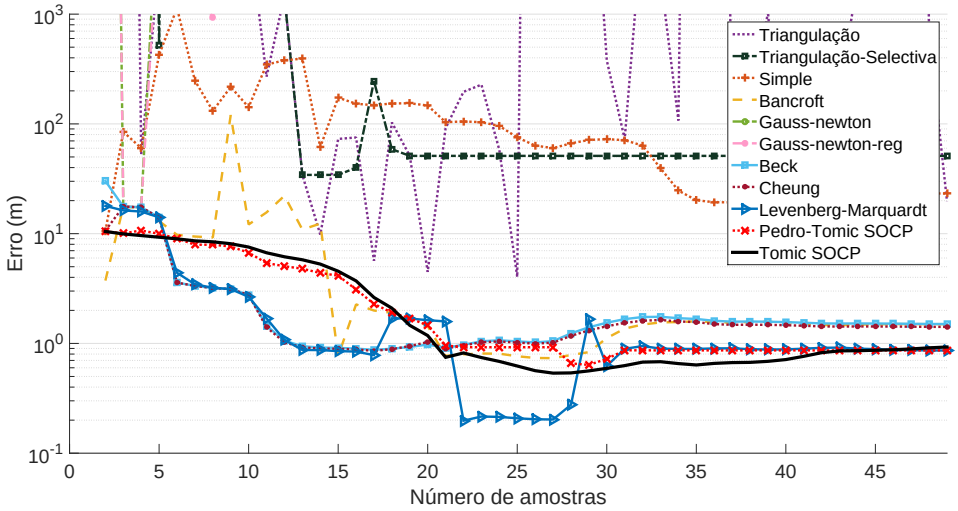


Figura 4.15: Comparação de erros dos diversos algoritmos com dados obtidos num Sony XPERIA Z1 Compact.

mais rápido que a versão original de Tomic. Mesmo assim, ainda é quase duas ordens de grandeza mais lento que o método de Levenberg-Marquardt.

É importante relembrar que os algoritmos testados não só lidam com os erros provenientes das medições de potência, mas também com os erros do módulo de localização.

Os resultados dos algoritmos utilizando as medições do Sony XPERIA foram bastante semelhantes aos do OnePlus 2, tendo havido mais pontos em que os algoritmos de trilateração tiveram dificuldades em calcular uma estimação.

Para realizar uma comparação global entre os diversos métodos, foi calculado o erro quadrático médio do conjunto de pontos em dois grupos :

1. Fase inicial: Para este conjunto foram considerados os valores de distâncias euclidianas com mais de 3 medições até $\frac{N-3}{2}$, ou seja, a primeira metade de medições a partir do momento em que é possível correr algoritmos.

2. Fase funcional: De forma análoga ao caso anterior, para este grupo consideraram-se os restantes pontos posteriores, ou seja, de $\frac{N-3}{2}$ até N, usando os pontos iniciais no cálculo do algoritmo.

A métrica de desempenho *Root Mean Square Error* (RMSE)(erro quadrático médio), é definida como

$$\sqrt{\sum_{i=1}^{M} \frac{\left\| t - \widehat{t_i} \right\|^2}{M}}, \tag{4.50}$$

onde M denota o tamanho do conjunto de amostras utilizadas no cálculo do RMSE, $\widehat{t_i}$ o valor da estimativa e t a posição real da fonte.

Tabela 4.3: Análise comparativa de tempos de execução e desempenho de precisão dos métodos estudados.

Algoritmo	**RMSE$_1$ (m)**	**RMSE$_2$ (m)**	**Tempo de execução (ms)**	
			MatLab	Android
Triangulação	∞	∞	4,86	<1
Triangulação-Selectiva	∞	35,810	3,110	<1
Simple	687,74	24,218	16,40	N/A
Bancroft	30,075	1,3575	19,95	N/A
Gauss-newton	∞	$95{,}95\times10^3$	40,50	N/A
Gauss-newton-reg	5350,3	$22{,}30\times10^3$	39,90	N/A
Beck	8,7120	1,2349	42,15	N/A
Cheung	7,5795	1,1602	28,50	N/A
Levenberg-Marquardt	7,7380	1,2602	325,0	13,472
Tomic SOCP	6,6390	0,7031	$8{,}65\times10^4$	N/A
Pedro-Tomic SOCP	6,4885	1.0855	$2{,}44\times10^4$	N/A

Na tabela 4.3 encontra-se presente o tempo que cada método de estimação demorou a correr e o desempenho numa fase inicial e numa fase de funcionamento. Visto que a aplicação guarda no máximo 35 pontos para um dado AP, na estimação do tempo de execução, foi também considerado este máximo. Para o cálculo do RMSE$_1$ foram utilizados os valores das medições da fase inicial com ambos os dispositivos, e de forma semelhante, na realização de $RMSE_2$ foram usados os dados da fase funcional. Na realização desta tabela, os algoritmos foram corridos em MATLAB 100 vezes para cada conjunto de dados (de ambos os smartphones) e foi realizada a média dos tempos obtidos. De forma semelhante, os testes aos algoritmos implementados em Android (detalhado na secção seguinte) foram testados, e obtiveram-se os resultados observados na coluna da direita. Os tempos medidos para os algoritmos de Trilateração, Trilateração Seletiva e Média, foram inferiores a 1 ms, a resolução mínima de medida suportada. No algoritmo de Levenberg-Marquardt a biblioteca matemática *apache* utilizada na aplicação, foi mais rápida que a biblioteca nativa do MatLab. Na secção seguinte é apresentado um resumo da implementação da interface FindAP no protótipo desenvolvido em Android.

4.6 Protótipo Android - Interface FindAP

A interface FindAP permite ao smartphone, no modo não cooperativo, estimar a localização de um AP dentro do alcance rádio. Na figura 4.16 pode-se observar a pasta da interface gráfica desta interface. As coordenadas na parte superior representam a localização do AP selecionado (X_t, Y_t) e as coordenadas na parte inferior indicam a posição atual do utilizador (X_c, Y_c) (proveniente do nó de Odometria).

Utilizando o ângulo de orientação actual θ_a do dispositivo e as coordenadas mencionadas, a orientação do utilizador para o AP é dada por,

Figura 4.16: Interface da aplicação quando se tenta localizar um AP.

$$\theta_a - arctan2(Y_t - Y_c, X_t - X_c)\,, \tag{4.51}$$

onde $arctan2(Y_t - Y_c, X_t - X_c)$ representa $arctan(\frac{Y_t - Y_c}{X_t - X_c})$ com todas as proteções associadas (i.e o denominador ou numerador ser *NaN*, zero, infinito entre outras) e devolvendo o resultado nos 4 quadrantes. A equação (4.51) é usada para mudar a orientação da seta apresentada na interface da aplicação, sendo atualizada sempre que existe um valor novo de coordenadas.

A figura 4.17 evidencia os principais módulos usados na implementação da interface FindAP. Resumidamente, utiliza a classe Odom que realiza a Odometria do dispositivo, a classe LatLonProvider, que utiliza a API *Location* da Google para obter as coordenadas de latitude e longitude, recebe eventos do scan WiFi e com esta informação estima a posição do AP.

A classe FindApFragment é o fragmento que contém as TextViews e ImageViews necessárias para a exibição gráfica. Esta classe implementa a interface *UpdateNotifier*, o que significa que implementa o método *update* que é invocado sempre que é gerado um scan WiFi. Este método pede ao odómetro as coordenadas relativas do utilizador e gera um PositionPoint com estas coordenadas e com a leitura WiFi. Em seguida, adiciona este novo ponto à sua instância da classe PositionData. Caso já exista uma estimação (tipicamente existe quando se obtêm 3 ou mais PositionPoints), actualiza as coordenadas da posição do AP, bem como o ângulo da ImageView (seta) conforme explicado anteriormente. No final actualiza sempre as coordenadas atuais do utilizador. Esta classe ainda permite que o utilizador reinicie o processo e troque de receptor para emissor, ficando em modo de *HotSpot*.

A classe PositionData é responsável pela estimação do emissor. Sempre que é inserido um PositionPoint, esta tenta realizar a estimação no modo selecionado pelo utilizador.

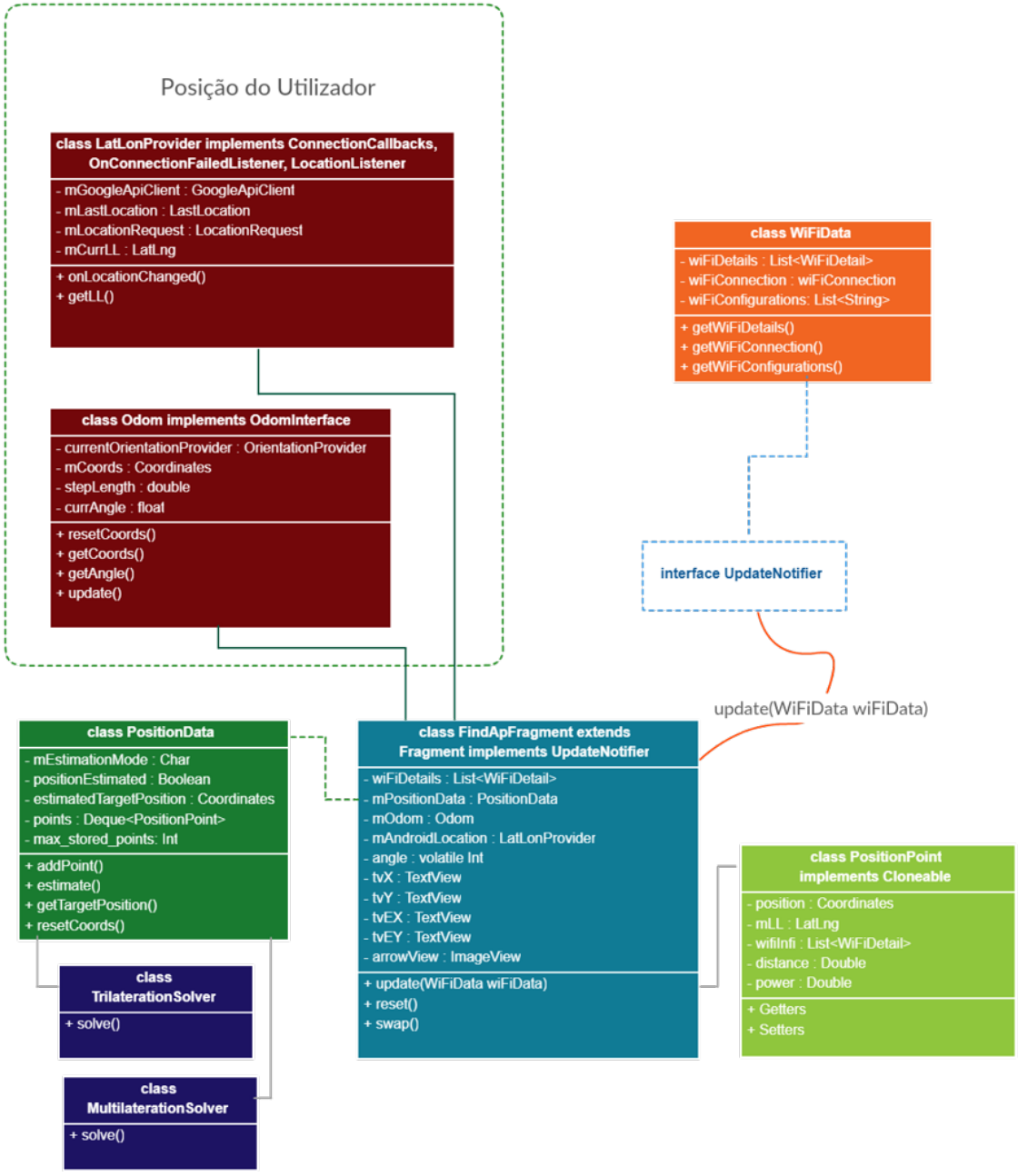


Figura 4.17: Integração dos módulos de software da interface FindAP.

Para isso, é utilizada a classe TrilaterationSolver ou a classe MultilaterarionSolver. No modo de trilateração foi implementado o algoritmo de trilateração selectiva, estudado anteriormente neste capítulo, enquanto na classe MultilaterationSolver utilizou-se a biblioteca de *org.apache.commons.math3.fitting.leastsquares.Levenberg MarquardtOptimizer*, com um número máximo de iterações $K_{max} = 1000$. Em ambos os algoritmos são utilizados um número máximo de pontos, que nos testes realizados se considerou $max\_stored = 35$.

Caso se pretenda estimar a localização de todos os APs presentes na leitura WiFiData, basta iterar a List<WiFiDetail> no PositionData, criando uma estimativa para cada AP.

Capítulo 5

Cooperação entre dispositivos

Este capítulo analisa diversos métodos cooperativos para estimar a localização. Para além dos ganhos a nível de precisão com poucas medições, estes sistemas permitem reduzir o tempo de estimação e transformar as coordenadas relativas estimadas em coordenadas absolutas utilizando um mecanismo de referenciação.

São apresentados dois métodos de distribuição de informação que foram implementados. Embora os algoritmos de cooperação tenham sido testados em MatLab, a sua implementação na aplicação/servidor juntamente com os mecanismos de distribuição e respectivos testes em tempo real foram deixados para trabalho futuro.

O sistema de cooperação assemelha-se a três esquemas apresentados no capítulo 2: Centaur, ZCL e Social-Loc. Contudo, o Centaur e o ZCL utilizam *hardware* que não é característico dos *smartphones* no mercado e o Social-LC não apresenta nenhuma solução para a junção de informação de diferentes dispositivos.

A estimação cooperativa requer a existência de uma rede, composta por um grupo de *smartphones*, um número de alvos e um servidor. Na rede cada *smartphone* pode comunicar com qualquer outro *smartphone* ou AP dentro do seu alcance rádio, e todos os alvos podem ser localizados simultaneamente.

A cooperação apresenta as seguintes vantagens :

- Velocidade de convergência: Os algoritmos cooperativos conseguem convergir mais rapidamente pois obtêm várias medições em simultâneo, conseguindo obter mais pontos de medição do que um smarphone isolado. Caso um utilizador entre num grupo que já se encontra em operação há algum tempo, consegue estimar os elementos em seu redor utilizando a informação dos vizinhos.

- Gasto energético: Num sistema cooperativo, o processamento pode ser distribuído

pelos dispositivos ou ser corrido num servidor. Desta forma, os dispositivos necessitam de correr os algoritmos menos vezes, o que se reflete numa poupança de bateria.

Contudo, para se implementar um esquema cooperativo são necessárias mais informações. Podem ser descritas como desvantagens:

- Referenciação: As coordenadas relativas estimadas por *smartphones* têm como ponto inicial o momento em que o utilizador inicializou a aplicação. Para que os dispositivos possam utilizar as medições uns dos outros, é necessário que estas coordenadas sejam representáveis no mesmo referencial, ou seja, que sejam transformadas em coordenadas absolutas.

- Cálculo do ganho de recepção G_r: Cada dispositivo utiliza uma antena diferente. As antenas presentes nos *smartphones*, dependendo do modelo, são diferentes a nível de hardware. Desta forma, existe a necessidade de se estimar este parâmetro para partilhar a informação. Caso G_r seja mal estimado, a precisão é directamente afectada. Este efeito dos ganhos das antenas foi previamente observado no gráfico 4.1, para os dois terminais usados nos testes.

- Arquitetura de comunicação: Para realizar a troca de informação, é necessário que seja implementado um protocolo de partilha de coordenadas centralizado ou distribuído.

Os pontos enumerados são abordados nas subsecções seguintes, sendo apresentadas soluções para os mesmos.

5.1 Cálculo do ganho de recepção

O ganho de recepção G_r e ganho de transmissão G_t que muitas vezes é considerado como parte da potência de transmissão P_t, tem uma grande influência na potência do sinal recebido.

Tendo em conta que muitas das fontes são totalmente desconhecidas, não é possível assumir que se sabe G_t e P_t. Contudo podemos tentar estimar o G_r do *smartphone* para minimizar as variações e para ser mais fácil juntar informações de diversos dispositivos.

Visto que o número de variáveis é bastante superior ao número de valores conhecidos, não é possível calcular este valor de uma forma direta, mesmo num esquema de *crowdsourcing*. Uma das soluções é seguir uma abordagem semelhante à usada no algoritmo de Pedro-Tomic, onde se transforma num problema convexo e se estima este valor. Esta abordagem tem a desvantagem de ser pesada a nível computacional. Outra abordagem é considerar um emissor com P_t conhecida no ponto de entrada de um edifício, e cada vez que um utilizador entrar, estimar G_r e guardar este valor. Corrigindo os valores de

RSS usados nos algoritmos com o desvio medido, é possível combinar medidas de vários terminais.

5.2 Referenciação

A referenciação é necessária para se transformar as coordenadas relativas obtida individualmente por cada dispositivo, em coordenadas absolutas partilháveis.

Para um utilizador obter as suas coordenadas absolutas numa região *indoor* sem serviço de localização ativo são possíveis duas abordagens: usar a última localização conhecida quando entra na zona, ou existirem APs com coordenadas conhecidas.

A abordagem mais simples ocorre quando o utilizador vem de uma zona onde é possível obter as suas coordenadas através da localização Android. Quando entrar num edifício sem localização, é gerado um evento de mudança de precisão na API de localização Android e a aplicação comuta para o modo de odometria, sendo o ponto de entrada referenciado como a última posição conhecida.

A segunda abordagem requer que existam APs com coordenadas conhecidas. Vamos supor que um utilizador A conhece a posição relativa de um utilizador B . Caso também conheça a posição relativa de um AP C com coordenadas absolutas conhecidas, é possível saber as coordenadas absolutas de B. Trocando esta informação através de um servidor ou diretamente entre utilizadores, é possível gerar uma teia de coordenadas absolutas na qual se sabem as coordenadas de todos os utilizadores.

Na figura 5.1 está representada uma rede na qual existe 1 AP com coordenadas conhecidas, 25 utilizadores e 4 APs com posições desconhecidas. Foi considerado que os utilizadores conseguem comunicar a 3 metros de distâncias. Caso a aplicação determine as posições relativas entre utilizadores, é possível calcular as posições absolutas a partir da posição do AP referenciado, através da cinemática inversa detalhada na secção 5.4. Relembra-se que todos os terminais usam a mesma orientação espacial para definir a longitude e latitude.

5.3 Modos de funcionamento cooperativo

Nesta secção são abordados dois modos de distribuição de conteúdos, utilizados para o funcionamento cooperativo. Em primeiro lugar é analisado o modo centralizado, baseado numa arquitetura com um servidor. Em seguida, é estudada uma organização distribuída.

5.3.1 Sistema Centralizado

Um sistema centralizado requer que todos os dispositivos enviem as suas medições para um servidor. Na figura 5.2 está representado um exemplo com 4 utilizadores num grupo, ligados ao servidor, num esquema centralizado.

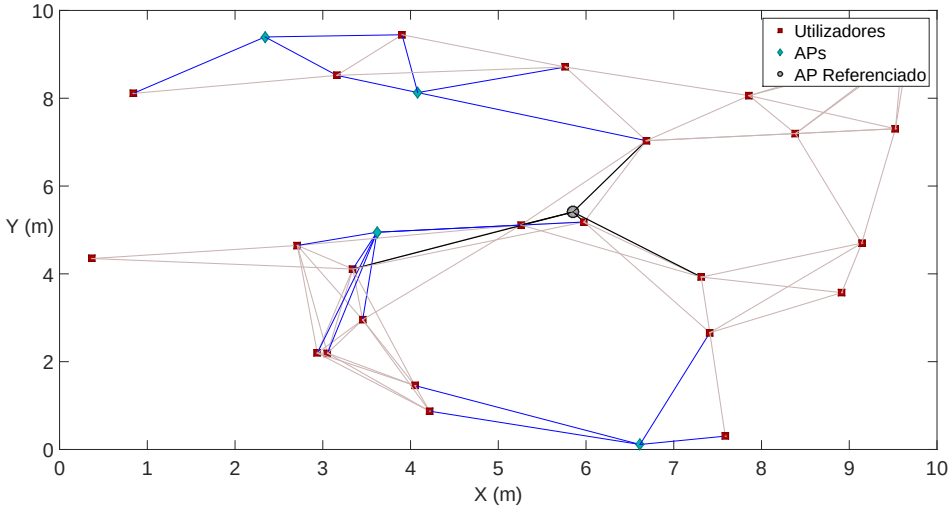


Figura 5.1: Exemplo de rede referenciada.

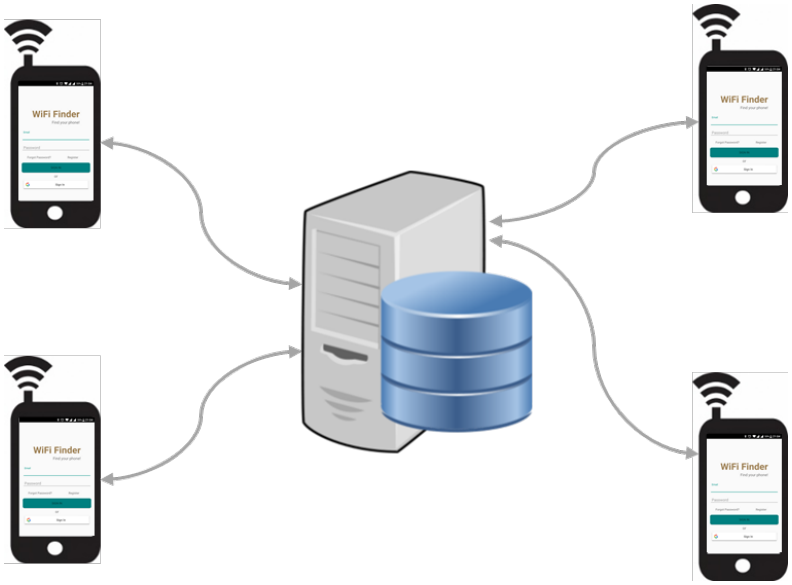


Figura 5.2: Esquema do servidor centralizado utilizado para melhorar as estimações realizadas.

Na aplicação desenvolvida na dissertação, as coordenadas e leituras dos utilizadores são transmitidas para um servidor[1] implementado em PHP, que se limita a guardar esta informação. Este servidor contém uma base de dados MySQL, cujo esquema se encontra detalhado na figura 5.3. A tabela *USER* contém a informação referente a um utilizador, que pode estar associado a um ou mais grupos. A tabela *GROUPS* identifica um grupo que pode conter um ou mais utilizadores. Devido às duas tabelas referidas necessitarem de uma ligação N para N, foi necessário criar a tabela *USER-GROUPS*.

A tabela *POSITION DATA* guarda a informação referente a um AP, sendo as várias medições de um determinado utilizador de um AP específico guardadas na tabela *POSITION POINT*. Nesta tabela os campos *USER X* e *USER Y* representam as coordenadas do utilizador quando foi realizada a medição e os campos *EST X* e *EST Y* guardam a estimação gerada com todas as medições até ao momento.

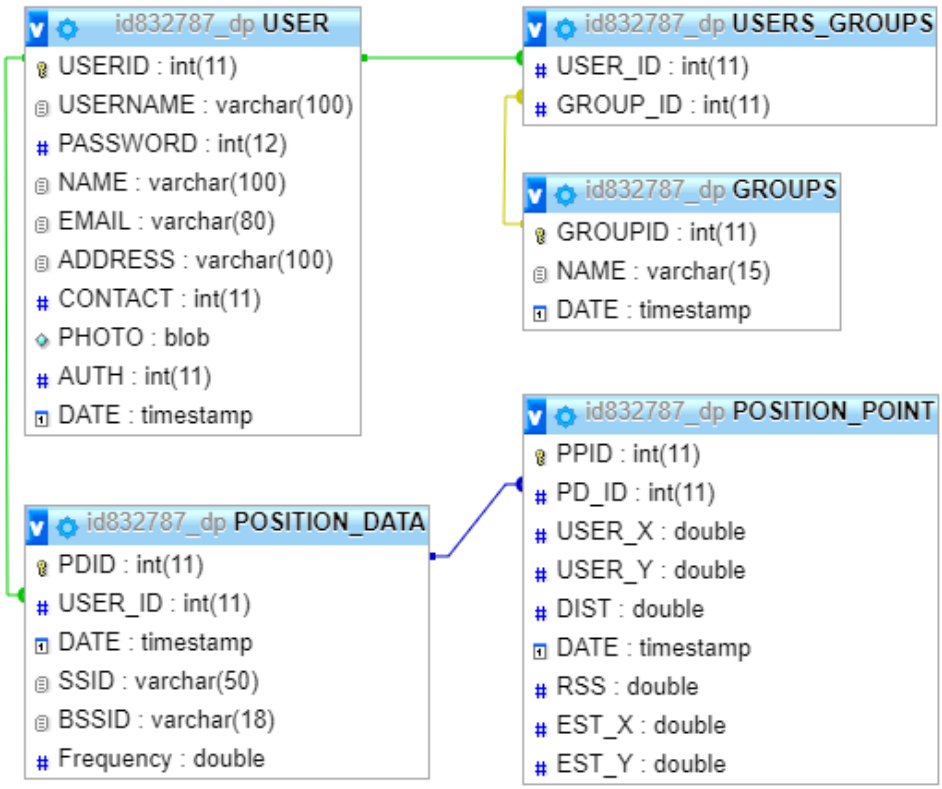


Figura 5.3: Esquema da Base de Dados do Sistema Desenvolvido.

A comunicação entre os utilizadores e o servidor foi implementada usando *WebServices* REST. Por exemplo, para realizar o Login na aplicação, esta envia um POST com o nome de utilizador e a palavra-passe[2]. Quando o servidor recebe esta informação em

[1] O servidor pode ser acedido em `http://dariopedro.com/wififinder.php`. Neste *site* é possível fazer download da aplicação directamente para o telemóvel

[2] Por razões de segurança, a palavra-passe é digirida utilizando *Message-Digest algorithm 5* (MD5).

`http://dariopedro.000webhostapp.com/login.php`, verifica se a mensagem foi enviada de um dispositivo Android e faz uma query à base de dados, sendo o resultado retornado para a aplicação.

Numa arquitetura centralizada, sempre que os *smartphones* geram um novo ponto com a informação de odometria e do scan do meio evolvente; ou enviam-no para o servidor; ou agrupam num conjunto de pontos, para reduzir o *overhead*. O servidor pode guardar a informação de todos os utilizadores na base de dados, e correr um algoritmo de estimação quando for necessário. No servidor também podem haver pontos de referência de APs guardados, o que possibilita que o utilizador abra a aplicação num ambiente *indoor* e consiga saber a sua posição absoluta ao estimar a sua posição relativa a um destes APs. Nesta dissertação, os testes cooperativos foram realizados usando o MatLab com dados exportados da aplicação, mas os algoritmos poderiam ser facilmente implementados no servidor.

A arquitetura onde se realiza as estimações no servidor apresenta grandes vantagens a nível energético. Os problemas de bateria dos *smartphones* não são novidade, e estar constantemente a processar todos os pontos recolhidos tem um peso computacional significativo, que pode ser passado para o lado do servidor.

5.3.2 Sistema Distribuído

Num sistema distribuído, os utilizadores de um grupo trocam informação entre si diretamente sem recorrem a um servidor, como está ilustrado na figura 5.4. Existe contudo a necessidade de existir um mecanismo para que os utilizadores se consigam autentificar na aplicação, ganhando acesso aos grupos em que estão inseridos.

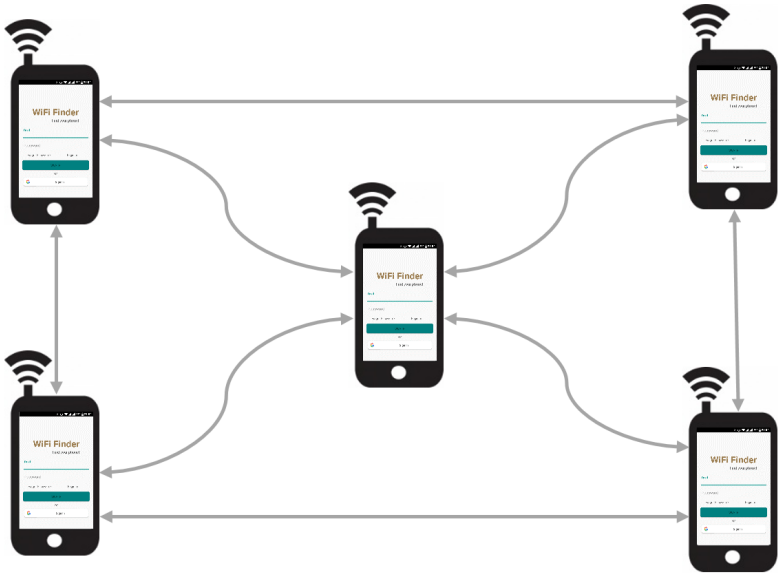


Figura 5.4: Esquema do Sistema Distribuído utilizado para melhorar as estimações realizadas.

Para trocarem a informação, poderiam usar um esquema de multicast para um endereço e porto conhecidos, na qual todos os utilizadores ligados a uma rede recebem a informação transmitida, e caso necessário utilizarem-na para melhorar as suas estimações. Nesta implementação alternativa seria necessário transmitir as medições efetuadas para os outros nós, podendo-se sectorizar as zonas para ser mais escalável. A troca de informação foi implementada de forma semelhante ao chat MultiCast explicado no Manual da aplicação no Anexo D. De forma análoga ao sistema centralizado, os algoritmos cooperativos testados em MatLab poderiam ser implementados na aplicação.

A arquitetura de estimação nos dispositivos tem a vantagem de não haver necessidade de existir um servidor, desde que se adopte um mecanismo de autenticação que funcione sem servidores (e.g. RSA).

5.4 Desempenho dos algoritmos de localização no modo cooperativo

Considerando um conjunto de J utilizadores em que todos conhecem a sua posição face a um ponto de referência comum, B, é possível combinar os vetores de cada *smartphone* individual para correr o algoritmo de localização. As coordenadas a usar para o terminal j devem ser, $S'_j = S_j + \overrightarrow{S_{j0}B}$, onde $\overrightarrow{S_{j0}B}$ é o vetor do ponto de partida do utilizador j até ao ponto conhecido B. Em seguida é necessário minimizar os efeitos que os diferentes tipos de *smartphones* têm nos valores RSS medidos, como é o caso de G_r. No algoritmo desenvolvido foi estimado o valor α (equação 4.1) para cada terminal, utilizando-se o seguinte procedimento:

1. Calcular a posição do dispositivo com todas as medições (neste ponto pode ser utilizado qualquer algoritmo de estimação estudado anteriormente).

2. Utilizar o método de Tomic (estudado em 4.3.2.9) para se obter uma estimativa do valor de P_0, $\widehat{P_0}$ e uma estimativa de γ , $\widehat{\gamma}$.

3. Com recurso as valores estimados na alínea anterior, estimar o valor da distância de cada medição, com

$$\widehat{d_i} = d_0 10^{\frac{\widehat{P_0}-P_i}{10\widehat{\gamma}}} , \tag{5.1}$$

   onde d_0 é a referência de curta distância (tipicamente considerada 1m[Tom17]) e P_i representa os valores RSS medidos.

4. Resolvendo a equação (4.1) em ordem a α obtemos

$$\widehat{\alpha_i} = 20\log_{10}\left(\widehat{d_i} f\right) - |P_i| . \tag{5.2}$$

Com os novos valores $\widehat{\alpha_i}$, é possível correr os algoritmos anteriormente estudados, com a influência de cada *smartphone* mitigada.

5.4.1 Teste realizado

Utilizando o algoritmo acima descrito, e misturando os dados dos dois *smartphones* utilizados no capítulo anterior, foi calculado o erro de estimação usando o algoritmo de Range-Bancroft, o algoritmo de Beck, o algoritmo de Cheung, o algoritmo de GN Regularizado, algoritmo de LM, a Relaxação de Tomic SOCP e o algoritmo de PT. Foram obtidos os resultados presentes na figura 5.5. Cada amostra na figura corresponde a um conjunto de duas medições (uma de cada dispositivo).

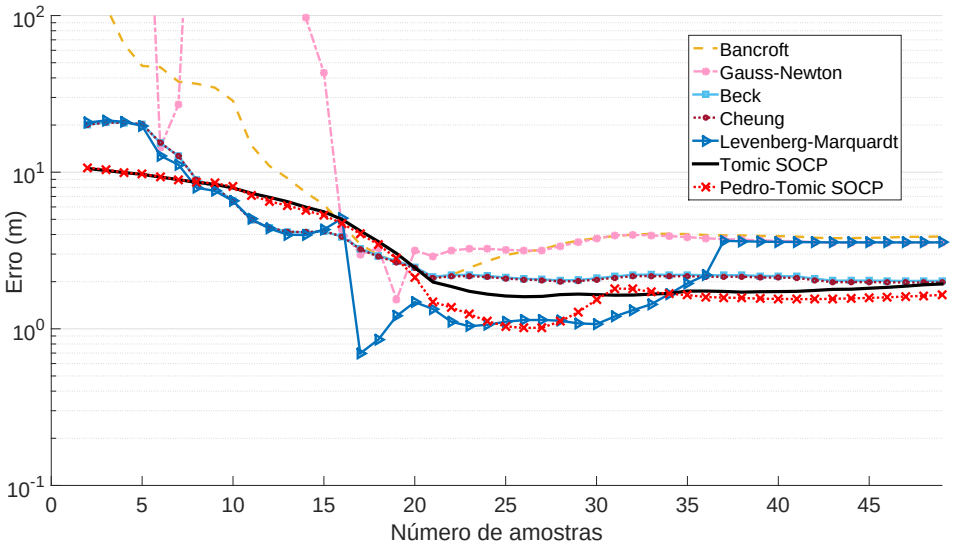


Figura 5.5: Comparação de erros dos diversos algoritmos com dados usando um esquema cooperativo.

Comparando com as figuras 4.14 e 4.15, é interessante observar que os algoritmos convergem muito mais rapidamente. Para além disso, alguns algoritmos que anteriormente divergiam conseguem convergir com qualquer número de amostras. Porém, ao utilizar este algoritmo a precisão piora ficando com um desempenho intermédio entre o melhor e o pior desempenho individual. Esta diversidade decorre dos diferentes tipos de sensores de cada *smartphone* a realizar Odometria e dos diferentes tipos de antenas.

Na tabela 5.1 é possível observar o desempenho dos diversos algoritmos utilizando os dados num modo cooperativo e realizar uma comparação com os resultados anteriormente obtidos. Os valores nas colunas da Média Individual representam uma média dos resultados obtidos no capítulo anterior, quando os algortimos foram testados individualmente em cada *smartphone*, enquanto na coluna Cooperativa são apresentados os resultados com o novo algoritmo.

Nos testes realizados, o algoritmo Pedro-Tomic foi o que obteve melhores resultados,

Tabela 5.1: Análise comparativa do desempenho dos algoritmos em modo Individual e Cooperativo.

Algoritmo	**Média Individual**		**Cooperativo**	
	$RMSE_1$ (m)	$RMSE_2$ (m)	$RMSE_1$ (m)	$RMSE_2$ (m)
Bancroft	30,075	1,3575	54,17	3,90
Gauss-newton-reg	5350	$22{,}30\times10^3$	$1{,}79\times10^6$	3,72
Beck	8,712	1,2349	10,27	2,16
Cheung	7,5795	1,1602	10,26	2,12
Levenberg-Marquardt	7,738	1,2602	9,99	2,86
Tomic SOCP	6,639	0,7031	6,85	1,78
Pedro-Tomic SOCP	6,4885	1,0855	6,78	1,57

sendo o RMSE da segunda metade das suas medições aproximadamente $1,57$ m, embora a tendência seja para os dois algoritmos baseados em otimização complexa terem desempenhos semelhantes.

Para os resultados serem mais facilmente observados, na figura 5.6 são apresentados os resultados dos algoritmos GN, PT e LM em cada dispositivo individualmente e no modo cooperativo. No modo cooperativo assume-se que os dois *smartphones* estão a correr o algoritmo concorrentemente e de forma sincronizada. O algoritmo GN foi analisado por ser o que apresenta maiores diferenças entre os dispositivos, sendo assim mais fácil verificar que o algoritmo cooperativo consegue minimizar erros elevados provenientes de alguns dispositivos. No gráfico da figura 5.6 não é possível observar os resultados do Xperia GN porque o erro nunca baixou de 10^4m. Deste modo, confirma-se que o algoritmo cooperativo conseguiu anular este erro, não ultrapassando 10m de erro depois de alcançar a estabilidade, isto é, depois de ter mais de 15 amostras.

Observando os resultados dos algoritmos LM e PT, também é possível confirmar que estes atingem a estabilidade mais cedo no modo cooperativo, por combinarem medições de dois terminais. Contudo, têm uma precisão inferior com mais amostras.

Na figura 5.7 estão representados os resultados dos algoritmos cooperativos, num cenário em que um dispositivo A se ligou a uma rede na qual já existia um outro dispositivo B. A figura 5.7a considerou que o OnePlus2 já se apresentava ligado à mais tempo, tendo realizado 50 amostras. Em seguida, outro utilizador com um Sony Xperia começou a realizar estimações da sua posição, ou seja, a primeira amostra do gráfico contém informação da primeira medição do Sony Xperia mais as 50 medições do OnePlus 2. Analogamente, a figura 5.7a demonstra os resultados do OnePlus2 utilizando dados obtidos pelo Sony Xperia. Desde modo, podemos concluir que o algoritmo cooperativo permite que um utilizador obtenha uma estimação com superior precisão face ao modo não cooperativo numa fase inicial.

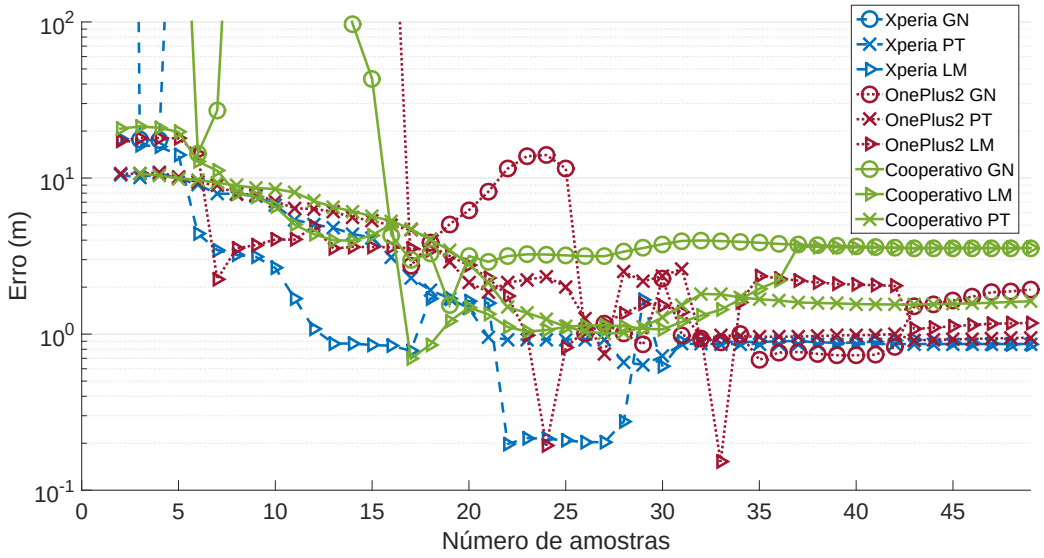
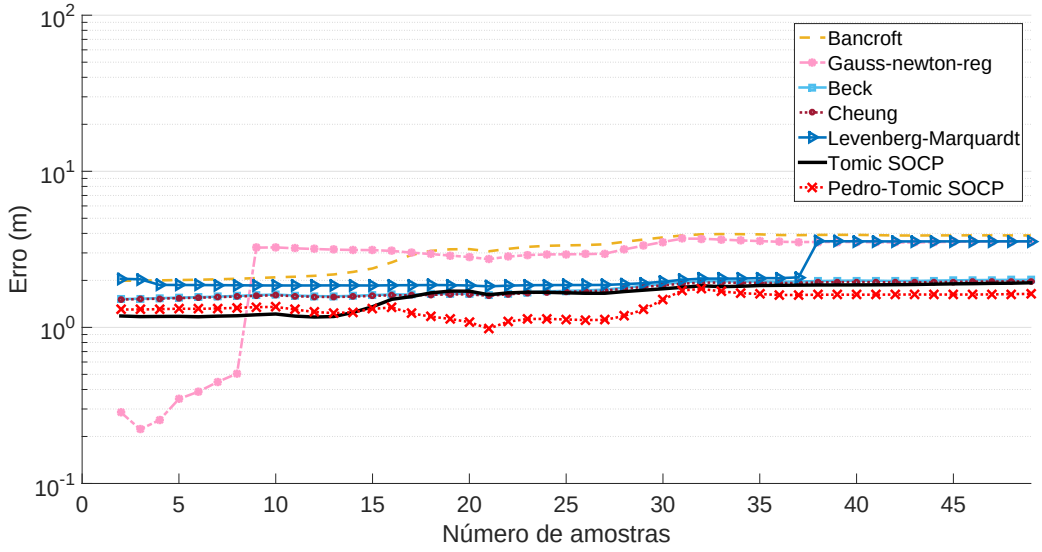
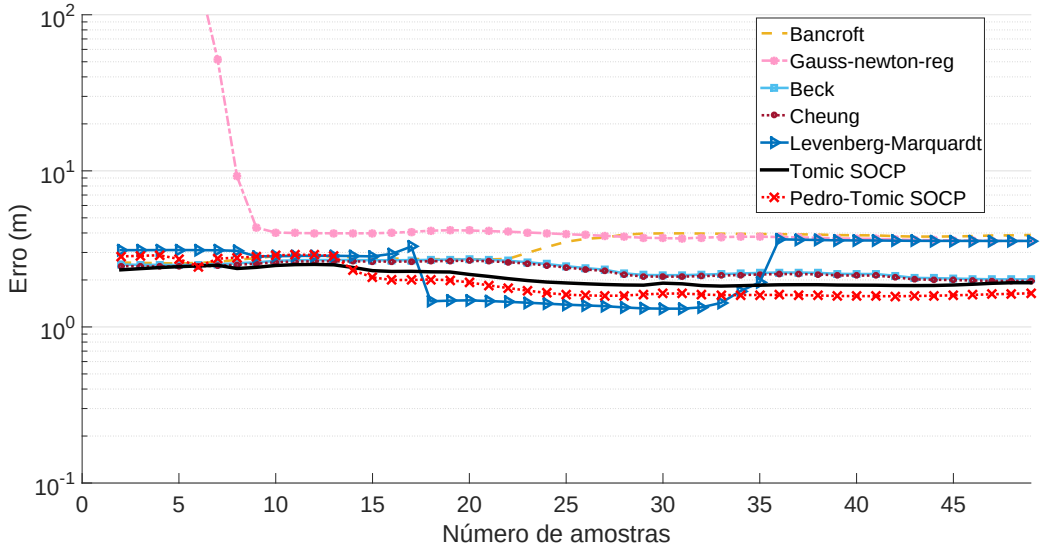


Figura 5.6: Comparação de erros entre o modo individual e cooperativo correndo o algoritmo de GN, LM e PT.



a Estimações do Sony Xperia utilizando informação do OnePlus2



b Estimações do OnePlus2 utilizando informação do Sony Xperia

Figura 5.7: Algoritmos cooperativos com dados *a priori* de outro dispositivo.

Capítulo 6

Conclusões e Trabalho Futuro

Este capítulo resume as principais contribuições que foram obtidas através dos resultados da tese e propõe algumas orientações previstas para pesquisas futuras sobre o tema destacando na secção 6.2 os pontos em que a aplicação desenvolvida pode ser melhorada.

6.1 Conclusões

O principal objetivo desta dissertação foi criar uma aplicação Android que permitisse o posicionamento de *smartphones* em ambiente indoor. Visto que tipicamente todos nós utilizamos um *smartphone*, foi aproveitada a informação num regime cooperativo.

Numa fase inicial foi realizado um processo de pesquisa relacionado com as tecnologias sem fios que poderiam ser utilizadas neste tipo de aplicação, as métricas e Algoritmos que se podem extrair a partir dos vários tipos de sinais, e os algoritmos e sistema de Localização mais conhecidos e semelhantes na literatura.

Em seguida foi desenvolvido um módulo capaz de utilizar sensores inercias dos *smartphones* modernos que permite obter a orientação do dispositivo e detetar os passos do utilizador. Com esta informação foi possível criar um nó que realiza a Odometria.

Utilizando a informação WiFi recebida em cada dispositivo e a posição relativa calculada pelo nó de Odometria, foram implementados os métodos de estimação habituais e desenvolvido um novo método (Pedro-Tomic) que melhor se adapta ao tipo de dados gerados pelos *smartphones* e que aproveita as principais características dos algoritmos. Posteriormente, foram realizados testes em diversos cenários que confirmam a viabilidade da aplicação num contexto real, com precisão a rondar os 2 m.

Foi ainda verificado que a robustez e a velocidade de convergência destes algoritmos pode ser melhorada utilizando um esquema cooperativo, para o qual também foi desenvolvido um algoritmo cooperativo inovador.

Ao longo de cada capítulo foi sempre realizada uma análise comparativa onde o desempenho dos algoritmos apresentados foi testado com dados experimentais obtidos através da aplicação desenvolvida sugerindo os resultados obtidos que os algoritmos funcionam bem no ambiente considerado.

6.2 Trabalho Futuro

Existem vários pontos que podem ser desenvolvidos de modo a melhorar a aplicação no futuro. Abaixo estão apresentados os principais:

- Novos métodos de estimação do tamanho do passo do utilizador [LS13; Li+12; Zij04]. Esta informação pode ser atualizada num esquema de realimentação do nó de Odometria, melhorando assim o desempenho a nível de precisão do mesmo.
- Correção das últimas N posições de Odometria após variações significativas do tamanho de passo e/ou efeitos de orientação. [Xia+15c].
- Exportação constante de informação para o servidor. Com um esquema que consiga exportar a nova informação para o servidor, é possível mover a decisão e estimação para o servidor e/ou criar métodos que corram em paralelo.
- Segurança na autentificação. O Login da aplicação apresenta-se como um método de identificação, mas a troca de dados não é realizada com a devida protecção.
- Registo de novos grupos a partir da aplicação.
- Percepção da melhor combinação de sensores na realização da Odometria, dependendo do telemóvel.
- Melhorar a troca de informação em modo cooperativo com *Multicast* (ex: segurança).

De um modo geral e mais do ponto de vista de investigação, seria interessante estudar e testar os seguintes tópicos:

- Implementação de todos os algoritmos de estimação em Android para uma comparação temporal mais fidedigna.
- Continuação de testes em vários ambientes *indoor*.
- Criação de novos algoritmos que melhor se adaptam ao tipo de dados do esquema apresentado nesta dissertação.
- Novos métodos cooperativos.
- Testes com grupos de utilizadores mais numerosos em regiões espaciais diferentes.
- Integração dos modelos apresentados com novos tipos de rádio frequência.

Bibliografia

[Aer] *AeroScout Company, WiFi Based RTLS*. 2011. URL: http://www.aeroscout.com (acedido em 13/01/2017).

[AP03] B. Alavi e K. Pahlavan. "Modeling of the distance error for indoor geolocation". Em: *IEEE Wireless Communications and Networking Conference, WCNC*. Vol. 1. 2003, pp. 668–672. ISBN: 0780377001. DOI: 10.1109/WCNC.2003.1200430.

[AL+09] S. Ali-Löytty, Tommi Perälä, V. Honkavirta e R. Piché. "Fingerprint Kalman Filter in indoor positioning applications". Em: *Proceedings of the IEEE International Conference on Control Applications*. 2009, pp. 1678–1683. ISBN: 9781424446025. DOI: 10.1109/CCA.2009.5281069.

[All15] W.-F. Alliance. *Wi-Fi Aware: Discover the World Nearby*. 2015. URL: http://www.wi-fi.org/discover-wi-fi/wi-fi-aware (acedido em 12/01/2017).

[Ang+12] A. D. Angelis, H Peter e J. Rantakokko. "Measurement report. Laser total station campaign in KTH R1 for Ubisense system accuracy evaluation." Em: *FOI Swedish Defence Research Agency* (2012), pp. 1–10.

[AR12] M. Angermann e P. Robertson. "FootSLAM: Pedestrian simultaneous localization and mapping without exteroceptive sensorshitchhiking on human perception and cognition". Em: *Proceedings of the IEEE*. Vol. 100. SPL CONTENT. 2012, pp. 1840–1848. DOI: 10.1109/JPROC.2012.2189785.

[Aru+02] M. Arulampalam, S. Maskell, N. Gordon e T. Clapp. "A tutorial on particle filters for online nonlinear/non-Gaussian\nBayesian tracking". Em: *IEEE Transactions on Signal Processing* 50.2 (2002), pp. 174–188. DOI: 10.1109/78.978374. arXiv: arXiv:1011.1669v3.

[Au+13] A. W. S. Au, C. Feng, S. Valaee, S. Reyes, S. Sorour, S. N. Markowitz, D. Gold, K. Gordon e M. Eizenman. "Indoor tracking and navigation using received signal strength and compressive sensing on a mobile device". Em: *IEEE Transactions on Mobile Computing* 12.10 (2013), pp. 2050–2062. ISSN: 15361233. DOI: 10.1109/TMC.2012.175.

[Azi+09] M. Azizyan, R. R. Choudhury e I. Constandache. "SurroundSense : Mobile Phone Localization via Ambience Fingerprinting". Em: *ACM MobiCom '09* (2009), pp. 261–272. DOI: 10.1145/1614320.1614350.

[BH15] M. BAGCI e C. HACIZADE. "Performance Analysis of GPS Based Orbit Determination via Numerical Methods for a LEO Satellite". Em: *7th International Conference on Recent Advances in Space Technologies (RAST) 2015*. 2015, pp. 731 –735. DOI: 10.1109/RAST.2015.7208437.

[Bah+00] P Bahl, V. Padmanabhan e A Balachandran. "Enhancements to the RADAR user location and tracking system". Em: *Microsoft Research* 2.MSR-TR-2000-12 (2000), pp. 775–784.

[BP00] P. Bahl e V. N. Padmanabhan. "RADAR: An In-building RF-based User Location and Tracking System". Em: *Proc. IEEE INFOCOM 2000. The 19th annual conference on Computer Communications* 2 (2000), pp. 775–784. DOI: 10.1109/INFCOM.2000.832252. eprint: 1106.0222.

[BDW06a] T. Bailey e H. Durrant-Whyte. "Simultaneous localization and mapping (SLAM): Part I". Em: *IEEE Robotics and Automation Magazine* 13.3 (2006), pp. 108–117. DOI: 10.1109/MRA.2006.1678144.

[BDW06b] T. Bailey e H. Durrant-Whyte. "Simultaneous localization and mapping (SLAM): Part II". Em: *IEEE Robotics and Automation Magazine* 13.3 (2006), pp. 108–117. DOI: 10.1109/MRA.2006.1678144.

[Ban+10] N. Banerjee, S. Agarwal, P. Bahl, R. Chandra, A. Wolman e M. Corner. "Virtual compass: Relative positioning to sense mobile social interactions". Em: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 6030 LNCS. 2010, pp. 1–21. DOI: 10.1007/978-3-642-12654-3_1.

[BJ14] M. R. Basheer e S. Jagannathan. "Localization and tracking of objects using cross-correlation of shadow fading noise". Em: *IEEE Transactions on Mobile Computing* 13.10 (2014), pp. 2293–2305. DOI: 10.1109/TMC.2013.34.

[Bec+08] a Beck, P Stoica e J. Li. "Exact and Approximate Solutions of Source Localization Problems". Em: *IEEE Transactions on Signal Processing* 56.5 (2008), pp. 1770–1778. DOI: 10.1109/TSP.2007.909342.

[BM14] W. M.Y. W. Bejuri e M. M. Mohamad. "Wireless LAN/FM radio-based robust mobile indoor positioning: An initial outcome". Em: *International Journal of Software Engineering and its Applications* 8.2 (2014), pp. 313–324. DOI: 10.14257/ijseia.2014.8.2.31.

[Bel+09] F. Belloni, V. Ranki, A. Kainulainen e A. Richter. "Angle-based indoor positioning system for open indoor environments". Em: *Proceedings - 6th Workshop on Positioning, Navigation and Communication, WPNC 2009*. 2009, pp. 261–265. DOI: 10.1109/WPNC.2009.4907836.

[BH13] A. Brajdic e R. Harle. "Walk detection and step counting on unconstrained smartphones". Em: *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing - UbiComp '13* (2013), p. 225. DOI: 10.1145/2493432.2493449.

[BR11] L. Bruno e P. Robertson. "WiSLAM: Improving FootSLAM with WiFi". Em: *2011 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2011*. 2011. DOI: 10.1109/IPIN.2011.6071916.

[Caf00] J. Caffery. "A new approach to the geometry of TOA location". Em: *Vehicular Technology Conference, 2000. IEEE-VTS Fall VTC 2000. 52nd* (2000). DOI: 10.1109/VETECF.2000.886153.

[Cha+10] N. Chang, R. Rashidzadeh e M. Ahmadi. "Robust indoor positioning using differential Wi-Fi access points". Em: *IEEE Transactions on Consumer Electronics* 56.3 (2010), pp. 1860–1867. DOI: 10.1109/TCE.2010.5606338.

[Cha08] S. S. Chawathe. "Beacon Placement for Indoor Localization using Bluetooth". Em: *2008 11th International IEEE Conference on Intelligent Transportation Systems* (2008), pp. 980–985. DOI: 10.1109/ITSC.2008.4732690.

[CK02] Y. C. Y. Chen e H. Kobayashi. "Signal strength based indoor geolocation". Em: *2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002* 1.1 (2002), pp. 436–439. DOI: 10.1109/ICC.2002.996891.

[Che+15] Z. Chen, H. Zou, H. Jiang, Q. Zhu, Y. C. Soh e L. Xie. "Fusion of WiFi, Smartphone Sensors and Landmarks Using the Kalman Filter for Indoor Localization". Em: *Sensors* (2015), pp. 715–732. DOI: 10.3390/s150100715.

[Che+04] K. W. Cheung, H. C. So, W. K. Ma e Y. T. Chan. "Least Squares Algorithms for Time-of-Arrival-Based Mobile Location". Em: *IEEE Transactions on Signal Processing* 52.4 (2004), pp. 1121–1128. DOI: 10.1109/TSP.2004.823465.

[Chi+10] Y. S. Chiou, C. L. Wang e S. C. Yeh. "An adaptive location estimator using tracking algorithms for indoor WLANs". Em: *Wireless Networks* 16.7 (2010), pp. 1987–2012. DOI: 10.1007/s11276-010-0240-8.

[Chu+11] J. Chung, M. Donahoe, C. Schmandt, I.-J. Kim, P. Razavai e M. Wiseman. "Indoor location sensing using geo-magnetism". Em: *Proceedings of the 9th international conference on Mobile systems, applications, and services - MobiSys '11* (2011), pp. 141–154. DOI: 10.1145/1999995.2000010.

[Coh07] T. A. Cohn. "Scaling conditional random fields for natural language processing". Tese de doutoramento. Department of Computer Science e Software Engineering, University of Melbourne, 2007.

[Com+11] C. R. Comsa, A. M. Haimovich, S. Schwartz, Y. Dobyns e J. a. Dabin. "Source localization using time difference of arrival within a sparse representation framework". Em: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2011), pp. 2872–2875. DOI: 10.1109/ICASSP.2011.5947084.

[Con+10] I. Constandache, X. Bao, M. Azizyan e R. R. Choudhury. "Did You See Bob?: Human Localization using Mobile Phones". Em: *16th Annual International Conference on Mobile Computing and Networking (MobiCom '10)* (2010), pp. 149–160. DOI: 10.1145/1859995.1860013.

[Cor11] I. Corp. *Place Lab, a privacy-observant location system*. 2011. URL: http://placelab.org (acedido em 16/11/2016).

[Dab+04] F. Dabek, R. Cox, F. Kaashoek e R. Morris. "Vivaldi : A Decentralized Network Coordinate System". Em: *Communication* 34 (2004), pp. 15–26. DOI: 10.1145/1030194.1015471.

[DM] R. B. Dan Murray. *Adaptiv: An Adaptive Jerk Pace Buffer Step Detection Algorithm*. URL: https://developer.android.com/studio/index.html (acedido em 10/03/2017).

[Dar+09] D. Dardari, A. Conti, U. Ferner, A. Giorgetti e M. Z. Win. "Ranging with ultrawide bandwidth signals in multipath environments". Em: *Proceedings of the IEEE* 97.2 (2009), pp. 404–425. DOI: 10.1109/JPROC.2008.2008846.

[Dea+12] G. Deak, K. Curran e J. Condell. "A survey of active and passive indoor localisation systems". Em: *Computer Communications* 35.16 (2012), pp. 1939–1954. DOI: 10.1016/j.comcom.2012.06.004.

[EM06] F. Evennou e F. Marx. "Advanced integration of WiFi and inertial navigation systems for indoor mobile positioning". Em: *Eurasip Journal on Applied Signal Processing* 2006 (2006). DOI: 10.1155/ASP/2006/86706.

[Fen+12] C. Feng, W. S. A. Au, S. Valaee e Z. Tan. "Received-signal-strength-based indoor positioning using compressive sensing". Em: *IEEE Transactions on Mobile Computing* 11.12 (2012), pp. 1983–1993. DOI: 10.1109/TMC.2011.216.

[Fer+07] B. D. Ferris, D. Fox e N. Lawrence. "WiFi-SLAM using Gaussian process latent variable models". Em: *In Proceedings of IJCAI 2007*. 2007, pp. 2480–2485. URL: https://www.aaai.org/Papers/IJCAI/2007/IJCAI07-399.pdf.

[Foy76] W. H. Foy. "Position Location Solutions by Taylor Series Estimation." Em: *IEEE Transactions on Aerospace and Electronic Systems* AES-12.2 (1976), pp. 187–194. DOI: 10.1109/TAES.1976.308294.

[Fur+12] E. Furey, K. Curran e P. McKevitt. "HABITS: a Bayesian filter approach to indoor tracking and location". Em: *International Journal of Bio-Inspired Computation* 4.2 (2012), p. 79. DOI: 10.1504/IJBIC.2012.047178.

[Gao+11] Y. Gao, Q. Yang, G. Li, E. Y. Chang, D. Wang, C. Wang, H. Qu, P. Dong e F. Zhang. "XINS: the anatomy of an indoor positioning and navigation architecture". Em: *Proceedings of the 1st international workshop on Mobile location-based service* (2011), pp. 41–50. DOI: 10.1145/2025876.2025884.

[Geo+15] K. Georgiou, T. Constambeys, C. Laoudias, L. Petrou, G. Chatzimilioudis e D. Zeinalipour-Yazti. "Anyplace: A Crowdsourced Indoor Information Service". Em: *Proceedings - IEEE International Conference on Mobile Data Management*. Vol. 1. 2015, pp. 291–294. DOI: 10.1109/MDM.2015.80.

[Gooa] Google. *Activity - Google*. URL: https://developer.android.com/reference/android/app/Activity.html (acedido em 13/06/2017).

[Goob] Google. *Android Studio The Official IDE for Android*. URL: https://developer.android.com/studio/index.html (acedido em 10/03/2017).

[Gooc] Google. *Fragments - Google*. URL: https://developer.android.com/guide/components/fragments.html (acedido em 13/06/2017).

[Good] Google. *Intent - Google*. URL: https://developer.android.com/reference/android/content/Intent.html (acedido em 13/06/2017).

[Gooe] Google. *Location Request - Google*. URL: https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest (acedido em 05/08/2017).

[Goof] Google. *Motion Sensors - Google*. URL: https://developer.android.com/reference/android/hardware/SensorEvent.html (acedido em 03/08/2017).

[Goog] Google. *Sensor Coordinate System - Google*. URL: https://developer.android.com/guide/topics/sensors/sensors_overview.html (acedido em 19/01/2017).

[Gooh] Google. *Sensor Event - Google*. URL: https://developer.android.com/guide/topics/sensors/sensors_motion.html (acedido em 19/01/2017).

[Gooi] *Google Project Tango*. URL: https://get.google.com/tango/ (acedido em 08/01/2017).

[Gooj] Googlemaps. *About - Google Maps*. URL: https://www.google.com/maps/about/partners/indoormaps (acedido em 13/01/2017).

[Gos+11] A. Goswami, L. E. Ortiz e S. R. Das. "WiGEM : A Learning-Based Approach for Indoor Localization". Em: *CoNEXT* (2011). DOI: http://doi.acm.org/10.1145/2079296.2079299.

[Guv03] I. Guvenc. "Enhancements to rss based indoor tracking systems using kalman filters". Em: (2003).

[Han+14] D. Han, S. Jung, M. Lee e G. Yoon. "Building a practical wi-fi-based indoor navigation system". Em: *IEEE Pervasive Computing* 13.2 (2014), pp. 72–79. DOI: 10.1109/MPRV.2014.24.

[Har+13] M. Hardegger, G Tröster e D. Roggen. "Improved actionSLAM for long-term indoor tracking with wearable motion sensors". Em: *Symposium on Wearable* (2013), pp. 1–8. DOI: 10.1145/2493988.2494328.

[Har13] R. Harle. *A survey of indoor inertial positioning systems for pedestrians*. 2013. DOI: 10.1109/SURV.2012.121912.00075.

[Har] D. Harrinman. *Why You Need a Gyroscope to Measure Position*. URL: https://www.linkedin.com/pulse/why-you-need-gyroscope-measure-position-doug-harriman (acedido em 16/07/2017).

[HH94] A. Harter e A. Hopper. "A distributed location system for the active office". Em: *IEEE Network* 8.1 (1994), pp. 1–17. DOI: 10.1109/65.260080.

[HC14] S. He e S. H. G. Chan. "Sectjunction: Wi-Fi indoor localization based on junction of signal sectors". Em: *2014 IEEE International Conference on Communications, ICC 2014*. 2014, pp. 2605–2610. DOI: 10.1109/ICC.2014.6883716.

[HC16a] S. He e S. H. G. Chan. "Tilejunction: Mitigating signal noise for fingerprint-based indoor localization". Em: *IEEE Transactions on Mobile Computing* 15.6 (2016), pp. 1554–1568. DOI: 10.1109/TMC.2015.2463287.

[HC16b] S. He e S. H. G. Chan. "Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons". Em: *IEEE Communications Surveys and Tutorials* 18.1 (2016), pp. 466–490. DOI: 10.1109/COMST.2015.2464084.

[HB01] J. Hightower e G. Borriello. "Location systems for ubiquitous computing". Em: *Computer* 34.8 (2001), pp. 57–66. DOI: 10.1109/2.940014.

[Hig+00] J. Hightower, G. Borriello e R. Want. "SpotON: An indoor 3D location sensing technology based on RF signal strength". Em: *UW CSE Technical Report* (2000), p. 16. URL: ftp://ftp.cs.washington.edu/tr/2000/02/UW-CSE-00-02-02.pdf.

[Hig+11] T. Higuchi, S. Fujii, H. Yamaguchi e T. Higashino. "An efficient localization algorithm focusing on stop-and-go behavior of mobile nodes". Em: *2011 IEEE International Conference on Pervasive Computing and Communications, PerCom 2011*. 2011, pp. 205–212. DOI: 10.1109/PERCOM.2011.5767587.

[Hig+14] T. Higuchi, S. Fujii, H. Yamaguchi e T. Higashino. "Mobile node localization focusing on stop-and-go behavior of indoor pedestrians". Em: *IEEE Transactions on Mobile Computing* 13.7 (2014), pp. 1564–1578. DOI: 10.1109/TMC.2013.139.

[Hil+14] S. Hilsenbeck, D. Bobkov, G. Schroth, R. Huitl e E. Steinbach. "Graph-based Data Fusion of Pedometer and WiFi Measurements for Mobile Indoor Positioning". Em: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '14 Adjunct* (2014), pp. 147–158. DOI: 10.1145/2632048.2636079.

[Hon+09] V. Honkavirta, T. Perälä, S. Ali-Löytty e R. Piché. "A comparative survey of WLAN location fingerprinting methods". Em: *Proceedings - 6th Workshop on Positioning, Navigation and Communication, WPNC 2009*. 2009, pp. 243–251. DOI: 10.1109/WPNC.2009.4907834.

[Hua+11] J. Huang, D. Millman, M. Quigley, D. Stavens, S. Thrun e A. Aggarwal. "Efficient, generalized indoor WiFi GraphSLAM". Em: *Proceedings - IEEE International Conference on Robotics and Automation*. 2011, pp. 1038–1043. DOI: 10.1109/ICRA.2011.5979643.

[Hyd] *Hydra web server*. URL: http://hydra.hellug.gr (acedido em 12/02/2017).

[Jia+12] Y. Jiang, X. Pan, K. Li, Q. Lv, R. P. Dick, M. Hannigan e L. Shang. "ARIEL: Automatic Wi-Fi based Room Fingerprinting for Indoor Localization". Em: *Proceedings of the 2012 ACM Conference on Ubiquitous Computing* (2012), pp. 441–450. DOI: 10.1145/2370216.2370282.

[Jia+13] Y. Jiang, Y. Xiang, X. Pan, K. L. Q. Lv, R. P. Dick, L. Shang e M. Hannigan. "Hallway based Automatic Indoor Floorplan Construction using Room Fingerprints ". Em: *Proc. Int. Conf. on Ubiquitous Computing* (2013), pp. 315–324. DOI: 10.1145/2493432.2493470.

[Jun+13] J. Jun, Y. Gu, L. Cheng, B. Lu, J. Sun, T. Zhu e J. Niu. "Social-Loc: Improving Indoor Localization with Social Sensing". Em: *ACM Conference on Embedded Networked Sensor Systems*. 2013, 14:1–14:14. DOI: 10.1145/2517351.2517352.

[KK12] K. Kaji e N. Kawaguchi. "Gaussian Mixture Model and Particle Filter". Em: *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 2012, pp. 1–9. DOI: 10.1109/IPIN.2012.6418943.

[KS12] A Kashevnik e M Shchekotov. "Comparative Analysis of Indoor Positioning Systems Based on Communications Supported by Smartphones". Em: *Proceedings of FRUCT Conference* (2012). URL: https://fruct.org/publications/fruct12/files/Kas.pdf.

[Kim+12] Y. Kim, H. Shin e H. Cha. "Smartphone-based Wi-Fi pedestrian-tracking system tolerating the RSS variance problem". Em: *2012 IEEE International Conference on Pervasive Computing and Communications, PerCom 2012*. 2012, pp. 11–19. DOI: 10.1109/PerCom.2012.6199844.

[Kja07] M. B. Kjaergaard. "A Taxonomy for Radio Location Fingerprinting". Em: *International Symposium on Location- and Context-Awareness* (2007), pp. 139–156. DOI: 10.1007/978-3-540-75160-1_9.

[KT07] R. Klinger e K. Tomanek. "Classical Probabilistic Models and Conditional Random Fields". Em: *Entropy* 51 (2007), pp. 282–289. DOI: 10.1007/s002390010117.

[KY10] H. Koyuncu e S. H. Yang. "A Survey of Indoor Positioning and Object Locating Systems". Em: *International Journal of Computer Science and Network Security (IJCSNS '10)* 10.5 (2010), pp. 121–128. DOI: 10.1109/INNOVATIONS.2011.5893813.

[Krz06] H. Krzysztof, W. Kolodziej Johan. *Local Positioning Systems: LBS Applications and Services*. Ed. por C. Press. 2006. ISBN: 9780849333491.

[Kuo+14] Y.-s. Kuo, P. Pannuto, K.-j. Hsiao, P. Dutta e A. Arbor. "Luxapose : Indoor Positioning with Mobile Phones and Visible Light". Em: *Mobicom'14* (2014), pp. 299–301. DOI: 10.1145/2639108.2639109.

[LaM+05] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, J. Tabert, P. Powledge, G. Borriello e B. Schilit. "Place Lab: Device Positioning Using Radio Beacons in the Wild". Em: *Pervasive Computing* 3468 (2005), pp. 116–133. DOI: 10.1007/11428572_8.

[LS13] K. C. Lan e W. Y. Shih. "On calibrating the sensor errors of a PDR-based indoor localization system." Em: *Sensors* 13.4 (2013), pp. 4781–4810. DOI: 10.3390/s130404781.

[Lao+12] C. Laoudias, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti e C. G. Panayiotou. "The airplace indoor positioning platform for Android smartphones". Em: *Proceedings - 2012 IEEE 13th International Conference on Mobile Data Management, MDM 2012*. 2012, pp. 312–315. DOI: 10.1109/MDM.2012.68.

[Li+12] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu e F. Zhao. "A reliable and accurate indoor localization method using phone inertial sensors". Em: *ACM Conference on Ubiquitous Computing (UbiComp)* (2012), pp. 421–430. DOI: 10.1145/2370216.2370280.

[Li-06] H.-h. C. Li-wei Chan, Ji-rung Chiang, Yi-chao Chen, Chia-nan Ke, Jane Hsu. "Collaborative Localization – Enhancing WiFi-Based Position Estimation with Neighborhood Links in Clusters". Em: *Pervasive Computing* 3968 (2006), pp. 50–66. DOI: 10.1007/11748625_4.

[Lim+10] H. Lim, L. C. Kung, J. C. Hou e H. Luo. "Zero-configuration indoor localization over IEEE 802.11 wireless infrastructure". Em: *Wireless Networks* 16.2 (2010), pp. 405–420. DOI: 10.1007/s11276-008-0140-3.

[Liu+12] H. Liu, Y. Gan, J. Yang, S. Sidhom, Y. Wang, Y. Chen e F. Ye. "Push the limit of WiFi based localization for smartphones". Em: *Proceedings of the 18th annual international conference on Mobile computing and networking (Mobicom '12)* (2012), p. 305. DOI: 10.1145/2348543.2348581.

[Liu+14] H. Liu, J. Yang, S. Sidhom, Y. Wang, Y. Chen e F. Ye. "Accurate WiFi based localization for smartphones using peer assistance". Em: *IEEE Transactions on Mobile Computing* 13.10 (2014), pp. 2199–2214. DOI: 10.1109/TMC.2013.140.

[Liu+07] H. Liu, H. Darabi, P. Banerjee e J. Liu. *Survey of wireless indoor positioning techniques and systems*. Vol. 37. 6. 2007, pp. 1067–1080. ISBN: 9783527619269. DOI: 10.1109/TSMCC.2007.905750.

[Liu+13] S. Liu, Y. Jiang e A. Striegel. "Face-to-Face Proximity Estimation Using Bluetooth On Smartphones". Em: *IEEE Transactions on Mobile Computing* (2013). DOI: 10.1109/TMC.2013.44.

[Lou05] M. I. a. Lourakis. "A Brief Description of the Levenberg-Marquardt Algorithm Implemened by levmar". Em: *Matrix* 3 (2005). DOI: 10.1016/j.ijinfomgt.2009.10.001.

[Ltd] U. Ltd. *The Ubisense Precise Real-time Location System - Series 7000 Sensor*. URL: http://www.ubisense.net/ (acedido em 09/10/2016).

[Lym+15] D. Lymberopoulos, J. Liu, X. Yang, R. R. Choudhury, V. Handziski e S. Sen. "A Realistic Evaluation and Comparison of Indoor Location Technologies: Experiences and Lessons Learned". Em: *Proceedings of the 14th International Conference on Information Processing in Sensor Networks* Table 1 (2015), pp. 178–189. DOI: 10.1145/2737095.2737726.

[Mad+05] D. Madigan, E. Einahrawy, R. P. Martin, W. H. Ju, P. Krishnan e a. S. Krishnakumar. "Bayesian indoor positioning systems". Em: *IEEE Conference on Computer and Communications* 2 (2005), pp. 1217–1227. DOI: 10.1109/INFCOM.2005.1498348.

[Mar+10] E. Martin, O Vinyals, G. Friedland e R Bajcsy. "Precise indoor localization using smart phones". Em: *Proceedings of the international conference on Multimedia Pages* (2010), pp. 787–790. DOI: 10.1145/1873951.1874078.

[MT13] V. Maximov e O. Tabarovsky. "Survey of Accuracy Improvement Approaches for Tightly Coupled ToA / IMU Personal Indoor Navigation System". Em: October. 2013. ISBN: 9781467319546.

[Men+16] S. Mengin, C. Portes, J. Vialle e B. Vincent. *Indoor Positioning & Navigation*. 2016. URL: https://www.infsoft.com (acedido em 07/02/2017).

[Mir+12] P. Mirowski, P. Whiting, H. Steck, R. Palaniappan, M. MacDonald, D. Hartmann e T. K. Ho. "Probability kernel regression for WiFi localisation". Em: *Journal of Location Based Services* 6.2 (2012), pp. 81–100. DOI: 10.1080/17489725.2012.694723.

[Mir+14] P. Mirowski, D. Milioris, P. Whiting e T. Kam Ho. "Probabilistic radio-frequency fingerprinting and localization on the run". Em: *Bell Labs Technical Journal* 18.4 (2014), pp. 111–133. DOI: 10.1002/bltj.21649.

[MB15] R. Monir Vaghefi e M. Buehrer. "Cooperative Localization in NLOS Environments Using Semidefinite Programming". Em: *IEEE Communications Letters* 19 (2015). DOI: 10.1109/LCOMM.2015.2442580.

[Nag] V. Nagpal. *Step Detector and Step Counters Sensors Book*. URL: www.packtpub.com/books/content/step-detector-and-step-counters-sensors (acedido em 19/05/2017).

[Nan+12] R. Nandakumar, K. K. Chintalapudi e V. N. Padmanabhan. "Centaur: locating devices in an office environment". Em: *Proc.18th Ann. Int. Conf. Mob. Comput. Netw. (MobiCom'12)* (2012), pp. 1–12. DOI: 10.1145/2348543.2348579.

[Ouy+12] R. W. Ouyang, A. K. S. Wong, C. T. Lea e M. Chiang. "Indoor location estimation with reduced calibration exploiting unlabeled data via hybrid generative/discriminative learning". Em: *IEEE Transactions on Mobile Computing* 11 (2012), pp. 1613–1626. DOI: 10.1109/TMC.2011.193.

[Pac16] A. Pacha. *Master Thesis "Sensor fusion for robust outdoor Augmented Reality tracking on mobile devices*. 2016. URL: http://my-it.at/media/MasterThesis-Pacha.pdf (acedido em 19/08/2016).

[Pen+07] C. Peng, G. Shen, Y. Zhang, Y. Li e K. Tan. "Beepbeep: a high accuracy acoustic ranging system using cots mobile devices". Em: *Proceedings of the 5th international conference on Embedded networked sensor systems* (2007), pp. 1–14. DOI: 10.1145/1322263.

[Pou+12] M. Pourhomayoun, Z. Jin e M. Fowler. "Spatial sparsity based indoor localization in wireless sensor network for assistive healthcare." Em: *Conference proceedings : IEEE Engineering in Medicine and Biology Society* 2012 (2012), pp. 3696–9. DOI: 10.1109/EMBC.2012.6346769.

[Pri+00] N. B. Priyantha, A. Chakraborty e H. Balakrishnan. "The Cricket location-support system". Em: *Proceedings of the 6th annual international conference on Mobile computing and networking - MobiCom '00* (2000), pp. 32–43. DOI: 10.1145/345910.345917.

[Que16] R. Queirós. *Android - Bases de Dados e Geolocalização*. Ed. por L. FCA - Editora de Informática. 1º edição. 2016, pp. 101–176.

[RM13] V. Radu e M. K. Marina. "HiMLoc: Indoor smartphone localization via activity aware pedestrian dead reckoning with selective crowdsourced WiFi fingerprinting". Em: *2013 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2013*. 2013. DOI: 10.1109/IPIN.2013.6817916.

[Rai+12] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan e R. Sen. "Zee: Zero-Effort Crowdsourcing for Indoor Localization". Em: *Proceedings of the 18th annual international conference on Mobile computing and networking - Mobicom '12* (2012). DOI: 10.1145/2348543.2348580.

[Rap02] T. S. Rappaport. "Wireless Communications: Principles and Practice". Em: *Prosltdcom Prosltdcom* 207 (2002), pp. 1–707. DOI: 10.1002/9781119992806.fmatter.

[Red] *Redes móveis - Basics of propagation*. 2014. URL: http://tele1.dee.fct.unl.pt/rm_2014_2015/secure/a1_rm.pdf (acedido em 12/02/2017).

[RG09] A. W. Reza e T. K. Geok. "Investigation of indoor location sensing via RFID reader network utilizing grid covering algorithm". Em: *Wireless Personal Communications* 49.1 (2009), pp. 67–80. DOI: 10.1007/s11277-008-9556-4.

[RFT11] RFTechnologies. *Pinpoint 3D-iD*. 2011. URL: www.rft.com (acedido em 16/12/2016).

[Row96] S. Roweis. *Levenberg-Marquardt Optimization*. Rel. téc. University Of Toronto, 1996.

[Roy+14] N. Roy, H. Wang e R. Roy Choudhury. "I am a smartphone and i can tell my user's walking direction". Em: *MobiSys '14* (2014), pp. 329–342. DOI: 10.1145/2594368.2594392.

[SK10] H. Schweinzer e G. Kaniak. "Ultrasonic device localization and its potential for wireless sensor network security". Em: *Control Engineering Practice* 18 (2010), pp. 852–862. DOI: 10.1016/j.conengprac.2008.12.007.

[Sei+10] J Seitz, T Vaupel, J Jahn, S Meyer, J. G. Boronat e J Thielecke. "A Hidden Markov Model for urban navigation based on fingerprinting and pedestrian dead reckoning". Em: *Information Fusion (FUSION), 2010 13th Conference on* (2010), pp. 1–8. DOI: 10.1109/ICIF.2010.5712025.

[Sen] Senion. *What is Indoor Positioning Systems?* URL: https://senion.com/indoor-positioning-system/ (acedido em 11/12/2016).

[She+13] G. Shen, Z. Chen, P. Zhang, T. Moscibroda e Y. Zhang. "Walkie-markie: Indoor pathway mapping made easy". Em: *Proceedings of the 10th USENIX conference on Networked Systems Design and Implementation* (2013), pp. 85–98.

[SW10] Y. Shen e M. Z. Win. "Fundamental limits of wideband localization - Part I: A general framework". Em: *IEEE Transactions on Information Theory* 56.10 (2010), pp. 4956–4980. DOI: 10.1109/TIT.2010.2060110.

[Shi+11] H. Shin, Y. Chon, K. Park e H. Cha. "FindingMiMo : Tracing a Missing Mobile Phone using Daily Observations". Em: *Signal Processing* 50 (2011), pp. 29–42. DOI: 10.1145/1999995.1999999.

[Shi+12] H. Shin, Y. Chon e H. Cha. "Unsupervised construction of an indoor floor plan using a smartphone". Em: *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews* 42.6 (2012), pp. 889–898. DOI: 10.1109/TSMCC.2011.2169403.

[Sho+12] R. Shokri, G. Theodorakopoulos, C. Troncoso, J.-P. Hubaux e J.-Y. Le Boudec. "Protecting Location Privacy: Optimal Strategy against Localization Attacks". Em: *Proceedings of the 2012 ACM conference on Computer and communications security* (2012), pp. 617–627. DOI: 10.1145/2382196.2382261.

[Sir10] N. Sirola. "Closed-form algorithms in mobile positioning: Myths and misconceptions". Em: *Proceedings of the 2010 7th Workshop on Positioning, Navigation and Communication, WPNC'10* (2010), pp. 38–44. DOI: 10.1109/WPNC.2010.5653789.

[Sky11] Skyhook. *Skyhook Wireless*. 2011. URL: http://www.skyhookwireless.com/ (acedido em 02/01/2017).

[Sou+13] M. Sousa, A. Techmer, A. Steinhage, C. Lauterbach e P. Lukowicz. "Human tracking and identification using a sensitive floor and wearable accelerometers". Em: *2013 IEEE International Conference on Pervasive Computing and Communications, PerCom 2013*. 2013, pp. 166–171. DOI: 10.1109/PerCom.2013.6526728.

[SG05] P Steggles e S Gschwind. *The Ubisense smart space platform*. 2005. DOI: 10.1145/313451.313476.

[Sun+13] W. Sun, J. Liu, C. Wu, Z. Yang, X. Zhang e Y. Liu. "MoLoc: On distinguishing fingerprint twins". Em: *Proceedings - International Conference on Distributed Computing Systems* (2013), pp. 226–235. DOI: 10.1109/ICDCS.2013.41.

[Sys11] B. Systems. *BLIP Systems*. 2011. URL: http://www.blipsystems.com/ (acedido em 02/01/2017).

[Thr00] S. Thrun. "Probabilistic Algorithms in Robotics". Em: *Science* 4.April (2000), pp. 93–109. DOI: 10.1007/s004539910017.

[Tom+15] S Tomic, M Beko e R Dinis. "RSS-based localization in wireless sensor networks using convex relaxation: Noncooperative and cooperative schemes". Em: *IEEE Transactions on Vehicular Technology* 64.5 (2015), pp. 2037–2050. DOI: 10.1109/TVT.2014.2334397.

[Tom17] S. Tomic. "Target Localization and Tracking in Wireless Sensor Networks". Tese de doutoramento. Universidade Nova de Lisboa, 2017.

[Tsa+10] J. Y. Tsai, P. G. Kelley, L. F. Cranor e N. Sadeh. "Location-Sharing Technologies : Privacy Risks and Controls". Em: *A Journal of Law and Policy for the Information Society* 6.2 (2010), pp. 119–151.

[Tsu+13] Y. Tsuda, Q. Kong e T. Maekawa. "Detecting and correcting WiFi positioning errors". Em: *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing - UbiComp '13* (2013). DOI: 10.1145/2493432.2493460.

[UN13] M. Uddin e T. Nadeem. "RF-Beep: A light ranging scheme for smart devices". Em: *2013 IEEE International Conference on Pervasive Computing and Communications, PerCom 2013*. 2013, pp. 114–122. DOI: 10.1109/PerCom.2013.6526721.

[VRE] VREMSoftwareDevelopment. *WiFiAnalyzer*. URL: https://github.com/VREMSoftwareDevelopment/WiFiAnalyzer (acedido em 19/10/2016).

[Wan+12] H. Wang, A. Elgohary e R. R. Choudhury. "No Need to War-Drive : Unsupervised Indoor Localization". Em: *Proceedings of the 10th international conference on Mobile systems, applications, and services (MobiSys '12)* (2012), pp. 197–210. ISSN: 10414347. DOI: 10.1145/2307636.2307655.

[Wan+92] R. Want, A. Hopper, V. Falcao, J. Gibbons, V. Falcão, J. Gibbons, V. Falcao e J. Gibbons. "The Active Badge Location System". Em: *ACM Transactions on Information Systems (TOIS)* 10.1 (1992), pp. 91–102. DOI: 10.1145/128756.128759.

[Xia+14] Z. Xiao, H. Wen, A. Markham e N. Trigoni. "Lightweight map matching for indoor localisation using conditional random fields". Em: *IPSN 2014 - Proceedings of the 13th International Symposium on Information Processing in Sensor Networks (Part of CPS Week)*. 2014, pp. 131–142. DOI: 10.1109/IPSN.2014.6846747.

[Xia+15a] Z. Xiao, H. Wen, A. Markham e N. Trigoni. "Indoor tracking using undirected graphical models". Em: *IEEE Transactions on Mobile Computing* 14.11 (2015), pp. 2286–2301. DOI: 10.1109/TMC.2015.2398431.

[Xia+15b] Z. Xiao, H. Wen, A. Markham, N. Trigoni, P. Blunsom e J. Frolik. "Non-Line-of-Sight Identification and Mitigation Using Received Signal Strength". Em: *IEEE Transactions on Wireless Communications* 14.3 (2015), pp. 1689–1702. DOI: 10.1109/TWC.2014.2372341.

[Xia+15c] Z. Xiao, H. Wen, A. Markham e N. Trigoni. "Robust indoor positioning with lifelong learning". Em: *IEEE Journal on Selected Areas in Communications* 33.11 (2015), pp. 2287–2301. DOI: 10.1109/JSAC.2015.2430514.

[YS15] C. Yang e H. R. Shao. "WiFi-based indoor positioning". Em: *IEEE Communications Magazine* 53.3 (2015), pp. 150–157. DOI: 10.1109/MCOM.2015.7060497.

[Yan+13] S. Yang, P. Dessai, M. Verma e M. Gerla. "FreeLoc: Calibration-free crowdsourced indoor localization". Em: *Proceedings - IEEE INFOCOM*. 2013, pp. 2481–2489. DOI: 10.1109/INFCOM.2013.6567054.

[Yan+14] Z. Yang, X. Feng e Q. Zhang. "Adometer: Push the limit of pedestrian indoor localization through cooperation". Em: *IEEE Transactions on Mobile Computing* 13.11 (2014), pp. 2473–2483. DOI: 10.1109/TMC.2014.2329855.

[YA08] M. Youssef e A. Agrawala. "The Horus location determination system". Em: *Wireless Networks* 14.3 (2008), pp. 357–374. DOI: 10.1007/s11276-006-0725-7.

[Zan09] P. A. Zandbergen. "Accuracy of iPhone locations: A comparison of assisted GPS, WiFi and cellular positioning". Em: *Trans. GIS*. Vol. 13. SUPPL. 1. Blackwell Publishing Ltd, 2009, pp. 5–25. DOI: 10.1111/j.1467-9671.2009.01152.x.

[Zha+15] Q. Zhang, Z. Zhou, W. Xu, J. Qi, C. Guo, P. Yi, T. Zhu e S. Xiao. "Fingerprint-free tracking with dynamic enhanced field division". Em: *Proceedings - IEEE INFOCOM*. Vol. 26. 2015, pp. 2785–2793. DOI: 10.1109/INFOCOM.2015.7218671.

[Zha+14] X. Zhao, Z. Xiao, A. Markham, N. Trigoni e Y. Ren. "Does BTLE measure up against WiFi? A comparison of indoor location performance". Em: *20th European Wireless Conference* (2014), pp. 1–6. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6843088&isnumber=6843048.

[Zho+11] Y. Zhou, C. L. Law, Y. L. Guan e F. Chin. "Indoor elliptical localization based on asynchronous UWB range measurement". Em: *IEEE Transactions on Instrumentation and Measurement*. Vol. 60. 1. 2011, pp. 248–257. DOI: 10.1109/TIM.2010.2049185.

[Zho+15] Z. Zhou, S. Member, Z. Yang, C. Wu, S. Member, L. Shangguan, S. Member, H. Cai, Y. Liu, L. M. Ni e A. W. Lans. "WiFi-Based Indoor Line-of-Sight Identification". Em: *IEEE Transactions on Wireless Communications* 14.11 (2015), pp. 6125–6136.

[Zij04] W. Zijlstra. "Assessment of spatio-temporal parameters during unconstrained walking". Em: *European Journal of Applied Physiology* 92.1-2 (2004), pp. 39–44. DOI: 10.1007/s00421-004-1041-5.

[Zir+10] S. Zirari, P. Canalda e F. Spies. "WiFi GPS based combined positioning algorithm". Em: *Wireless Communications, Networking and Information Security (WCNIS), 2010 IEEE International Conference* (2010), pp. 684–688. DOI: 10.1109/WCINS.2010.5544653.

Apêndice A

Sistemas de Localização

O apêndice A analisa os sistemas de localização mais conhecidos actualmente, sendo no final realizada uma comparação entre vários sistemas.

A.1 SpotON

O SpotON [Hig+00] é uma tecnologia de localização *indoor* 3D baseada na intensidade do sinal de Radio Frequency (RF). O sistema é semelhante aos sistemas Microsoft Research (MSR), WaveLAN e Pinpoint [KY10]. Os autores indicam a possibilidade de obter melhor resolução e precisão em comparação com o sistema MSR e com um custo menor do que o sistema Pinpoint. O SpotON é baseado num produto chamado *AIR ID*, vendido pela *RFIDeas*. Uma das limitações do *AIR ID* é o protocolo de estação base RS232. O RS232 tem um número limitado de portas série no servidor. Para ter mais flexibilidade, o sistema utiliza o servidor *Hydra microweb*, que é um servidor *HTTP multi-threaded* com um conjunto de *threads* constante, mas configurável. O *Hydra microweb* permite que cada thread consiga lidar com várias ligações, tornando assim ilimitado o número de ligações.

A.2 Ubisense

O Ubisense é um sistema de localização comercial [Ltd]. Tem uma precisão muito alta, de aproximadamente 30*cm* para 95% das leituras, devido ao uso da triangulação de sinais ativos [SG05]. Os principais componentes de um sistema Ubisense são: os sensores, as Ubitags(etiquetas) a serem monitorizadas e a plataforma de software. As Ubitags usam RF Radio (2,4 GHz) para coordenar o tempo de transmissão dos pulsos UWB (6-8 GHz). O sistema usa TDOA e AoA para estimar as localizações das *tags*. A localização 3D de uma Ubitag pode ser estimada usando no mínimo dois recetores. O Ubisense não requer que

as *tags* estejam em "Linha-de-Vista" devido à tecnologia UWB. O sinal pode ser filtrado e o efeito multi caminho é desta forma minimizado . Esta é uma grande vantagem porque o efeito multicaminho representa a principal causa da baixa precisão dos sistemas *indoor*. Por essa razão, é habitual usar-se um algoritmo de filtragem dinâmico do tipo *default 6 static fixed height* disponibilizado pela Ubisense [Ang+12], que é aplicado directamente aos dados recolhidos pelo smartphone.

O Ubisense pode cobrir grandes áreas e oferece a possibilidade de monitorizar um grande número de utilizadores em tempo real. A cobertura espacial é assegurada através de métodos de *clustering* que executam um grande número de serviços com um uso mais reduzido da largura de banda. Para que isto seja possível, a hierarquia tem quatro tipos de sensores: sensores Ubisensor individuais; sensores de localização, nas quais as informações do sensor são filtradas e apresentadas como dados abstratos de localização; sensores de geometria, nas quais as relações espaciais entre objetos são geridas; e um sensor do local que gere os dados em grande parte constantes de uma região. Todos os serviços são locais, incluindo os serviços das aplicações desenvolvidas pelos utilizadores, de modo a que até mesmo os programas possam ser escaláveis para grandes áreas. O sistema funciona de forma semelhante a uma rede móvel, onde o ambiente é organizado em células com pelo menos quatro sensores/leitores [Ang+12].

Os principais inconvenientes do Ubisense são os cabos que interligam as várias Ubitag e o preço. O Ubisense é um dos sistemas com maior precisão no mercado, mas exige hardware dedicado que geralmente não está presente nos terminais móveis. Portanto, infelizmente, não pode ser aplicado usando apenas smartphones Android.

A.3 Ekahau RTLS

O Ekahau é um sistema de posicionamento comercial que usa WLANs e monitoriza dispositivos eletrónicos como *tags*, *Personal Digital Assistants* (PDAs), PCs, e dispositivos wireless. A localização é calculada correlacionando informações de espaço com a RSSI medida quando o dispositivo recebe sinal de pelo menos 6 WiFi APs [Zir+10].

O Ekahau RTLS é uma combinação de ferramentas (veja a figura A.1), como o Ekahau Client, o Ekahau Positioning Engine, o Ekahau Manager, o Ekahau Planner, o Ekahau Application Framework e o SDK.

O Ekahau Client é executado em segundo plano nos dispositivos como um serviço que permite a comunicação com o *Positioning Engine* implementado num servidor. Um dos benefícios de usar a comunicação WiFi padrão é que o Cliente pode ser incorporado em qualquer dispositivo usando um rádio 802.11. As *tags* (etiquetas) Ekahau podem ser utilizadas em qualquer tipo de dispositivos sem comunicação por rádio 802.11 ou serem transportadas por pessoas. Vários sistemas de identificação [Hig+00; RG09] usam *tags* eletrónicos semelhantes com comunicação por radiofrequência RFID ou infravermelho (IR). No entanto, as *tags* Ekahau não precisam estar na proximidade de um leitor. A comunicação entre o Cliente e o Mecanismo de Posicionamento usa uma largura de banda

e potência de processamento mínimas devido ao facto de que a única informação enviada é o RSSI. O Positioning Engine é um software em Java que pode calcular dois tipos de posição: localização do cliente (x, y, piso, velocidade, direção) ou como informações lógicas ("escritório", "sala de reuniões").

Os mapas de localização podem ser criados como modelos de posicionamento usando o Ekahau Manager. Esta é uma ferramenta autónoma para desenhar áreas lógicas, testando informações de posicionamento e analisando a precisão da localização. O Ekahau também inclui uma ferramenta que permite planear a rede WiFi.

Um inconveniente do Ekahau é a impossibilidade de obter localizações em pontos "cegos" (áreas não cobertas pelos APs). Uma solução neste caso é instalar APs adicionais. No entanto, isso aumenta o custo do sistema. Outra solução é apresentada em [Fur+12], onde a localização é prevista com base em hábitos de movimento.

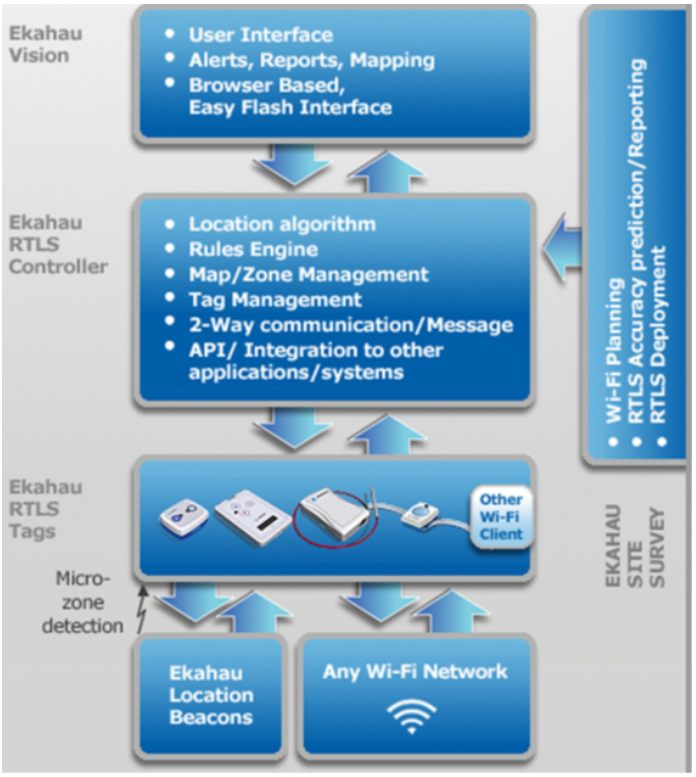


Figura A.1: Arquitetura do sistema Ekahau RTLS [Fur+12].

A.4 Microsoft research radar

O Radar foi provavelmente o primeiro sistema de posicionamento a usar redes IEEE 802.11 [Bah+00; BP00]. O Radar usa o método de *fingerprinting* (assinatura). O sistema calcula a localização monitorizando a intensidade do sinal dos dispositivos e compara o valor com as entradas da base de dados usadas para criar o mapa de rádio. Um segundo método é baseado na medição das propriedades de propagação de rádio do ambiente a ser

monitorizado. Neste método, o mapa de rádio é reproduzido a partir das propriedades de propagação no interior da localidade e pelo *layout* do ambiente.

O Radar tem uma precisão de $2-3m$ (aproximadamente o tamanho de uma sala de escritório). A principal desvantagem do sistema Radar é a necessidade de se refazer um novo mapa sempre que existirem mudanças no localização que possam afetar a propagação do sinal (i.e dentro de casa) [Bah+00]. O sistema também permite ao utilizador fornecer a sua posição clicando no mapa.

Uma das vantagens de ser tudo calculado no dispositivo é a garantia de privacidade, pois a localização não é partilhada com outros dispositivos. Contundo, perde-se assim a oportunidade de melhorar a precisão da localização por métodos cooperativos, como já foi abordado anteriormente.

A.5 AeroScout

O sistema AeroScout usa uma técnica de localização baseada em WiFi [Aer]. Utiliza a infra-estrutura sem fios existente para calcular a localização de qualquer dispositivo móvel com interface 802.11b/g, sendo também possível monitorizar as *tags* AeroScout. O sistema usa TDOA para ambientes *indoor* maiores (ou ao ar livre) e utiliza o RSSI para ambientes de menor tamanho. Os recetores AeroScout são usados diferentemente com base na técnica de localização implementada. Assim, para TDOA os recetores são leitores WiFi de longo alcance e para RSSI o sistema usa os mesmos recetores e/ou APs da Cisco.

O sistema AeroScout, representado na figura A.2, pode ser implementado usando *tags* AeroScout Wifi, recetores de localização e excitadores.

O AeroScout Exciter é um componente de hardware opcional. Os Exciters monitorizam os dispositivos e permitem alcançar melhor precisão. Têm a capacidades de detecção usando um ponto de passagem utilizando as mesmas *tags* WiFi que também são usadas para estimação da localização. Os Exciters usam sinais de baixa frequência para comunicar com as *tags*. As *tags*, quando recebem um sinal, respondem com uma mensagem que é recebida nos diversos APs WiFi e nos receptores de localização do AeroScout. Desta forma, obtém-se uma precisão de aproximadamente $6,5m$. A fim de estender a área de cobertura, múltiplos Exciters podem ser ligados comportando-se como um único Exciter.

Os receptores AeroScout têm o papel de gravar as medições TDOA e enviá-las para o AeroScout Engine. Cada recetor tem a capacidade de processar cerca de 300 medições por segundo. No caso em que se usam medições RSSI, o sistema AeroScout usa a infra-estrutura WiFi existente. O AeroScout pode ser implementado como um sistema de localização *indoor-outdoor* usando uma mistura de APs e receptores AeroScout.

A.6 Intel Place Lab

O Intel Place Lab [Cor11] representa outra abordagem de localização que usa WiFi. É possível ser utilizado *indoor-outdoor* [LaM+05]. Uma vez que o utilizador tenha uma

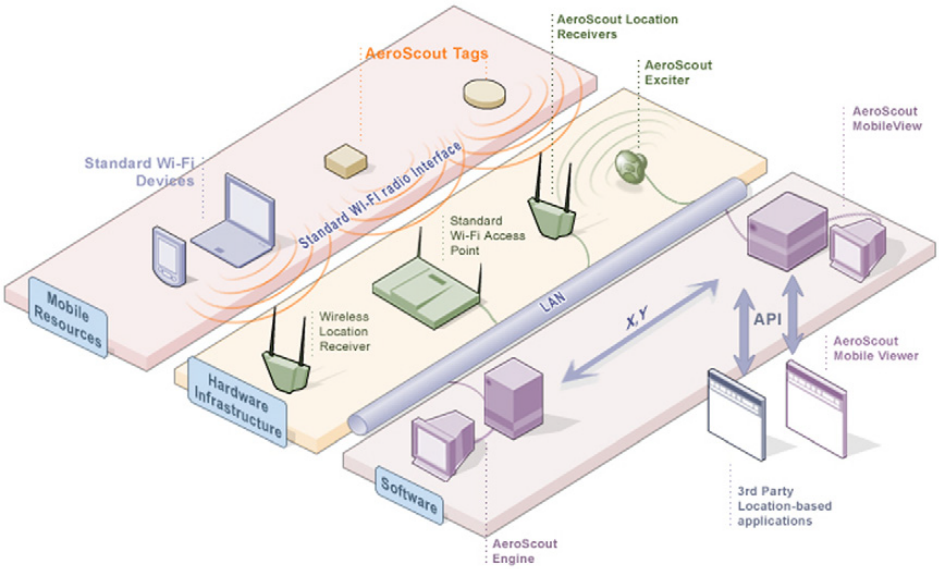


Figura A.2: Arquitetura do AeroScout [Aer] .

conexão WiFi, não há necessidade de hardware extra, transformando o Intel Place Lab num sistema de localização barato. O utilizador pode instalar o software de localização gratuitamente no site da Place Lab. A maioria dos sistemas de localização que usam WiFi exigem que o dispositivo se encontre na linha de visão dos APs. O Place Lab não precisa de linha de visão, mas apenas oferece uma precisão de 20 m. Na figura A.3 é apresentada uma visão geral dos componentes do Intel Place Lab. A aplicação cliente a correr no dispositivo do utilizador faz um levantamento da área a ser monitorada procurando APs e registando os IDs dos APs encontrados. O método para calcular a localização do dispositivo do utilizador é a triangulação, comparando o ID enviado pelos APs com as entradas registadas na base de dados e extraindo informações de localização. A base de dados vai sendo "incrementada" com o movimento dos seus utilizadores, sendo que sempre que um dispositivo encontra um AP novo , este é registado na base de dados. A precisão do sistema pode ser melhorada, aumentando o número de AP existentes. A vantagem de usar o Lab Place é o facto de ser completamente gratuito a nível de software, e de não ser necessário qualquer tipo novo de hardware para usar o sistema. A diferença entre esta abordagem e outros sistemas de localização (por exemplo, Ekahau) usando WiFi é que o Place Lab não requer uma pesquisa por rádio ou *fingerprints* e não requer que a posição dos APs seja fixa.

Um inconveniente da implantação de um sistema de localização baseado em 802.11 é o número reduzido de APs existentes em áreas menos povoadas [Krz06]. Para superar esse problema, o Intel Place Lab usa 2G (GSM) e dispositivos Bluetooth em paralelo com os APs 802.11, fornecendo uma precisão de $20 - 30m$. A precisão depende bastante da quantidade de APs registados na base dados para uma determinada área.

O Place Lab enfrenta menos problemas de privacidade, pois a localização do dispositivo do utilizador é calculada inteiramente no próprio dispositivo usando intervalos de

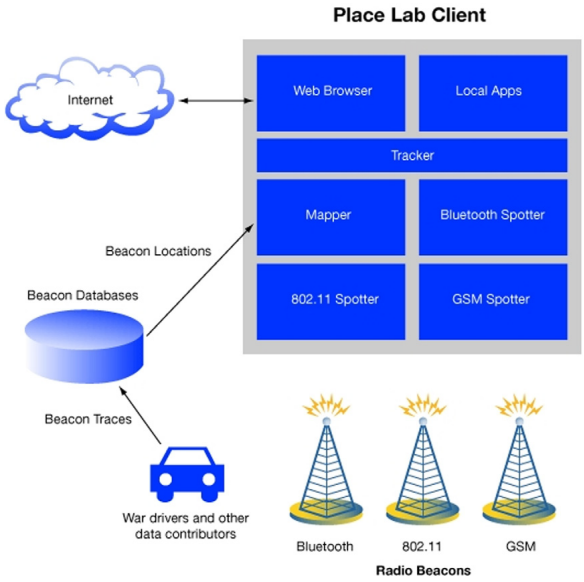


Figura A.3: Visão geral do funcionamento do *Intel Lab* [Cor11]

confiança em comparação com a base de dados para um local específico. Sempre que um novo cliente inicia sessão ou um cliente existente está a movimentar-se para um novo local, o dispositivo faz download de segmentos da base de dados dos APs na sua proximidade. O utilizador pode optar por fornecer a sua localização para receber informações relevantes da sua localização, tais como restaurantes, hotéis e correios mais próximos.

A.7 PinPoint 3D-iD

O *PinPoint 3D* é um produto comercial da *RFTechnologies* [Hig+00; RFT11]. O sistema utiliza tags RTLS que permitem obter informações da localização com um toque num botão. O sistema envia um pedido que obtém resposta usando o software *Help Alert*. O RTLS Pin-Point pode ser implementando usando a infra-estrutura WiFi existente ou usando uma rede ZigBee dedicada para ambientes sem infra-estruturas WiFi.

O PinPoint 3D-iD é um sistema comercial que complementa o sistema PinPoint com a utilização de *GNSS*. O sistema PinPoint requer dois componentes principais: o *multi-antenna interrogator* (uma matriz de antenas) e as *tags*. O *interrogator* envia beacons na frequência 2,44 GHz quando precisa de informação das *tags*. Quando a *tag* recebe o sinal, responde com um sinal de baixa potência de 5,8 GHz.

O PinPoint foi projetado para o ambiente *indoor*, sendo um sistema que é facilmente escalável, e que tem como principais desvantagens o preço face aos outros sistemas e a incapacidade de funcionamento *outdoor*.

A.8 Active Bats

O Active Basts [HH94; Wan+92] foi um dos primeiros sistemas de localização desenvolvidos na AT&T Cambridge. O Active Badges é implementado usando pequenas *tags* que emitem *beacons* no infravermelho a cada $10s$, e que devem ser transportadas pela pessoa a ser localizada. Os dados são recolhidos por um servidor central e guardados numa base de dados. O utilizador pode ter acesso às informações de localização usando uma API fornecida pelo servidor central. O alcance das etiquetas IR é de aproximadamente $6m$, não atravessando paredes. A tecnologia IR é barata, o que representa uma vantagem na implementação de sistemas de localização baseados em IR. Uma desvantagem do sistema Active Badges é não funcionar *outdoor*.

A localização de uma *tag* é apenas simbólica, ou seja, é indicada a sala onde a *tag* está localizada. O sistema é afetado por iluminação fluorescente e luz solar, o que faz com que nestas situações a precisão diminua. O tamanho da região para uma *tag* ser detetada é limitado a salas de tamanho pequeno ou médio . Ambientes maiores podem ser cobertos por múltiplos beacons infravermelhos. A funcionalidade pode ser estendida usando micro-controladores de baixa potência ou outra tecnologia mais complexa que pode oferecer suporte para mais funções [HB01].

A.9 BLIP Systems

Os sistemas BLIP são construídos sobre uma rede Bluetooth (Figura A.4), conhecida como BlipNet [Krz06; Sys11]. A rede BlipNet oferece acesso à *Local Area Network* (LAN) / *Wide Area Network* (WAN) via Bluetooth. Os componentes necessários para a implementação de uma rede BlipNet são os seguintes: dispositivos móveis com bluetooth integrado, BlipNodes, BlipServer e um servidor que executa o algoritmo de localização. Os dispositivos encontrados na proximidade de um BlipNode podem-se conectar a esse BlipNode de três maneiras:

1. Uma abordagem é registar um serviço chamado *Serial Port Profile* (SPP) no serviço BlipNode, que será usado pelo dispositivo para estabelecer a conexão.

2. Um segundo método é consultar continuamente a partir do BlipNodes os dispositivos ao seu alcance e fazer uma seleção dos dispositivos que podem ser usados na rede BlipNet. Esses métodos requerem alguma informação do dispositivo, como o endereço Bluetooth, e informações necessárias para estabelecer uma conexão usando um dos protocolos existentes.

3. O terceiro e último método para conectar um dispositivo a um BlipNode é usando uma conexão LAN enviando solicitações HTTP para um URL específico. Sempre que o dispositivo envia uma solicitação HTTP, o endereço Bluetooth pode ser verificado pelo BlipServer. O BlipServer passa os dados da conexão para o dispositivo.

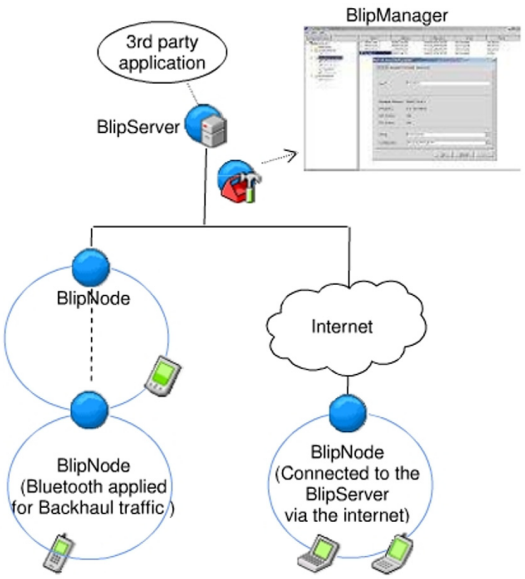


Figura A.4: Visão geral do sistema *BLIP* [Dea+12]

Um BlipNode representa um ponto de acesso Bluetooth (AP) que oferece acesso LAN via Bluetooth aos dispositivos móveis. Os BlipNodes requerem um BlipServer operacional. Um BlipNode estabelece uma conexão com o BlipServer através de LAN, Internet ou através de um link para outro BlipNode. O BlipServer representa o núcleo do Blip System e é responsável pela configuração, monitorização e controle dos BlipNodes na rede.

A localização Bluetooth oferece a possibilidade de localizar qualquer dispositivo móvel Bluetooth sem a necessidade de qualquer hardware adicional. O BlipNode pode tratar até 21 conexões ao mesmo tempo, o que para ambientes grandes ou grande número de dispositivos móveis pode ser visto como uma limitação. Outra desvantagem é o alcance do Bluetooth que depende da Classe Bluetooth. O alcance máximo é de aproximadamente 100m para a classe Bluetooth 1. O alcance pode ser aumentado com antenas direcionais e amplificadores de sinal, mas isso aumenta o custo de implementação.

A.10 MIT Cricket

O sistema híbrido MIT Cricket é um sistema de localização *indoor* desenvolvido pelo MIT [Krz06; Pri+00].

Como o nome indica, o sistema assemelha-se a um grilo, sendo composto por um dispositivo chamado de "Cricket" que contém um emissor de Beacons, e software para aplicações baseadas em localização que funcionam em vários dispositivos móveis. O sistema Cricket tem uma precisão de 1m a 3m e suporta a descoberta de múltiplos alvos. A posição do utilizador não é compartilhada a menos que o mesmo aceite anunciar essa informação. Assim, não há problemas de privacidade devido à localização a ser estimada no dispositivo móvel. Os dispositivos móveis escutam as informações publicadas pelos

Crickets, que são instalados em locais fixos no ambiente monitorado. Os Beacons transmitem informações específicas sobre o meio ambiente, que são usadas para calcular a localização.

A posição do utilizador é calculada utilizando uma mistura de mensagens de radiofrequência (RF) e impulsos de ultra-som. Os Cricket são montados na parede ou no teto usando uma banda AM de 418 MHz para enviar informações específicas do local. Cada mensagem RF é seguida por um pulso ultra-sónico. Após a receção das mensagens RF, os ouvintes utilizam o impulso ultra-sónico para estimar a distância a partir dos Beacons. Os Beacons utilizados no sistema Cricket são ativos e os ouvintes são passivos. Assim o sistema tem um par de benefícios. Primeiro, a escalabilidade do sistema não é afetada pelo aumento do número de ouvintes. Em segundo lugar, a localização é estimada no dispositivo usando as informações recebidas dos Beacons, o que significa que o Cricket não é um sistema de monitorizamento centralizado. Como mencionado anteriormente, os Crickets são pequenos dispositivos montados no teto ou noutro local que ofereçem linha de visão com os ouvintes.

As limitações de Cricket são a ausência de uma gestão centralizada e funções de monitorização. As solicitações computacionais também são bastante elevadas, o que significa um consumo de alta potência para sincronizar o pulso de ultra-som e os dados de RF.

A.11 Skyhook

O WPS Skyhook é um sistema de localização metropolitana *indoor-outdoor* que usa a infra-estrutura existente WiFi para estimar a localização [Sky11].

O Skyhook realiza localização com base em mapas [Krz06]. O WPS Skyhook é um sistema híbrido, que combina dados recolhidos nos dispositivos móveis com informações sobre localização de APs recolhidas por dispositivos exteriores que usam o *GNSS*. Ele explora a capacidade de obter o ID de quase todos os APs. O método de triangulação é usado para identificar a localização de um dispositivo, baseado no tempo que leva para um dispositivo receber uma resposta de qualquer AP disponível na área após o envio de uma solicitação de verificação.

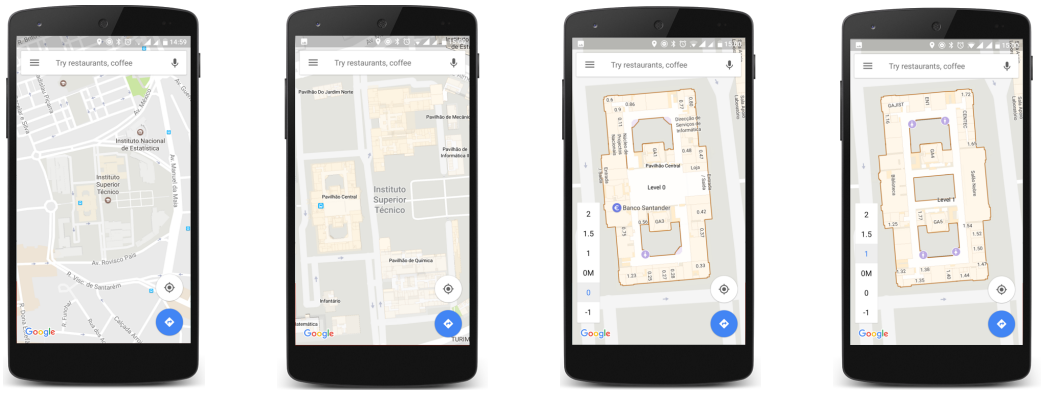
É um sistema de localização que permite localizar os utilizadores com uma precisão de $10 - 20m$. Os dados brutos são recolhidos a partir de pontos de acesso WiFi, satélites e torres de células disponíveis. Os dados são processados posteriormente usando algoritmos de localização híbrida. Um cliente Skyhook em execução no dispositivo móvel recolhe dados das fontes e envia-os para o Servidor de Localização, que retorna o local estimado. Para minimizar o custo dos dados e maximizar a duração da bateria, o cliente comunica com o servidor somente se o local não puder ser obtido localmente. Desta forma, a parte de otimização de energia, sincronização e cálculo da localização ficam ao encargo do cliente, ficando o servidor encarregue de fornecer as informações de localização mais atualizadas sempre que isso for requisitado pelo cliente.

A Skyhook tem um *Software Development Kit* (SDK) disponível gratuitamente, que permite mais facilmente criar aplicações de localização com os dados obtidos. O SDK está disponível para vários sistemas operativos e pode melhorar a precisão da informação de localização disponível em dispositivos móveis.

A.12 Google Maps

O Google Maps é provavelmente o sistema de localização mais conhecido e utilizado atualmente. É o resultado de um conjunto de serviços como o Google Earth e Google Street View, desenvolvidos pela Google.

Tipicamente os utilizadores utilizam o Google Maps em ambiente outdoor. Contudo existe em [Gooj] a possibilidade de criar o mapa indoor de uma determinada localização. Para isto, é necessário que se submeta um mapa bastante detalhado do interior da localização. Posteriormente, quando o utilizador navegar no interior de um edifício dessa zona, existe a possibilidade de fazer zoom no mapa fazendo a transição entre ambiente *indoor-outdoor* de uma forma simples, como é possível observar na figura A.5. No modo indoor existe uma opção de selecionar o piso para se visualizar os diferentes níveis do edifício.



a Outdoor

b Outdoor com mais informação que (a).

c Indoor Piso 0

d Indoor Piso 1

Figura A.5: Transição indoor-outdoor de visualização do Pavilhão Central do Instituto Superior Técnico.

É fornecida a possibilidade de calcular a rota dentro do edifício entre vários andares, navegar, ver fotos e panoramas em 360 graus. Há também a possibilidade de permitir notificações sobre locais próximos e o contexto envolvente. Em funcionamento indoor, são utilizadas as redes móveis e a rede WiFi para a orientação, obtendo-se uma precisão de posicionamento que varia de 5 a 10 metros [KS12] .

Em [Que16] são descritas as principais quatro funcionalidades da API(Interface de Programação de Aplicações) integrada no pacote *com.google.android.gms.location*, que permitem desenvolver aplicações sobre o Google Maps sem que o programador tenha que

saber informações sobre as tecnologias de localização. Estas funcionalidades são:

- API Location – periodicamente obtém a localização do utilizador, gastando o mínimo possível de energia. Pode ser utilizada para realizar o tracking do dispositivo, facilitando a receção e tratamento da localização;
- API Geofencing – é possível denotar zonas, sendo que sobre essas zonas o programador pode definir ações específicas como a entrada e saída de utilizadores. Desta forma, é possível monitorizar determinadas regiões geográficas, denominadas de geofences;
- API Activity Recognition – permitem obter a atividade do utilizador, como por exemplo, se este está andar a pé ou de carro;
- API Places – permite aceder a mais informação da Google, sendo assim possível os utilizadores obterem uma experiência mais contextual, utilizando informações de lugares e nomes dos mesmos para coordenadas específicas.

A.13 Comparação de sistemas

A Tabela A.1 mostra uma comparação sobre as características dos Sistemas de Localização ativos analisados anteriormente. A tecnologia de posicionamento mais utilizada para áreas *indoor* é a WLAN devido à infraestrutura já existente que é tipicamente aproveitada, tornando assim o sistema financeiramente mais económico. O RFID é mais adequado para ambientes densos. Tecnologias como: UWB, infravermelho, ultra-som, Bluetooth e também sistemas híbridos foram apresentados nesta secção.

É importante salientar que cada sistema tem a sua particularidade, mas todos necessitam de hardware específico e/ou um estudo e análise da região onde será implementado o sistema.

Tabela A.1: Comparação entre sistemas de localização ativa

Sistema	Tecnologia & Métricas	Algoritmo	Precisão	Dimensões	Custo
SpotON [Hig+00]	RFID RSSI	Ad-Hoc lateration	depende do tamanho do cluster	2D	Baixo
Ubisense [SG05]	UWB TDOA e AoA	Least square	~ 0.3 m	3D	Médio
Ekahau RTLS [Krz06]	WLAN RSSI	Algoritmo probabilístico	2 - 3 m	2D	Médio
Microsoft RADAR [Bah+00; BP00]	WLAN RSSI	kNN	2 - 4.3 m	3D	Médio
AeroScout [Aer]	WiFi ToA, TDOA e RSSI	Servidor AeroScout	1 - 5 m	2D	Baixo
Intel Place Lab [Cor11; LaM+05]	Triangulação WiFi	Triangulação baseada em mapas	20 - 30 m	N/A	Baixo
PinPoint 3D-iD [RFT11]	UHF TDOA e WiFi	Algoritmos de localização híbridos	1 - 10 m	3D	Alto
Active Bats [Wan+92]	Beacons infravermelhos	Bayesian	~ 0.09 m	3D	Médio
BLIP Systems [Sys11]	Bluetooth RSSI	Lateration	0.1 - 10 m	2D	Baixo
MIT Cricket [Krz06; Pri+00]	Beacons RF	Least square	~ 0.1 m	3D	Alto
Skyhook [Sky11]	WLAN, *GNSS* e redes móveis	Algoritmos de localização híbridos	10 - 30 m	N/A	Baixo
Google Maps [Gooj]	Redes móveis e WiFi	Triangulação	5 - 10 m	3D	Baixo

Apêndice B

Porquê o Android Studio?

B.1 Implementação: Ambientes de Desenvolvimento

"Software is the tool you use to solve a problem, not the problem itself. So use your time and brainpower to solve the problem, and let the tools give you a hand." - Arieh Bibliowicz

O *Integrated Development Environment* (IDE)(Ambientes de Desenvolvimento) é o software que consolida as ferramentas básicas que os programadores precisam para conceber e testar software. Normalmente, um IDE contém um editor de código, um compilador ou um interpretador e um depurador que o programador pode aceder por meio de uma *Graphical User Interface* (GUI)(interface gráfica).

Atualmente os dois principais IDE para Android são o Eclipse e o Android Studio. O Eclipse existe à bastante mais tempo que o Android Studio, por isso muitos dos exemplos existentes são para este IDE. Contudo, o Android Studio tem tido uma adesão gigantesca nos últimos 3 anos e vai sendo regularmente actualizado, tal como é habitual nos produtos criados pela Google. Nas últimas versões é possível migrar a maioria dos projetos criados em Eclipse para Android Studio, o que faz com que a transição dos programadores não seja muito brusca. É importante salientar que o Eclipse foi desenvolvido para diferentes linguagens e plataformas de programação, enquanto que o Android Studio se foca unicamente em Android. Em conclusão, ambos os IDEs conseguem atingir os mesmos objetivos para programar Android, mas o Android Studio apresenta um leque superior de ferramentas o que faz com que este tenha sido o ambiente escolhido para o desenvolvimento deste projeto

B.1.1 Android Studio

O IDE Android Studio [Goob] foi criado pela Google em Maio 2013. Foi construído exclusivamente para o desenvolvimento de aplicações Android, e por isso apresentada

diversas vantagens que permitem a aceleração e simplicidade do desenvolvimento das apps, sendo as principais:

1. **Integração usando Gradle:** Gradle é um sistema de compilação que usa os melhores recursos de vários outros sistemas de compilação, combinando-os. É um sistema de compilação baseado em *Java Virtual Machine* (JVM), o que significa que é possível criar um script em Java, que pode ser posteriormente utilizado pelo Android Studio.

   O sistema de compilação automaticamente leva todos os arquivos *source* (.java ou .xml), e aplica a ferramenta apropriada (por exemplo, toma os arquivos do tipo java e converte-os em arquivos dex) e agrupa todos num arquivo compactado, o APK.

2. **Auto-complete de código:** O sistema é baseado em JetBrains, por isso estão disponíveis todas as suas ferramentes de ajuda.

3. ***User Interface* (UI):** As ferramentas e itens do menu no Android Studio tendem a estar onde queremos e ser relativamente fácil encontrá-las.

4. **Organização do projeto:** O Android Studio usa módulos para gerir e organizar os pacotes de código e tem arquivos de compilação do Gradle (invisíveis para o programador), o que significa que ele pode indicar suas próprias dependências.

5. ***Drag-and-Drop*** O Android Studio tem uma GUI que simplifica bastante a construção das interfaces gráficas que cada vez mais importância e influência têm para os utilizadores. Na figura B.1 é possível observar a utilização desta funcionalidade.

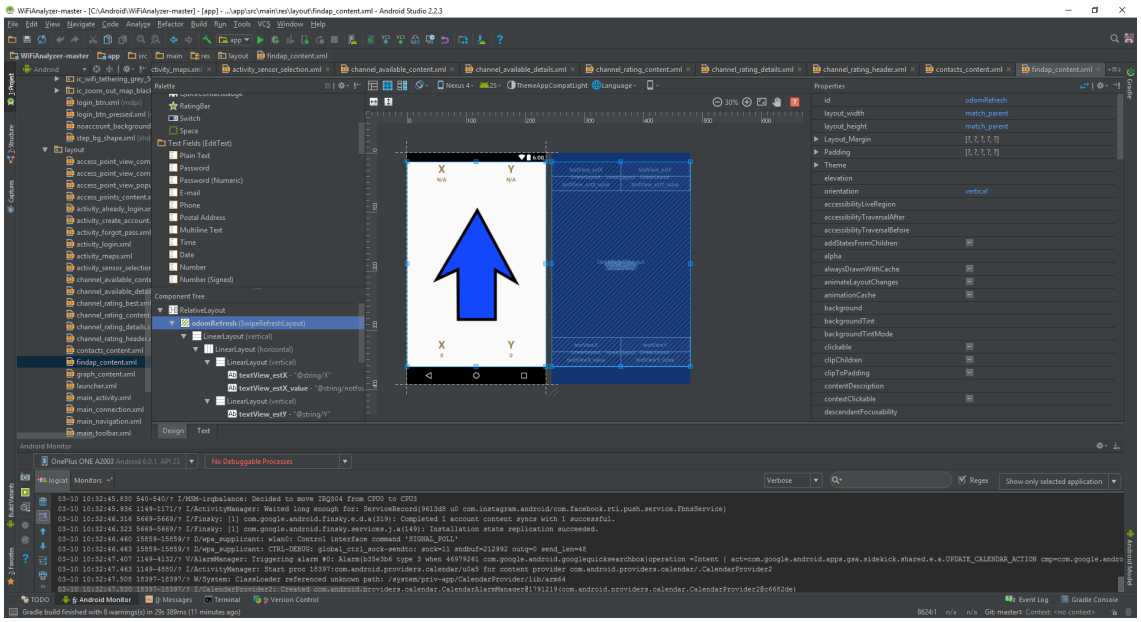


Figura B.1: Android Studio IDE.

A última versão do Android Studio é a 2.2.3 e como é baseado em JetBrains, tem disponíveis todas as suas ferramentes de ajuda. Este ambiente está representado na figura

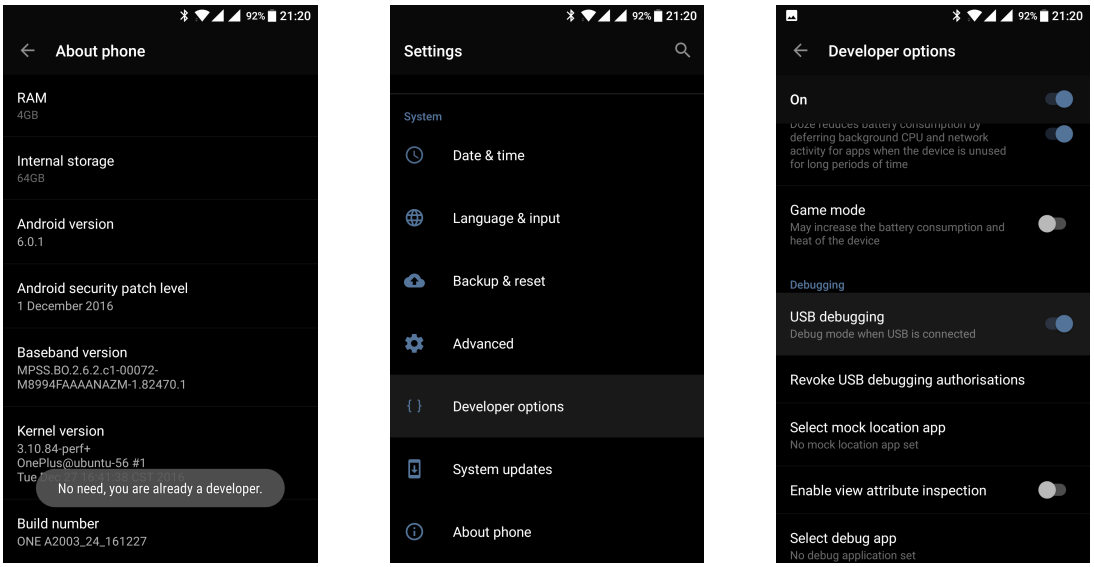
B.1. Esta foi a plataforma escolhida para o desenvolvimento da aplicação por todas as razões previamente referidas.

Para a utilização de todo o potencial deste IDE convém activar o modo depurador e o uso de bibliotecas adequadas para cada aplicação. No apêndice C está mostrado como o fazer.

Apêndice C

Utilização do Android Studio - Preparação

O Android Studio apresenta muitas ferramentas que aumentam a produtividade e melhoram a organização do trabalho. Uma das mais importantes é o modo *debug*. Para utilizar este modo é necessário activar o modo de programador no smartphone. Para isso é aconselhável seguir os seguintes passos:



a Activar o modo programador

b Entrar nas opções de programador

c Activar o modo *debug*

Figura C.1: Modo *debug* por USB

1. **Activar o modo programador:** Para ativar o modo de programador, deve-se ir a Definições->Acerca do Telemóvel, e nesta secção carregar cinco vezes no número de construção como é possível observar na figura C.1a.

2. **Activar o modo *debug*:** Se voltar às definições, terá aparecido um novo menu, o menu do modo programador, apresentado na figura C.1b. Ao entrar no mesmo estão disponíveis várias propriedades do telemóvel, em que uma delas é a possibilidade de fazer *debug* a aplicações. Deve-se activar aí o modo *debug*, como está representado na C.1c

Além disso, existem bibliotecas que permitem poupar algum trabalho, como é o caso da biblioteca de gráficos utilizado na aplicação. Basta para tal adicionar nas dependências da gradle que compile as bibliotecas, por exemplo:

Listagem C.1: Adicionar bibliotecas

```
1  //Google Layouts
2  compile 'com.android.support:appcompat-v7:25.1.0'
3  compile 'com.android.support:design:25.1.0'
4  compile 'org.apache.commons:commons-lang3:3.5'
5  //Graficos
6  compile 'com.jjoe64:graphview:4.2.1'
7  //Google Maps
8  compile 'com.google.android.gms:play-services:10.0.1'
9  //Google Account Connection
10 compile 'com.google.android.gms:play-services-ads:10.0.1'
11 compile 'com.google.android.gms:play-services-auth:10.0.1'
12 compile 'com.google.android.gms:play-services-gcm:10.0.1'
```

Apêndice D

WiFi Finder: Manual do utilizador

Neste Apêndice encontra-se presente alguma informação sobre a aplicação desenvolvida nesta dissertação, o **WiFi Finder** .

Para mais informação recomenda-se ao leitor visitar `https://github.com/dario-pedro/wifi_finder`, onde pode encontrar todo o código Android desenvolvido. Caso tenha interesse em saber mais sobre o site e/ou deseje descarregar a aplicação para o seu telemóvel pode aceder a `http://dariopedro.com/wififinder.php`, que está representado na figura D.1 .

Figura D.1: Servidor e Website que suporta a aplicação.

D.1 Aplicação

A aplicação desenvolvida para dispositivos Android está preparada parar correr em qualquer smartphone que suporte uma versão igual ou superior à *Jelly Bean* (versão 4.1 - API 16). Para se ter a melhor experiência possível é recomendado que se utilize um dispositivo com a versão *Marshmallow* (versão 6.0 - API 23).

Esta apresenta-se dividida em três atividade principais:

1. **Launcher** (Carregamento): Assim que o utilizador corre a aplicação, o Launcher é corrido e apresenta uma animação enquanto verifica se já existe um utilizador associado ao dispositivo. Caso exista, envia a informação do mesmo para o servidor de modo a realizar a autentificação, caso contrário gera um *intent* [1] para a próxima atividade, o Login.

2. **Login** (Autentificação): Atividade que possibilita ao utilizador fazer operações sobre a sua conta, como por exemplo a autentificação.

3. **Main Activity** (Actividade Principal): Nesta atividade é possível aceder (através do menu lateral) aos diversos fragmentos e atividades que constituem a aplicação, sendo que é aqui que são guardadas todas as informações (i.e. testes realizados, informação do utilizador, etc).

A sequência de transições entre estas três atividades pode ser observada na figura D.2.

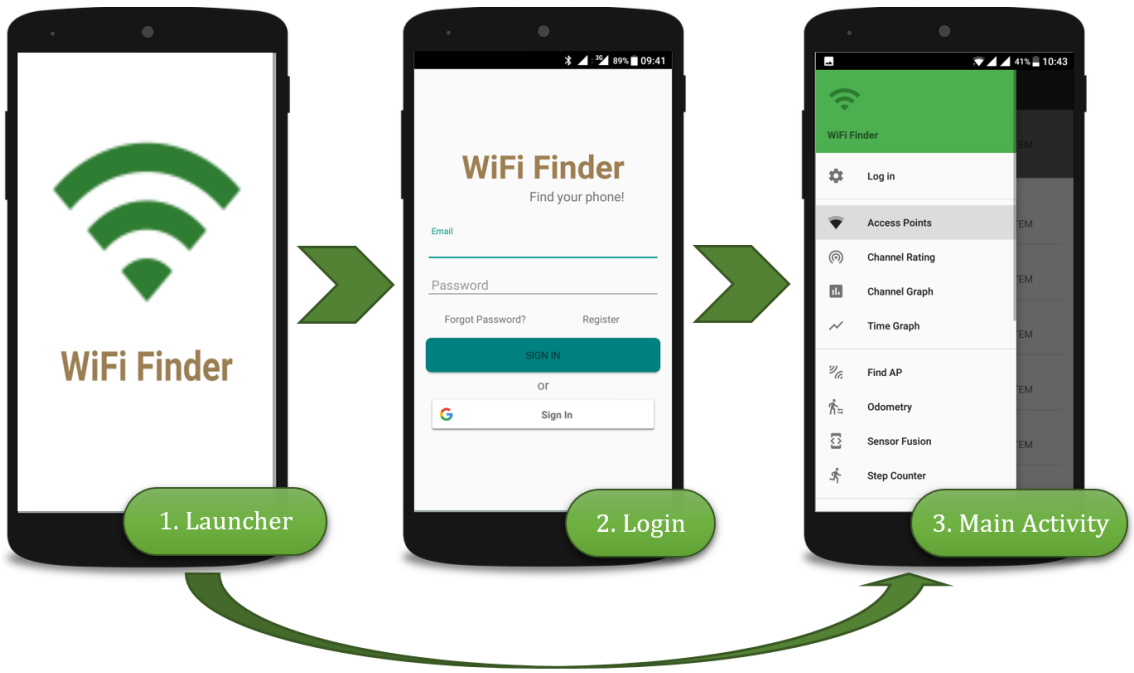


Figura D.2: Sequência de transição de atividades até à atividade principal.

A partir da Atividade Principal o utilizador pode navegar para as diversas atividades[2]

[1] Em Android um *intent* é uma descrição abstrata de uma operação a ser realizada[Good].
[2] Uma atividade tem que ser única e gere quase todas as interações com o utilizador [Gooa].

e fragmentos[3] da aplicação. Em seguida, serão descritas as principais funcionalidades dos principais fragmentos e atividades.

Gráfico dos Canais

O fragmento do Channel Graph(Gráfico dos Canais) permite que o utilizador visualize a potência do sinal no eixo das ordenadas, separados pelos diversos canais no eixo das abcissas, como está representado na figura D.3a. O gráfico é atualizado à frequência a que é feito scan ao meio, que é definido nas Definições. Também existe a opção de deslizar para atualizar manualmente o conteúdo da tela.

Odometria

O separador de Odometry (Odometria) permite testar o nó desenvolvido no capítulo 3.2, como pode ser observado na figura D.3b.

Definições

O segmento das Settings (Definições) é uma atividade que permite guardar na memória do telemóvel algumas predefinições do utilizador. Na figura D.3c estão exemplificados os tipos de variáveis que é possível definir.

APs nas proximidades

Este fragmento apresenta informações de todos os emissores que se encontram no alcance do smartphone, sendo possível visualizar alguma informação sobre os mesmos como potência, distância, SSID e a frequência. A interface APs nas proximidades pode ser observada na figura D.3d.

Gráfico Temporal

O Time Graph (Gráfico Temporal) é semelhante ao Gráfico dos Canais, mas permite fazer uma análise da potência ao longo do tempo. Este tipo de análise permite, por exemplo, avaliar como o sinal flutua enquanto o dispositivo se encontra num regime estacionário, como está representado na figura D.3e.

Localização Atual

A Localization (Localização Atual) utiliza a localização Android e o Maps para mostrar a posição do utilizador. Este módulo deve ser preferencialmente utilizado em ambiente outdoor.

[3] Um Fragmento representa o comportamento ou uma parte da interface do usuário, sendo possível uma atividade conter vários fragmentos a correr em simultâneo [Gooc].

Detecção de Passos

O separador de Passos permite testar as várias opções implementadas nesta dissertação para a descobrir o evento de um passo. Este módulo pode ser visualizado na figura D.3f.

Sensor Fusion

De forma semelhante ao módulo de detecção de Passos, a atividade Sensor Fusion (Orientação) permite fazer testes com a detecção da orientação do dispositivo.

Exportação

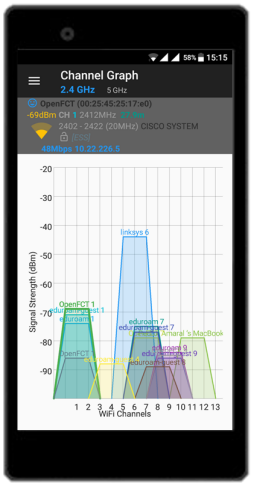
A atividade de Export (Exportação) estende uma outra atividade nativa de Android (Chooser Activity) que permite ligar a app com as restantes apps do smartphone que consigam partilhar informação como é possível observar na figura D.3g. Desta forma, é possível enviar a informação gerada pela aplicação por email ou Facebook.

Procurar APs

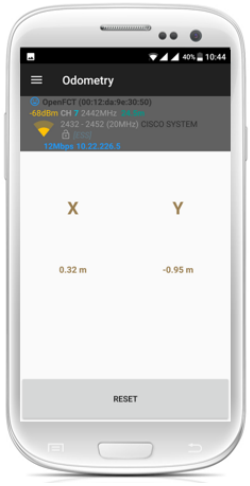
O fragmento Find APs (Procurar APs) incorpora praticamente todo o trabalho realizado ao longo da dissertação, e realiza a estimação da posição do AP ao qual o dispositivo se encontra conectado. Utilizando a informação de Odometria e da estimação, aponta a seta apresentada na figura D.3h do utilizador para o AP.

Chat de Grupo

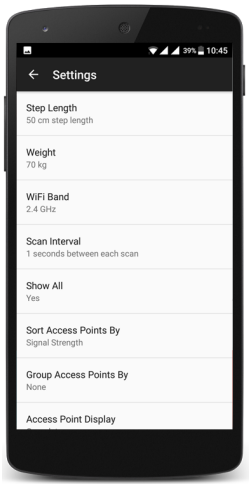
O Multicast Chat (Chat de Grupo) permite que os utilizadores de um determinado grupo comuniquem entre si. Utilizando este módulo é possível fazer testes da distribuição de informação das estimações e medições do meio realizados pelos diversos smartphones. O Chat pode ser observado na figura D.3i.



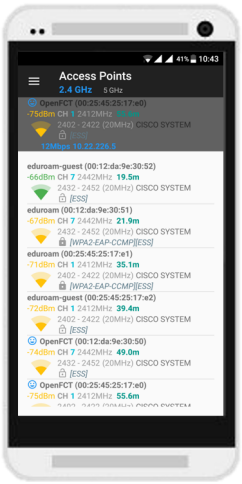
a Gráfico dos Canais.



b Odometria.

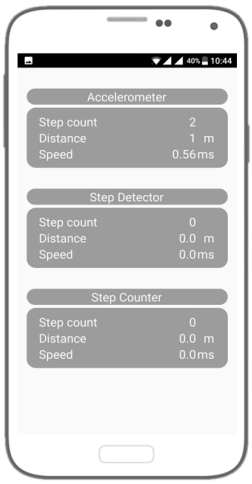


c Definições.

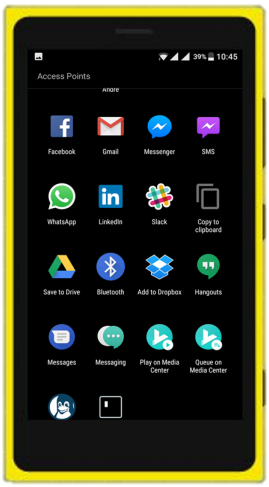


d APs.

e Gráfico Temporal

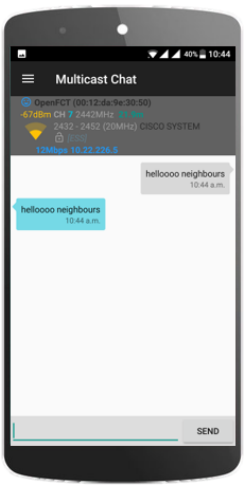


f Detecção de Passos.



g Exportação.

h Procurar APs.



i Chat.

Figura D.3: Exemplos de Fragmentos da aplicação WiFi Finder.

Apêndice E

Triangulação

A triangulação permite obter soluções para problemas iguais aos da figura E.1.

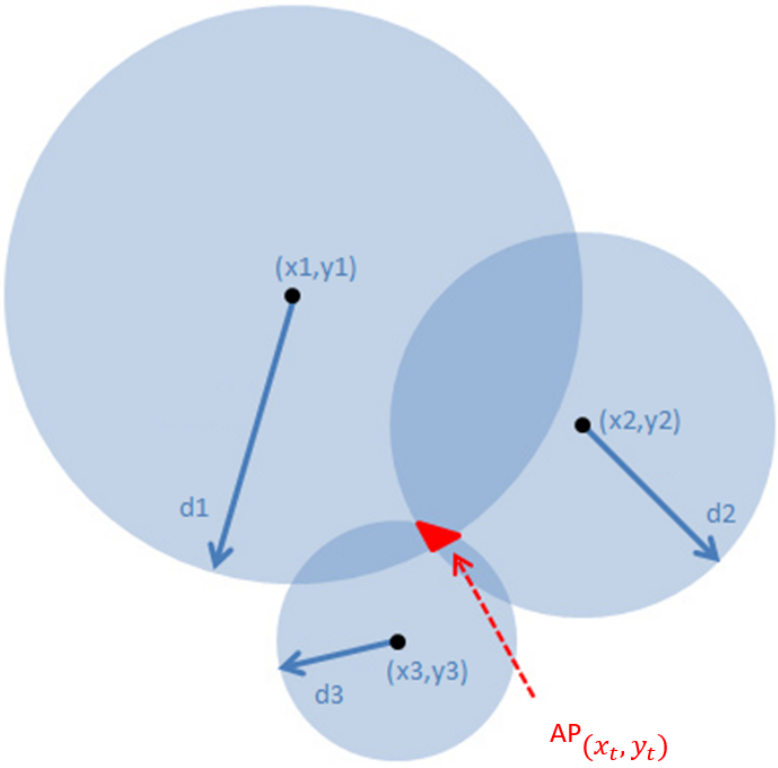


Figura E.1: Trilateração

Supondo que existem 3 posições conhecidas $(x_i, y_i), i = 1, 2, 3$, e que um AP se encontra na posição (x_t, y_t) a uma distância $d_i, i = 1, 2, 3$, utilizando o teorema de Pitágoras, podemos obter o sistema de equações E.1,

$$\begin{cases} (x_1 - x_t)^2 + (y_1 - y_t)^2 = d_1^2 \\ (x_2 - x_t)^2 + (y_2 - y_t)^2 = d_2^2 \\ (x_3 - x_t)^2 + (y_3 - y_t)^2 = d_3^2 \end{cases}. \tag{E.1}$$

Para resolver este sistema de equações, é mais conveniente realizar a transformação de coordenadas ,

$$\begin{gathered} (x_2, y_2) \rightarrow (0, 0) \\ (x_2, y_2) \rightarrow (\Delta x_2 = x_2 - x_1, \Delta y_2 = y_2 - y_1) \\ (x_3, y_3) \rightarrow (\Delta x_3 = x_3 - x_1, \Delta y_3 = y_3 - y_1) \\ (x_t, y_t) \rightarrow (\Delta x_t = x_t - x_1, \Delta y_t = y_t - y_1) \end{gathered}. \tag{E.2}$$

Podemos desta forma rescrever as equações iniciais como,

$$\begin{cases} \Delta x_t^2 + \Delta y_t^2 = d_1^2 \\ (\Delta x_2 - \Delta x_t)^2 + (\Delta y_2 - \Delta y_t)^2 = d_2^2 \\ (\Delta x_3 - \Delta x_t)^2 + (\Delta y_3 - \Delta y_t)^2 = d_3^2 \end{cases}. \tag{E.3}$$

Subtraindo à primeira equação as outras duas, obtemos as duas equações,

$$\begin{cases} 2\Delta x_2 \Delta x_t + 2\Delta y_2 \Delta y_t = d_1^2 + \Delta x_2^2 + \Delta y_2^2 - d_2^2 \\ 2\Delta x_3 \Delta x_t + 2\Delta y_3 \Delta y_t = d_1^2 + \Delta x_3^2 + \Delta y_3^2 - d_3^2 \end{cases}. \tag{E.4}$$

Escrevendo na forma matricial obtemos

$$2 \begin{bmatrix} \Delta x_2 & \Delta y_2 \\ \Delta x_3 & \Delta x_3 \end{bmatrix} * \begin{bmatrix} \Delta x_t \\ \Delta x_t \end{bmatrix} = \begin{bmatrix} d_1^2 + \Delta x_2^2 + \Delta y_2^2 - d_2^2 \\ d_1^2 + \Delta x_3^2 + \Delta y_3^2 - d_3^2 \end{bmatrix}, \tag{E.5}$$

e colocando em evidência $(\Delta x_t, \Delta y_t)$, obtemos

$$\begin{bmatrix} \Delta x_t \\ \Delta x_t \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \Delta x_2 & \Delta y_2 \\ \Delta x_3 & \Delta x_3 \end{bmatrix}^{-1} \begin{bmatrix} d_1^2 + \Delta x_2^2 + \Delta y_2^2 - d_2^2 \\ d_1^2 + \Delta x_3^2 + \Delta y_3^2 - d_3^2 \end{bmatrix}. \tag{E.6}$$

Resolvendo o sistema matricial em ordem a $(\Delta x_t, \Delta y_t)$, obtemos

$$\begin{cases} \Delta x_t = \frac{d_1^2 - d_2^2 + \Delta x_2^2 + \Delta y_2^2}{2(\Delta x_2 - \Delta y_2)} - \frac{\Delta y_2 (d_1^2 - d_3^2 + \Delta x_3^2 + \Delta y_3^2)}{2(\Delta x_2 \Delta x_3 - \Delta x_3 \Delta y_2)} \\ \\ \Delta y_t = \frac{\Delta x_2 (d_1^2 - d_3^2 + \Delta x_3^2 + \Delta y_3^2)}{2(\Delta x_2 \Delta x_3 - \Delta x_3 \Delta y_2)} - \frac{d_1^2 - d_2^2 + \Delta x_2^2 + \Delta y_2^2}{2(\Delta x_2 - \Delta y_2)} \end{cases}, \tag{E.7}$$

Fazendo a transformação inversa para as variáveis iniciais, ficamos com

$$\begin{cases} x_t = \frac{d_1^2 - d_2^2 + (x_2 - x_1)^2 + (y_2 - y_1)^2}{2((x_2 - x_1) - (y_2 - y_1))} - \frac{(y_2 - y_1)(d_1^2 - d_3^2 + (x_3 - x_1)^2 + (y_3 - y_1)^2)}{2((x_2 - x_1)(x_3 - x_1) - (x_3 - x_1)(y_2 - y_1))} + x_1 \\ \\ y_t = \frac{(x_2 - x_1)(d_1^2 - d_3^2 + (x_3 - x_1)^2 + (y_3 - y_1)^2)}{2((x_2 - x_1)(x_3 - x_1) - (x_3 - x_1)(y_2 - y_1))} - \frac{d_1^2 - d_2^2 + (x_2 - x_1)^2 + (y_2 - y_1)^2}{2((x_2 - x_1) - (y_2 - y_1))} + y_1 \end{cases} \tag{E.8}$$

Para finalizar, é importante notar que nesta dissertação as posições conhecidas $(x_i, y_i), i = 1, 2, 3$, são medições realizadas por um smartphone ao longo do percurso do utilizador, e

que a distância a que o utilizador se encontra do AP é calculado a partir do RSSI. Sendo assim, uma representação mais realista do esquema está presente na figura E.2.

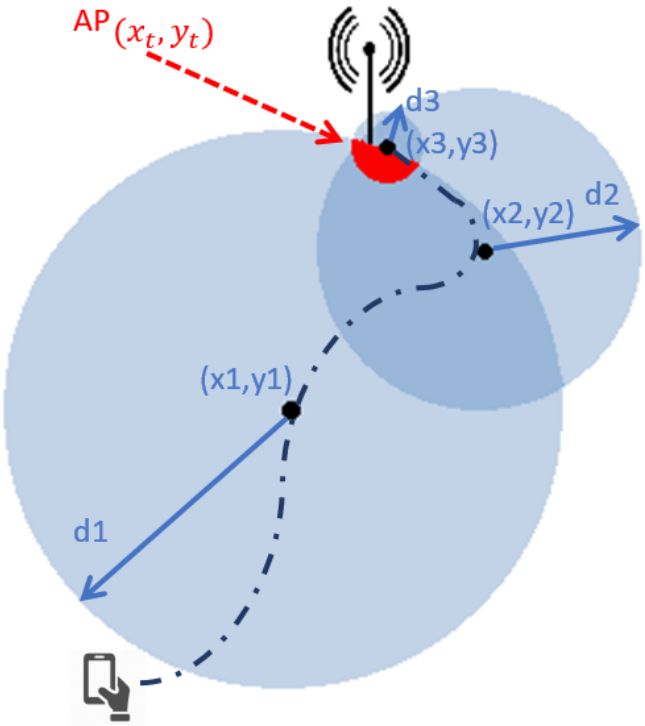


Figura E.2: Trilateração com pontos provenientes de odometria.

APÊNDICE F

Método de Gauss-Newton Regularizado

Alguns dos problemas do algoritmo WLS podem ser corrigidos com o uso de informação prévia.

Como sugestão de que tipo de informação prévia utilizar, o estudo de resultados numéricos dos métodos de forma fechada, indicou que os métodos têm alguma preferência embutida para posições perto das coordenadas mais distantes. Este é um comportamento razoável, uma vez que um receptor consegue obter medições de maior precisão caso esteja perto do emissor. Este tipo de suposição anterior pode ser facilmente incorporada no algoritmo de Gauss-Newton como um termo de regularização. Uma justificação teórica para este acontecimento é que o algoritmo resultante é equivalente a um algoritmo Bayesiano máximo-a-posteriori com distribuição prévia escolhida em torno do ponto central.

Para se implementar este método basta modificar o cálculo de Δt do algoritmo apresentado em 4.3.2.7. Desta forma, para utilizar este algoritmo, basta seguir o procedimento:

1. Escolher uma estimativa inicial t_0 e tolerância de paragem δ A. Definir $k = 0$.

2. Calcular

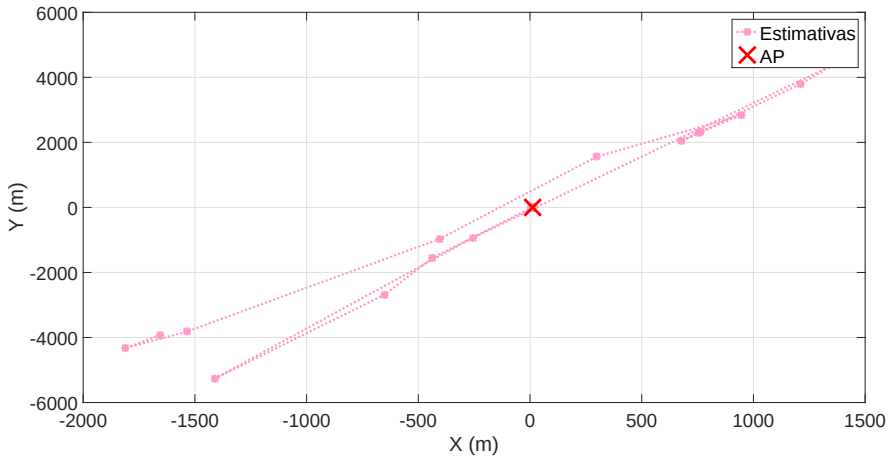
$$J_k(t) = \begin{bmatrix} \frac{(s_1 - t)^T}{\|(s_1 - t)\|} \\ \vdots \\ \frac{(s_N - t)^T}{\|s_N - t\|} \end{bmatrix} \tag{F.1}$$

3. Definir $t_{k+1} = t_k + \Delta t_k$ onde

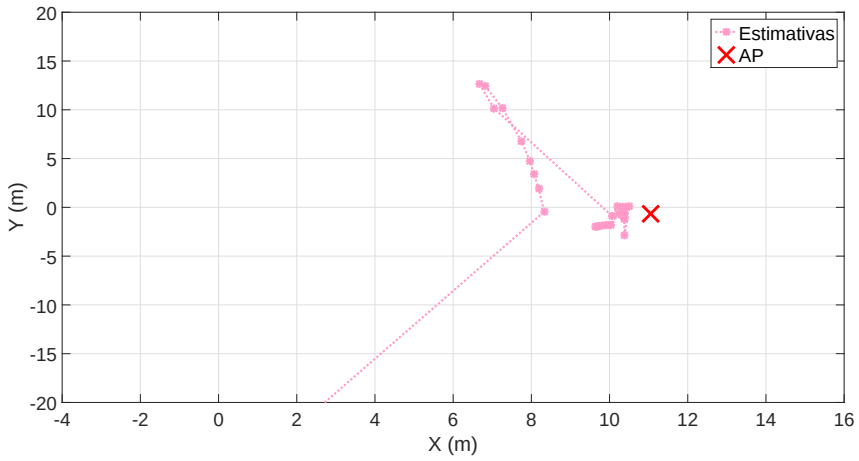
$$\Delta t_k = -\frac{\sum^{-\frac{1}{2}} J_k + cI}{\sum^{-\frac{1}{2}} (h(t_k) - d) + c(t - t_d)}\,, \tag{F.2}$$

onde c representa o coeficiente de normalização e t_d denota o ponto média dos pontos de odometria S.

4. Se a condição $\|\Delta t_k\| < \delta$ não for satisfeita e $k < K_{max}$, incrementar k e repetir e voltar a repetir o ponto 2.



a Normal.



b Zoom.

Figura F.1: Testes de estimação utilizando o algoritmo Gauss-Newton Regularzado.

Nas figuras em 4.10 estão representados os resultados das estimações utilizando o algoritmo de Gauss-Newton regularizado. Como é possível observar, quando se aplica a regularização ao método de Gauss-Newton, as estimações tendem a oscilar menos, e a atingir valores de erro máximo inferior.

Apêndice G

Relaxação Tomic SOCP com Potência de transmissão desconhecida

Este apêndice demonstra como realizar a relaxação Tomic SOCP para o caso em que a Potência de transmissão P_t é desconhecida [Tom+15]. A suposição de que é possível o smartphone conhecer a potência de transmissão das fontes pode ser muito forte na prática, pois exigiria hardware adicional tanto na fonte quanto nos dispositivos móveis. Desta forma, o valor das perdas ao longo do percurso L_0 é assumido como desconhecido e tem que ser estimado. A estimativa conjunta ML de t e L_0 pode ser formulada como

$$\hat{\theta} = \underset{\theta=[t;L_0]}{arg\ min} \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \left[\left(L_i - l^T \theta \right) - 10\ \gamma\ log_{10} \frac{\left\| A^T \theta - s_i \right\|}{d_0} \right]^2 , \tag{G.1}$$

onde $l = [0_{2\times1}; 1]$ e $A = [I_2; 0_{1\times2}]$.

Assumindo que P_t é desconhecido, do modelo de propagação 4.31,podemos escrever que

$$\psi_i \left\| t - s_i \right\| \approx \eta d_0 , \tag{G.2}$$

onde $\psi_i = 10^{-\frac{L_i}{10\gamma}}$ e $\eta = 10^{-\frac{L_0}{10\gamma}}$.

Uma maneira de estimar a localização do AP t é através mínimos quadrados. Assim, de acordo com G.2, o problema de estimação pode ser formulado como

$$\hat{t} = \underset{\boldsymbol{t},\boldsymbol{\eta}=[\boldsymbol{t}^T,b_i]^T}{\arg\min} \sum_{i=1}^{N} (\psi \| [\boldsymbol{t} - \boldsymbol{s}_i \| - \eta d_0)^2 . \tag{G.3}$$

Introduzindo as variáveis auxiliares $r_i = \|t - s_i\|$ e $\|z\|^2 = p$; e utilizando *Second Order Cone Constraint* (SOCC), isto é, $r_i \geq \|t - s_i\|$ e $\|z\|^2 \leq p$ conseguimos realizar a estimação, resolvendo o problema SOCP

$$\begin{aligned} &\underset{t,g,z,\eta,p}{\text{minimize}}\ p \\ &\text{subject to}\ \ \|[2z; p-1]\| \leq p+1,\ \|t-s_i\| \leq g_i,\ z_i = \psi_i g_i - \eta d_0,\ i=1,\ldots,N\ . \end{aligned} \tag{G.4}$$

Mesmo que a abordagem em G.4 resolva eficientemente G.1, podemos melhorar ainda mais seu desempenho. Para isso, podemos explorar as estimativas ML de $L0$, ou seja, $\hat{L}_0$, que é calculada pela expressão

$$\hat{L}_0 = \frac{\sum_{i=1}^{N}\left(L_i - 10\gamma \log_{10}\frac{\|\hat{t}'-s_i\|}{d_0}\right)}{N}\ . \tag{G.5}$$

Depois, este valor estimado pode ser usado para resolver um outro problema SOCP, considerando que a potência é conhecida em G.5.

Rescrevendo a equação G.2, podemos chegar ao problema da seguinte forma :

$$\hat{\alpha}_i \|t-s_i\| \approx d_0\ , \tag{G.6}$$

onde $\hat{\alpha}_i = 10^{\frac{L_0-L_i}{10\gamma}}$.

Introduzindo a variável auxiliar $\gamma_i = r_i^2$, expandindo G.3 e retirando d_0^2 (que não tem efeito no problema de minimização por ser constante), obtém-se

$$\begin{aligned} &\underset{t,\gamma,r}{\text{minimize}}\ \left(\hat{\alpha}_i^2\gamma i - 2d_0\hat{\alpha}_i r_i\right) \\ &\text{subject to}\ \ \gamma_i = r_i^2,\ r_i^2 = \|t-si\|,\ i=1,\ldots,N \end{aligned} \tag{G.7}$$

onde $\hat{\alpha}_i = 10^{\frac{L_i-\hat{L}_0}{10\gamma}}$. É possível relaxar este problema numa otimização convexa, em que a restrição não convexa $\gamma_i = r_i^2$ pode ser substituída pela SOCC (restrição de cone de segunda ordem) $r_i^2 \leq \gamma_i$. Definindo como variável auxiliar $y = \|t\|^2$, e relaxando esta restrição $y \geq \|t\|^2$ que é evidentemente um SOCC. Fazendo uso das restrições, o problema em G.7 pode ser aproximado SOCP, como o problema de otimização

$$\begin{aligned} &\underset{t,\gamma,r,y}{\text{minimize}}\ \left(\hat{\alpha}_i^2\gamma i - 2d_0\hat{\alpha}_i r_i\right) \\ &\text{subject to} \\ &\left\|[2t; y-1]\right\| \leq y+1,\ \left\|[2r_i; \gamma_i-1]\right\| \leq \gamma_i+1,\ \gamma_i = y - 2s_i^T t + \|s_i\|^2,\ i=1,\ldots,N \end{aligned} \tag{G.8}$$

Apêndice H

Case Study : Localização de dispositivos remotos num cenário veicular

Os veículos precisam de localizar outros veículos e elementos da infra-estrutura de rede em sistemas de UAV. Os passageiros humanos também precisam de localizar e ser localizados pelos veículos, preferencialmente usando um dispositivo portátil, como um smartphone.

Neste apêndice foi analisado o desempenho dos algoritmos de localização considerando um cenário de veículo usando as informações de localização fornecidas pelo serviço Android, que tem uma precisão esperada para um terminal estático de cerca de 8 m para A-GPS [Zan09]. Um AP estático foi colocado no local representado na figura H.1 com P_T = 28 dBm. Durante os testes, todas as medidas RSS de APs conhecidos, a localização do terminal e a hora atual foram armazenadas em tempo real. As medições RSS estão contaminadas com ruído e há flutuações de sinal que aumentam a distância estimada do AP. Elas foram registadas a cada segundo usando um smartphone OnePlus 2, com Android 6.0.1. A aplicação permite que o utilizador exporte os dados, que posteriormente foram processados em Matlab, e todos os algoritmos descritos na secção 4 foram executados usando os mesmos dados. Os algoritmos estudados na dissertação foram corridos com $\gamma^0 = 2.5$, $\gamma_{min} = 1.5$, $\gamma_{max} = 3.5$ e $K^T_{max} = 3$. A métrica utilizada para comparar a precisão das estimações foi a distância euclidiana.

Foram realizados dois conjuntos de testes, seguindo o caminho descrito na Figura H.1, dentro do Campus Caparica da Universidade da NOVA de Lisboa, a duas velocidades médias diferentes: 20 km/h e 30 km/h. A figura H.2 mostra os valores RSS medidos nos dois testes. Devido à taxa de amostragem constante, foi medida uma menor densidade de amostras a maior velocidade e os valores de RSS medidos têm um intervalo de variação mais estreito, provavelmente devido a alguma filtragem implementada na biblioteca do Android.

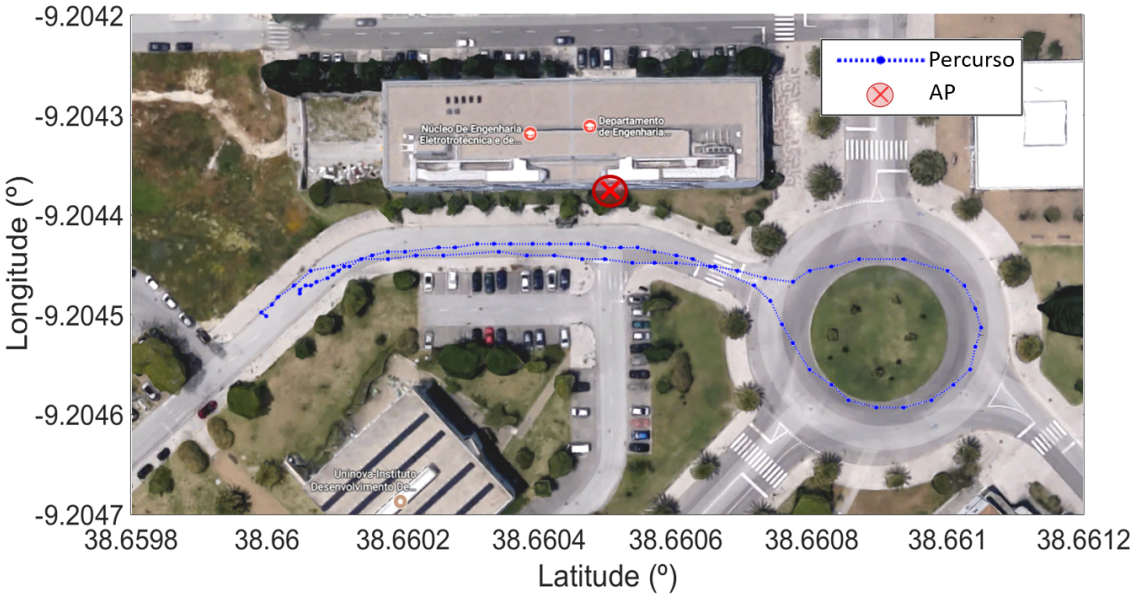


Figura H.1: Cenário de testes veiculares.

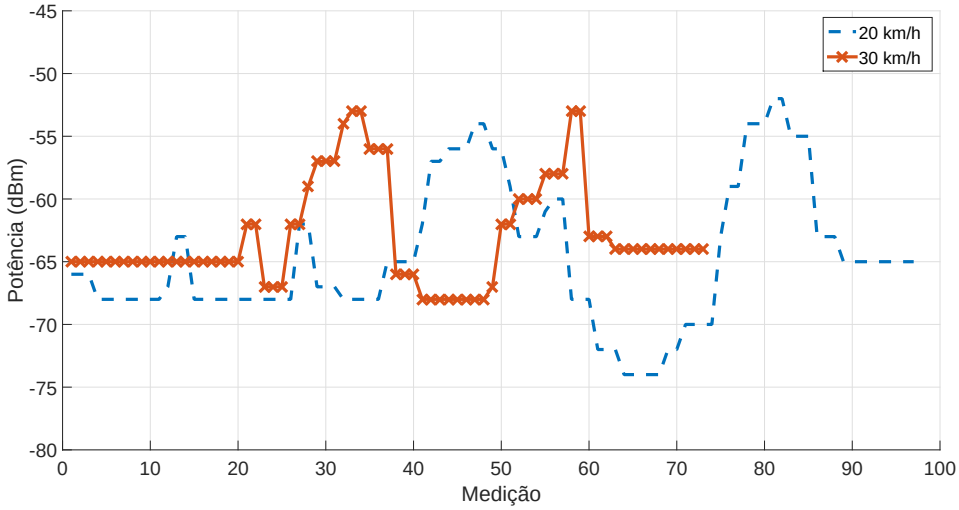


Figura H.2: RSS medido a 20 km/h e 30 km/h.

As figuras H.3 e H.4 descrevem a distância entre a localização estimada e a localização exata do AP, após a medição i para as duas velocidades. Observe que a amostra i usa as todas as medições anteriores, e algumas variações podem ocorrer devido a novas medições que contêm um erro maior.

Nas figuras H.3 e H.4 pode ser visto que a precisão diminuiu para uma velocidade mais alta, provavelmente devido aos valores de localização do veículo e RSS serem menos precisos. A 20 km/h, foi possível ter estimativas de localização com uma precisão abaixo de 40m com apenas 2 ou 3 amostras de RSS, enquanto que a 30 km/h só foi possível ter uma precisão abaixo de 50m após 3 medições para dois dos métodos. A precisão permanece estável até à primeira variação RSS significativa, ou seja, quando o veículo

se aproxima do AP e passa pela rotunda. O desempenho dos algoritmos é apresentado na tabela H.1 para duas regiões, RMSE1 e RMSE2. RMSE1 corresponde às primeiras 50 amostras para 20 km/h e às 38 primeiras a 30 km/h, ou seja, o "ponto de ruptura" foi a posição em que o veículo atinge o ponto mais distante do AP; RMSE2 contém as amostras restantes, mas usa todas as amostras anteriores, incluindo RMSE1. Pode ver-se que o método simples e o Gauss-Newton tem a pior precisão do RMSE2, enquanto o algoritmo Tomic tende a ter a melhor precisão, como esperado. O algoritmo com maior complexidade computacional foi capaz de compensar parte do erro de localização do veículo, alcançando uma precisão de 4,18 m, muito abaixo da precisão do A-GPS a 20 km/h. Por outro lado, o desempenho do algoritmo Levenberg-Marquardt foi o melhor dentro dos estimadores fechados, com uma precisão de 6,58 m, também abaixo da precisão do A-GPS, superando o algoritmo de Tomic para RMSE2 no cenário de 30 km/h, onde os erros das medições RSS e da localização do veículo foram muito maiores. Vale a pena salientar que os estimadores de forma fechada foram executados com valores γ e P_T predefinidos, que introduzem erros adicionais, uma vez que o γ tende a variar durante o percurso. Assim, estes estão menos preparados para cenários diferenciados e/ou cooperativos, onde os valores P_T e γ considerados podem não ser otimizados.

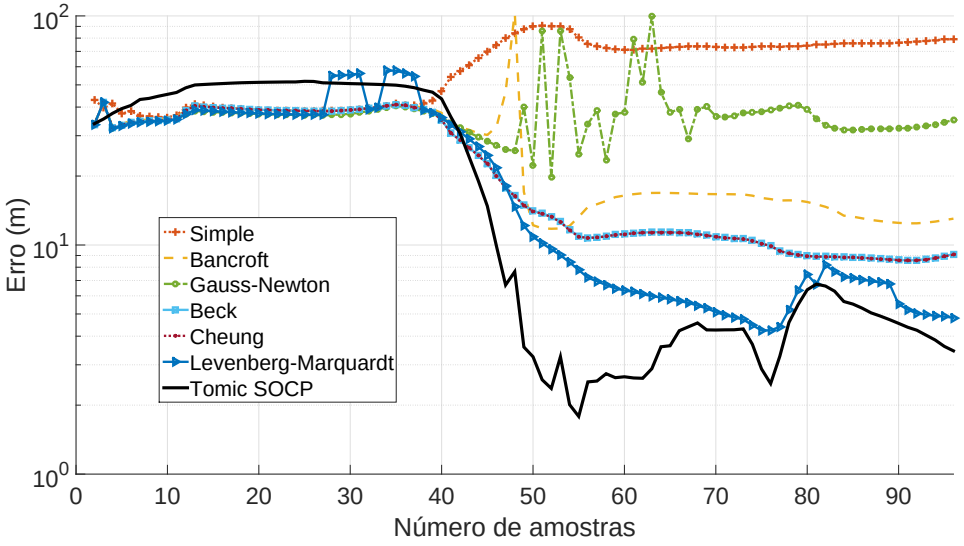


Figura H.3: Precisão de localização a 20 km/h.

A tabela H.1 também apresenta o tempo de cálculo médio utilizado pelos algoritmos em MatLab com um vetor RSS com 30 amostras. Para executar a aplicação em tempo real, mitigar efeitos de leituras antigos e estimar o processo sem influenciar a fluidez da aplicação, decidimos limitar as medidas de RSS armazenadas no protótipo Android não cooperativo a 30. Pode-se ver que Levenberg-Marquardt é o segundo algoritmo mais pesado computacional na implementação da MatLab. Além disso, usando a biblioteca *org.apache.commons.math3* e considerando o tamanho máximo de vetor de 30 amostras, conseguimos medir um tempo de execução médio de 13,47 ms no smartphone. Portanto,

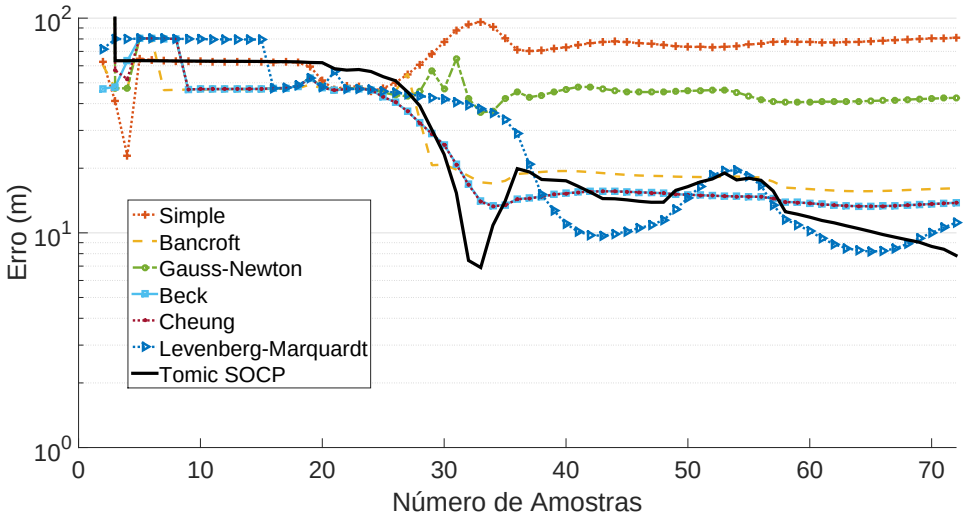


Figura H.4: Precisão de localização a 30 km/h.

Tabela H.1: Comparação do erro médio dos diversos algoritmos

Algoritmo	**Tempo de execução (ms)**	**20 km/h**		**30 km/h**	
		RMSE1 (m)	RMSE2 (m)	RMSE1 (m)	RMSE2 (m)
Simple Intersection	8.66	47.76	77.16	65.35	77.64
Bancroft	12.31	40.61	14.95	44.83	17.72
Gauss-Newton	26.5	36.51	43.42	52.53	44.33
Beck	26.8	36.25	10.5	47.56	14.7
Cheung	15.2	36.26	10.48	47.46	14.68
Levenberg-Marquardt	249	39.44	6.58	60.51	12.47
Tomic SOCP	$3{,}91\times10^4$	44.83	4.18	53.24	14.26

devido ao excelente trade-off entre precisão e tempo de execução, esse algoritmo foi escolhido para implementação de aplicativos Android para localização não cooperativa.