SISTEMA DE NAVEGAÇÃO INERCIAL PARA VEÍCULOS BASEADO EM APRENDIZADO DE MÁQUINA E CORRESPONDÊNCIA DE MAPAS

Lucas de Carvalho Gomes

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Luís Henrique Maciel Kosmalski Costa

Rio de Janeiro
Setembro de 2020

SISTEMA DE NAVEGAÇÃO INERCIAL PARA VEÍCULOS BASEADO EM APRENDIZADO DE MÁQUINA E CORRESPONDÊNCIA DE MAPAS

Lucas de Carvalho Gomes

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Orientador: Luís Henrique Maciel Kosmalski Costa

Aprovada por: Prof. Luís Henrique Maciel Kosmalski Costa
Prof. Marcelo Gonçalves Rubinstein
Prof. Rodrigo de Souza Couto

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2020

Gomes, Lucas de Carvalho

Sistema de Navegação Inercial para Veículos Baseado em Aprendizado de Máquina e Correspondência de Mapas/Lucas de Carvalho Gomes. – Rio de Janeiro: UFRJ/COPPE, 2020.

XIV, 56 p.: il.; 29, 7cm.

Orientador: Luís Henrique Maciel Kosmalski Costa

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2020.

Referências Bibliográficas: p. 52 – 56.

1. Navegação Inercial. 2. Sistemas de Posicionamento por Satélite. 3. Aprendizado de Máquina. 4. Correspondência de Mapas. I. Costa, Luís Henrique Maciel Kosmalski. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

À minha família.

Agradecimentos

Primeiramente, agradeço à minha família, que sempre me deu suporte e incentivo para que eu batalhasse em prol dos meus objetivos e do meu aprimoramento como pessoa.

Em especial, agradeço à minha mãe por sempre acreditar em mim, por acompanhar tão de perto o meu percurso até aqui, por ser um exemplo de força e determinação e por todo o carinho, todo o suporte e todo o aprendizado que você me proporcionou. Devo uma grande parte do que sou hoje a você.

Também agradeço muito ao meu pai, que, para mim, é um grande exemplo de perseverança. Muito obrigado pelo suporte e pelo carinho.

Agradeço à Universidade Federal do Rio de Janeiro (UFRJ), instituição que se tornou a minha segunda casa desde 2013. Tive acesso aqui a uma educação pública, gratuita e de qualidade, que me deu elementos para seguir uma formação profissional e oportunidades de amadurecimento. Sinto-me honrado de ser um "filho da Minerva".

Também sou muito agradecido ao pessoal do Grupo de Teleinformática e Automação (GTA), espaço onde agora completo 6 anos e 7 meses de pesquisa, começando como um aluno de graduação que entrava na Iniciação Científica no seu terceiro período e chegando ao fim do meu mestrado. Tive aqui uma grande oportunidade de crescimento profissional e pessoal.

Em especial no GTA, agradeço demais ao professor Luís Henrique Costa pela paciência, pelo suporte que sempre me foi dado como meu orientador na graduação e no mestrado e por me orientar neste trabalho. Também agradeço por ter fornecido o carro para fazer a campanha de coleta dos dados utilizados neste trabalho!

Eu sou muito grato também ao João Batista Pinto Neto (*in memoriam*), o grande JB, doutorando, professor, co-orientador e grande amigo, com quem trabalhei por bastante tempo no GTA em projetos, artigos e congressos. A atitude sempre positiva e a perseverança na resolução dos problemas, por mais complexos que fossem, eram, definitivamente, uma inspiração.

Nestes quase 7 anos de GTA, também convivi com vários alunos de graduação, mestrado e doutorado, que também me apoiaram e me ajudaram no desenvolvimento dos meus trabalhos e com quem compartilhei bons momentos de descontração. Destes, cito especialmente Tatiana Sciammarella, Vitor Borges, Alyson Santos, Marcus

Braga, Dianne Medeiros, Fernando Molano, Thales Almeida, Mariana Maciel, Ana Elisa Ferreira, Fabio Vieira, Hugo Sadok, Luana Gantert, Daiane Pereira, Roberto Pacheco e Pedro Cruz.

Agradeço também aos professores Rodrigo de Souza Couto e Marcelo Gonçalves Rubinstein por aceitarem participar da banca de avaliação desta dissertação de mestrado.

Um muito obrigado aos funcionários do Programa de Engenharia Elétrica da COPPE/UFRJ pela prestatividade e agilidade com que realizam seus trabalhos.

Por fim, agradeço aos meus amigos de fora da UFRJ, que conheci em diferentes fases da minha vida, pelas conversas, pelos momentos divertidos e pelo companheirismo.

Este trabalho é uma forma de retribuir todo o apoio que recebi durante tantos anos.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

SISTEMA DE NAVEGAÇÃO INERCIAL PARA VEÍCULOS BASEADO EM APRENDIZADO DE MÁQUINA E CORRESPONDÊNCIA DE MAPAS

Lucas de Carvalho Gomes

Setembro/2020

Orientador: Luís Henrique Maciel Kosmalski Costa

Programa: Engenharia Elétrica

Atualmente, sistemas de posicionamento por satélite (GNSS – *Global Navigation Satellite System*) são empregados em diversos contextos da vida cotidiana. Eles fornecem a localização em tempo real de qualquer objeto ou pessoa portador de um receptor, através da comunicação entre este receptor e satélites por ondas eletromagnéticas. O desempenho deste posicionamento está sujeito a diversos fatores ambientais e técnicos. Além disto, alguns cenários, como cânions (urbanos ou geográficos), florestas e túneis, são especialmente desafiadores, dado que a cobertura neles é ausente ou pouco confiável, produzindo informações imprecisas ou não sendo possível obter nenhum dado. Por conta destes fatores, sistemas que requerem esta informação com alta disponibilidade e precisão costumam empregar outros sensores. No entanto, reduzir a quantidade de dispositivos de sensores pode ser benéfico, já que reduz os custos e o consumo de energia. Buscando melhorar a confiabilidade e a disponibilidade de sistemas baseados no posicionamento por satélite, este trabalho propõe um sistema de navegação inercial. Ele utiliza a última localização conhecida e dados obtidos através de sensores auxiliares para estimar a localização atual. Por usar apenas sensores já disponíveis em veículos comerciais, o sistema mantém, simultaneamente, o custo-benefício. As estimativas de localizações são calculadas através de modelos de Aprendizado de Máquina, e aprimoradas com um procedimento de correspondência de mapas (*map matching*). O desempenho deste procedimento foi avaliado por simulações computacionais alimentadas com dados reais de posições e sensores, observando-se a sua capacidade em reproduzir trajetórias em um cenário urbano.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

VEHICULAR DEAD RECKONING SYSTEM BASED ON MACHINE LEARNING AND MAP MATCHING

Lucas de Carvalho Gomes

September/2020

Advisor: Luís Henrique Maciel Kosmalski Costa

Department: Electrical Engineering

Nowadays, Global Navigation Satellite Systems (GNSS) are employed in various contexts of the daily life. They yield real-time location information for any vehicle or person bearing a receiver, through the communications by electromagnetic waves between itself and satellites. The performance of this positioning technique is subject to several environmental and technical factors. Furthermore, certain scenarios, such as canyons (urban or geographic), forests and tunnels, are particularly challenging, since the coverage in them is absent or scarcely reliable, producing rogue positioning information or even no information at all. Due to these factors, systems which depend on this information frequently deploy other sensing devices. However, reducing the amount of these devices may be beneficial, since it reduces costs and the energy consumption. Aiming to improve the reliability and the availability of systems based on satellite positioning, keeping, simultaneously, the cost-effectiveness, this work proposes a dead reckoning system, which employs the last known position and data from auxiliary sensors to estimate the current location. The sensors deployed here are available in commercial vehicles. The estimates are calculated by Machine Learning models and improved by a Map Matching procedure. The performance of this procedure was evaluated through computer simulations fed with real data of locations and sensors, keeping track of the system capability of reproducing trajectories in an urban scenario.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

ABS	*Anti-lock Braking System*, p. 12
ACK	*Acknowledgement*, p. 13
CAN-H	*CAN High*, p. 12
CAN-L	*CAN Low*, p. 12
CAN	*Controller Area Network*, p. 11
CRC	*Cyclic Redundancy Check*, p. 13
DLC	*Data Length Code*, p. 13
ECEF	*Earth-Centered, Earth-Fixed*, p. 16
ECU	*Electronic Control Unit*, p. 12
EM	Erro Médio, p. 40
EQ	Erro Quadrático, p. 24
GLONASS	*Globalnaya Navigatsionnaya Sputnikovaya Sistema*, russo para Sistema de Navegação Global por Satélite, p. 2
GNSS	*Global Navigation Satellite System*, p. vii
GPS	*Global Positioning System*, p. 2
IDE	*Extended IDEntifier*, p. 13
IMU	*Inertial Measurement Unit*, p. 7
LIDAR	*Light Detection and Ranging*, p. 3
NED	*North-East-Down*, p. 15
OBD-2	*On-Board Diagnostics 2*, p. 11
OBU	*On-Board Unit*, p. 11

OSRM *Open Source Routing Machine*, p. 28

QZSS *Quasi-Zenith Satellite System*, p. 2

RTR *Remote Transmission Request*, p. 12

WGS *World Geodetic System*, p. 17

Capítulo 1

Introdução

Para apresentar as motivações que levaram à elaboração deste trabalho, este capítulo apresenta os sistemas de posicionamento por satélite, fornecendo uma descrição de alto nível das suas operações, apontando aplicações contemporâneas que requerem a localização geográfica fornecida por eles para operar, os principais desafios que eles enfrentam e, por fim, fornece uma descrição resumida da proposta, que será explorada em mais detalhes nos próximos capítulos.

1.1 Sistemas de Posicionamento por Satélite

Atualmente, sistemas de posicionamento por satélite são usados em vários cenários. Exemplos de aplicações deles são:

- A localização de objetos, pessoas ou veículos no mundo;
- Aplicações de navegação ponto a ponto: indicam um caminho entre dois locais com instruções, dadas passo a passo;
- Sistemas de segurança no trânsito

No domínio de segurança no trânsito, várias aplicações de veículos inteligentes têm surgido. Sistemas de prevenção de colisões, como o desenvolvido por Pinto Neto *et al.* [3], necessitam da localização de veículos para determinar a distância entre eles e avaliar o risco de colisão, para, se necessário, emitir avisos sonoros ou visuais, ou tomar o controle do veículo e acionar o sistema de frenagem. Sistemas de controle de tráfego veicular inteligentes, sejam eles dependentes da infraestrutura de semáforos, como no trabalho de Fleck *et al.* [4], ou distribuídos, como nas propostas de Ferreira *et al.* [5] e Rapelli *et al.* [6], dependem da disponibilidade de localizações acuradas de veículos para detectar em quais ruas estão os veículos e a quantidade deles em cada rua. Com estas informações, eles atribuem de maneira adaptativa

períodos para cada rua, durante os quais seus veículos podem trafegar. Sistemas de frenagem autônoma, como a proposta de Pinto Neto *et al.* [7], pedem a disponibilidade da localização para determinar o momento em que a frenagem é acionada. Sistemas de condução autônoma de veículos, como o trabalho de Emzivat *et al.* [8], requerem a localização do próprio veículo, além da posição de outros veículos, de pedestres, de ciclistas e de obstáculos, para determinar as ações desempenhadas em cada momento.

Os GNSSs (*Global Navigation Satellite Systems*) calculam a posição através do tempo de propagação de ondas eletromagnéticas [9] enviadas de um ou mais satélites a um receptor, que pode estar instalado em diversos dispositivos. Os satélites usados nos GNSS seguem órbitas não-geoestacionárias, ou seja, os satélites estão em movimento, considerando o referencial da Terra. Para o GPS (*Global Positioning System*), por exemplo, os satélites se posicionam a uma altitude de aproximadamente 20.184 km em relação à terra, muito menor que a altitude necessária para um satélite geoestacionário, de cerca de 36.000 km [9]. Por padrão, a posição de um receptor GNSS é expressa através das coordenadas geodésicas, indicando-a através da latitude (distância em graus do receptor ao Equador, variando de 0º a 90º para norte e sul) e da longitude (distância em graus ao Meridiano de Greenwich, variando de 0º a 180º para leste e oeste). Além destas duas coordenadas, os GNSSs determinam a altitude do seu receptor, em metros.

Atualmente, quatro sistemas de posicionamento por satélite operam mundialmente: o americano GPS, o russo GLONASS (*Globalnaya Navigatsionnaya Sputnikovaya Sistema*, russo para Sistema de Navegação Global por Satélite), o europeu GALILEO e o chinês BeiDou. A diferença principal entre eles são as constelações de satélites que cada um acessa. Além destes sistemas, está em desenvolvimento o QZSS (*Quasi-Zenith Satellite System*), um sistema japonês que visa empregar satélites adicionais para aprimorar o desempenho do GPS na Ásia e na Oceania, com foco especial no Japão [10]. Através dos satélites adicionais e de correções que têm como base a órbita dos satélites, erros causados nos relógios dos dispositivos envolvidos e fatores atmosféricos, a precisão das posições se encontra na ordem de centímetros, como relatado em Wang *et al.* [10]. Os fatores das correções são analisados por estações de monitoramento, que também enviam as correções aos receptores QZSS através dos satélites.

1.1.1 Desafios do Posicionamento por Satélite

Diversos fatores ambientais e técnicos influenciam no desempenho de tais sistemas. Os fatores listados a seguir têm efeitos na precisão e na acurácia das posições extraídas de um GNSS:

- Camadas da atmosfera terrestre: a troposfera e a ionosfera causam desvios nas ondas eletromagnéticas propagadas;
- Condições do tempo atmosférico: nuvens também provocar desvios nos sinais, prejudicando a transmissão;
- Efeito de múltiplos caminhos (ou multicaminho): reflexões dos sinais por obstáculos, como prédios, árvores, veículos e até pela própria atmosfera fazem com que o sinal emitido pelo satélite possa chegar à antena do receptor mais de uma vez, por mais de um caminho. Assim, dois ou mais pulsos do mesmo sinal são recebidos. Esses pulsos, ao chegarem, podem ter diferenças de fase entre si, causando interferências que geram ruídos e contribuem para atenuar o sinal;
- Atrasos nos relógios dos satélites: os relógios dos satélites são usados na medida do tempo de propagação; a acurácia deles influencia a acurácia da posição calculada;
- Erros nas posições dos satélites (ou *ephemeris*): os satélites também possuem sensores para obter as suas posições, usados para determinar a posição do receptor em relação a eles; a precisão e a acurácia deles afetam a precisão e a acurácia finais obtidas.

Os GNSSs também são sujeitos à perda de cobertura em alguns ambientes, como florestas muito densas, cânions (geográficos - acidentes geográficos - ou urbanos - o acúmulo de edifícios), e túneis. Em tais cenários, os desvios ou reflexões sofridos pelo sinal são muito fortes, prejudicando fortemente a recepção, ou não há linha de visada entre o satélite e o receptor. Isto impede a comunicação entre satélite e receptor, impedindo a obtenção de localizações. Este é um problema especialmente grave e pode atrapalhar ou impedir o funcionamento de aplicações dependentes da posição geográfica, a depender dos requisitos acerca da precisão, da acurácia e da disponibilidade desta informação. A perda de cobertura é um grande risco a aplicações que envolvem segurança no trânsito ou condução autônoma de veículos, pois, com o funcionamento prejudicado, acidentes tornam-se mais prováveis. Dada a gravidade do problema, muitos pesquisadores voltam-se para tecnologias adicionais de sensoriamento, de modo a obter uma informação de posicionamento adequada para determinadas finalidades através de outra fonte. Exemplos de fontes alternativas são sensores de ultrassom, câmeras ou LIDAR (*Light Detection and Ranging*), que permitem a obtenção de posições relativas [11]. Contudo, evitar o emprego de tecnologias de sensoriamento adicionais tem como benefícios a economia de energia e a redução de custos envolvidos com instalações e manutenção.

1.2 Descrição da Proposta

Situando-se neste contexto, a proposta deste trabalho é a elaboração de um sistema de navegação inercial para veículos terrestres. Seu objetivo é prover, em tempo real, a posição geográfica durante eventos de perda de cobertura, buscando manter a acurácia em níveis aceitáveis [12]. O algoritmo proposto usa a última localização conhecida e informações obtidas de sensores instalados no veículo para calcular estimativas da posição atual. Estes sensores, como será visto nas próximas seções, estão disponíveis em veículos comerciais e permitem inferir curvas e obter sua velocidade. Para relacionar os valores dos sensores aos deslocamentos do veículo, modelos de Aprendizado de Máquina para regressão (cálculo de estimativas) e clusterização (encontrar agrupamentos no conjunto de dados) são treinados com dados obtidos de um cenário real e utilizados para calcular as estimativas.

A técnica de clusterização encontra os padrões presentes nos dados e atribui um algoritmo de regressão para cada padrão, tanto para fins de treino quanto de testes, gerando algoritmos especializados em uma categoria de movimento. Esta organização evita que desbalanceamentos de dados, em relação aos seus tipos de movimento, prejudiquem a acurácia final. O uso de vários modelos de regressão segundo determinado critério possui semelhanças com a área de Aprendizado de Máquina conhecida como *Ensemble Learning*, que usa conjuntos de modelos leves para formar um modelo mais acurado que suas partes - exemplos de técnicas deste ramo são os algoritmos *Bagging* e *AdaBoost* [13].

Além destas técnicas, os dados são tratados por métodos de pré-processamento, de modo a evitar efeitos nocivos à acurácia que são provocados por ruídos nos dados ou escalas desbalanceadas. As estimativas são, posteriormente, aprimoradas com um algoritmo de correspondência de mapas, que usa um mapa digital como base de dados auxiliar com posições válidas que o veículo pode assumir num cenário urbano. Usar esta técnica é vantajoso para veículos terrestres, visto que suas trajetórias, em cenários urbanos, são restritas a ruas e estradas, representadas nos mapas digitais.

1.3 Organização do Trabalho

O restante do trabalho se apresenta da seguinte forma: o Capítulo 2 apresenta uma descrição de trabalhos relacionados à proposta deste projeto; o Capítulo 3 detalha a metodologia utilizada para determinar as estimativas das posições, explicando os conceitos ligados a todas as técnicas empregadas aqui, informações sobre os dados envolvidos e detalhes importantes acerca do sistema de coordenadas geográficas utilizado nesta pesquisa, além de descrever o equipamento utilizado e o tratamento dos dados utilizados na avaliação de desempenho do sistema; o Capítulo 4 expõe

os resultados obtidos, primeiramente mostrando os dados coletados de posições e sensores e analisando-os e, posteriormente, detalhando a avaliação de desempenho da proposta. Por fim, o Capítulo 5 apresenta as conclusões obtidas com esta pesquisa, também apontando direções futuras do trabalho.

Capítulo 2

Trabalhos Relacionados

Diversos trabalhos propõem, através de diferentes abordagens, a estimativa de posições e trajetórias assumidas por veículos terrestres sob condições desfavoráveis de cobertura de satélites. As propostas podem ser divididas em duas categorias, de acordo com a utilização ou não de uma infraestrutura de comunicações. Neste capítulo, apresentam-se estas duas categorias e comparam-se os trabalhos encontrados na literatura com a presente proposta.

2.1 Propostas sem Comunicações ou Infraestrutura

Belhajem *et al.* [14] empregam um Filtro de Kalman Estendido, um modelo de cálculo não linear de estimativas, para estimar posições com dados obtidos a partir de um acelerômetro e um giroscópio. Utilizando apenas este algoritmo, os erros rapidamente acumularam. Para obter uma compensação a este acúmulo, utilizou-se uma rede neural e uma máquina de vetor de suporte, ajustadas por técnicas de otimização, para calcular valores de correção. Estes modelos são treinados e testados com dados reais, tirados de um ambiente urbano com um carro em movimento. Os dados de treino e teste fazem parte de conjuntos separados, o que significa que, durante os testes, o sistema lida com dados até então desconhecidos, embora ele tenha aprendido um padrão entre as variáveis envolvidas durante o treino. Esta separação é importante para avaliar a capacidade de generalização dos modelos. Cada teste durou entre 60 e 90 segundos. Ainda com estas correções, os erros apresentam um rápido acúmulo, embora com uma menor intensidade. Em menos de um minuto, o erro supera 100 m. O erro médio obtido pela proposta varia entre 23 m e 315 m na componente norte e entre 20 m e 304 m na componente leste, a depender da seção do circuito da coleta de dados.

Pinto Neto *et al.* [15] usam uma família de modelos de regressão linear como estimadores de posição, selecionados de maneira manual por limiares de valores de sensores. Os sensores usados medem as frequências de rotação de rodas de um veí-

culo. O tipo de trajetória é determinado observando-se a adequação aos limiares estabelecidos e a posição é calculada pelo modelo que foi treinado com os dados relativos àquele tipo de trajetória. Como em [14], um problema enfrentado é a acumulação de erros, por conta da reutilização de estimativas passadas, cuja gravidade aumenta à medida que a duração da perda de cobertura aumenta. Para evitar este problema, um fator de correção é calculado e aplicado dinamicamente. O sistema foi testado com dados reais extraídos do GPS e de sensores, usando um carro e um robô. Para o caso do carro, o erro médio permanece abaixo de 5 m por uma duração máxima de 80 segundos. No entanto, após esta duração, o erro assume um comportamento de crescimento rápido.

Ahmed *et al.* [16] usam sensores que indicam a rotação e a velocidade do veículo, além de uma unidade de medida de inércia (ou IMU – *Inertial Measurement Unit*) que, sendo uma combinação de giroscópios, acelerômetros e magnetômetros (estes últimos apenas para alguns modelos de IMUs), indica a força produzida pelo corpo, seu deslocamento angular e sua orientação. As leituras destes dispositivos são aplicadas a um filtro de partículas. Um filtro de partículas é um algoritmo de cálculo de estimativas que estima a distribuição de probabilidades de uma variável desejada. Com ele, é determinado o deslocamento do veículo. As previsões são, então, comparadas com um mapa que fornece dados das localizações geográficas das ruas, com granularidade suficiente para distinguir suas faixas. Um procedimento de navegação inercial, usando a velocidade e a rotação, calcula o deslocamento do veículo quando uma troca de faixa é detectada. Estas trocas são detectadas analisando-se a variância dos valores fornecidos por um giroscópio, dentro de uma janela. Somente o erro na direção longitudinal foi avaliado. Nesta métrica, para os 100 m iniciais (correspondentes a 11,2 s a uma velocidade média de 32 km/h assumida pelo veículo), os valores superam 10 m, com um valor máximo de aproximadamente 90 m. Após este trecho de 100 m, o erro pôde ser mantido abaixo de 1 m. Este comportamento se dá pela necessidade de um período de convergência para que se alcance valores mais precisos nas partículas usadas.

Zhang *et al.* [17] fazem um sistema de navegação inercial tendo uma preocupação especial com as leituras dos sensores. As duas medidas que os autores utilizam para calcular o deslocamento do veículo, em relação à última posição conhecida, são a velocidade de translação do veículo e o ângulo de rumo. Este ângulo indica a direção para a qual o veículo está trafegando, em relação a um eixo de referência – no caso do trabalho destes autores, os autores não informaram qual era esta referência. Para a velocidade, utiliza-se o odômetro do veículo, calibrado com informações de velocidade extraídas do GPS, e uma unidade de medida inercial, da qual se extrai a aceleração. Um Filtro de Kalman usa o odômetro e a aceleração para gerar um valor final ajustado. Para medir o ângulo de rumo, utiliza-se um giroscópio. O

desempenho do sistema foi avaliado com três testes feitos dirigindo-se um carro, um sendo num curto percurso em linha reta, o segundo sendo um curto trajeto com curvas, e o terceiro sendo uma trajetória mais longa com curvas e retas. Os testes curtos duraram 60 segundos; o mais longo durou 200 segundos. Os comprimentos dos percursos e as velocidades assumidas pelo veículo não foram informados. Obtiveram-se erros de, no máximo, 6 m, para o teste longo, e de 2 m, para os testes mais curtos.

2.2 Propostas com Comunicação

A aplicação de navegação ponto a ponto Waze está integrada a um projeto recente, chamado Waze Beacons [18]. Ele fornece posições em tempo real e a possibilidade de enviar alertas de trânsito dentro de túneis. A operação se dá através de microcontroladores nomeados pelos desenvolvedores como *beacons*, que são instalados nos túneis e se comunicam com os *smartphones* dos usuários do aplicativo móvel através da rede sem fio *Bluetooth*: as posições dos *smartphones* são estimadas através de métricas extraídas desta comunicação. Este projeto está em operação em algumas cidades no mundo, tais como Haifa (Israel), Chicago (Estados Unidos) e Rio de Janeiro (Brasil). A bateria dos microcontroladores dura entre 4 e 6 anos. Cada *beacon* custa US$ 28,50 e são necessários aproximadamente 42 *beacons* por milha de um túnel. Utilizando os valores fornecidos pelo Waze, pode-se concluir, então, que o custo de instalar os dispositivos para cobrir um túnel é de cerca de R$ 8.033,00 por quilômetro. Este sistema tem a desvantagem de necessitar da instalação e da manutenção de uma infraestrutura fixa, o que, além de apresentar o custo mencionado, depende dos interesses governamentais. Além disto, este projeto não apresenta uma cobertura ubíqua, o que significa que ele não garante o posicionamento do veículo em qualquer ambiente. Já um sistema que realiza esta localização de maneira independente, utilizando sensores instalados no veículo, possibilita obter a localização em qualquer localidade.

Em outro cenário desafiador, o da localização de veículos submarinos, Li *et al.* [19] propõem uma estratégia de localização cooperativa. Como veículos submarinos não conseguem utilizar sistemas de posicionamento por satélite, eles se localizam por sistemas de navegação inercial com sensores, encontrando o mesmo problema de acumulação de erros por reuso de estimativas que o encontrado nos trabalhos de Pinto Neto *et al.* [15] e Belhajem *et al.* [14]. Para evitar este acúmulo, costuma-se trazer periodicamente estes veículos para a superfície para que eles se localizem novamente via um GNSS. Depois, eles se submergem novamente para retomar suas atividades. Para evitar o gasto de energia e tempo envolvido na volta periódica à superfície, Li *et al.* propõem uma alternativa cooperativa. Ela envolve períodos de auto-localização com um algoritmo de navegação inercial autônomo, com sensores,

e períodos menos numerosos de localização cooperativa com trocas de mensagens entre os veículos, a partir das quais se consegue obter uma melhor estimativa. A proposta consegue manter o erro abaixo de 50 m durante uma operação submersa de 150 s, superando propostas similares para o mesmo ambiente.

O uso de comunicações para estimar posições quando não há a cobertura dos GNSS é uma estratégia empregada com frequência para localizar pedestres em ambientes internos. Por exemplo, as propostas de Ma *et al.* [20] e de Chattha e Naqvi [21] usam técnicas do tipo para calcular a posição dos pedestres em ambientes internos. Um desafio específico de tais ambientes é a escassez de mapas digitais. Por isto, Ma *et al.* [20] usam a própria técnica de navegação inercial para gerar mapas dos ambientes internos. Estes podem ser utilizados como uma referência para realizar a navegação dentro de tais ambientes por outros algoritmos de navegação inercial, como o proposto por Chattha e Naqvi [21]. As estimativas, neste cenário, também podem ser aprimoradas pela potência do sinal recebido, pelo dispositivo móvel usado pelo pedestre, de pontos de acesso de redes sem-fio. Os valores de potência podem ser aplicados para ter uma medida adicional da direção tomada pelo pedestre, que pode ser usada para correções, como na proposta de Shen e Wong [22]. Embora seja possível analisar trabalhos deste cenário e comparar ou reaproveitar alguns procedimentos, deve-se reconhecer que há características de ambientes internos trafegados por pedestres que diferem de ambientes urbanos limitados por veículos. Exemplos são as limitações de trajetórias – veículos são restritos às ruas –, velocidades, intensidades de curvas, a disponibilidade de mapas digitais e a disponibilidade de uma infraestrutura que pode prestar um papel auxiliar ao algoritmo, como os pontos de acesso de redes sem-fio usados no trabalho de Shen e Wong [22].

2.3 Comparação

O projeto proposto neste trabalho possui similaridades com as propostas do cenário veicular terrestre exibidas nesta seção, tanto pelo emprego de técnicas de Aprendizado de Máquina com a intenção de calcular a estimativa de posição, quanto pelos sensores utilizados e pelo algoritmo de correspondência de mapas, como será visto nas próximas seções. No entanto, a abordagem proposta aqui difere por empregar uma família de modelos de regressão, cada modelo sendo atribuído automaticamente a uma categoria movimento, usando como critério leituras de sensores em determinado momento, sendo esta separação organizada por um algoritmo de análise de agrupamentos - o que difere da proposta de Pinto Neto *et al.*, que faz essa separação manualmente. Cada modelo de regressão recebe um ajuste fino para o tipo de movimento a que ele se destina, como poderá ser observado no próximo capítulo, o que evita efeitos negativos sobre a acurácia que são causados por eventuais des-

balanceamentos no conjunto de dados. O uso simultâneo destas técnicas também diferencia este trabalho das outras propostas. A respeito dos resultados, como será discutido mais tarde, o sistema apresenta vantagens em relação a outras propostas. Apresenta um erro de comportamento mais controlado que o indicado na proposta de Belhajem *et al.* [14], não requer um período de adaptação inicial para alcançar uma melhor acurácia, como na proposta de Ahmed *et al.* [16] – o que torna o sistema apto a responder rapidamente a falhas de recepção de posições – e consegue manter uma acurácia aplicável para alguns domínios de aplicações em falhas com durações mais longas que as avaliadas em outras propostas. Outra questão é que a proposta deste trabalho não requer comunicações externas para o seu funcionamento, evitando custos adicionais que envolveriam uma infraestrutura fixa.

A seguir, o Capítulo 3 apresenta, em detalhes, os componentes lógicos e físicos desta proposta, descrevendo os algoritmos utilizados, os formatos de dados convenientes a eles e o equipamento utilizado para realizar a captura das posições geográficas e dos dados dos sensores.

Capítulo 3

Metodologia

Este capítulo detalha as características do sistema de navegação inercial proposto, descrevendo os dispositivos utilizados, o tratamento dos dados obtidos e o procedimento para calcular as estimativas de posições.

3.1 Equipamento Utilizado

O sistema de posicionamento por satélite utilizado foi o GPS e, por isto, seu nome será usado para se referir a sistemas de posicionamento por satélite a partir desta seção. No entanto, cabe ressaltar que a proposta descrita aqui pode ser aplicada com qualquer outro GNSS. O sistema deste trabalho, como descrito anteriormente, tem como foco os veículos terrestres. Ele foi desenvolvido para uma OBU (*On-Board Unit*) ou Unidade de Bordo, um equipamento empregado em aplicações de veículos inteligentes, que podem envolver comunicações sem-fio entre veículos e entre veículos e infraestrutura fixa. Foi utilizada a OBU comercial de modelo MK5, do fabricante *Cohda Wireless*. Esta OBU vem equipada com interfaces de comunicação sem-fio, o receptor GPS U-blox M8N, de acurácia teórica de 2,5 m e período de atualização de 250 ms, e uma conexão com o barramento CAN.

Os sensores mencionados anteriormente foram acessados através do barramento CAN. Este é uma rede cabeada, através da qual se pode acessar sensores e atuadores de um carro, usando equipamentos compatíveis. O CAN possibilita analisar as condições do veículo, identificar defeitos e configurar mecanismos do carro, além de estabelecer a comunicação entre sensores e atuadores. Os atuadores agem de maneira autônoma com base nas informações fornecidas pelos sensores, de modo a garantir a operação de componentes do veículo, como sistemas de controle de tração e estabilidade, e frenagem anti-bloqueio. A Figura 3.1 apresenta um diagrama que ilustra a conexão por este barramento.

Pode-se observar, na imagem, o conector OBD-2, utilizado como interface entre o sistema proposto e o barramento CAN. Ao barramento estão conectados os sensores

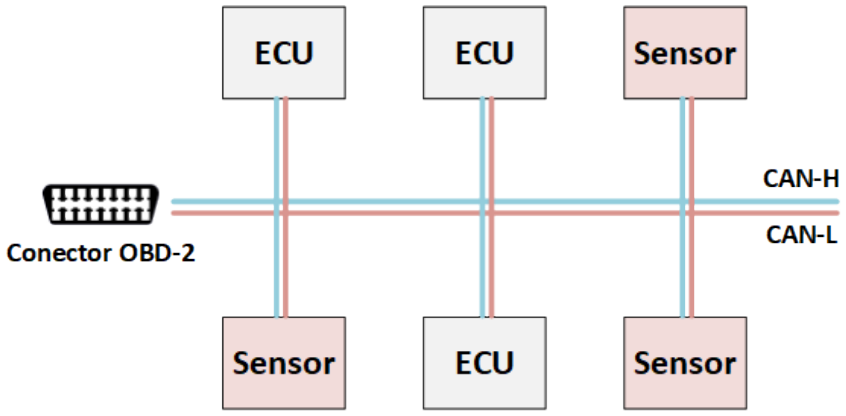


Figura 3.1: Exemplo de diagrama de conexões em um barramento CAN.

do veículo e as ECUs. As ECUs são dispositivos que leem dados fornecidos por sensores do veículo e controlam atuadores de diversos sistemas do carro, como, por exemplo, o sistema de frenagem anti-bloqueio das rodas (ABS). Há ainda dispositivos que permitem melhorar o desempenho do motor e reduzir o consumo de combustível. Os fios CAN-H e CAN-L são usados para a transmissão e a recepção dos dados. Utiliza-se um esquema chamado de sinalização diferencial, no qual o valor lógico transmitido pelo barramento é obtido a partir da diferença entre as tensões nos dois fios, o que gera dois valores lógicos. A vantagem de executar este procedimento é a maior tolerância a ruídos [1].

Qualquer um dos nós da rede CAN pode transmitir ou receber dados para qualquer outro nó, através do barramento, que é um meio físico compartilhado entre todos os dispositivos. Geralmente, também é possível a conexão de um dispositivo externo ao barramento CAN através de um conector fêmea padrão OBD-2. Isto permite a obtenção dos dados de sensores de interesse através da conexão OBD-2 disponível na OBU.

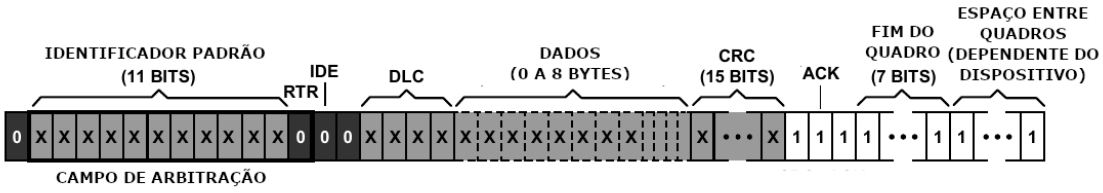


Figura 3.2: Representação gráfica de um quadro de dados na rede CAN. Figura adaptada de [1].

A Figura 3.2 apresenta um quadro de dados transmitido pelo barramento CAN, sendo cada retângulo um *bit*. *Bits* em X são categorizados como *don't care*, ou valores irrelevantes. Ou seja, não afetam a transmissão de dados da rede e carregam valores que dependem do dispositivo que utiliza a rede e dos dados transmitidos pelo quadro. O *bit* RTR (*Remote Transmission Request*) é usado quando um dispositivo

solicita, a outro dispositivo, a transmissão de algum dado requerido. O bit IDE (*Extended IDEntifier*) serve para indicar que o identificador padrão, que identifica o dispositivo que envia o quadro, usará mais *bits*. O campo DLC (*Data Length Code*) é usado para indicar o comprimento da seção de dados, indicando valores entre 0 e 8. O campo CRC (*Cyclic Redundancy Check*) contém uma sequência gerada a partir do conteúdo do quadro, que é usada no mecanismo de verificação de erros do protocolo do CAN. Por fim, o *bit* ACK (*Acknowledgement*) é usado para sinalizar a recepção de uma mensagem anterior.

Para evitar colisões, que corromperiam os dados transmitidos no barramento, a rede CAN segue um protocolo de controle de acesso ao meio, baseado nos campos chamados identificadores (ou IDs) de arbitração [1]. Estes identificadores são formados pelos campos iniciais da mensagem transmitida, sendo o principal (e primeiro) o identificador da mensagem, e ditam a prioridade entre as mensagens. No caso de um quadro de dados, ele é equivalente ao identificador padrão. Na ocorrência de um conflito, ou seja, caso dois ou mais dispositivos enviem mensagens simultaneamente, o dispositivo que enviou a mensagem com o menor ID ganha o conflito: ele continua a enviar quadros, enquanto os restantes interrompem suas transmissões. O identificador de arbitração também tem a utilidade de identificar o dispositivo transmissor dos dados: sabendo-se este valor para o sensor de interesse, pode-se recuperar seus dados das mensagens que o carregam. É importante notar que os IDs de arbitração são valores proprietários; para ter esta informação, é necessário consultar o fabricante do veículo ou inferi-los por tentativa e erro, analisando-se os valores codificados durante a operação normal do veículo para detectar se a informação desejada é transmitida através de algum ID.

Para obter as informações do barramento, a OBU monitora os dados transmitidos no barramento CAN de modo passivo, ou seja, sem enviar dados. Como os sensores informam seus dados ao CAN periodicamente, sem a necessidade de uma requisição, isto é o suficiente para obter os dados de todos os dispositivos. Para obter os valores de interesse, basta filtrar as mensagens com os identificadores de arbitração dos sensores correspondentes.

Através da rede CAN, foram extraídos os valores das frequências de rotação das rodas do veículo e sua velocidade. Os valores das frequências de rotação das rodas esquerdas e direitas diferem entre si em curvas, o que permite detectá-las. Isto ocorre porque o veículo possui juntas homocinéticas, peças que ajustam as velocidades de rotação em uma curva, de modo a igualar a velocidade angular de todas as rodas, para facilitar curvas. Este efeito será exibido com os dados coletados nas próximas seções. Já o sensor de velocidade indica quanto o veículo se deslocou, independentemente da direção. Desta forma, escolheu-se utilizar os sensores das frequências de rotação das rodas traseira esquerda e traseira direita e o sensor de velocidade. A

Tabela 3.1: Descrição do equipamento utilizado.

Tipo	Nome	Características Importantes
Unidade de Bordo	Cohda Wireless MK5	Conector OBD-2
Receptor GPS	U-blox M8N	Período de Atualização: 250 ms Acurácia: 2,5 m
Veículo	Peugeot 408 (2016)	Período de Atualização: (39,9 +/- 0,2) ms

escolha de utilizar apenas sensores já contidos no veículo é interessante, por conta do melhor custo-benefício e da independência de comunicações com o exterior do veículo, uma característica de interesse em cenários nos quais a comunicação com satélites não é confiável. Além disto, desta forma, o sistema fica independente de qualquer tipo de infraestrutura externa. Os sensores mencionados aqui estão disponíveis em veículos comerciais dotados de um sistema de frenagem ABS [23]. A Tabela 3.1 fornece as especificações dos equipamentos mencionados. O período de atualização dos sensores é o valor médio; sua incerteza é o desvio padrão, obtido através da análise dos dados coletados, descrita nas próximas seções.

3.2 Dados Utilizados

Esta seção visa esclarecer como são organizados e apresentados os dados usados no projeto.

3.2.1 Posições

Como mencionado no Capítulo 1, as posições geográficas são dadas, por padrão, por um sistema de coordenadas tridimensional. A latitude varia de 0º a 90º para norte e para sul. No receptor GPS, a direção é dada pelo sinal do valor: valores positivos correspondem ao hemisfério norte e valores negativos correspondem ao hemisfério sul. A longitude varia de 0º a 180º para leste ou para oeste. Da mesma forma que na latitude, a direção é dada pelo sinal: valores positivos indicam o hemisfério leste e valores negativos são relacionados ao hemisfério oeste. Além destes, há a altitude, indicada em metros, que será desconsiderada neste trabalho, dado que deslocamentos feitos por veículos terrestres são contidos, na maior parte dos casos, às direções horizontal e vertical. Também é possível obter, através de um receptor GPS, o ângulo de rumo, um indicador da direção na qual o receptor está trafegando. Ele é calculado com base nas duas mais recentes posições. Dentro do sistema de coordenadas usado no GPS, é igual ao ângulo que o movimento entre estas duas localizações faz com o norte magnético da Terra.

As leituras são acompanhadas de uma marca de tempo (também chamadas

de *timestamp*), que informa quando aquela leitura foi obtida. Por padrão, o formato desta marca é o *Unix Time*, indicando quanto tempo passou, em segundos, desde a meia-noite do dia primeiro de janeiro de 1970. O receptor GPS indica este valor com a precisão de centésimos de segundos, assumindo uma unidade como um segundo - as frações de segundos são dadas pelas casas decimais em ponto flutuante. Um exemplo de leitura que pode ser obtida do receptor GPS é `1483559537.15,-22.861166833,-43.227934`: o primeiro campo é o *timestamp*, que corresponde às 16:52 do dia 01/04/2017, o segundo campo é a latitude e o último valor é a longitude.

Conversão para o sistema NED

O sistema de coordenadas apresentado, composto por latitude e longitude, também chamado de geodésico, não permite tratar de modo separado os deslocamentos lineares e angulares. Desta forma, optou-se por converter estas coordenadas para o sistema NED(*North-East-Down*), que estabelece coordenadas cartesianas, para obter uma representação mais conveniente. O sistema NED é um sistema de coordenadas de plano tangente local: ele determina as posições dentro de um plano tangente em relação a um ponto de referência, efetivamente considerando a superfície da Terra como plana em uma pequena área. Desconsiderar a curvatura da Terra não causa grandes desvios quando se considera curtas distâncias, até um raio de aproximadamente 4 km [24–26]. Outros trabalhos utilizam sistemas de coordenadas de plano tangente local, como o NED, em suas implementações [14, 15].

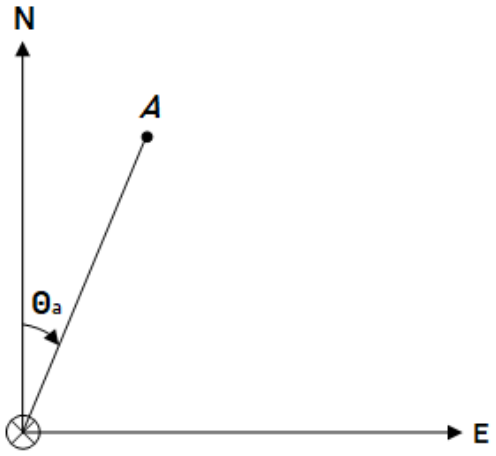


Figura 3.3: O modelo NED.

Como a Figura 3.3 mostra, as posições são representadas nos eixos vertical e horizontal em relação ao ponto de referência, que é considerada a origem. Também há uma terceira dimensão, representada pela marcação em X na origem, apontando para baixo, que representa a profundidade: esta dimensão não é utilizada na nossa

proposta. O movimento de um objeto qualquer da origem ao ponto A da figura possui um *ângulo de rumo* θ_a, que é um indicativo da direção do seu movimento. O ângulo de rumo é calculado em relação ao eixo norte, apresentando valores positivos para rotações no sentido horário.

A conversão entre o sistema geodésico e o NED também envolve a conversão intermediária entre o primeiro sistema e o sistema ECEF(*Earth-Centered, Earth-Fixed*) cartesiano. Essa necessidade se justifica pelo fato de já existirem expressões bem definidas na literatura para realizar a conversão entre geodésico e ECEF, e ECEF e NED, o que não vale para uma conversão direta entre o sistema geodésico e NED. O sistema ECEF representa as posições por um sistema tridimensional de coordenadas lineares que tem como origem o centro da Terra. O nome do sistema vem de duas características do sistema: sua origem e o fato de seus eixos serem fixos em relação à Terra - ou seja, o sistema de coordenadas gira com o planeta. A Figura 3.4 ilustra o modelo ECEF.

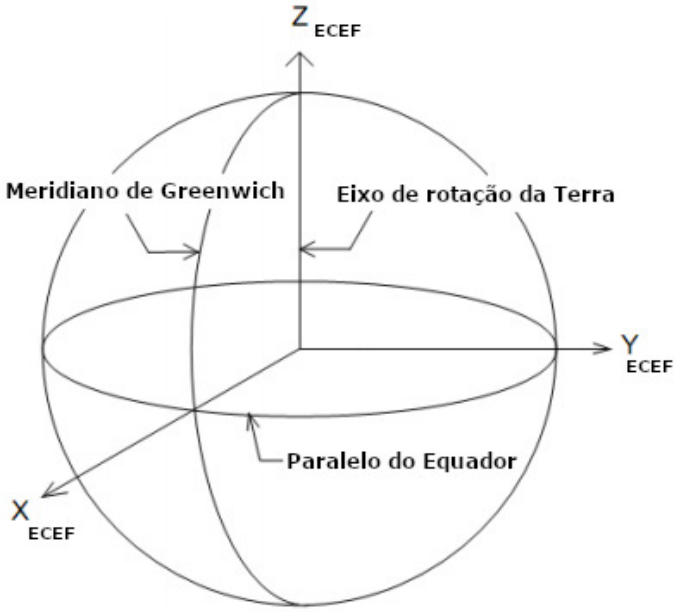


Figura 3.4: O modelo ECEF. X_{ECEF}, Y_{ECEF} e Z_{ECEF} representam os três eixos do modelo. Figura adaptada de [2].

Esta conversão intermediária é feita através das seguintes equações, de acordo com [24]:

$$X_{ECEF} = cos(lon)\ cos(lat) \times (h + \frac{a^2}{\sqrt{a^2\ cos^2(lat) + b^2\ sin^2(lat)}}), \tag{3.1}$$

$$Y_{ECEF} = sen(lon)\ cos(lat) \times (h + \frac{a^2}{\sqrt{a^2\ cos^2(lat) + b^2\ sin^2(lat)}}), \tag{3.2}$$

$$Z_{ECEF} = sen(lat) \times (h + \frac{b^2}{\sqrt{a^2\ cos^2(lat) + b^2\ sin^2(lat)}}), \tag{3.3}$$

nas quais X_{ECEF}, Y_{ECEF} e Z_{ECEF} representam, em metros, a posição do ponto em cada um dos eixos do ECEF, *lon* representa a longitude, *lat* representa a latitude, h representa a altitude (no cenário apresentado aqui, será considerada nula), a indica o raio equatorial (também chamado eixo semimaior) e b corresponde ao raio polar (também chamado de eixo semimenor). No geoide usado como referência atualmente para o GPS, o WGS 84, a equivale a 6378137 m e b é igual a 6356752,3142 m.

Estas equações consideram a situação da Figura 3.4: o eixo X cruza o Paralelo do Equador no meridiano primário no qual a longitude equivale a 0º, ou seja, no Meridiano de Greenwich. A latitude e a longitude devem ser usadas em radianos.

Resta agora realizar a conversão entre os sistemas ECEF e NED. Para isto, são usadas as seguintes equações, também de acordo com [24]:

$$\begin{aligned} N = -cos(lon_{ref})\ sen(lat_{ref})\ (p_x - r_x) - \\ sen(lon_{ref})\ sen(lat_{ref})\ (p_y - r_y) + \\ cos(lat_{ref})\ (p_z\ -\ r_z), \end{aligned} \tag{3.4}$$

$$E = -sen(lon_{ref})\ (p_x - r_x)\ +\ cos(lon_{ref})\ (p_y - r_y), \tag{3.5}$$

$$\begin{aligned} D = -cos(lon_{ref})\ cos(lat_{ref})\ (p_x - r_x) - \\ sen(lon_{ref})\ cos(lat_{ref})\ (p_y - r_y)\ sen(lat_{ref})\ (p_z - r_z), \end{aligned} \tag{3.6}$$

nas quais N, E e D representam, em metros, a posição do ponto nos três eixos do sistema NED, a latitude e a longitude do ponto de referência do plano tangente (representadas por lat_{ref} e lon_{ref}) são novamente usadas em radianos, p_x, p_y e p_z representam as coordenadas nos três eixos do sistema ECEF para o ponto do qual se deseja saber a posição no sistema NED, e r_x, r_y e r_z indicam as coordenadas do ponto de referência no ECEF. Para o cenário examinado aqui (deslocamentos bidimensionais), a Equação 3.6 é desconsiderada e o valor da altitude expressa por ela é dado como 0.

3.2.2 Sensores

Como mencionado anteriormente, os sensores de interesse no veículo indicam as frequências de rotação de suas rodas e a sua velocidade de translação. Estes dados são transmitidos no barramento CAN em somente uma mensagem. Estes valores estão codificados nos 8 *bytes* reservados para dados. Na Tabela 3.2, indicam-se as informações codificadas nos *bytes* da seção de dados.

Tabela 3.2: Leituras dos sensores.

Bytes	**Roda**	**Informação**	**Unidade**
1 e 2	1 (Dianteira Esquerda)	Frequência de Rotação	Hz
3 e 4	2 (Traseira Esquerda)	Frequência de Rotação	Hz
5 e 6	3 (Traseira Direita)	Frequência de Rotação	Hz
7 e 8	4 (Dianteira Direita)	Velocidade de translação da roda e do carro	m/s

As frequências de rotação são obtidas por um sensor que identifica a quantidade de vezes que uma roda gira em torno de seu próprio eixo em um determinado período. É possível perceber, na tabela, que o único par de rodas em lados opostos que têm codificadas as frequências de rotação é o par 2 e 3, ou seja, as rodas traseira esquerda e traseira direita, sendo esse o par de frequências de rotação escolhido. Por fim, a leitura da roda 4, que indica a velocidade de translação do veículo, é utilizada como a velocidade.

As leituras extraídas do CAN também vêm acompanhadas de um *timestamp*, também assumindo a unidade como segundos e fazendo a medida no *Unix Time*. A marca de tempo do CAN apresenta duas casas decimais adicionais, em relação à do GPS. Um exemplo de leitura que pode ser extraída é: `1483559091.5779,2.41015625,2.3619791667,5.1080729167`, na qual os campos são, respectivamente, a marca de tempo (em segundos), que corresponde às 15:44 do dia 1/4/2017, a frequência de rotação da roda traseira esquerda (em Hz), a frequência de rotação da roda traseira direita (também em Hz) e a velocidade de translação do veículo (em m/s).

3.3 Procedimento de Estimativa

Com base na discussão anterior a respeito da posição geográfica, seus sistemas de coordenadas e dos valores dos sensores, pode-se elaborar um modelo para calcular a nova posição do veículo a partir dos valores dos sensores utilizados. O modelo completo envolve um submodelo matemático de navegação inercial, para identificar os deslocamentos feitos pelo veículo a partir dos sensores, métodos de pré-processamento para tratar os dados dos sensores usados no sistema e um pós-processamento para corrigir as estimativas.

3.3.1 Modelo de Navegação Inercial

Para auxiliar a identificar as medidas envolvidas no cálculo do deslocamento do veículo, a Figura 3.5 apresenta, no sistema de coordenadas NED, um movimento entre dois pontos.

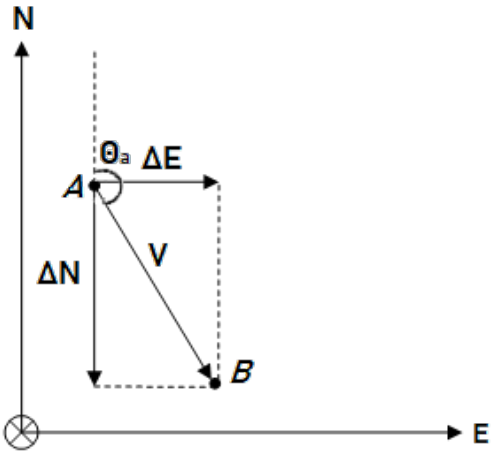


Figura 3.5: Deslocamento entre dois pontos no sistema NED.

Além do ângulo de rumo θ_a, apresentado na Seção 3.2.1, o deslocamento apresenta componentes no eixo leste e no eixo norte, além da sua velocidade. As relações entre todos estes elementos são:

$$\Delta N = V cos\theta_a \; t, \tag{3.7}$$

$$\Delta E = V sin\theta_a \; t, \tag{3.8}$$

onde ΔN, ΔE, θ_a, t e V são, respectivamente, as variações da posição nas direções norte e leste, o ângulo de rumo, a duração do movimento e a velocidade. A velocidade pode ser obtida por um sensor através do barramento CAN, como discutido anteriormente, e a duração do movimento é o tempo entre duas atualizações do GPS, ou seja, o valor do seu período de atualização. O objetivo é calcular os deslocamentos em cada direção, de modo a ajustar a última posição conhecida para que se obtenha a estimativa da posição atual. Para isto, o ângulo de rumo também é necessário, o que requer duas posições. Os cenários de interesse deste projeto são privados da cobertura do GPS. Portanto, obtê-lo diretamente não é possível. No entanto, pode-se estimar a sua variação através das frequências de rotação das rodas, dado que elas são indicadores de direção. Como as frequências de rotação diferem apenas numa curva, o indicador de que uma curva está sendo realizada, bem como da sua direção, também pode ser a diferença entre estas frequências, como indicado na equação a seguir:

$$\Delta f = f_{te} - f_{td}, \tag{3.9}$$

onde f_{te} e f_{td} representam, respectivamente, as frequências das rodas traseiras esquerda e direita. Neste caso, um valor positivo desta diferença (f_{te} maior) indica uma curva para a direita; um valor negativo (f_{td} maior) indica uma curva para o

sentido oposto. A estimativa da variação do ângulo de rumo é expressa, então, por:

$$\Delta\theta = \phi(\Delta f), \tag{3.10}$$

Se esta relação for aproximada para um modelo linear, ela pode ser representada como:

$$\Delta\theta \approx a\Delta f + b, \tag{3.11}$$

onde a e b são as constantes que devem ser ajustadas pelo modelo linear. O novo ângulo de rumo, então, pode ser obtido a partir do anterior:

$$\theta'_a = \theta_a + \Delta\theta, \tag{3.12}$$

sendo θ'_a o novo ângulo de rumo. Este resultado pode ser aplicado às Equações 3.7 e 3.8, obtendo-se o deslocamento do veículo. Por fim, a nova posição do veículo é determinada por:

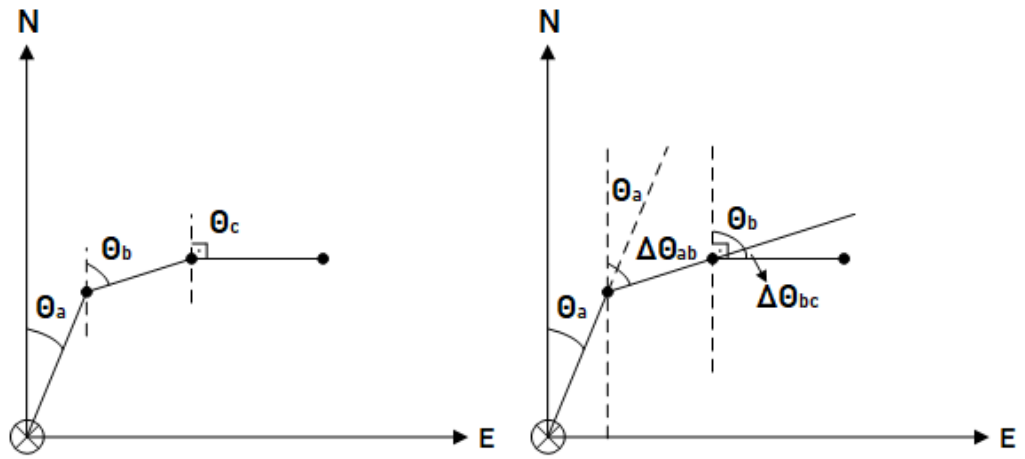
$$N' = N + \Delta N, \tag{3.13}$$

$$E' = E + \Delta E, \tag{3.14}$$

onde N' e E' são os novos valores da posição nos dois eixos. A razão pela qual as frequências das rodas são usadas para calcular a variação do ângulo de rumo, no lugar do próprio ângulo, é relacionada à própria definição de ângulo de rumo. Como mencionado anteriormente, este ângulo indica a direção do movimento em relação ao eixo norte, o que significa que um veículo pode trafegar em linha reta e apresentar um ângulo de rumo não nulo. Por outro lado, numa curva, o ângulo de rumo varia a uma taxa proporcional à sua intensidade. As Figuras 3.6a e 3.6b exemplificam isto. Nelas, pode se ver um objeto deslocando-se três vezes, realizando uma curva. No primeiro deslocamento, o ângulo de rumo é igual a θ_a; no segundo, igual a θ_b; e, no terceiro, igual a θ_c, como mostrado na Figura 3.6a. Após o deslocamento inicial, a partir da origem, o ângulo de rumo é alterado duas vezes: primeiro, para $\theta_b = \theta_a + \Delta\theta_{ab}$; por último, para $\theta_c = \theta_b + \Delta\theta_{bc}$, o que é indicado na Figura 3.6b.

3.3.2 Técnicas de Pré-Processamento

Antes de fornecer os dados a um algoritmo de Aprendizado de Máquina, é uma prática comum utilizar sobre eles técnicas de pré-processamento, que efetuam transformações convenientes aos valores, com o objetivo de reduzir o erro de predição. Os procedimentos usados são descritos a seguir.



(a) Apenas com os ângulos de rumo assumidos a cada instante.

(b) Destacando as variações sofridas pelo ângulo de rumo.

Figura 3.6: Deslocamento entre dois pontos em uma curva.

Normalização

A normalização é um procedimento de baixo custo computacional que ajusta a escala das variáveis de um conjunto de dados. A partir de N leituras de uma variável qualquer x, é calculada sua média amostral e, com este valor, o desvio padrão amostral de x, $\hat{\sigma}_x$, é obtido pela expressão:

$$\hat{\sigma}_x = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{\mu}_x)}, \tag{3.15}$$

onde

$$\hat{\mu}_x = \frac{1}{N}\sum_{i=1}^{N} x_i, \tag{3.16}$$

ou seja, a média de x. Cada valor x_i corresponde a uma observação de x. Com estes valores, realiza-se a seguinte transformação em x, gerando uma nova variável x':

$$x' = \frac{x - \hat{\mu}_x}{\hat{\sigma}_x}. \tag{3.17}$$

A nova variável x' possui média nula e desvio padrão unitário. Esta técnica, apesar de simples, é empregada em diversos problemas devido à sensibilidade de muitos modelos de Aprendizado de Máquina à escala das variáveis. O uso dela evita que variáveis com escalas maiores tenham maiores influências nas medidas de erro usadas no treino dos modelos, evitando que elas tenham uma importância desproporcional no ajuste dos parâmetros.

Filtragem

Assim como a escala dos valores pode afetar o desempenho de uma predição, precisões baixas podem afetar a acurácia do modelo. Precisões baixas são refletidas em oscilações consideráveis do valor durante suas observações no tempo. Estas oscilações tanto prejudicam o ajuste dos coeficientes de um modelo de regressão quanto prejudicam o cálculo de uma estimativa por um modelo já treinado. Como uma iniciativa a fim de aprimorar a acurácia das estimativas, a utilização de um algoritmo de filtragem de janela deslizante é proposta aqui.

Um filtro de janela deslizante analisa a evolução de uma série temporal, um conjunto de valores de uma variável x, que evolui com o tempo. A cada instante t, o filtro calcula uma operação agregadora sobre uma janela de tamanho W_s, ou seja, composta dos W_s valores mais recentes de x. Este valor é atribuído à leitura mais recente. Esta operação pode ser resumida pela seguinte equação:

$$x'(t) = \phi(x(t), x(t-1), ..., x(t-W_s+1)), \tag{3.18}$$

onde ϕ é a operação agregadora que se realiza. Duas candidatas a operações são consideradas aqui, dados os seus potenciais de revelarem uma tendência geral dos dados considerados: a média e a mediana. No caso da média, o filtro também pode ser chamado de média móvel. O filtro de média móvel pode ser definido pela equação a seguir, em função do tamanho da janela e para um instante de tempo arbitrário t:

$$x'(t) = \mu(x, W_s) = \frac{\sum_{i=0}^{W_s-1} x(t-i)}{W_s}. \tag{3.19}$$

A mediana é o ponto que separa as metades inferior e superior de uma amostra de valores. O seu cálculo depende se o tamanho da amostra é par ou ímpar. Quanto o tamanho da amostra é ímpar, o valor da mediana corresponde ao valor do elemento intermediário. Seja a lista de valores $L = \{x_1, x_2, ..., x_n\}$, contendo n elementos. Se n for um valor ímpar, para esta lista, a mediana pode ser calculada como:

$$m(L) = x_{\lfloor (n/2) \rfloor + 1}, \tag{3.20}$$

sendo a operação $\lfloor i \rfloor$, sobre um número real i, o cálculo do seu piso, ou seja, do maior número inteiro menor ou igual a i. Como exemplo desta operação, seja a lista de elementos $\{1, 2, 3, 4, 5\}$. A mediana dela é igual a 3, o elemento da terceira posição ($3 = \lfloor \frac{5}{2} \rfloor + 1$).

Caso o tamanho da amostra seja par, não há apenas um elemento intermediário, e sim dois. Neste caso, a mediana é a média entre eles. Para a mesma lista L definida anteriormente, se seu tamanho n for par, sua mediana pode ser definida

por:

$$m(L) = \frac{x_{n/2} + x_{(n/2)+1}}{2}. \tag{3.21}$$

Como exemplo deste cálculo, suponha a existência da lista de elementos $\{1, 2, 3, 4, 5, 6\}$. Para esta, visto que 3 e 4 são os elementos intermediários, a mediana é igual a $3,5$.

3.3.3 Técnicas de Aprendizado de Máquina

Nesta seção são descritos os algoritmos usados junto com o modelo de navegação inercial, discutido na subseção anterior, para obter as estimativas.

Regressão Linear

A Regressão Linear é um modelo de predição de relações lineares entre variáveis, que apresenta baixo custo computacional e é utilizado frequentemente na literatura. A descrição a seguir, sobre as propriedades deste modelo, está de acordo com o exposto por [13].

Um modelo de regressão linear expressa uma relação linear entre uma ou mais variáveis dependentes e um conjunto de variáveis independentes. Supondo apenas uma variável dependente e dado um vetor de variáveis independentes ou de entrada $\mathbf{x}^T = [x_1, x_2, ..., x_p]$ (aqui, transposto), estima-se o valor da variável dependente ou de saída y como:

$$\hat{y} = a_0 + \sum_{i=1}^{p} a_i x_i, \tag{3.22}$$

onde os coeficientes a_i correspondem aos pesos das variáveis independentes x_i e a_0 é um valor arbitrário, também chamado de coeficiente linear ou *bias*. Para apenas uma variável de entrada e uma variável de saída, o procedimento realizado é um ajuste de reta.

Outra representação do modelo pode ser feita adicionando-se o valor constante 1 ao início do vetor $\mathbf{x}$, juntando-se a_0 e os demais coeficientes a_i em ordem no vetor $\mathbf{a}$ e escrevendo-se a equação no formato vetorial, como um produto interno:

$$\hat{y} = \mathbf{x}^T \mathbf{a}, \tag{3.23}$$

em que $\mathbf{x}^T$ é uma transposição vetorial.

No caso de um modelo com múltiplas saídas, a equação pode ser expandida para um vetor de saídas $\mathbf{y}$. Neste caso, o vetor $\mathbf{a}$ se torna a matriz de coeficientes $\mathbf{A}$, em que cada coluna corresponde a uma variável dependente.

Existem diversos métodos para ajustar os coeficientes a_i. O mais frequentemente utilizado é o Método dos Mínimos Quadrados, que busca minimizar o erro quadrático

(EQ) através de operações com a sua derivada. Considerando um modelo com apenas uma variável dependente y, a existência de n pares entrada-saída (y_i, x_i), que compõem o conjunto de dados usado no treinamento do modelo, e que as estimativas calculadas para as saídas y_i a partir das entradas x_i sejam $\hat{y_i}$, escreve-se o EQ como:

$$EQ(\mathbf{a}) = \sum_{i=1}^{n} (y_i - \hat{y_i})^2. \tag{3.24}$$

Aplicando a Equação 3.23 a esta expressão, obtém-se:

$$EQ(\mathbf{a}) = \sum_{i=1}^{n} (y_i - x_i^T \mathbf{a})^2. \tag{3.25}$$

O EQ, por ser uma função quadrática dos parâmetros $\mathbf{a}$, sempre possui um valor mínimo. No entanto, é possível que ele não seja único. Reescrevendo-o em uma notação matricial, para múltiplas variáveis dependentes, temos:

$$EQ(\mathbf{a}) = (\mathbf{y} - \mathbf{X}\mathbf{a})^T (\mathbf{y} - \mathbf{X}\mathbf{a}), \tag{3.26}$$

em que $\mathbf{X}$ é uma matriz $n \times p$, cada linha sendo um vetor com valores para todas as variáveis de entrada, de um par entrada-saída, e $\mathbf{y}$ é um vetor de dimensão n, que representa todas as saídas do conjunto de treinamento. Derivando o EQ em relação a $\mathbf{a}$ e igualando esta derivada a 0 (a condição válida em um ponto de extremo), obtêm-se as equações normais:

$$\mathbf{X}^T (\mathbf{y} - \mathbf{X}\mathbf{a}) = 0. \tag{3.27}$$

Se $\mathbf{X}^T X$ não for uma matriz singular (isto é, se ela possuir uma matriz inversa), então a solução única é dada por:

$$\hat{\mathbf{a}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}. \tag{3.28}$$

As estimativas para as saídas podem, por fim, ser calculadas como:

$$\hat{y_i} = \hat{y}(x_i) = (x_i)^T \hat{\mathbf{a}}. \tag{3.29}$$

Neste projeto, o modelo de Regressão Linear é aplicado no modelo de navegação inercial apresentado na Seção 3.3.1, especificamente calculando a aproximação linear, dada pela Equação 3.11, da relação entre as frequências de rotação e a variação do ângulo de rumo.

Particionamento dos Dados por Análise de Agrupamentos

Em certos casos, convém particionar um conjunto de dados em partes com diferentes características e aplicar um tratamento individualizado a cada uma destas partes. No cenário deste projeto, isto envolve a aplicação de um modelo de regressão para cada um destes agrupamentos. A consequência disto é que cada um dos modelos de regressão se torna especializado em um tipo de comportamento, pois seus ajustes são mais finos em favor daquele tipo.

Como será visto mais adiante, com o conjunto de dados que foi coletado, este particionamento é uma estratégia conveniente: os dados possuem quantidades de leituras diferentes para trechos retos e curvas, o que é uma característica da trajetória realizada, que conteve mais trechos retos do que em curva. Para prevenir que este desequilíbrio entre leituras prejudique o desempenho do algoritmo de regressão dentro de um cenário, um conjunto de regressões lineares foi criado, cada uma sendo aplicada a um tipo de movimento. A quantidade de tipos de movimento que se pode separar é dependente do conjunto de dados - ou seja, de quantos padrões diferenciáveis existem nas suas variáveis - e do algoritmo usado para realizar esta separação. A quantidade de tipos de movimento que se adequou melhor a este problema e o algoritmo usado para descobrir os agrupamentos presentes nos dados são assuntos discutidos a seguir, nesta seção, e também no próximo capítulo. Duas maneiras de se segmentar o conjunto de dados são:

- separar manualmente, caso se saiba determinar quais critérios são interessantes e os limiares convenientes para a separação;
- separar utilizando um algoritmo de análise de agrupamentos ou *clusterização*, um algoritmo de Aprendizado de Máquina de Aprendizado Não-Supervisionado. Este busca conjuntos de valores de variáveis especificadas que seguem comportamentos parecidos, sem que se especifique em qual conjunto cada valor está.

Dado que há dificuldades em se estabelecer os limiares ideais para separar os conjuntos, a abordagem escolhida foi empregar um algoritmo de análise de agrupamentos, o K-Médias. Esta escolha foi feita por conta de seu baixo custo computacional, tanto em memória quanto em processamento. Esta é uma característica importante para conjuntos de dados grandes e para o caso de este sistema ser treinado em tempo real. O K-Médias será descrito a seguir.

Esta abordagem é semelhante a algoritmos do ramo do Aprendizado de Máquina conhecido como *Ensemble Learning*. Modelos deste ramo são dotados de acurácia e precisão aprimoradas e são criados a partir de um conjunto de modelos computacionalmente simples, que operam juntos, sendo organizados como se fossem um "comitê"

para tomar uma decisão. Esta técnica foi inicialmente empregada para problemas de classificação, mas tem sido adotada também em problemas de regressão [13].

K-Médias

Como mencionado, o K-Médias (ou *K-means*) é um algoritmo de análise de agrupamentos, isto é, um algoritmo que encontra subconjuntos, ou agrupamentos, de valores em um conjunto de dados, sem indicações dos agrupamentos aos quais cada valor deveria pertencer. Esta última característica é o que o diferencia de um algoritmo de classificação. Ele também é classificado como um algoritmo de quantização vetorial, empregado na área de processamento de sinais.

O processo executado no K-Médias é iterativo e encontra agrupamentos que minimizem a variância dentro de cada agrupamento, também chamada de variância intra-grupos. Esta variância é definida como:

$$V(\Omega) = \sum_{j=1}^{N} \sum_{i=1}^{K} u_i(x_j)||x_j - \omega_i||^2, \tag{3.30}$$

onde N indica a quantidade de pontos de dados disponíveis no conjunto de dados, K representa a quantidade de agrupamentos, x_j é um valor das variáveis consideradas, Ω é um vetor com os pontos centrais de todos os grupos, ω_i é o centro do agrupamento i e u_i é uma função de pertencimento de um registro x_j a um agrupamento G_i, que assume o valor 1 se $x_j \in G_i$ e o valor 0 se $x_j \notin G_i$. Em resumo, minimizar esta variância cria grupos nos quais os registros apresentam as menores distâncias possíveis em relação aos centros. Os centros e pontos podem ser representados em uma ou mais dimensões.

A quantidade de agrupamentos é definida pelo usuário. Dado um conjunto inicial de centros fornecido, o algoritmo opera alternando os passos a seguir:

- Para cada centro, identificam-se os pontos que sejam mais próximos dele do que são de outros centros. Estes pontos tornam-se membros do grupo deste centro;

- Para cada grupo, calcula-se a média dos valores, e o valor médio torna-se o novo centro.

O algoritmo executa até que um número máximo de iterações seja atingido ou até que a alteração no valor da variância intra-grupos após um passo seja menor que um limiar de tolerância. A distância pode ser calculada através de diversas métricas, sendo a mais comum a distância Euclidiana. Dados dois registros de tamanho n

$\mathbf{x} = (x_1, x_2, \ldots, x_n)$ e $\mathbf{y} = (y_1, y_2, \ldots, y_n)$, esta pode ser calculada como:

$$D(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}. \tag{3.31}$$

Para x e y unidimensionais, ela é representada mais simplesmente por:

$$D(x, y) = \sqrt{(x - y)^2} = |x - y|. \tag{3.32}$$

O K-Médias é classificado como um algoritmo guloso: em vez de calcular uma solução analítica para minimizar a variância intra-grupos, executa um procedimento iterativo que reduz, a cada passo, o valor desta métrica. Sendo assim, não é garantido que o mínimo global da função seja encontrado. É possível que a convergência ocorra em um ponto de mínimo local.

Devido a isto, uma preocupação ao se executar o algoritmo são os valores iniciais dados aos centros dos agrupamentos. É comum a utilização de centros iniciais aleatórios para escapar de mínimos locais. No entanto, pode ser necessária a reexecução do algoritmo para encontrar uma solução melhor. Para evitar a necessidade de repetições, neste trabalho utiliza-se um critério alternativo, o algoritmo K-Médias++, proposto por Arthur e Vassilvitskii [27]. Sua ideia geral é encontrar centros iniciais o mais afastados possível entre si, através de outro processo iterativo. Para executá-lo, define-se uma função de distância $D(x)$, que corresponde à distância de um registro x ao centro mais próximo dele, dentre os já escolhidos. Os passos do K-Médias++ são os seguintes:

- Do conjunto de registros X, escolher aleatoriamente um centro ω_1;
- Escolher o próximo centro c_i, selecionando $c_i = x' \in X$ com a probabilidade $\frac{D(x')^2}{\sum_{x \in X} D(x)^2}$;
- Repetir o passo anterior até que se gere a quantidade desejada de centros.

Este procedimento de inicialização torna mais provável que os centros iniciais estejam mais afastados. De acordo com o trabalho que propôs o algoritmo [27], este tipo alternativo de inicialização apresenta ganhos de desempenho, em relação ao K-Médias tradicional com centros aleatórios.

O K-Médias, como dito anteriormente, será usado para separar as leituras em diferentes conjuntos, que refletem tipos diferentes de movimento. O critério para esta divisão será o módulo da diferença entre as frequências de rotação ($|\Delta f|$), pois ela é a única variável de entrada dos algoritmos de regressão linear utilizados e porque ela é um indicativo da intensidade de uma curva. O módulo é utilizado porque o critério de interesse é a intensidade da variação na direção do veículo.

Assim, usando-se o módulo, pode-se ter mais leituras associadas a um conjunto de intensidades próximas, evitando que a família de modelos de regressão tenha baixa acurácia por uma escassez de dados, em tipos de movimentos menos frequentes.

3.3.4 Correspondência de Mapas

Como mencionado no capítulo 1, um algoritmo de correspondência de mapas, também conhecido como *map matching*, também é utilizado, com a finalidade de corrigir as estimativas geradas em uma etapa de pós-processamento.

Estes algoritmos ajustam para um mapa posições geográficas que recebem, fornecendo novas localizações corrigidas. Estas posições são estabelecidas sobre trechos definidos no mapa, como ruas ou estradas, próximos o suficiente da localização original. Vale observar que a acurácia da localização final depende da acurácia das posições dos locais representados nos mapas e da granularidade da informação representada. Quanto mais acurada for a representação do mapa e mais fina a granularidade, melhor será o desempenho. No caso de um mapa urbano que representa ruas, um mapa que consegue separar as faixas das ruas tem uma granularidade mais fina. Isto é interessante porque, ao se executar o *map matching*, a posição consegue ser ajustada à faixa da rua. Se não for possível distinguir as faixas da rua, a posição é ajustada para o seu meio, considerando a direção perpendicular a ela. Esta técnica pode ser utilizada inclusive para aprimorar a qualidade do posicionamento por satélite, mesmo em cenários nos quais não há problemas com a cobertura. O desempenho da localização também irá depender da frequência de amostragem das localizações. Quddus *et al.* [28] fazem uma revisão sobre diversos algoritmos de *map matching*, indicando suas vantagens e desvantagens.

Nesta proposta, utiliza-se a implementação de código aberto da ferramenta OSRM [29], um *software* cujo objetivo é fornecer os caminhos mais curtos entre dois pontos (roteamento), para fins de navegação. A base de dados utilizada como mapa é o *OpenStreetMap* [30], iniciativa livre e gratuita, com uma licença aberta, que fornece um mapa digital mundial. O mapa dado por esta iniciativa fornece informações das ruas sem indicar as faixas de cada rua. Ou seja, como mencionado anteriormente, um procedimento de *map matching* operado com este mapa ajusta as posições para o ponto médio das ruas, considerando-se o eixo perpendicular a elas.

O algoritmo de correspondência de mapas implementado no OSRM é a proposta de Newson e Krumm [31]. Os autores desenvolveram um Modelo Oculto de Markov, modelando um conjunto de processos que percorrem um caminho, por múltiplos estados com medidas incertas, sendo algumas transições entre estados mais prováveis que outras. Na implementação dos autores, os estados são os segmentos das ruas em

que o veículo pode estar, e as medidas associadas a eles são as posições geográficas fornecidas por um GNSS. Como as transições podem ser simplesmente modeladas como a passagem de um segmento a outro, esta representação é conveniente para o problema. Dentro do conjunto de estados possíveis, estão apenas trechos com uma distância máxima de 200 m da posição atual. O modelo destes autores requer um conjunto de no mínimo duas localizações geográficas para operar.

Com apenas uma posição, a implementação reverte para um simples algoritmo geométrico, escolhendo a posição ajustada em um dos segmentos próximos que seja o menos distante possível da localização original. Ainda se exclui, nesta consideração, segmentos com mais de 200 m de distância da posição original. A implementação do OSRM fornece, além da posição corrigida escolhida, outras possíveis correções, acompanhadas dos identificadores dos seus segmentos, de modo que se pode fazer uma avaliação das correções ou uma depuração.

3.3.5 Procedimento Completo

Com base na discussão realizada neste capítulo, pode-se estabelecer uma descrição resumida dos procedimentos que serão efetuados no treinamento e no teste. No treinamento do modelo, os seguintes passos são realizados:

- Coleta do conjunto de dados;
- Conversão das posições para o sistema NED, conforme descrito na Seção 3.2.1;
- Execução da filtragem e da normalização sobre os valores da diferença entre as frequências de rotação (Δf);
- Treinamento do modelo K-Médias, usando como inicialização o algoritmo K-Médias++ (Seção 3.3.3). É fornecida uma quantidade de agrupamentos e o critério de separação é o módulo da diferença entre as frequências de rotação ($|\Delta f|$);
- Criação e treino de um conjunto de algoritmos de Regressão Linear (Seção 3.3.3), de mesma cardinalidade que o conjunto de agrupamentos. Para isto:
  - Para cada leitura de entrada, usa-se a estrutura de agrupamentos criada para identificar o agrupamento ao qual ela pertence;
  - A Regressão Linear associada ao agrupamento da leitura é treinada com a leitura. Os demais modelos não sofrem alterações.

Ao fim do procedimento, o resultado é uma estrutura de agrupamentos dada pelo K-Médias e uma família de modelos de Regressão Linear que modelam a relação entre a diferença entre as frequências de rotação das rodas (Δf) e a variação do ângulo de rumo ($\Delta\theta_a$), sendo a primeira variável independente, e a segunda dependente. A Figura 3.7 apresenta um fluxograma que resume o procedimento de treino descrito.

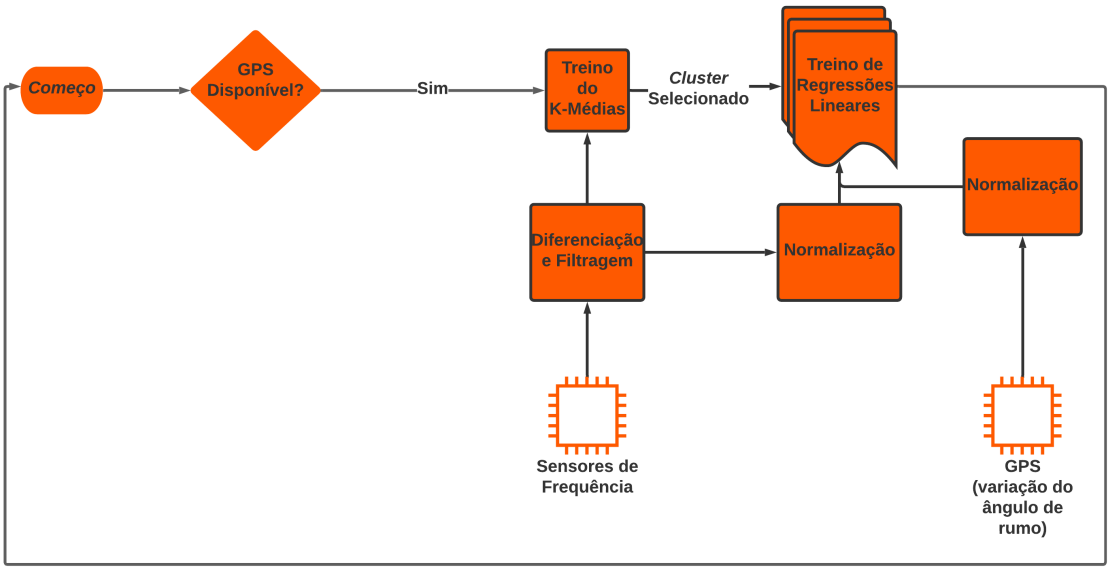


Figura 3.7: Fluxograma ilustrando o procedimento realizado no treino.

Na etapa de testes, simula-se uma falha de recepção de informações do GPS, então somente são recebidas leituras dos sensores e dispõe-se da última localização obtida pelo GPS e do último ângulo de rumo θ_a. O procedimento é o seguinte:

- Recepção de uma leitura com dados dos sensores de frequências de rotação e de velocidade;
- O valor recebido de Δf é filtrado e normalizado;
- Após este tratamento, ele é usado para identificar o tipo de movimento;
- O modelo de Regressão Linear relativo ao agrupamento escolhido é usado para calcular $\Delta\theta_a$ em função de Δf;
- Com $\Delta\theta_a$, obtém-se o novo ângulo de rumo, θ'_a e, a partir do mesmo, obtém-se os deslocamentos nas direções norte e leste, ΔN e ΔE;
- Com ΔN e ΔE, obtém-se a nova posição, representada por suas novas coordenadas no sistema NED, N' e E';
- N' e E' alimentam o algoritmo de *map matching*, gerando uma estimativa de posição final, com coordenadas N_{final} e E_{final}.

A Figura 3.8 apresenta um fluxograma que apresenta os passos do procedimento de testes descrito, indicando também os dados usados em cada etapa.

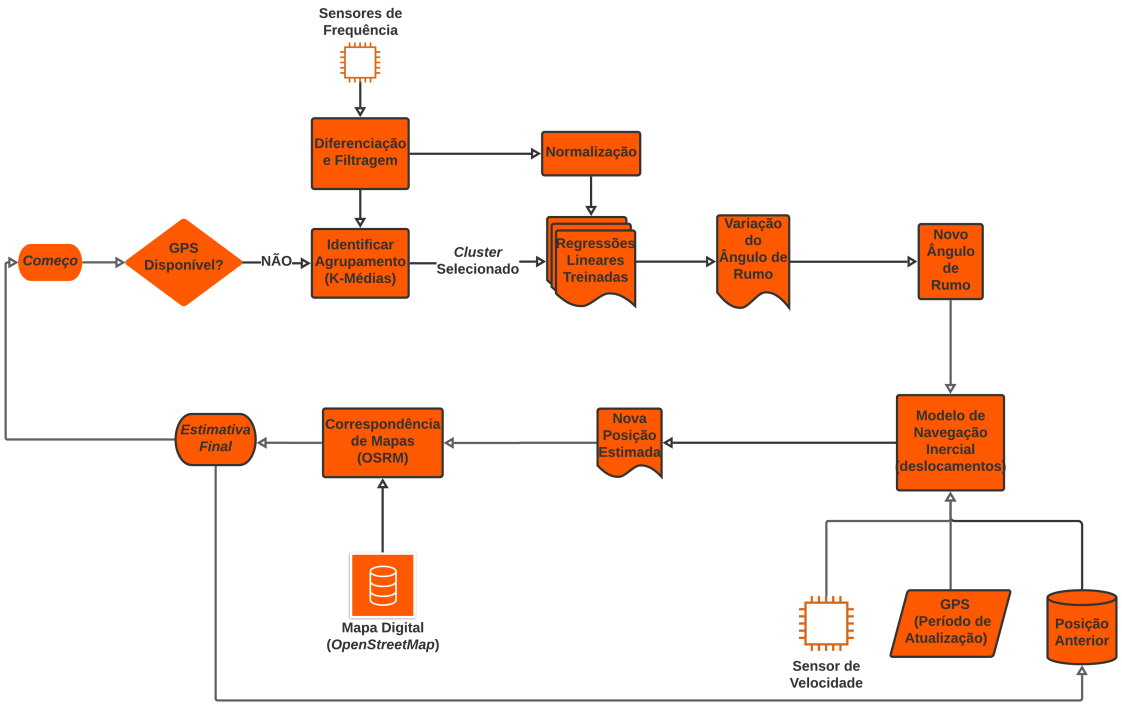


Figura 3.8: Fluxograma ilustrando o procedimento realizado no teste.

A metodologia discutida durante este capítulo, usada na implementação do sistema, será avaliada a seguir, no próximo capítulo. Para isto, também foi necessário realizar a coleta de dados reais do GPS e dos sensores do CAN, procedimento também detalhado a seguir.

Capítulo 4

Resultados e Discussões

Este capítulo se dedica a descrever a aplicação do procedimento descrito no capítulo anterior, voltado ao cálculo de estimativas de posições, bem como a análise do seu desempenho. Primeiramente, é discutido o procedimento de coleta dos dados em um cenário real, seguido da análise dos dados coletados. Posteriormente, descreve-se a execução das simulações, que visam avaliar o sistema proposto, observando-se as métricas de avaliação, a divisão do conjunto de dados em seções de treino e teste, e os parâmetros utilizados. Por fim, os resultados são exibidos e comentados.

4.1 Coleta de Dados

Para coletar os dados do GPS e dos sensores, foi realizado um procedimento de coleta de dados de um cenário real - o *campus* da Universidade Federal do Rio de Janeiro, na Ilha do Fundão. Para isso, um carro percorreu o local enquanto uma Unidade de Bordo estava conectada ao seu barramento CAN com um conector OBD-2, conforme descrito na Seção 3.1. Uma aplicação em Python foi desenvolvida para coletar periodicamente os dados enquanto o veículo percorria a região da coleta.

Durante o procedimento, as condições eram as ideais: era um dia ensolarado, com o céu sem nuvens; além disso, a região não continha obstáculos que pudessem causar prejuízos à cobertura do sinal do GPS, dado que o local era uma área predominantemente aberta. Dado que as condições atmosféricas e a presença de obstáculos afetam a acurácia do GPS e a sua possibilidade de obter novas posições, estes parâmetros são dignos de nota e as condições da coleta eram as melhores para coletar dados para treinar os modelos. A Tabela 4.1 exibe as principais características do conjunto de dados coletado, indicando a quantidade de leituras obtidas pelo GPS e pelo CAN, o tamanho em *bytes* do conjunto de dados, a duração do procedimento de coleta e os valores coletados, e a Figura 4.1 mostra a trajetória percorrida pelo veículo, representada com o auxílio da plataforma *Google Maps*. A trajetória tem um comprimento aproximado de 3,6 km e foi percorrida no sentido anti-horário,

Tabela 4.1: Principais características do conjunto de dados coletado.

Leituras do GPS	2107
Leituras do CAN	13614
Tamanho	796,8 kB
Duração da Coleta	544 s
Características	Marca de Tempo (GPS e CAN) Latitude, Longitude, Frequências de Rotação (Rodas Traseiras Esquerda e Direita), Velocidade

com velocidades até 51 km/h. Nota-se que há uma quantidade significativamente maior de leituras do CAN que de leituras do GPS. Isto é devido aos seus períodos de amostragem, respectivamente, 39,9 ms e 250 ms.

Figura 4.1: Trajetória percorrida na coleta de dados (Fonte: *Google Maps*).

4.2 Análise do Conjunto de Dados

As Figuras 4.2 e 4.3 representam, de forma gráfica, os dados coletados. Para os dois mapas, o eixo horizontal representa o eixo leste, em metros, e o vertical representa a direção norte, na mesma unidade. A Figura 4.2 sinaliza os valores de Δf, a diferença entre as frequências de rotação das rodas traseira esquerda e traseira

direita, indicadas por uma escala de cores. Valores negativos indicam curvas para a esquerda, que são identificadas por tons azulados; quantidades positivas são indicadoras de curvas à direita, representadas por tons avermelhados; e movimentos em linha reta ou de muito baixas variações na direção são indicados por cores próximas ao marrom, que indicam baixos valores absolutos. Já a Figura 4.3 mostra os valores da variação do ângulo de rumo ($\Delta\theta_a$) obtidos. Aqui, é usada uma legenda, no lugar de uma escala de cores. Valores inferiores a $-1,5°$, indicadores de movimentos à esquerda, são exibidos em azul, valores superiores a $+1,5°$, indicando curvas à direita, são representados por pontos vermelhos e quantidades situadas entre estes dois limites, que são movimentos em linha reta ou de alterações desprezíveis na direção, são mostradas em preto. O critério indicado por esses limites em graus é arbitrário e foi empregado apenas com a finalidade de aprimorar a visualização. Ele não foi empregado na implementação do sistema, dado que poderia afetar a acurácia do aprendizado, por parte do algoritmo de cálculo de estimativas, da relação entre diferenças de frequências e variação do ângulo de rumo. Alguns pontos coloridos podem ser vistos em trechos aparentemente retos: isto pode indicar mudanças de faixa, movimentos em zigue-zague ou erros de posição do GPS. Para diferenciar cada um destes casos, deve-se observar se o veículo mostra novos valores não desprezíveis da variação do ângulo de rumo ou não, e, caso isto ocorra, se o sinal desta variação altera periodicamente. Uma comparação visual dos dois mapas evidencia a correlação entre Δf e $\Delta\theta_a$, que é a característica que permitiu a concepção do algoritmo discutido no capítulo anterior.

Uma segunda análise pode ser feita sobre os valores de Δf, relativa à sua precisão. As Figuras 4.4 e 4.6 apresentam mapas de trechos em linha reta e com uma curva, de onde se coletou valores de Δf, apresentados nas Figuras 4.5 e 4.7. Nos mapas, os trechos em vermelho são os trechos do trajeto usados em cada análise, e os segmentos em preto correspondem ao restante do circuito. O eixo horizontal corresponde à direção leste, e o eixo vertical é relativo à direção norte. Nas Figuras 4.5 e 4.7, o eixo horizontal indica o tempo e o eixo vertical apresenta Δf; em preto, estão os valores originais, em vermelho eles estão filtrados com a média e em azul eles são mostrados filtrados pela mediana. Os filtros foram usados com janelas deslizantes de tamanho 5. Ainda que o valor absoluto dos desvios possa ser considerado baixo, é possível notar uma diferença significativa entre os valores filtrados e os valores não filtrados. Além disto, o filtro de mediana, ao menos em uma comparação visual, mostra uma maior capacidade de suprimir picos, dado que as leituras filtradas pela média não são totalmente cobertas no gráfico pelas leituras filtradas pela mediana. A comparação entre os gráficos do trecho em linha reta e do trecho em uma curva também mostram, mais uma vez, que há alterações em Δf quando se executa uma curva. Nas próximas seções, será discutido o tamanho mais eficiente de janela, bem

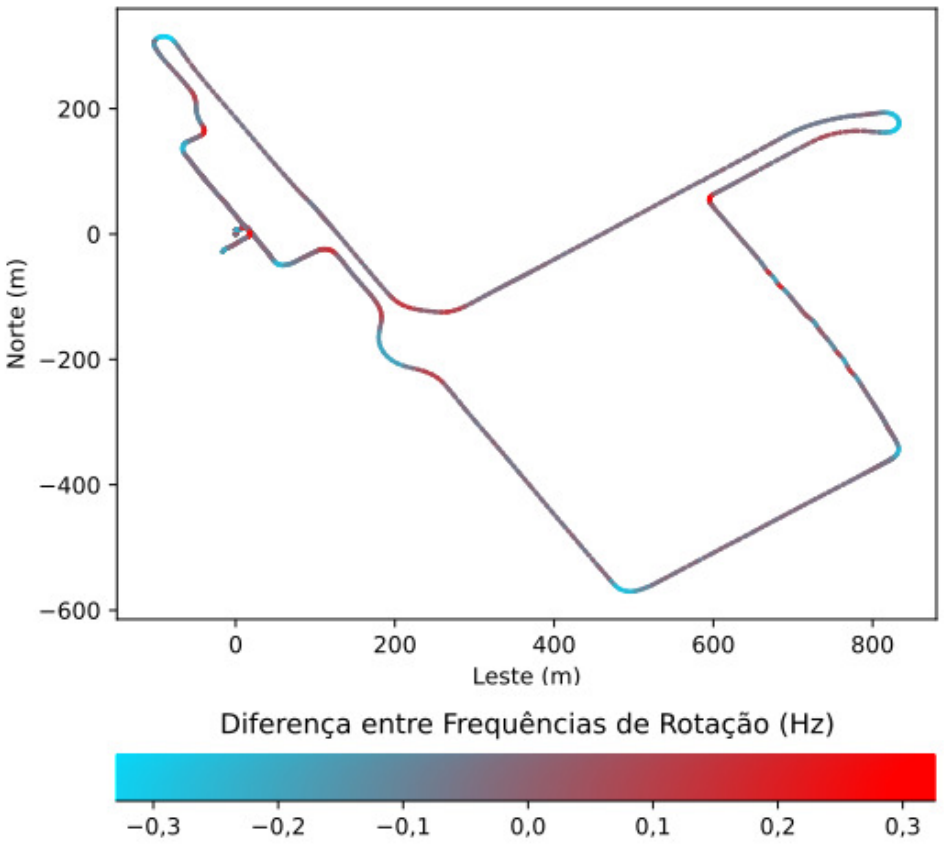


Figura 4.2: Mapa de calor que indica as diferenças de frequências.

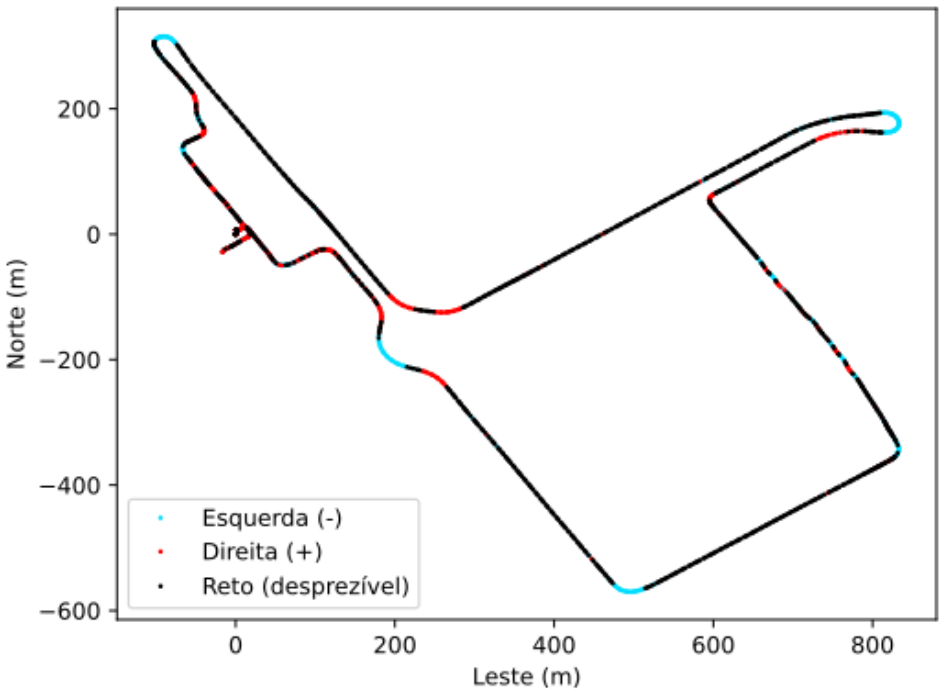


Figura 4.3: Mapa que sinaliza as variações do ângulo de rumo.

como qual das duas operações foi escolhida para compor o filtro.

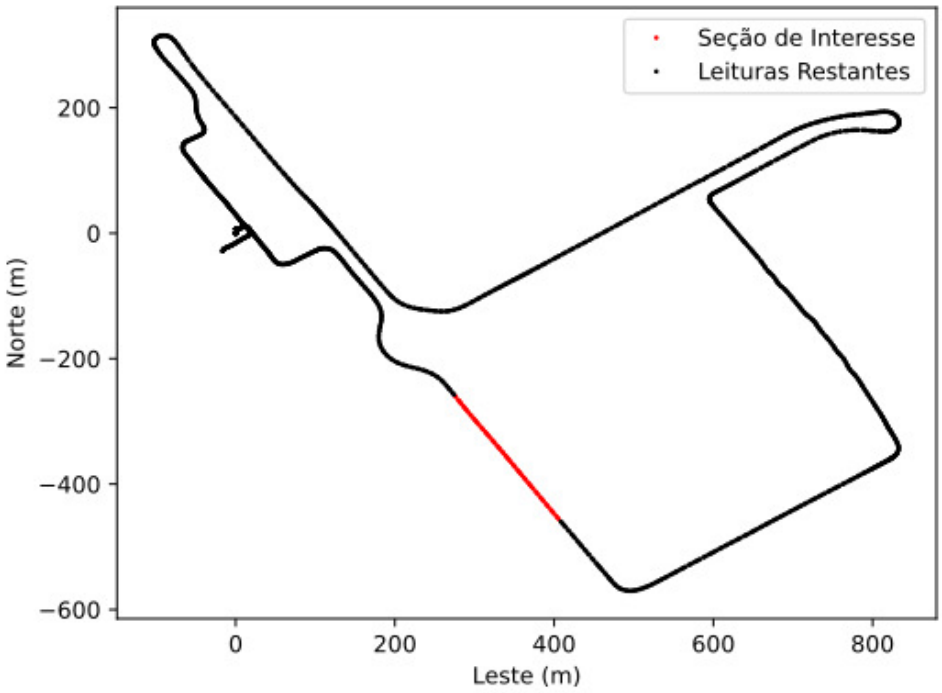


Figura 4.4: Mapa do trecho em linha reta onde ocorreu a filtragem.

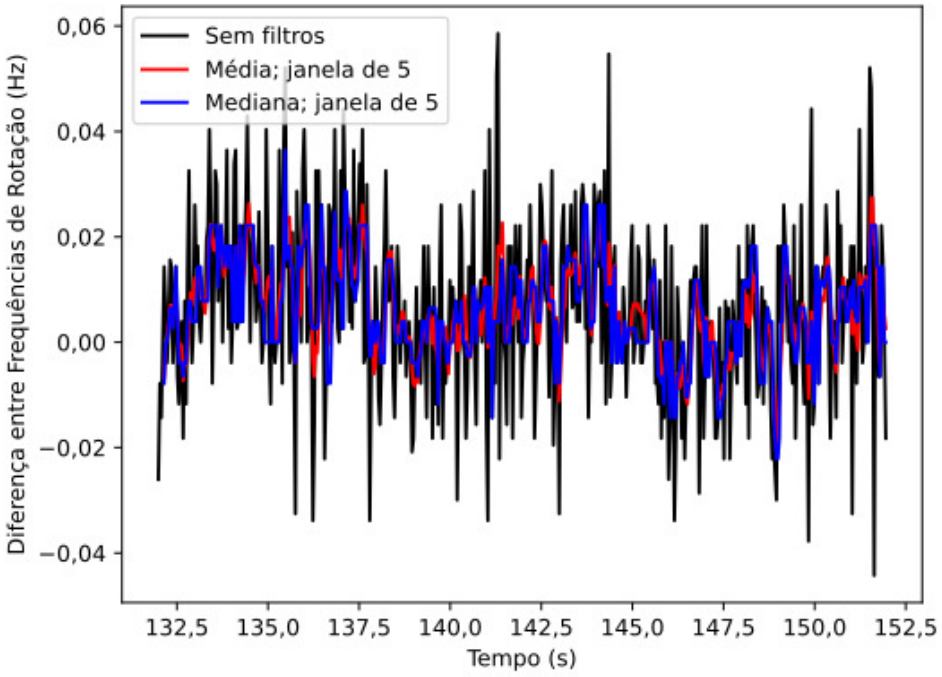


Figura 4.5: Leituras do trecho em linha reta, com o resultado da filtragem.

4.3 Divisões dos Dados para Validação dos Modelos

Para avaliar o desempenho das previsões dadas por modelos de Aprendizado de Máquina, processo também chamado de validação, o conjunto de dados deve ser separado em duas partes: um conjunto de treinamento e outro de testes. Como os

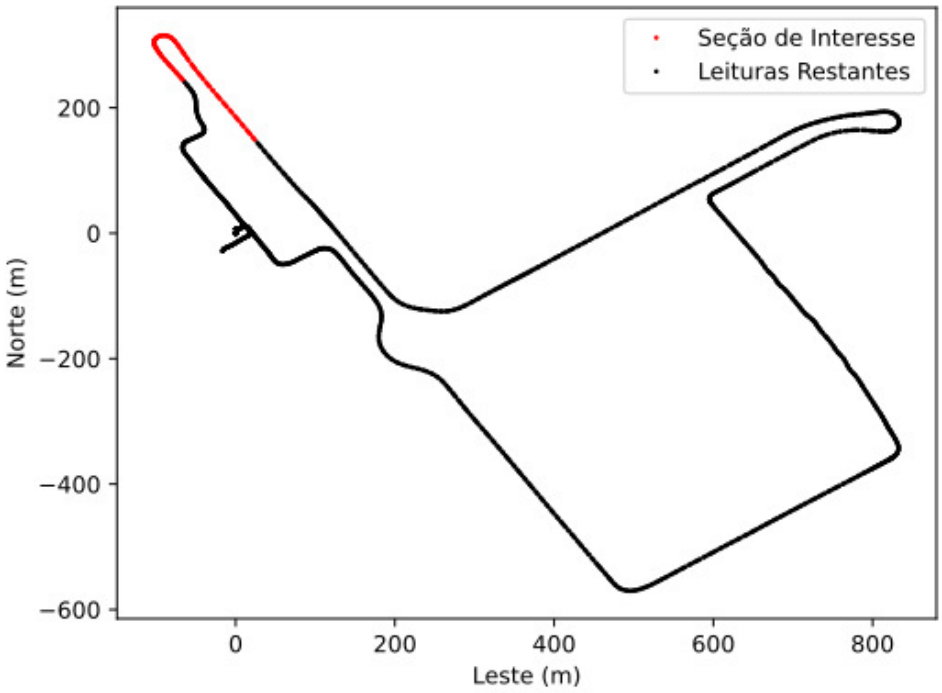


Figura 4.6: Mapa do trecho em curva onde ocorreu a filtragem.

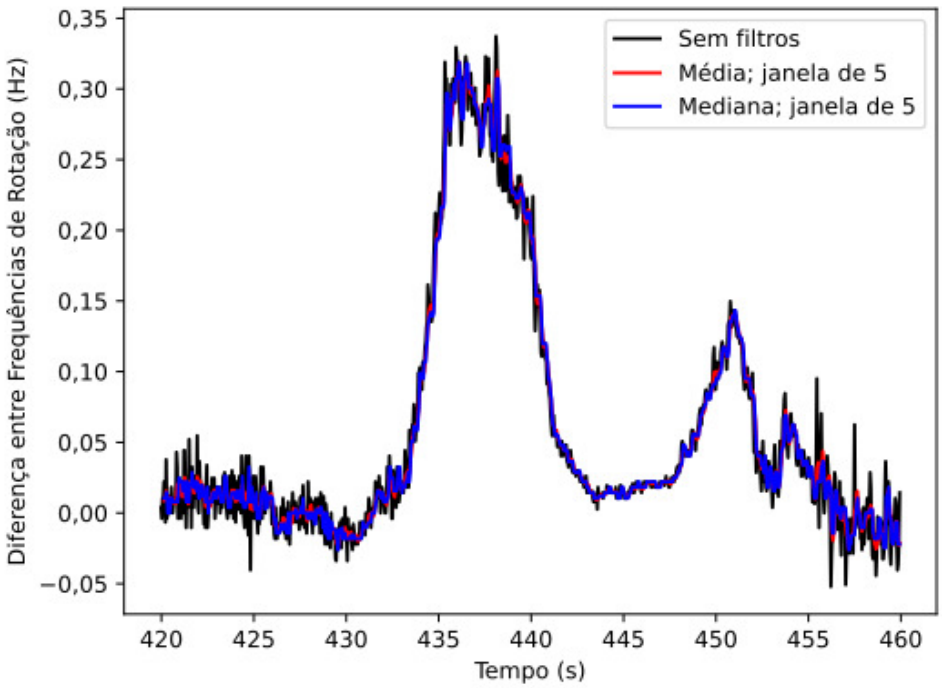


Figura 4.7: Leituras do trecho em curva, com o resultado da filtragem.

nomes já indicam, o conjunto de treinamento corresponde aos dados usados para ajustar o modelo e o de testes contém dados não empregados no treino, usados para calcular estimativas que são comparadas com os valores reais, verificando-se, assim, o desempenho do algoritmo.

Como apresentado na descrição do algoritmo de Regressão Linear, o critério

para ajustar seus parâmetros é a minimização do erro quadrático. Como o ajuste realizado depende dos dados do conjunto de treinamento, o desempenho varia de acordo com os dados fornecidos. Por este motivo, existem diferentes métodos para escolher os conjuntos de treino e teste. Alguns dos principais são definidos a seguir.

4.3.1 Validação Simples

Esta é a maneira mais simples de particionar os dados. Escolhe-se uma porcentagem dos dados para compor o conjunto de treino, sendo as leituras restantes usadas nos testes. Os dados podem ser selecionados de forma aleatória ou não. Sendo aleatória, o método também é conhecido como Método *Holdout*. Uma fração recorrentemente utilizada é atribuir 70% dos dados ao conjunto de treinamento e os 30% restantes ao conjunto de testes.

4.3.2 Validação Cruzada

Este é um método iterativo, que gera várias divisões dos dados [13]. O objetivo com isto é verificar o desempenho dos algoritmos em conjuntos de dados com diferentes características, usando-se o mesmo conjunto de dados. O desempenho final de uma validação cruzada é a média dos desempenhos de todas as divisões. Há diversas categorias de validações cruzadas; as duas principais são descritas abaixo:

- *Leave-p-out*: funciona separando uma quantidade p, escolhida pelo usuário, de leituras para o conjunto de teste, sendo o restante o conjunto de treinamento. Todas as combinações possíveis são testadas. Isso significa que, sendo n a quantidade de leituras separadas do conjunto de dados e p a quantidade de leituras do conjunto de testes, $C_n^p = \frac{n!}{(n-p)!\ p!}$, ou seja, as combinações de n elementos tomados p a p, são a quantidade de rodadas realizadas. Isto torna o *leave-p-out* um método muito longo para muitos casos, sendo impraticável em alguns. Um exemplo: sendo $n = 100$ e $p = 30$, aproximadamente 3×10^{25} rodadas são executadas. Por testar todas as combinações possíveis, este método é considerado um tipo exaustivo de validação cruzada;

- *K-fold*: este procedimento separa o conjunto de dados em uma quantidade arbitrária de k partes de mesmo tamanho. A cada rodada, uma das partes é usada nos testes e as restantes formam o conjunto de treino. Este procedimento é muito mais rápido que o *leave-p-out*, pois apenas executa k rodadas. Por outro lado, consegue avaliar melhor a capacidade de generalização dos modelos de Aprendizado de Máquina que uma validação simples. Valores típicos para k são 4 e 10.

Dado que as eficiências dos algoritmos de Aprendizado de Máquina, que aqui são a Regressão Linear e o K-Médias, podem variar com as características do conjunto de dados, a Validação Cruzada se mostra uma alternativa interessante por executar testes independentes em mais de uma partição do conjunto de dados, gerando vários conjuntos de treino e testes com diferentes características. Por isto, ela foi aplicada neste projeto. Dado que o conjunto de dados apresenta uma grande quantidade de leituras, a versão *K-fold* foi escolhida. O valor típico de 4 partições do conjunto de dados foi escolhido.

Dado que o problema deste trabalho é estimar trajetórias a partir de informações passadas, a partição dos dados foi realizada de uma maneira diferente da habitual. Em validações cruzadas, costuma-se dividir os dados aleatoriamente entre as partições, sem considerar a ordenação das leituras no conjunto de dados. Já neste projeto, cada partição corresponde a um trecho contínuo relativo a um quarto da trajetória feita durante a coleta. Assim, pôde-se treinar e testar o procedimento da maneira como ele será aplicado num cenário real.

4.4 Simulações

Esta seção tem como objetivo detalhar as simulações usadas na avaliação do desempenho do sistema, descrevendo o cenário considerado, a divisão do conjunto de dados para treinar e testar o algoritmo proposto, os critérios usados para a avaliação e os resultados obtidos.

4.4.1 Descrição e Critérios de Avaliação

Com os dados coletados no procedimento da Seção 4.1, o procedimento para avaliar o modelo foi executado com simulações. O cenário representado pelas simulações é o seguinte:

- O veículo trafega normalmente nas regiões correspondentes ao conjunto de treino;
- Ao chegar à região de testes, ocorre uma falha do GPS e a informação de localização geográfica torna-se indisponível;
- Com a última posição geográfica (de coordenadas N e E) e o último valor para o ângulo de rumo θ_a, deve-se estimar a trajetória percorrida pelo veículo em toda a seção que corresponde ao conjunto de testes, somente com as informações dos sensores.

Este cenário comunica informações importantes do sistema, respondendo:

- Se o sistema consegue reproduzir com fidelidade as trajetórias do veículo;
- Por quanto tempo pode-se manter uma acurácia aceitável;
- Quão forte é o efeito da acumulação do erro com o tempo. Esta acumulação é causada pela reutilização de estimativas passadas, que fornece seus erros ao sistema - este problema foi enfrentado nos trabalhos de [14] e [15];
- Qual é a melhor configuração para os modelos do sistema, em termos de seus parâmetros.

Para identificar estas informações, que dependem da acurácia, é necessário definir um critério para se calcular o erro. Escolheu-se calcular a distância entre a posição real e a posição estimada, que permite estabelecer uma comparação com receptores GPS e estabelecer a adequação da plataforma a diferentes aplicações.

Estando as estimativas no sistema NED, como definido na Seção 3.2.1, o erro de estimativa pode ser calculado como a distância euclidiana entre a posição estimada e a posição real. Isto é, num instante t:

$$Err(t) = \sqrt{(E_{estimado}(t) - E_{real}(t))^2 + (N_{estimado}(t) - N_{real}(t))^2}, \tag{4.1}$$

sendo E e N as coordenadas respectivas aos eixos leste e norte. A partir do erro de estimativa, também pode ser calculado o erro médio (EM), indicador da acurácia:

$$EM = \frac{\sum_{t=0}^{n} Err(t)}{n}, \tag{4.2}$$

sendo n a quantidade de leituras contida no conjunto de testes. A melhor proposta de algoritmo será considerada aqui aquela que apresentar o menor valor para o erro médio.

Os parâmetros do sistema que podem ser variados para alterar sua configuração são, conforme a discussão dos capítulos anteriores:

- O número de agrupamentos, ou *clusters*, que o K-Médias deverá encontrar (n_c). Variado entre 1 (equivalente a não usar o algoritmo) e 10;
- O tipo de filtro escolhido, podendo empregar a média ou a mediana;
- O tamanho da janela do filtro, W_s, variada aqui entre 2 e 10;
- O intervalo entre consultas ao serviço de *map matching* (n_m), definido como a quantidade de estimativas realizadas antes da solicitação de uma nova correção. n_m foi variado de 1 a 10.

O algoritmo seguirá o procedimento descrito na Seção 3.3.5, variando-se os parâmetros acima. Cabe ressaltar aqui que o serviço de *map matching* foi usado para correções ponto-a-ponto, sem enviar um lote de dados. Desta forma, as correções seguem o procedimento geométrico mencionado anteriormente, que ajusta a posição para o trecho cuja leitura corrigida é a mais próxima da estimativa. De modo a evitar ajustes incorretos, caso o serviço encontre mais de um ajuste possível, a correção é descartada. Em avaliações preliminares, esta medida evitou alguns ajustes imprecisos. Conforme mencionado na Seção anterior, será usada uma Validação Cruzada *K-Fold* com quatro partições. Isto é, três serão usadas para o treino e a restante será usada para testes, gerando quatro permutações diferentes usadas para a avaliação do desempenho. Com esta divisão, os conjuntos de testes de todas as permutações têm a duração de 136 s, e os conjuntos de treino somam 408 s, dos 544 s totais que correspondem à duração da coleta de dados.

As simulações foram programadas em Python, com o auxílio das bibliotecas *scikit-learn* [32] - que contém implementações de técnicas de Aprendizado de Máquina - e *pyplot* [33] - para a elaboração dos gráficos.

4.4.2 Resultados

A partir de todas as combinações possíveis obtidas variando-se o número de clusters (n_c) entre 1 e 10, o intervalo entre consultas ao serviço de *map matching* (n_m) entre 1 e 10 (consultando-se o serviço após cada estimativa ou apenas após 10 estimativas) e o tamanho da janela do filtro (W_s) entre 2 e 10, o melhor caso encontrado corresponde ao uso de $n_c = 10$, $n_m = 10$ e um filtro de medianas com $W_s = 5$, que obteve um erro médio global de 19,1 m. Não houve diferenças significativas de acurácia com tamanhos de janela maiores que 5. Dada a quantidade de permutações, torna-se proibitivo exibir os resultados, textualmente e graficamente, de todos os casos examinados. Serão exibidos, portanto, os resultados do melhor caso.

A Figura 4.8 exibe a divisão em 10 *clusters* realizada. No eixo horizontal do gráfico, está representado o tempo, e no eixo vertical está a diferença entre frequências de rotação (Δf). Nota-se que a cor pertencente a um cluster pode aparecer em valores positivos e negativos, pois, como mencionado na descrição da operação do sistema (Seção 3.3.5), o critério para a divisão dos grupos é o módulo de Δf. Esta divisão foi usada para escolher qual modelo de regressão linear, dentre os 10 gerados, recebe um determinado valor de Δf, tanto para fins de treino quanto para fins de testes.

As Figuras 4.9, 4.10, 4.11 e 4.12 apresentam os resultados obtidos de forma gráfica. Os gráficos à esquerda são mapas que mostram os trechos usados para o treino, em preto, os trechos com os valores reais da seção de testes, em azul, e os

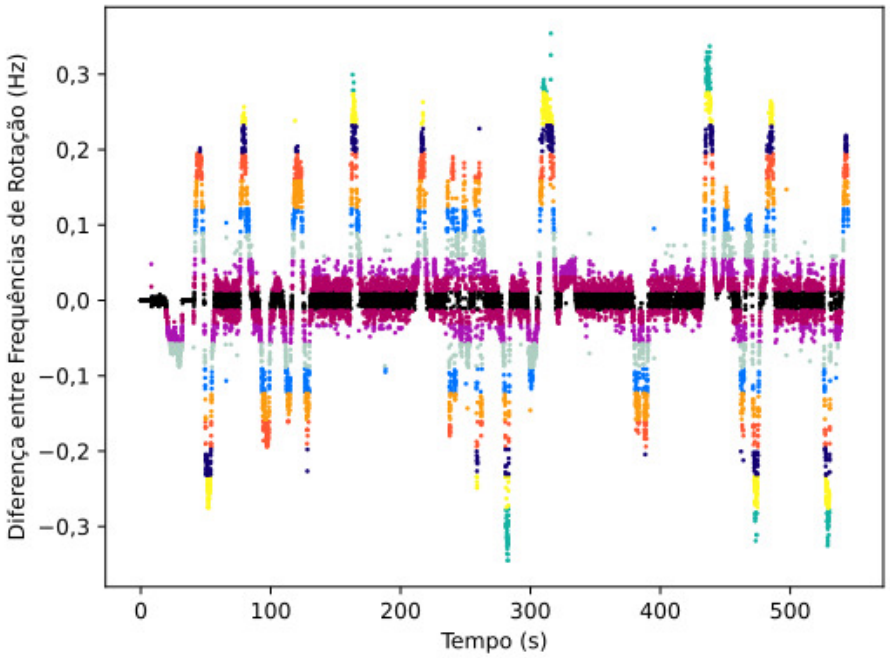
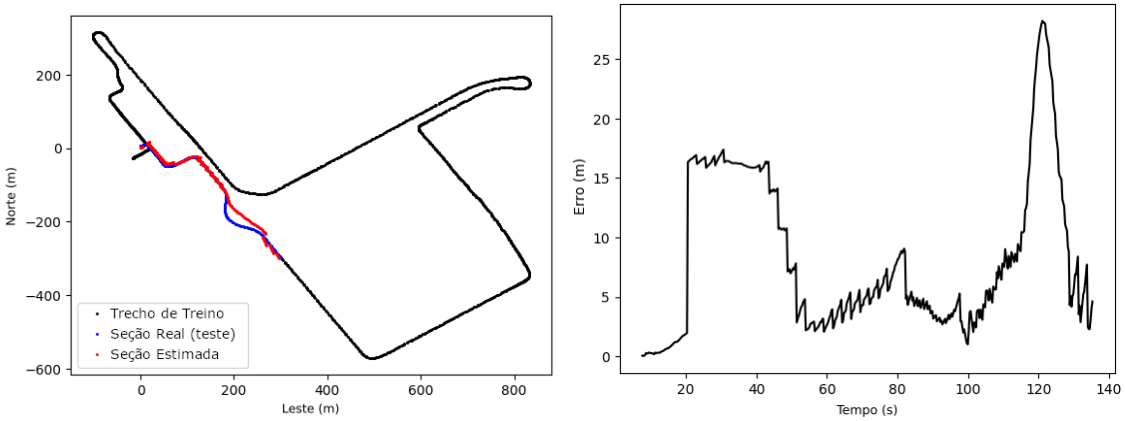


Figura 4.8: Divisão em 10 agrupamentos dos valores de diferença entre frequências de rotação, pelo K-Médias.



(a) Mapa comparativo.

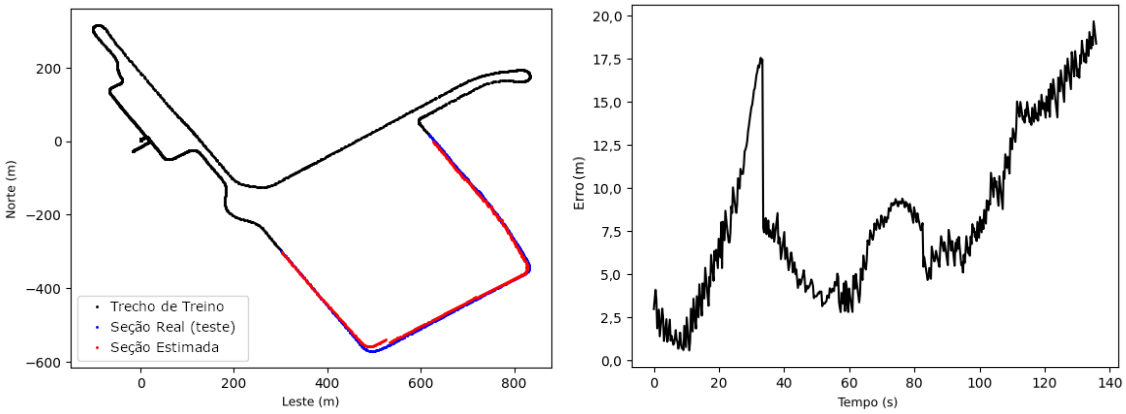
(b) Comportamento do erro.

Figura 4.9: Resultados para a permutação 1.

trechos com as estimativas da seção de testes geradas pelo algoritmo, em vermelho. O eixo horizontal representa a direção leste e o eixo vertical representa a direção norte. Vale lembrar que os trechos foram percorridos sempre no sentido anti-horário. Já os gráficos à direita apresentam a evolução do erro de estimativa com o tempo. O eixo horizontal indica o tempo, em segundos, e o vertical mostra o erro, em metros.

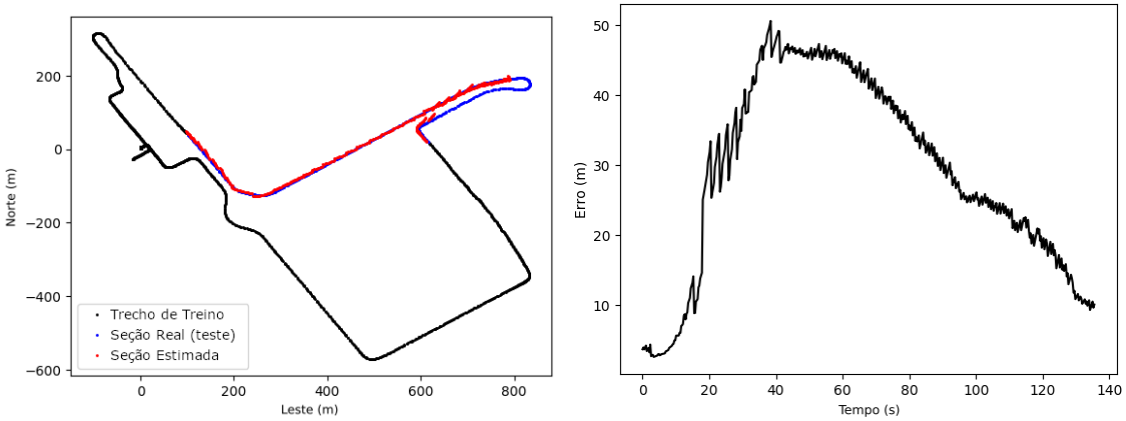
De todas as permutações, a primeira e a última apresentam as menores velocidades assumidas pelo veículo, pois correspondem ao início e ao fim do circuito, seções nas quais o veículo parte ou se aproxima do estacionamento.

Para a permutação 1 (Figura 4.9), o erro médio encontrado foi de 8,3 m. É



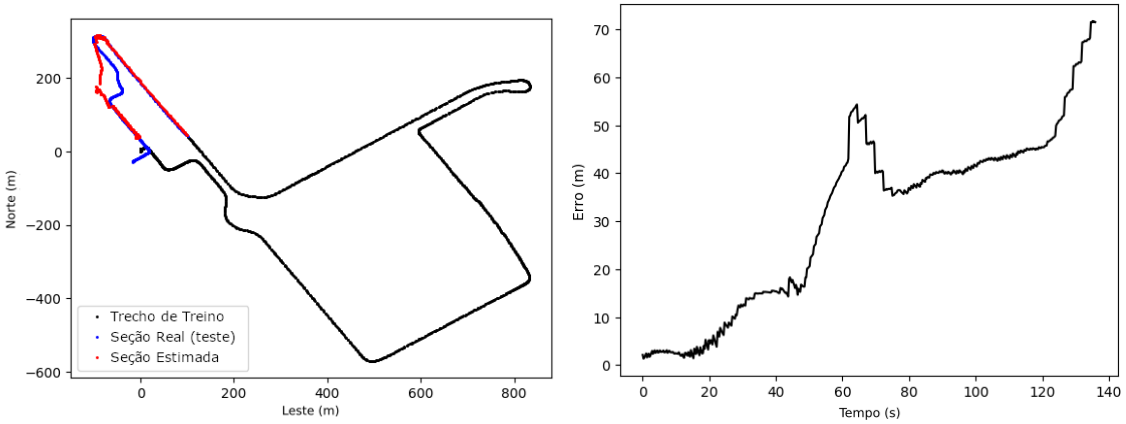
(a) Mapa comparativo. (b) Comportamento do erro.

Figura 4.10: Resultados para a permutação 2.



(a) Mapa comparativo. (b) Comportamento do erro.

Figura 4.11: Resultados para a permutação 3.



(a) Mapa comparativo. (b) Comportamento do erro.

Figura 4.12: Resultados para a permutação 4.

possível observar uma reprodução fiel do trajeto, exceto por uma curva em que ocorreu o ajuste da trajetória, pelo *map matching*, a uma rua que o veículo não

percorreu, embora seja próxima da trajetória real. O erro apresenta dois picos, sendo o último consequência deste ajuste impreciso. É interessante observar que após este evento, o sistema consegue recuperar sua acurácia, mantendo o erro abaixo de 10 m mesmo após uma falha de 136 s.

Na permutação 2 (Figura 4.10), o erro médio também foi de 8,3 m. Neste caso, não foram encontrados ajustes incorretos por parte da correspondência de mapas. No entanto, pode-se perceber uma tendência crescente do erro, diretamente proporcional ao tempo. Nota-se que a componente do erro que cresce em maior proporção é o erro na componente paralela às ruas, que não é ajustado pelo *map matching*. Isto é perceptível ao comparar-se a seção de testes real e a sua estimativa, dado que o ponto final da trajetória estimada localiza-se atrás do ponto final real. Este comportamento é o principal responsável pelo erro máximo de 20 m neste cenário.

Na terceira permutação, o erro médio é maior: 28,9 m. O principal fator que contribuiu para este aumento é um ajuste impreciso realizado entre a primeira e a segunda curvas feitas pelo veículo, no canto superior direito do mapa. É possível notar que a trajetória estimada foi ajustada para uma rua paralela próxima da rua onde o veículo de fato trafegava naquele momento. As trajetórias real e estimada se encontram posteriormente, o que provoca a queda do erro. Excetuando-se este erro de ajuste, é possível notar que, visualmente, a reprodução da trajetória apresenta fidelidade com a trajetória real. É um detalhe interessante a capacidade de recuperação do sistema após esta falha. Como é possível ver no canto direito do gráfico do erro, ao final da simulação, o erro decresce, chegando a menos de 20 m.

A quarta e última permutação apresenta o maior valor para o erro médio: 30,9 m. Além da acumulação do erro com o tempo, há um ajuste impreciso durante a execução de uma curva, levando o veículo para uma rua que não foi trafegada. Isto causa um pico próximo dos 60 segundos de teste. As trajetórias real e estimada se encontram novamente após o final do teste, causando uma baixa do erro. Posteriormente, o erro volta a subir, alcançando um máximo de 70 m. Vale ressaltar que, como mencionado anteriormente, o veículo, na seção final deste teste, apresenta as menores velocidades do trecho, pois está estacionando. Até cerca de 45 segundos, no entanto, o erro foi mantido abaixo de 20 m, o que permite ao sistema fornecer localizações aproximadas por uma duração interessante para algumas aplicações.

Pode-se depreender observações importantes a partir destes resultados:

- A acurácia obtida, com o erro médio de 19,1 m e um erro máximo de 70 m após uma falha de duração de 136 s, torna o sistema aplicável para obter trajetórias e localizações aproximadas, rastreamento de objetos e navegação ponto-a-ponto em ambientes com cobertura de satélites prejudicada;

- O sistema obteve este nível de acurácia sem utilizar comunicações externas, tornando-o independente de infraestrutura. Contudo, é possível que seu desempenho seja aprimorado por meio do uso de técnicas cooperativas de forma oportunista;
- O sistema consegue se recuperar de ajustes imprecisos do sistema de correspondência de mapas, o que é indicado nos gráficos da evolução dos erros em cada permutação por picos seguidos de quedas dos valores;
- Por outro lado, o nível de acurácia pode ser aprimorado para atender a mais tipos de aplicação:
  - Pode-se aprimorar o desempenho do sistema aprimorando-se o cálculo dos deslocamentos do veículo, dado que, como foi possível observar, a componente do erro que mais influenciou em sua acumulação foi a componente paralela à rua trafegada, dado que a componente perpendicular foi contida pelo *map matching*;
  - Outro ponto do sistema que pode ser aprimorado para alcançar uma melhor acurácia é a implementação da correspondência de mapas, uma vez que ela produziu alguns ajustes imprecisos.

Em relação aos trabalhos relacionados, uma comparação quantitativa relacionada aos valores de erro obtidos não é interessante, dado que vários fatores deles diferem dos deste trabalho:

- Os equipamentos utilizados para coletar os dados;
- As medidas geradas pelos sensores usados, bem como a acurácia e a precisão delas;
- A acurácia dos receptores GPS usados para coletar as posições;
- As formas de divisão dos dados entre conjunto de treino e teste;
- A duração das perdas de cobertura simuladas: as deste trabalho foram mais longas que as dos trabalhos relacionados apresentados aqui;
- Especificamente para o trabalho de Ahmed *et al.* [16], a plataforma de mapas utilizada e a granularidade da informação que ela apresenta: o OpenStreetMap não distingue faixas, enquanto o mapa usado por Ahmed *et al.* o faz.
- Por fim, os cenários da coleta de dados.

No entanto, pode ser feita uma comparação qualitativa, relacionada ao comportamento do erro com o tempo. Neste sentido:

- Em comparação com a proposta de Belhajem *et al.* [14], o sistema aqui proposto mostra uma evolução do erro em relação ao tempo mais gradual. Além disto, mesmo no pior caso (permutação 4), o erro não supera 100 m; em contraste a proposta de Belhajem *et al.* supera este valor em menos de 1 minuto;

- Comparando se à proposta de Ahmed *et al.* [16], o sistema deste trabalho não requer um período de convergência inicial para obter uma melhor acurácia. O trabalho de Ahmed *et al.* necessita de um período de aproximadamente 11 s até a acurácia alcançar a sua precisão sub-métrica. Em contraste, o projeto deste trabalho, dentro deste mesmo período de 11 s, consegue manter o erro abaixo de 10 m, possibilitando uma rápida resposta a falhas de um GNSS;

- Em relação à proposta de Pinto Neto *et al.* [15], a vantagem relacionada ao comportamento do erro se destaca ao final das trajetórias. Na proposta de Pinto Neto *et al.*, o erro tem um rápido aumento ao final de algumas trajetórias. Já para o sistema proposto aqui, na maioria das permutações consideradas, o erro consegue se manter num nível não muito distante do seu valor inicial. Mesmo após ajustes imprecisos do *map matching*, o sistema apresenta uma capacidade de recuperação, retornando o erro para valores de erro que são próximos aos valores antes destes ajustes.

Como visto pelos resultados e pela comparação visual das trajetórias, é possível reproduzir as trajetórias de modo satisfatório para algumas aplicações - obter percursos e localizações aproximados, rastreamento de objetos e navegação ponto-a-ponto em ambientes com cobertura de satélites prejudicada -, salvo os casos em que a correspondência de mapa realizou ajustes imprecisos.

Comparando-se com receptores GPS convencionais, que possuem acurácias típicas entre 5 m e 10 m [9] e atendem a uma gama de aplicações relacionadas à navegação ponto-a-ponto, entretenimento e rastreio de pessoas, objetos e veículos, e considerando cada uma das permutações da Validação Cruzada individualmente, é possível observar que:

- Para a permutação 1, este nível é atendido pelos 20 primeiros segundos e depois por mais aproximadamente 60 segundos, quando se encerra o primeiro pico do erro. Após o segundo pico, é possível observar que o sistema poderia manter este grau de acurácia mesmo após os 136 s da falha;

- Na segunda permutação, este nível de acurácia é mantido por aproximadamente 30 segundos. Após um valor de pico do erro, o sistema consegue voltar a este nível de acurácia por aproximadamente mais 70 segundos;

- Na terceira permutação, devido a um ajuste impreciso do *map matching*, este nível é atingido por apenas 15 segundos. Nota-se, no entanto, que ao final da falha, o valor do erro aproxima-se dos 10 m, dado que o sistema se recuperou dela;

- Na quarta permutação, este grau de acurácia é atingido dentro dos primeiros 20 s. Após isto, o ajuste feito a uma rua incorreta e a acumulação dos erros causaram um aumento do erro;

Além disto, o erro médio é de 19,1 m, maior que o erro médio de um receptor GPS convencional.

Um outro comparativo pode ser feito com aplicações de segurança do contexto veicular. Algumas exigem que a informação seja acurada o suficiente para distinguir a faixa em que um veículo está situado, como sistemas de detecção de mudanças de faixa ou de prevenção de ultrapassagens perigosas [34]. As faixas de uma rua têm larguras entre 2,5 m e 3,5 m, portanto, para atender a este tipo de aplicação em ambientes com coberturas prejudicadas do GPS, o sistema aqui proposto requer aprimoramentos.

Análises similares devem ser feitas comparando-se o grau de acurácia exigido por aplicações específicas com o comportamento do erro apresentado aqui.

Por fim, vale ressaltar que o uso de diferentes conjuntos de dados também influencia no desempenho deste sistema. Quanto mais abundantes forem as leituras para cada categoria de movimento executado do veículo, mais acuradas tendem a ser as estimativas feitas. Por isto, realizar mais coletas de dados para treinar os modelos aqui propostos também é uma medida interessante, desde que as coletas sejam feitas em condições ambientais favoráveis. Na presença de um tempo atmosférico instável e de obstáculos, a acurácia das leituras extraídas do GPS é prejudicada, o que pode prejudicar o treino dos modelos.

Capítulo 5

Conclusões

Esta dissertação de mestrado teve como proposta um sistema de navegação inercial para veículos terrestres, com o objetivo de fornecer a localização geográfica de um veículo, mesmo em ambientes em que a cobertura de sistemas de posicionamento por satélite é severamente prejudicada ou em que simplesmente não há linha de visada com os satélites. Exemplos de ambientes em que isto ocorre são áreas densamente arborizadas, com grandes quantidades de edifícios, acidentes geográficos e túneis. Esta funcionalidade permite a aplicações que dependem de localizações geográficas continuar operando, por durações que dependem dos seus requisitos de acurácia, nestes ambientes.

A partir de dados obtidos através do GPS, quando ele está operando, e de sensores de frequência de rotação das rodas traseiras e de velocidade, acessíveis através do barramento CAN – que são assumidos como a única informação de que se dispõe quando há uma falha de cobertura –, a proposta reúne:

- Conceitos de Aprendizado de Máquina: empregando o algoritmo da Regressão Linear para cálculo de estimativas, assim como o K-Médias para realizar a análise de agrupamentos que permite estabelecer diferentes tipos de movimento, cada um tratado por uma Regressão Linear diferente;

- Um modelo matemático de navegação inercial para calcular o deslocamento do veículo a partir dos valores dos sensores, facilitado por um sistema de coordenadas conveniente para representar as posições geográficas, o sistema NED;

- Técnicas de pré-processamento - normalização e filtragem por janela deslizante com medianas - para aprimorar a acurácia, evitando efeitos prejudiciais causados por desequilíbrios de escala entre as variáveis e por ruídos dos sensores;

- Como pós-processamento, a técnica de correspondência de mapas, ou *map matching*, que ajusta estimativas de posições às suas ruas mais próximas, numa

tentativa de conter o erro na componente perpendicular à rua. Esta técnica foi usada através de uma implementação de código aberto.

Dados reais foram coletados para averiguar o desempenho da proposta. A coleta foi realizada no *campus* da Universidade Federal do Rio de Janeiro, dirigindo-se um carro por um percurso de cerca de 3,6 km. Durante o percurso, foram coletados dados do GPS de uma Unidade de Bordo, equipamento para aplicações de veículos inteligentes, e dos sensores do veículo, através do barramento CAN. A análise de tais dados permitiu identificar uma correlação entre a diferença entre as frequências de rotação das rodas e a variação do ângulo de rumo, que indica a variação da direção do veículo. Os dois valores são alterados de modo diretamente proporcional na execução de uma curva. Esta correlação permitiu o desenvolvimento da estrutura anteriormente descrita.

A partir dos dados reais coletados, simulações foram realizadas com diferentes configurações em termos do tamanho da janela dos filtros, da operação realizada pelo filtro - média ou mediana -, da quantidade de agrupamentos buscados pelo K-Médias e da periodicidade de consultas ao serviço de *map matching*, chegando-se ao melhor caso. Este foi com um filtro de medianas com uma janela de 5 leituras de diferenças de frequências de rotação das rodas traseiras, 10 *clusters* e uma consulta ao serviço de correspondência de mapas a cada 10 estimativas (ou a cada 2,5 s, dado que o período de atualização do receptor GPS usado é de 250 ms). Para avaliar a capacidade do modelo de reproduzir uma trajetória sendo treinado com conjuntos de dados diferentes e de lidar com dados desconhecidos, um procedimento de Validação Cruzada foi realizado para dividir o conjunto em 4 partes, das quais 3 são para treino e 1 para testes, possibilitando 4 permutações. A melhor configuração do sistema obteve um erro médio de 19,1 m, considerando-se todas as permutações, e foi capaz de reproduzir os trajetos com razoável fidelidade, salvo nos casos em que houve ajustes imprecisos realizados pelo serviço de *map matching*, que levaram trajetórias a ruas incorretas, embora próximas das ruas de fato trafegadas. Após estes casos, vale ressaltar, o sistema consegue se recuperar posteriormente, o que é marcado pelo reencontro entre as trajetórias real e estimada em momentos posteriores. Também é um ponto interessante que o sistema, diferentemente da proposta de Ahmed *et al.* [16], não requer um período de baixa acurácia no qual ocorre uma convergência: o sistema pode fornecer posições imediatamente após a ocorrência de uma falha de recepção de um GNSS. Há uma acumulação do erro com o tempo em 2 dos 4 testes. No entanto, diferentemente da proposta de Belhajem *et al.* [14], o acúmulo ocorre de forma mais lenta, sendo mais pronunciado na direção paralela à rua, não corrigida pelo *map matching*. Os resultados mostram que, além de conseguir reproduzir trajetórias, o sistema, ao operar em conjunto com um GNSS, pode corrigir falhas pontuais, o que beneficia tais sistemas mesmo fora de ambientes muito

desafiadores para a cobertura dos satélites.

No estado em que a proposta se encontra, ela é adequada para obter trajetórias e localizações aproximadas em cenários com cobertura prejudicada de satélites, rastreamento de objetos e navegação ponto-a-ponto.

As direções futuras para este trabalho estão ligadas à melhoria da acurácia:

- A coleta de mais dados para enriquecer o conjunto de dados disponível. Como mencionado anteriormente, a quantidade de leituras disponíveis afeta a qualidade do procedimento de treino dos modelos e a acurácia das estimativas;

- Investigações de outros métodos de filtragem para os sensores, de modo a reduzir quaisquer componentes do erro de estimativa que sejam resultados dos seus ruídos;

- Testes com outros valores para os parâmetros avaliados: a quantidade de agrupamentos para os valores das diferenças de frequências, o tamanho das janelas dos filtros e o período entre consultas ao *map matching*;

- Ajuste dos modelos de Regressão Linear por outra técnica: aqui foi utilizado o Método dos Mínimos Quadrados, uma solução analítica obtida a partir da derivada do erro quadrático. No entanto, há outros procedimentos de ajuste de parâmetros que podem ser feitos em função do erro, como o Método do Gradiente ou heurísticas. A escolha do método de ajuste de parâmetros tem efeitos na acurácia do modelo final;

- Avaliações de técnicas da correção da velocidade do veículo. Por exemplo, a técnica avaliada por Zhang *et al.* [17] usa um sensor auxiliar (um acelerômetro) para realizar correções. Este ponto é interessante, visto que foi possível ver um acúmulo do erro na componente paralela à rua trafegada, não ajustada pelo *map matching*. Um exemplo que sinaliza isto é o final da trajetória da permutação 2 (Figura 4.9), no capítulo anterior, que indica que o ponto final da trajetória estimada se localiza alguns metros atrás do ponto final real;

- Mecanismos de corrigir imprecisões do receptor GPS usado: leituras mais acuradas de posições geográficas levam a valores mais próximos da real variação do ângulo de rumo do veículo, aprimorando a acurácia de um modelo de Aprendizado de Máquina que envolva esta medida. Se o modelo que relaciona esta medida à diferença de frequências de rotação das rodas traseiras, a variável de entrada usada, for mais acurado, a acumulação de erros será menor e a probabilidade de se efetuar um ajuste impreciso por parte do *map matching* é reduzida;

- Aprimoramentos do esquema de correspondência de mapas, dado que ele foi executado fornecendo-se um ponto por vez, caso no qual a localização é ajustada para a rua mais próxima, e dado que ele não incorpora dados de faixas às ruas. Estes dois motivos contribuem para os valores apresentados do erro de predição, dado que houve ajustes imprecisos, levando a trajetória estimada a ruas incorretas, e dado que veículos podem trafegar por faixas diferentes nas ruas. Sem incluir as informações de faixas à correspondência de mapas, as localizações sempre serão ajustadas para a faixa do meio e, se os veículos não estiverem originalmente nela, isto gerará um erro não desprezível de predição;

- Uso de localização cooperativa oportunística: no trabalho de Li *et al.* [19], envolvendo veículos submarinos, citado no Capítulo 2, o uso de comunicações com outros veículos foi benéfico para a acurácia do sistema. O uso de um mecanismo parecido na proposta deste trabalho, em casos nos quais há veículos próximos, pode fornecer uma informação adicional que pode ser usada para correções, de forma colaborativa.

Referências Bibliográficas

[1] WATTERSON, C. "Application Note 1123 - Controller Area Network (CAN) Implementation Guide". Analog Systems (`http://www.analog.com/`), 2017. Acessado em setembro de 2020.

[2] POPESCU, G. "Pixel geolocation algorithm for satellite scanner data", *Scientific Papers. Series E. Land Reclamation, Earth Observation & Surveying, Environmental Engineering, Vol. III, Bucharest*, pp. 127–136, 2014.

[3] NETO, J. B. P., GOMES, L. C., CASTANHO, E. M., et al. "An Error Correction Algorithm for Forward Collision Warning Applications". In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1926–1931, Nov 2016. doi: 10.1109/ITSC.2016.7795867.

[4] FLECK, J. L., CASSANDRAS, C. G., GENG, Y. "Adaptive Quasi-Dynamic Traffic Light Control", *IEEE Transactions on Control Systems Technology*, v. 24, n. 3, pp. 830–842, maio 2016. ISSN: 1063-6536, 1558-0865. doi: 10.1109/TCST.2015.2468181. Disponível em: <`http://ieeexplore.ieee.org/document/7229317/`>.

[5] FERREIRA, M., FERNANDES, R., CONCEIÇÃO, H., et al. "Self-organized traffic control". In: *Proceedings of the seventh ACM international workshop on VehiculAr InterNETworking - VANET '10*, p. 85, Chicago, Illinois, USA, 2010. ACM Press. ISBN: 978-1-4503-0145-9. doi: 10.1145/1860058.1860077. Disponível em: <`http://portal.acm.org/citation.cfm?doid=1860058.1860077`>.

[6] RAPELLI, M., CASETTI, C., SGARBI, M. "A Distributed V2V-Based Virtual Traffic Light System". In: *2020 International Conference on COMmunication Systems & NETworkS (COMSNETS)*, pp. 122–128, Bengaluru, India, jan. 2020. IEEE. ISBN: 978-1-72813-187-0. doi: 10.1109/COMSNETS48256.2020.9027339. Disponível em: <`https://ieeexplore.ieee.org/document/9027339/`>.

[7] NETO, J., GOMES, L., COSTA, L., et al. "Um Sistema Redundante de Frenagem Segura para Trens de Levitação Magnética". In: *Anais de XXXIV Simpósio Brasileiro de Telecomunicações*. Sociedade Brasileira de Telecomunicações, 2016. doi: 10.14209/sbrt.2016.111. Disponível em: <http://biblioteca.sbrt.org.br/articles/1174>.

[8] EMZIVAT, Y., IBANEZ-GUZMAN, J., MARTINET, P., et al. "Adaptability of automated driving systems to the hazardous nature of road networks". In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, Yokohama, out. 2017. IEEE. ISBN: 978-1-5386-1526-3. doi: 10.1109/ITSC.2017.8317638. Disponível em: <http://ieeexplore.ieee.org/document/8317638/>.

[9] ZOGG, J. *GPS: Essentials of Satellite Navigation - Compendium*. 1 ed. Suíça, U-blox AG, 2009. ISBN: 9783033021396.

[10] WANG, G., KUANG, C., CAI, C., et al. "Real-time PPP based on the Quasi-Zenith Satellite System MADOCA-LEX signal", *IET Radar, Sonar & Navigation*, v. 12, n. 5, pp. 494–498, maio 2018. ISSN: 1751-8784, 1751-8792. doi: 10.1049/iet-rsn.2017.0387. Disponível em: <https://digital-library.theiet.org/content/journals/10.1049/iet-rsn.2017.0387>.

[11] LI, L., ZHANG, Q., QI, G., et al. "Algorithm of Obstacle Avoidance for Autonomous Surface Vehicles based on LIDAR Detection". In: *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, pp. 1823–1830, Chengdu, China, mar. 2019. IEEE. ISBN: 978-1-5386-6243-4. doi: 10.1109/ITNEC.2019.8729203. Disponível em: <https://ieeexplore.ieee.org/document/8729203/>.

[12] GOMES, L. D. C., COSTA, L. H. M. K. "Vehicular Dead Reckoning Based on Machine Learning and Map Matching". In: *2020 IEEE 92nd Vehicular Technology Conference (VTC-Fall)*, p. 5, out. 2020. Aceito para publicação.

[13] HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J. *The Elements of Statistical Learning*. 2 ed. Stanford, Califórnia, Springer, 2009. ISBN: 978-0387848570.

[14] BELHAJEM, I., MAISSA, Y. B., TAMTAOUI, A. "Improving low cost sensor based vehicle positioning with Machine Learning", *Control Engineering Practice*, v. 74, pp. 168 – 176, 2018. ISSN: 0967-0661. doi: https://doi.org/10.1016/j.conengprac.2018.03.006.

[15] NETO, J. B. P., MITTON, N., CAMPISTA, M. E. M., et al. "Dead Reckoning Using Time Series Regression Models". In: *Proceedings of the 4th ACM MobiHoc Workshop on Experiences with the Design and Implementation of Smart Objects*, SMARTOBJECTS '18, pp. 6:1–6:6, New York, NY, USA, 2018. ACM. ISBN: 978-1-4503-5857-6. doi: 10.1145/3213299.3213305.

[16] AHMED, H., TAHIR, M., ALI, K. "Terrain Based GPS Independent Lane-Level Vehicle Localization Using Particle Filter and Dead Reckoning". In: *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, pp. 1–5, Sep. 2016. doi: 10.1109/VTCFall.2016.7881249.

[17] ZHANG, M., YANG, J., ZHAO, J., et al. "A Dead-Reckoning Based Local Positioning System for Intelligent Vehicles". In: *2019 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS)*, pp. 513–517, Shenyang, China, jul. 2019. IEEE. ISBN: 978-1-72813-720-9. doi: 10.1109/ICPICS47731.2019.8942565. Disponível em: <https://ieeexplore.ieee.org/document/8942565/>.

[18] "Waze Beacons". Disponível em: <https://www.waze.com/en/beacons>. Acessado em setembro de 2020.

[19] LI, Y., WANG, Y., YU, W., et al. "Multiple Autonomous Underwater Vehicle Cooperative Localization in Anchor-Free Environments", *IEEE Journal of Oceanic Engineering*, v. 44, n. 4, pp. 895–911, out. 2019. ISSN: 0364-9059, 1558-1691, 2373-7786. doi: 10.1109/JOE.2019.2935516. Disponível em: <https://ieeexplore.ieee.org/document/8826251/>.

[20] MA, L., TANG, L., XU, Y., et al. "Indoor Floor Map Crowdsourcing Building Method Based on Inertial Measurement Unit Data". In: *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, pp. 1–5, June 2017. doi: 10.1109/VTCSpring.2017.8108337.

[21] CHATTHA, M. A., NAQVI, I. H. "PiLoT: A Precise IMU Based Localization Technique for Smart Phone Users". In: *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, pp. 1–5, Sep. 2016. doi: 10.1109/VTCFall.2016.7881166.

[22] LIN SHEN, L., WONG SHUNG HUI, W. "Improved Pedestrian Dead-Reckoning-Based Indoor Positioning by RSSI-Based Heading Correction", *IEEE Sensors Journal*, v. 16, n. 21, pp. 7762–7773, nov. 2016. ISSN: 1530-437X, 1558-1748, 2379-9153. doi: 10.1109/JSEN.2016.2600260. Disponível em: <http://ieeexplore.ieee.org/document/7543502/>.

[23] DAISS, A., KIENCKE, U. "Estimation of vehicle speed fuzzy-estimation in comparison with Kalman-filtering". In: *Proceedings of International Conference on Control Applications*, pp. 281–284, Sep. 1995. doi: 10.1109/CCA.1995.555716.

[24] GREWAL, M., WEILL, L., ANDREWS, A. *Global Positioning Systems, Inertial Navigation and Integration*. Los Angeles, California, John Wiley & Sons, Inc., 2001. ISBN: 0-471-20071-9. doi: 10.1145/3213299.3213305.

[25] IVIS, F. "Calculating Geographic Distance: Concepts and Methods". `https://www.lexjansen.com/nesug/nesug06/dm/da15.pdf`, 2006. Acessado em setembro de 2020.

[26] "Digital Imaging and Remote Sensing Image Generation - Coordinate Systems". `http://www.dirsig.org/docs/new/coordinates.html`. Acessado em setembro de 2020.

[27] ARTHUR, D., VASSILVITSKII, S. "k-means++: The Advantages of Careful Seeding". In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics*, 2007.

[28] QUDDUS, M. A., OCHIENG, W. Y., NOLAND, R. B. "Current map-matching algorithms for transport applications: State-of-the art and future research directions", *Transportation Research Part C: Emerging Technologies*, v. 15, n. 5, pp. 312–328, out. 2007. ISSN: 0968090X. doi: 10.1016/j.trc.2007.05.002. Disponível em: <`https://linkinghub.elsevier.com/retrieve/pii/S0968090X07000265`>.

[29] LUXEN, D., VETTER, C. "Real-time routing with OpenStreetMap data". In: *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '11, pp. 513–516, New York, NY, USA, 2011. ACM. ISBN: 978-1-4503-1031-4. doi: 10.1145/2093973.2094062.

[30] "OpenStreetMap". Disponível em: <`https://www.openstreetmap.org`>. Acessado em setembro de 2020.

[31] NEWSON, P., KRUMM, J. "Hidden Markov map matching through noise and sparseness". In: *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '09*, p. 336, Seattle, Washington, 2009. ACM Press. ISBN: 978-1-60558-649-6.

doi: 10.1145/1653771.1653818. Disponível em: <http://portal.acm.org/citation.cfm?doid=1653771.1653818>.

[32] PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., et al. "Scikit-learn: Machine Learning in Python", *Journal of Machine Learning Research*, v. 12, pp. 2825–2830, 2011.

[33] HUNTER, J. D. "Matplotlib: A 2D graphics environment", *Computing In Science & Engineering*, v. 9, n. 3, pp. 90–95, 2007. doi: 10.1109/MCSE.2007.55.

[34] PINTO NETO, J. B., GOMES, L. C., ORTIZ, F. M., et al. "An accurate cooperative positioning system for vehicular safety applications", *Computers & Electrical Engineering*, v. 83, pp. 106591, maio 2020. ISSN: 00457906. doi: 10.1016/j.compeleceng.2020.106591. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0045790618326776>.