André Filipe Pêgas Grilo

Licenciado em Ciências da Engenharia
Eletrotécnica e de Computadores

Analysis of AGV indoor tracking supported by IMU sensors in intra-logistics process in automotive industry

Dissertação para obtenção do Grau de Mestre em

Engenharia Eletrotécnica e de Computadores

Orientador: Professor Doutor Ricardo Luís Rosa Jardim Gonçalves, Professor Catedrático,
Universidade NOVA de Lisboa

Co-orientador: Ruben Duarte Dias da Costa, Investigador, FCT-UNL

Júri

Presidente: Doutor José Manuel Matos Ribeiro da Fonseca - FCT/UNL
Arguente: Doutor André Dionísio Bettencourt da Silva Rocha - FCT/UNL

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Julho, 2021

Analysis of AGV indoor tracking supported by IMU sensors in intra-logistics process in automotive industry

Este documento foi gerado utilizando o processador (pdf)LaTeX, com base no template "novathesis" [1] desenvolvido no Dep. Informática da FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

To my family

Agradecimentos

Gostaria de agradecer, em primeiro lugar, ao meu orientador, o professor Ricardo Jardim Gonçalves, por me fornecer a oportunidade de poder desenvolver esta tese.

Queria agradecer também aos meus dois co-orientadores, Ruben Costa e Paulo Figueiras, por me manterem no caminho correcto, dando me orientação sempre que encontrava dificuldades que abrandavam o desenvolvimento desta tese.

Agradecer à Faculdade de Ciências e Tecnologias, que correspondeu a um novo capítulo na minha vida. Aos meus amigos e colegas de curso que me acompanharam nesta nova etapa, estando presentes durante os bons e maus momentos.

E finalmente queria agradecer à minha família, e um agradecimento muito especial à minha mãe e à minha irmã, que me acompanharam e me puxaram para continuar a ser uma melhor pessoa, ao longo da minha vida.

Resumo

A indústria 4.0 consiste numa nova revolução que está a introduzir uma mudança de paradigma na indústria. A automação, a descentralização e a modulação são conceitos que se estão a tornar significativamente mais relevantes, originando uma transformação na indústria. Nomeadamente, dentro da indústria automóvel, o incremento da utilização de *Automated Guided Vehicles* (AGVs) é essencial para que haja uma maior eficiência nos processos intra-logísticos, uma vez que eles permitem a automação no transporte de materiais essenciais para esse processo, criando assim uma maior qualidade de serviço.

No entanto, alguns AGVs não possuem capacidades sensoriais que sejam capazes de fornecer dados, em tempo-real, referentes ao estado dos mesmos, impossibilitando assim a aplicação do conceito de indústria 4.0, isto é, uma fábrica conectada, nestas fases. Apesar de certos AGVs não possuírem estas capacidades, que permitem uma monotorização constante dos mesmos, como a sua localização, é necessário arranjar soluções que possibilitam então adquirir estes mesmo dados para efeitos de monotorização, sem que seja necessário investir em AGVs mais avançados.

A solução proposta no âmbito desta tese, tem como objectivo, é uma contribuição para o desenvolvimento de um gestor de frota de AGVs, onde será possível acompanhar todo o processo referente ao transporte de baterias entre a zona de sequenciação, onde as baterias são ordenadas e catalogadas, e a linha de montagem, chamada de *point-of-fit*, desde o fornecimento de uma análise quantitativa e qualitativa de todo o processo, bem como a detecção de falhas/anomalias que possam ocorrer durante o mesmo. Esta solução irá trazer uma maior capacidade de optimização e eficiência para todo o processo, melhorando os aspectos referentes à produção automóvel. Todo este trabalho insere-se no âmbito do projeto Europeu BOOST 4.0, e que foi validado na Volkswagen Autoeuropa.

Palavras-chave: AGV, localização *indoor*, gestor de frota, indústria 4.0, IMU

Abstract

The Industry 4.0 is a new revolution that is introducing a paradigm shift in the industry. Automation, decentralization and modulation are concepts that are becoming significantly more relevant, leading to a transformation in the industry. Namely, within the motor industry, the increase in the use of Automated Guided Vehicles (AGVs) is essential for there to be a bigger efficiency in intralogistics processes, since they allow automation in the transport of essential materials for this process, providing a better quality of service. However, some AGVs do not have sensory capabilities that are capable of providing data, in real time, regarding their status, thus making it impossible to apply the concept of industry 4.0, meaning, a connected factory, in these phases. Even though these AGVs do not possess said capabilities, which enable a constant monitoring, like their positioning, it is necessary to find solutions that would allow the data from them to be acquired for the monitoring, without being necessary an investment in more advanced AGVs.

The proposed solution under this thesis, as an objective, is a contribution to the development of an AGV fleet manager, where it will be possible to monitor the entire process regarding the transport of batteries, between the sequencing zone, where the batteries are arranged and catalogued, and the assembly line, called point-of-fit, from providing a quantitative and qualitative analysis of the entire process, as well as the detection of failures / anomalies that may occur during the same. This solution will bring a greater capacity for optimization and efficiency for the entire process, improving aspects related to automotive production.

All this work is part of the European project BOOST 4.0, which was validated at Volkswagen Autoeuropa.

Keywords: AGV, Indoor localization, fleet management, industry 4.0, IMU

Índice

Lista de Figuras

Lista de Tabelas

Siglas

AGV *Automated guided vehicle (Veículo guiado automaticamente)*

EKF *Extended Kalman Filter* (Filtro de Kalman estendido)

GWO *Grey Wolf Optimizer* (Optimização Lobo Cinzento)

HF *High Frequency* (Altas frequências)

IEKF *Invariant Extended Kalman Filter* (Filtro de Kalman estendido invariante)

IMU *Inertial Measurement Unit* (Unidade de medição inercial)

LF *Low Frequency* (Baixas frequências)

RFID *Radio Frequency Identification* (Identificação por Radiofrequência)

RSSI *Received Signal Strength Indicator* (Indicador de intensidade do sinal recebido)

UHF *Ultra High Frequency* (Ultra Altas Frequências)

UKF *Unscented Kalman Filter* (Filtro de Kalman unscented)

Capítulo 1

Introdução

O presente capítulo serve para fazer uma contextualização do tema a ser apresentado neste documento. Este capítulo está dividido em quatro secções, onde a primeira corresponde a apresentação do plano geral onde o tema se enquandra, sendo neste caso a indústria 4.0. A segunda secção corresponde apresentação do caso de estudo, onde é apresentado a situação onde o problema se enquadra, sendo esta um processo logístico de montagem de baterias da fábrica de automóveis Volkswagen Autoeuropa. A secção 1.3 contém a descrição do problema que será o ponto de trabalho deste documento. A última secção faz uma breve apresentação de uma solução possível para o problema apresentado.

1.1 Motivação

Desde que a indústria foi introduzida pela primeira vez na sociedade, aproximadamente em 1760, como uma forma de melhorar a manufatura existente, introduzindo novos métodos de produção que permitiriam um aumento substancial da produção e consequentemente um maior crescimento económico, esta esteve sujeita a evoluções. Esta evoluções são referidas como revoluções industriais. Estas revoluções, presentes na imagem 1.1, vieram aumentar a capacidade de produção industrial, através da mecanização e da introdução de maquinaria a vapor na primeira revolução, da criação de linhas de montagem na segunda revolução e da automação da produção através do uso de robôs na terceira revolução. Atualmente está a acontecer a quarta revolução industrial, onde o objetivo, como todas as outras, é o aumento da produção, aumentando a eficácia da mesma, através da criação de *Smart Factories*, que seriam capazes de se gerir a elas próprias, criando mais automação, e também a utilização de *machine learning* para fornecer a maior eficácia possível em todos os níveis logísticos, tais como os processos de transporte, de produção e armazenamento [1]. Esta nova revolução é chamada a industria 4.0 devido a ser a

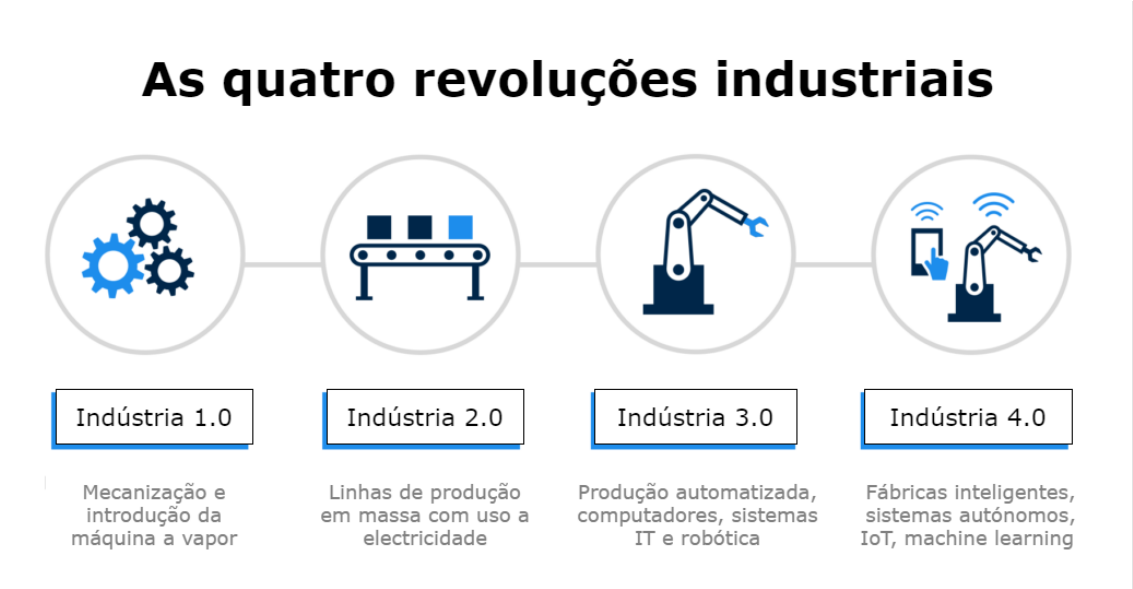


Figura 1.1: Evolução da industria ao longo das várias revoluções

quarta revolução, como foi referido, fazendo uma comparação à nomenclatura utilizada nos programas de computador.

Esta revolução está a causar uma mudança de paradigma correspondente à produção, introduzindo novos conceitos na indústria que não existiam previamente. Esses conceitos são a automação da produção, permitindo assim que esta seja executada e controlada somente por máquinas, sem a intervenção humana, aumentando a eficiência em todo o processo de produção; a descentralização, que tem como objetivo a redução das componentes hierárquicas do processo de produção, tornando assim a tomada de decisões e/ou intervenções mais rápida; a modelação, que facilitará a alteração ou a atualização dos processos já existentes. Para além disso, outros dois conceitos são o conceito de "*smart manufacturing*", onde todo o processo de produção está presente no próprio produto, como informação disponível, desde todas as tarefas já executadas, até às necessárias, para que este chegue ao estado de produto final [2], e o conceito de "*digital twin*"[3], que corresponde à digitalização da planta da fábrica, nomeadamente dos processos de produção, disponibilizando todos os dados numa plataforma centralizada, facilitando assim a visualização dos mesmos. No entanto, o objetivo desta tese não envolve o desenvolvimento de um "*digital twin*"nem de "*smart manufacturing*", e que o foco será um dos sub objetivos do projeto BOOST 4.0 [4], que será descrito em detalhe na secção 1.2. Todos estes novos conceitos irão causar um aumento na eficiência da produção, bem como a sua flexibilidade e customização, permitindo assim maiores lucros, reduzindo ao mesmo tempo os custos de manutenção.

1.2 Caso de Estudo

O caso de estudo que corresponde ao tema desta tese está insirido no projeto BOOST 4.0 [5], onde o objetivo principal é a implementação de técnicas/*standarts*, tais como:

- Uso de uma arquitetura *Reference Architectural Model Industry 4.0 (RAMI 4.0)* [6], para a criação de um padrão global
- Infraestruturas digitais seguras, fazendo uso de tecnologias como a *cloud*, de modo a garantir um desempenho superior do espaço industrial Europeu
- *Middleware* de *Big Data* confiável, fazendo uso de 4 iniciativas *open source* europeias (*(Industrial Data Space*, *FIWARE*, *Hyperledger*, *Big Data Europe)*), de modo a suportar o desenvolvimento de *open connectors* e *big data middleware* com suporte a um *blockchain* nativo
- Plataformas de manufactura digital, permitindo o uso de serviços de análises mais avançadas, bem como a visualização dos dados disponíveis
- Certificação europeia do equipamento, infraestruturas e serviços de *Big Data*

Dentro do consórcio que corresponde ao projeto BOOST 4.0, uma das empresas parceiras é a Volkswagen. Esta empresa possui uma fábrica que foi utilizada como cenário para a aplicação dos objetivos deste projeto, criando assim uma revolução inicial que se aproxima daquilo que corresponde a uma indústria 4.0.

A Volkswagen Autoeuropa, que pertence ao grupo Volkswagen, corresponde a uma industria de manufactura automóvel localizada em Portugal (Palmela), sendo que ela própria é responsável por todo o processo de construção automóvel, desde a prensagem, construção de carroçarias, pintura até à linha de montagem onde se obtém o produto final. A Volkswagen Autoeuropa produziu o ano passado 890 viaturas diariamente[7] , conseguindo assim chegar a um total de 254,600 mil unidades durante o ano de 2019 [8]. No entanto, apesar destes valores positivos obtidos no ano de 2019, muitos aspetos relacionados com os processos de fabrico não estão optimizados e/ou requerem muita intervenção humana que pode causar atrasos durante a produção.

Uma parte desta produção corresponde à montagem de baterias nos automóveis que possui várias fases até chegar à linha de montagem, referida como "*point-of-fit*". O procedimento logístico atual corresponde a:

- Descarregamento: As baterias são descarregadas e verificadas usando protocolos para verificar a integridade da carga (procedimento manual).
- Armazenagem: As baterias são armazenadas no armazém ou redirecionadas diretamente para a linha de montagem.

Figura 1.2: Processo logístico atual referente ao fluxo das baterias, desde que ocorre a entrega até ao transporte para a linha de montagem (*point-of-fit)* (Retirado de [4])

- Transporte (para sequenciação): As baterias são transportadas, usando um rebocador, desde o armazém até à zona de sequenciação.
- Sequenciação: Um operador coloca as baterias por uma sequencia previamente definida (procedimento manual):
- Transporte (para "*point-of-fit*"): Os contentores com as baterias sequenciadas são transportados para a linha de montagem através de AGVs (*Automated Guided Vehicles*).

Todas as fases descritas acima apresentam desafios ligados aos procedimentos, nomeadamente, a necessidade de uma intervenção manual ao longo de grande parte do mesmo. Outro desafio corresponde aos transportes que, atualmente, não possuem capacidade de fornecer dados durante a sua execução, o que impossibilita o controlo dos mesmos. A figura 1.2 corresponde uma representação gráfica de todo o processo intralogístico referido acima (processo atual). É possível observar que em todo este processo existe algum tipo de intervenção humana, como por exemplo na receção das baterias no armazém da fábrica, onde é necessário que estas sejam descarregadas manualmente, ou também no transporte entre a sequenciação e o *point of fit*, onde apesar do transporte ser feito por AGVs, estes são bastante dependentes da intervenção humana para a execução do transporte (ativação manual). Estes são só alguns dos exemplos presentes na intralogística atual, o que introduz atrasos e por conseguinte, diminui a eficácia da produção. Também é possível observar a falta de informação que existe durante os transportes, impossibilitando a supervisão dessas tarefas.

Tendo em conta que estes são apenas alguns dos desafios presentes durante todo o processo, e que todas elas causam atrasos na produção, impossibilitando assim que esta esteja optimizada, a Volkswagen Autoeuropa decidiu que havia uma necessidade de implementar o conceito de indústria 4.0 nas suas instalações, de modo a criar um ambiente produtivo superior ao atual. Para tal, é necessário que haja uma adaptação do

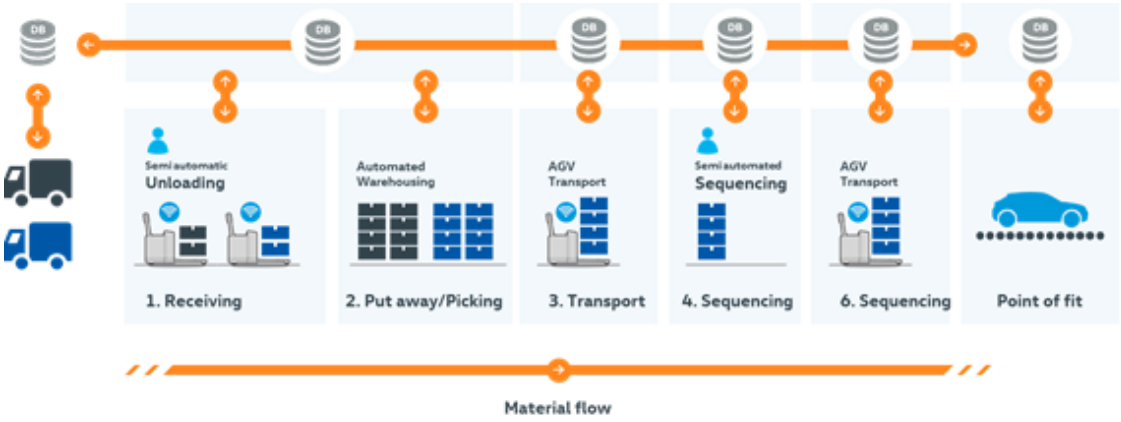


Figura 1.3: Processo logístico a ser implementado referente ao fluxo das baterias desde que ocorre a entrega até ao transporte para a linha de montagem (*point-of-fit)*. Todo este processo, em relação ao processo representado na figura 1.2, apresenta uma automação que está presente em todos os pontos do processo (Retirado de [4])

equipamento presente, como os seus AGVS que têm uma capacidade sensorial muito limitada, de modo a que o processo possa ser todo ele autónomo, criando assim uma fábrica *smart*.

O objetivo da Volkswagen Autoeuropa é então automatizar todo o processo referente ao trannsporte de baterias, desde que estas chegam ao armazém da fábrica, até que estas sejam enntreges no "*point of fit*", ponto este onde estas são montadas nos veículos que estejam na linha de montagem. Para além da automatização, este deverá ter disponível todos os dados referentes a todas as etapas deste processo, de modo a facilitar a monitorização. O esquema descrito está presente na imagem 1.3, onde é possível observar uma diminuição da intervenção humana, como por exemplo, no transporte de baterias, bem como a disponibilidade de todos os dados presentes numa base de dados. É importante salientar que o objetivo desta tese não é o desenvolvimento de todo este sistema apresentado, e sim proporcionar um desenvolvimento incial sobre uma das etapas deste mesmo processo, nomeadamente do transporte entre a sequenciação e o "*point of fit*", sendo que os detalhes dos desafios desta etapa serão discutidos na secção 1.3.

1.3 Descrição do problema

Como é possível aferir com base na secção anterior (1.2), o trabalho desta tese está enquadrado no processo logístico da montagem de baterias nos veículos. Todo este processo, previamente descrito (figura 1.2), possui várias etapas, onde cada uma apresenta vários desafios. O ponto de interesse corresponde à etapa de transporte entre a sequenciação, e o "*point-of-fit*", onde são utilizados AGVs. Estes AGVs não têm a capacidade de calcular nem fornecer informação sobre o seu estado atual, como a sua localização, o seu estado operacional, isto é, se o AGV se encontra com uma avaria ou com falta de bateria. Para além das incapacidades de fornecer qualquer tipo de informação, este tipo de AGVs não

tem a capacidade de detectar e/ou contornar obstáculos que possam aparecer durante o tranporte, e sempre que ocorra uma paragem, quer por ter chegado aos postos de trabalho (sequenciação e "*point-of-fit*"), quer por ser outra razão de vido a uma anomalia durante o processo de transporte, este só volta a retormar o seu estado de trânsito normal com intervenção humana. Outro desafio, que está relacionado com todos os outros mencionados anteriormente, é a inexistência de um sistema de gestão de frota para estes AGVs, onde este estaria encarregue de toda a gestão referente às rotas que os AGVs teriam de percorrer, bem como a capacidade de diaagnóstico em caso de anomalias, como a falta de bateria do AGV ou avaria.

Em suma, os desafios que estão presentes durante o processo de transporte entre a a sequenciação, e o "*point-of-fit*"correspondem a:

- Falta de um sistema centralizado de gestão de frota para este tipo de AGVs
- Localização deste tipo de AGVs, que não utilizam formas de localização direta (e.g. GPS)
- Notificações em caso de anomalias, tais como falta de bateria do AGV ou avarias
- Incorporação da capacidade de deteção e de contornar obstáculos
- Eliminar a necessidade da intervenção humana em todo o processo de transporte destes AGVs

Todos estes desafios estão presentes no documento referente ao projeto BOOST 4.0 [4], e são válidos para a criação de um procedimento mais eficiente relacionado com a etapa de transporte. No entanto, o objetivo relacionado com o desenvolvimento desta tese não corresponde à implementação de uma solução para estes desafios, mas sim iniciar o desenvolvimento referente a um dos pontos acima apresentados, nomeadamente o ponto relacionado com a localização destes AGVs.

1.4 Solução proposta

Os problemas/desafios apresentado na secção anterior (1.3), uma solução possível seria a implementação se um sistema de gestão de frota, capaz de monotorizar todos os AGVs que estejam ligados ao processo de transporte entre a sequenciação e o "*point-of-fit*". Este gestor de frota estaria encarregado de todo o processo de transporte, desde a escolha do método óptimo de entrega, à detecção e reconhecimento do caminho dos outros robôs, através do "*tracking*"de AGVs, evitando colisões, bem como a deteção de falhas/avarias nos mesmos. Apesar desta ser uma provável solução para o problema referido, o objetivo desta dissertação estará enquadrado principalmente no "*tracking*"de AGVs, através do uso de IMUs (*Inertial Measurement Unit*) e da implementação de um algoritmo capaz de efetuar esse mesmo "*tracking*", abordagem esta que contribui para o desenvolvimento do

tal sistema de gestão de frotas, sem a qual, tal sistema não seria possível. A razão do uso de IMUs nesta implementação deve-se ao facto de estes serem um tipo de tecnologia que funciona num ambiente "*indoors*", que corresponde a um ambiente numa planta de uma fábrica, ao contrário do GPS que só proporciona um bom tracking num cenário "*outdoors*", é um tipo de tecnologia não intrusiva, isto é, a sua implementação não afeta os processos logísticos que já estejam presentes na fábrica, onde, em contrapartida, a utilização de RFIDs envolveria a implementação destes dispositivos na planta da fábrica. Para além destas duas razões apresentadas a favor da escolha de IMUs como tecnologia escolhida, estes ainda possuem custos baixos, sendo mais um argumento a favor da escolha da mesma.

Este algoritmo envolverá o uso das equações de movimento de Newton, de modo a obter o deslocamento através da aceleração medida pelos IMUs, bem como a implementação de filtros, que permitirão uma eliminação de grande parte do ruído que possa existir no processo de obtenção de dados e de execução do algoritmo. Para que o objetivo seja exequível, é necessário que este seja decomposto em vários pontos, que representam as várias etapas, sendo elas:

1. Análise do estado da arte de sistemas de localização *indoor*, bem como sistemas de gestão de frota que já tenham sido aplicados na área da indústria de manufactura.

2. Estudo da adequabilidade de cada uma das técnicas usadas para o caso de estudo específico.

3. Especificação e desenvolvimento de um sistema de localização de baixo custo para ser implementado em AGVs.

4. Implementação do sistema de localização referido no ponto anterior.

5. Implementação de filtros capazes de reduzir o máximo ruído provenientes da obtenção dos dados provenientes do IMU.

6. Desenvolvimento de uma solução que permita, em tempo-real, a localização de toda a frota de AGVs, bem como uma monitorização dos mesmos (detecção de eventos que não correspondam ao processo normal, tais como avarias).

Pergunta de Investigação: É possível calcular a localização atual de um AGV num ambiente fechado, sem o auxílio de técnicas de localização normais, como a tecnologia GPS?

Hipótese de Investigação: Se, através do uso de dados incerciais fornecidos pelo IMU, utilizando técnicas de processamento de dados, juntamente com o uso de equações do movimento, então é possível efetuar o cálculo da posição dos AGVs.

1.5 Estrutura do documento

A organização deste documento está dividida em 4 capítulos. O capítulo 1 corresponde à introdução, onde foi apresentado a motivação, caso de estudo, descrição do problema e solução proposta. O capítulo 2, que corresponde ao estado da arte, contém informação sobre vários trabalhos que foram desenvolvidos e que tenham um cenário idêntico ao cenário apresentado nesta tese, isto é, um caso onde a localização de AGVs num espaço indoor seja efetuado através do uso de sensores, tais como IMUs ou RFIDs. Este capítulo está dividido em duas secções, sendo elas a secção 2.1, onde são apresentados métodos de localização através do uso de RFIDs, IMUs, ou através do uso da combinação de vários tipos de sensores (sistemas híbridos), e a secção 2.2, onde é introduzido alguns exemplos do desenvolvimento de gestores de frota para AGVs. Os dois últimos capítulos correspondem ao capítulo 3 onde é apresentado o método de desenvolvimento do sistema de localização implementado, e o capítulo 4, onde estão presentes os resultados do sistema apresentado no capítulo anterior.

Capítulo 2

Estado da Arte

O presente capítulo serve para apresentar o estudo feito sobre o trabalho relacionado com o problema apresentado no capítulo 1.

A secção 2.1 apresenta um conjunto de métodos de localização interior para objetos móveis, nomeadamente AGVs, fazendo referência também aos sensores necessários e/ou técnicas usadas. Casos de estudo que estejam enquadrados com a área industrial serão considerados. A secção 2.2 contém um conjunto de soluções relativas a gestores de frota, que inclui técnicas de agendamento de tarefas. A secção 2.2.3 apresenta soluções referentes à detecção de falhas/anomalias que possam acontecer durante as tarefas dos AGVs.

2.1 Métodos de localização interior

Atualmente existem várias soluções apresentadas na temática da robotização, devido aos vários tipos de aplicação, no entanto, a base tecnológica dessas soluções pode ser dividida em três grupos. Segundo [9], os três tipos de tecnologia utilizada no contexto de localização são o uso das propriedades das ondas magnéticas, tais como a fase e o ângulo, o uso de visão computacional, que envolve a comparação de imagens com contenham pontos de interesse e/ou a implementação de técnicas de visão computorizada como o SLAM, e a detecção de movimento, feita atrevés do uso de acelerómetros, giroscópios e magnetómetros, que permitem estimar a direção do movimento do robô e da sua orientação. Para além desta distinção feita nas tecnologias, ainda existe uma divisão em dois grupos nos métodos existentes. O primeiro corresponde a medidadas de posição relativa, como o uso da odometria e da navegação inercial, e o segundo corresponde a medidas de posição absoluta, que contêm métodos como o GPS (Global Positioning Systems) [10].

Dependendo do caso em especifíco, os métodos e as tecnologias a serem utilizados podem variar, devido às diferentes características que eles apresentam. No caso industrial, mais concretamente, onde o AGV terá de ser capaz de navegar num ambiente bastante movimentado e que ele necessitará de evitar obstáculos, como seres humanos ou outros AGV's, é necessário que a precisão seja elevada para uma *room-level accuracy*. A figura 2.1 apresenta algumas tecnologias para localização em ambientes fechados, onde relaciona a sua exatidão com a característica de campo de visão, isto é, se o receptor (presente no AGV) necessita de estar dentro do campo de visão do transmissor ou não.

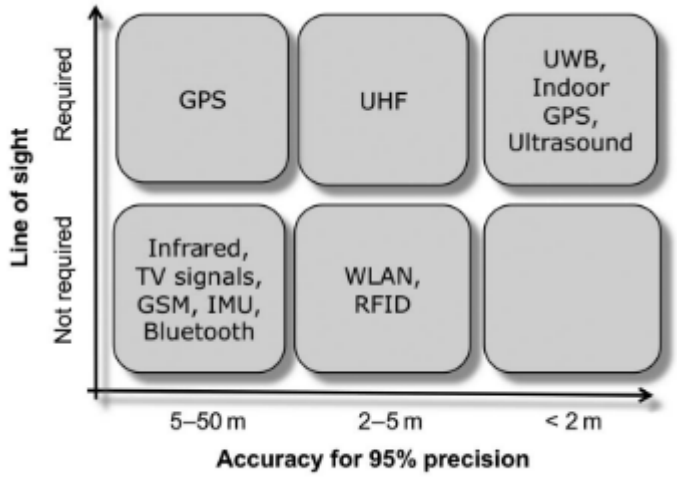


Figura 2.1: Tecnologias para localização interna. (Retirado de [9])

Apesar da sua alta exatidão para uma precisão a 95%, a necessecidade que o receptor esteja dentro do campo de visão dos satélites, faz com que as medições se tornem fracas quando aplicadas a um ambiente fechado. Isto pode ser comprovado através do estudo feito po Kjaergaard et al. [11], onde o uso de receptores de GPS com tecnologia mais recente, foi usado em vários ambientes fechados, nomeadamente um caso particular que foi num armazém. Os resultados obtidos, em relação ao erro médio correspondente à posição, foram em média 8.8m, que, tendo em conta o caso de estudo desta tese, não é um valor de erro aceitável. Deste modo, a solução a implementar terá que usar tecnologias que não necessitem do campo de visão, tais como RFID, que correspondem a tecnologias que usam as propriedades das ondas eletromagnéticas, como referido anteriormente, ou o uso de IMU's, que correspondem a tecnologias de detecção de movimento. O estudo destas duas tecnologias será então aprofundado nas próximas duas secções, onde será referido casos de estudo que estejam enquadrados com a situação do problema. A subsecção 2.1.1 irá apresentar soluções para a localização interior através do uso da tecnologia RFID, e a subsecção 2.1.2 será sobre soluções que usem o método de *deadreckoning*, nomeadamente através do uso de IMUs.

2.1.1 RFID

O RFID (Radio Frequency Identification) é um tipo de tecnologia que usa como método medidas de posição absoluta, nomeadamente, o uso de pontos de referência artificias na forma de transponders. Estes podem ser de natureza ativa, i. e., possuem uma bateria incorporada no próprio transporder, ou tag, tendo um tempo de funcionamento igual ao tempo de vida da bateria, ou então podem ser tags com natureza passiva, sendo que esta já não possui bateria e recebe a energia necessária para funcionar do próprio leitor. Este segundo tipo de tags possui um tempo de vida inversamente proporcional ao número de leituras feitas, no entanto, devido ao seu custo reduzido e grande portabilidade, o uso destas tags como solução de localização de objetos tem vindo a aumentar. A localização através de RFID usa a técnica de RSSI (Received Signal Strength Indication) [9] como forma de obter a distância ao objeto, também conhecida como *lateration*. Em [12] é referido uma comparação entre o uso de tags ativas e passivas que, como já mencionado anteriormente, a segunda possui um custo muito inferior à primeira. Também é referido a importância da escolha da frequência de trabalho para o qual o RFID irá funcionar, onde tags de UHF (*Ultra-High Frequencies*) são preferidas em relação a tags de LF (*Low Frequencies*) e de HF (*High Frequencies*) devido ao seu alcance superior. No entanto, tags de HF também são uma boa escolha em relação as UHF devido ao seu custo de fabrico inferior e maior resistência. Uma solução que usa este tipo de tags RFID é apresentada por Roehrig et al. [13], que utiliza tags passivos de HF, embebidos numa fibra de vidro reforçada (NaviFloor®). O uso do NaviFloor® facilita a instalação dos transponders necessários, visto que a tecnologia RFID necessita de uma infraestrutura previamente montada no local. Esta solução utiliza os transponders como detectores de presença, isto é, sabendo as posições fixas das tags, quando o leitor ,que se encontra no objeto, é capas de detectar uma tag, a posição dessa mesma tag é associada ao mesmo como medida de localização. Isto deve-se ao facto de esta solução admitir que a probabilidade de detectar um transponder é binária, isto é, será 1 quando o leitor se encontra dentro da área de detecção, caso contrário será 0. Caso haja a detecção de mais do que 1 transponder, é feita a intersecção das áreas de todos os transponders detectados. A orientação do objeto é dada pela orientação da antena do leitor. Para o movimento associado à deslocação de um transponder para outro é usado odometria [10] como método de localização relativa que, através do uso do Filtro de Kalman estendido [14], é capaz de fornecer informação sobre o movimento que vai acontecendo ao longo do tempo.

2.1.2 *Dead-reckoning*

O método de *dead-reckoning* corresponde a um método de localização relativa onde é usada a posição anterior para calcular a atual, com o auxílio de informação obtida por sensores, como as velocidades (angulares e lineares). Este método é bastante usado como auxílio aos métodos de posição absoluta, como o GPS, em situações onde estes possam falhar tais como um carro que esteja a atravessar um túnel, onde o sinal poderá estar

indisponível o que impossibilita a sua localização. Por esta razão, o uso do método de *dead-reckoning* na navegação automóvel tem vindo a aumentar, principalmente em casos de localização interior, onde o uso de sinais de GPS não é um método viável. Para a implementação deste método é necessário usar um sistema que faça uso de uma unidade de medição inercial (IMU em inglês), isto é, um conjunto de acelerómetros, giroscópios e magnetómetros que medem a força especifica e a rotação. Como este método não necessita de qualquer infraestrutura para ser implementado e, no caso dos IMUs, estes possuem custos bastante reduzidos, o uso de *dead-reckoning* para a localização interior é bastante procurado. No entanto, estes tipos de sistemas são muito suscetíveis a erros, nomeadamente, erros associados ao ruído da leitura dos sensores que vão se acumulando após cada leitura, devido à integração [10], prejudicando assim os dados obtidos durante um uso perlongado [15], como é possível observar na figura 2.2, onde é possível observar que um sistema de IMU puro desvia-se muito cedo do percurso estabelecido. É então necessário implementar um componente que consiga eliminar este ruído que se vai acumulando, de modo a tornar os dados obtidos pelos sensores o mais exatos possíveis. Um componente muito usado na área da automação para combater este problema é o Filtro de Kalman [16]. Este filtro consiste em duas fases onde a primeira fase consiste na previsão das variáveis do estado atual (como a velocidade) com base em modelos conhecidos, tais como o modelo físico. A fase seguinte envolve uma a comparação entre as variáveis estimadas com os dados recebidos dos sensores. Com esta segunda fase, é possível reduzir a quantidade de ruído presente nas medidas obtidas pelos sensores e, como este filtro funciona num modo recursivo, este só necessita da informação das variáveis de estado anteriores para garantir um estado atual o mais exato possível (design-of-a-positioning-system-for-agv-navigation). Usando este filtro, a exatidão das medições irá depender não só da precisão e exatidão dos sensores, mas também dos parâmetros definidos de ruído. Uma descrição mais detalhada da implementação de filtros de Kalman será abordada na secção 2.3, mais precisamente em 2.3.2.

Brossard et al. [15] propõe o o uso de um sistema de *dead-reckoning* com IMU, que utiliza dois módulos. O primeiro módulo corresponde a um *Invariant Extended Kalman Filter* (IEKF), que é uma variante do filtro de Kalman ([17]), e o segundo módulo corresponde a adaptador de parâmetros de ruído que utiliza inteligência artificial, nomeadamente redes neuronais, para definir esses mesmo parâmetros com base nas medidas obtidas pelo IMU. A implementação da rede neuronal é a forma que o autor apresenta de conseguir adaptar os parâmetros relacionados com o ruído referido aos sensores, criando assim uma constante calibração por parte da mesma de modo a obter melhores resultados usando o filtro de Kalman. A tabela 2.1 corresponde aos resultados correspondentes ao método proposto por [15] e a mais outros 3 métodos que servirão para comparação de resultados.

É possível observar que o método que usa o IMU puro possui um erro de translação superior aos restantes e que quanto maior o tempo de execução, maior será esse mesmo erro, devido à constante integração de valores com ruído medidos pelos sensores. Para além disso, comparando os resultádos obtidos pelo método IMLS com o proposto pelo

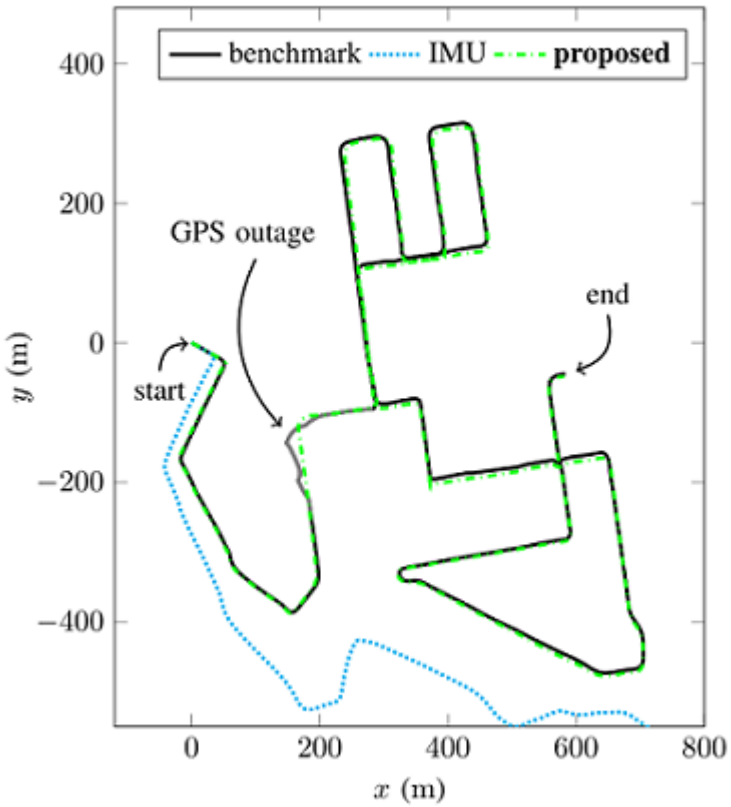


Figura 2.2: Gráfico contendo uma comparação entre localização feita usando um sistema de IMU puro e o proposto por [15] em relação a um caminho predefinido (Retirado de [15])

Tabela 2.1: Resultados de 4 testes distintos de localização onde $t_{rel}(\%)$ corresponde ao erro relativo de translação e o $r_{rel}(\%)$ corresponder ao erro relativo de rotação. O IMLS corresponde a um método retirado do sistema online de *benchmark* da KITTI, que se encontra em terceiro no rank; O ORBSLAM corresponde a um método baseado em câmeras; O IMU corresponde ao método que usa os valores diretamente dos sensores; O *proposed* corresponde ao método proposto por [15] (Retirado de [15])

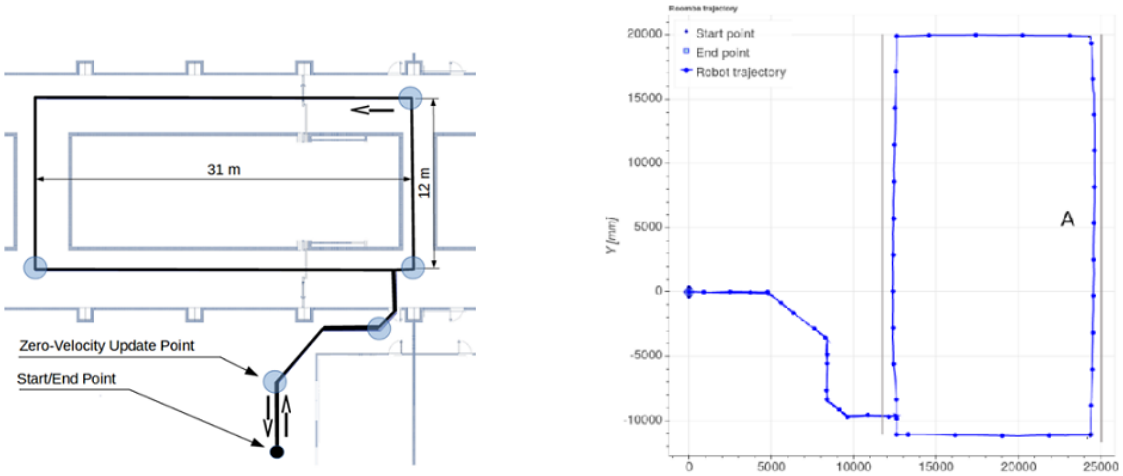
test seq.	length (km)	duration (s)	environment	IMLS		ORB-SLAM		IMU		proposed	
				t_{rel} (%)	r_{rel} (deg/m)	t_{rel} (%)	r_{rel} (deg/m)	t_{rel} (%)	r_{rel} (deg/m)	t_{rel} (%)	r_{rel} (deg/m)
01	2.6	110	highway	**0.82**	**0.10**	(%)	0.19	(%)	0.12	(%)	0.12
03	-	80	country	-	-	-	-	-	-	-	-
04	0.4	27	country	**0.33**	0.12	0.47	0.22	0.97	0.10	1.22	**0.04**
06	1.2	110	urban	**0.33**	**0.08**	0.73	0.22	5.78	0.19	1.57	0.19
07	0.7	110	urban	**0.33**	**0.15**	0.91	0.49	12.6	0.30	1.32	0.30
08	3.2	407	urban, country	**0.80**	**0.18**	1.03	0.30	549	0.56	1.08	0.32
09	1.7	159	urban, country	**0.55**	**0.12**	0.81	0.25	23.4	0.32	0.82	0.22
10	0.9	120	urban, country	**0.53**	0.17	**0.66**	0.31	4.58	0.25	1.05	0.25
average scores				**0.64**	**0.12**	0.99	0.26	171	0.31	0.97	0.23

autor, é observável que o primeiro aparenta ser superior que o segundo. No entanto, o autor refere que o objetivo da representação destes resultados serve apenas para demonstrar que é possível desenvolver um método que use apenas um IMU razoável, para obter resultados comparáveis ao de um método de estado da arte, que possui sensores bastante mais caros.

Este método descrito acima possui uma complexidade de implementação relativamente alta, devido ao uso do módulo de inteligência artificial, para colmatar o uso de um IMU com precisão moderada. No entanto, existem atualmente sensores *smart* que possuem já alguma capacidade de autocalibração, reduzindo assim a necessidade da implementação de métodos mais complexos. O uso deste tipo de sensores é apresentado como uma solução por Cechowicz et al. [18]. Neste caso, a solução utiliza um sensor MEMS (*Microelectromechanical systems*), que faz uso de um giroscópio e de um acelerómetro, e ainda software que, quando aplicados em conjunto, possuem capacidades tais como a capacidade de autocalibração durante o arranque, como referido acima, a capacidade de detetar a rotação do IMU não tendo em conta a sua orientação, a capacidade de detetar momentos de paragem, entre outros (descrição completa das funcionalidades em [18]). Esta última função descrita é usada para aplicar um método que o autor referencia como o *Zero-velocity Update* (ZUPT) [19], que consiste na recalibração do sensor nesses momentos de paragem. Esta implementação usa o sensor MEMS como forma de descobrir a orientação e os estados de paragem, onde o método referido acima possa ser utilizado para recalibrar o mesmo. Para a deteção do movimento é utilizado encoders rotativos para determinar a distância percorrida através da equação $dL = \frac{dX_L + dX_R}{2}$, onde dL corresponde à distância total percorrida, dX_L corresponde à distância percorrida pela roda da esquerda e dX_R corresponde à distância percorrioda pela roda da direita. A tabela 2.2 corresponde aos resultados obtidos em relação ao erro absoluto ao longo de todo o percurso representado na figura 2.3b em relação as coordenadas x e y, e o erro correspondente à distância entre o robô e o ponto inicial do percurso. É importante referir que neste teste não foram considerados erros sistemáticos, tais como a derrapagem, que influenciam negativamente a localização se não forem aplicadas medidas capazes de lidar com este tipo de erros [20].

Tabela 2.2: Resultados da exatidão obtida usando o método descrito em [18] (Retirado de [18])

Test nº	Absolute error (x,y) [m]	Distance error [m]
1	0.16; 0.15	0.22
2	0.05; 0.40	0.40
3	0.01; 0.44	0.44
4	0.9; 0.10	0.13



(a) Percurso de teste do sensor MEMS (b) Percurso do robô usando o sensor MEMS

Figura 2.3: A imagem da direita representa o percurso de teste a ser percorrido pelo robô, estando indicado os locais onde ocore o *Zero-velocity Update*; A imagem da esquerda representa o percurso feito pelo robô (Retirado de [18])

2.1.3 Métodos híbridos

Os métodos de localização apresentados nas duas subsecções anteriores utilizavam um método ou tecnologia predominante, seja ele através de um IMU ou do uso das propriedades das ondas eletromágneticas. Nessas mesmas soluções, a localização era feita utilizando apenas essas tecnologias, ou seja, a localização ou parte da componente da localização, como a orientação, era feita individualmente por essas tecnologias. No entanto, existem soluções que fazem a localização através do uso de várias tecnologias/sensores. Este tipo de solução é apresentado por [21], onde é utilizado odometria, IMU e ainda métodos SLAM que utilizem EKF. As primeiras duas correspondem a tecnologias de localização relativas enquanto que a última corresponde a uma tecnologia de localização absoluta, pois usa o reconhecimento de pontos de interesse como forma de localização. Para juntar a informação proveniente destes 3 tipos de sensores é utilizado EKF. A razão de se utilizar múltiplos sensores para a resolução de problemas de localização é o facto de esta união fazer com que as leituras obtidas de vários sensores sejam muito mais precisas e exatas do que se fossem usadas individualmente [22].

Outro método híbrido que faz uso dos dois tipos de tecnologias mencionadas é apresentado por Ionel Stanculeanu er al. [23], onde a solução faz uso de um sistema de navegação inercial, isto é, de um IMU, e do uso de módulos de ZigBee, que utilizam a técnica de indicador de intensidade do sinal recebido (RSSI). O RSSI é então utilizado para calcular as distâncias dos vários nós de ZigBees, em que são necessários pelo menos 3 nós para obter uma estimativa relativamente exata da posição do veículo. O cálculo da distância é feito utilizando a expressao de potência transferida entre o nó e o receptor [24]

$$P_r(d) = -10n_p \log_{10}(d) + A \tag{2.1}$$

onde se obtém

$$d = 10^{\frac{A-P_r}{10n_p}} \tag{2.2}$$

Por sua vez a localização é feita através da resolução de um sistema de equações sendo ela

$$\begin{bmatrix} 2(x_2-x_1) & 2(y_2-y_1) \\ 2(x_3-x_1) & 2(y_2-y_1) \\ 2(x_4-x_1) & 2(y_2-y_1) \end{bmatrix} \begin{bmatrix} x_p \\ y_p \end{bmatrix} = \begin{bmatrix} d_1^2 - d_2^2 + x_2^2 + y_2^2 - x_1^2 - y_1^2 \\ d_1^2 - d_3^2 + x_3^2 + y_3^2 - x_1^2 - y_1^2 \\ d_1^2 - d_4^2 + x_4^2 + y_4^2 - x_1^2 - y_1^2 \end{bmatrix} \tag{2.3}$$

As variáveis presentes no sistema de equações 2.3 são referentes às variáveis presentes na figura 2.4.

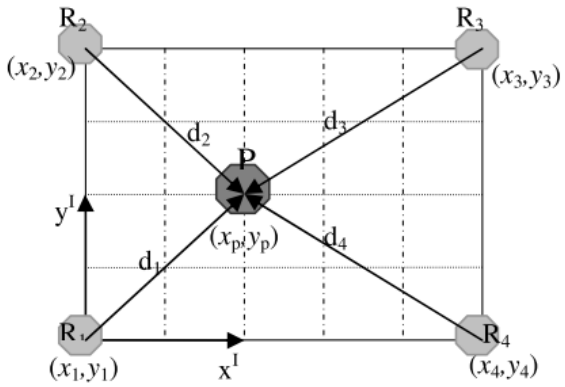


Figura 2.4: Referêncial cartesiano com 4 nós (R1,...,R4) representados. A localização é feita através da aplicação da *lateration*. As disntâncias são então utilizadas para determinar o x_p e o y_p (Retirado de [23])

A fase posterior corresponde então à fusão entre a localização obtida pelo RSSI e os dados recebidos pelo IMU. Esta fusão é feita utilizando a mesma técnica referida na solução de [21], ou seja, é feita através do uso de um filtro de Kalman estendido. A forma de fusão é feita usando um modelo que receba informação dos dois sensores em causa, neste caso a localização em forma de coordenadas x e y, por parte do uso do RSSI, a aceleração linear e a velocidade angular, provenientes do IMU. Todo o processo do filtro de Kalman está representado na figura 2.5. O x_k corresponde ao vetor de estado que é representado por

$$\begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix}_k \tag{2.4}$$

onde x e y correspondem às coordenadas cartesianas, e u e v correspondem as velocidades dentro do referêncial, onde u representa a velocidade em x e v a velocidade em y. O output do sistema z_k é representado por

$$z_k = \begin{bmatrix} x \\ y \end{bmatrix}_{k+1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix}_k \tag{2.5}$$

O A_k e o H_k correspondem às matrizes de transição e de medidas, respetivamente , sendo representados por

$$A_k = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad H_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{2.6}$$

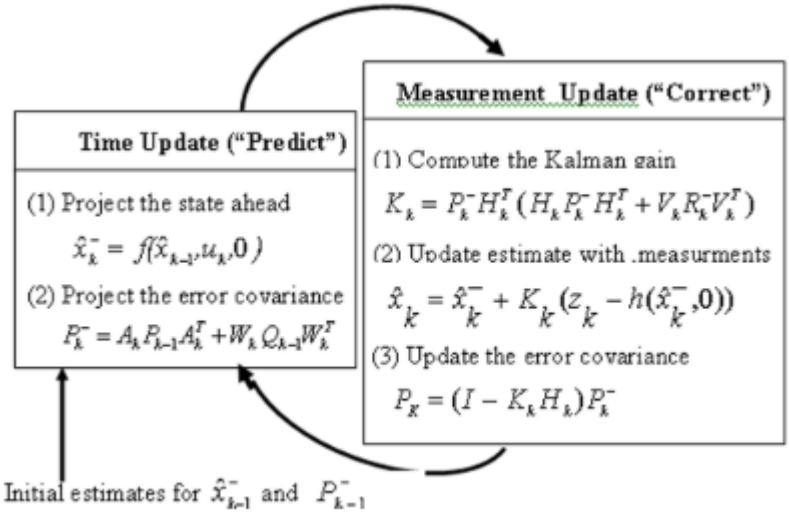
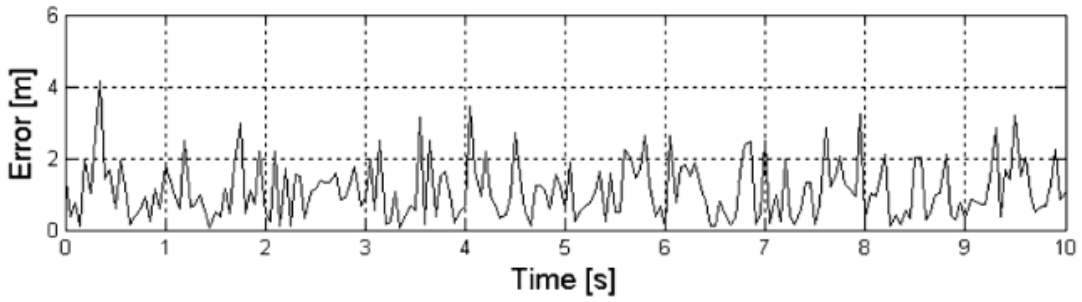


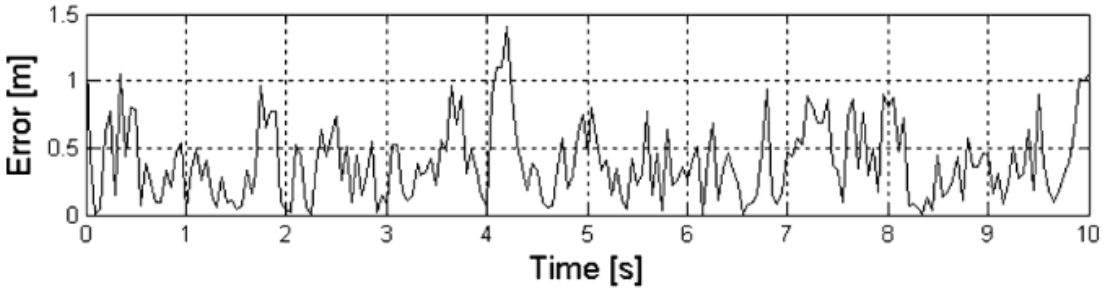
Figura 2.5: Processo referente a um "*Extended Kalman Filter*".(Retirado de [23])

Está presente na figura 2.6 dois gráficos que correspondem aos erros dos resultados experimentais obtidos ao longo do tempo, em que a figura 2.6a corresponde aos erros obtidos utilizando apenas o RSSI como forma de localização e a figura 2.6c corresponde aos erros obtidos utilizando a fusão de sensores através do filtro de Kalman. Como era de esperar, o método que envolve apenas o uso do RSSI apresenta resultados mais fracos, onde a localização fica sujeita a poluição dos dados devido ao ruído. Os resultados referentes à fusão de sensores aparentam ser bastante mais positivos, sendo que o valor máximo de erro obtido aparenta ser inferior que a média do erro correspondente ao método RSSI.

De todos os métodos apresentados nas sub secções 2.1.1, 2.1.2 e 2.1.3, os que aparentam ser mais promissores são as soluções apresentadas por Brossard et al. [15], por Cechowicz et al. [18] e por Hakan Temeltas [21]. A razão deve-se ao facto de estas soluções apresentam não só resultados bastante positivos em ambientes fechados e semelhantes ao de um ambiente fabril, mas também porque estas soluções fazem uso de tecnologias



(a) Gráfico corresponde ao erro dos resultados obtidos usando apenas o módulo de RSSI para a localização



(c) Gráfico corresponte ao erro dos resultados obtidos usando a fusão de sensores, através do módulo de EKF, para a localização

Figura 2.6: Gráficos correspondentes aos erros obtidos pelos testes feitos por Stanculeanu et al. [23]. (Retirado de [23])

baratas, como os IMUs, em contra partida às outras soluções, que utilizam tecnologias como o RFID, que necessita que seja implementada toda uma infraestrutura para que este método seja aplicável, fazendo com que seja uma solução mais cara.

2.2 Métodos de gestão inteligente de frotas de AGV

No capítulo 1 foi referido a mudança de paradigma na indústria da manufactura por parte da indústria 4.0, onde é mencionado a automação do transporte de baterias por parte dos AGVs. Como tal, é necessário implementar uma componente vital que permita esta automação, sendo ela um sistema de gestão de frota de AGVs. Apesar de existirem vários tipo de gestores de frota, estes em muitos dos casos são bastante limitados em relação à capacidade de integrar informação proveniente da fábrica, isto é, a análise em tempo real do estado das máquinas, AGVs em particular, e a conecção desses dados com o respetivo *digital twin* [3]. É então necessário definir as características que o gestor de frota deverá conter e quais os objetivos a cumprir dentro dessas características. Um conjunto detalhado de características e objetivos necessários para um gestor de frota estão presentes em [25]. É ainda apresentado por [26] um estudo feito utilizando *fuzzy set/qualitative comparative analysis* (fsQCA) sobre várias condições que um sistema de gestão de frota poderá conter,

desde factores técnicos a financeiros, e a sua respetiva importância.

2.2.1 Tamanho da frota

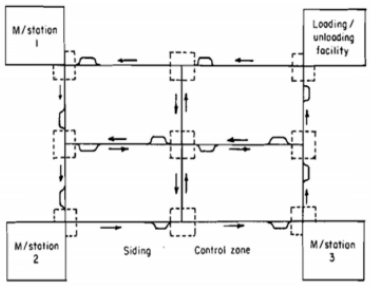
Inicialmente é necessário determinar qual o tamanho de frota óptimo a usar, tendo em conta o tipo de trabalho a ser executado pelos AGVS, de modo a obter o máximo de rendimento possível utilizando o mínino de recursos possíveis. Os factores que podem afetar a decisão da escolha do tamanho da frota variam conforme o autor, sendo que [27] faz referência a:

1. *Layout* do sistema;
2. Localização dos pontos de carga e descarga;
3. Comutação entre postos de trabalho por unidade de tempo;
4. Estratégias de despacho;
5. Fiabilidade do sistema;
6. Velocidade de viagem dos AGVs

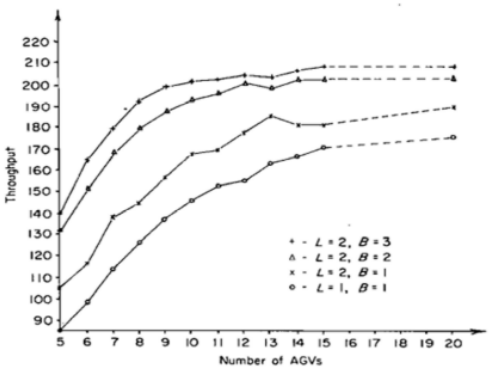
Em relação aos pontos apresentados acima, Chawala et al. [28] só considera os pontos 1, 2 e 4 como factores importantes em relação a estimar o tamanho de frota óptimo.

A abordagem apresentada por [27] faz uso de modelos analíticos como forma de obter uma estimativa para o tamanho da frota necessária, onde é utilizado um rácio do tempo total necessário pelo AGV durante o seu turno e o tempo disponível pelo AGV durante esse mesmo turno, referido como "*empty travel time*". O tempo total necessário pelo AGV durante o seu turno corresponde à soma do tempo de carga, descarga e de "*empty travel time*". É apresentado depois, em relação a este último parâmetro, várias formas de o definir. É apresentado ainda métodos de simulação para validar o tamanho da frota obtido através dos métodos analíticos, sendo que na figura 2.7 é possível observar um tipo de simulação que relaciona o número de AGVs à tava de transferência, utilizando três parâmetros, sendo eles a capacidade do AGV, o número de paletes e o tamanho do buffer que cada estação possui. É possível observar que existe sempre um ponto que, apesar de se continuar a aumentar o número de AGVs disponíveis, a taxa de transferência mantêm-se constante, mostrando assim a importância de estabelecer um tamanho de frota que obtenha o melhores resultados utilizando o menor número de recursos.

Outra simulação apresentada, corresponde também é taxa de transferância em relação ao número de AGVs, mas desta vez é utilizado o tipo de *layout* do local como parâmetro de comparação, isto é, se os caminhos percorridos pelos AGVs são bidirecionais ou unidirecionais. Esta simulaçãol permite não só obter também o número ideal de AGvs para cada situação, mas tabmém permite perceber que tipo de *layout* deve ser usado no espaço correspondente ao cenário de interesse.



(a) *Layout* utilizado no estudo feito por [27]



(b) Resultados obtidos pelo estudo analítico feito por [27]

Figura 2.7: A figura 2.7b representa o resultado correspondente à taxa de transferência média obtida em relação ao número de AGVs, utilizando como parametros a capacidade individual de cada um (L), o tamanho do buffer dos pontos de carga e descarga (B) e o número de paletes (P), tendo em conta o *layout* utilizado 2.7a (retirado de [27])

Um estudo feito por Chawala et al. [28] mostra, no entanto, que o uso de métodos analíticos para estimar o tamanho óptimo da frota pode demonstrar falhas em relação aos requisitos correspondentes a um sistema de tempo real. Este estudo apresenta uma comparação entre um modelo analítico, utilizando como parâmetros a sequência de trabalhos a ser executada, a quantidade de postos de trabalho, o tempo total disponível, o tempo de execução de dentro dos postos, o tempo médio gasto, entre outros, e um algoritmo de optimização de *Grey wolf*. Este algoritmo utiliza como base a teoria por detrás dos métodos de caça e de liderança dos lobos-cinzentos. Proposto por Mirzalili et al. [29], este algoritmo faz uso de 4 "lobos"(funções de fitness) categorizados como alpha, beta, delta e omega, sendo que, a melhor solução irá corresponder ao alpha, a segunda melhor ao beta, a terceira melhor ao delta e a quarta melhor ao omega. O procedimento detalhado do funcionamento deste algoritmo está presente em [28].

A tabela 2.3 apresenta os resultados experimentais correspondentes ao uso de um modelo analítico e o algoritmo de optimização de *Grey wolf* em relação a estimar o número de AGVs necessários para 3 diferentes tipos de *layouts*, sendo que a diferença entre eles é o número de postos de trabalho presentes na planta. O algoritmo de optimização,

em relação aos 3 casos apresentados, apresenta melhores resultados do que o modelo analítico, visto que é utilizado menos recurso para obter o mesmo resultado final.

Tabela 2.3: Resultados obtidos através do uso de um modelo analítico e do algoritmo de optimização *Grey wolf* (GWO), em relação a 3 *layouts* diferentes. (Retirado de [28])

FMS Layout	No. of Jobs	No. of Work centres	No. of Sequences	No. of AGVs	
				Analytical	GWO
1	5	6	2	2	1
2	5	8	2	4	2
3	5	11	2	5	3

2.2.2 Sistemas de gestão de frota

Como referido no inicio de secção 2.2, existem várias abordagens quando se trata no desenvolvimento um sistema de gestão de frota dependendo do caso de aplicação. Um tipo de sistema é apresentado em [30], onde é considerada uma *framework* que possui três níveis, estando cada um associado a um conjunto de tarefas. O nível superior é descrito como o *mission scheduler*, que está encarregue de atribuir as tarefas aos vários AGVs presentes no sistema, detetar *deadlocks*, gerir os AGVs que se encontrem inativos, e também está encarregue de controlar quando o veículo necessita de operações de manutenção. O nível seguinte, correspondente ao nível intermédio, está encarregue de computar qual o caminho que o AGV deverá percorrer para completar a tarefa atribuída, no mínino de tempo possível e sem colisões. Para tal é utilizado o algoritmo de Dijkstra [31], que irá calcular o caminho mais curto possível entre o AGV e o destino final. Depois de ter o caminho definido, é utilizado um outro algoritmo, o coordenador, que ficará engarregue de validar os vários segmentos desse caminho, de modo e evitar colisões. O último nível está associado ao AGV em si, estando encarregue do acompanhamento constante da trajetória do AGV, sendo que sempre que é necessário aplicar uma rotina de emergência em casos de perigo, é este nível que está encarregue de aplicar essas mesmas rotinas.

Esta solução descreve a planta do local como um conjunto de caminhos previamente definidos, chamados de segmentos, e o conjunto de todos os segmentos forma o *roadmap*. É através deste *roadmap* que o nível intermédio decide qual o melhor caminho a percorrer, sendo que no final é obtido um caminho que corresponde a um conjunto de segmentos. É graças a estes mesmos segmentos que é possível detetar futuras colisões, através da comparação de caminhos de dois AGVs, e verificar se existem segmentos estejam presentes em ambos. Esta comparação é feita através do uso de um diagrama de coordenação, um algoritmo desenvolvido por O'Donnell e Lozano-Pérez [32]. É possível observar na figura 2.8 uma simplificação no modo de funcionamento deste algoritmo. Sempre que um segmento precisa de ser alocado no caminho do AGV, o nível intermédio verifica se esse segmento já está reservado para outro veículo, onde os segmentos reservados estão representados a preto e a cinzento, e todos os segmentos que não se encontram reservados

encontram-se a branco. O algortimo do módulo coordenador, que faz uso do diagrama de coordenação para atribuir segmentos, está totalmente descrito em [30].

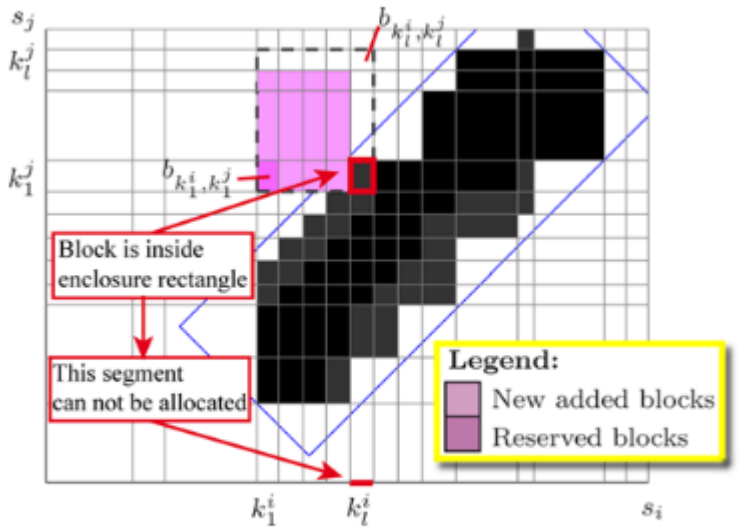


Figura 2.8: Representação de um diagrama de coordenação de dois veículos. Segmentos reservados estão preenchidos a preto e a cinzento, e segmentos não reservados não se encontram preenchidos (Retirado de [30])

A informação exterior necessária para este sistema funcionar provém maioritariamente dos AGVs, sendo que a localização é a informação mais importante. Significa que esta solução faz pouco uso de um componente utilizado por [3], sendo esse componente o *digital twin*. Este componente é utilizado como forma de optimização para os processos ligados ao agendamento das tarefas a serem cumpridas pelos AGVs, já que este módulo virtualiza todo o plano correspondente à manufactura, o que permite a análise de dados provenientes de vários pontos, tais como os tempos de execução de certos procedimentos. Para além deste módulo, a solução apresentada por [3] faz uso de ainda mais 5 módulos, sendo que o sistema de frota contém no total 6 módulos. Eles são:

- Módulo de comunicação - Encarregue da comunicação feita através do uso de uma rede industrial, utilizando os seus protocolos para recolher informação proveniente dos dispositivos presentes na fábrica, e transformando esses dados num formato que possa ser analisado.

- Módulo de armazenamento de dados - Este módulo contém todos os dados disponíveis, servindo de base de dados para os outros módulos.

- Módulo de processamento de dados - Este módulo faz toda a análise necessária para a monitorização e a previsão de todo o processo de manufactura.

- Módulo de *Decision-making* - Este módulo está encarregue do agentamento das tarefas relativas aos AGVs de uma forma dinâmica e optimizada.
- Módulo de Interface humana-máquina - Este módulo disponibiliza informação ao operário sobre os processos que estão a decorrer bem como as tarefas que correspondem a cada AGV.
- Módulo de *Digital twin* - Este módulo produz uma imagem virtual da fábrica, onde é possível prever e optimizar o processo de attribuição de tarefas.

A figura 2.9 representa um esquema que contém todos os módulos referidos acima, bem como a conectividade entre eles. As setas representam a ligação entre os módulos, feita através do uso do módulo de comunicação.

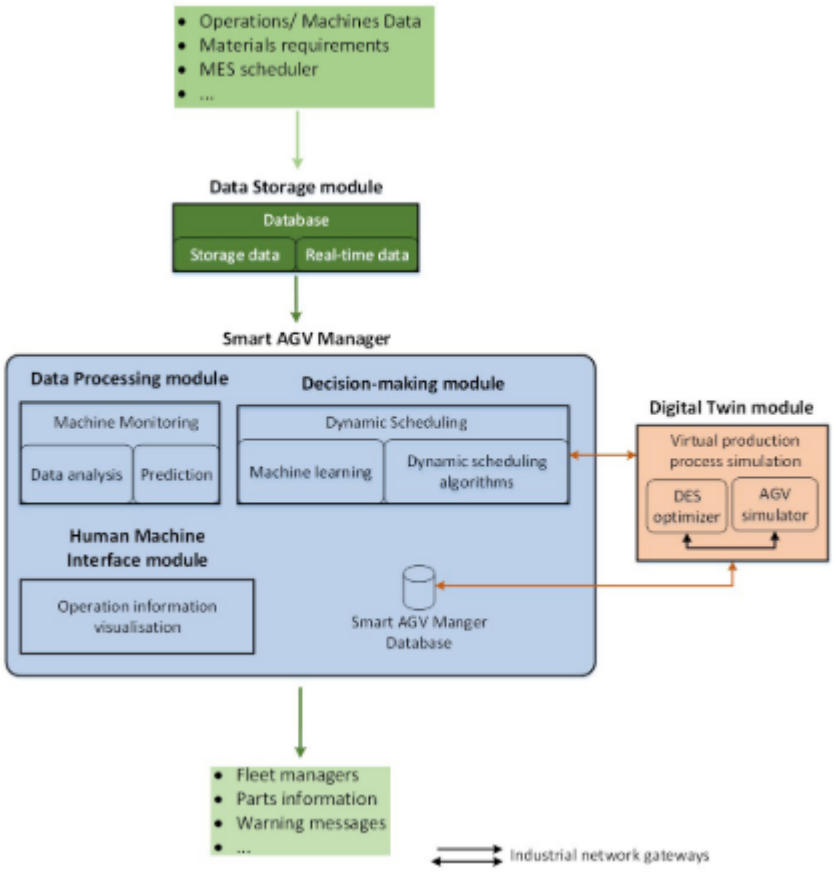


Figura 2.9: Arquitetura correspondente ao sistema proposto por [3] (Retirado de [3])

2.2.3 Detecção de falhas

Para a detecção de falhas que possam ocorrer durante a execução de tarefas que estajam a ser efectuadas pelos AGVs, Pandu Sandi Pratama et al. [33] fazem uso de uma técnica previamente usada, presente em vários casos na secção 2.1, referente à localização. Esta

técnica corrresponde ao uso de um filtro de Kalman, mais concretamente, de um *Extended Kalman Filter* (EKF). O objetivo do uso deste filtro será fazer uma previsão da localização do AGV e comparar os dados provenientes dos sensores aos dados do modelo previamente definido. A detecção da falha é feita através da distância de Mahalanobis, utilizando a expressão $s_t = \tilde{y}_k^T \times S_k^{-1} \times y_k$, onde s_t corresponde ao resíduo, e comparando esse mesmo resíduo a um *threshold* previamente definido. Este passo corresponde apenas à detecção de uma possível falha, sendo que só depois de aplicar o método descrito pelo autor como a isolação da falha é que é possível identicar a mesma e a sua origem, isto é, de que componente e/ou sensor é que causou essa mesma falha. Para tal um conjunto de testes são efectuados, estando presentes na tabela 2.4. De notar que o autor, para o seu caso de estudo, utilizou 4 módulos que forneciam informação sobre a posição do veículo, estando descritos na legenda da mesma.

Tabela 2.4: Relação entre testes e falhas. O normal corresponde ao estado normal do veículo, a falha F1 corresponde a uma falha no encoder, detectada pelos testes T2 e T3. A falhas F2 corresponde a uma falha no motor utilizado no caso de teste, detectada pelos testes T1 e T3 e a falha F3 corresponde a uma falha no laser utilizado no caso de teste, detecta pelos testes T1 e T2. A falha F4 corresponde a uma falha no sistema NAV, que é um sistema que utiliza o método de triangulação, possuindo um laser e um refletor, para efectuar a localização. Esta última falha é detectada apenas pelo teste T4

TESTES		**Falhas**				
		N	F1	F2	F3	F4
		Normal	*Encoder*	Motor BLDC	*Scanner Laser*	NAV
T1	Velocidade de Referência + Scanner Laser (LMS)	0	0	X	X	0
T2	Encoder + Scanner Laser (LMS)	0	X	0	X	0
T3	Velocidade de Referência + Encoder	0	X	X	0	0
T4	Velocidade de Referência + NAV	0	0	0	0	X

É importante referir que neste método, é possível adaptar esta solução de modo a obter falhas noutros módulos, sendo necessário alguma adaptação por parte dos *thresholds* escolhidos, bem como os testes para isolação da falha correspondente a esse mesmo módulo.

2.3 Técnicas de redução de ruído

Quando se fazem leituras/transformações da realidade para o digital, como no caso da captura de informação pelo uso de sensores, estes dados estarão "manchados"com ruído. Este ruído, na grande maioria dos casos, afeta negativamente os dados que se estão a obter, tornando assim a sua leitura de pouca confiança. É possível observar na figura 2.10 o que o ruído pode fazer a um sinal digital, onde este sinal chega a perder informação,

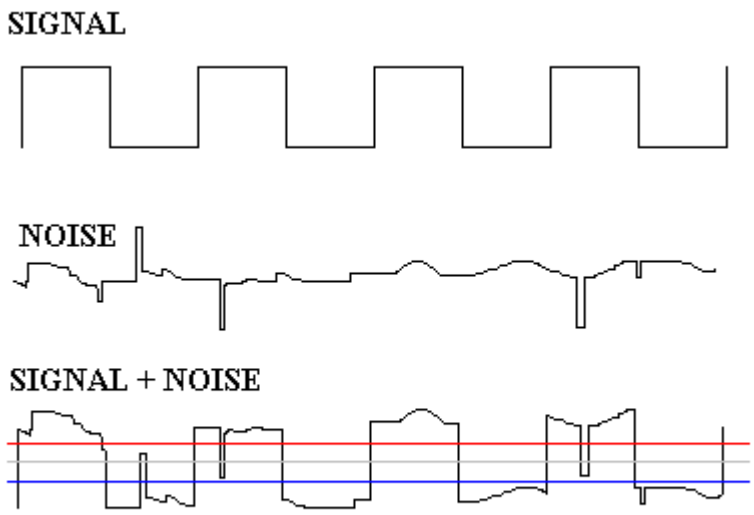


Figura 2.10: Visualização de um sinal digital a ser afetado por ruído.

devido a esse mesmo ruído. Existem vários tipos de ruído, sendo que todos afetam os dados obtidos, e dependendo do tipo de ruído, existem formas de específicas de reduzir o impacto desse mesmo ruído num "*dataset*". Os vários tipos de ruído estão descritos em [34], onde também são apresentados exemplos de origens dos vários tipos de ruído.

Para tentar reduzir ruído indesejado que está presente nos sinais/dados, uma das técnicas utilizadas é a implementação de filtros.

Os filtros são uma classe de processamento de sinal que é capaz de remover partes de um sinal que possam não ser desejadas, como ruído. A forma como estes filtros funcionam para remover as partes indesejadas é através da supressão do próprio. Esta supressão normalmente ocorre em zonas do sinal onde este corresponde mais a ruído do que a informação.

O tipo de filtro depende da situação onde este vai ser aplicado, e como existem vários tipos de filtros, é necessário distingui-los uns dos outros. Como tal, os filtros podem ser divididos e classificados de várias formas, tais como:

- Serem lineares ou não lineares
- Invariantes ou variantes no tempo
- Analógicos ou digitais
- Discretos (amostras) ou contínuos
- Causal ou não causal

Para o cenário em específico, o filtro deverá ser um filtro discreto, visto que o sinal a ser processado será um sinal amostrado, e também deverá ser causal, visto que neste

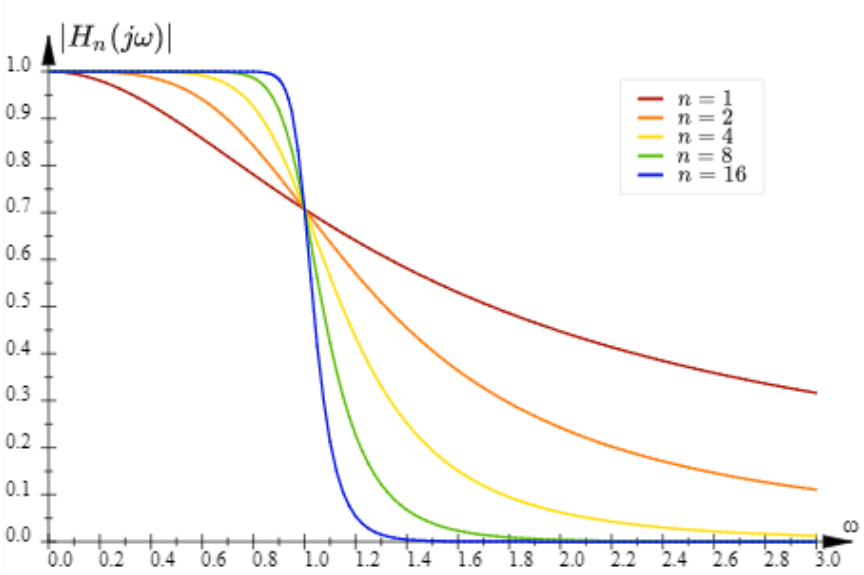


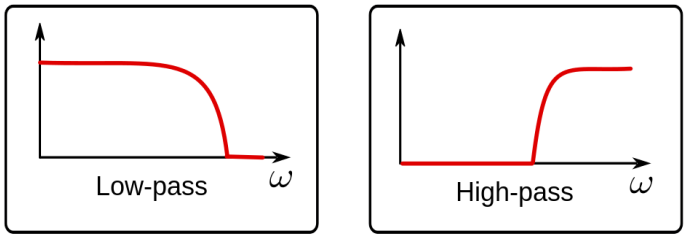
Figura 2.11: Representação gráfica de um filtro "*Butterworth*". O eixo do x corresponde à frequência angular e o eixo do y corresponde ao ganho do filtro. O comportamento do filtro em relação a um sinal será que este deixará o sinal intacto até a frequência de corte, onde ocorre a depressão no gráfico. Esta depressão é afectada pela ordem do filtro, sendo mais precisa quanto maior for a ordem (retirado de [36])

caso, tendo em conta o que foi apresentado na subsecção 2.1.2, a saida do filtro deverá de depender da entrada no instante anterior.

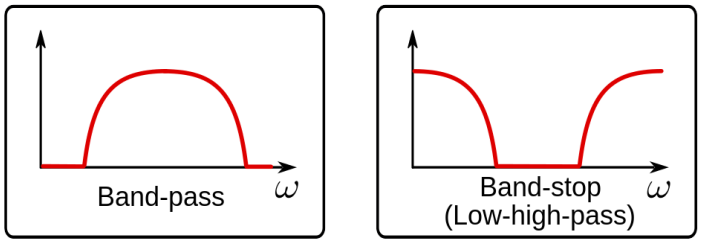
2.3.1 Filtro de Butterworth

Uma opção na implementação do filtro poderá ser o uso de um filtro "*Butterworth*". Este tipo de filtro é bastante utilizado no processamento de sinais no domínio da frequência, devido a estes serem filtros que se enquandram na classificação de analógicos. No entanto, como é demonstrado em [35], onde é possível utilizar este filtro numa forma discreta, que se enquadra à situação de um problema no domínio digital, como é o caso do cenário de estudo. Para o dimensionamento de um filtro "*Butterworth*"é necessário ter definidos os parâmetros da frequência de corte, onde a partir dessa frequência o sinal deverá de ser descartado, a frequência de amostragem do sinal original e a ordem do filtro. É possível observar na figura 2.11 uma resposta em frequência de um filtro de "*Butterworth*", onde é possível observer de que modo a ordem afeta o filtro. Quanto maior a ordem do filtro, maior será a sua precisão de corte, isto é, o filtro conseguirá eliminar melhor todas as frequências que estão acima da frequência escolhida, a qual todas as frequências acima serão descartadas.

No entanto, este filtro não está limitado a eliminar apenas as frequências que estajam acima de uma frequência de corte escolhida, que corresponde a um filtro passa baixo, mas também poderá ter um comportamento inverso, onde a partir da frequência de corte é que este deixa passar o sinal original, eliminando as frequências que estejam abaixo



(a) O gráfico da esquerda corresponde a um filtro passa baixo e o gráfico da direita corresponde a um filtro passa alto



(b) O gráfico da esquerda corresponde a um filtro passa band e o gráfico da direita corresponde a um filtro elimina banda

Figura 2.12: Representação de 4 tipos de filtros. (retirado de [37])

da frequência de corte. Este filtro descrito é um filtro passa alto. Ainda existem mais dois tipos de filtro, o passa banda e o elimina banda, onde o primeiro elimina todo o sinal, excepto num intervalo entre duas frequências, e o segundo tem o comportamento inverso, onde o sinal é só eliminado entre essas duas frequências. Para estes dois casos é necessário duas frequências de corte para definir um intervalo de corte. Na figura 2.12 está demonstrado o comportamento destes 4 filtros.

O uso deste filtro, no caso de sinais digitais, está ligado à redução de ruído de altas frequências, isto é, a aplicação de um filtro passa baixo no sinal/dados a serem processados. Este tipo de processamento reduz o ruído presente nos dados, no entanto, este tipo de filtro não verifica se o valor presente nesses mesmo dados corresponde à realidade ou não (figura 2.13).

2.3.2 Filtro de Kalman

O filtro de Kalman corresponde a um tipo de filtro muito utilizado no processamento de sinais/dados. Isto deve se ao facto de este filtro não só é capaz de reduzir o ruído presente nas medições provenientes de sensores, como acelerómetros, mas ao mesmo tempo, este filtro é capaz de "recuperar"dados perdidos devidos ao ruído. Isto acontece pois este filtro utiliza não só as medidas obtidas por sensores, mas necessita também de um modelo do

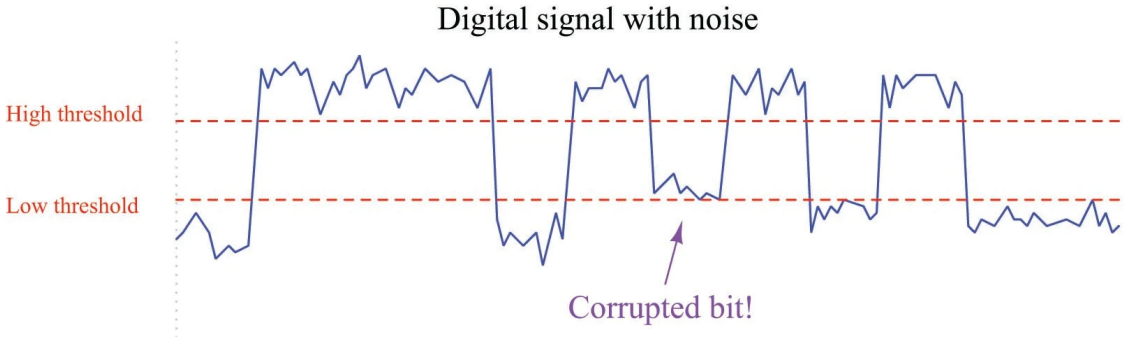


Figura 2.13: Gráfico que contém uma demonstração de como o ruído pode afetar um sinal digital, alterando o valor do mesmo (retirado de [38])

sistema associado ao sensor, como por exemplo, o carro ser o sistema e o acelerómetro o sensor. O objetivo do filtro utilizar tanto um sistema e as medições deve se ao facto de este fazer uma "comparação"entre o que o filtro espera que seja o comportamento do sistema num instante, devido a informação proveniente do instante anterior, e as medidas obtidas. Deste modo, este filtro é capaz de suavisar os dados, reduzindo uma oscilação que possa ocorrer nas leituras, e consegue aferir o estado atual do sistema através da comparação entre dois tipos de informação diferente, uma teórica e outra real.

Este filtro é bastante utilizado em cenários onde o objetivo é fazer o "*tracking*"de veículos, tais como AGVs, em ambientes fechados ("*indoors*"), como é possível observar em [15, 23, 39, 40]. As diferenças aparentes destes trabalhos mencionados correspondem apenas aos modelos escolhidos, sendo que cada um depende do cenário em concreto bem como o equipamento utilizado para fazer as medições, e o sub tipo de filtro de Kalman utilizado. No entanto, e como é possivel obervar na figura 2.5 na página 17, a base do funcionamento deste filtro é sempre a mesma, sendo que a cada iteração este executa dois passos, um primeiro passo que corresponde à previsão, e o segundo passo que corresponde ao "*update*".

A previsão é o passo onde o filtro processa e calcula os valores teóricos referentes ao estado do sistema, tendo em conta a informação do estado anterior desse mesmo sistema. Numa situação onde o estado anterior do sistema correspondia a estar parado, e a informação recebida é de que não existem forças exteriores a afetar este mesmo sistema, então, no passo da previsão, o filtro de Kalman vai calcular que o estado atual do sistema deverá de corresponder a este encontrar-se parado. Isto corresponde apenas a um exemplo demonstrativo do funcionamento da primeira etapa do filtro de Kalman. Esta fase é caracterizada por duas equações, sendo elas,

$$\bar{\mathbf{x}} = \mathbf{F}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{2.7}$$

$$\bar{\mathbf{P}} = \mathbf{F}\mathbf{P}\mathbf{F}^{\mathrm{T}} + \mathbf{Q} \tag{2.8}$$

onde **x** corresponde à variável de estado do sistema contendo toda a informação referente a esse mesmo estado, como por exemplo, informação sobre a velocidade e posição do sistema. Normalmente esta variável corresponde a um vetor, como é possível ver na matriz 2.9, onde a variável de estado neste caso corresponde à posição x e à velocidade $\dot{x}$.

$$\mathbf{x} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \tag{2.9}$$

Na equação 2.7, a variável **x** corresponde ao estado anterior do sistema e a variável $\bar{\mathbf{x}}$ corresponde ao estado atual do sistema. **F** corresponde à matriz função de transição de estado do sistema, isto é, representa todas as caracteristícas que promovem à mudança de um estado, por exemplo, mudança de velocidade através da equação do movimento $v = v_0 + at$. A parte da equação que contém **Bu** corresponde a se o sistema recebe "*inputs*"do exterior, tais como um comando do utilizador a forçar a alteração do comportamento do sistema onde, tendo em conta que o cenário em questão, que corresponde a monitorização de AGVs, esta parte do filtro de Kalman pode ser negligenciada.

A segunda equação (2.8) corresponde à matriz de covariância, que contém a covariância entre todas as variáveis presentes no vetor **x**. Estas covariâncias representam o quanto cada variável depende das outras, isto é, utilizando o exemplo da velocidade e da posição, a covariância da velocidade com a posição indica-nos de que modo é que a posição varia em relação a variações da velocidade. Desta forma, para um vetor **x** de $1 \times n$, esta matriz terá um tamanho de $n \times n$. Para este exemplo esta matriz seria, por exemplo,

$$\mathbf{P} = \begin{bmatrix} xx & x\dot{x} \\ \dot{x}x & \dot{x}\dot{x} \end{bmatrix} \tag{2.10}$$

onde xx corresponde à covariância da posição em relação à posição (também conhecida apenas por variância, neste caso, da posição), $x\dot{x}$ corresponde à covariância da posição em relação à velocidade, $\dot{x}x$ corresponde à covariância da velocidade em relação à posição e $\dot{x}\dot{x}$ corresponde à covariância da velocidade em relação à velocidade (variância da velocidade). Da mesma forma que a cada iteração de $\dot{\mathbf{x}}$ é calculada à custa de **Fx**, a covariância do estado atual é calculada através de **F** e de **F** transposto ($\mathbf{F^T}$), e ainda da matriz **Q**, que corresponde à covariância relativa ao ruído do processo, isto é, o ruído que pode ser introduzido através dos cálculos.

Após feito o passo referente à previsão, o próximo passo, que corresponde ao passo do "*update*", é iniciado, onde o objetivo deste passo é juntar a informação obtida no passo anterior com a informação obtida pelas medidas recebidas pelos sensores. Deste modo o filtro, dependendo das especificações, não irá ser muito afetado por ruído que possa estar presente nas medições dos sensores. Esta fase é caracterizada pelas seguintes equações,

$$\mathbf{y} = \mathbf{z} - \mathbf{H}\bar{\mathbf{x}} \tag{2.11}$$

$$\mathbf{K} = \bar{\mathbf{P}}\mathbf{H}^\mathbf{T}(\mathbf{H}\bar{\mathbf{P}}\mathbf{H}^\mathbf{T} + \mathbf{R})^{-\mathbf{1}} \tag{2.12}$$

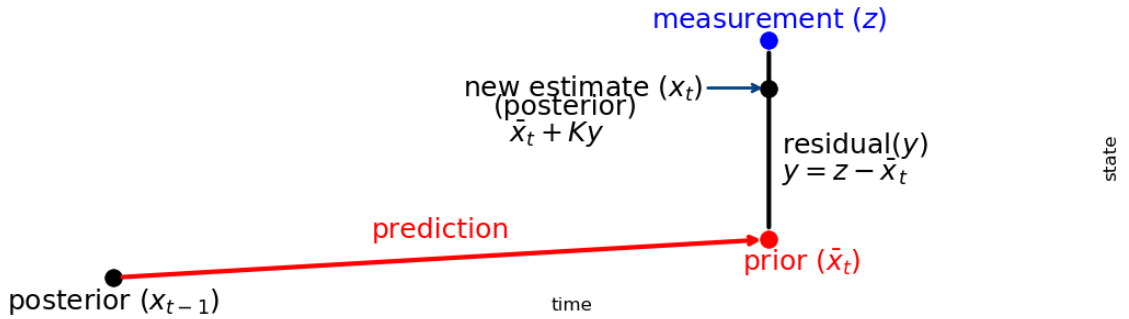


Figura 2.14: Demonstração simplificada do funcionamento de um filtro de Kalman. Contêm o passo referente à previsão, representado pela linha vermelha, e o passo correspondente ao "*update*", representado pelo ponto preto (retirado de [41].

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{K}\mathbf{y} \tag{2.13}$$

$$\mathbf{P} = (\mathbf{I} - \mathbf{K}\mathbf{H})\bar{\mathbf{P}} \tag{2.14}$$

onde $\mathbf{z}, \mathbf{R}$ correspondem à matriz de medidas provenientes dos sensores e a matriz das covariâncias dos respetivos ruídos associados a essas medidas. A matriz $\mathbf{H}$ corresponde à função de medidas, que converte/elimina toda a informação presente na matriz de estado para que esta seja semalhante em termos de conteúdo (como por exemplo conter uma velocidade com a escala correta) para que esta depois possa ser subtraída à matriz $\mathbf{z}$. As matrizes $\mathbf{y}$ e $\mathbf{K}$ correspondem ao resíduo e ao ganho de Kalman. O resíduo corresponde à diferença entre a medida obtida pelos sensores e o valor que está presente em $\bar{\mathbf{x}}$ que representa essa medida calculada no passo da previsão. O ganho de Kalman corresponde a um valor que irá ser utlizado para o cálculo do novo $\mathbf{x}$ e do novo $\mathbf{P}$ que irão ser utilizados no passo da previsão da próxima iteração. Finalmente, a matriz $\mathbf{I}$ representa a matriz identidade com dimensão $\mathbf{n} \times \mathbf{n}$. A imagem 2.14 representa uma demonstração do funcionamento teórico de um filtro de Kalman. O "*posterior*"representa $\mathbf{x}$ e o "*prior*"corresponde a $\bar{\mathbf{x}}$.

Apesar de o filtro de Kalman ser bastante promissor para o processo de "*tracking*", a sua forma original apenas funciona para casos lineares. Isto complica a sua aplicação em situações que representam casos reais, visto que o comportamento de um AGV não é sempre linear, devido a mudanças constantes de velocidade, derrapagens que possam acontecer, atrito que possa existir que afete o movimento do AGV, etc. Visto que estas situações existem e afetam o sistema, foram desenvolvidas iterações do filtro de Kalman, sendo que as mais comuns, para casos não lineares e que sejam utilizados para casos onde é necessário fazer o "*tracking*"de veículos num espaço interior, são o "*Unscented Kalman Filter*", utilizado em [40], e o "*Extended Kalman Filter*", utilizado em [23] e em [15], sendo que este neste último trabalho foi utilizado uma variação do "*Extended Kalman Filter*", chamada "*Invariant Extended Kalman Filter*".

Apesar de existirem várias variantes do filtro de Kalman, o processo ligado ao seu desenvolvimento corresponde sempre aos mesmo 6 passos, sendo eles:

1. Escolher variáveis de transição para o vetor **x** (tais como as que estão presentes em 2.9)
2. Projectar a matriz de transição de estado, que correspondem à matriz **F** (um exemplo de uma matriz de transição para o caso 2.9 poderá ser 2.15)
3. Projectar a matriz de ruído do processo, que corresponde à matriz **Q**
4. Projectar a matriz de conversão de medidas, sendo ela a matriz **H**
5. Projectar a matriz de ruído associado às medições feitas pelos sensores, que corresponde à matriz **R**
6. Definir uma condição inicial para ser colocada no vetor das variáveis de transição

Seguindo estes passos descritos acima, é possível desenvolver um filtro de Kalman e que, se os parâmetros estiverem adequados ao cenário e ao equipamento escolhido, este pode ser uma opção para a redução de ruído. É importante salientar que, visto que o cenário em questão contém várias situações que retiram a linearidade ao problema, o filtro de Kalman a ser escolhido deverá de estar apto para ser utilizado em sistemas não lineares, como é o caso das duas iterações do filtro acima apresentadas, o "*Unscented Kalman Filter*"e "*Extended Kalman Filter*".

$$\mathbf{F} = \begin{bmatrix} 1 & \Delta t \\ 0 & t \end{bmatrix} \tag{2.15}$$

O filtro de "*Kalman*"que aparente ser promissor na aplicação para uma possível solução corresponde ao "*Unscented Kalman Filter*", sendo que uma descrição bastante simples está presente em [41], que corresponde a um livro interactivo desenvolvido pelo autor da biblioteca "*filterpy*", que corresponde a uma biblioteca de python, onde esta contém vários métodos relacionados com filtros de "*Kalman*", tais como a criação e aplicação de filtros sobre um conjunto de dados que estejam previamente armazenados num ficheiro, ou que estes estejam disponiveis em um buffer. Este livro corresponde a uma boa leitura no que toca a tentar perceber de que maneira é que são implementados estes filtros, qual a sua teoria, juntamente com as suas fórmulas associadas, mas tentando sempre simplificar, facilitando assim a compreensão.

Capítulo 3

Trabalho desenvolvido

Serve o presente capítulo para apresentar todo o trabalho desenvolvido nesta tese, bem como a arquitectura proposta para a solução no desenvolvimento de um gestor de frota para AGVs, estando este dividido em várias secções. A secção 3.1 apresenta a referida arquitectura conceptual proposta, bem como uma lista de requisitos para a sua implementação. A secção 3.2 introduz a tecnologia utilizada, e a secção 3.3 introduz o IMU utilizado para o desenvolvimento deste trabalho, incluindo uma descrição dos dados disponíveis, como é que estes estão disponíveis, e os passos necessários para a configuração do equipamento. A secção 3.4 apresenta o desenvolvimento do algoritmo de *tracking*, sendo que este algoritmo não possui filtros implementados que eliminem algum do ruído que possa existir, sendo que este algoritmo só trabalha sobre as leis da física e equações do movimento. A secção 3.5 introduz uma implementação que envolve o tratamento dos dados recebidos, utilizando filtros, de modo a eliminar ruído que possa estar presente e afete negativamente os resultados.

3.1 Arquitectura conceptual

A arquitetura conceptual para este projeto está presente na figura 3.1. Esta arquitetura corresponde a uma visão global, que corresponde à solução proposta no âmbito desta tese para o projeto apresentado no capítulo 1, nomemadamente na secção 1.4. O foco principal do trabalho desenvolvido corresponde a um desenvolvimento inicial ligado ao módulo de localização apresentado nesta arquitetura.

Esta conceptualização terá como requisitos principais:

- Manter uma constante monitorização da localização "*indoor*"de todos os AGVs, sem recurso a GPS.

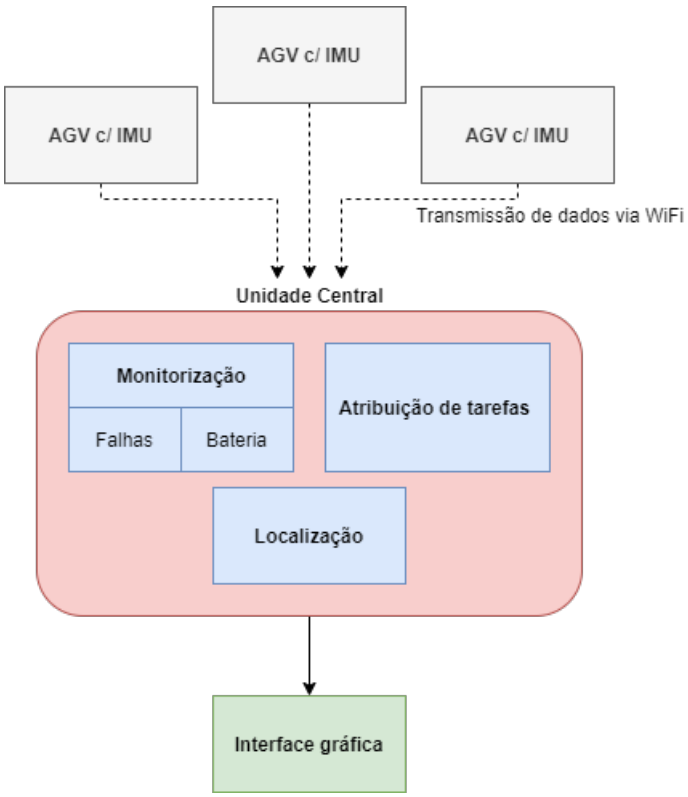


Figura 3.1: Arquitetura conceptual do projecto a ser alcançado

- Atribuir tarefas a todos os AGVs, bem como a atribuição do plano de trajeto.
- Monotorizar todos os AGVs, de modo a verificar se estes estão em movimento ou se se encontram estacionários.
- Avaliar se os AGVs estão em execução normal ou se existe alguma anomalia.
- Manter uma monitorização do consumo das baterias dos AGVs.
- Notificação em caso de falha e/ou anomalia.

3.2 Tecnologia escolhida - IMU

A tecnologia utilizada no âmbito do desenvolvimento desta tese corresponde à tecnologia referente a sensores inerciais, tais como acelerómetros e giroscópios. Este tipo de sensores fornecem informação sobre a aceleração e velocidade que um corpo sofre numa dada orientação, seja ela uma aceleração linear, no caso do acelerómetro, ou a velocidade angular, para o giroscópio. O livro "*Handbook of Signal Processing Systems* [42] contém um capítulo dedicado apenas a este tipo de sensores, contendo informação sobre a forma de captação dos dados de cada um dos sensores (acelerómetros e giroscópios), as várias área/aplicações que este tipo de sensores tem, tais como na navegação, desporto ou até

em produtos do consumidor, e ainda fala de detalhes tais como de que maneira vários tipos de ruído afetam a performance destes sensores. Este livro ainda faz referência ao facto de este tipo de sensores nem sempre são precisos, e que os que são possuem grandes dimensões e custos que reduzem a sua aplicabilidade geral. No entanto é descrito ainda que é possível contornar a precisão dos sensores mais baratos com várias técnicas.

Tendo em conta o cenário referente ao processo logístico numa fábrica, onde já existem procedimentos intrínsecos, a escolha de uma tecnologia para efetuar o *tracking* destes AGVs envolve 6 parâmetros, sendo eles:

- A tecnologia não pode ser intrusiva ao processo logístico presente
- Tem que funcionar num ambiente fechado, isto é, sem auxilio de um GPS
- Tem que possuir custos baixos (montagem e/ou manutenção)
- Tem que ser robusta
- Tem que ser uma tecnologia que implique um baixo consumo de energia
- Tem estar apta a funcionar em tempo real

Como tal, uma tecnologia que respeita estes 3 pontos acima corresponde à tecnologia que faz uso de sensores inerciais, nomeadamente, unidades de medida inercial (IMUs), o que levou a escolha dos mesmos para a implementação e desenvolvimento do sistema de *tracking* desenvolvido no âmbito desta tese.

3.3 IMU - Especificações

O IMU escolhido para o desenvolvimento desta tese foi o BITALINO R-IOT KIT FULL-FEATURED 9DOF WIRELESS IMU IN A STAMP-SIZED PACKAGE WITH DIRECT OSC STREAMING OVER WIFI. Este IMU possui, como componentes físicos, um acelerómetro tri axial, que mede a aceleração instantânea em força G nos 3 eixos, um giroscópio tri axial, que fornece a velocidade angular em graus por segundo de cada eixo. Uma representação visual dos eixos relativos ao IMU estão presentes na imagem 3.2.

Para além destes dois sensores, este IMU possui ainda um magnetómetro tri axial, que mede a intensidade do campo magnético em Gauss nos três eixos, e um sensor de temperatura, que fornece a temperatura em graus centígrados. Para além dos sensores, este IMU possui ainda 2 ports analógicos, uma resolução de 16 bits por cada canal do IMU e de 12 bits por cada canal analógico. Possui um "*sampling rate*"de 200 Hz, isto é, a cada segundo são amostrados e enviados 200 amostras de dados referentes a todos os sensores do IMU, e é capaz de efetuar comunicação através do protocolo WiFi 2.4 GHz. As dimensões do kit correspondem a 34x23x7mm. Na imagem 3.3 é possível observar um kit BITalino RIoT com as especificações acima apresentadas.

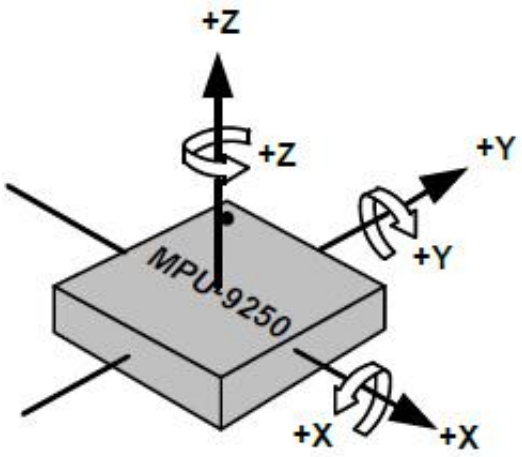


Figura 3.2: Representação gráfica dos eixos num IMU (o dispositivo em si não é representativo do dispositivo utilizado

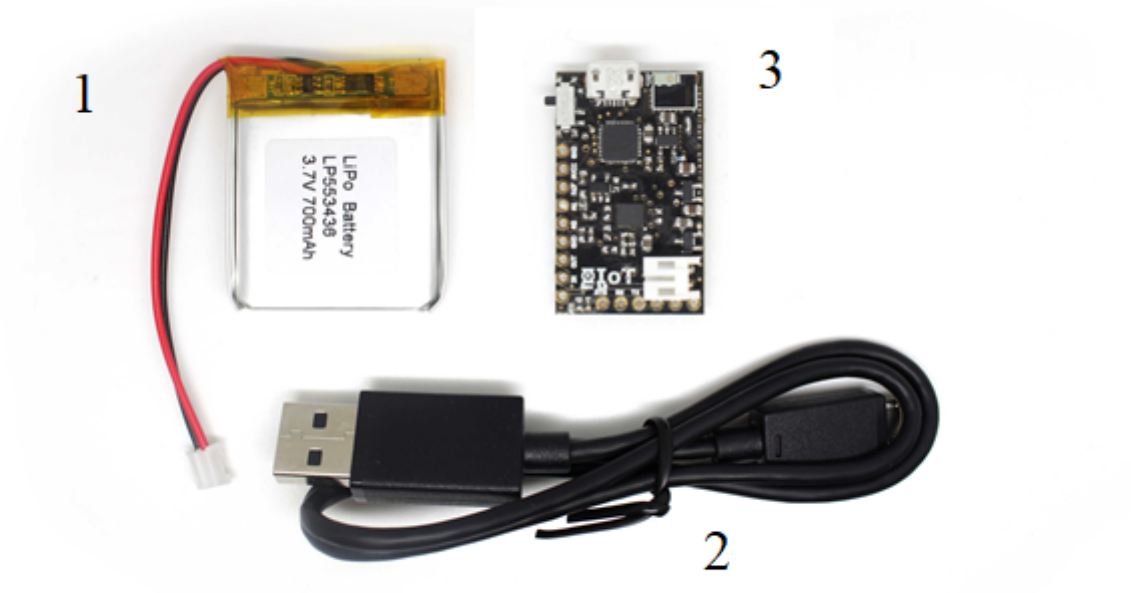


Figura 3.3: Kit de IMU utilizado no desenvolvimento desta tese. Este kit é composto por uma bateria (1), um cabo USB (2) e um IMU (3).

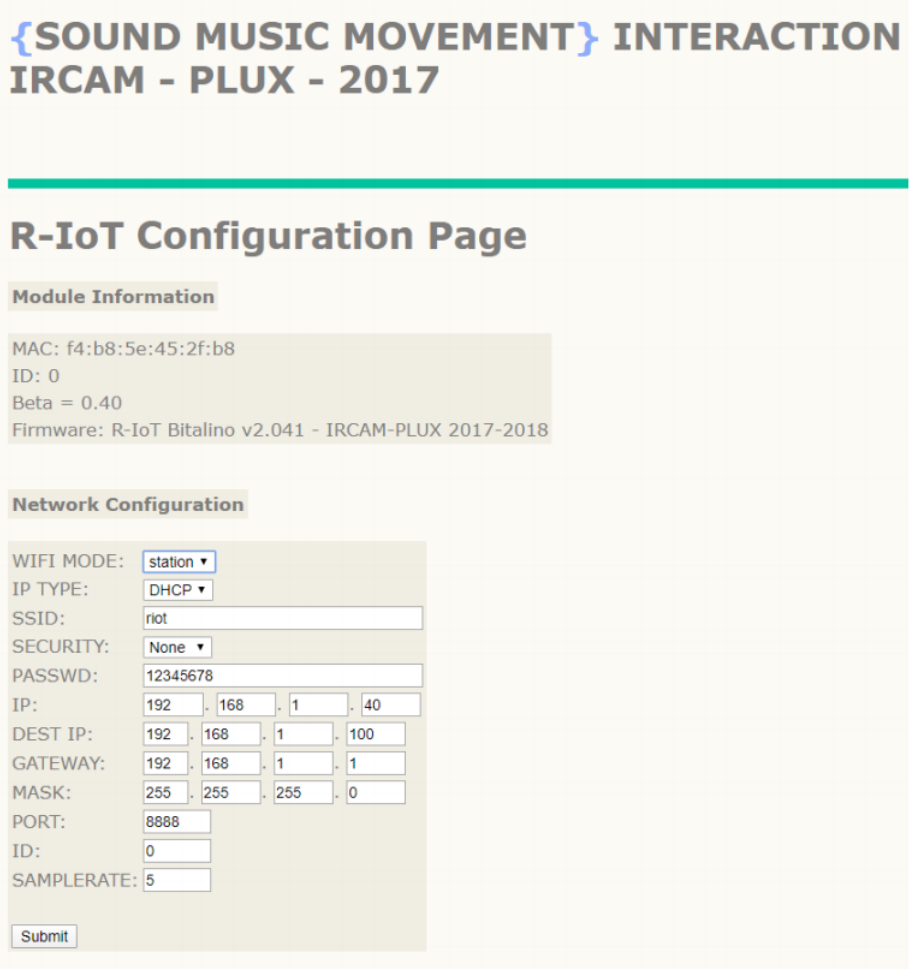


Figura 3.4: Página de configuração do IMU.

3.3.1 Configuração

O "*firmware*" presente no kit fornece uma utilização "*out of the box*" onde todas as configurações essenciais são feitas no terminal onde os dados irão ser recolhidos, bem como na rede WiFi presente. Quando o kit é ligado, este acende um LED vermelho a indicar a inicialização do firmware. Quando este está totalmente inicializado, o LED passa a verde intermitente, que significa que o kit está à procura da rede WiFI para qual este está configurado (o "default" é uma rede com ssid riot e palavra pass riot), e quando ele se liga a uma rede, o LED passa a azul. É possível configurar o ssid da rede onde o kit se liga, juntamente com a respetiva palavra pass, mas também é possível escolher a atribuição de IP, bem como o IP e port do destinatário para onde os dados são enviados. A página de configuração está presente na imagem 3.4, e é possível aceder a e essa página colocando o kit como ponto de acesso (tutorial completo da configuração do kit encontra-se neste link https://bitalino.com/docs/R-IoT_Configuration_Guide.pdf).

Tabela 3.1: Tabela com a informação fornecida pelos sensores presentes no IMU e respetiva descrição.

Sensores	**Descrição**	**Unidade**
Acelerómetro triaxial	Aceleração linear nos eixos x, y e z	Força G
Giroscópio triaxial	Velocidade angular nos eixos x, y e z	Graus p/ segundo (°/s)
Magnetómetro triaxial	Intensidade do campo magnético nos eixos x, y e z	Gauss (B)
Temperatura	Temperatura do meio ambiente	Graus centígrados (°)

Tabela 3.2: Tabela com a informação fornececida pelo "*firmware*"do IMU e respetiva descrição.

Dados	**Descrição**
Quaternion 1	Fornece informação do quaternion associado ao vector x
Quaternion 2	Fornece informação do quaternion associado ao vector y
Quaternion 3	Fornece informação do quaternion associado ao vector z
Quaternion 4	Fornece informação do quaternion associado ao veetor w
Yaw	Fornece informação da orientação sobre o eixo z
Pitch	Fornece informação da orientação sobre o eixo y
Roll	Fornece informação da orientação sobre o eixo x
Heading	Fornece informação da direcção relativa do sensor

3.3.2 Dados

Os dados que são fornecidos pelo IMU são as acelerações nos 3 eixos do sensor do acelerómetro, sendo que as unidades correspondem a força G, as velocidades angulares também nos 3 eixos, onde as unidades são graus por segundo, a intensidade do campo magnético, em Gauss, nos 3 eixos e a temperatura em graus centígrados. Para além destes dados, este kit ainda é capaz de fornecer os dados sobre o estado dos dois "*switches*"presentes na placa, possui dois canais analógicos, e ainda fornece a dados referentes aos seus "*quaternions*", "*yaw*", "*pitch*", "*roll*"e "*heading*", onde estes últimos 4 estão em graus centígrados. Os dados fornecidos pelos sensores do IMU estão presentes na tabela 3.1 e os dados fornecidos pelo "*firmware*"do IMU estao presentes na tabela 3.2.

A transmissão é feita através de uma rede WiFi, onde a captação dos dados e a sua respetiva transmissão é feita a 200Hz, isto é, cada conjunto de 200 dados é enviado num espaço de um segundo. O protocolo de comunicação corresponde ao "*Open Sound Control*" (OSC). Este protocolo é descrito em [43] como eficiente, independente do tipo de dados a serem enviados, sendo um tipo de protocolo que se baseia na mensagem. Este é utilizado na troca de informação entre computadores, sintetizadores e outros dispositivos multimédia. O software indicado pela empresa, que é compatível com o protocolo de streaming, é o "*OpenSignals*", que permite captar os dados do kit em tempo real, mas também permite o armazenamento desses mesmo dados em ficheiros nos formatos de texto (txt), Hierarchical Data Format 5 (h5) e European Data Format (edf).

Em relação à perda de pacotes, que corresponde à perda de informação proveniente

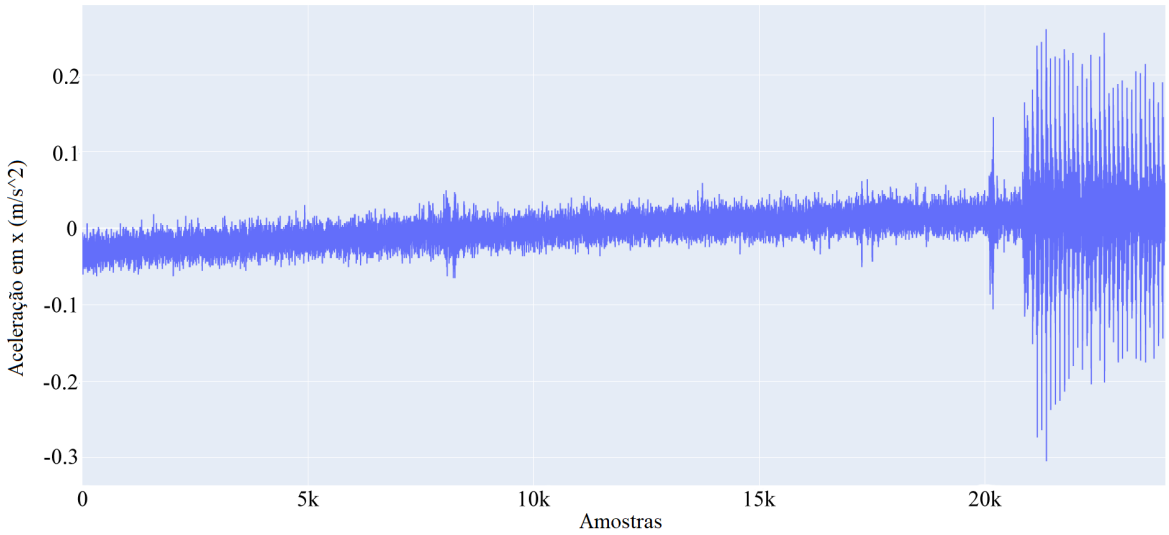


Figura 3.5: Representação gráfica de um conjunto de dados obtidos pelo IMU. O teste envolve o IMU estar parado durante um período de tempo de 2 minutos. É possível observar o ruído presente nas leituras, principalmente na parte final do teste, sendo que é desconhecido a causa deste aumento.

do IMU durante o transporte entre este e o terminal, nos testes feitos no decorrer desta tese, não foram detectados quaisquer perdas, salientando que os testes nunca passaram um tempo de execução para além dos 5 minutos, devido a limitações referentes ao espaço onde os testes foram realizados, bem como o equipamento disponível. No entanto, a ligação entre o IMU e o respetivo Ponto de Acesso (AP) está limitada por este último, nomeadamente à distância máxima que estes dois podem estar um em relação do outro, podendo ocorrer um corte na ligação pré estabelecida.

Os dados transmitidos, quando observados na figura 3.5, é possível verificar que estes possuem bastante ruído, sendo que este está relacionado com os sensores, isto é, durante a sua captação, e também poderá estar relacionado com o empacotamento dos mesmos, que por sua vez está ligado ao "*firmware*"do dispositivo. A presença de campos magnéticos pode ser também uma causa que aumenta ainda mais o efeito do ruído presente. A razão pela qual a parte final do teste apresenta um aumento substancial do ruído presente é desconhecida.

Para a implementação das soluções que irão ser abordadas nas secções seguintes, os dados necessários correspondem à aceleração no eixo x e o "*yaw*", fornecido pelo próprio "*firmware*"do dispositivo. O IMU deverá de estar colocado no AGV como está representado na figura 3.6, de modo a garantir que as leituras estejam concordantes com o algoritmo desenvolvido.

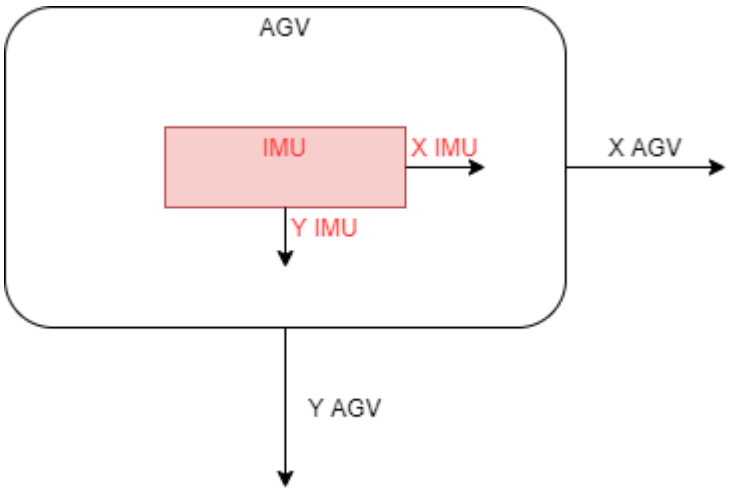


Figura 3.6: Esquema representando a orientação da colocação do IMU no AGV de modo a permitir a correta funcionalidade do algoritmo deseenvolvido

3.4 Metodologias de localização

Nesta secção irá ser apresentado a abordagem utilizada, capaz de efetuar o "*tracking*" através do uso do IMU apresentado na secção anterior. Este tipo de "*tracking*" é chamado de "*Dead Reckoning*" [15] pois envolve o cálculo da posição/estado atual à custa da posição/estado anterior, bem como da aceleração ou da velocidade durante um intervalo de tempo. A implementação que irá ser apresentada foi desenvolvida em código python, utilizando como ambiente de desenvolvimento integrado (IDE) o PyCharm. As bibliotecas utilizadas durante o desenvolvimento do algoritmo serão descritas na secção 3.4.2, juntamente com o algoritmo desenvolvido. A subsecção 3.4.1 introduzirá aos cálculos gerais do movimento utilizados em ambas as implementações.

3.4.1 Equações do movimento

Como já foi referido, para a implementação do algoritmo de localização, é necessário o uso das equações do movimento, pois estes algoritmos dependem da aceleração obtida pelo acelerómetro presente no IMU. Como tal, tendo em conta que a única variável disponível para indicar que houve deslocamento é a aceleração, então as equações que permitirão obter o deslocamento serão então,

$$\begin{aligned} v &= \int a\,dt \\ d &= \int v\,dt = \iint a\,dt \end{aligned} \tag{3.1}$$

onde a primeira equação corresponde à velocidade, v, obtida pela integração da aceleração, a, ao longo do tempo, t. A segunda equação corresponde ao deslocamento, d, que é obtido

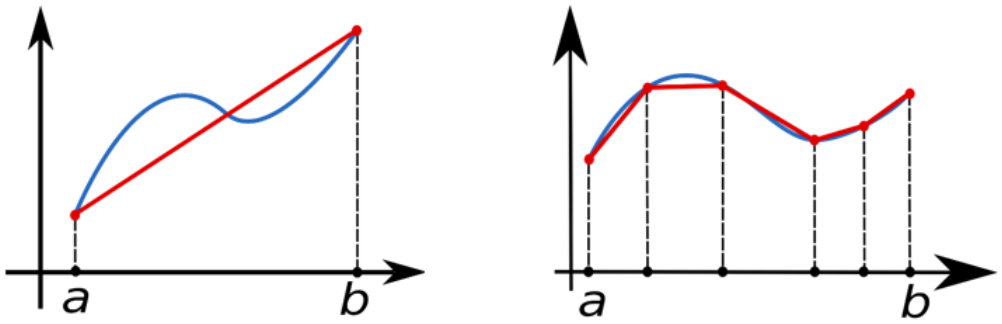


Figura 3.7: Representação gráfica da aplicação da regra do trapézio. É possível observar que a aproximação será tanto melhor quanto o número de intervalos que dividem a função original

através da integral ao longo do tempo da velocidade, v, ao longo do tempo, t, que por sua vez pode ser obtido através do integral duplo da aceleração, a. Desenvolvendo os integrais acima apresentados, obtemos as seguintes equações,

$$\begin{aligned} v &= v_0 + a\Delta t \\ d &= d_0 + v_0\Delta t + \frac{1}{2}a\Delta t^2 \end{aligned} \tag{3.2}$$

onde a,v e d correspondem respetivamente à aceleração, à velocidade e ao deslocamento, como referido acima. Quanto a v_0 e a d_0, estes correspondem à velocidade inicial e ao deslocamento inicial. A variável Δt corresponde ao intervalo de tempo.

3.4.2 Primitivação pela regra do Trapézio

O primeiro algoritmo apresentado por esta tese consiste no uso das duas primeiras equações apresentadas acima, onde corresponde à primitivação da aceleração e da velocidade. No entanto, a capacidade computacional para o cálculo de integrais é bastante limitada, o que torna essencial o uso de técnicas que reduzam a complexidade do cálculo integral. Uma desses técnicas é o cálculo do integral utilizando a regra do trapézio. Esta regra consiste em simplificar o cálculo do integral, fazendo com que este passe a ser o cálculo da área de um trapézio ou a soma da área de vários trapézios. A forma de desenhar o trapézio é através da escolha de 2 pontos da função a ser integrada, como está representado na figura abaixo com os pontos a e b, onde $f(a)$ e $f(b)$ correspondem às bases do trapézio, e $b-a$ corresponde à base do trapézio. Como esta regra consiste em fazer aproximações da área real do integral, quanto menor for o intervalo entre os pontos que correspondem a um trapézio, melhor será a aproximação.

O integral passa então a ser calculado da seguinte forma,

$$\int_a^b f(x)dx \approx \sum_{i=1}^{N} \frac{f(x_{i-1}) + f(x_i)}{2} \times (x_i - x_{i-1}) \tag{3.3}$$

onde o integral $\int_a^b f(x)dx$ é calculado pela soma da área de todos os trapézios. Na implementação do algoritmo, como por exemplo para o cálculo da velocidade através da integração da aceleração, o $f(x_i)$ e $f(x_{i-1})$ correspondem à aceleração da amostra atual e da amostra anterior, e o intervalo $(x_i - x_{i-1})$ corresponde ao frequencia de amostragem do IMU, podendo ser substituído na fórmula anterior por uma constante $c = 200Hz$. Esta substituição permite ter um número constante de trapézios e igualmente espaçados, e o facto de a frequência corresponder a 200 Hz, faz com que a aproximação, quando se aplica a regra do trapézio, esteja bastante aproximada da realidade, tendo assim um erro baixo em relação à função original. A constante N corresponde ao número total de samples para o caso "*offline*". Para o caso "*online*", onde o programa seria executado em tempo real, a constante N corresponderia ao número de samples presentes num buffer.

3.4.3 Algoritmo com a primitivação pela regra do trapézio

O fluxograma que corresponde ao algoritmo desenvolvido está presente na figura 3.8. Este algoritmo está desenvolvido para uma execução offline, isto é, é necessário obter todos os dados referentes ao movimento do AGV, armazená-los num ficheiro do tipo texto (.txt), e depois aplicar o algoritmo nesses mesmo dados. Serão necessárias fazer alterações de modo a acomodar a necessidade de uma execução em tempo real.

Como foi referido no início da secção 3.4, a implementação deste algoritmo foi feita em python, sendo que para a implementação da primitivação por partes foi utilizada a biblioteca "*scipy*", que é uma biblioteca "*open source*" que contém vários módulos que têm como objetivo simplificar a aplicação de conceitos de matemática, como o cálculo de matrizes, ciência e engenharia. Dentro dessa biblioteca existe um módulo "*integrate*" que contêm vários métodos de integração implementados em python, sendo dois deles o método "*trapz*" e o método "*cumtrapz*". O método "*trapz*" executa a primitivação de uma função $f(x)$, podendo ser integrada ao longo de um eixo ou utilizando um intervalo constante [44]. O método "*cumtrapz*" faz o mesmo que o método "*trapz*" onde a única diferença é que este método executa o cálculo da primitivação cumulativamente, isto é, a cada intervalo do cálculo ele soma essa área à área anterior até esse intervalo. Para o caso de estudo, o cálculo da velocidade e do deslocamento foi feita utilizando o método "*cumtrapz*".

Para além da biblioteca "*scipy*", foram ainda necessárias as bibliotecas "*math*", "*numpy*", "*plotly* e "*filterpy*", sendo que esta última só será abordada na secção seguinte, referente ao método de redução de ruído. Em relação às outras 3 bibliotecas apresentadas, a biblioteca "*math*"foi utilizada pois esta contém várias funções matemáticas que não está definidas em python, como as funções seno e coseno, que foram necessárias para obter os deslocamentos nos dois eixos x e y. A biblioteca "*numpy*"foi utilizada para a criação de "*nparrays*", onde a velocidade de acesso e técnicas de manipulação são superiores aos dos arrays pre definidos em python, tais como listas ou tuples. Finalmente, a biblioteca "*plotly* foi utilizada para a visualização gráfica dos parâmetros da aceleração, velocidade e deslocamento.

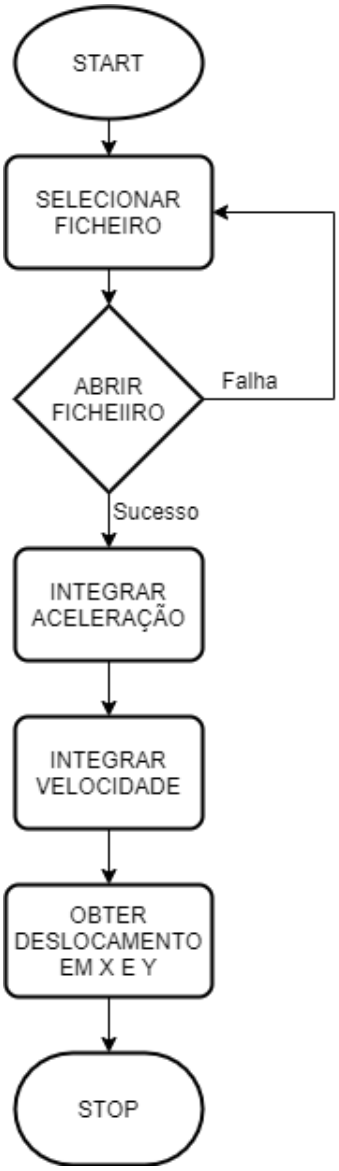


Figura 3.8: Fluxograma do algoritmo desenvolvido

O algoritmo desenvolvido está presente na listagem 1, não incluindo o código correspondente à visualização gráfica dos dados (aceleração, velocidade, deslocamento total, delocamento num plano 2D). O primeiro método presente ("*get_data_from_file(dir_str)*) corresponde à leitura de um ficheiro que contenha os dados produzidos pelo IMU. Este método deverá de ser adaptado conforme os dados selecionados a serem armazenados no ficheiro em questão, sendo que os dados recolhidos para esta iteração foram as três conmponentes da aceleração provenientes do acelerómetro (x, y e z), as três componentes da velocidade angular provenientes do giroscópio (x, y e z), e o "*yaw*", obtido diretamente pelo "*firmware*"do IMU utilizado. Apesar de serem retirados todos estes dados, os únicos dados necessários para a execução deste algoritmo correspondem apenas à aceleração na componente x, e o "*yaw*", sendo que este último é calculado utilizando os dados referentes

Algorithm 1 Algoritmo "*tracking*"(pseudo código)

```
function INTEGRATE(arr, Δt)
    intgrtd_arr ← CUMTRAPZ(arr, Δt)        ▷ integração de arr com intervalos de Δt
    return intgrtd_arr
end function

function TRACKING(filename)
    acc, yaw ← filedata                    ▷ Parse filedata em dois arrays acc e yaw
    vel ← INTEGRATE(acc, Δt)
    dis ← INTEGRATE(vel, Δt)
    M ← 0                                  ▷ Deslocamento total anterior
    A, B ← 0                               ▷ Deslocamento anterior em x e em y
    for N em dis do
        Δx ← A + (N − M) × cos(rad_yaw)    ▷ cálculo do deslocamento em x
        Δy ← B + (N − M) × sin(rad_yaw)    ▷ cálculo do deslocamento em y
        A ← Δx
        B ← Δy
    end for
end function
```

à velocidade angular, no próprio "*firmware*".

O segundo método que está presente no algoritmo corresponde ao método de integração. Este método apenas envia um "*array*"a ser integrado, como o "*array*"da aceleração e da velocidade, e retorna um "*array*"com a integração calculada, sendo estes "*arrays*"da velocidade e do deslocamento, respetivamente.

Finalmente, a última parte do algoritmo corresponde ao cálculo dos deslocamentos nas suas respetivas componentes (x e y). Para este cálculo é necessário então a informação do deslocamento anterior de cada componente ("*x_pos[-1]*"e "*y_pos[-1]*"), e somar o deslocamento que ocorreu nessa iteração, onde "*disp*"corresponde ao deslocamento total à iteração atual e "*disp_ant*"corresponde ao deslocamento total até à iteração anterior. Para obter a orientação, isto é, o deslocamento nas componentes x e y, é necessário multiplicar o valor do deslocamento pelo coseno se for para a componente em x, e seno se for para a compenente em y.

Todos os testes e respetivos resultados feitos serão apresentados no capítulo 4, que corresponde ao capítulo de validação.

3.5 Método de redução de ruído

O ruído é um factor que está sempre presente, principalmente quando se converte realidade em informação digital, como é o caso de sensores. Os IMUs não são excepção, sendo que estes são bastante afetados por ruído, como é possível observar na imagem 3.5 na página 39, onde para além de o sensor estar constantemente a oscilar, também é possível observar um incremento na aceleração, quando esta deveria de ser constante. Este ruído

afeta negativamente as medições obtidas pelo sensor, alterando assim o comportamento para algo não desejado. Como tal, é necessário utilizar métodos que consigam minimizar o impacto do ruído nas medições.

Os filtros são formas de reduzir esse ruído que esteja presente nas medições, sendo que existem vários tipos de filtros, cada um apto para uma dada situação.

Durante o desenvolvimento desta tese foram aplicados 3 tipos de filtro, sendo eles um filtro de "*Butterworth*", que será apresentado na secção 3.5.1, um filtro elimina-banda, presente na secção 3.5.2, e um filtro de Kalman, que será apresentado na secção 3.5.3.

3.5.1 Filtro Butterworth

A implementação do filtro "*Butterworth*"[35] será uma implementação com o objetivo de eliminar o ruído que corresponde às oscilações que se podem observar na imagem 3.5 que se encontra na página 39. Como tal, este filtro corresponderá a um filtro passa baixo, onde o objetivo é eliminar as oscilações que sejam demasiado rápidas para corresponder à realidade.

A implementação do filtro em si já se encontra desenvolvida numa biblioteca python, que corresponde à "*scipy.signal*". Esta biblioteca está descrita no início da subsecção 3.4.3, sendo que para o filtro é utilizado uma sub biblioteca que corresponde ao "*signal*". Esta possui métodos que estejam relacionados com o processamento e análise de sinais, tais como filtros que estão previamente desenvolvidos, como é para o caso do filtro de "*Butterworth*". A escolha dos métodos que são necessários para esta implementação e que estão presentes na biblioteca "*signal*"são o método "*butter*", que corresponde ao método de criação do filtro com os parâmetros desejados, e o "*lfilter*", que será o método que irá aplicar o filtro desenvolvido ao conjunto de dados a serem processados.

A implementação deste filtro está presente em 2, em pseudo código. O código original encontra-se no anexo III

Nesta implementação, os parâmetros que deverão de ser alterados correspondem ao parâmetro da "*order*", que indica qual a ordem do filtro desejada, a "*fs*", que correspondem à frequência de amostragem ao qual os nossos dados foram amostrados, e finalmente o parâmetro "*cutoff*", que correspondem à frequência de corte, isto é, a partir de que ponto é que os dados são "filtrados"pelo filtro, sendo que todos os dados que estejam abaixo dessa frequência mantêm-se inalterados.

A validação correspondente ao filtro de "*Butterworth*"está presente no capítulo seguinte, que corresponde ao capítulo de validação do trabalho desenvolvido no âmbito desta tese.

3.5.2 Filtro elimina banda

A implementação do filtro elimina banda [45] tem como objetivo eliminar o ruído associado a oscilação constante que se pode observar em 3.5. Sabendo que a teoria diz que o sensor, quando este se encontra estacionário, deverá de indicar uma aceleração nula,

Algorithm 2 Algoritmo "*Butterworth*"(pseudo código)

```
function FilterData(Y)
    O ← ordem filtro                                   ▷ Parâmetros do filtro
    F ← frequência amostragem
    C ← frequência corte
    Filt ← butter_lowpass_filter(Y, C, F, O)
    return Filt
end function

function butter_lowpass(C, F, O)
    Nyq ← 0.5 × F                                      ▷ Frequência de Nyquist
    NC ← C / Nyq                                       ▷ Normalização da frequência de corte
    A, B ← init_butterworth                            ▷ Inicialização do filtro Butterworth
    return A, B
end function

function butter_lowpass(Y, C, F, O)
    B, A ← butterworth(C, F, O)
    Filt ← lfilter(B, A, Y)                            ▷ Método que aplica o filtro aos dados
    return Filt
end function
```

então o objetivo deste filtro será tentar eliminar esse ruído. Este tipo de filtro irá apenas eliminar o ruído que esteja presente a volta do eixo horizontal, isto é, o eixo do x, sendo que este atuará até aos limites superiores e inferiores pre estabelecidos. Este filtro então apenas atuará sobre os dados que cumpram esses requisitos, enquanto que o filtro apresentado na subsecção anterior (3.5.1) atua sobre a totalidade dos dados.

A sua implementação, também feita em python, corresponde apenas em definir um intervalo de valores entre os quais o valor que o IMU fornece é na realidade zero. O fluxograma referente a este filtro está presente na figura 3.9.

Tal como para o filtro de "*Butterworth*", os testes e análise de resultados referentes a este filtro estão presentes no capítulo 4.

3.5.3 Filtro de Kalman

O último tipo de filtro implementado corresponde ao filtro de Kalman, mais precisamente, o "*Unscented Kalman Filter*". Este tipo de filtro, como já foi referido no capítulo 2, mais precisamente na secção 2.3 presente na página 24, funciona para o caso onde o sistema em questão não possui um comportamento linear. A razão para o uso deste filtro deve-se ao facto de que este filtro é bastante utilizado em sistemas de "*tracking*"que não utilizam GPS, mas que fazem uso de tecnologias como o IMU, o que corresponde ao cenário proposto nesta tese.

Para esta implementação é necessário seguir os passos apresentados no capítulo 2, na secção 2.3, mais precisamente na subsecção referente ao filtro de Kalman (2.3.2),

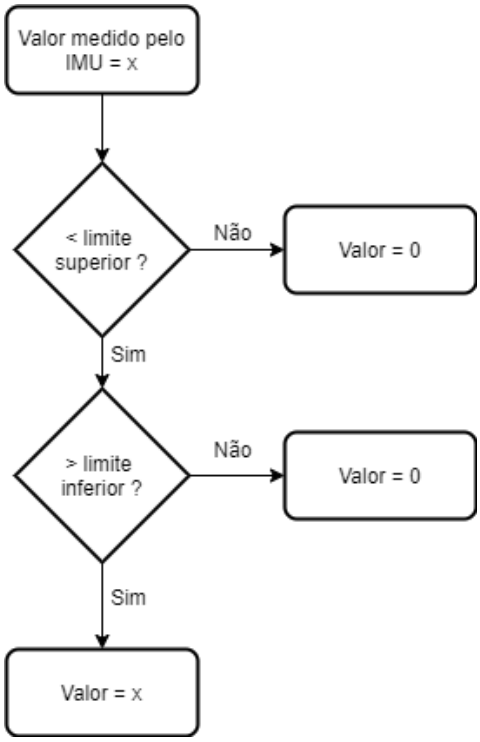


Figura 3.9: Fluxograma correspondente ao funcionamento do filtro elimina banda

sendo que o primeiro passo corresponde à escolha das variáveis de estado. Estas variáveis correspondem não só aos dados recebidos pelos sensores, sendo eles a aceleração e o "*yaw*"(ângulo de orientação), mas também variáveis obtidas através do uso das equações do movimento (3.2) apresentadas na subsecção 3.4.1, sendo elas a velocidade e o deslocamento. Para o caso do deslocamento, este vai estar dividido em dois deslocamentos, um para cada eixo do plano 2D (x, y). A sua representação no vetor **x**, que corresponde ao vector de variáveis de estado, será,

$$\mathbf{x} = \begin{bmatrix} \Delta dx & \Delta dy & v & a & \theta \end{bmatrix}^T \tag{3.4}$$

onde Δdx e Δdy correspondem à variação do deslocamento em cada um dos eixos em cada estado. A variável v corresponde à velocidade sobre o eixo x do IMU e a variável a corresponde à aceleração sobre o mesmo eixo x do IMU. É importante notar que o eixo x do IMU não corresponde ao mesmo eixo x do deslocamento Δdx. Finalmente a variável θ correspondem ao "*yaw*"que é retirado do IMU.

O segundo passo para o desenvolvimento do "*Unscented Kalman Filter*"corresponde ao dimensionamento da matriz de função de transição. Para o caso desta iteração do filtro, o desenvolvimento desta matriz é substituído apenas pela função de transição. O objetivo é obter uma transição de fase desta forma,

$$\bar{x} = x + f(x) + \mathcal{N}(0, Q) \tag{3.5}$$

onde $f(x)$ corresponde a função de transição de estado a ser desenvolvida. $\bar{x}$ e x representam apenas a forma não matricial dos vetores das variáveis de estado, sendo que estes representam o estado atual e o estado anterior, respetivamente. A última parte da equação ($\mathcal{N}(0,Q)$) representa o ruído de processo, sendo que este é assumido como um ruído com uma distribuição Gaussiana de média zero e de variância Q, sendo que a variância ainda será parametrizada no passo seguinte ao atual. A função de transição será composta por todas as equações que, a cada respetiva variável de estado, causem que estas possam mudar para outro estado. Como tal, estas equações são

$$\begin{cases} \Delta dx_k &= \Delta dx_{k-1} &+ (\Delta d \times \cos\theta_{k-1}) \\ \Delta dy_k &= \Delta dy_{k-1} &+ (\Delta d \times \sin\theta_{k-1}) \\ v_k &= v_{k-1} &+ \Delta t \times a_{k-1} \\ a_k &= a_{k-1} & \\ \theta_k &= \theta_{k-1} & \end{cases} \tag{3.6}$$

onde a função de transferência corresponde aos elementos que somam às variáveis de estado, isto é, correspondem a $\Delta d \times \cos\theta_{k-1}$ e $\Delta d \times \sin\theta_{k-1}$ para as variáveis Δdx_k e Δdy_k, respetivamente, e $\Delta t \times a_{k-1}$ para a variável v_k. Como a aceleração e o "*yaw*"correspondem a elementos que provêm do IMU, a sua atualização de estado para estado estará ligado à leitura dos respetivos valores dos sensores, sendo que para esta fase a sua respetiva função de transição de estado corresponde a 1.

O passo seguinte do desenvolvimento do filtro de Kalman corresponde ao dimensionamento da matriz de ruído de processo, matriz Q, mencionada na expressão 2.15. Para a escolha dos valores foi aplicado a teoria presente no livro interactivo [41], mais precisamente no capítulo referente à matemática do filtro de Kalman, onde o autor refere que uma simplificação possível que se pode efetuar no dimensionamento da matriz Q corresponde a colocar todos os elementos a zero excepto o elemento que varia mais rapidamente. Aplicando esta teoria e com algum ajuste, o dimensionamento para o ruído do processo será

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0.005 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

A função de converção de medidas, isto é, a função **H**, corresponde à forma de como é feita a passagem das variáveis de estado, presentes no vector **x**, para as medidas provenientes dos sensores, através da equação

$$\mathbf{z} = \mathbf{H}\mathbf{x} \tag{3.8}$$

Como as variáveis que correspondem às medidas dos sensores são a aceleração e o "*yaw*", a matriz **z** corresponderá a um vetor contendo $\begin{bmatrix} a & \theta \end{bmatrix}^T$, que tem dimensão 2×1.

Como o vector de variáveis de estado possui uma dimensão de 5×1, a matriz de função de conversão das medidas terá que ter uma dimensão de 2×5. Isto deve-se pois, numa multiplicação entre duas matrizes, **A** e **B**, com dimensões $M \times N$ e $N \times P$, respetivamente, o produto corresponderá a uma outra matriz **C** com dimensão $M \times P$. Sabendo que a matriz **z** corresponde à matriz **C**, onde $M \times P$ será 2×1, e a matriz **B** corresponde à matriz **x**, onde $N \times P$ correspondem a 5×1, então a matriz **H** terá a dimensão da matriz **A**, onde para $M \times N$, M será 2 e N será 5. Para os valores que estarão presentes desta matriz, visto que **z** foi definida como $\begin{bmatrix} a & \theta \end{bmatrix}^T$, e **x** corresponde a $\mathbf{x} = \begin{bmatrix} \Delta dx & \Delta dy & v & a & \theta \end{bmatrix}^T$, e como não é necessário efectuar nenhuma conversão de unidades (metros para milhas por exemplo), então a matriz **H** será

$$\mathbf{H} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

A última matriz que é necessário dimensionar corresponde à matriz **R** que será a matriz de ruído associado às medidas dos sensores. Esta matriz tem dimensão $m \times m$, onde m será o número de medidas que são retiradas dos sensores. Como tal, **R** será uma matriz 2×2, e o conteúdo dessa matriz será zero excepto na diagonal principal, onde esta deverá conter as variâncias associado aos sensores das respetivas medidas. É importante referir que estes ruídos são todos considerados aproximações de distribuições Gaussianas, como e referido no livro utilizado para o dimensionamento [41]. Visto que o ruído não corresponde a uma distribuição Gaussiana perfeita, foi necessário obter uma apróximação para estes valores de variância.

Em relação à variância associada ao sensore de acerelação, esta foi obtida através da aproximação entre a distribuição dos dados referentes a um teste onde o sensor se encontrava parado, e uma distribuição normal com os parâmetros de variância e média. A razão para este teste ser necessário o sensor estar estacionário deve-se ao facto de que o ruído é caracterizado, para o desenvolvimento deste filtro, como uma distribuição Gaussiana com média igual a zero e variância não nula, como está apresentado na equação de transição de estado 3.5. A figura 3.10 representa esta apróximação, onde uma distribuição normal com os parâmetros $\mu = 0$ e $\sigma^2 = 0.017^2$, sendo que μ corresponde à média e σ^2 representa a variância, foi aproximada aos dados referentes do teste em questão.

O segundo elemento que é recebido do IMU corresponde aos dados do "*yaw*". Para estes dados, a aproximação feita para a aceleração não é possível visto que o "*yaw*"não é fornecido por um sensor, mas sim é um dado que é calculado pelo "*firmware*"do próprio IMU, onde este utiliza informação de mais do que um sensor. Isto altera o comportamento do ruído, impossibilitando a sua aproximação a uma distribuição normal. É possível observar na figura 3.11, que corresponde ao mesmo teste do que foi feito para a aceleração, onde a informação retirada foi a do "*yaw*", que é impossível aproximar estes dados a uma distribuição normal. Como não é possível definir um valor para a variância do "*yaw*"que aproxime os dados obtidos a uma distribuição normal, então foi necessário obter este valor de outra forma. A definição de σ^2 corresponde à variação máxima dos valores que

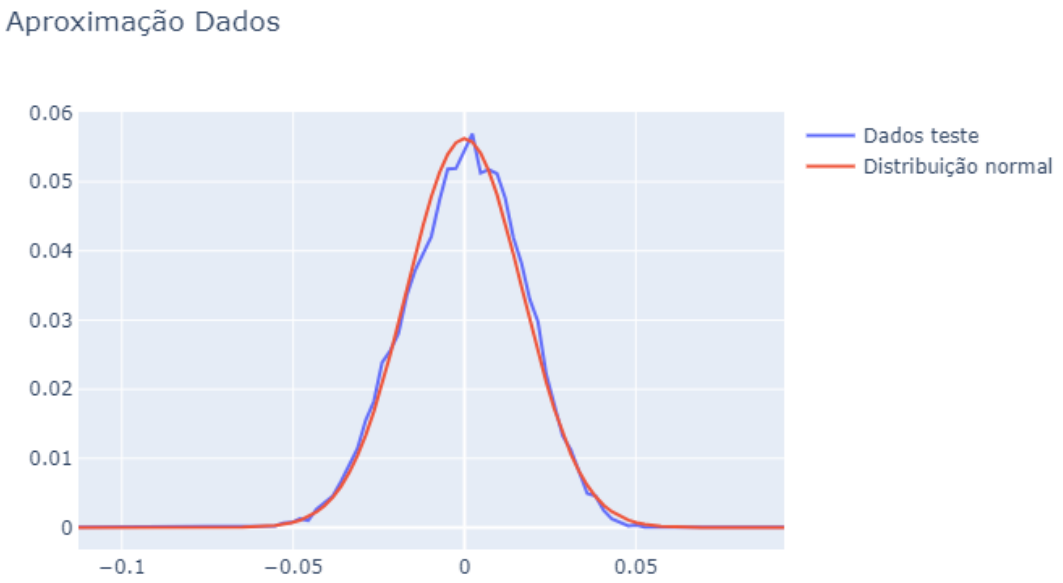


Figura 3.10: Representação gráfica da aproximação dos dados da aceleração do IMU a uma distribuição normal

o sensor pode medir em relação à média, isto é, a diferença diferença entre a média e o máximo e o mínimo corresponderá à variância. Ao efetuar esta análise, uma variância possível para o "*yaw*"será $\sigma^2 \approx 0.57$

A matriz **R** poderá ser formalizada e esta estará definida da seguinte forma

$$\mathbf{R} = \begin{bmatrix} 0.017^2 & 0 \\ 0 & 0.75^2 \end{bmatrix} \tag{3.10}$$

onde $0.75^2 \approx 0.57$.

Finalmente o último passo para o desenvolvimento do "*Unscented Kalman Filter*"corresponde a definir um estado inicial para o sistema, isto é, introduzir valores de estado na matriz **x** que representem o sistema no inicio da execução do filtro. Seguinto as recomendações de [41], o recomendado para a definição destes valores corresponde a inicializar todos os valores a zero, excepto aqueles que são adquiridos pelos sensores, onde estes deverão de ser iguais à primeira medição obtida pelos mesmos. Utilizando esta abordagem, então os valores iniciais para **x** corresponderão a

$$\mathbf{x} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ acc\_data[0] \\ yaw\_data[0] \end{bmatrix} \tag{3.11}$$

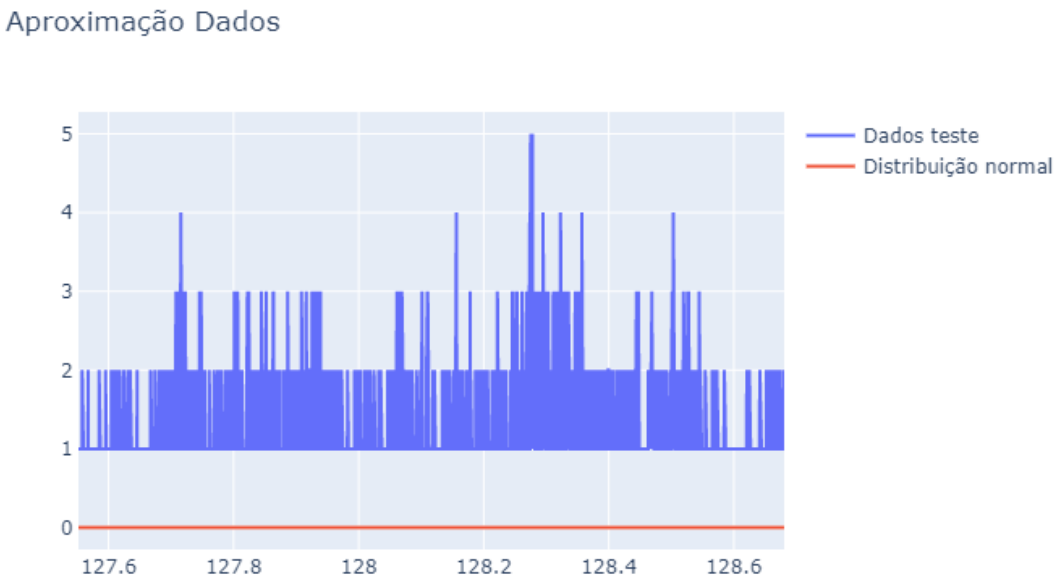


Figura 3.11: Representação gráfica da tentativa de aproximação dos dados do "*yaw*"do IMU a uma distribuição normal

onde os primeiros três valores inicializados a zero correspondem, respetivamente, a Δdx, Δdy e a v, e os últimos dois valores, que representam a e θ, são inicializados com o primeiro valor recebido pelo IMU.

Os resultados obtidos, bem como a sua análise, estão presentes no capítulo 4, que contém toda a validação do trabalho referente a esta tese.

3.6 Algoritmo de transmissão de dados - OSC

Apesar de todo o trabalho desenvolvido, incluindo os testes feitos, estarem ligados a uma vertente offline, isto é, os dados são primeiramente recolhidos, guardados, e só depois analisados, o objetivo final será o uso de um sistema online, em tempo real, de modo a obter a localização dos AGVs atualizada constantemente durante os processos de transporte dos mesmos. Como tal, foi desenvolvido, no âmbito desta tese, a fase inicial do algoritmo que é capaz de receber informação em tempo real proveniente do IMU. Estes dados ficam depois disponíveis para manipulação.

O algoritmo desenvolvido utiliza o protocolo OSC ("*OpenSound Control*") [46], que corresponde a uma forma de enviar mensagens na forma de pacotes, numa rede. Apesar deste algoritmo ser bastante utilizado na indústria da música[43, 47, 48], este tipo de formatação de mensagens também começou a ser utilizado em aplicações de IoT ("*Internet of Things*"), como por exemplo em [49], pois este tipo de protocolo de mensagem, como já foi apresentado na secção 3.3.2, é independente do tipo de dados a serem enviados, o que

corresponde a uma mais valia em cenários de IoT, onde vários tipos de sensores podem ter tipos de dados diferentes. Isto facilita a integração de sistemas, que corresponde a um grande objetivo do IoT e da Indústria 4.0.

Presente na listagem 3 encontra-se o pseudo código referente ao algoritmo desenvolvido para a comunicação entre o IMU e um servidor, sendo que este último neste caso corresponde a um computador. No anexo II está disponível o código desenvolvido. Este código foi desenvolvido utilizando as especificações do protoco "*OpenSound Control*", sendo que as especificações da comunicação utilizadas estão presentes em [46].

Algorithm 3 Algoritmo do servidor para receber dados do IMU (pseudo código)

```
function PRINT_VALUES(A, L)
    val_acc = L[0]      ▷ Passar o valor da aceleração da lista L para a variável val_acc
    val_yaw = L[5]      ▷ Passar o valor do yaw da lista L para a variável val_yaw
    Printval_acc
    Printval_yaw
end function

function TRACKING(filename)
    arg = IP, port      ▷ Definir parâmetros de IP e porto a serem utilizadados
    func = Print_values ▷ Definir função a ser chamada quando o servidor detectar
informação do IMU
    Server = arg, func  ▷ Inicializar o servidor com os parâmetros definidos
end function
```

O algortimo desenvolvido contém uma função chamada "*Print_values*", onde os valores que são recebidos pelo IMU são passados para duas variáveis locais, sendo elas "*val_acc*"e "*val_yaw*", que contêm o0s valores da aceleração e do yaw respetivamente. Estes valores ficam então disponíveis para manipulação, onde neste caso são apenas impressos na consola. Todos os valores disponíveis pelo IMU encontram-se na lista L, atualizada constantemente a cada 5 milisegundos (devido à taxa de atualização de 200 Hz do próprio IMU). Para aceder aos valores da aceleração, é necessário ir buscar os valores que estão no índice 0 da lista, e os valores do yaw estão presentes no índice 5 da mesma lista. A informação referente aos dados recebidos do IMU foi retirada da análise feita ao firmware utilizado pelo prórpio IMU, presente no repositório [50].

Capítulo 4

Resultados experimentais

Serve o presente capítulo para apresentar os testes feitos utilizando os métodos descritos no capítulo anterior, bem como uma análise dos mesmos, fornecendo assim uma validação sobre o trabalho feito no âmbito desta tese. A secção 4.2.1 contém os resultados referentes aos testes feitos utilizando o método descrito em 3.4.3 (algoritmo de primitivação pela regra do trapézio). As secções 4.2.2, 4.2.3 apresenta os resultados obtidos pelos testes efectuados aos métodos apresentados em 3.5, mais precisamente os métodos presentes em 3.5.1 e em 3.5.2, referentes ao filtro de Butterworth e ao filtro passa baixo, respetivamente. Finalmente, a secção 4.2.4 contém a validação obtida ao último método apresentado no capítulo 3, sendo ele o método do uso de um Filtro de Kalman (*"unscented"*), presente em 3.5.3.

4.1 Configuração experimental

A configuração experimental consiste no uso de um carrinho, de modo a emular os AGVs que foram descritos no decorrer do primeiro capítulo. A razão pela qual é utilizado um carrinho em vez dos AGVs em questão, presentes na fábrica da Volkswagen Autoeuropa, deve-se ao facto de que estes AGVs estão todos a ser utilizados nos processos de produção da fábrica, e isso implicaria parar essa mesma produção de modo a realizar-se as experiências para os casos apresentados no capítulo anterior. Este carrinho não possui motor, o que implica uma acção humana para que haja movimento. O IMU é colocado sobre a zona frontal do carrinho, onde a orientação do sensor deverá de corresponder com a orientação do carrinho, como está exemplificado na imagem 3.6 presente na página 40. Para a execução dos testes, foi utilizado uma fita métrica de modo a poder ter uma referência em relação à distância percorrida durante o mesmo. A figura 4.1 contêm a configuração experimental completa utilizada no decorrer dos testes.

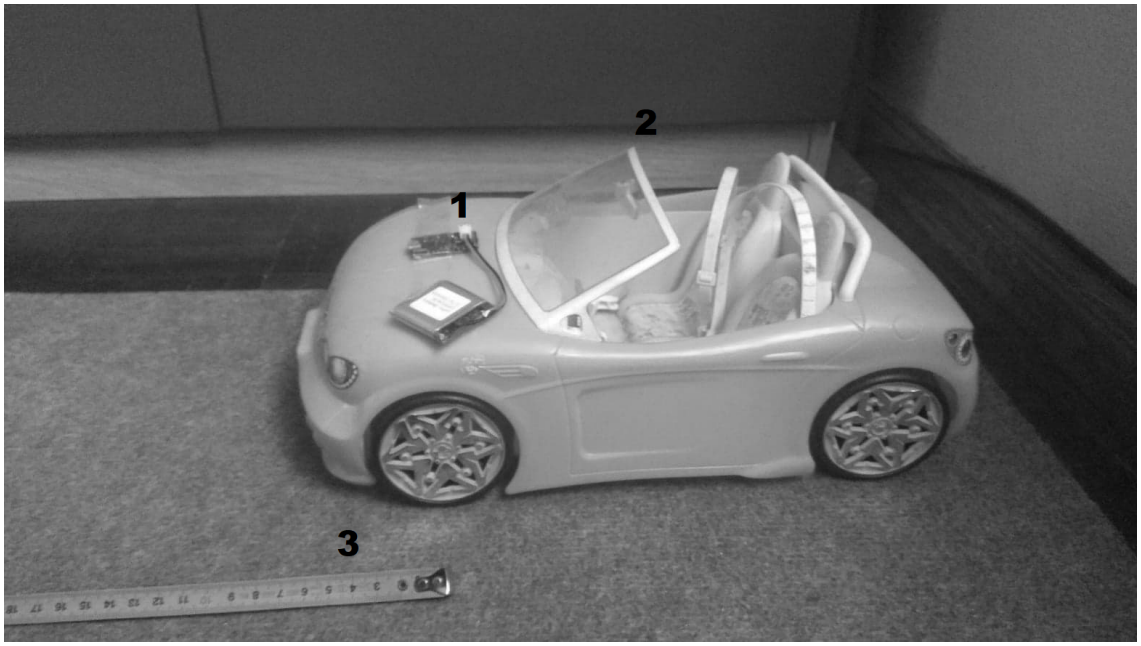


Figura 4.1: Montagem utilizada para o procedimento experimental. O IMU (1) encontra-se na zona frontal do carrinho (2). Foi utilizado também uma fita métrica (3) para a medição do percurso a ser percorrido durante o teste.

4.2 Resultados

Esta secção contém os resultados obtidos para os testes feitos ao algoritmo de "*tracking*", bem como os resultados da aplicação dos filtros descritos no capítulo 3, estando dividido em duas subsecções.

Os testes consistem em percorrer várias distâncias pré definidas, de 1 m, 2 m, e 3 m, onde o percurso corresponde a uma linha reta, como forma de validação da componente do acelerómetro. Outro teste realizado corresponde à realização de uma curva de 90º após um deslocamento de 1 m, de modo a verificar e validar a componente de localização que corresponde ao giroscópio.

4.2.1 Algoritmo de *tracking*

O objetivo destes resultados é demonstrar a eficácia que o algoritmo de "*tracking*"apresenta. A sua precisão irá definir se este algoritmo pode servir como possível solução para o desafio proposto por esta tese, referente ao "*tracking*"de AGVs que não possuem capacidades de localização tradicionais (GPS). Estes resultados estão divididos em duas categorias, sendo elas o resultado para o teste de deslocamento em linha reta, e o deslocamento com curva, de modo a testar os componentes da aceleração linear e da velocidade angular. Estes testes visam a substituir os testes originais que iriam ser feitos na fábrica da Autoeuropa em Palmela com os AGVs em questão, sendo que estes apresentam algumas limitações em relação ao tipo de teste, ambiente de teste, e equipamento utilizado para a sua realização.

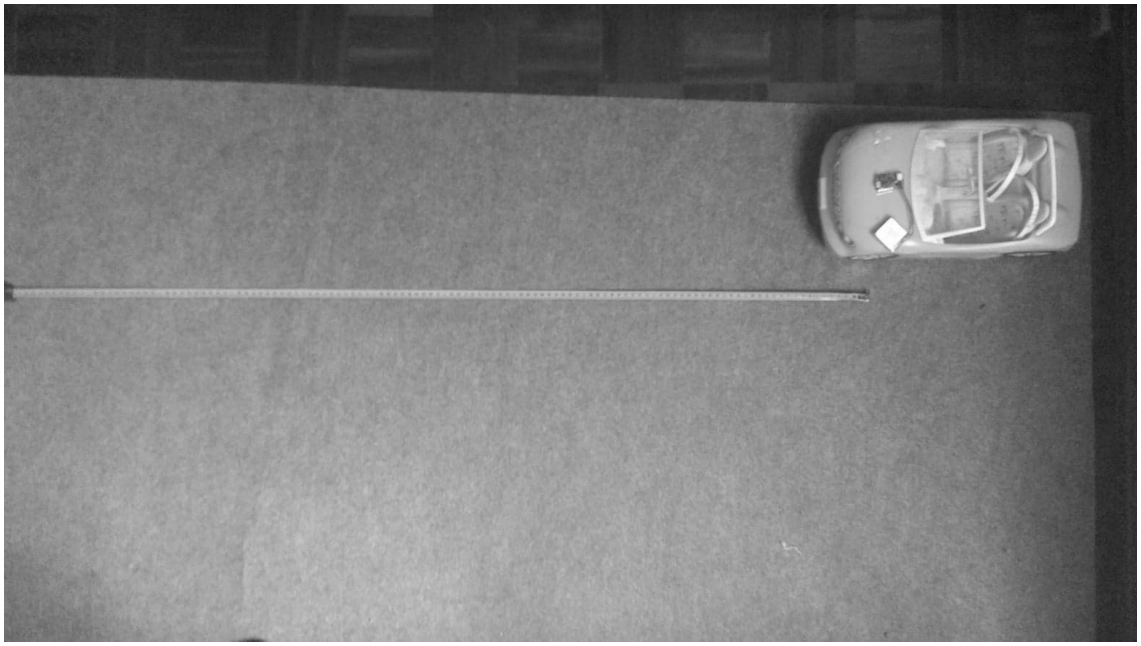


Figura 4.2: Procedimento para a execução de um teste de deslocação em linha reta de 1 m.

4.2.1.1 Linha reta

O método referente aos testes em linha reta está presente na imagem 4.2. Estes testes foram realizados utilizando um carrinho, sendo que este não possuí motor nem nenhum mecanismo de arranque, onde o impulso para o arranque foi fornecido pela aplicação de um força externa sobre o carrinho. Os resultados referentes aos testes estão presentes nas tabelas 4.1, 4.2 e 4.3, sendo que estas correspondem aos testes de 1 m, 2 m e de 3 m.

Tabela 4.1: Tabela com os resultados para os testes referentes a andar em linha reta (1 m)

#	Deslocamento Teórico (m)	Deslocamento Teste (m)	Erro (%)
1	1	0.75	25%
2	1	0.94	6%
3	1	0.90	10%
4	1	1.04	4%
5	1	0.90	10%

É possível observar que os resultados serão piores quanto maior for a distância percorrida durante os testes. É importante referir que algumas distâncias percorridas presentes na tabela 4.3 estão negativas devido à orientação associada pelo sensor, isto é, uma aceleração negativa corresponde a uma aceleração para trás e uma aceleração positiva corresponde a uma aceleração para a frente. Estas acelerações irão corresponder a velocidades, e estas velocidades irão corresponder a distâncias, onde a relação que estas têm com o sinal é igual à relação que a aceleração tem com o sinal. Deste modo, uma deslocação negativa corresponde a uma deslocação para trás desde a origem. Em relação aos testes,

Tabela 4.2: Tabela com os resultados para os testes referentes a andar em linha reta (2 m)

#	Deslocamento Teórico (m)	Deslocamento Teste (m)	Erro (%)
1	2	1.84	8%
2	2	1.84	8%
3	2	2.32	16%
4	2	2.45	23%
5	2	2.33	16%
6	2	2.34	17%
7	2	2.28	14%

Tabela 4.3: Tabela com os resultados para os testes referentes a andar em linha reta (3 m)

#	Deslocamento Teórico (m)	Deslocamento Teste (m)	Erro (%)
1	3	-1.77	159%
2	3	-0.79	126%
3	3	0.60	80%
4	3	1.02	66%
5	3	-2.03	167%

como já foi referido, estes apresentam resultados piores quanto maior for a distância de deslocamento, sendo que para os testes referentes ao deslocamento de 3 m estes chegam a apresentar erros de 159%, como é o caso no testes nº 1, onde o deslocamento final obtido pelo algoritmo corresponde a -1.77 m, isto é, o algoritmo calculou um deslocamento para trás, sendo que a distância entre o ponto final do teste e o deslocamento teórico previsto (3 m) é de 4.77 m, apresentando assim um erro de 159%.

Estes testes comprovam o estudo feito no capítulo 2, onde foi apresentado que os IMUs tendem a acumular ruído durante o seu funcionamento, devido a erros de leitura [15], como é possível observar no aumento do erro consuante o aumento da distância percorrida, impossibilitando o seu uso em cenários de *tracking* sem o auxílio de técnicas de processamento de dados, tais como a utilização de filtros.

4.2.1.2 Curva

O método referente aos testes da curva é igual ao método utilizado para a linha reta, onde a única alteração está no percurso efetuado pelo carrinho, sendo que para este teste o percurso corresponde a uma linha reta inicial de 1 m, seguido de uma curva com um ângulo de rotação de 90º, e a fase final corresponde a outro percurso em linha reta de 1 m. Para a avaliação do desempenho neste teste, é considerado o deslocamento efetuado, tal como no teste em linha reta, e a diferença entre o ângulo inicial e o ângulo final do deslocamento. Para os testes serem perfeitos, os resultados têm de obter um deslocamento de 2 m (positivo devido à explicação feita na subsecção anterior) e a diferença do ângulos deverá de ser igual a 90º.

Os resultados obtidos nos testes efetuados estão presentes na tabela 4.4, onde as

Tabela 4.4: Tabela com os resultados para os testes referentes a uma curva de 90º, sendo que antes e depois de efetuar a curva é feito um deslocamento de 1 m.

#	Deslocamento Teórico Linear (m)	Deslocamento Teste (m)	Erro (%)	Rotação Teórica (º)	Rotação Teste (º)	Erro (%)
1	2	1.5	25%	90	115	28%
2	2	-0.64	132%	90	112	24%
3	2	-0.55	128%	90	114	27%
4	2	-1.97	199%	90	113	26%
5	2	-0.80	140%	90	115	28%
6	2	-1.5	175%	90	109	21%
7	2	0.49	76%	90	113	25%

primeiras 3 colunas correspondem ao deslocamento linear, e as últimas colunas correspondem à rotação do corpo. Em relação ao deslocamento linear, e tendo em conta os resultados dos testes em linha reta, é possível observar uma falta de precisão e exatidão em relação ao cálculo do deslocamento, como era esperado. No entanto, para a componente de rotação, é possível observar que as medições são de algum modo precisas, visto que os erros têm erros bastante constantes, no entanto não são muito precisas, visto que os erros associados encontram-se entre os 20% e os 30%, sendo que podem ser considerados demasiado elevados para a precisão necessária, associada ao cenário em questão, sendo este um cenário industrial, onde ocorre interações máquina entre máquina e máquina e ser humano.

A aplicação de filtros terá como objetivo principal reduzir ruído que possa estar a afetar negativamente a leitura/processamento dos dados referentes à execução deste algoritmo.

4.2.2 Filtro elimina banda

O primeiro método de processamento de dados para melhorar os resultados do algoritmo de "*tracking*"corresponde ao filtro elimina banda. A aplicação deste filtro terá como objetivo reduzir o impacto que o ruído tem durante momentos em que o veículo esteja parado e que o algoritmo "considere"que este se encontre em movimento. Como tal, este filtro é aplicado aos dados da aceleração obtidas pelo IMU, antes do algoritmo executar os cálculos da velocidade e de deslocamento.

Na figura 4.3 está representado um gráfico que contém os dados referentes à aceleração obtida durante teste nº 1 do deslocamento de 3 m, presente na tabela 4.3. É possível observar no gráfico destacado do original, que corresponde a uma ampliação do primeiro, que mesmo numa fase em que o veículo se encontra estacionário, o sensor oscila constantemente, devido ao ruído. Este ruído faz com que o algoritmo, ao calcular a velocidade a partir da aceleração, em vez de obter uma velocidade nula, esta aumenta a cada iteração. A figura 4.4 contém o mesmo gráfico da aceleração presente na imagem 4.3, contém o gráfico da velocidade obtida à custa da aceleração, e contém o gráfico do deslocamento

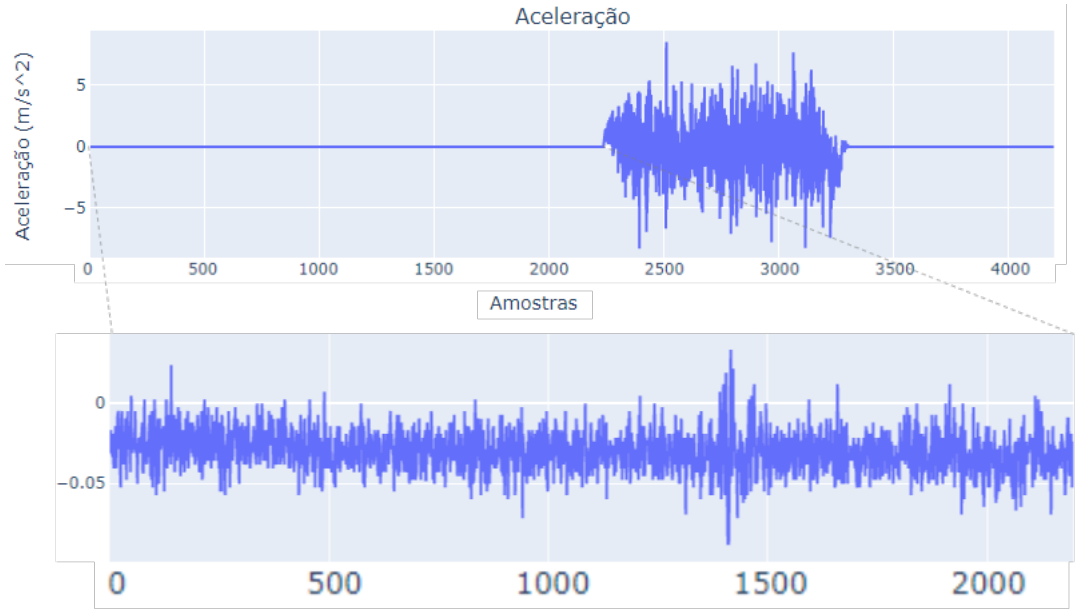


Figura 4.3: Dados obtidos durante o teste nº 1 referentes ao deslocamento de 3 m. O gráfico destacado corresponde a uma ampliação feita ao gráfico original, numa zona onde não ocorre nenhuma aceleração devido a movimento.

Tabela 4.5: Tabela com os resultados para os testes referentes a andar em linha reta (1 m) com um filtro elimina banda aplicado.

#	Deslocamento Teórico (m)	Deslocamento Teste (m)	Erro (%)
1	1	0.81	19%
2	1	1.13	13%
3	1	0.85	15%
4	1	0.94	6%
5	1	0.86	14%

obtido através da velocidade. É aparente a implicação que o ruído tem durante a execução do teste, sendo que este faz com que o algoritmo calcule movimento onde este não existe, obtendo assim um deslocamento errado.

As tabelas 4.5, 4.6 e 4.7 contêm os resultados do algoritmo, quando aplicado o filtro elimina banda aos mesmos dados que originaram as tabelas 4.1, 4.2 e 4.3. É possível observar, através da comparação entre as respetivas tabelas dos respetivos testes, que a melhoria mais significativa dos resultados está no teste que corresponde à linha reta de 3 m, onde os resultados que anteriormente chegavam a erros enormes, tais como para o teste nº 1 com um erro de 159%, passaram a ter erros bastante mais moderados, como é o caso de 21% para o teste respetivo. Isto prova uma melhoria na performance do algoritmo quando aplicado algum tipo de processamento de dados. Para o caso dos testes de 1 m e de 2 m, estes apresentam pequenas melhorias na maior parte dos casos, sendo que certos testes obtiveram resultados significativamente melhores em relação ao teste sem filtro, nomeadamente os testes nº 6 e 7 respetivos ao deslocamento de 2 m.

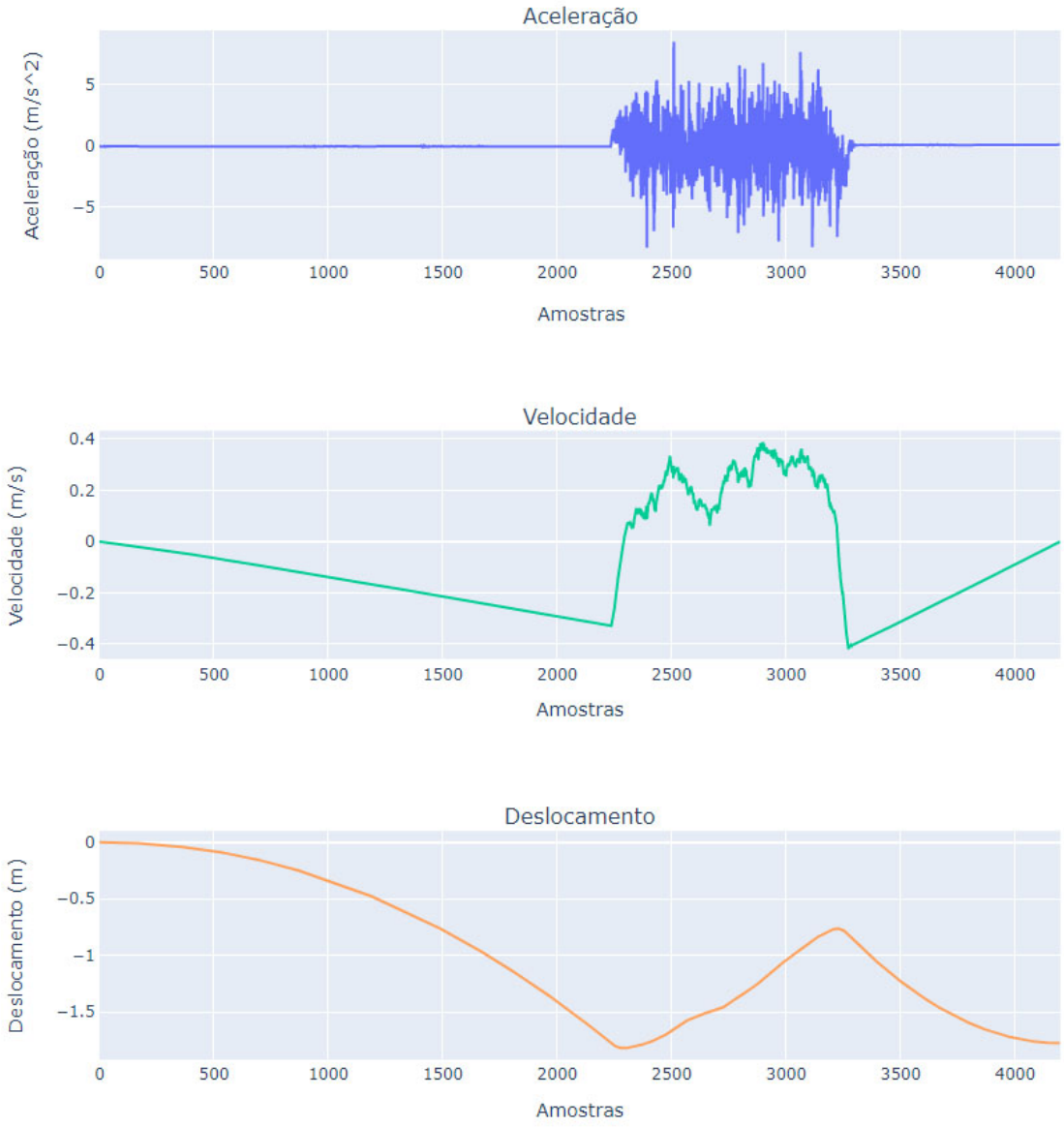


Figura 4.4: Gráficos referentes ao teste nº 1 do deslocamento de 3 m. O primeiro gráfico corresponde ao gráfico da aceleração. O segundo gráfico corresponde à velocidade obtida através da aceleração e o terceiro gráfico corresponde ao deslocamento obtido através da velocidade.

Tabela 4.6: Tabela com os resultados para os testes referentes a andar em linha reta (2 m) com um filtro elimina banda aplicado.

#	Deslocamento Teórico (m)	Deslocamento Teste (m)	Erro (%)
1	2	1.57	21%
2	2	1.67	16%
3	2	2.35	17%
4	2	2.65	33%
5	2	2.24	12%
6	2	2.10	5%
7	2	2.15	7%

Tabela 4.7: Tabela com os resultados para os testes referentes a andar em linha reta (3 m) com um filtro elimina banda aplicado.

#	Deslocamento Teórico (m)	Deslocamento Teste (m)	Erro (%)
1	2	2.69	10%
2	2	2.18	27%
3	2	2.09	30%
4	2	2.96	1%
5	2	4.16	39%

4.2.3 Filtro de Butterworth

A aplicação do filtro "*Butterworth*"foi feita para os dois testes feitos em 4.2.1, nomeadamente os testes de linha reta e curva. Em relação aos testes de linha reta, estes foram feitos para dois casos, sendo eles a aplicação de apenas este filtro, e da aplicação deste filtro no algoritmo juntamente com o filtro elimina banda. Como tal, os testes e os resultados obtidos estão divididos em 2 subsecções, referentes aos resultados dos testes em linha reta e para a curva. A aplicação deste filtro é igual à aplicação do filtro anterior, onde os dados da aceleração são processados pelo filtro, e depois o algoritmo executa os cálculos para a velocidade e para o deslocamento.

4.2.3.1 Linha reta

Os resultados referentes aos testes de linha reta, quando aplicado o filtro de "*Butterworth*", estão presentes nas tabelas 4.8, 4.9 e 4.10, onde estão os resultados quando aplicado o filtro ao algoritmo com e sem o filtro elimina banda.

Os resultados demonstram que, apenas comparando os resultados das várias distâncias entre si, os testes que envolviam o uso do filtro "*Butterworth*"sozinho apresentam resultados superiores, em termos de resultados mais baixos, excepto quando a distância de deslocamento corresponde a 3 m. Nesse caso, a utilização dos dois filtros, o de "*Butterworth*"e o elimina banda, apresentam resultados com erros inferiores. Aliás, os resultados para o teste de 3 m onde é aplicado os dois filtros são muito semelhantes aos resultados apresentados na tabela 4.7. Isto indica que este tipo de filtro ("*Butterworth*") tem um rendimento mais baixo quanto maior for a distância de deslocamento.

Tabela 4.8: Tabela com os resultados para os testes referentes a andar em linha reta (1 m) com o filtro "*Butterworth*"aplicado.

#	Deslocamento Teórico Linear (m)	Deslocamento Teste (m)	Erro (%)	Deslocamento Teste c/ Filtro EB (m)	Erro(%)
1	1	0.86	14%	0.81	19%
2	1	0.98	2%	1.12	12%
3	1	0.93	7%	0.85	15%
4	1	1.06	6%	0.94	6%
5	1	0.98	2%	0.86	14%

Tabela 4.9: Tabela com os resultados para os testes referentes a andar em linha reta (2 m) com o filtro "*Butterworth*"aplicado.

#	Deslocamento Teórico Linear (m)	Deslocamento Teste (m)	Erro (%)	Deslocamento Teste c/ Filtro EB (m)	Erro(%)
1	2	1.87	6%	1.57	21%
2	2	1.86	7%	1.68	16%
3	2	2.32	16%	2.33	17%
4	2	2.46	23%	2.64	32%
5	2	2.36	18%	2.24	12%
6	2	2.34	17%	2.10	5%
7	2	2.28	14%	2.15	7%

Tabela 4.10: Tabela com os resultados para os testes referentes a andar em linha reta (3 m) com o filtro "*Butterworth*"aplicado.

#	Deslocamento Teórico Linear (m)	Deslocamento Teste (m)	Erro (%)	Deslocamento Teste c/ Filtro EB (m)	Erro(%)
1	3	1.06	65%	2.69	10%
2	3	1.25	58%	2.18	27%
3	3	1.82	39%	2.09	30%
4	3	2.14	29%	2.96	1%
5	3	0.85	71%	4.13	38%

Deste modo, a aplicação deste filtro fica inválida numa situação como o cenário em causa, onde os deslocamentos facilmente chega as centenas de metros por dia, como é o caso da deslocação de AGVs que têm que fazer várias viagens entre pontos de recolha e entrega de equipamento/material.

4.2.3.2 Curva

Os resultados para o teste da curva, quando aplicado o filtro "*Butterworth*"estão presentes na tabela 4.11. Em relação a este teste, é possível observar que a aplicação deste filtro na componente de deslocamento angular ("*yaw*") não afeta o resultado final dessa mesma componente, visto que os resultados obtidos para este teste são iguais aos resultados presentes na tabela 4.4, onde não é utilizado nenhum filtro. Isto demonstra que para esta componente, não é necessário aplicar este tipo de filtro.

Tabela 4.11: Tabela com os resultados para os testes referentes a uma curva de 90º utilizando o filtro "*Butterworth*", sendo que antes e depois de efetuar a curva é feito um deslocamento de 1 m.

#	Deslocamento Teórico Linear (m)	Deslocamento Teste (m)	Erro (%)	Rotação Teórica (º)	Rotação Teste (º)	Erro(%)
1	2	1.35	33%	90	116	29%
2	2	1.12	44%	90	111	24%
3	2	1.19	41%	90	115	27%
4	2	1.49	25%	90	113	26%
5	2	1.32	34%	90	115	28%
6	2	1.20	34%	90	110	22%
7	2	1.17	42%	90	112	24%

Em relação ao deslocamento linear, apesar deste não ser a componente em foco neste teste, estes apresentam resultados bastante mais positivos em relação aos resultados da mesma tabela (4.4), visto que neste deslocamento foi aplicado os dois filtros utilizados neste subsecção e na subsecção anterior (filtro elimina banda e filtro "*Butterworth*".

4.2.4 Filtro de Kalman ("*unscented*")

O último filtro aplicado para a redução de ruído presente nas medições e no processo ligado ao algoritmo de "*tracking*"corresponde ao filtro de Kalman. Este filtro, ao contrário dos outros dois filtros apresentados, não é aplicado antes da execução do algortimo, mas sim durante a sua execução. Deste modo, foi necessário adaptar o algoritmo de modo a que este pudesse utilizar o filtro de Kalman durante a sua execução. No entanto, os resultados serão apresentados da mesma forma que os filtros anteriores.

4.2.4.1 Linha reta

Os resultados referentes ao uso do filtro de Kalman estão presentes nas tabelas 4.12, 4.13 e 4.14. Estes resultados consistem na utilização apenas do filtro de Kalman, sem o uso dos filtros apresentados anteiormente. Os resultados dos percursos em linha reta, comparando com os resultados presentes nas tabelas apresentadas na subsecção 4.2.1, apresentam melhorias significativas em alguns dos testes, nomeadamente nos testes do percurso em linha reta de 1 m, e nos dois primeiros testes do deslocamento de 2 m. Em relação ao deslocammento de 3 m, o filtro conseguiu reduzir bastante o erro presente nos dados, onde erros que chegavam a 150% foram reduzidos para erros na ordem dos 90%. No entanto, apesar da grande redução do erro presente, estes resultados não são positivos no contexto apresentado.

Estes resultados, como já foi referido, consistem no uso de apenas o filtro de Kalman aos dados disponibilizados pelo IMU. Uma tentativa de melhorar os dados, nomeadamente para o deslocamento de 3 m, foi a aplicação dos filtros anteriores aos dados provenientes do IMU, e aplicar o filtro de Kalman sobre esses mesmo dados. Os resultados

Tabela 4.12: Tabela com os resultados para os testes referentes a andar em linha reta (1 m) com o filtro Kalman aplicado.

#	Deslocamento Teórico (m)	Deslocamento Teste (m)	Erro (%)
1	1	0.83	17%
2	1	1.03	3%
3	1	0.99	1%
4	1	1.15	15%
5	1	0.99	1%

Tabela 4.13: Tabela com os resultados para os testes referentes a andar em linha reta (2 m) com o filtro Kalman aplicado.

#	Deslocamento Teórico (m)	Deslolamento Teste (m)	Erro (%)
1	2	1.92	4%
2	2	1.92	4%
3	2	2.44	22%
4	2	2.56	28%
5	2	2.47	23%
6	2	2.48	24%
7	2	2.42	21%

Tabela 4.14: Tabela com os resultados para os testes referentes a andar em linha reta (2 m) com o filtro Kalman aplicado.

#	Deslocamento Teórico (m)	Deslocamento Teste (m)	Erro (%)
1	3	0.07	98%
2	3	0.2	93%
3	3	-0.03	101%
4	3	-0.11	104%
5	3	0.20	93%

para este novo teste estão presentes nas tabelas 4.15, 4.16 e 4.17.

Estes novos resultados apresentam valores muito semelhantes para o deslocamento de 3 m, continuando a ter erros na ordem dos 90%. Em relação dos resultados para o deslocamento de 1 m e 2 m, estes não apresentam melhorias significativas. Isto demonstra que o filtro de Kalman não necessita de outros filtros para redução de ruído presente nos dados.

Tabela 4.15: Tabela com os resultados para os testes referentes a andar em linha reta (1 m) com o filtro Kalman aplicado (incluindo o filtro elimina banda e Butterworth).

#	Deslocamento Teórico (m)	Deslocamento Teste (m)	Erro (%)
1	1	0.83	17%
2	1	1.03	3%
3	1	0.99	1%
4	1	1.14	14%
5	1	0.99	1%

Tabela 4.16: Tabela com os resultados para os testes referentes a andar em linha reta (2 m) com o filtro Kalman aplicado (incluindo o filtro elimina banda e Butterworth).

#	Deslocamento Teórico (m)	Deslocamento Teste (m)	Erro (%)
1	2	1.93	3%
2	2	1.97	4%
3	2	2.45	23%
4	2	2.57	29%
5	2	2.48	24%
6	2	2.49	25%
7	2	2.43	22%

Tabela 4.17: Tabela com os resultados para os testes referentes a andar em linha reta (3 m) com o filtro Kalman aplicado (incluindo o filtro elimina banda e Butterworth).

#	Deslocamento Teórico (m)	Deslocamento Teste (m)	Erro (%)
1	3	0.07	98%
2	3	0.20	93%
3	3	-0.02	101%
4	3	-0.10	103%
5	3	0.21	93%

4.2.4.2 Curva

Os resultados sobre o teste da curva serão apresentados da mesma forma que os resultados para o teste da linha reta, onde a primeira tabela (4.18) contem os resultados para o teste onde só foi aplicado o filtro de Kalman, e a última tabela (4.19) contêm os resultados quando aplicado os filtros elimina banda e "*Butterworth*"com o filtro de Kalman.

É evidente que, em concordância com os testes feitos para a linha reta, que a aplicação do filtro de Kalman com os outros dois tipos de filtro não trás quaisquer benefícios em relação aos resultados obtidos. Isto implica que a utilização do filtro de Kalman é melhor quando este é aplicado sem o auxílio de outros. No entanto, os resultados da tabela 4.18 apresentam erros maiores em comparação aos que estão presentes na tabela 4.4, quando não é aplicado nenhum filtro. Isto poderá ser explicado por uma de duas razões. A primeira corresponde à forma de como é executado o teste, tendo em conta que toda o movimento do carrinho no contexto experimental foi feito com a intervenção humana, onde está incluído o movimento de fazer a curva. Este tipo de movimento introduz mais imprecisão e ruído durante o teste, visto que um movimento feito por um robô será sempre mais precisa do que uma curva feita por um ser humano. A figura 4.5 mostra que no início do movimento da curva, primeiro é detectado um movimento para o lado errado, como está representado.

A segunda razão, que poderá estar ligada aos erros maiores em relação à não utilização de qualquer filtro, poderá ser as especificações do filtro referentes ao "*Yaw*", que corresponde à componente que onde é medido a orientação do sensor.

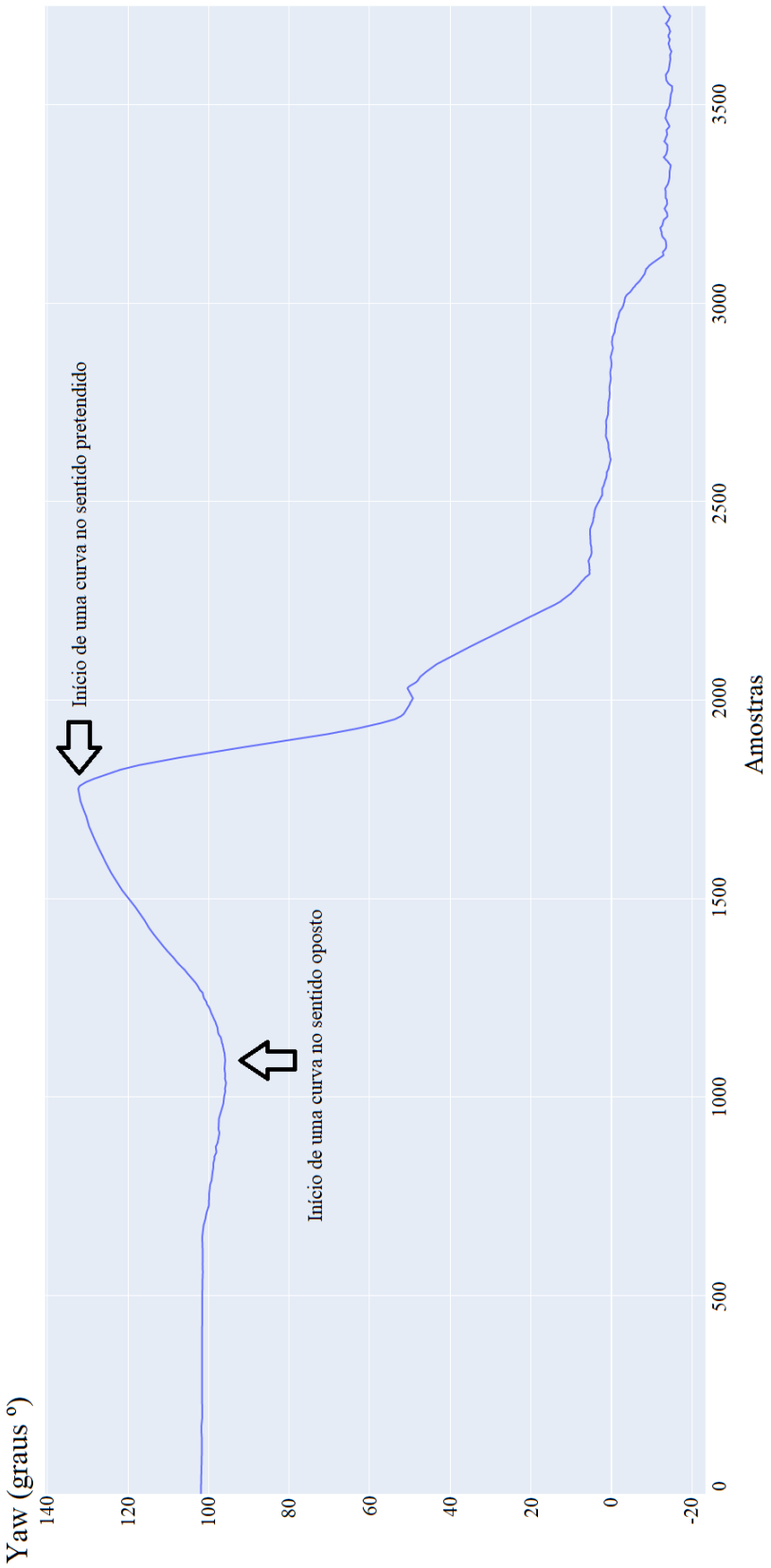


Figura 4.5: Gráfico do "*Yaw*"sem a aplicação de um filtro.

Tabela 4.18: Tabela com os resultados para os testes referentes a uma curva de 90º utilizando o filtro Kalman, sendo que antes e depois de efetuar a curva é feito um deslocamento de 1 m.

#	Deslocamento Teórico Linear (m)	Descolamento Teste (m)	Erro (%)	Rotação Teórica (º)	Rotação Teste (º)	Erro(%)
1	2	1.41	30%	90	132	47%
2	2	-0.03	102%	90	129	43%
3	2	0.04	98%	90	132	46%
4	2	-1.41	170%	90	130	45%
5	2	-0.28	114%	90	132	47%
6	2	-0.74	137%	90	125	39%
7	2	0.76	62%	90	130	44%

Tabela 4.19: Tabela com os resultados para os testes referentes a uma curva de 90º utilizando o filtro Kalman, juntamente com o filtro elimina banda e Butterworth, sendo que antes e depois de efetuar a curva é feito um deslocamento de 1 m.

#	Deslocamento Teórico Linear (m)	Descolamento Teste (m)	Erro (%)	Rotação Teórica (º)	Rotação Teste (º)	Erro(%)
1	2	0.70	65%	90	137	52%
2	2	-0.60	116%	90	134	48%
3	2	0.22	89%	90	136	51%
4	2	0.44	78%	90	136	51%
5	2	0.13	93%	90	138	53%
6	2	-0.11	106%	90	131	45%
7	2	0.23	89%	90	135	50%

4.2.5 Análise de resultados

Os resultados obtidos através dos vários testes feitos para as várias técnicas utilizadas, revelam que, tendo em conta os parâmetros utilizados, que o IMU não corresponde a uma técnica eficaz de monotorização da localização, isto é, o uso isolado do IMU como fonte de dados para o cálculo do deslocamento não apresentam resultados bons o suficiente para a sua aplicação no contexto desejado. É possível observar que o uso deste equipamento, juntamente com técnicas de processamento de dados (filtros), é possível obter resultados com alguma precisão, como é possível observar na tabela 4.13, presente na página 63, onde os testes apresentaram erros semelhantes, com a excepção dos dois primeiros testes. De modo a analisar de um modo mais geral cada método, as tabelas 4.20 e 4.21 foram contruídas, sendo que estas contêm as médias dos erros obtidos para cada teste, presentes nas tabelas de resultados apresentadas ao longo deste capítulo. É possível obser que, para o teste de linha reta, os testes que obtêm melhores resultados são os testes que utilizam o filtro Butterworth, para as deslocações de 1 m e 2 m, e para o teste de 3 m, foi o teste que utiliza o filtro Elimina banda em conjunto com o filtro Butterworth. Todos os testes, inclusive o teste sem filtro, apresentam bons resultados para os testes de 1 m e 2 m, no entanto estes possuem resultados bastante negativos para o deslocamento de 3 m. Apenas

Tabela 4.20: Tabela com as médias dos erros de todos os testes feitos para a linha reta, com todas as técnicas utilizadas. A verde está assinalado o melhor resultado da linha correspondente.

#	Sem Filtro	Elimina Banda	Butterworth	Elim B + Butterworth	Unscented Kalman	UKF + Elim B + Butterworth
1m	11%	13%	6%	13%	7%	7%
2m	15%	16%	15%	16%	18%	18%
3m	120%	22%	53%	21%	98%	98%

Tabela 4.21: Tabela com as médias dos erros de todos os testes feitos para a curva, com todas as técnicas utilizadas. A verde está assinalado o melhor resultado da linha correspondente.

#	Sem Filtro	Elimina Banda	Butterworth	Elim B + Butterworth	Unscented Kalman	UKF + Elim B + Butterworth
2m	125%	125%	37%	37%	102%	93%
90º	26%	26%	26%	26%	44%	50%

o teste que utiliza os filtros elimina banda apresentam bons resultados. Isto poderá estar ligado ao facto de que o filtro elimina banda reduz o ruído presente na medições que gera oscilações constantes dos valores medidos, como é possível observar na figura 4.3. Estas oscilações, tendo em conta os resultados obtidos, correspondem à maior componente de erro introduzido nas medições, sendo que este erro é bastante reduzido quando aplicado o filtro adequado.

Em relação ao filtro de Kalman, este apresenta bons resultados para os dois primeiros testes, mas apresenta um erro extremamente elevado, o que poderá indicar que o filtro poderá não estar bem adaptado ao filtro e aos testes efectuados. Isto poderá estar ligado aos parâmetros escolhidos no desenvolvimento do mesmo, e que os resultados poderão ser melhorados com uma mudança desses mesmo parâmetros.

Para o teste referente à curva, estes apresentam bons resultados para o filtro de Butterworth e para filtro elimina banda utilizado em conjunto com o filtro de Butterworth. No entanto, os resultados não são positivos o suficiente para aplicar estas soluções no contexto real, no cenário proposto. Mais uma vez, o filtro de Kalman, que correspondia a técnica de processamento de dados mais antecipada a ter bons resultados, não obteve valores positivos. Isto poderá estar relacionado, como foi apresentado no parágrafo anterior, ao facto de os parâmetros utilizados não serem os ideais, e que estes necessitam alguns ajustes, de modo a tentar obter melhores resultados.

Em suma, os resultados obtidos não foram os esperados de modo a alcançar os objetivos propostos, no entanto, correspondem a um desenvolvimento inicial e que deverá de possibilitar uma continuação no desenvolvimento de uma sistema de localização para AGVs para ser aplicado num contexto fabril.

Capítulo 5

Conclusões e Trabalho Futuro

O presente capítulo está dividido em duas secções. A secção 5.1 consiste na apresentação de uma conclusão sobre a validação experimental, face à pergunta de investigação. A secção 5.2 consiste na exposição do trabalho futuro para o melhoramento do trabalho feito no âmbito deste tese.

5.1 Conclusão

Esta dissertação propôs que, de acordo com a pergunta de investigação e a respetiva hipótese, que num ambiente fechado, onde não é possível o uso de técnicas de investigação normais, entre elas o uso de GPS, seria possível fazer a monitorização do movimento de AGVs presentes na planta de uma fábrica, onde estes não possuem capacidade de calcular nem de transmitir informação sobre o seu estado atual, através do uso de sensores não intrusivos, como é o caso do IMU, e de um algoritmo de "*tracking*", fornecendo assim as suas localizações. O objetivo consistia no desenvolvimento inicial de um algoritmo, que servirá como base para desenvolvimentos futuros de monitorização da localização de AGVs que não têm a capacidade sensorial para obter informação sobre o seu estado, como a sua localização ou se este se encontra em normal funcionamento. Todo o trabalho desenvolvido corresponde a uma contribuição no desenvolvimento de um sistema integrado de gestão de frota de AGVs em ambiente fabril.

O estudo inicial feito, que corresponde ao estado da arte, permitiu revelar técnicas de processamento de dados, utilizados em sistemas que fazem uso de IMUs, sendo elas a aplicação de filtros, tais como os filtros de Kalman, mesmo tendo em conta que estes sistemas de localização não utilizavam apenas IMUs, e faziam uso de tecnologias como odometria e/ou "*beacons*"de RFID para complementar os IMUs. Este estudo poderá ser aplicável num trabalho futuro, onde o foco estará em aplicar outro tipo de tecnologia,

de modo a melhorar o desempenho do IMU. Um estudo mais focado na técnicas de processamento de dados, nomeadamente os filtros, permitiu a aplicação de vários tipos de filtros no desenvolvimento do sistema de localização de AGVs.

O desenvolvimento deste trabalho exigiu uma constante adaptação por parte da implementação do sistema desenvolvido, bem como na forma de execução dos testes, que apresentaram algumas dificuldades durante a sua execução. Algumas dessas dificuldades foram o "*setup*"inicial para que o IMU transmitisse os dados para o terminal desejado, sendo neste caso um computador pessoal, a falta de equipamento de modo a simular um AGV, sendo que só foi possível utilizar um carrinho que necessita de intervenção humana para andar, o espaço de teste reduzido, impossibilitando testes de grande duração e/ou grandes percursos. Estes obstáculos foram, de um modo geral, ultrapassados, tendo em conta que algumas das soluções encontradas não correspondem à solução ideal, como o carrinho utilizado para simular o AGV.

As tecnologias utilizadas no desenvolvimento deste trabalho correspondem a sensores inerciais, na forma de IMU, WiFi para a transmissão de dados entre os sensores e o terminal, tecnologias de processamento de dados, tais como os filtros elimina banda, filtro de Butterworth e filtro de Kalman "*unscented*", e a linguagem de programação Python, com as bibliotecas necessárias para o desenvolvimento do algoritmo de localização.

Tendo em conta os objetivos propostos para esta tese, presentes no capítulo 1, na secção 1.4, apresentados como pergunta de investigação e hipótese, que correspondiam à utilização de IMUs para calcular a posição relativa de um AGV, sem o auxílio de tecnologias tais como o GPS, num ambiente fechado, que corresponde à planta de uma fábrica, estes não foram cumpridos, visto que os requisitos para o deslocamento não foram cumpridos, nomeadamente, os erros obtidos são demasiado elevados para o cenário em questão.

Os resultados obtidos no capítulo anterior (capítulo 4) demonstrou que o uso de filtros, quando aplicados numa situação que se enquadre ao filtro em questão, melhora os resultados obtidos em relação à aquisição de dados fornecidos pelo IMU, nomeadamente a aceleração proveniente do acelerómetro. Isto é possível observar no caso onde foi aplicado um filtro elimina banda, de modo a reduzir o ruído existente que causa oscilações na aceleração mesmo quando este se encontra parado, fazendo com que o algoritmo calcule movimento onde não existe.

No entanto, foi possível observar que, mesmo com a aplicação de filtros que reduzem a quantidade de ruído presente nos dados fornecidos pelo IMU, os resultados obtidos para a maior das distâncias dos testes efetuados revela que, com as técnicas usadas, não é possível aplicar este algoritmo num ambiente fechado, para AGVs sem capacidade sensorial.

O filtro de Kalman, que foi o filtro mais promissor apresentado no estudo feito no capítulo do estado da arte (capítulo 2), demonstrou resultados promissores excepto também para os testes que apresentavam uma distância maior durante a sua execução. Uma razão possível para estes resultados poderá estar ligado às especificações do filtro, que

poderão não estar optimizados para os sensores ou para o algoritmo. Deste modo, será apresentado na secção seguinte, trabalhos que poderão aumentar o rendimento que este filtro terá sobre o algoritmo e à monitorização da localização dos sensores em questão.

Finalmente, em relação à pergunta de investigação e a respetiva hipótese, este tese demonstrou que, para as técnicas usadas com as especificações utilizadas, não é possível utilizar IMUs para a aquisição de dados respetivos à cinética de um AGV, de modo a ser possível calcular a localização em tempo real do mesmo, devido à sua inconsistência quanto maior for a distância do movimento. Deste modo, o IMU não corresponde a uma solução para os objetivos propostos. No entanto, o seu uso poderá ser feito em complemento com outras tecnologias de localização, tais como tecnologias de localização absoluta através de "*beacons*", ou de tecnologias como a odometria.

5.2 Trabalho Futuro

Como trabalho futuro, serão apresentados alguns pontos de interesse que servirão de melhoramentos para monitorização do AGV no que toca à sua localização atual, mas também as possíveis bases para o início do desenvolvimento do gestor de frota de AGVs, capaz de fornecer informações tais como o estado de funcionamento dos AGVs, isto é, se estes se encontram em funcionamento normal ou se possuem uma avaria, o estado das suas baterias, e também da atribuição de tarefas.

Os pontos referentes a trabalho futuro em relação ao algoritmo desenvolvido ligado à componente de localização dos AGVs são os seguintes:

- .
- Execução de testes num ambiente semelhante ao ambiente do cenário em questão (ambiente "*indoors*").
- Execução de testes com o equipamento onde esta tese está focada, nomeadamente AGVs sem capacidade sensorial nem de comunicação.
- Implementação de uma comunicação direta entre o IMU e o algoritmo em questão, para uma execução em tempo real, sem auxílio a ficheiros.
- Utilização de outras técnicas de localização, para complementar as técnicas utilizadas com IMUs

Em relação aos pontos que correspondem ao desenvolvimento de trabalho ligado à criação de uma plataforma de gestão de frota de AGVs, estes são:

- Implementação de uma interface gráfica para o utilizador, capaz de fornecer informação sobre a frota de AGVs.

- Desenvolvimento de um algoritmo, juntamente com a aplicação de sensores, de modo a detectar avarias.
- Desenvolvimento de uma algoritmo para monitorização do consumo de energia das baterias dos AGVs, com auxílio a sensores.
- Desenvolvimento de um algoritmo para a atribuição de tarefas aos AGVs.

Bibliografia

[1] M. et al Rüßmann. "Future of Productivity and Growth in Manufacturing". Em: *Boston Consulting* April (2015). ISSN: 9783935089296. DOI: 10.1007/s12599-014-0334-4. arXiv: arXiv:1011.1669v3.

[2] H. Lasi, P. Fettke, H. G. Kemper, T. Feld e M. Hoffmann. "Industry 4.0". Em: *Business and Information Systems Engineering* 6.4 (2014), pp. 239–242. ISSN: 18670202. DOI: 10.1007/s12599-014-0334-4.

[3] Politechnic Institute of Bragança, Universidade do Porto, Institute of Electrical and Electronics Engineers, IEEE Industrial Electronics Society, Universidade Nova de Lisboa e INESC TEC (Organization). *IEEE 16th International Conference on Industrial Informatics (INDIN) : proceedings : Faculty of Engineering of the University of Porto, Porto, Portugal, 18-20 July, 2018*. ISBN: 9781538648292.

[4] Volkswagen Autoeuropa, Uninova e INNOVALIA. *Boost 4.0 Chapter 1*.

[5] *BOOST 4.0*. URL: https://boost40.eu/.

[6] M. Yli-Ojanperä, S. Sierla, N. Papakonstantinou e V. Vyatkin. "Adapting an agile manufacturing concept to the reference architecture model industry 4.0: A survey and case study". Em: *Journal of Industrial Information Integration* 15.November 2018 (2019), pp. 147–160. ISSN: 2452414X. DOI: 10.1016/j.jii.2018.12.002. URL: https://doi.org/10.1016/j.jii.2018.12.002.

[7] *Autoeuropa bate recorde de produção*. "Acedido: 09/01/2020 às 13:15". URL: https://expresso.pt/economia/2019-11-18-Autoeuropa-bate-recorde-de-producao.

[8] *Volkswagen_Autoeuropa*. "Acedido: 15/01/2020 às 15:45. URL: https://pt.wikipedia.org/wiki/Volkswagen_Autoeuropa.

[9] S. Taneja, A. Akcamete, B. Akinci, J. H. Garrett, L. Soibelman e E. W. East. "Analysis of three indoor localization technologies for supporting operations and maintenance field tasks". Em: *Journal of Computing in Civil Engineering* 26.6 (2012), pp. 708–719. ISSN: 08873801. DOI: 10.1061/(ASCE)CP.1943-5487.0000177.

[10] J Borenstein, H. R. Everett, L Feng e D Wehe. "Mobile Robot Positioning & Sensors and Techniques". Em: *Special Issue on Mobile Robots* 14.4 (), pp. 231–249. ISSN: 0704-0188.

[11] M. B. Kjærgaard, H. Blunck, T. Godsk, T. Toftkjær, D. L. Christensen e K. Grønbæk. "Indoor Positioning Using GPS Revisited". Em: *Pervasive Computing*. Ed. por P. Floréen, A. Krüger e M. Spasojevic. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 38–56. ISBN: 978-3-642-12654-3.

[12] J. Zhou e J. Shi. "RFID localization algorithms and applications-A review". Em: *Journal of Intelligent Manufacturing* 20.6 (2009), pp. 695–707. ISSN: 09565515. DOI: 10.1007/s10845-008-0158-5.

[13] C. Roehrig e C. Röhrig. *Global Localization and Position Tracking of Automatic Guided Vehicles using passive RFID Technology*. Rel. téc. 2014, pp. 401–408. URL: https://www.researchgate.net/publication/275713069.

[14] M. Granados-Cruz, J. Pomárico-Franquiz, Y. S. Shmaliy e L. J. Morales-Mendoza. "Triangulation-based indoor robot localization using extended FIR/Kalman filtering". Em: *2014 11th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*. 2014, pp. 1–5. DOI: 10.1109/ICEEE.2014.6978256.

[15] M. Brossard, A. Barrau e S. Bonnabel. *AI-IMU Dead-Reckoning*. 2019. arXiv: 1904.06064 [cs.RO].

[16] K. Dhwaj, J. M. Kovitz, H. Tian, L. J. Jiang e T. Itoh. "Half-mode cavity-based planar filtering antenna with controllable transmission zeroes". Em: *IEEE Antennas and Wireless Propagation Letters* 17.5 (2018), pp. 833–836. ISSN: 15361225. DOI: 10.1109/LAWP.2018.2818058.

[17] W. Schwarting, J. Alonso-Mora e D. Rus. "Planning and Decision-Making for Autonomous Vehicles". Em: *Annual Review of Control, Robotics, and Autonomous Systems* 1.1 (2018), pp. 237–257. ISSN: 2573-5144. DOI: 10.1146/annurev-control-060117-105157.

[18] R. Cechowicz e M. Bogucki. "Indoor vehicle tracking with a smart MEMS sensor". Em: *MATEC Web of Conferences* 252 (2019), p. 02004. DOI: 10.1051/matecconf/201925202004.

[19] D. A. Grejner-Brzezinska, Y. Yi e C. K. Toth. "Bridging GPS Gaps in Urban Canyons: The Benefits of ZUPTs". Em: *Navigation* 48.4 (2001), pp. 216–226. ISSN: 0028-1522. DOI: 10.1002/j.2161-4296.2001.tb00246.x.

[20] A. Azenha e A. Carvalho. "Dynamic analysis of AGV control under dead-reckoning algorithm". Em: *Robotica* 26.5 (2008), pp. 635–641. ISSN: 02635747. DOI: 10.1017/S0263574708004244.

[21] H. Temeltas. "A REAL-TIME LOCALIZATION METHOD FOR AGVS IN SMART FACTORIES". Em: *Science. Business. Society.* 3.2 (2018), pp. 45–50.

[22] K. O'Keefe, A. Moreira, M. Petovello, G. Lachapelle, University of Calgary, Institute of Electrical and Electronics Engineers e IEEE Geoscience and Remote Sensing Society. *2015 International Conference on Indoor Positioning and Indoor Navigation : 13-16 October 2015, Banff, Alberta, Canada*. ISBN: 9781467384025.

[23] I. Stanculeanu e T. Borangiu. "Enhanced RSSI localization system for asset tracking services using Non expensive IMU". Em: *IFAC Proceedings Volumes (IFAC-PapersOnline)*. Vol. 14. PART 1. 2012, pp. 1838–1843. ISBN: 9783902661982. DOI: 10.3182/20120523-3-RO-2023.00430.

[24] G. Giorgetti. "RF-BASED LOCALIZATION IN GPS-DENIED APPLICATIONS by". Em: March (2016).

[25] I. F. Vis. "Survey of research in the design and control of automated guided vehicle systems". Em: *European Journal of Operational Research* 170.3 (2006), pp. 677–709. ISSN: 03772217. DOI: 10.1016/j.ejor.2004.09.020.

[26] C. Llopis-Albert, F. Rubio e F. Valero. "Designing Efficient Material Handling Systems Via Automated Guided Vehicles (AGVs)". Em: *Multidisciplinary Journal for Education, Social and Technological Sciences* 5.2 (2018), p. 97. ISSN: 2341-2593. DOI: 10.4995/muse.2018.10722.

[27] P. Valmiki, A. Simha Reddy, G. Panchakarla, K. Kumar, R. Purohit e A. Suhane. "A Study on Simulation Methods for AGV Fleet Size Estimation in a Flexible Manufacturing System". Em: *Materials Today: Proceedings*. Vol. 5. 2. Elsevier Ltd, 2018, pp. 3994–3999. DOI: 10.1016/j.matpr.2017.11.658.

[28] V. K. Chawla, A. K. Chanda e S. Angra. "Automatic guided vehicles fleet size optimization for flexible manufacturing system by grey wolf optimization algorithm". Em: *Management Science Letters* 8.2 (2018), pp. 79–90. ISSN: 19239343. DOI: 10.5267/j.msl.2017.12.005.

[29] S. Mirjalili, S. M. Mirjalili e A. Lewis. "Grey Wolf Optimizer". Em: *Advances in Engineering Software* 69 (2014), pp. 46–61. ISSN: 09659978. DOI: 10.1016/j.advengsoft.2013.12.007. URL: http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.

[30] R. Emilia. "Coordination of Industrial AGVs Roberto Olmi * Cristian Secchi and Cesare Fantuzzi". Em: x.x (2011).

[31] D. Li e K. Niu. "Dijkstra's algorithm in AGV". Em: *Proceedings of the 2014 9th IEEE Conference on Industrial Electronics and Applications, ICIEA 2014* 51375058 (2014), pp. 1867–1871. DOI: 10.1109/ICIEA.2014.6931472.

[32] P. A. O'Donnell e T. Lozano-Perez. *Deadlock-free and collision-free coordination of two robot manipulators*. 1989. DOI: 10.1109/robot.1989.100033.

[33] P. S. Pratama, Y. D. Setiawan, D. H. Kim, Y. S. Jung, H. K. Kim, S. B. Kim, S. K. Jeong e J. I. Jeong. "Fault detection algorithm for automatic guided vehicle based on multiple positioning modules". Em: *Proceedings of the 2014 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2014*. Institute of Electrical e Electronics Engineers Inc., 2014, pp. 751–757. ISBN: 9781479930791. DOI: 10.1109/ICACCI.2014.6968511.

[34] S. V. Vaseghi. "Noise and Distortion". Em: *Advanced Digital Signal Processing and Noise Reduction* 9 (2001), pp. 29–43. DOI: 10.1002/0470841621.ch2.

[35] I. W. Selesnick e C. Sidney Burrus. "Generalized digital butterworth filter design". Em: *IEEE Transactions on Signal Processing* 46.6 (1998), pp. 1688–1694. ISSN: 1053587X. DOI: 10.1109/78.678493.

[36] "Acedido: 18/11/2020 às 15:46". URL: https://tttapa.github.io/Pages/Mathematics/Systems-and-Control-Theory/Analog-Filters/Butterworth-Filters.html.

[37] "Acedido: 18/11/2020 às 15:36". URL: https://forum.smartcitizen.me/t/about-filters-and-convolution-signal-processing-for-audio-part-iv/945.

[38] "Acedido: 18/11/2020 às 16:07". URL: https://control.com/textbook/instrument-connections/electrical-signal-and-control-wiring/.

[39] H. Hellmers, A. Norrdine, J. Blankenbach e A. Eichhorn. "An IMU/magnetometer-based indoor positioning system using Kalman filtering". Em: *2013 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2013* October (2013). DOI: 10.1109/IPIN.2013.6817887.

[40] W. Chai, C. Chen, E. Edwan, J. Zhang e O. Loffeld. "INS/Wi-Fi based indoor navigation using adaptive Kalman filtering and vehicle constraints". Em: *WPNC'12 - Proceedings of the 2012 9th Workshop on Positioning, Navigation and Communication* (2012), pp. 36–41. DOI: 10.1109/WPNC.2012.6268735.

[41] *Livro de Kalman interactivo*. "Acedido: 23/11/2020 às 10:07". URL: https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python.

[42] S. S. Bhattacharyya, E. F. Deprettere, R. Leupers e J. Takala. *Handbook of signal processing systems*. 2018, pp. 1–1210. ISBN: 9783319917344. DOI: 10.1007/978-3-319-91734-4.

[43] M. Wright e A. Freed. "Open Sound Control: A New Protocol for Communicating with Sound Synthesizers". Em: *International Computer Music Conference (ICMC)*. International Computer Music Association (ICMA). Thessaloniki, Hellas: International Computer Music Association (ICMA), 1997, pp. 101–104. URL: http://cnmat.berkeley.edu/http://hdl.handle.net/2027/spo.bbp2372.1997.033/open_sound_control_new_protocol_communicating_sound_synthesizers.

[44] *Scipy Library*. "Acedido: 13/11/2020 às 17:39". URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.trapz.html#scipy.integrate.trapz.

[45] S. Winder. *Analog and Digital Filter Design*. Newnes, 2002.

[46] "Acedido: 12/02/2021 às 13:45". URL: https://web.archive.org/web/20030914224904/http://cnmat.berkeley.edu/OSC/OSC-spec.html.

[47] M. WRIGHT. "Open Sound Control: an enabling technology for musical networking". Em: *Organised Sound* 10.3 (2005), 193–200. DOI: 10.1017/S1355771805000932.

[48] A. Schmeder, A. Freed e D. Wessel. "Best Practices for Open Sound Control". Em: *Linux Audio Conference* May 2014 (2010), p. 10.

[49] D. Grzelak, C. Mai e U. Aßmann. "Towards a software architecture for near real-time applications of IoT". Em: *IoTBDS 2019 - Proceedings of the 4th International Conference on Internet of Things, Big Data and Security* IoTBDS (2019), pp. 197–206. DOI: 10.5220/0007414101970206.

[50] "Acedido: 12/02/2021 às 15:39". URL: https://github.com/Ircam-R-IoT/bitalino-riot-firmware.

ANEXO I

Anexo 1 - Algoritmo de tracking

Listagem I.1: Algoritmo "*tracking*"

```
import math
import scipy
import numpy as np
''' Lists where the data will be stored after calling the get_data_from_files method
# acc_data -> Acceleration data
# yaw_data -> Yaw data
# len_data -> List with all sample numbers (for plotting)
'''
acc_data, yaw_data, len_data = get_data_from_file(dir_str)

''' vel_data -> ndarray with the velocity data obtained through the method get_velocity
    ↪ which integrates the acc_data list '''
vel_data = integrate(acc_data)

''' integral_x_disp -> ndarray with the displacement data obtained through the method
    ↪ get_displacement which integrates the vel_data list '''
integral_x_disp = integrate(vel_data)

disp_ant = 0
x_pos = [0]
y_pos = [0]

for j, disp in enumerate(integral_x_disp):
    x_pos.append(x_pos[-1] + (disp - disp_ant) * float(math.cos(math.radians(yaw_data[j]))
        ↪ ))
    y_pos.append(y_pos[-1] + (disp - disp_ant) * float(math.sin(math.radians(yaw_data[j]))
        ↪ ))

    disp_ant = disp
```

ANEXO II

Anexo 2 - Algoritmo de comunicação

Listagem II.1: Algoritmo de comunicação

```
import chart_studio.plotly
import argparse
from pythonosc import dispatcher
from pythonosc import osc_server
from typing import List, Any

acc_buffer = []
ang_buffers = []
len_buffer = []

def print_accX_angZ(address: str, *args: List[Any]):
    try:
        value1 = args[0]
        value2 = args[5]
        print(f"values: {value1}, {value2}")
    except ValueError:
        pass


chart_studio.tools.set_credentials_file(username='a.grilo', api_key='a95wuDJrcTYHOqvNNmoP
    ↪ ')

if __name__ == "__main__":
    parser = argparse.ArgumentParser()
    parser.add_argument("--ip",
                        default="192.168.1.100", help="The ip to listen on")
    parser.add_argument("--port",
                        type=int, default=8888, help="The port to listen on")
    args = parser.parse_args()
```

```
29
30      dispatcher = dispatcher.Dispatcher()
31      dispatcher.map("/0/raw", print_accX_angZ)
32
33      server = osc_server.BlockingOSCUDPServer(
34          (args.ip, args.port), dispatcher)
35      print("Serving on {}".format(server.server_address))
36      server.serve_forever()
```

ANEXO III

Anexo 3 - Algoritmo do filtro Butterworth

Listagem III.1: Algoritmo do filtro Butterworth

```
from scipy.signal import butter, lfilter

def filter_data(Y):
    order = 3
    fs = 200.0 # sample rate, Hz
    cutoff = 1.5 # desired cutoff frequency of the filter, Hz

    # Get the filter coefficients so we can check its frequency response.
    y = butter_lowpass_filter(Y, cutoff, fs, order)

    return y


def butter_lowpass(cutoff, fs, order=5):
    nyq = 0.5 * fs
    normal_cutoff = cutoff / nyq
    b, a = butter(order, normal_cutoff, btype='low', analog=False)
    return b, a


def butter_lowpass_filter(data, cutoff, fs, order=5):
    b, a = butter_lowpass(cutoff, fs, order=order)
    y = lfilter(b, a, data)
    return y
```