



Maria Fernanda Rodriguez Ruiz

Desenvolvimento de um Sistema de Localização Híbrido para Navegação Autônoma de Veículos Terrestres em Ambiente Simulado

89/2014

CAMPINAS
2014



**UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA**

Maria Fernanda Rodriguez Ruiz

Desenvolvimento de um Sistema de Localização Híbrido para Navegação Autônoma de Veículos Terrestres em Ambiente Simulado

Orientador: Prof. Dr. Janito Vaqueiro Ferreira

Coorientador: Prof. Dr. Arthur de Miranda Neto

Dissertação de Mestrado apresentada à Faculdade de Engenharia Mecânica da Universidade Estadual de Campinas, para a obtenção do título de Mestra em Engenharia Mecânica, na Área de Mecânica dos Sólidos e Projeto Mecânico.

ESTE EXEMPLAR CORRESPONDE À VER-
SÃO FINAL DA DISSERTAÇÃO DEFENDIDA
PELO ALUNO María Fernanda Rodríguez Ruiz, E
ORIENTADO PELO PROF. DR. Janito Vaqueiro
Ferreira.

Janito Vaqueiro Ferreira

ASSINATURA DO ORIENTADOR

CAMPINAS
2014

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

R618d Rodriguez Ruiz, Maria Fernanda, 1986-
Desenvolvimento de um sistema de localização híbrido para navegação autônoma de veículos terrestres em ambiente simulado / Maria Fernanda Rodriguez Ruiz. – Campinas, SP : [s.n.], 2014.

Orientador: Janito Vaqueiro Ferreira.
Coorientador: Arthur de Miranda Neto.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Engenharia Mecânica.

1. Simulação computacional. 2. Kalman, Filtragem de. 3. Navegação autônoma. 4. Veículos autônomos. I. Vaqueiro Ferreira, Janito, 1961-. II. Miranda Neto, Arthur de. III. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Development of a hybrid localization system for autonomous navigation of ground vehicles in a simulated environment

Palavras-chave em inglês:

Computer simulation

Kalman filter

Autonomous navigation

Autonomous vehicles

Área de concentração: Mecânica dos Sólidos e Projeto Mecânico

Titulação: Mestra em Engenharia Mecânica

Banca examinadora:

Janito Vaqueiro Ferreira [Orientador]

Ely Carneiro de Paiva

Claudio Garcia

Data de defesa: 26-08-2014

Programa de Pós-Graduação: Engenharia Mecânica

**UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
COMISSÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA
DEPARTAMENTO DE MECÂNICA COMPUTACIONAL**

DISSERTAÇÃO DE MESTRADO ACADÊMICO

**Desenvolvimento de um Sistema de Localização
Híbrido para Navegação Autônoma de
Veículos Terrestres em Ambiente Simulado**

Autor: Maria Fernanda Rodriguez Ruiz

Orientador: Janito Vaqueiro Ferreira

A Banca Examinadora composta pelos membros abaixo aprovou esta Dissertação:

Janito Vaqueiro Ferreira

Prof. Dr. Janito Vaqueiro Ferreira, Presidente
DMC/FEM/UNICAMP

Ely Carneiro de Paiva

Prof. Dr. Ely Carneiro de Paiva
DPM/FEM/UNICAMP

Claudio Garcia

Prof. Dr. Claudio Garcia
USP/São Paulo

Campinas, 26 de Agosto de 2014.

Dedicatória

Com tudo o meu carinho e amor, aos meus queridos pais Manuel e María Helena, e minhas irmãs Diana e Valentina, sempre minha maior motivação, minha fortaleza e toda minha inspiração.

De maneira especial a Camilo Ariza Zambrano... só você sabe o verdadeiro significado deste processo. Sempre ao meu lado me incentivando, meu companheiro incondicional, que me deu a mão quando senti que o meu caminho terminava. Sempre muito grata.

Agradecimentos

Primeiramente, a Deus pela sabedoria, paciência e fortaleza. Por permitir-me viver esta experiência de crescimento pessoal e profissional. Não foi fácil, mas ao final tudo deu certo.

Ao meu orientador, Prof. Dr. Janito Vaqueiro Ferreira, pela grande oportunidade que me deu, pela toda sua confiança, ensino constante e ajuda no desenvolvimento deste projeto. Tenho muito orgulho de ter sido parte de seu grupo de pesquisa.

Ao meu coorientador, Prof. Dr. Arthur de Miranda Neto, pela sua confiança, seu apoio, seus conselhos e constante motivação durante este período. Foi muito agradável o trabalho.

Aos membros das bancas de qualificação e defesa Prof. Dr. Pablo Siqueira, Prof. Dr. Ely Carneiro de Paiva e Prof. Dr. Claudio Garcia, pelas sugestões e conselhos para o melhoramento deste trabalho.

A meus colegas de trabalho Camilo Ariza, Olmer Garcia e Tibério Ferreira pela sua importante ajuda e contribuição no desenvolvimento deste projeto. Sem a sua colaboração e conhecimento, o resultado não teria sido o mesmo.

A minha família, pela paciência e compreensão, vocês me inspiram a ser melhor a cada dia. Aos meus pais e irmãs, que ao longo da minha vida, me apoiaram, motivaram e acreditaram em mim em todos os momentos e sempre estão a meu lado.

A todos os meus amigos e colegas, a quem conheci durante este tempo neste maravilho país. Aos moradores da república Miguel Cárdenas e Camilo Ariza, com quem compartilhei muitas histórias, meus melhores momentos. A meus colegas Diana Martinez, Oscar Rojas, Carlos Reyes, Germán Castañeda, Ramiro Chamorro, Suranny Jiménez, Paola González e Marcela Corredor porque havia sempre tempo para se divertir, viajar, comemorar, comer, reir, dançar... estão em meu coração as lembranças das experiências e instantes inesquecíveis, que mesmo estando longe de casa, fizemos uma grande família. Aos meus poucos amigos brasileiros Gilberto Luís, Bruna Freitas, Ana Soubhia, Bruno Miyamoto, Leonardo Miranda e Aline Vinci que me mostraram a grande diversidade cultural deste país que eu gosto tanto, tomo de vocês o melhor do Brasil. E também aos que me ensinaram um pouco da cultura de cada um de seus países durante este etapa, Amanda Pe López da Espanha, Richard Huaman do Peru e Mohammad Shaterzadeh do Irã.

A todos meus companheiros do laboratório, meus colegas do grupo LMA, ao pessoal do DMC e a todas aquelas pessoas que de alguma forma fizeram parte de este projeto e de esta experiência, meus agradecimentos.

A CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, pelo indispensável apoio financeiro.

*Sempre faço o que não consigo fazer para
aprender o que não sei.*

Pablo Picasso

Resumo

RODRÍGUEZ RUIZ, María Fernanda. Desenvolvimento de um sistema de localização híbrido para navegação autônoma de veículos terrestres em ambiente simulado. 2014. 99p. Dissertação (Mestrado). Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, Campinas.

Os veículos autônomos são uma realidade, mas seu desenvolvimento requer altos custos. No entanto, a fim de fortalecer os avanços na robótica móvel, há um grande esforço no desenvolvimento de aplicações de baixo custo orientadas aos veículos autônomos, sendo o estudo de métodos de localização uma das áreas de maior interesse, a fim de uma navegação segura. Esta dissertação de mestrado propõe um método de localização híbrida, composto por sensores proprioceptivos e exteroceptivos, e informações adicionais como mapas digitais e reconhecimento de objetos chave, para melhorar a estimativa da posição de um veículo terrestre. O sistema está baseado nas informações de posicionamento obtidas por um GPS de baixo custo e também em informações obtidas pelo método de localização referenciada, baseada em um conjunto de dados geográficos disponíveis em uma base de dados que compõe o mapa digital. A técnica de fusão de dados selecionada é baseada no filtro de Kalman estendido (EKF), que combina as informações de diferentes sistemas a fim de aumentar a robustez do sistema de localização. O desempenho do método de localização proposto é verificado através de uma plataforma de localização composta por ferramentas como um servidor de mapas baseado em OpenStreetMap e uma outra de simulação composta por ferramentas como ROS e Gazebo, que inclui um veículo terrestre com sensores embarcados, um mapa digital e objetos chave. A arquitetura selecionada é de fácil adequação para a realização de testes, já que permite simular objetos e sensores bem próximos à realidade.

Palavras-chave: Ambiente simulado, Filtro de Kalman, Localização referenciada, Mapas digitais, Navegação autônoma, Veículos autônomos.

Abstract

RODRÍGUEZ RUIZ, María Fernanda. Development of a hybrid localization system for autonomous navigation of ground vehicles in a simulated environment. 2014. 99p. Dissertação (Mestrado). Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, Campinas.

The autonomous vehicles are a reality, but its development is equivalent to high costs. However, in order to strengthen the advances in mobile robotics, there is a large effort in developing low cost oriented applications to autonomous vehicles, with the study of methods of finding the area of most interest in relation to the fact achieve a navigation secure. This dissertation proposes a method of hybrid location consists of proprioceptive and exteroceptive sensors, and additional information such as digital maps and recognition of key objects, to improve the estimation of a land vehicle position. The system is based on the positioning information via GPS low cost, and information obtained by the method referenced location, based on a set of available spatial data in a database comprising the digital map. The fusion technique selected data is based on the extended Kalman filter (EKF) that combines the information of different systems in order to increase the robustness of the location system. The performance of the proposed method of location is verified through a location platform consists of a server tools like maps based on OpenStreetMap and another comprising simulation tools such as ROS and Gazebo, which includes a ground vehicle with embedded sensors, a map digital and key objects. The selected architecture is easily suitability for testing, since it allows you to simulate objects and sensors very close to reality.

Keywords: Autonomous navigation, Autonomous vehicles, Digital maps, Kalman filter, Referenced localization, Simulated environment.

Lista de Ilustrações

1.1	Representação das informações que compõem a solução de um método de localização híbrida.	3
1.2	(Esqueda) Logotipo LMA. (Direita) VILMA plataforma robótica (Veículo Inteligente do LMA) com base no Fiat Punto.	7
1.3	Método de localização proposto baseado em informações de sensores e mapa digital.	8
1.4	Método de localização proposto para estimar a localização do veículo baseado em um GPS de baixo custo.	8
1.5	Plataforma de localização e simulação para simular o veículo autônomo.	9
1.6	Proposta do sistema híbrido de localização.	9
3.1	Eixos de referência do modelo de observação do método de localização referenciada.	20
3.2	Medidas da câmera e do laser.	22
3.3	Representação da visão da câmera embarcada no veículo.	22
3.4	Transformada da posição do ponto r com respeito ao sistema de referência X_o, Y_o .	23
3.5	Resumo do processo proposto para medir, calcular e transformar os dados do sistema de visão.	24
3.6	Transformada da posição do ponto Xr_o, Yr_o , dada em metros, para o sistema de coordenadas geográfico, latitude e longitude, tomando-se como referência a coordenada ll	24
3.7	Ilustração, consulta do ponto Lat_p, Lon_p na base de dados.	26
3.8	Método de localização referenciada para a localização estimada do veículo usando as informações da câmera, laser e a base de dados.	26
3.9	Transformada para calcular a posição do veículo $Xcar_m, Ycar_m$ a partir da posição do objeto Xr_star, Yr_star	28
3.10	Correção da posição do veículo medida $Xcar_m, Ycar_m$ a partir das informações do sistema de visão artificial.	29
3.11	Localização pelo GPS de baixo custo $Pos1$ e localização pelo MLR $Pos2$	29
3.12	Composição do sistema de localização. Dados de entrada para a fusão de dados. . .	30
3.13	Esquema dos dados de entrada e saída do sistema completo que compõe o sistema híbrido para o EKF.	31
4.1	Módulos de composição da plataforma de localização e simulação.	37

4.2	PostGIS é uma extensão do PostgreSQL para tratar dados em um banco de dados geográfico.	40
4.3	Processo de transformação de dados para a visualização do <i>mapa.osm</i>	40
4.4	Módulos que compõem o Servidor de Mapas. OSM, PostgreSQL - PostGIS e Mapnik.	41
4.5	Integração entre o servidor de mapas (OSM, PostgreSQL - PostGIS e Mapnik) e o servidor <i>web</i> (Apache, Leaflet, HTML).	42
4.6	Interface da composição da plataforma de localização.	43
4.7	Visualização do mapa, resultado do processo feito com o arquivo <i>mapa.osm</i> , mais informações de latitude, longitude de um ponto localizado no mapa.	44
4.8	Base de dados do <i>mapa.osm</i> . (a) Base de dados principal. (b) Dados de uma das tabelas da base de dados do <i>mapa.osm</i>	44
4.9	Conexão do GPS de baixo custo do modelo do veículo ao servidor de <i>web</i> . Interação entre a plataforma de localização e a plataforma de simulação.	45
4.10	Módulos do mapa digital, composto pelo servidor de mapas e servidor <i>web</i>	46
4.11	Visualização do mapa digital no navegador <i>web</i> junto as informações da base de dados e a posição do marcador em latitude, longitude.	47
4.12	Adição da tabela <i>info_vilma</i> ao banco de dados espaciais com informações próprias para a localização do modelo do veículo.	48
4.13	Interface da composição da plataforma de simulação.	49
4.14	Gazebo, módulo de composição do simulador 3D.	50
4.15	Representação da conexão entre os módulos do ROS, Gazebo e Python.	52
4.16	Resultados. (a) Percurso do modelo do veículo de um ponto a outro em linha reta. (b) Dados do GPS. (c) Informações da câmera.	53
4.17	Esquema final completo da plataforma de localização e simulação.	54
4.18	Plataforma final. Interface gráfica final da plataforma de localização.	55
4.19	Plataforma final. Interface gráfica final da plataforma de simulação.	55
5.1	Sistema de Localização Híbrida.	57
5.2	Resultado da plataforma de localização na posição inicial.	59
5.3	Resultado do GPS ideal e do GPS de baixo custo na plataforma de localização durante o percurso do modelo do veículo.	60
5.4	Teste 3. Medições obtidas na plataforma de localização. GPS ideal vs. GPS de baixo custo.	61
5.5	Erro da posição de cada ponto do GPS de baixo custo com respeito ao GPS ideal.	61
5.6	Objetos. (a) Modelo sinal de trânsito pare. (b) Base de dados <i>info_vilma</i>	63

5.7	Módulos que descrevem a conexão entre ROS - Gazebo e OpenCv para a detecção dos objetos.	64
5.8	Resultado da detecção do sinal de pare, enquanto o modelo de veículo fazia o percurso.	64
5.9	Resultado dos dados capturados pelo laser em um instante de tempo. Nuvem de pontos com as informações da cena.	65
5.10	Resultado do alinhamento entre as imagens da câmera e o mapa de profundidade. .	66
5.11	Resultado da distância medida entre o sinal de pare detectado e o centro da câmera ligada ao modelo do veículo.	66
5.12	Módulos que descrevem a conexão entre a plataforma de simulação e o sistema operacional do veículo autônomo para a detecção dos objetos e medição da distância. .	67
5.13	<i>Scripts</i> em Python dos cálculos de cada um dos sensores e os dados que são publicados em nós de ROS.	68
5.14	Esquema do processo para calcular a posição do sinal de pare com respeito ao mundo X_o, Y_o	69
5.15	Esquema do processo para calcular a transformação do ponto X_{r_o}, Y_{r_o} dada em metros a latitude, longitude.	70
5.16	Esquema do processo para consultar na base de dados info_vilma a coordenada aproximada que foi calculada Lat_p, Lon_p	70
5.17	Esquema geral das medidas do método de localização referenciada.	71
5.18	Resultado final da implementação do método de localização referenciada.	72
5.19	Resultados obtidos pelo MLR. GPS ideal vs. MLR.	73
5.20	Erro da posição de cada ponto do MLR com respeito ao GPS ideal.	73
5.21	Resultados do EKF para o sistema de localização somente com dados do GPS de baixo custo.	76
5.22	Erro da posição estimada do EKF com dados do GPS de baixo custo com respeito ao GPS ideal.	76
5.23	Resultados da localização híbrida. EKF = MLR + GPS + Bússola + Modelo Matemático.	77
5.24	Erro da posição estimada da localização híbrida com respeito ao GPS ideal.	77
A.1	Modelo de <i>Ackerman</i> para no veiculo de 4 rodas.	90
A.2	Modelo simplificado bicicleta.	91
A.3	Modelo cinemático bicicleta. Deslocamento da roda traseira.	92
A.4	Modelo cinemático bicicleta. Deslocamento da roda dianteira.	93
B.1	Algoritmo do EKF.	99

Lista de Tabelas

5.1	Erros de localização do GPS de baixo custo com respeito ao GPS ideal.	62
5.2	Erros de posição do MLR.	72
5.3	Erros da posição estimada, fusão de dados e localização híbrida.	75

Lista de Abreviaturas e Siglas

Matrizes e Vetores

- \dot{x} - Vetor de estados
 z - Modelo de observação do sistema de medição
 P, Q, R - Matriz de covariância

Letras Latinas

- RN - Raios de curvatura vertical principal do planeta
 RM - Raio de curvatura do meridiano do planeta
 R - Raio equatorial do planeta
 f - Achatamento do planeta
 L - Distância/Comprimento entre os eixos
 w - Largura entre as rodas

Letras Gregas

- ψ - Ângulo em graus entre o eixo X_o e o Norte, sentido horário
 μ_o - Latitude do ponto de referência
 ω - Ângulo de esterçamento
 θ - Ângulo de orientação

Siglas

GPS	- <i>Global Positioning System</i> (Sistema de Posicionamento Global)
INS	- <i>Inertial Navigation System</i> (Sistema de Navegação Inercial)
EKF	- <i>Extended Kalman Filter</i> (Filtro estendido de Kalman)
VILMA	- Veículo Inteligente do LMA
MLR	- Método de Localização Referenciada
LMA	- Laboratório de Mobilidade Autônoma
OSM	- <i>OpenStreetMap</i>
ROS	- <i>Robot Operating System</i>

Unidades de medição

[m]	- Metro - Medida de comprimento
[s]	- Segundo - Medida de tempo

Outras Notações

3D	- Elemento tridimensional
----	---------------------------

SUMÁRIO

Lista de Ilustrações	xvii
Lista de Tabelas	xxi
Lista de Abreviaturas e Siglas	xiii
SUMÁRIO	xxv
1 INTRODUÇÃO	1
1.1 Introdução ao problema da localização	2
1.2 Breve revisão bibliográfica	4
1.2.1 História dos veículos autônomos	5
1.3 Objetivos	6
1.4 Desenvolvimento proposto	7
1.5 Resumo da dissertação	9
2 REVISÃO BIBLIOGRÁFICA	11
2.1 Introdução	11
2.2 Sistemas de percepção	11
2.2.1 Sensores	12
2.2.2 Sistema de posicionamento global	12
2.3 Localização	13
2.3.1 Métodos de localização	13
Localização relativa	14
Localização absoluta	14
Localização por visão	15
Localização baseada em mapas	15
Localização por fusão de dados	16
2.4 Simuladores de robôs móveis	17
3 PROPOSTA DE UM MÉTODO DE LOCALIZAÇÃO REFERENCIADA	19
3.1 Introdução	19

3.2	Eixos de referência do modelo de observação	20
3.3	Processo de localização do MLR	21
3.3.1	Etapa 1: Localização de um objeto de interesse	21
3.3.2	Etapa 2: Consulta na base de dados	23
3.3.3	Etapa 3: Correção da posição do veículo	26
3.4	Estimação da localização do veículo por fusão de dados	30
3.4.1	Parâmetros do filtro	31
3.4.2	Fase de predição	33
3.4.3	Fase de correção	34
4	PLATAFORMA DE LOCALIZAÇÃO E SIMULAÇÃO	37
4.1	Introdução	37
4.2	Servidor de mapas	38
4.2.1	Mapa digital - OpenStreetMap	38
4.2.2	Sistemas de informação geográfica	38
4.2.3	Banco de dados espaciais	40
4.2.4	Servidor web	41
4.3	Plataforma de localização	43
4.3.1	Mapa Digital	43
4.3.2	Resultado da plataforma de localização	44
4.4	Plataforma de simulação	49
4.4.1	Gazebo	49
4.4.2	ROS	51
4.4.3	Resultado da plataforma de simulação	52
4.5	Plataforma final	53
5	RESULTADOS NUMÉRICOS	57
5.1	Plataforma de localização	57
5.1.1	Teste 1. Inserção de dados	58
5.1.2	Teste 2. Visualização dos dados da plataforma	59
5.1.3	Teste 3. Resultados dos sensores GPS	60
5.2	Plataforma de simulação	62
5.2.1	Teste 4. Objetos chave	62
5.3	Sistema operacional do veículo autônomo	63
5.3.1	Teste 5. Detecção de objetos e medição da distância	64

5.4	Localização referenciada	67
5.4.1	Teste 6. Etapas dos cálculos realizados	68
5.4.2	Teste 7. Resultados da posição estimada pelo MLR	72
5.5	Localização híbrida	74
5.5.1	Resultado 1. Fusão de dados EKF = GPS + Bússola + Modelo Matemático	74
5.5.2	Resultado 2. Fusão de todos os dados EKF = MLR + GPS + Bússola + Modelo Matemático	75
5.6	Resultados fusionados	75
6	CONCLUSÕES E TRABALHOS FUTUROS	79
6.1	Conclusões	79
6.2	Trabalhos futuros	81
Referências		83
APÊNDICES		89
A – Modelo matemático do veículo		89
A.1	Introdução	89
A.2	Sistema de coordenadas	89
A.3	Modelo cinemático	90
A.4	Desenvolvimento do modelo cinemático	92
B – Filtro de Kalman		97
B.1	Introdução	97
B.2	Filtro de Kalman Estendido	97
B.2.1	Predição	98
B.2.2	Correção	99
B.2.3	Resumo	99

1 INTRODUÇÃO

A fim de fortalecer e promover avanços na robótica móvel, há um grande esforço da comunidade científica, e também industrial, no desenvolvimento de aplicações móveis autônomas capazes de atuar em ambientes desconhecidos e dinâmicos, sem a necessidade da participação contínua ou direta do homem. Fruto de maior interesse neste trabalho, com o objetivo na navegação autônoma e assistência ao condutor, o estudo de métodos de localização se justifica pois busca alcançar sistemas mais precisos, confiáveis e acessíveis (TAO, 2012).

Para uma navegação segura de veículos autônomos, a localização é a base (TAO *e outros*, 2013). Deste modo, um conjunto de sensores embarcados no veículo pode ser utilizado para melhorar a localização. Dentre os mais utilizados, os Sistemas de Posicionamento Global (GPS) permitem a localização no referencial terrestre, sendo a ferramenta mais utilizada atualmente (TAYLOR E BLEWITT, 1999).

Na literatura, são inúmeras as diferentes abordagens encontradas para a solução do problema de localização em robótica móvel. Alguns dos métodos mais usados são: localização relativa por meio de sensores como acelerômetros, giroscópios e encoders que permitem estimar o deslocamento do robô (LANEURIT *e outros*, 2003); localização *dead reckoning* que calcula o deslocamento relativo baseado nas informações proprioceptivas, como velocidade e direção (XIAOYUN E HENG, 2008); localização auxiliada pela visão computacional, a partir da detecção de marcas na via pública, utilizando somente uma câmera (LANEURIT *e outros*, 2003) ou com duas câmeras (LENG E GRUYER, 2003), permitindo neste caso estimar o erro lateral de localização do robô; localização com a utilização de mapas (MATTERN *e outros*, 2010); localização integrando dados geográficos (HENTSCHEL E WAGNER, 2010) e localização por fusão de dados de diferentes sensores (LANEURIT *e outros*, 2003).

O estudo de métodos de localização por fusão de dados tem atraído o interesse de pesquisadores. Dentre os métodos, há aqueles que combinam informações do GPS, visão computacional e mapas digitais (LANEURIT *e outros*, 2003), (MATTERN *e outros*, 2010) e (TAO, 2012), fornecendo uma estimativa de localização mais próxima à real, devido ao grande conteúdo de informação, abundante em detalhes, fornecidos pela visão computacional e também por mapas digitais. Contudo, a fusão de dados pode requerer um alto custo computacional, dependendo da complexidade e da robustez desejadas. Por outro lado, o problema de localização pode ser descrito por um algoritmo

probabilístico, onde as implementações mais comuns utilizam o filtro de Kalman (TAO, 2012).

1.1 Introdução ao problema da localização

Sistemas mais precisos possuem alto custo, por outro lado, sistemas de menor custo oferecem menor precisão.

Um dos sistemas mais utilizados atualmente é o GPS (TAYLOR E BLEWITT, 1999). Porém, este tipo de sensor depende da visualização direta de um número mínimo de satélites, e de outros fatores naturais que podem acarretar erros que podem chegar a até 100m (HIDE E MOORE, 2005).

Um aspecto importante na pesquisa de sistemas de navegação autônomos é sua ampla área de aplicação. Destacam-se, ADAS (do inglês *Advanced Driver Assistance Systems*) e IDAS (do inglês *Intelligent Driver Assistance System*), no monitoramento de zonas radioativas, de ambientes de difícil acesso, tais como minas, espaço e oceanos (NAVARRO, 2009). Nos últimos anos, o interesse em veículos autônomos foi crescente, e com grande interesse da comunidade científica a fim de alcançar sistemas com maior confiabilidade e seguros, o que inclui a precisão e acessibilidade (TAO, 2012). Depois do desafio *Grand Challenge*, patrocinado pelo DARPA (*Defense Advanced Research Projects Agency*), talvez os resultados mais notáveis tenham sido os veículos autônomos da Google¹. Entre seus mais recentes trabalhos, cabe destacar o projeto *Google self-driving car project*², um carro para duas pessoas, sem volante, capaz de realizar deslocamentos autônomos guiado por mapas digitais e sensores.

A localização desempenha papel importante no contexto de veículos autônomos, isto é, o sistema deve ter a habilidade de se localizar de maneira precisa no ambiente em que atua. Sem esta informação de localização, tarefas como navegar em ambiente desconhecido ou seguir uma trajetória, podem se tornar inviáveis. A localização é peça chave em diversos contextos, como exemplo em ambientes dinâmicos.

Muitos sistemas de localização precisos podem ser adquiridos no mercado prontos para serem usados, no entanto, seu custo é alto. Isto é uma motivação para o desenvolvimento de sistemas

¹<http://googleblog.blogspot.com.br/2014/05/just-press-go-designing-self-driving.html>

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

baseados em sensores de baixo custo.

Além da utilização de sensores de baixo custo, complementarmente, a literatura apresenta que mapas digitais podem ser utilizados para auxiliar a tarefa de localização. Os mapas digitais quando são utilizados em conjunto com informações do GPS, fornecem maior confiabilidade e precisão. Ainda oferecem uma visão mais ampla do ambiente, pois informações armazenadas previamente no mapa digital podem contribuir com as decisões do sistema de percepção e melhorar a localização. Esta combinação pode diminuir o erro de localização (TAO, 2012).

A combinação de informações disponíveis em mapas digitais, sensores GPS de baixo custo, além de câmeras para o reconhecimento de objetos chave, pode ser feita através de métodos de fusão de dados, a fim de atenuar erros de localização (TAO, 2012). Neste sentido, a principal motivação para a utilização de métodos de fusão de dados neste trabalho é a diminuição dos efeitos diretos de ruídos oriundos de cada sensor individualmente, e o melhoramento da localização baseada em multi-sensores e multi-informações.

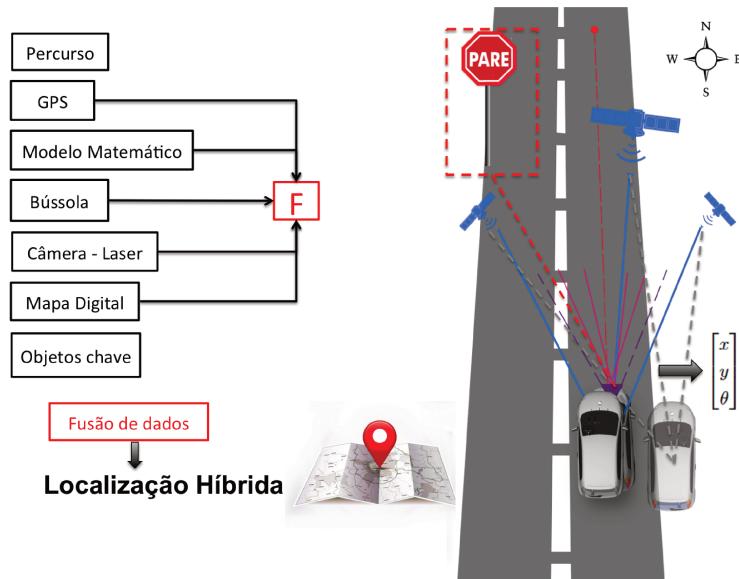


Figura 1.1: Representação das informações que compõem a solução de um método de localização híbrida.

A Figura 1.1 apresenta uma representação das informações que compõem a solução proposta neste trabalho, a saber, um método de localização híbrida. Basicamente, o veículo realiza um deslocamento, um sensor GPS retorna a localização, mas segundo a Seção 1.1 este tipo de sistema pode apresentar erros de posição. Neste caso, outros sensores podem ser utilizados, como bússola, câ-

mera e laser, além de informações do modelo do veículo (movimento de rodas e orientação) e de mapas digitais, juntamente com métodos de reconhecimento de objetos chave. A fusão de dados combina as entradas a fim de obter a localização do veículo.

1.2 Breve revisão bibliográfica

A robótica é uma área de pesquisa que envolve ciência e tecnologia, e esforços têm sido evidenciados para dotar robôs da capacidade de realização de tarefas de forma eficaz. Neste contexto, robôs móveis podem ser descritos como veículos que possuem sistema sensorial e de atuação, em muitos casos, com a capacidade de executar tarefas pré-definidas pelo usuário através de uma arquitetura de controle. Podem ser classificados de acordo com o nível de autonomia, e na literatura assumem diferentes nomes, como robôs autônomos ou não tripulados (ÍTALO LOIOLA, 2008).

Robôs móveis ou veículos autônomos possuem, basicamente, três capacidades inter-relacionadas (NAVARRO, 2009):

- **Percepção:** capacidade de identificar e/ou reconhecer objetos ou características do ambiente.
- **Planejamento:** capacidade de planejar a execução de tarefas em função de um objetivo pré-determinado.
- **Ação:** capacidade de executar uma tarefa pré-definida.

As aplicações contemplam as áreas de defesa e segurança, agricultura, transporte, etc.

Uma das características mais importantes de qualquer sistema autônomo trata da capacidade de perceber e adquirir conhecimento de seu ambiente para poder agir sobre este. Isto é alcançado por meio de sensores que obtêm diferentes tipos de medidas físicas e as transformam em informações úteis para o sistema.

Os sistemas de localização formam a base para projetos envolvendo veículos autônomos (TAO e outros, 2013). Estes, por sua vez, fazendo uso de seus sensores, devem se localizar e se orientar no ambiente em que atuam. Muitos sistemas usam GPS para determinar a posição, e se tornou a ferramenta de navegação mais amplamente utilizada (TAYLOR E BLEWITT, 1999).

Existem diferentes processos para localizar um veículo, tais como, métodos de localização relativa, absoluta e híbrida (LANEURIT *e outros*, 2003), além de outras, como a localização por visão (LENG E GRUYER, 2003) e localização baseada em mapas (NAVARRO, 2009). A vantagem oferecida por métodos combinados como visão e GPS, é que permitem melhorar a estimativa da localização, e em casos da falta temporária de um sinal, esta pode ser compensada pelo outro.

1.2.1 História dos veículos autônomos

A história apresenta a evolução dos veículos autônomos. Os primeiros experimentos foram realizados nos anos 70 no *Tsukuba Mechanical Engineering Lab* onde foi desenvolvido o primeiro veículo autônomo que seguia linhas brancas com velocidades de até 30km/h (SICILIANO E KHATIB, 2008).

Nos anos 80, Ernst Dickmanns, da Universidade de Bundeswehr München (UniBW) na Alemanha, desenvolveu um sistema de navegação autônomo baseado no método de visão conhecido como *saccadic vision*. A plataforma utilizada foi uma Mercedes Benz, e neste caso, a velocidade limite foi de 96km/h (DICKMANNS, 2004).

Nos anos 90, a Comissão Europeia iniciou alguns projetos. O *Eureka Prometheus Project* contemplou o desenvolvimento de veículos autônomos. Outro projeto, ARGO (*the ARGO Project*), foi desenvolvido pela Universidade de Parma na Itália. Baseado em estéreo visão, teve o mérito de realizar o seguimento de faixas brancas com velocidade inferior a 90km/h (BROGGI *e outros*, 1999).

Porém, os avanços mais significativos foram obtidos a partir de 2004 através de dois grandes desafios organizados pelo DARPA (*Defense Advanced Research Projects Agency*) nos Estados Unidos, o *Grand Challenge*, que consistiu em realizar um percurso de 240 km com obstáculos, sendo a equipe vencedora *Stanford Racing Team* da Universidade de Stanfort (THRUN *e outros*, 2006). Essa experiência fez ressurgir a pesquisa em veículos autônomos.

No Brasil, diferentes universidades e centros de pesquisa trabalham em diferentes projetos dirigidos à navegação autônoma.

O Grupo de Pesquisa e Desenvolvimento de Veículos Autônomos, da Escola de Engenharia

da Universidade Federal de Minas Gerais (UFMG) foi um dos pioneiros. CADU (acrônimo para Carro Autônomo Desenvolvido na UFMG) é o nome do principal projeto. Este é capaz de realizar pequenas distâncias sem a intervenção contínua do condutor, e se locomover recebendo comandos de computador, tendo como referência coordenadas GPS previamente enviadas ao sistema de navegação. Um sistema de visão computacional realiza as tarefas de detecção de obstáculos. O grupo vem realizando seus trabalhos desde 2007 (DE REZENDE, 2010).

O Centro de Tecnologia da Informação (CTI) Renato Archer tem um projeto chamado VERO (VEículo RObótico). Tem como objetivo o desenvolvimento de metodologias de navegação autônoma para veículos terrestres em ambientes externos, capacitando-os para realizar diferentes classes de aplicações em campo e em ambiente urbano, através de processos como controle e seguimento de trajetórias, controle baseados em visão e esquemas de percepção (DE PAIVA *e outros*, 2010).

O projeto do veículo inteligente CARINA II (Carro Robótico Inteligente para Navegação Autônoma) teve início em 2011. O veículo foi adaptado com uma série de equipamentos por um grupo de pesquisadores do Instituto de Ciências Matemáticas e da Computação (ICMC) e da Escola de Engenharia do campus da USP em São Carlos. Em outubro de 2013, o veículo foi testado com sucesso nas ruas da cidade de São Carlos sem nenhuma intervenção de motorista humano, sendo o primeiro carro autônomo autorizado a trafegar em ruas de uma cidade (DE OLIVEIRA, 2013).

Em meados de 2008, o Laboratório de Mobilidade Autônoma (LMA) foi formado no Departamento de Mecânica Computacional (DMC) da Faculdade de Engenharia Mecânica (FEM) da Universidade Estadual de Campinas (UNICAMP), com o interesse principal em robótica móvel e segurança veicular. A plataforma robótica VILMA (Veículo Inteligente do LMA) está em processo de automatização e é apresentada na Figura 1.2. As linhas de pesquisa principais no grupo são: Percepção, Localização, Navegação e Controle.

1.3 Objetivos

Esta dissertação tem como objetivos:

- Propor um método de localização híbrida para melhorar a estimativa da posição de um veículo



Figura 1.2: (Esquerda) Logotipo LMA. (Direita) VILMA plataforma robótica (Veículo Inteligente do LMA) com base no Fiat Punto.

terrestre.

- Construir uma plataforma de localização e simulação para realizar os testes de validação do método de localização proposto.

1.4 Desenvolvimento proposto

Neste trabalho, o desenvolvimento de um sistema de localização híbrido é apresentado, composto por sensores proprioceptivos e exteroceptivos e informações adicionais, como mapas digitais e reconhecimento de objetos chave. O sistema está baseado em informações de posicionamento obtidas por um GPS de baixo custo, e em informações obtidas por um método de localização referenciada. Este método é baseado num conjunto de dados geográficos de um mapa digital, disponíveis em uma base de dados. A técnica de fusão de dados selecionada é baseada no filtro de Kalman estendido (EKF), que combina informações a fim de aumentar a robustez do sistema de localização. Esta técnica pode ser considerada simples em termos de formulação e é aplicada a um amplo número de problemas de localização em veículos autônomos (TAO, 2012).

O desempenho do método de localização proposto é verificado através de uma plataforma de localização e uma outra de simulação, que inclui um veículo terrestre com sensores embarcados, um mapa digital e objetos chave. A arquitetura selecionada é de fácil adequação para a realização de testes, já que permite simular objetos e sensores bem próximos à realidade.

No primeiro passo, propõe-se um método de localização referenciada, que faz uso de uma câmera para detectar objetos; um sensor laser a fim de obter informações de distância em relação a objetos detectados; e uma base de dados local com informações geográficas de uma área específica,

que contribui na forma de um mapa digital. Além disso, um sensor GPS de baixo custo integra o método proposto para auxiliar na localização.

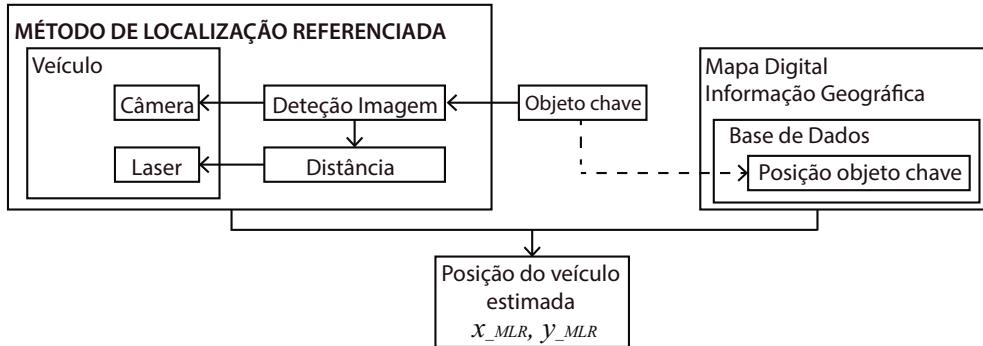


Figura 1.3: Método de localização proposto baseado em informações de sensores e mapa digital.

Conforme apresentado na Figura 1.3, o método de localização referenciada realiza consultas na base de dados para obter informações a respeito de um objeto detectado, retornando a posição do mesmo, e estimar a sua posição com relação ao veículo. Na Figura 1.4 apresenta-se o sistema de localização baseado nas informações de um GPS de baixo custo embarcado no veículo. Como resultado se tem duas posições: uma estimada pelo método de localização referenciada e outra estimada pelo GPS de baixo custo.

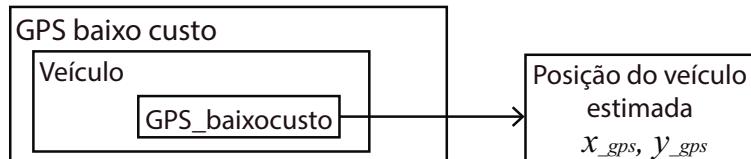


Figura 1.4: Método de localização proposto para estimar a localização do veículo baseado em um GPS de baixo custo.

O segundo passo envolve a utilização de um simulador de ambientes 3D, que por sua vez está ligado a uma ferramenta de auxílio ao desenvolvimento de aplicações robóticas, e este ligado a um servidor de mapas (informações geográficas), como apresentado na Figura 1.5. A união destas ferramentas permite a inclusão de sensores para leitura no ambiente de simulação, além de um veículo para simular a dinâmica e cinemática.

A plataforma desenvolvida contempla aspectos de localização e simulação. Conforme apresentado na Figura 1.5, a plataforma de localização contempla um servidor de mapas baseado em OpenStreetMap, PostgreSQL e PostGIS, e permite obter e apresentar informações geográficas. A

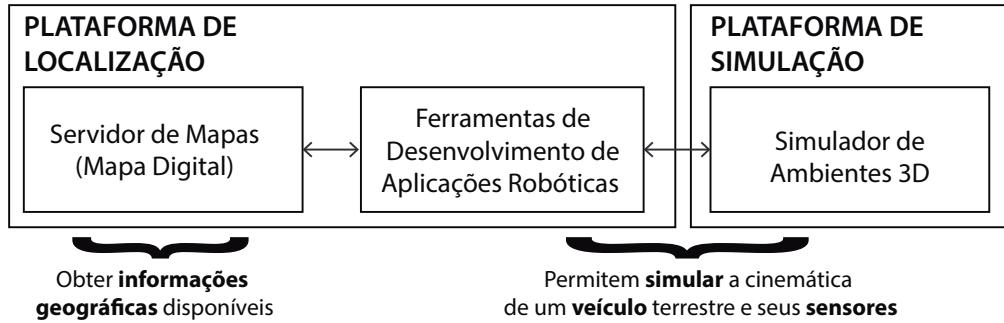


Figura 1.5: Plataforma de localização e simulação para simular o veículo autônomo.

plataforma de simulação inclui um simulador de ambientes 3D baseado em Gazebo versão 1.4, com ligação ao ROS, ferramenta de desenvolvimento de aplicações robóticas, permitindo a simulação do ambiente e também de um veículo terrestre com sensores embarcados.

A etapa final consiste em implementar o filtro de Kalman estendido (EKF do inglês, *extended Kalman filter*) e estimar a localização com base no método de localização referenciada, GPS de baixo custo, bússola e modelo matemático do veículo, como é apresentado na Figura 1.6.

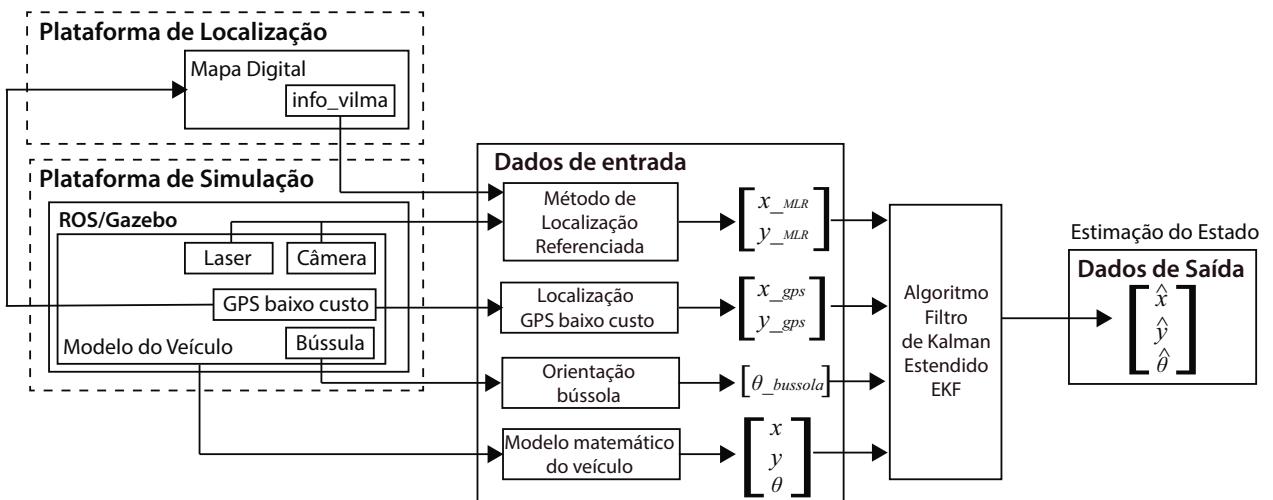


Figura 1.6: Proposta do sistema híbrido de localização.

1.5 Resumo da dissertação

Este trabalho está organizado em seis capítulos, uma seção dedicada às referências bibliográficas e dois apêndices.

O Capítulo 1 contextualiza a problemática relacionada à localização de robôs móveis, relacionando-a aos objetivos deste trabalho.

O Capítulo 2 apresenta um revisão da literatura, tendo como principal objetivo fundamental principalmente as áreas de percepção e localização, combinadas para um objetivo comum, a localização baseada em fusão de dados.

O Capítulo 3 apresenta o método de localização referenciada proposto, apresentando cada passo que compõe o sistema de localização. No final, há um destaque para a composição das entradas e saídas do modelo final.

O Capítulo 4 mostra em detalhes o desenvolvimento e construção da plataforma de localização e simulação vislumbrada para validar este trabalho. São descritas as características principais do ROS, Gazebo, OpenStreetMap e a construção do servidor de mapa digital. Por fim, a integração dos módulos que compõem o sistema é detalhada, assim como as conexões entre cada módulo e seu funcionamento.

O Capítulo 5 apresenta a sequência de cálculos e o conjunto de resultados numéricos realizados na plataforma de localização e simulação apresentada no Capítulo 4. Finalmente, é apresentada a fusão de dados e a localização híbrida baseada no filtro de Kalman estendido. No final desse capítulo alguns comentários e comparações dos resultados são apresentados.

O Capítulo 6 apresenta as conclusões e algumas propostas de trabalhos futuros. Posteriormente, as referências citadas no texto.

O Apêndice A apresenta detalhes do modelo matemático do veículo adotado neste trabalho.

O Apêndice B inclui conceitos base do método de fusão de sensores, apresentando as formulações utilizadas na implementação do filtro de Kalman estendido.

2 REVISÃO BIBLIOGRÁFICA

Este capítulo tem como principal objetivo apresentar a revisão da literatura relacionada à localização de robôs móveis e/ou veículos inteligentes. Primeiramente, definições básicas de robôs móveis e veículos autônomos são apresentadas. Em seguida, uma descrição dos sistemas de percepção e os diferentes tipos de métodos de localização. Por fim, são introduzidos alguns dos simuladores de robôs móveis como plataformas de testes.

2.1 Introdução

Um dos principais objetivos de investigação na área de robótica móvel pode se resumir no desenvolvimento de métodos para suporte à navegação autônoma (ÍTALO LOIOLA, 2008), sendo necessário como um de seus componente principais: A percepção.

2.2 Sistemas de percepção

Uma das características mais importantes de qualquer sistema autônomo trata da capacidade de perceber e adquirir conhecimento de seu ambiente para poder agir sobre este. Isto é alcançado por meio de sensores que obtêm diferentes tipos de medidas físicas e as transformam em informações úteis para o sistema. Os sensores podem ser classificados em dois grupos principais: os sensores internos ou proprioceptivos e os sensores externos ou exteroceptivos. Os sensores proprioceptivos representam, por exemplo, as variáveis cinemáticas do sistema. Entre os sensores mais comumente utilizados estão: giroscópios, acelerômetros, potenciômetros e os encoders. Os sensores exteroceptivos interagem com o ambiente, quantificando as variáveis e interpretando esses dados para extrair características do ambiente, como exemplos de distância, orientação e localização. Entre os sensores mais comumente utilizados estão os sensores baseados em visão por computador, possuindo grande destaque e interesse da indústria nos dias de hoje (NAVARRO, 2009).

2.2.1 Sensores

Entre os sensores proprioceptivos e exteroceptivos cabe destacar:

- **Encoders** (sensor proprioceptivo) é um dos dispositivos mais populares e está acoplado diretamente ao eixo do motor ou das rodas do veículo. O mecanismo baseia-se em um feixe de luz que é interceptado por áreas transparentes e opacas, que são alternados em um disco rotativo acoplado ao eixo de interesse. O encoder fornece informação de velocidade angular e posição. Uma limitação é que se apresenta apenas um feixe de luz sendo incapaz de resolver o sentido de rotação, além do que podem apresentar erros de posição quando ocorre derrapagem nas rodas do robô móvel (LEI E WANG, 2003).
- **Bússola** (sensor proprioceptivo) ajuda a determinar a orientação do robô móvel com base na medição dos componentes do campo magnético da terra. Frequentemente é usado como suporte para outros sistemas sensoriais, como giroscópio e encoder, porque sua utilização, por exemplo, dentro de um ambiente interior, pode ser afetada por outras fontes eletromagnéticas (VENKATRAMAN *e outros*, 2010).
- **Giroscópio** (sensor proprioceptivo) é utilizado para determinar a orientação em relação a um referencial fixo. Este dispositivo fornece o ângulo ou a velocidade angular do robô móvel. Com respeito à bússola, o giroscópio capta informação no âmbito relativo, enquanto que a bússola no âmbito absoluto (VENKATRAMAN *e outros*, 2010).
- **Câmera** (sensor exteroceptivo) é um dos sensores mais utilizados, pois proporciona uma grande quantidade de informações sobre o ambiente, o que permite uma melhor interação em ambientes dinâmicos. Nesta área são utilizados vários dispositivos, tais como as câmeras mono ou estéreo (YUEN E MACDONALD, 2005).

2.2.2 Sistema de posicionamento global

O Sistema Global de Navegação por Satélite, (GNSS, do inglês *Global Navigation Satellite Systems*) é um termo genérico que se refere aos sistemas de navegação por satélite para determinar a posição e localização de um receptor em qualquer lugar do mundo, seja por terra, mar ou ar. Entre estes sistemas tem-se o desenvolvido pelos Estados Unidos, o GPS. O Sistema de Posicionamento

Global, (GPS, do inglês *Global Positioning System*) foi desenvolvido pela Força Aérea dos Estados Unidos no início dos anos 60 inicialmente somente para uso militar. Atualmente seu uso e aplicação são livres com certas restrições. Consiste de 24 satélites que orbitam a Terra a 20.200 km duas vezes por dia, em seis planos orbitais que estão inclinados a 55 graus em relação ao plano do Equador. O funcionamento do sistema GPS baseia-se no princípio da triangulação, em que, com pelo menos três satélites, o receptor pode determinar a posição (longitude, latitude) que é chamada posição fixa ou bidimensional, e com 4 ou mais satélites para se determinar a posição tridimensional (latitude, longitude, altitude). Para determinar a posição, o GPS é baseado em medição de distâncias a partir dos sinais de rádio transmitidos pelos satélites, que são então capturados e decodificados pelos receptores. Com a posição e estimativa de distância de pelo menos 3 satélites em relação a um ponto na Terra, pode-se estimar a sua posição (SOARES, 2005).

Um número significativo de trabalhos na literatura baseiam-se no uso do GPS para localização (VENKATRAMAN *e outros*, 2010), (MATTERN *e outros*, 2010), (XIAOYUN E HENG, 2008), (YANG E SUN, 2007), (LENG E GRUYER, 2003), (LANEURIT *e outros*, 2003), (TAO *e outros*, 2013), entre outros.

2.3 Localização

Os sistemas de localização formam a base para projetos envolvendo veículos autônomos (TAO *e outros*, 2013). Estes, por sua vez, devem fazer uso de seus sensores, se localizar e orientar no ambiente em que atuam. Muitos sistemas usam GPS para determinar a posição, e tornou-se a ferramenta de navegação mais amplamente utilizada (TAYLOR E BLEWITT, 1999). Mas o crescimento da população nas grandes cidades têm como resultado a construção de edifícios altos e novas rotas de transporte, tais como pontes e túneis, o que torna difícil a captura dos sinais dos GPS, consequentemente, resultando em perda de informações para a localização.

2.3.1 Métodos de localização

Existem diferentes processos para localizar um veículo, tais como, métodos de localização relativa, absoluta e híbrida (LANEURIT *e outros*, 2003), além de outras, como a localização por visão (LENG E GRUYER, 2003) e localização baseada em mapas (NAVARRO, 2009). A vantagem

oferecida por métodos combinados, visão e GPS, permitem melhorar a estimativa da localização, e em casos da falta temporária de um sinal, esta pode ser compensada pelo outro. Nota-se que a utilização de sensores GPS de baixo custo, requisito neste trabalho, podem ocasionar erros de posição de até 100m (HIDE E MOORE, 2005).

Localização relativa

O método de localização relativa ou localização local refere-se ao uso das informações observáveis, através da utilização de sensores proprioceptivos como acelerômetros, giroscópios e encoders, que permitem estimar o deslocamento do robô (LANEURIT *e outros*, 2003). Um caso típico é conhecido como *dead reckoning*, que calcula o deslocamento relativo e para isso considera informações de velocidade e direção, além do tempo entre a última posição e a atual (XIAOYUN E HENG, 2008). Porém, o principal problema associado a este método é a divergência para grandes distâncias (LANEURIT *e outros*, 2003). Outra técnica utilizada para localização relativa é baseada em informações de sensores de odometria ligados às rodas do robô, porém são sensíveis a erros relacionados ao escorregamento das rodas (NAVARRO, 2009). Apesar das desvantagens, estas técnicas, *dead reckoning* e localização por odometria, são amplamente utilizadas na literatura (LEI E WANG, 2003), (SHENGBO *e outros*, 2003), (BONNIFAIT *e outros*, 2001) visto que apresentam boa precisão a baixo custo, pois os dados de odometria, quando fusionados com outros sensores, minimizam os erros nos deslocamentos realizados pelo robô.

Localização absoluta

O método de localização absoluta ou localização global refere-se ao uso das informações externas do ambiente de trabalho do robô, por exemplo, através da utilização de sensores exteroceptivos como o GPS, que indicam a posição do veículo em um referencial global. Este método envolve algum tipo de adaptação ao ambiente no sistema de referência local. A principal desvantagem deste tipo de método é a possível perda da informação solicitada, relativa à visibilidade dos satélites (LANEURIT *e outros*, 2003). A localização absoluta ou global também pode ser obtida através da captura de informações externas a partir de imagens (MATTERN *e outros*, 2010), e outros métodos (GOEL *e outros*, 1999).

Localização por visão

A localização por visão refere-se ao uso das observações externas do ambiente através da utilização de dispositivos, tais como câmeras de vídeo, que oferecem informações adicionais como a cor, textura e forma, além do que muitos dispositivos atualmente fornecem ótima resolução (YUEN E MACDONALD, 2005). A literatura apresenta a utilização de câmeras de vídeo como fonte para a odometria visual (CARDENAS, 2013), embora existam métodos que se baseiam na utilização de outros sistemas comercialmente disponíveis no mercado (TAO, 2012). Outras aplicações exploram câmeras para melhorar os resultados da localização, a partir da detecção de marcações na via pública com uma câmera (LANEURIT *e outros*, 2003) ou duas câmeras (LENG E GRUYER, 2003) para estimar a posição lateral do robô.

Localização baseada em mapas

A localização baseada em mapas refere-se ao uso de informações geográficas e/ou geométricas fornecidas através de um Sistema de Informação Geográfica (GIS). A partir destas informações, busca-se uma de correspondência entre dados locais, ou topológicos, com dados obtidos por um sistema de percepção (NAVARRO, 2009).

A correspondência entre informações previamente disponíveis num mapa digital com, por exemplo, marcações visuais é caracterizada como *Map-Matching* (MATTERN *e outros*, 2010), (TAYLOR E BLEWITT, 1999).

O processo de *Map-Matching* habitualmente está ligado a receptores GPS através da integração de sistemas com filtros de estimação, tais como filtro de Kalman (TAYLOR E BLEWITT, 1999) e *Map-Matching* usando múltiplas técnicas de hipótese (PYO *e outros*, 2001).

Os mapas digitais são uma representação da malha rodoviária composta de linhas e polígonos que representam espaços, segmentos e cruzamentos. A informação do mapa digital pode ser utilizada como comando para a navegação do robô, ou ainda como informação de posicionamento, quando esta é combinada com um dispositivo GPS. Os mapas digitais oferecem uma visão mais ampla do ambiente, onde diferentes tipos de dados podem ser visualizados, tais como, dados geográficos, topológicos, assim como informações de sinalização de trânsito. Uma vantagem deste

tipo de metodologia é permitir a minimização do erro de localização, visto que combina informações de diferentes sensores (TAO, 2012). A literatura apresenta diferentes abordagens e aplicações utilizando mapas digitais; por exemplo, baseado na identificação de marcações de trânsito, neste caso linhas na pista (MATTERN *e outros*, 2010); fusão de informações referentes à sinalização de trânsito com sinais detectados por câmera (JAMSHIDI *e outros*, 2011); mapas digitais para estimar a direção na estrada (KONRAD *e outros*, 2012); mapas digitais e sensores de baixo custo, câmera e GPS (LANEURIT *e outros*, 2005), (MATTERN *e outros*, 2010).

Pode-se destacar ainda a utilização de ferramentas como o OpenStreetMap (OSM), de domínio público que oferece um rico conjunto de informações geográficas relacionadas a ruas, prédios, pontos de interesse, entre outros (HENTSCHEL E WAGNER, 2010). Alguns exemplos são apresentados na literatura, tais como: tarefas de localização e planejamento de trajetórias integradas a dados geográficos (HENTSCHEL E WAGNER, 2010); geradores de traços de mobilidade baseados em dados geográficos do OSM (GUNES *e outros*, 2010) e sistemas de localização para dispositivos móveis (RIFAT *e outros*, 2011). O OSM também permite integração com diferentes bases de dados, como é o caso do PostgreSQL e PostGIS, um exemplo é apresentado por Zheng *e outros* (2013).

Outro método tradicional na robótica móvel trata da localização e mapeamento simultâneos (SLAM, do inglês *Simultaneous Localization and Mapping*). Uma proposta de aplicação para SLAM é apresentada em Dissanayake *e outros* (2001); outra usando características naturais em ambientes externos pode ser vista em Guivant *e outros* (2002).

Localização por fusão de dados

A localização por fusão de dados combina as informações a fim de aumentar a confiabilidade do resultado (LANEURIT *e outros*, 2003), (YANG E SUN, 2007). Existem dois tipos de fusão de dados, a primeira combina dados a partir de um único sensor em momentos diferentes e a segunda combina dados de vários sensores e outras informações simultaneamente. O resultado aumenta a precisão da informação usando diferentes sensores ou dados, devido às características individuais de cada sensor.

Por exemplo, o encoder fornece velocidade angular e posição, porém os erros de posição podem ocorrer devido ao escorregamento de rodas, comum na robótica móvel. O giroscópio deter-

mina a orientação e o acelerômetro é utilizado para a determinação da aceleração linear do veículo. Além disso, o GPS indica a posição do veículo sobre uma estrutura de referência global. Assim, se estas observações são combinadas adequadamente, a posição estimada pode ser melhorada.

Alguns dos métodos de fusão sensorial mais conhecidos são os filtros de Kalman e partículas (LANEURIT *e outros*, 2003), (YANG E SUN, 2007). Alguns trabalhos que implementam a fusão sensorial através de algoritmos como o filtro de Kalman para a localização de robôs móveis e veículos autônomos, combinam dados de sensores como: GPS/INS (YANG E SUN, 2007); GPS e *dead reckoning* (XIAOYUN E HENG, 2008), (VENKATRAMAN *e outros*, 2010); odometria, GPS, LIDAR e visão (LANEURIT *e outros*, 2003); GPS, visão, (*Digital Maps*) e *Map-Matching* (TAO, 2012), (TAO *e outros*, 2013), (MATTERN *e outros*, 2010).

2.4 Simuladores de robôs móveis

Os simuladores virtuais são ferramentas úteis que oferecem possibilidades para a realização de testes de forma eficaz, evitando falhas no hardware, situações perigosas ou inesperadas, permitindo também a implementação de novos conceitos e algoritmos. Existem várias ferramentas disponíveis que permitem simulação 2D e 3D, com capacidade de recriar ambientes complexos para robôs móveis (KOENIG E HOWARD, 2004).

A seguir, são apresentados alguns dos simuladores mais comuns de código aberto e de livre uso, destacando-se suas principais características (DOS SANTOS, 2013):

- *Stage*¹ é um simulador multi-plataforma para robôs móveis, sensores e objetos para ambientes 2D. Geralmente é usado como um módulo *plugin* de *Player*². Tem suporte para linguagens de programação como C, C++, Python e Java através de *Player* ou ROS.
- *Simbad*³ é um simulador 3D só para Java, multi-plataforma, focado principalmente em fins educativos e científicos. Permite simulação de sensores, distâncias, etc, porém não inclui as propriedades físicas e a dinâmica de objetos.

¹<http://playerstage.sourceforge.net/index.php?src=stage>

²<http://playerstage.sourceforge.net/index.php?src=player>

³<http://simbad.sourceforge.net/>

- *USARSim*⁴ (do inglês *Unified System for Automation and Robot Simulation*) é um simulador 3D para Windows e Linux. Tem suporte para linguagens de programação como C, C++ e Java através de *Player* ou ROS. Pode simular odometria, distância, câmeras, sensores e paisagens, além de ambientes realistas e inclui manipulação das propriedades físicas e a dinâmica de objetos.
- *Gazebo* é um simulador 3D multi-robô só para Linux. Tem suporte para linguagens de programação como C, C++ e Python através de ROS. Pode simular odometria, sensores, paisagens, câmeras, ambientes com grau elevado de realidade, incluindo as propriedades físicas e a dinâmica de objetos.

Dos simuladores analisados, o Gazebo se mostrou mais adequado, pois oferece dispositivos para simulação em tempo real e fácil compatibilidade ao hardware (KOENIG E HOWARD, 2004). Algumas aplicações foram apresentadas na literatura, por exemplo, na criação e simulação de missões de busca e salvamento através de ambientes virtuais (DOS SANTOS, 2013); integrado ao ROS para permitir a simulação de robôs de escolha do usuário (LAUE *e outros*, 2005).

Os códigos escritos em qualquer uma das linguagens de programação através de ROS (do inglês *Robotic Operating System*), têm a vantagem de serem portáveis a plataformas robóticas reais (TRIQUELL, 2011).

⁴http://sourceforge.net/apps/mediawiki/usarsim/index.php?title=Main_Page

3 PROPOSTA DE UM MÉTODO DE LOCALIZAÇÃO REFERENCIADA

Este capítulo apresenta as noções referentes ao método de localização referenciada (MLR) bem como detalhes do filtro de Kalman Estendido (EKF), que realiza a fusão de todos os dados de localização disponíveis, a fim de compor o sistema de localização híbrida proposto. A primeira parte descreve o processo de localização referenciada dividido em três etapas, que se inicia com a localização do objeto de interesse, em seguida uma consulta é realizada na base de dados, em seguida uma correção da posição do veículo, concluindo o MLR. No passo final, com todas as informações disponíveis, estima-se a localização do veículo por fusão de dados utilizando o EKF.

3.1 Introdução

Veículos terrestres autônomos geralmente possuem diferentes tipos de sensores embarcados que fornecem várias informações, onde suas melhores características poderão ser utilizadas por diferentes métodos para ajudar a melhorar a localização do veículo em um ambiente urbano. Um destes métodos que diminui a imprecisão da localização é baseado na fusão de dados, dados estes adquiridos tanto de medidas obtidas de sensores embarcados, bem como de outras informações, como de dados de saídas de modelos matemáticos.

O método de localização referenciada, proposto aqui, compõe uma das entradas do método de fusão de dados e consiste em realizar uma estimativa mais precisa da localização do veículo baseando-se também na utilização dos sensores embarcados bem como do reconhecimento de objetos chave que é a forma de como as pessoas se localizam num ambiente, ou seja, quando uma pessoa que se encontra num ambiente desconhecido identifica alguma coisa que é reconhecida, então ela consegue saber por meio deste objeto onde é que ela se encontra.

A proposta portanto é de se utilizar informações fornecidas pelos sensores embarcados no veículo, de um sistema de visão que possua a capacidade de identificar os objetos chave do ambiente, de uma plataforma de simulação que permita criar um ambiente virtual e interagir com ele, e de uma plataforma de localização onde vão estar armazenadas todas as informações geográficas do ambiente em um mapa digital.

O método de localização referenciada (MLR) aqui proposto é composto pelos seguintes sensores embarcados: um GPS que fornece informações da posição em latitude e longitude, uma câmera monocular que detecta objetos de interesse, um laser que fornece a distância do objeto de interesse detectado pela câmera, e uma bússola que fornece a orientação do veículo.

A seguir serão detalhados os componentes, e o funcionamento, que compõem o sistema de localização híbrida proposto.

3.2 Eixos de referência do modelo de observação

Para o modelo de observação do MLR é necessário definir os eixos de referência do veículo e de cada um dos elementos do sistema, como do GPS, da câmera e do laser, em uma base de referência local comum.

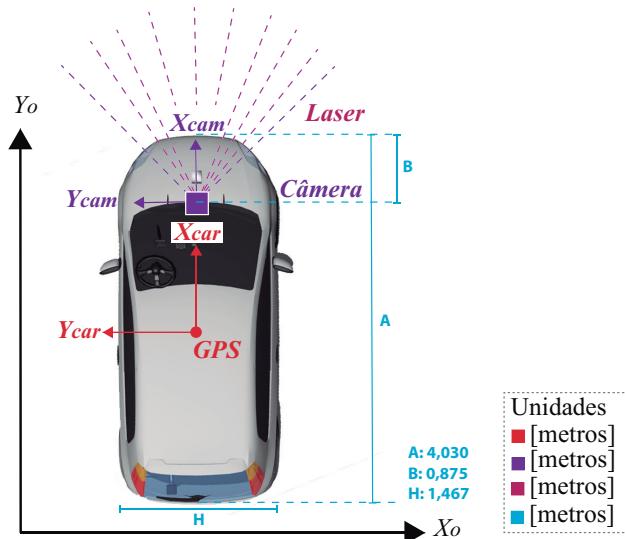


Figura 3.1: Eixos de referência do modelo de observação do método de localização referenciada.

O modelo do veículo a ser utilizado é um Fiat Punto baseado na plataforma robótica VILMA (Veículo Inteligente do LMA). Pode-se observar na Figura 3.1 as dimensões do veículo (FIAT, 2008), e o sistema de coordenadas local da câmera, do laser e do veículo dado pelo GPS, no quadro de referência local de navegação. É importante notar que: X_o, Y_o sistema de referência local de navegação, com o eixo X_o apontando para o leste, o eixo Y_o ao norte e o eixo Z_o para cima em relação ao Modelo de Referência Global WGS84; X_{car}, Y_{car} são as coordenadas do

GPS com referência ao veículo, o eixo X_{car} longitudinal e eixo Y_{car} lateral; X_{cam}, Y_{cam} representam o quadro de referência da câmera, com o eixo X_{cam} longitudinal e o eixo Y_{cam} lateral, e o quadro de referência do laser apresenta sua origem deslocada no eixo X_{car} longitudinal positivo em relação à câmera, a uma distância B .

3.3 Processo de localização do MLR

Conforme descrito anteriormente, o processo do MLR é composto por três etapas que são descritas a seguir.

A primeira etapa é apresentada na Subseção 3.3.1, localiza o objeto chave de interesse e possibilita obter a distância do mesmo através das informações fornecidas pelo laser.

A segunda etapa, na Subseção 3.3.2, transformações entre coordenadas planares e geográficas são realizadas, permitindo que consultas possam ser realizadas na base de dados geográficos.

Na terceira etapa, descrita na Subseção 3.3.3, a posição do veículo em relação ao objeto chave é estimada.

3.3.1 Etapa 1: Localização de um objeto de interesse

A primeira etapa, apresentada na Figura 3.2, tem como objetivo detectar objetos chave bem definidos, onde são conhecidas previamente as suas posições em coordenadas latitude e longitude. Em seguida, tendo sido o objeto chave detectado, informações do laser permitem obter a distância ao mesmo.

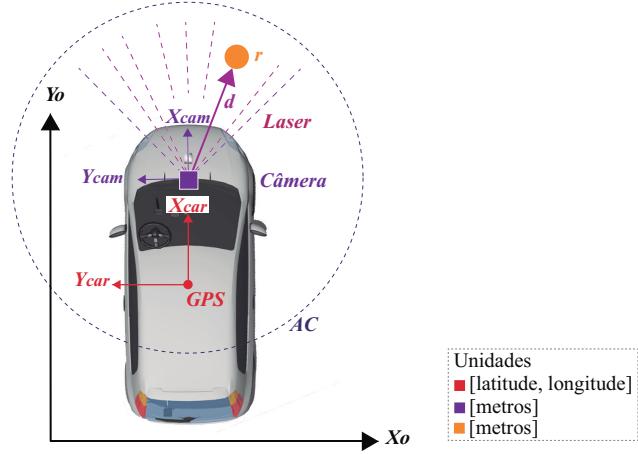


Figura 3.2: Medidas da câmera e do laser.

A Figura 3.2 apresenta um objeto conhecido r em uma área AC ao redor do veículo, tendo sido detectado por uma câmera monocular. Considerando que os sensores laser e câmera estão calibrados, uma varredura horizontal e vertical com o mesmo ângulo de abertura da câmera é realizada pelo laser, e o resultado final é apresentado na Figura 3.3.

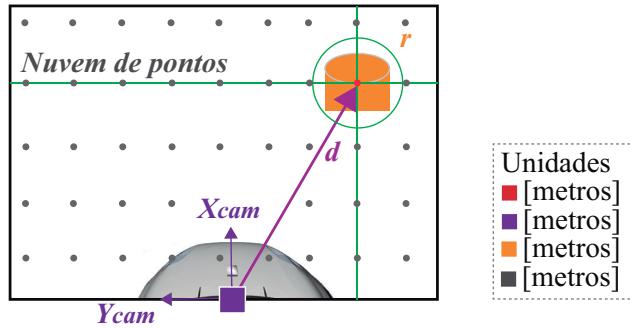


Figura 3.3: Representação da visão da câmera embarcada no veículo.

Quando a câmera identifica o objeto r no ambiente, o sistema vai representá-lo por um ponto. O laser retorna então a distância d , em metros, entre o ponto r e o centro da câmera X_{cam}, Y_{cam} . Mas é preciso conhecer esta distância d no sistema de referência X_o, Y_o . Então para que o ponto r possa ser interpretado em X_o, Y_o , é necessário realizar uma transformação homogênea, como ilustra a Figura 3.4.

A transformação é apresentada na Equação 3.1.

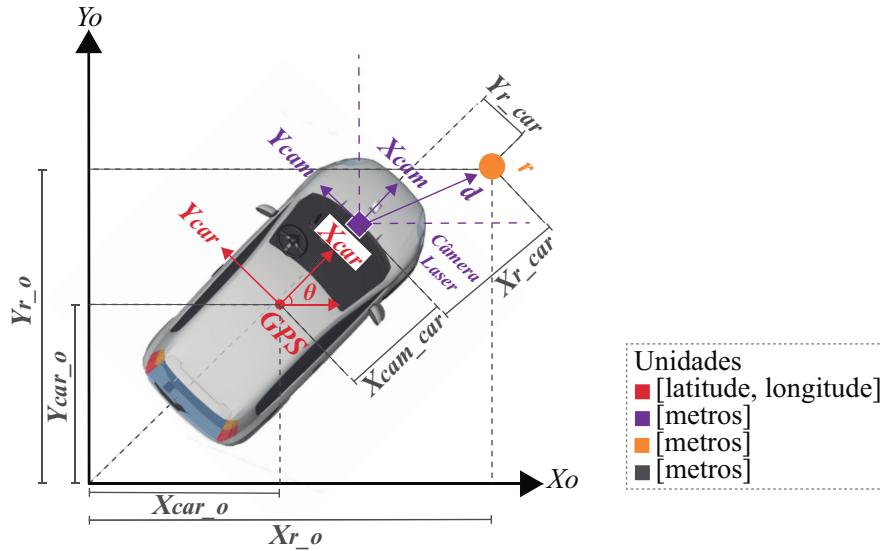


Figura 3.4: Transformada da posição do ponto r com respeito ao sistema de referência X_o, Y_o .

$$\begin{bmatrix} X_{r_o} \\ Y_{r_o} \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} X_{cam_car} + X_{r_cam} \\ Y_{cam_car} + Y_{r_cam} \end{bmatrix} + \begin{bmatrix} X_{car_o} \\ Y_{car_o} \end{bmatrix} \quad (3.1)$$

onde:

X_{r_o}, Y_{r_o} é a posição do objeto r em metros com respeito ao sistema de referência X_o, Y_o .

X_{car_o}, Y_{car_o} é a posição do centro geométrico do veículo com respeito ao sistema de referência X_o, Y_o .

X_{cam_car}, Y_{cam_car} é a posição da câmera-laser com respeito ao centro geométrico do veículo.

X_{r_car}, Y_{r_car} é a distância do objeto r à posição da câmera laser X_{cam_car}, Y_{cam_car} .

θ é o ângulo da orientação do veículo.

3.3.2 Etapa 2: Consulta na base de dados

A segunda etapa, apresentada na Figura 3.5, consiste em transformar a medida X_{r_o}, Y_{r_o} , obtida em metros, da posição do objeto em relação ao veículo para X_o, Y_o , em coordenadas geográficas dadas em latitude e longitude com relação à posição do objeto r com respeito ao mundo.

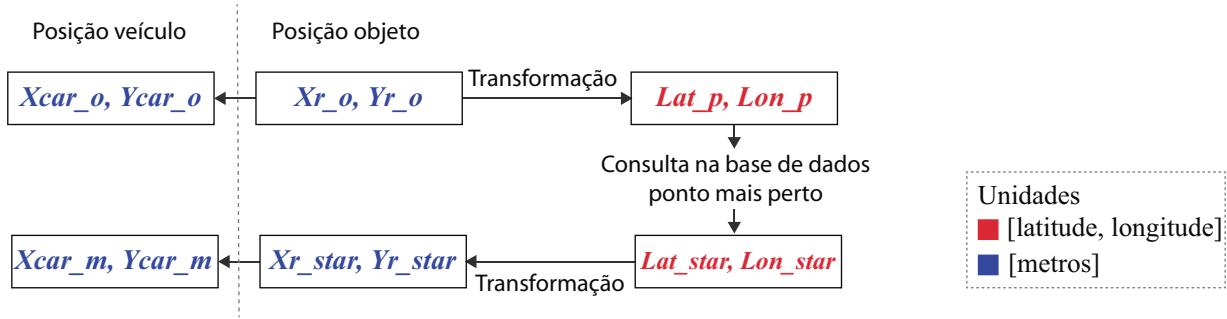


Figura 3.5: Resumo do processo proposto para medir, calcular e transformar os dados do sistema de visão.

Com os dados de latitude e longitude obtidos com base no ponto geográfico de referência ll como apresentado na Figura 3.6, pode-se fazer a conexão à base de dados que armazena as informações da posição correta do objeto r em latitude e longitude. Com esta informação, a base de dados retorna o objeto chave detectado, que tem seu posicionamento em longitude e latitude, uma transformação é realizada e permite representá-lo em coordenadas planares (em metros).

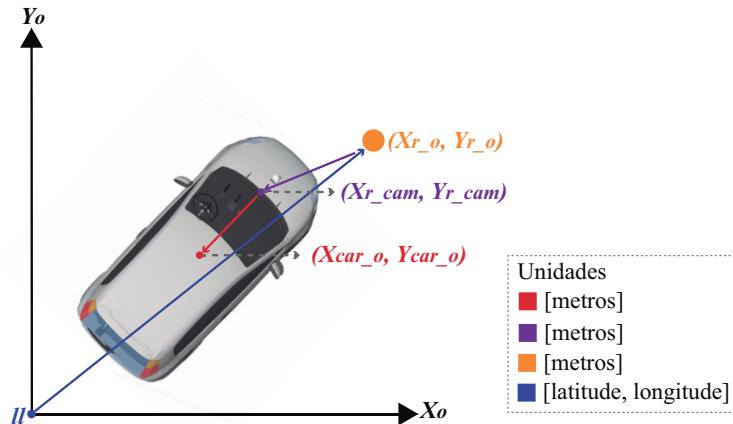


Figura 3.6: Transformada da posição do ponto Xr_o, Yr_o , dada em metros, para o sistema de coordenadas geográfico, latitude e longitude, tomando-se como referência a coordenada ll .

Portanto, a seguir são apresentados os passos e as equações de transformação de coordenadas planas dadas em metros a coordenadas geográficas dadas em latitude e longitude (ETKIN; STEVENS E LEWIS, 2012; 2003).

O primeiro passo da transformada é converter o ponto Xr_o, Yr_o em coordenadas Norte e

Leste, como mostrado nas Equações 3.2 e 3.3,

$$North = Xr_o \cos \psi - Yr_o \sin \psi \quad (3.2)$$

$$East = Xr_o \sin \psi + Yr_o \cos \psi \quad (3.3)$$

onde ψ é o ângulo em graus entre o eixo Xo e o norte, sentido horário.

Agora, para converter o Norte e Leste em coordenadas geográficas em latitude e longitude, utiliza-se para a estimação o raio de curvatura vertical principal RN e o raio de curvatura no meridiano RM , definidas como se apresenta a seguir nas Equações 3.4 e 3.5,

$$RN = \frac{R}{\sqrt{1 - (2f - f^2) \sin^2 \mu_o}} \quad (3.4)$$

$$RM = RN \frac{1 - (2f - f^2)}{1 - (2f - f^2) \sin^2 \mu_o} \quad (3.5)$$

onde R é o raio equatorial do planeta, f é o achatamento do planeta e μ_o é a latitude do ponto de referência ll .

O passo seguinte faz uma aproximação na latitude e longitude a partir da posição Norte e Leste, como mostra as Equações 3.6 e 3.7,

$$dLat = \arctan \left(\frac{1}{RM} \right) North \quad (3.6)$$

$$dLon = \arctan \left(\frac{1}{RN \cos \mu_o} \right) East \quad (3.7)$$

Finalmente, a latitude e longitude do ponto Xr_o, Yr_o são calculadas a partir da latitude e longitude do ponto de referencia ll , somando-se as variações medidas, como apresentado nas Equações 3.8 e 3.9,

$$Lat_p = \frac{180}{\pi} dLat + Lat_o \quad (3.8)$$

$$Lon_p = \frac{180}{\pi} dLon + Lon_o \quad (3.9)$$

onde Lat_p, Lon_p são as coordenadas do ponto Xr_o, Yr_o , agora expressas em coordenadas geográficas dadas em latitude e longitude.

Com as coordenadas do ponto r dadas por Lat_p, Lon_p que foram obtidas das Equações 3.8 e 3.9, uma consulta na base de dados é realizada para encontrar o objeto chave correspondente, como se ilustra na Figura 3.7. A construção da base de dados será discutida no Capítulo 4.

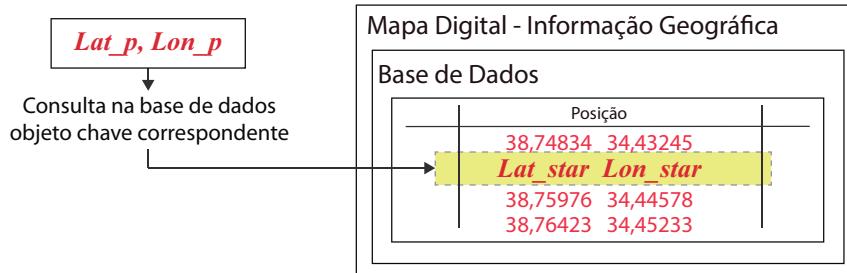


Figura 3.7: Ilustração, consulta do ponto Lat_p, Lon_p na base de dados.

Na consulta à base de dados, o ponto do objeto chave armazenado, correspondente ao ponto de Lat_p, Lon_p é denominado de lat_star, lon_star . Este ponto permite posteriormente volta de coordenadas geográficas a coordenadas planas.

3.3.3 Etapa 3: Correção da posição do veículo

A terceira etapa, apresentada na Figura 3.8, consiste em obter a posição estimada do veículo em relação ao objeto chave.

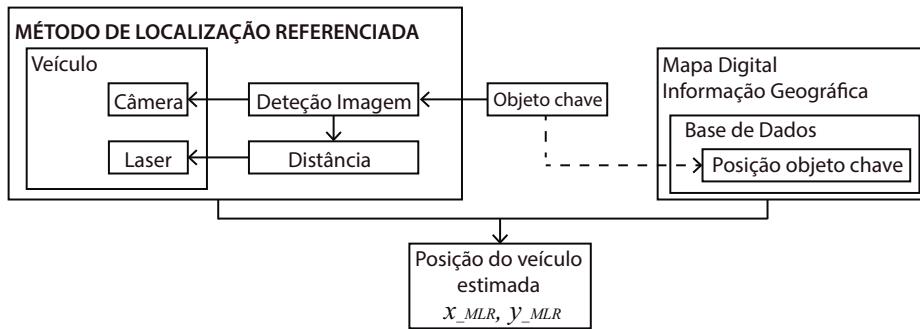


Figura 3.8: Método de localização referenciada para a localização estimada do veículo usando as informações da câmera, laser e a base de dados.

Com este ponto lat_star, lon_star de coordenadas exatas obtido da base de dados, conforme o mapa digital, calcula-se a posição do veículo. Em seguida, o procedimento é realizado novamente, onde o ponto Lat_star, Lon_star de coordenadas geográficas é transformado para coordenadas planas em metros. A seguir apresentam-se os passos e as equações de transformação de coordenadas geográficas dadas em latitude e longitude para coordenadas planas dadas em metros (ETKIN; STEVENS E LEWIS, 2012; 2003).

O primeiro passo é transformar as coordenadas Lat_star, Lon_star do objeto chave reconhecido no banco de dados em coordenadas $dLat, dLon$ no referencial do ponto de referência ll , como apresentado nas Equações 3.10 e 3.11

$$dLat = Lat_star - Lat_o \quad (3.10)$$

$$dLon = Lon_star - Lon_o \quad (3.11)$$

onde Lat_o, Lon_o são as coordenadas do ponto de referência ll .

Agora para transformar coordenadas $dLat, dLon$ em coordenadas $dNorth, dEast$ como se mostra nas Equações 3.6 e 3.7, é preciso utilizar o raio de curvatura vertical principal RN e o raio de curvatura do meridiano RM , onde RN e RM são definidas conforme apresentado nas Equações 3.4 e 3.5, resultando nas equações 3.12 e 3.13,

$$dNorth = \frac{dLat \left(\frac{\pi}{180} \right)}{\arctan \left(\frac{1}{RM} \right)} \quad (3.12)$$

$$dEast = \frac{dLon \left(\frac{\pi}{180} \right)}{\arctan \left(\frac{1}{RN \cos \mu_o} \right)} \quad (3.13)$$

onde μ_o é a latitude do ponto de referência em radianos.

Finalmente, pode-se transformar as coordenadas $dNorth, dEast$ da posição do ponto Lat_star, Lon_star em coordenadas planas Xr_star, Yr_star com a seguinte transformação, conforme apresentado nas Equações 3.14 e 3.15,

$$Xr_star = dNorth \cos \psi - dEast \sin \psi \quad (3.14)$$

$$Yr_star = dNorth \sin \psi + dEast \cos \psi \quad (3.15)$$

onde ψ é o angulo em graus entre o eixo Xo e o Norte, sentido horário.

Depois de obter a medida estimada Xr_star, Yr_star em coordenadas planas da posição Lat_star, Lon_star do objeto chave conhecido obtido da base de dados, deve-se obter a posição do veículo $Xcar_m, Ycar_m$ no eixo de referencial, conforme ilustrado na Figura 3.9.

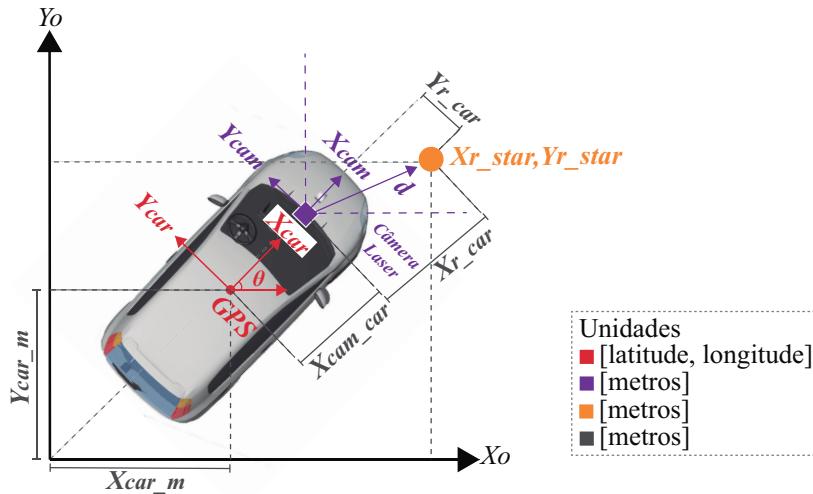


Figura 3.9: Transformada para calcular a posição do veículo $Xcar_m, Ycar_m$ a partir da posição do objeto Xr_star, Yr_star .

O cálculo da posição do veículo $Xcar_m, Ycar_m$ é obtido através da matriz de transformação homogênea, como mostrado na Equação 3.16.

$$\begin{bmatrix} Xcar_m \\ Ycar_m \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} -Xcam_car - Xr_cam \\ -Ycam_car - Yr_cam \end{bmatrix} + \begin{bmatrix} Xr_star \\ Yr_star \end{bmatrix} \quad (3.16)$$

onde:

$Xcar_m, Ycar_m$ é a posição medida do veículo em metros com respeito a Xr_star, Yr_star .

Xr_star, Yr_star é a posição medida do objeto com respeito à base de dados.

$Xcam_car, Ycam_car$ é a posição da câmera-laser com respeito ao centro geométrico do veículo $Xcar, Ycar$.

Xr_car, Yr_car é a distância da posição Xr_star, Yr_star à posição da câmera-laser $Xcam_car, Ycam_car$.

θ é o ângulo da orientação do veículo.

Finalmente se obtém uma nova medida da posição de veículo X_{car_m}, Y_{car_m} fornecida pelas observações do sistema de visão e das informações do banco de dados, como apresentado na Figura 3.10.

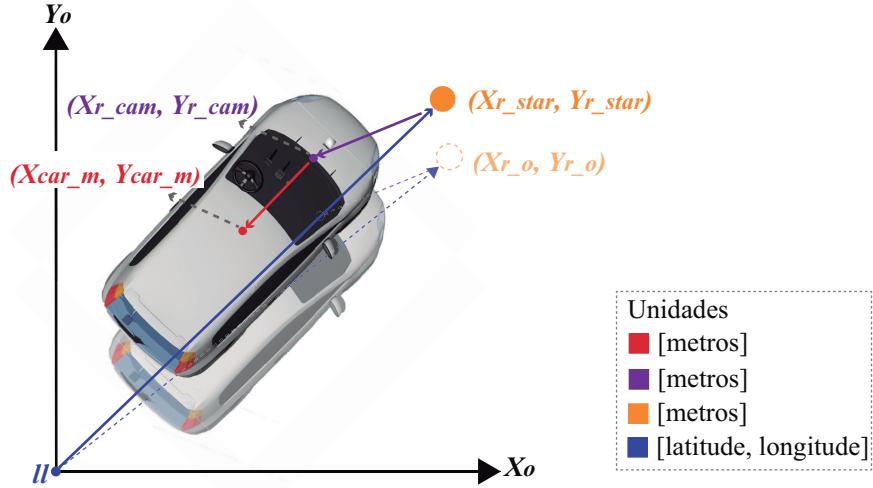


Figura 3.10: Correção da posição do veículo medida X_{car_m}, Y_{car_m} a partir das informações do sistema de visão artificial.

Na Figura 3.11 são apresentadas a posição $Pos1$ como resultado da posição do veículo dada pelo GPS de baixo custo e a posição $Pos2$ estimada calculada como resultado da posição do veículo dada pelo MLR.

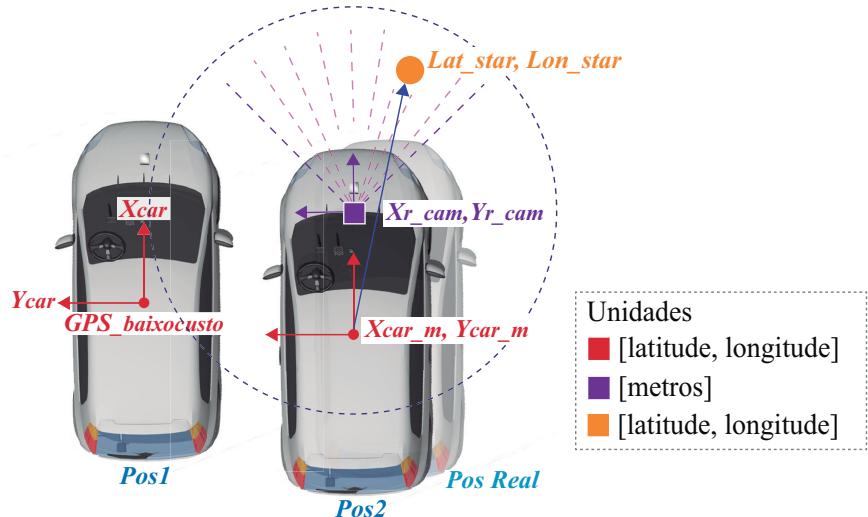


Figura 3.11: Localização pelo GPS de baixo custo $Pos1$ e localização pelo MLR $Pos2$.

3.4 Estimação da localização do veículo por fusão de dados

A estimativa da localização do veículo, mostrada na Figura 3.12, consiste em realizar a correção da localização do veículo por um método de fusão de dados. A figura mostra em detalhes como o sistema de localização está composto e as informações que serão utilizadas pelo método de fusão.

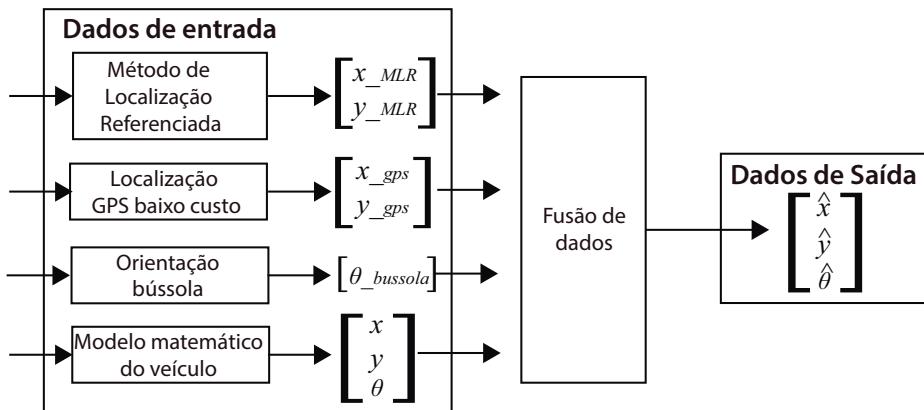


Figura 3.12: Composição do sistema de localização. Dados de entrada para a fusão de dados.

O método de fusão de dados escolhido foi o filtro de Kalman Estendido que está descrito em detalhes no Apêndice B. Serão utilizados no sistema proposto os dados provenientes dos sensores, e de cada um deles serão aproveitadas as suas características mais importantes, observando que algumas vezes alguns dos sistemas não estão disponíveis, as quais quando disponíveis vão ajudar a melhorar a medida de estimativa da localização do veículo em um ambiente urbano.

O método usa as informações dos dados da localização estimada obtida do MLR e das informações obtidas pelo sensor GPS de baixo custo, das informações de orientação do veículo dadas pela bússola, e das informações de saída do modelo matemático do veículo (ver Apêndice A).

O algoritmo do EKF é utilizado para estimar a posição do veículo $\hat{x}, \hat{y}, \hat{\theta}$. O esquema do funcionamento é apresentado na Figura 3.13, onde se tem os dados de entrada, tais como, a posição x_{MLR}, y_{MLR} dada pelo MLR, a posição x_{gps}, y_{gps} dada pelo GPS de baixo custo, a orientação $\theta_{bussola}$ dada pela bússola e a posição e orientação x, y, θ dada pelo modelo matemático do veículo.

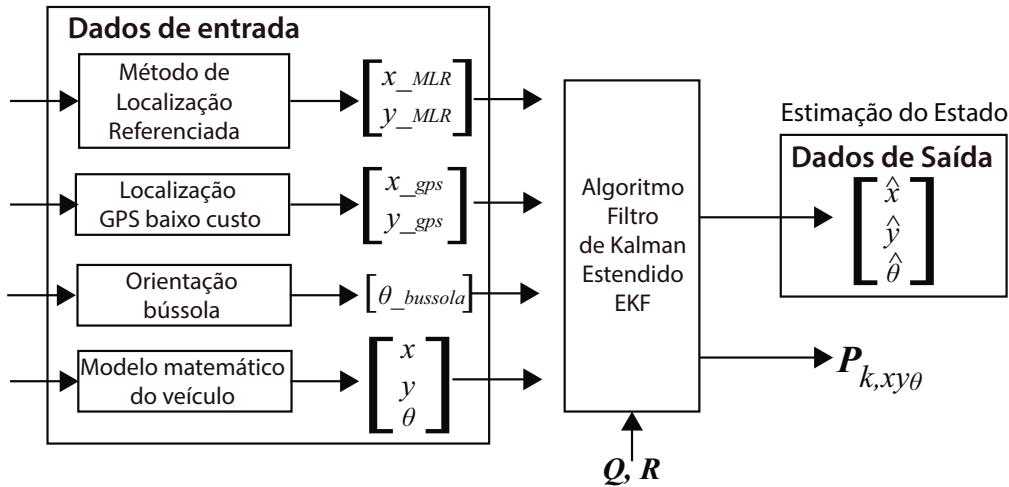


Figura 3.13: Esquema dos dados de entrada e saída do sistema completo que compõe o sistema híbrido para o EKF.

O detalhamento dos parâmetros utilizados no filtro de Kalman Estendido estão detalhados na Subseção 3.4.1.

3.4.1 Parâmetros do filtro

Para implementar o EKF foram definidos as condições iniciais do sistema e do modelo de observação, como foi apresentado na Seção B.2 nas Equações B.1 e B.2. Então tem-se que o sistema e modelo de observação estão definidos como apresenta-se nas Equações 3.17 e 3.18,

$$x = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \quad (3.17)$$

$$z = \begin{bmatrix} x_{GPS} \\ y_{GPS} \\ \theta \\ x_{MLR} \\ y_{MLR} \end{bmatrix} \quad (3.18)$$

onde x, y é a posição e θ é a orientação do modelo do veículo, e z é o modelo de observação do

sistema de medição.

Além das condições iniciais, é preciso conhecer os valores da matriz de covariância do processo \mathbf{Q} e a matriz de covariância das medições \mathbf{R} . A matriz de covariância do processo \mathbf{Q} vai definir a precisão do modelo que descreve o processo e os termos das diagonais correspondem às variâncias de cada estado (MARÍN, 2011). O resultado é apresentado na Equação 3.19,

$$\mathbf{Q} = \begin{bmatrix} w_x & 0 & 0 \\ 0 & w_y & 0 \\ 0 & 0 & w_\theta \end{bmatrix} \quad (3.19)$$

onde w_x , w_y e w_θ corresponde ao ruído aditivo ao processo. Como o processo vai ser simulado, o modelo do processo vai comportar-se em condições estáveis, isto é, não vai se apresentar deslizamentos ou escorregamentos no veículo, então o valor que vai ser definido para cada variável é um valor pequeno próximo a zero.

A matriz de covariância das medições \mathbf{R} pode-se obter através da medição da variância e covariância dos sensores utilizados. O processo para ajustar os valores da matriz \mathbf{R} geralmente é feita de forma manual, isto é, os valores pequenos na diagonal da matriz \mathbf{R} vão indicar que o sensor correspondente tem uma alta confiabilidade e assim a estimativa do filtro tenderá a ter uma maior confiabilidade nesta medida, e pelo contrário, um valor maior na diagonal faz com que o filtro use mais a informação da predição do modelo e/ou a medição de outros sensores (MARÍN, 2011).

No caso do processo de localização, na sequência das informações obtidas conforme destacado, nem sempre vão se ter todos os dados. Têm-se intervalos no tempo que o sistema de visão não está identificando objeto nenhum e somente vai se ter informações por exemplo do GPS de baixo custo e da bússola. No caso, têm-se duas matrizes \mathbf{R} , uma matriz \mathbf{R}_1 para quando se tem somente as medições do GPS de baixo custo e da bússola, e uma matriz \mathbf{R}_2 para quando tem-se todas as medições, GPS de baixo custo, bússola e MLR. Então as matrizes vão estar definidas como apresentadas nas Equações 3.20 e 3.22, junto com a matriz das medições da função \mathbf{H}_1 e \mathbf{H}_2 , respectivamente, segundo o vetor de estados da Equação A.1, como apresentadas nas Equações 3.21 e 3.23.

$$\mathbf{R}_1 = \begin{bmatrix} v_{GPSx} & 0 & 0 \\ 0 & v_{GPSy} & 0 \\ 0 & 0 & v_\theta \end{bmatrix} \quad (3.20)$$

$$\mathbf{H}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.21)$$

e,

$$\mathbf{R}_2 = \begin{bmatrix} v_{GPSx} & 0 & 0 & 0 & 0 \\ 0 & v_{GPSy} & 0 & 0 & 0 \\ 0 & 0 & v_\theta & 0 & 0 \\ 0 & 0 & 0 & v_{MLRx} & 0 \\ 0 & 0 & 0 & 0 & v_{MLRy} \end{bmatrix} \quad (3.22)$$

$$\mathbf{H}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.23)$$

onde v_{GPSx} e v_{GPSy} é o desvio padrão do ruido Gaussiano da medição do GPS baixo custo, v_θ é o desvio padrão da medição da bússola e v_{MLRx} e v_{MLRy} é o desvio padrão da medição do sistema MLR. Como o processo para se ajustar os valores das matrizes \mathbf{R}_1 e \mathbf{R}_2 geralmente é feito de forma manual, os valores que são atribuídos a cada sensor são baseados nos dados que são adquiridos de cada sensor.

3.4.2 Fase de predição

Na fase de predição, o EKF propaga o estado e a covariância para cada passo do tempo k . Então calcula-se a estimativa pelo modelo das variáveis de estado x, y, θ , segundo o modelo matemático apresentado no Apêndice A na Equação A.21. Então tem-se a Equação 3.24,

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + v_k \Delta t \cos \theta \\ y_k + v_k \Delta t \sin \theta \\ \theta_k + \frac{v_k}{L} \Delta t \tan \omega \end{bmatrix} \quad (3.24)$$

onde x_{k+1} , y_{k+1} e θ_{k+1} calculam a variação da posição do modelo do veículo em cada passo do

tempo k , L é a distância entre as rodas dianteiras e traseiras do modelo do veículo (FIAT, 2008), v é a velocidade linear do modelo do veículo e ω é ângulo de direção do modelo do veículo.

Mas o sistema da Equação 3.24 é um sistema dinâmico não linear, sendo linearizado através de Séries de Taylor. O resultado é apresentado na Equação 3.25,

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & -v\Delta t \sin \theta \\ 0 & 1 & v\Delta t \cos \theta \\ 0 & 0 & 1 \end{bmatrix} \quad (3.25)$$

onde \mathbf{F} é a matriz Jacobiana que contém as derivadas parciais do sistema não linear com respeito ao estado x .

3.4.3 Fase de correção

A seguir, depois da fase de predição, se faz a correção das medidas, primeiro verificando se todas medidas estão ou não disponíveis. Então se não há todas as medidas tem-se a matriz \mathbf{R}_1 , \mathbf{H}_1 e o vetor das medidas \mathbf{Z}_1 como se apresenta na Equação 3.26, e se há todas as medidas tem-se a matriz \mathbf{R}_2 , \mathbf{H}_2 e o vetor das medidas \mathbf{Z}_2 como se apresenta na Equação 3.27.

$$\mathbf{Z}_1 = \begin{bmatrix} x_{GPS} \\ y_{GPS} \\ \theta \end{bmatrix} \quad (3.26)$$

$$\mathbf{Z}_2 = \begin{bmatrix} x_{GPS} \\ y_{GPS} \\ \theta \\ x_{MLR} \\ y_{MLR} \end{bmatrix} \quad (3.27)$$

Posteriormente, se fazem os cálculos de acordo com as Equações B.5, B.6 e B.7 do Apêndice B. Primeiro se calcula o ganho do filtro de Kalman como se apresenta na Equação 3.28

$$\mathbf{K} = \mathbf{P} \mathbf{H}^T (\mathbf{H} \mathbf{P} \mathbf{H}^T + \mathbf{R})^{-1} \quad (3.28)$$

onde \mathbf{H} é a matriz das medições da função e \mathbf{R} é a medida do ruído das medições dos sensores. Finalmente, atualiza-se a estimativa da medida \hat{x}^+ e a covariância do erro \mathbf{P} , como se apresenta nas Equações 3.29 e 3.30.

$$\hat{x}^+ = \hat{x}^- + \mathbf{K}(\mathbf{Z} - \mathbf{H}\hat{x}^-) \quad (3.29)$$

$$\mathbf{P} = (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P} \quad (3.30)$$

4 PLATAFORMA DE LOCALIZAÇÃO E SIMULAÇÃO

Este capítulo apresenta o esquema da plataforma de localização e simulação baseada nas necessidades do método de localização referenciada e do sistema de localização híbrida apresentado no Capítulo 3. A primeira parte apresenta a construção da plataforma de localização baseada no OpenStreetMap para o desenvolvimento do mapa digital. Em seguida, apresenta-se a construção da plataforma de simulação baseada em ROS e Gazebo. Posteriormente são apresentadas as interfaces como resultado de cada plataforma.

4.1 Introdução

No projeto nasce a necessidade de propor e construir uma plataforma de localização e outra de simulação para fazer testes do método de localização referenciada, do sistema de localização híbrida e dos sensores de baixo custo vistos no Capítulo 3.

A plataforma é composta de três módulos principais que estão interligados, como apresentado na Figura 4.1. Estes três módulos são respectivamente um servidor de mapas, uma ferramenta de desenvolvimento de aplicações robóticas e um simulador de ambientes 3D.

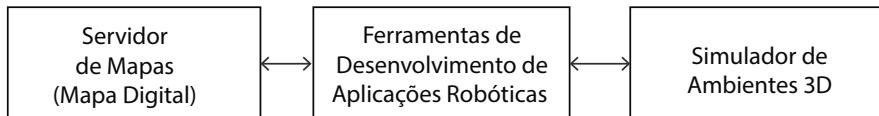


Figura 4.1: Módulos de composição da plataforma de localização e simulação.

O servidor de mapas permite obter as informações geográficas disponíveis em uma base de dados que compõe o mapa digital, além de permitir a edição, atualização e inclusão de informações geográficas de interesse. O simulador de ambientes 3D está ligado a uma ferramenta de desenvolvimento de aplicações robóticas, com o objetivo de simular a cinemática do modelo do veículo terrestre e de seus sensores.

O resultado da integração do mapa digital com o simulador de ambientes 3D e da ferramenta de desenvolvimento de aplicações robóticas é uma plataforma versátil, capaz de avaliar diferentes métodos de localização para veículos terrestres autônomos em um ambiente urbano em tempo real.

Os softwares utilizados para a construção da plataforma são OpenStreetMap, Gazebo e ROS. O OpenStreetMap como a base de informação geográfica para o servidor de mapas do mapa digital, o Gazebo como o simulador de ambientes 3D e o ROS como a ferramenta de desenvolvimento de aplicações robóticas.

4.2 Servidor de mapas

No mundo, os objetos possuem uma relação espacial entre elas, e um mapa consegue representar essa relação espacial de forma visual, mostrando onde os objetos se encontram. Com o objetivo de se ter as informações de um espaço geográfico contidas em um mapa, como por exemplo, cidades, rios, prédios, redes de ruas, etc, que estão armazenadas em uma base de dados e de poderem ser visualizadas em um mapa dinâmico, é preciso entender o funcionamento dos Sistemas de Informação Geográfico (SIG) e dos servidores de mapas (*MapServer*), e com base nessas informações, construir um servidor próprio de mapas (KILIMCI E KALIPSIZ, 2007).

4.2.1 Mapa digital - OpenStreetMap

A construção do servidor de mapas digital está baseada no software de mapas chamado OpenStreetMap. O OpenStreetMap¹ (OSM) é um mapa digital do mundo, criado por pessoas, de uso livre e sob uma licença aberta. É um projeto de mapeamento colaborativo para criar um mapa livre e editável do mundo.

4.2.2 Sistemas de informação geográfica

Os Sistemas de Informação Geográfica (SIG) (GIS, do inglês *Geographic Information Systems*) são aqueles que manipulam as informações geográficas e espaciais do mundo. Na literatura encontram-se outras definições como Galán e López (2003) que dizem que é um *Conjunto de ferramentas informáticas desenhadas para a aquisição, armazenamento, análise e representação de dados espaciais*, ou ainda como Teixeira e Chritofolletti (1997) que dizem que é um *Sistema base-*

¹http://wiki.openstreetmap.org/wiki/Main_Page

ado em computador, que permite ao usuário coletar, manusear e analisar dados georreferenciados.

Os SIG estão compostos basicamente por um sistema de informação convencional com dados de entrada e saída e uma base de dados para o armazenamento da informação. No entanto, pode-se estruturar um sistema web mais robusto, disponível através do modelo cliente-servidor utilizando o protocolo de comunicação e linguagem de programação HTML (GORNI e outros, 2007).

Então, para se criar um servidor de mapas próprio baseado nas informações de um mapa geográfico, o sistema necessitará ser composto de alguns componentes que serão listados a seguir (GORNI e outros, 2007):

- (i) Mapa (OpenStreetMap)
- (ii) Cliente (navegador de Internet)
- (iii) Servidor web (Apache)
- (iv) Linguagem de programação (JavaScript, Python, Shell, PHP, SQL)
- (v) Banco de dados espacial (PostgreSQL - PostGIS)
- (vi) Servidor de mapas (MapSever)

Com estes componentes implementados é possível obter as informações armazenadas em uma base de dados sobre os dados geográficos de uma área determinada pelas coordenadas de uma localização dada por um sensor GPS de baixo custo.

Para começar a construção do servidor de mapas, o primeiro passo é baixar os arquivos de dados do OSM referente a uma região do mundo através do site web <http://www.openstreetmap.org/>. Neste caso, este mapa digital desta região escolhida com suas informações geográficas vai ser denominada de *mapa.osm*.

Agora é preciso importar este mapa digital *mapa.osm* a um banco de dados espacial escolhido, como o banco de dados PostgreSQL - PostGIS. Como o OSM se compõe de um modelo de dados próprio, que não está dentro dos padrões dos SIG, se faz necessário utilizar o comando *osm2pgsql*,² que converte os dados de OSM para o banco de dados PostgreSQL - PostGIS.

²<http://wiki.openstreetmap.org/wiki/Osm2pgsql>

4.2.3 Banco de dados espaciais

Uma das características dos bancos de dados espaciais é sua capacidade de armazenamento e de realizar operações especiais com os dados armazenados, como calcular a distância de um ponto a outro, obter as entidades que estão mais próximas a um ponto, ou de filtrar informações por determinada característica, etc. O banco de dados PostgreSQL - PostGIS possue este conjunto de características, que facilitam a obtenção destas informações (GORNI *e outros*, 2007).

PostgreSQL³ é um sistema gerenciador de bases de dados objeto-relacional (SGBD) convencional de código aberto e livre. O sistema de PostgreSQL permite incluir *plugins* adicionais como o PostGIS⁴, que é uma extensão que habilita o PostgreSQL a tratar dados espaciais, adiciona suporte para objetos geográficos e além disso permite fazer consultas SQL de localização.

Então, quando é adicionado PostGIS ao módulo geográfico PostgreSQL, que tem funções de armazenamento e manipulação de dados geográficos, resulta em um banco de dados geográficos, como se mostra na Figura 4.2.

$$\text{PostgreSQL} + \text{PostGIS} = \text{Banco de Dados Geográficos}$$

Figura 4.2: PostGIS é uma extensão do PostgreSQL para tratar dados em um banco de dados geográfico.

Com as informações do *mapa.osm* no banco de dados espaciais, o seguinte passo é converter toda essa informação de pontos, linhas e polígonos, em imagens, onde se possa visualizar todo o conteúdo em um mapa dinâmico. Para realizar esta conversão se utiliza a biblioteca Mapnik⁵, que é uma ferramenta de uso livre multi-plataforma para a implementação de aplicações de mapeamento, e sua arquitetura lhe permite ler diferentes fontes de dados, entre eles, arquivos de dados PostGIS.

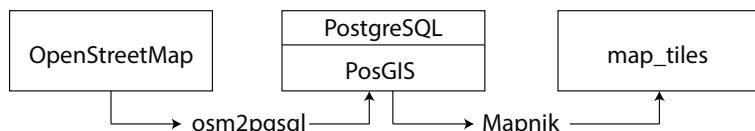


Figura 4.3: Processo de transformação de dados para a visualização do *mapa.osm*.

³<http://www.postgresql.org/>

⁴<http://postgis.net/>

⁵<http://mapnik.org/>

Como apresentado na Figura 4.3, a informação do OSM é convertida em *mapa.osm* e carregada em PostgreSQL com a extensão PostGIS fazendo uso do *osm2pgsql*. Uma vez que os dados foram carregados no banco de dados espacial, se utiliza do Mapnik para criar uma estrutura de pequenas regiões em imagens conhecidas como *map_tiles* e as prepara para o servidor de mapas.

Em resumo, os resultados obtidos até este ponto na criação do servidor de mapas podem ser visualizados na Figura 4.4, que mostra em forma de módulos como é composta e as conexões entre eles.

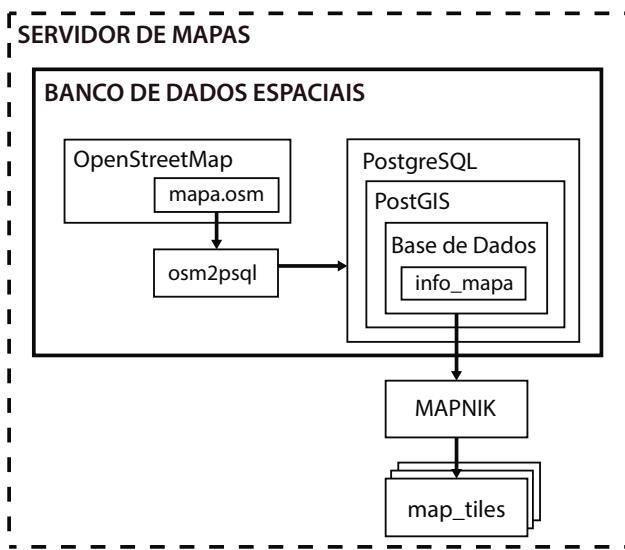


Figura 4.4: Módulos que compõem o Servidor de Mapas. OSM, PostgreSQL - PostGIS e Mapnik.

4.2.4 Servidor web

Para utilizar as imagens *map_tiles*, geradas pelo Mapnik, de forma dinâmica em um navegador da Internet, é preciso gerar uma estrutura que permita interagir com as informações que já foram obtidas, e também permita incluir e implementar novas funções, com o objetivo de poder interagir com estes dados disponíveis para gerar novos resultados.

O servidor Apache⁶ possui estas características, é um servidor web livre, compatível com o protocolo HTTP de código aberto, e fornece um serviço seguro, eficiente e extensível. Na base do servidor Apache, foi instalada Leaflet⁷ uma biblioteca JavaScript, livre e de código aberto para

⁶<http://httpd.apache.org/>

⁷<http://leafletjs.com/>

mapas interativos que aproveita HTML5 e CSS3 para criar mapas com um *design* mais simples, moderno e de fácil uso. Leaflet importa as imagens map_tiles através de funções JavaScript. Estas funções estão contidas em um arquivo HTML (do inglês *Hypertext Mark-up Language*), que finalmente será visualizado por meio do servidor *web* em um navegador da Internet, apresentando o resultado do *mapa.osm* agora convertido para *info_mapa* na base de dados.

A integração entre o servidor de mapas (OSM, PostgreSQL - PostGIS e Mapnik) e o servidor *web* (Apache, Leaflet, HTML), compõe o esquema do mapa digital que pode ser visualizado na Figura 4.5.

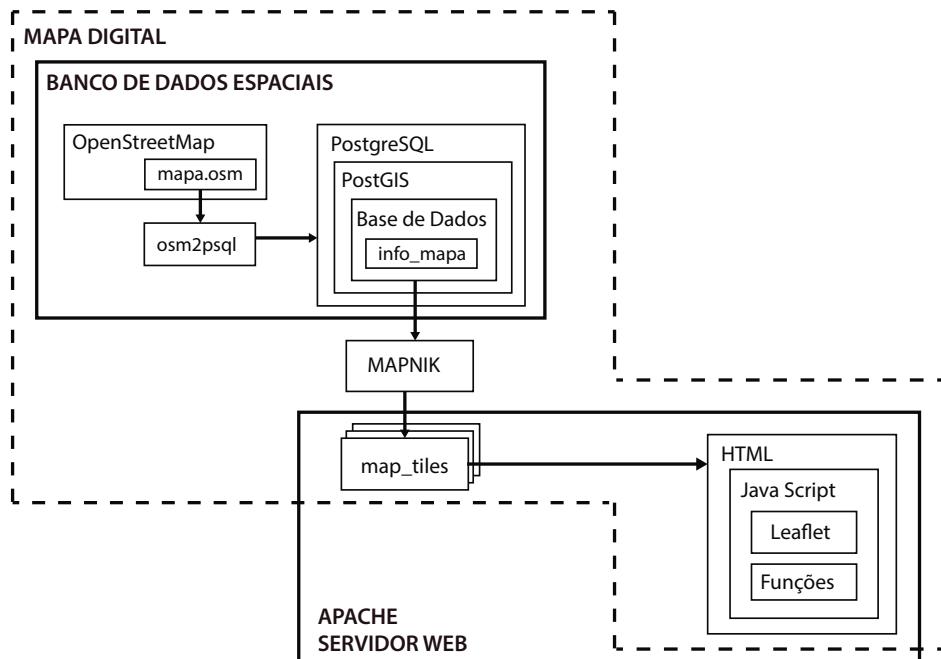


Figura 4.5: Integração entre o servidor de mapas (OSM, PostgreSQL - PostGIS e Mapnik) e o servidor *web* (Apache, Leaflet, HTML).

4.3 Plataforma de localização

Baseado na construção do servidor de mapas visto na Seção 4.2, na composição da interface da plataforma de localização procura-se dividi-la em campos, onde se consegue visualizar diferentes funções ao mesmo tempo, como a do mapa digital, das informações do mapa digital, das informações da posição atual enviada pelo GPS de baixo custo e das informações recuperadas da bases de dados, conforme mostrado na Figura 4.6.

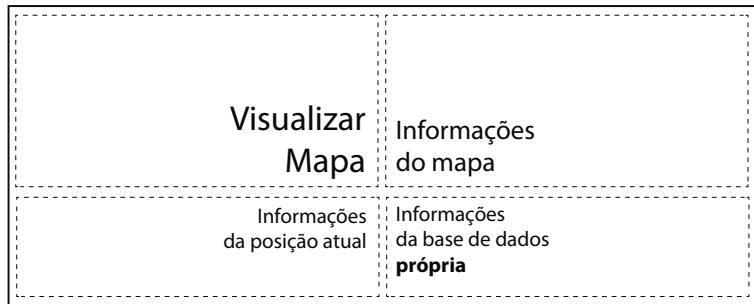


Figura 4.6: Interface da composição da plataforma de localização.

A seguir apresenta-se o processo de criação da plataforma de localização baseado nesta interface.

4.3.1 Mapa Digital

O resultado da integração dos componentes que formam o servidor de mapas é aqui chamado de mapa digital. A Figura 4.7 mostra a resposta gráfica do arquivo resultante `info_mapa` mostrado na Figura 4.5, que agora se visualiza em um navegador de Internet. No mapa dinâmico final, através de JavaScript, pode-se criar novas funções para acrescentar as informações visuais. Este mapa dinâmico final compõe o primeiro campo da plataforma de localização.

Para poder ter acesso à base de dados do `info_mapa`, que representa o banco de dados espacial, são executadas sentenças e consultas através da linguagem de consulta SQL, que permitem ter acesso a todos os dados, além de que permitem criar, editar, atualizar e armazenar informações novas, que é o objetivo principal da contrução de um servidor de mapas.

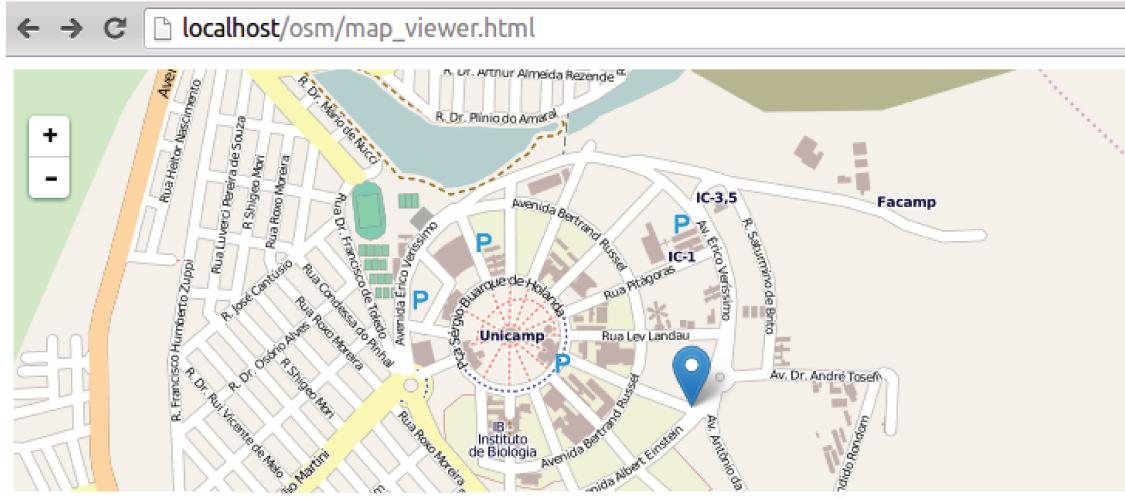


Figura 4.7: Visualização do mapa, resultado do processo feito com o arquivo *mapa.osm*, mais informações de latitude, longitude de um ponto localizado no mapa.

Na Figura 4.8 se visualizam duas tabelas. A imagem da esquerda representa a base de dados que contém todos os dados do *info_mapa*. A imagem da direita representa uma das tabelas da base de dados, que apresenta informações respectivas dos nomes de lugares e suas localizações em coordenadas latitude e longitude. A tabela da Figura 4.8 (b) compõe o campo das informações do mapa na plataforma de localização.

List of relations			
Schema	Name	Type	Owner
public	geography_columns	view	postgres
public	geometry_columns	table	gis
public	planet_osm_line	table	mafda
public	planet_osm_point	table	mafda
public	planet_osm_polygon	table	mafda
public	planet_osm_roads	table	mafda
public	spatial_ref_sys	table	gis

amenity	name	st_astext
school	Praça do Coco	POINT(-47.0826565934461 -22.8215708969913)
restaurant	Escola Estadual Barão Geraldo de Rezende	POINT(-47.0816566934462 -22.8246562969909)
cafe	Lótus Restaurante	POINT(-47.0812172934462 -22.8284172969905)
fast_food	Supermercado Pague Menos	POINT(-47.0812006934462 -22.8279491969905)
fuel	Hotel Son Inn	POINT(-47.0801656934464 -22.8227561969911)
fast_food	Fran's Café	POINT(-47.0798601934465 -22.8243036969909)
	Barão Geraldo	POINT(-47.0798320934465 -22.8218715969912)
	Subway	POINT(-47.0797976934464 -22.8245635969909)
	Auto Posto Campineira - Texaco	POINT(-47.0795361934464 -22.8252512969908)
	McDonalds	POINT(-47.0793719934465 -22.8256831969908)

Figura 4.8: Base de dados do *mapa.osm*. (a) Base de dados principal. (b) Dados de uma das tabelas da base de dados do *mapa.osm*.

4.3.2 Resultado da plataforma de localização

Com o sistema composto pelo mapa digital funcionando com relação ao objetivo proposto, o seguinte passo é fazer a conexão ao módulo de ROS. O objetivo desta etapa é conseguir obter, baseado nos dados da base de dados *info_mapa*, as informações de uma área definida ao redor da localização que o GPS de baixo custo, que está embarcado no modelo do veículo na plataforma de simulação, está fornecendo.

As informações do GPS de baixo custo do modelo do veículo, são publicadas pelo ROS através de ferramentas comuns disponíveis como a ferramenta *rostopic*⁸ de depuração, que exibe as informações sobre os tópicos de ROS. Fazendo uso desta ferramenta se obtém em cada instante do tempo um dado de localização como foi apresentado na Figura 4.8. Por outro lado, ao invés de usar ferramentas prontas para gerar páginas dinâmicas, decidiu-se aproveitar as vantagens que o servidor *web* oferece para implementar um *script* na tecnologia CGI⁹ (do inglês *Common Gateway Interface*), permitindo a um navegador *web* passar parâmetros a outros programas que estão no servidor web com as características específicas necessárias.

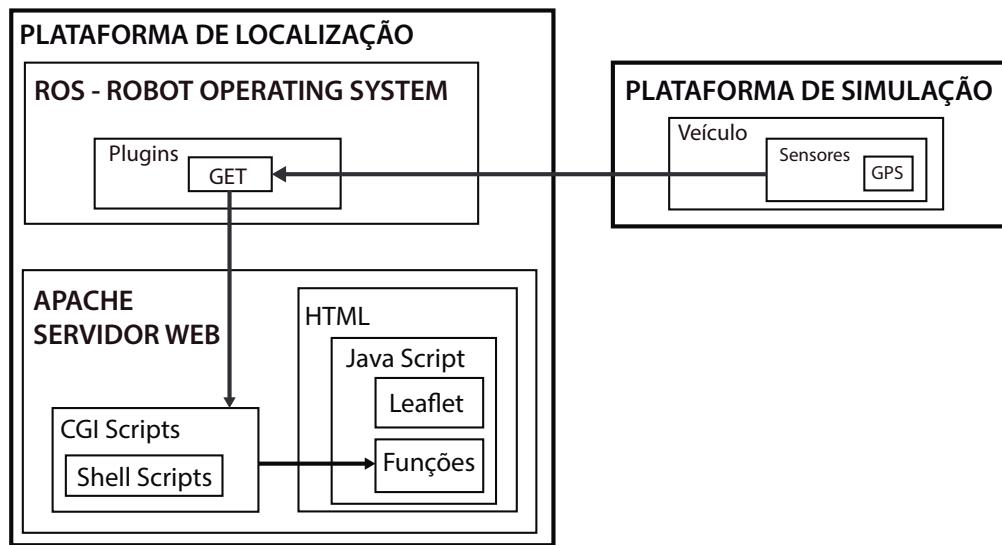


Figura 4.9: Conexão do GPS de baixo custo do modelo do veículo ao servidor de *web*. Interação entre a plataforma de localização e a plataforma de simulação.

Assim, fazendo uso do *rostopic* e do CGI, passam-se os dados publicados pelo GPS de baixo custo ao servidor de mapas. Na Figura 4.9 pode-se visualizar a conexão entre os módulos deste processo. O ROS obtém os dados do GPS de baixo custo através da linha de comandos *rostopic*, estes dados são enviados dinamicamente por um *script* CGI e os dados são recebidos por meio de uma função *JavaScript*. Os dados recebidos por *JavaScript* são utilizados para representar graficamente no mapa, por meio de um marcador, a localização atual do veículo, e assim, quando o modelo do veículo faz algum movimento, a posição do marcador é atualizada e apresentada para cada ins-

⁸<http://wiki.ros.org/rostopic>

⁹<http://www.w3.org/CGI/>

tante do tempo. Esta informação compõe o campo da informação da posição atual na plataforma de localização.

Agora, os dados de localização que o GPS de baixo custo em ROS está enviando, e a função do JavaScript que as está recebendo, vão ser utilizados para obter informações da base de dados info_mapa. Em outra função do JavaScript se recebem os dados que são enviados através de uma conexão feita em PHP¹⁰, que é uma linguagem interpretada de uso livre para o desenvolvimento de aplicações no lado do servidor. Com PHP, implementa-se um novo *script* que vai ter em sua estrutura sentenças e consultas SQL, e assim, primeiro se faz a conexão com a base de dados info_mapa e depois se seleciona a tabela com os campos de interesse a serem publicados, esta conexão de informação é representada na Figura 4.10, que apresenta a estrutura toda do servidor de mapas e do servidor web, tendo como resultado o mapa digital.

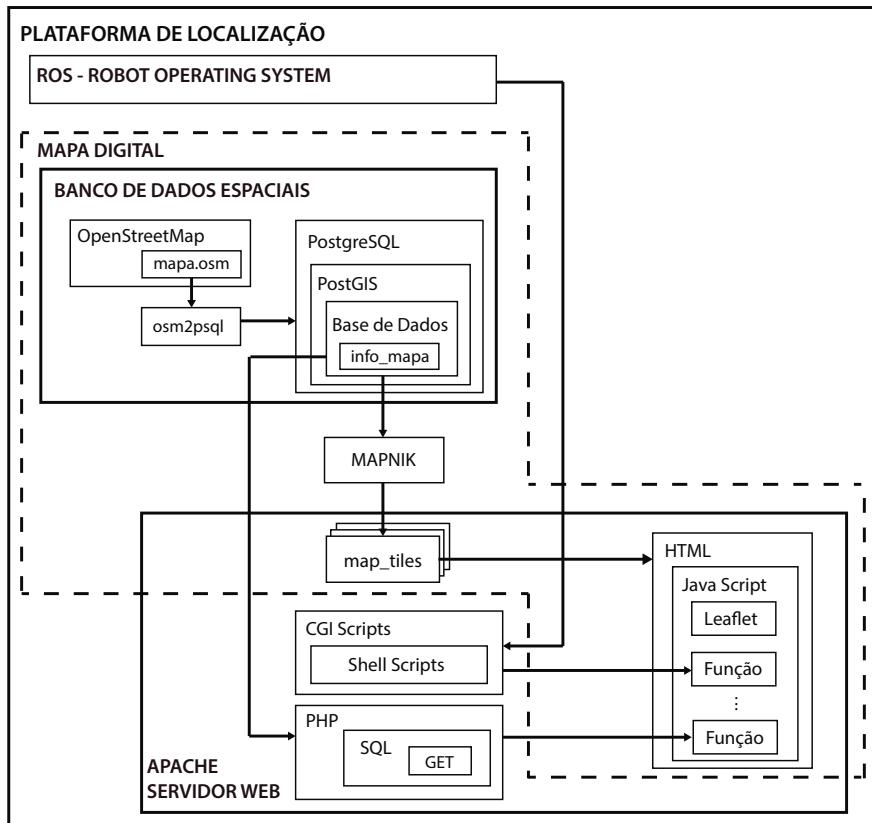


Figura 4.10: Módulos do mapa digital, composto pelo servidor de mapas e servidor web.

¹⁰<http://www.php.net/>

É preciso dizer, que muitas das informações contidas na tabela info_mapa, estão armazenadas em um tipo de codificação geográfico ou geométrico. PostGIS tem dois tipos padrão de expressar objetos especiais, a primeira forma é WKT do inglês *Well-Known Text* e a segunda forma é WKB do inglês *Well-Known Binary*, as duas incluem informações como as coordenadas que conformam o objeto e o tipo de objeto (POSTGIS, 2012).

Então, por meio de operações de transformação espacial próprias do PostGIS (POSTGIS, 2012), pode-se converter estes dados em informação legível em coordenadas de latitude e longitude, tudo isto feito no módulo PHP apresentado na Figura 4.10, que finalmente enviará estes dados que vão ser recebidos por uma função JavaScript e apresentados dinamicamente no arquivo HTML visualizado pelo navegador web junto ao mapa.

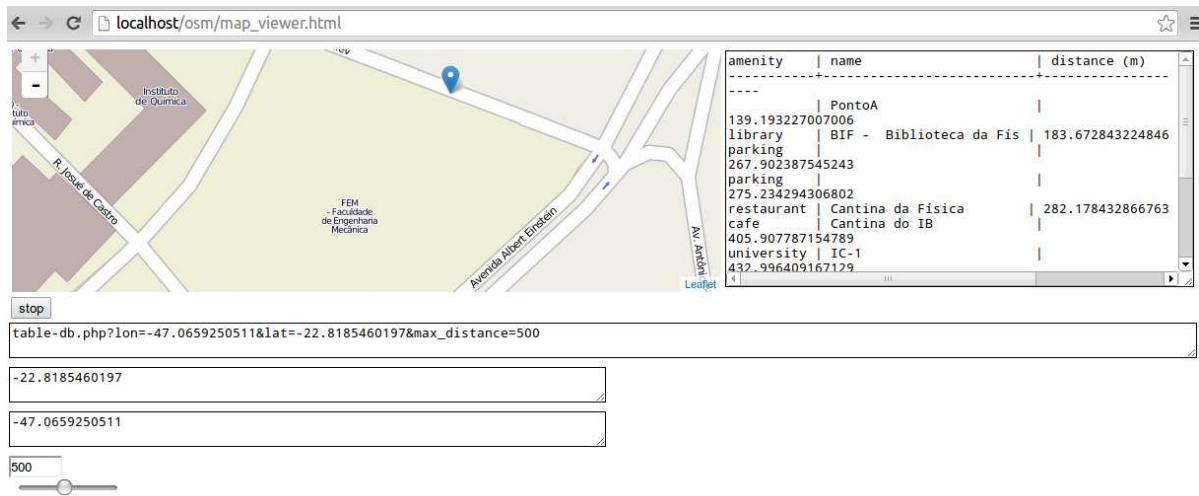


Figura 4.11: Visualização do mapa digital no navegador web junto as informações da base de dados e a posição do marcador em latitude, longitude.

O resultado deste processo é apresentado na Figura 4.11, de acordo ao proposto na interface da Figura 4.6, com um mapa dinâmico, o qual apresenta opções de *zoom-in*, *zoom-out* e *panning*. Nele também tem-se um marcador que vai ser atualizando segundo a informação enviada pelo GPS de baixo custo da plataforma de simulação, e além disso, vai ser apresentando em forma de texto nos campos que estão embaixo do mapa, como a latitude e longitude atual. Tem-se também a tabela que representa as informações da base de dados info_mapa. Estas informações são filtradas de acordo com a área ao redor do ponto atual do GPS de baixo custo, previamente definida, onde se tem informado o tipo de objeto, a descrição do objeto e a distância em metros do ponto do GPS de baixo custo ao ponto do objeto chave, calculado no módulo PHP através de sentenças SQL e operações de transformação espacial PostGIS.

Na Figura 4.11 as informações que são apresentadas na tabela, estão filtradas a uma distância de 500 metros ao redor do ponto que está marcando o GPS de baixo custo do modelo do veículo. Esta distância apresenta uma faixa de escolha de 0 a 1000 metros e estas informações podem ser alteradas enquanto o modelo do veículo está em movimento.

Para finalizar a construção do mapa digital e cumprir com o objetivo principal de ter uma base de dados com informações próprias, que vão ser posteriormente consultadas no processo do método de localização referenciada visto no Capítulo 3, vai ser criada uma tabela adicional com a ferramenta Psycopg¹¹, que é uma biblioteca que converte de PostgreSQL para Python e permite escrever *scripts* que criam, editam, leem, atualizam todas as funções e operações necessárias a se realizar em uma base de dados.

Assim, inclui-se uma tabela nova no banco de dados espaciais, como se apresenta na Figura 4.12, com informações de objetos chave como, nome, descrição e posição do objeto em coordenadas geográficas.

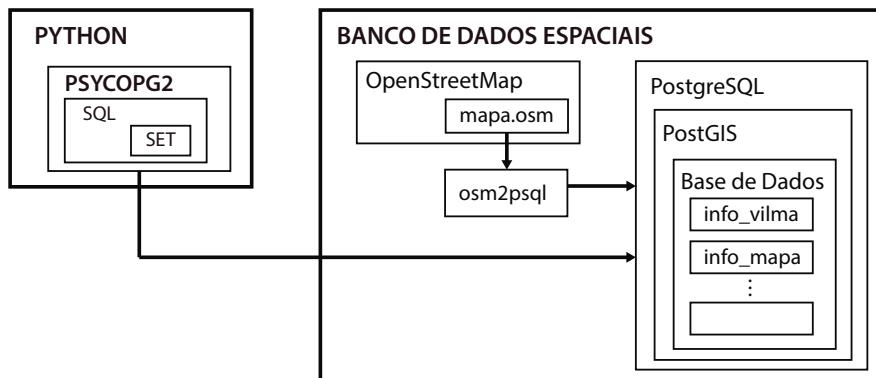


Figura 4.12: Adição da tabela info_vilma ao banco de dados espaciais com informações próprias para a localização do veículo.

Finalmente, esta informação compõe o campo das informações da base de dados própria na plataforma de localização.

¹¹<http://initd.org/psycopg/>

4.4 Plataforma de simulação

Na composição da interface da plataforma de simulação procura-se obter um campo onde se possa visualizar o ambiente 3D que contenha o modelo do veículo, os sensores embarcados no veículo e os objetos chave. Além disso, que a interação do módulo de simulação seja feita através de ferramentas de desenvolvimento de aplicações robóticas, como mostrado na Figura 4.13.

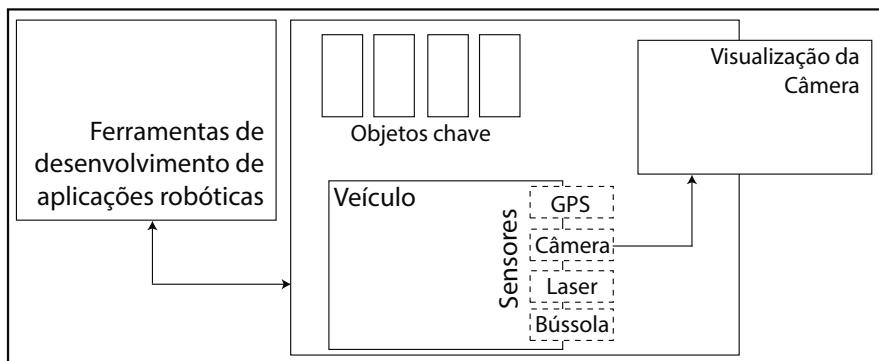


Figura 4.13: Interface da composição da plataforma de simulação.

Então, baseado nesta interface, apresenta-se a seguir todo o processo de criação da plataforma de simulação.

4.4.1 Gazebo

Gazebo¹² é um simulador 3D multi-robô apoiado pela *Open Source Robotics Foundation*¹³ (OSRF) sendo completamente livre e de código aberto.

Gazebo foi desenhado para contribuir no processo de desenvolvimento de algoritmos para plataformas robóticas, trabalha em ambientes internos e externos, e simula robôs, sensores e objetos refletindo com precisão a dinâmica do robô e tudo o que ele possa encontrar no seu entorno. O resultado é uma interface virtual que representa o mundo real com condições bem próximas à realidade (KOENIG E HOWARD, 2004).

¹²<http://gazebosim.org/>

¹³<http://www.osrfoundation.org/>

No projeto, foi preciso simular um veículo terrestre, e foi escolhido o veículo sendo preparado no LMA para realizar deslocamentos autônomos, um FIAT PUNTO. Para isso foi encontrado um modelo já feito de uso livre no banco de modelos 3D do SketchUp¹⁴ e se faz necessário realizar somente a sua importação ao Gazebo. Ao modelo do veículo vão ser embarcados diferentes sensores que fazem necessários para realizar a simulação desejada, como por exemplo GPS, câmera monovisão, laser, bússola, etc.

No caso do GPS, vai ser utilizado um sensor do pacote hector_gazebo_plugins¹⁵, que simula um receptor GNSS, que vai estar ligado ao modelo do veículo. A informação da posição do robô é dada em coordenadas WGS84 dadas em latitude, longitude e altitude, e o ponto de referência, que corresponde à origem, é configurado usando parâmetros XML. No caso da câmera, vai ser utilizada uma câmera simples embarcada na frente do modelo do veículo, como foi apresentado na Figura 3.1 do Capítulo 3, segundo o marco de referência. No caso do sensor laser, vai ser utilizado um Velodyne 3D LIDAR¹⁶, que obtém a medida da distância com respeito a um determinado objeto distante. A bússola vai-se simular para obter as informações de orientação do modelo do veículo.

O módulo do simulador de ambientes 3D é composto pelo simulador Gazebo, pelo robô representado pelo modelo do veículo terrestre e pelos seus sensores GPS, câmera, laser e bússola, como se mostra na Figura 4.14.

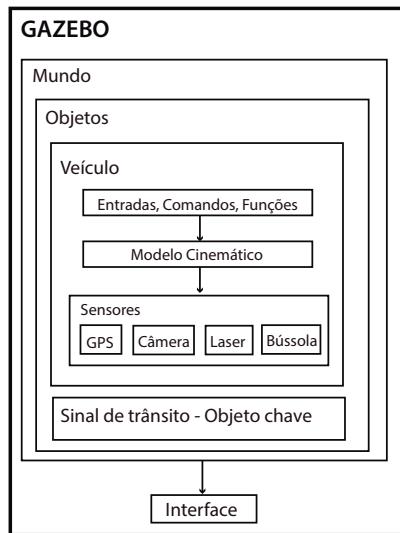


Figura 4.14: Gazebo, módulo de composição do simulador 3D.

¹⁴<https://3dwarehouse.sketchup.com/>

¹⁵http://wiki.ros.org/hector_gazebo_plugins

¹⁶<http://wiki.ros.org/velodyne>

O Gazebo é utilizado frequentemente em conjunto com o ROS, fornecendo um grande número de pacotes que são úteis para o controle dos robôs simulados ou físicos. Na plataforma de simulação o ROS vai fornecer as ferramentas para o desenvolvimento da aplicação robótica para o modelo do veículo terrestre através de *plugins*, que permitem compartilhar dados dos processos de acordo com o objetivo da simulação (DOS SANTOS, 2013).

4.4.2 ROS

ROS¹⁷ (do inglês *Robotic Operating System*), fornece as bibliotecas e ferramentas para ajudar os desenvolvedores a criar aplicações robóticas. ROS foi desenvolvido em 2007 pelo Stanford Artificial Intelligence Laboratory e é um sistema meta-operativo para robôs de código aberto (TRIQUELL, 2011).

ROS utiliza conceitos como nós, mensagens, tópicos e outros tipos de conceitos. Os nós são processos que executam cálculos e podem se comunicar com outros nós através de mensagens. As mensagens são uma estrutura de dados, onde um nó envia uma mensagem que é publicada em um tópico. Os tópicos são um canal de comunicação entre dois ou mais nós e a comunicação entre eles é através da publicação ou inscrição em tópicos (ANDRES *e outros*, 2013).

O ROS apresenta uma particularidade importante, que devido as suas características de conceitos de comunicação, se for adotada para o desenvolvimento de uma plataforma robótica, uma vez que o código do sistema com sistemas simulados foi desenvolvido, este código pode ser utilizado na aplicação real sem alteração. Devido a estas suas características, o ROS foi adotado aqui e no desenvolvimento da plataforma VILMA (Veículo Autônomo do LMA).

O Gazebo está ligado ao ROS através de *plugins*. Como apresentado na Figura 4.15, para a interação entre os módulos do ROS e Gazebo é preciso utilizar uma linguagem de programação que forneça a possibilidade de comunicação entre as duas ferramentas. Assim, para a construção da plataforma de teste utilizou-se a linguagem Python, que através da biblioteca tipo cliente Rospy¹⁸, permite uma rápida comunicação com os tópicos, serviços e parâmetros do ROS.

¹⁷<http://www.ros.org/>

¹⁸<http://wiki.ros.org/rospy>

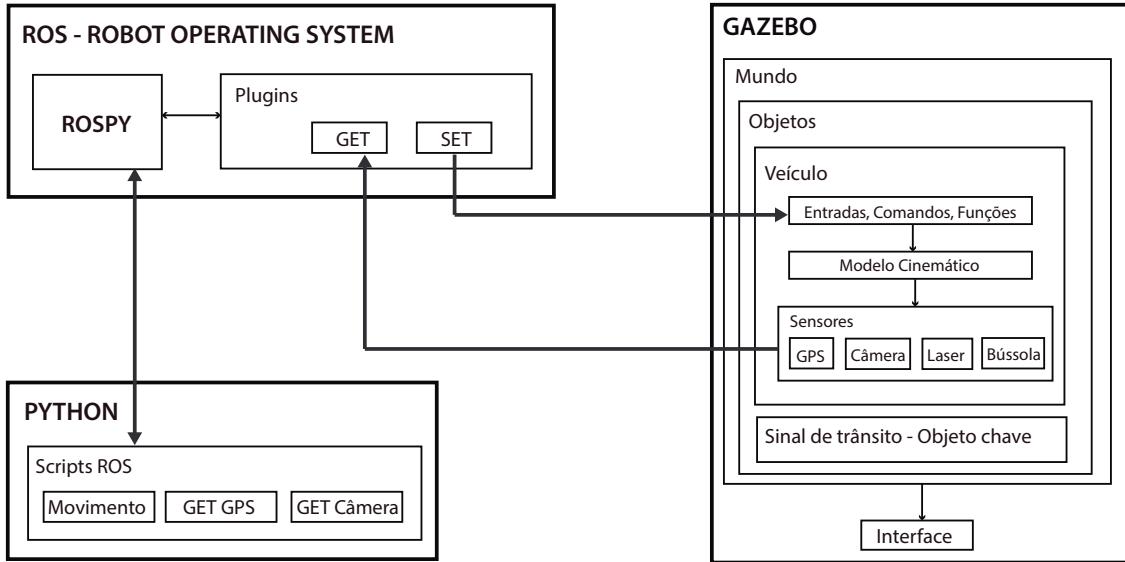


Figura 4.15: Representação da conexão entre os módulos do ROS, Gazebo e Python.

Na Figura 4.15, no módulo do Python, foram feitos alguns *scripts* para interagir com a plataforma. Tem-se o *script Movimento* que move o modelo do veículo de um ponto a outro, fazendo o modelo do veículo percorrer uma linha reta. Tem-se o *script GET GPS* que retorna as informações de captura de dados do GPS embarcado no modelo do veículo. Tem-se também o *script Get Câmera* que apresenta as informações de visão da câmera.

4.4.3 Resultado da plataforma de simulação

Os resultados da plataforma de simulação são apresentados na Figura 4.16, onde o percurso a ser realizado pelo modelo do veículo é feito operando-se manualmente os comandos do veículo, realizando as alterações nos valores do pedal do acelerador, do freio de mão e do pedal de freio, e o GPS é apresentado em coordenadas de latitude e longitude. Para o *script* da câmera foi implementado um algoritmo de reconhecimento de faixas de sinalização de ruas usando tanto a biblioteca do OpenCv¹⁹, também uma biblioteca de uso livre para o desenvolvimento de aplicações na área de visão computacional, como o CvBridge²⁰ que converte as imagens entre ROS e OpenCv.

¹⁹<http://opencv.org/>

²⁰http://wiki.ros.org/cv_bridge

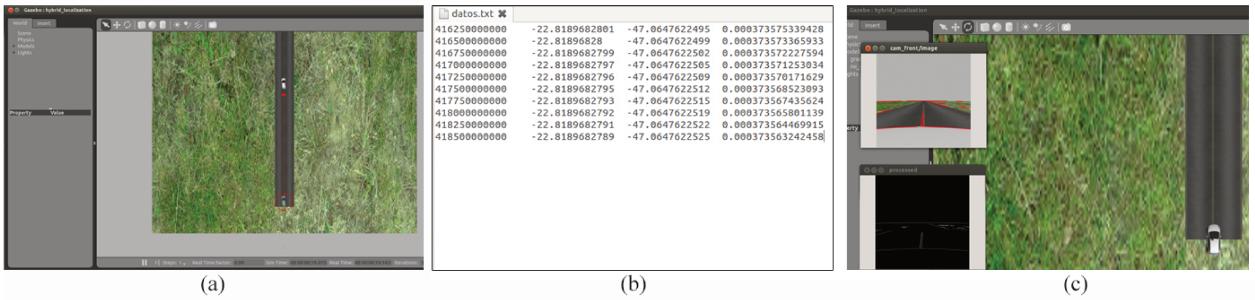


Figura 4.16: Resultados. (a) Percurso do modelo do veículo de um ponto a outro em linha reta. (b) Dados do GPS. (c) Informações da câmera.

Com os resultados obtidos, apresentados na Figura 4.16, o próximo passo é a criar a conexão com um sistema de informação geográfica, com o objetivo de poder ter acesso às informações geográficas do entorno armazenadas na base de dados.

4.5 Plataforma final

O resultado final da plataforma de localização e simulação está representada em duas camadas, uma camada *front-end*, que representa tudo o que é manipulado e visualizado através de uma interface gráfica pelo usuário, e uma outra camada *back-end*, que é tudo o que está do lado do servidor. A representação de todos os módulos conectados da plataforma de localização e simulação é apresentada na Figura 4.17. A parte da esquerda representa a camada *back-end*, que é o que está do lado do servidor, além de outros *scripts* de programação. A parte da direita representa a camada *front-end*, que é toda a parte de interface gráfica que interatua com o usuário.

Na Figura 4.18 se visualiza o resultado final da plataforma de localização. Aproveitando os recursos do HTML, se fez um *design* que permite visualizar todas as informações requeridas. Tem-se portanto do lado esquerdo o mapa digital com todas as informações da base de dados *info_mapa*, um marcador que é uma representação gráfica da posição atual do modelo do veículo, e embaixo do mapa dois campos que representam esta mesma informação com os valores de latitude e longitude. Ao lado direito têm-se duas tabelas, uma que representa as informações da base de dados *info_mapa* com a informação filtrada, de acordo com o raio de observação adotado centrado no ponto que está marcando o GPS, e a outra tabela representa as informações da base de dados *info_vilma* com as informações dos objetos, como sua posição exata em coordenadas latitude e longitude.

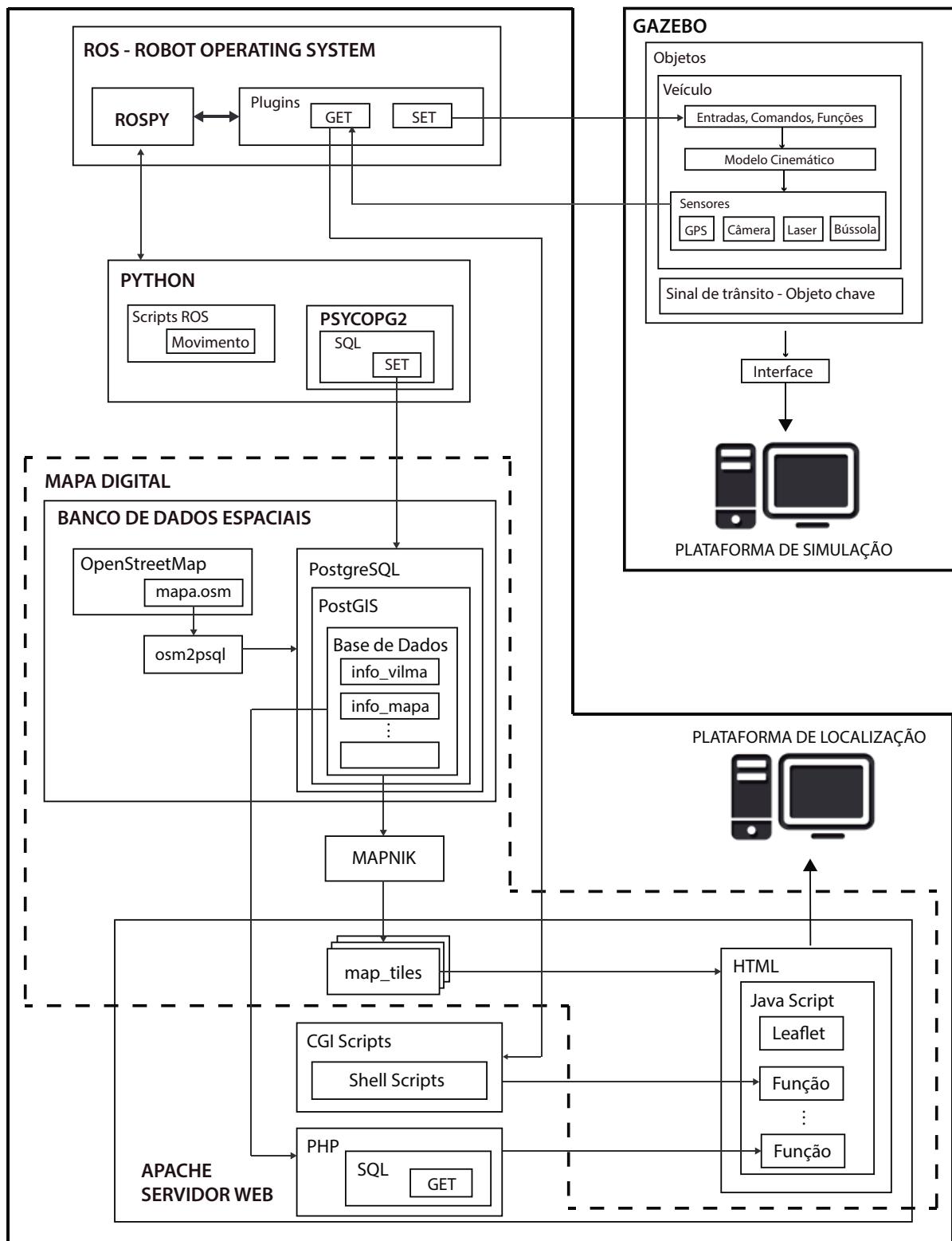


Figura 4.17: Esquema final completo da plataforma de localização e simulação.

Na Figura 4.19 se visualiza o resultado final da plataforma de simulação, onde apresenta-se a interface gráfica do Gazebo com o modelo do veículo terrestre, os sensores embarcados e os objetos chave no ambiente.

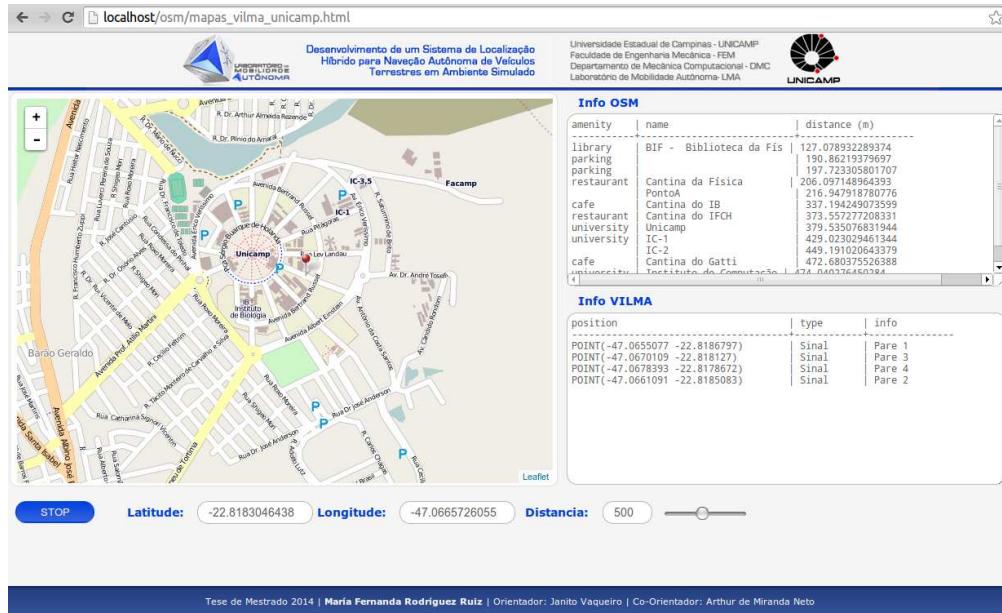


Figura 4.18: Plataforma final. Interface gráfica final da plataforma de localização.

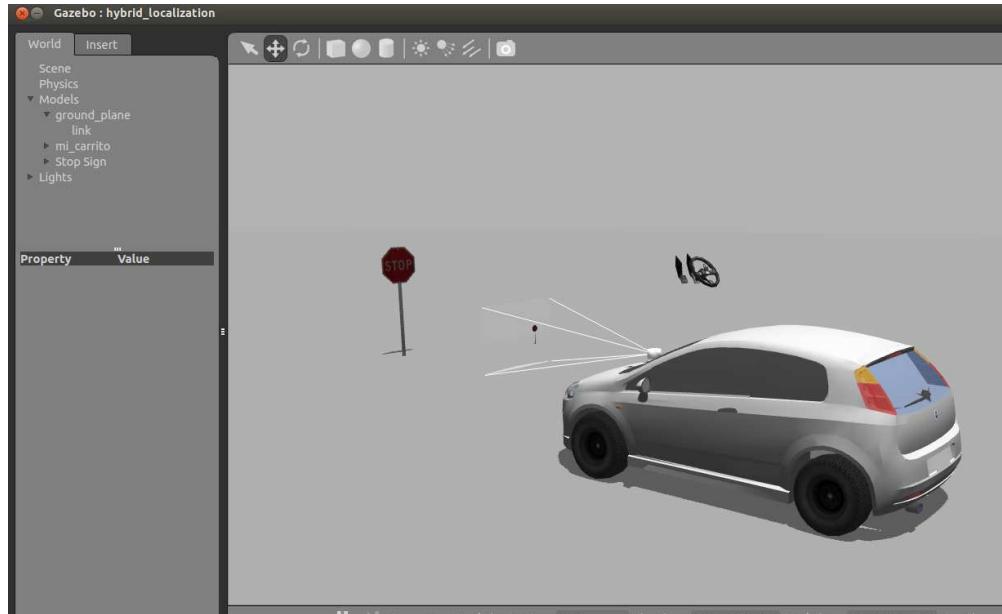


Figura 4.19: Plataforma final. Interface gráfica final da plataforma de simulação.

5 RESULTADOS NUMÉRICOS

Este Capítulo apresenta os resultados numéricos dos testes realizados na implementação do MLR e da estimativa da posição do modelo do veículo através da fusão de dados descrita no Capítulo 3, e das plataformas de localização e de simulação desenvolvidas e descritas no Capítulo 4.

Na Figura 5.1 mostra-se em detalhe como o sistema de localização híbrida desenvolvido está composto e portanto os módulos que serão testados. Deste modo, para obter a estimativa correta do estado $\hat{x}, \hat{y}, \hat{\theta}$ é necessário testar se os dados de entrada estão armazenados corretamente para realizar a fusão, se os resultados obtidos dos dados do GPS de baixo custo (GPS), do método de localização referenciada (MLR), da bússola (Buss.) e do modelo matemático do veículo (Modelo Mat.), e dos resultados calculados pelo método de fusão foram obtidos corretamente. Para isso, vários testes foram realizados e estão detalhados na seções deste capítulo.

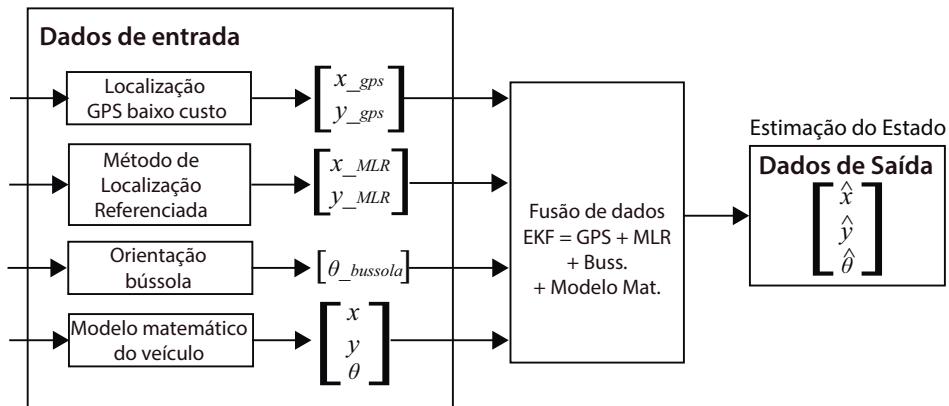


Figura 5.1: Sistema de Localização Híbrida.

Ao final do capítulo são apresentados os comentários dos resultados obtidos na Seção 5.6.

5.1 Plataforma de localização

Esta seção descreve os testes realizados na plataforma de localização. A simulação de referência dos testes consiste em usar o modelo do veículo FIAT Punto com dois sensores GPS, um que vai simular um GPS ideal *ground truth* como medida de referência e um outro que vai simular o GPS de baixo custo. O GPS ideal *ground truth* apresenta também como ponto inicial de referência -22,8189685, -47,0647617 em coordenadas latitude, longitude, localizado na Rua Mondelev

com Avenida Albert Einstein aproximadamente, na Universidade Estadual de Campinas, Campinas, São Paulo, Brasil, e seus dados são adquiridos sem interrupção e sem ruído. O GPS de baixo custo apresenta uma taxa constante de atualização de 100 milissegundos (equivalente a 10Hz) e ruído gaussiano equivalente ao desvio padrão do erro. Os dados foram tomados com base em um tipo de GPS de baixo custo simulado, onde se podem obter erros de posição desde 30m até de 100m (HIDE E MOORE, 2005). Baseado nas configuração dos dois GPS ligados ao modelo do veículo, realiza-se a simulação fazendo-se um percurso em linha reta de um ponto a outro no mapa.

Para efeitos de calibração dos dados durante a simulação, a captura dos dados de entrada é feita no mesmo instante de tempo, com uma frequência de amostragem de 5Hz. O percurso total é de aproximadamente 400m a uma velocidade de 5m/s.

O primeiro teste descrito na Subseção 5.1.1, descreve em detalhe a captura dos dados de entrada correspondente aos sensores GPS de baixo custo que vão ser simulados, fazendo a conexão entre a plataforma de simulação que vai simular os sensores que estão embarcados no modelo de veículo e a plataforma de localização. Em seguida é feito o segundo teste descrito na Subseção 5.1.2, onde são visualizados os dados capturados pelos sensores no mapa digital. Por fim, no terceiro teste descrito na Subseção 5.1.3 são apresentadas as medidas dos sensores GPS obtidos no percurso descrito.

5.1.1 Teste 1. Inserção de dados

O primeiro teste feito para a plataforma de localização, descrita no Capítulo 4, está baseado nas informações do GPS de baixo custo embarcado no modelo do veículo, que é simulado na plataforma de simulação, como foi apresentado na Figura 4.9, onde mostra-se a conexão do GPS de baixo custo ao servidor de *web* do mapa digital que representa a interação entre a plataforma de localização e a plataforma de simulação.

O GPS que vai ser simulado é um sensor do pacote *hector_gazebo_plugins*¹, que simula um receptor GNSS - Global Navigation Satellite System, que vai estar ligado ao modelo do veículo. A informação da posição do modelo do veículo é dada em coordenadas WGS84 dadas em latitude, longitude, altitude e o ponto de origem de referência é configurado usando parâmetros XML.

¹http://wiki.ros.org/hector_gazebo_plugins

Na Figura 5.2 tem-se os dois GPS no ponto inicial do percurso. O GPS ideal *ground truth* está representado pela cor vermelha, enquanto o GPS de baixo custo está representado pela cor azul. A tabela da base de dados Info OSM apresenta a informação filtrada num raio de 500 metros com respeito ao ponto inicial dos GPS, e a tabela da base de dados Info VILMA apresenta as informações dos objetos armazenados em info_vilma e que vão ser identificados pelo MLR (ver Capítulo3).

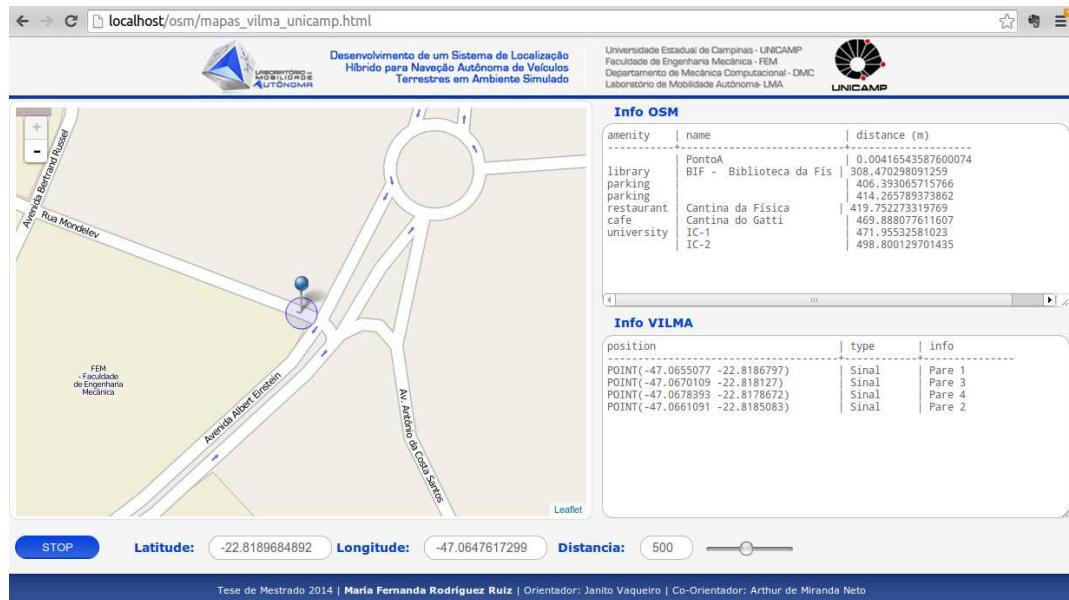


Figura 5.2: Resultado da plataforma de localização na posição inicial.

5.1.2 Teste 2. Visualização dos dados da plataforma

Com as condições iniciais configuradas, o seguinte teste consiste em visualizar a informação dos dados capturados através do movimento do modelo do veículo na plataforma de simulação. Na Figura 5.3 tem-se uma imagem amplificada do mapa com os marcadores em movimento. O percurso se inicia através de um *script* de Python que faz conexão com o ROS e movimenta o modelo do veículo da plataforma de simulação de um ponto a outro, fazendo trocas nos valores do pedal do acelerador, freio de mão e pedal de freio, como foi apresentado na Subseção 4.4.2 e na Figura 4.16.

A linha vermelha é marcada pelo GPS ideal *ground truth*, com captura de informação perfeita e sem interrupção. A linha azul é marcada pelo GPS de baixo custo, que tem ruído Gaussiano de

3.16, equivalente ao desvio padrão de 10 metros que é o raio do erro e da taxa de atualização equivalente a 10Hz. No percurso, a captura da informação tem interrupções, não é fluída, assim como sua variação da posição.



Figura 5.3: Resultado do GPS ideal e do GPS de baixo custo na plataforma de localização durante o percurso do modelo do veículo.

Após os dados obtidos neste teste, inicia-se a verificação dos dados obtidos pelo GPS de baixo custo, que simula mais de perto a realidade de um GPS do mercado. Os dados fornecidos pelo GPS ideal *ground truth* vão ser utilizados somente como uma medida de referência para avaliar a eficiência do resultado do sistema de localização híbrido, dos dados do GPS de baixo custo e dos dados do MLR.

5.1.3 Teste 3. Resultados dos sensores GPS

O terceiro teste apresenta os resultados dos sensores GPS, quando o modelo do veículo faz o percurso em linha reta de um ponto a outro, como se mostra na Figura 5.4.

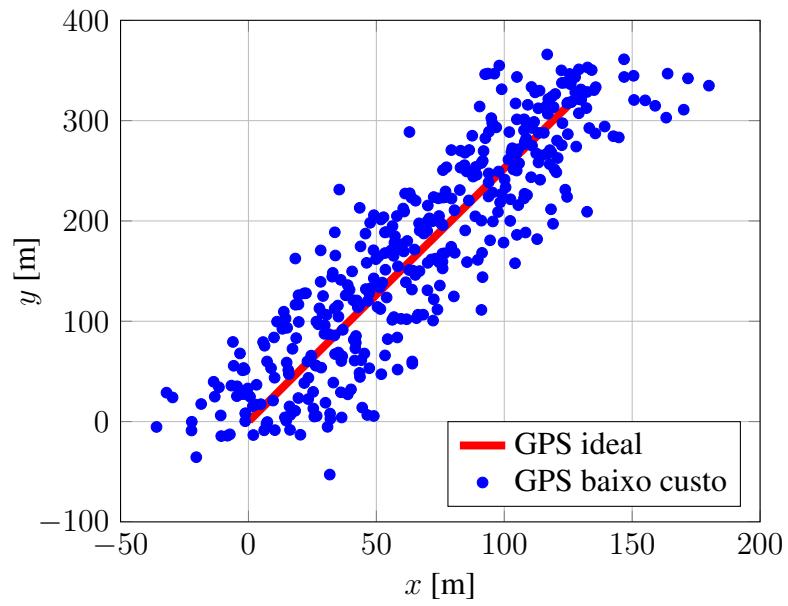


Figura 5.4: Teste 3. Medições obtidas na plataforma de localização. GPS ideal vs. GPS de baixo custo.

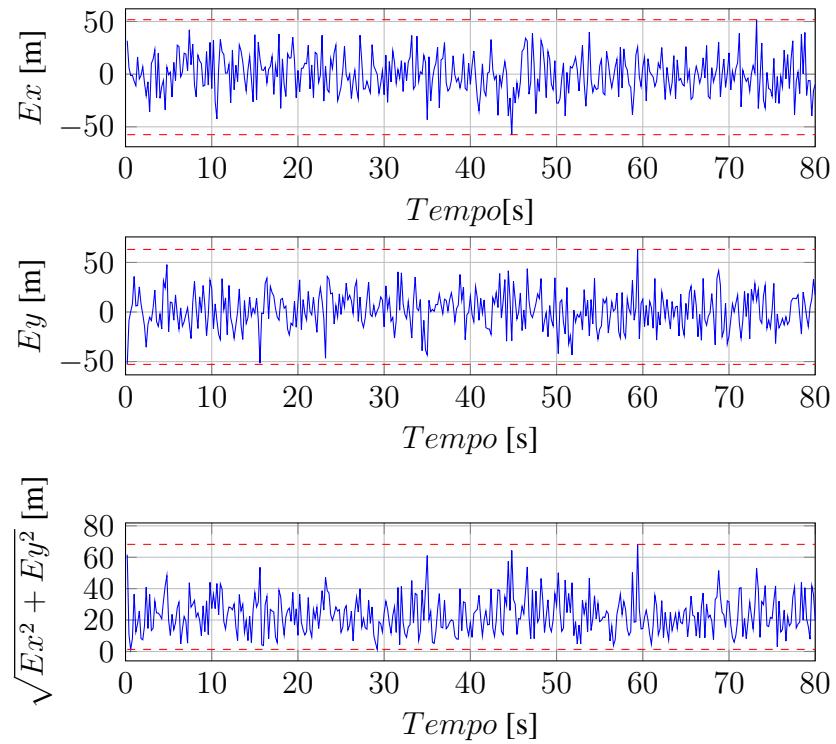


Figura 5.5: Erro da posição de cada ponto do GPS de baixo custo com respeito ao GPS ideal.

Na Figura 5.4, a linha da cor vermelha representa os dados capturados pelo GPS ideal e os pontos da cor azul representam os dados capturados pelo GPS de baixo custo. Posteriormente, é calculado o erro da posição de cada ponto do GPS de baixo custo com respeito ao GPS ideal. Então tem-se na Figura 5.5 os gráficas que mostram o erro calculado para direção x como sendo Ex , o erro para direção y como sendo Ey e o erro da distância sendo $\sqrt{Ex^2 + Ey^2}$, todos em unidades métricas ao longo do *tempo*.

Resumindo, os valores dos resultados dos erros do GPS de baixo custo com respeito ao GPS ideal são apresentados na Tabela 5.1. Na Figura 5.5 está apresentado o erro na posição dado pelo GPS de baixo custo, tendo um erro máximo de 68m e um erro mínimo de 1m aproximadamente, sendo a média de 23m do erro do GPS de baixo custo com um desvio padrão de 12m aproximadamente.

Tabela 5.1: Erros de localização do GPS de baixo custo com respeito ao GPS ideal.

Erro	GPS de baixo custo
Distância máxima	68,2m
Distância mínima	1,3m
Distância média	23,3m
Distância desvio padrão	11,9m

5.2 Plataforma de simulação

Esta seção descreve a inserção no mundo virtual de Gazebo dos modelos que vão representar os objetos chave na plataforma de simulação, segundo a proposta do MLR.

5.2.1 Teste 4. Objetos chave

De acordo com a proposta do MLR que foi apresentada no Capítulo 3 e as informações a que se vai ter acesso no mapa digital visto no Capítulo 4, são agregados modelos que vão representar os objetos chave.

Para começar, deve-se incluir no mundo de Gazebo os modelos que vão representar os objetos chave, onde as informações como a localização de cada um destes objetos estão armazenadas na

base de dados info_vilma, como foi apresentado na Subseção 4.3.2 e representado na Figura 4.12. O resultado é mostrado na Figura 5.6, onde tem-se na imagem um dos sinais de trânsito no mundo de Gazebo, que representa um objeto chave e a base de dados info_vilma, que apresenta as informações de localização exata destes objetos.



Figura 5.6: Objetos. (a) Modelo sinal de trânsito pare. (b) Base de dados info_vilma.

Com as informações disponíveis na base de dados info_vilma, onde tem-se armazenadas as características de localização de cada um dos objetos, é que os objetos são localizados em um ponto no mundo de Gazebo, com o objetivo de se ter uma única relação de informação e posição entre ROS, Gazebo e o mapa digital.

5.3 Sistema operacional do veículo autônomo

O sistema operacional do veículo autônomo é quem está lendo e recebendo todas as informações que estão sendo enviadas pelos sensores simulados na plataforma de simulação. Estes dados são processados pelo sistema operacional e utilizados para determinados fins, segundo as características de cada um dos sensores simulados. Deste modo, são lidas as informações do GPS, da câmera, do laser e da bússola.

Esta seção descreve os procedimentos e testes realizados para se detectar um objeto chave. Neste caso, o sistema operacional do veículo autônomo lê as informações que estão sendo enviadas pela câmera e o laser e estes dados vão ser utilizados para se detectar os objetos chave e medir a distância entre o objeto chave e o centro da câmera embarcada no modelo do veículo.

5.3.1 Teste 5. Detecção de objetos e medição da distância

Uma vez criado o cenário de simulação visto na Subseção 5.2.1, a primeira etapa deste teste se inicia com a captura da informação visual por meio da câmera embarcada no modelo do veículo simulado na plataforma de simulação. Então é implementado um *script* de detecção de objetos através da biblioteca OpenCv e CvBridge, onde o objetivo é identificar objetos da cor vermelha, aproveitando as características do sinal de pare. Os módulos que descrevem este processo são apresentados na Figura 5.7.

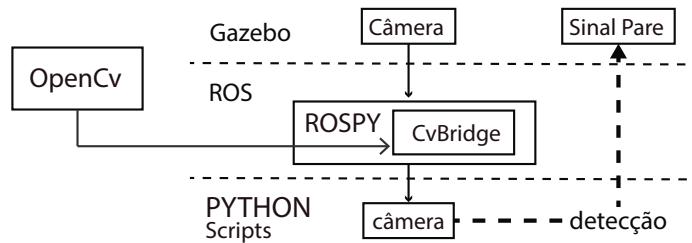


Figura 5.7: Módulos que descrevem a conexão entre ROS - Gazebo e OpenCv para a detecção dos objetos.

As imagens capturadas pela câmera que está ligada no modelo do veículo na plataforma de simulação são interpretadas pelo ROS e através da biblioteca CvBridge as imagens são convertidas entre ROS e OpenCv. Então por meio de um *script* no módulo de Python e implementando bibliotecas de reconhecimento de padrões com OpenCv e CvBridge, são detectados todos objetos de cor vermelha que estão no mundo de Gazebo, a uma distância máxima de 40m.

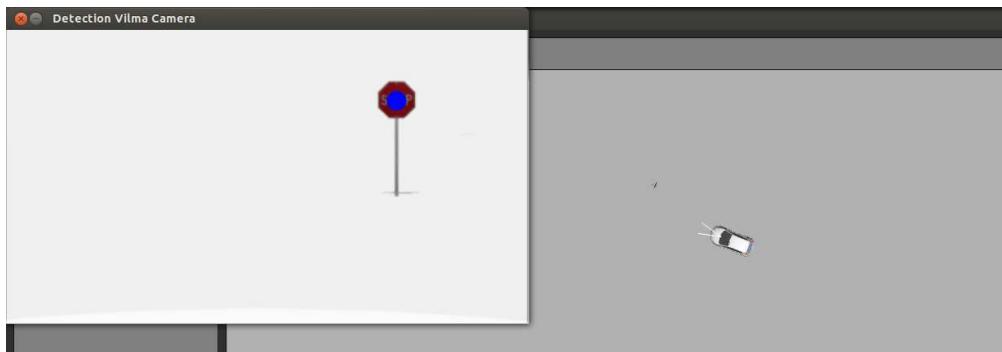


Figura 5.8: Resultado da detecção do sinal de pare, enquanto o modelo de veículo fazia o percurso.

Uma vez implementados os módulos de detecção de objetos chave, se realizou o teste de validação, que consistiu em verificar se os sinais de pare estavam sendo detectados enquanto o

modelo do veículo se movimentava fazendo o percurso. Um resultado deste teste pode ser visto na Figura 5.8, onde o sinal de pare detectado é destacado por um ponto da cor azul em seu centro.

Uma vez que os sinais de pare foram detectados durante o percurso do modelo do veículo através das informações fornecidas pela câmera, a etapa seguinte é conseguir medir a distância entre o centro da câmera ligada ao modelo do veículo e o sinal de pare. Para medir a distância, conforme foi descrito na Subseção 3.3.1, utiliza-se um sensor laser simulado tipo Velodyne Lidar ligado ao modelo do veículo e que se encontra na mesma posição da câmera.

Aproveitando-se desta posição coincidente com a câmera, os raios do laser fazem uma varredura horizontal e vertical no ambiente com o mesmo ângulo de abertura da câmera, retornando uma coordenada da posição em metros do ponto onde o raio do laser faz interseção com algum objeto. O resultado é um arquivo que contém todos os pontos da cena, e este número de pontos pode variar em relação ao número de raios horizontais e verticais que estão sendo projetados, de forma que quanto mais raios se utilizam, mais pontos vão-se ter, demandando um alto custo computacional.

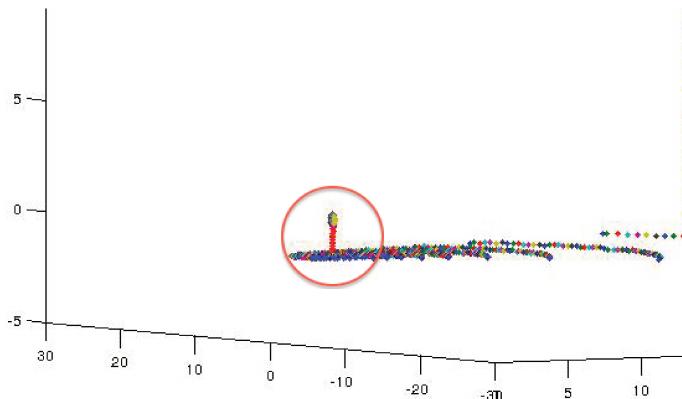


Figura 5.9: Resultado dos dados capturados pelo laser em um instante de tempo. Nuvem de pontos com as informações da cena.

Nesta etapa, foram utilizados 64 raios horizontais e 48 raios verticais para um total de 3072 pontos na cena. Na Figura 5.9 pode-se visualizar o teste de verificação da distribuição dos pontos na cena em um instante do percurso do modelo do veículo. Os dados foram importados no ambiente Matlab somente com a finalidade de se testar a informação obtida no processo de captura de dados do laser.

Dentro do círculo vermelho do resultado apresentado na Figura 5.9, pode-se visualizar o resultado dos pontos da interseção entre o raio do laser e de um dos sinais de pare. A etapa seguinte

é calibrar a imagem da câmera com a imagem gerada da nuvem de ponto do laser. Este processo é feito em Matlab, onde a nuvem de pontos é transformada em um mapa de profundidade. Esta calibração é verificada com o objetivo de se garantir que as informações geradas pela câmera coincidam com as informações geradas pelo laser, e assim, conseguir calcular a distância corretamente do objeto ao centro da câmera.

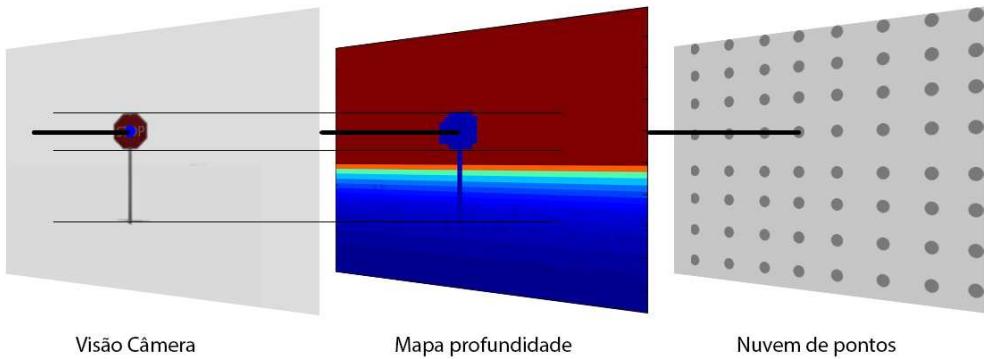


Figura 5.10: Resultado do alinhamento entre as imagens da câmera e o mapa de profundidade.

Na Figura 5.10 apresentam-se 3 imagens, a primeira representa a visão da câmera, a segunda representa o mapa de profundidade e a última representa a nuvem de pontos. Uma vez feita a detecção do objeto chave, o cálculo da distância é feito quando o ponto que indica o objeto detectado coincide com o ponto da nuvem de pontos feita pelo laser, e como se conhece as coordenadas deste ponto x,y,z , pode-se calcular a distância d , como se mostra na Equação 5.1.

$$d = \sqrt{x^2 + y^2 + z^2} \quad (5.1)$$

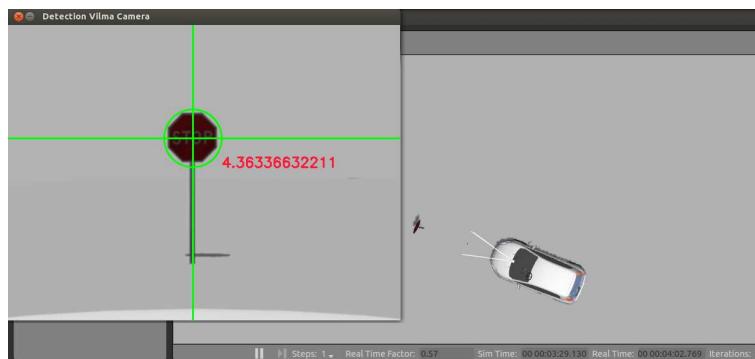


Figura 5.11: Resultado da distância medida entre o sinal de pare detectado e o centro da câmera ligada ao modelo do veículo.

No mesmo *script* de Python feito para a detecção de objetos, calcula-se este procedimento, e

o resultado retornado é a posição em metros entre o centro da câmera ligada ao modelo do veículo e o sinal de pare. O resultado é apresentado na Figura 5.11.

Finalmente, os módulos que descrevem o processo na detecção de objetos chave, neste caso representados por um sinal de pare, e a medição da distância entre o objeto chave e o centro da câmera embarcada no modelo do veículo são apresentados na Figura 5.12, representando a conexão entre a plataforma de simulação e o sistema operacional do veículo autônomo.

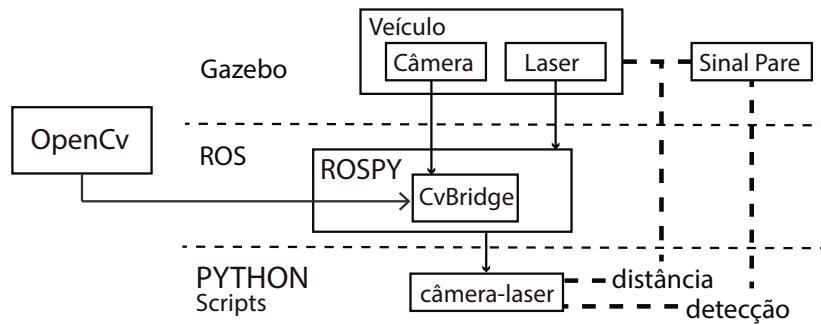


Figura 5.12: Módulos que descrevem a conexão entre a plataforma de simulação e o sistema operacional do veículo autônomo para a detecção dos objetos e medição da distância.

5.4 Localização referenciada

Com os resultados obtidos na captura de dados da câmera e do laser descrito na Subseção 5.3.1, conforme descrito no Capítulo 3 onde foi descrito o MLR, precisa-se fazer as transformações para os eixos de referência, para as coordenadas de latitude e longitude, e depois fazer as consultas à base de dados info_vilma.

Para desenvolver todos estes processos, gera-se no módulo de Python diferentes *scripts* que vão calcular os processos necessários para se obter uma localização estimada baseada no MLR para o modelo do veículo.

A seguir são descritas as etapas dos cálculos e testes realizados para se estimar a posição do modelo do veículo segundo o MLR apresentado na Subseção 5.4.1, e os resultados da posição estimada pelo MLR descrita na Subseção 5.4.2.

5.4.1 Teste 6. Etapas dos cálculos realizados

Para começar, os *scripts* em Python serão escritos em nós que vão publicar e subscrever mensagens dos dados obtidos por cada *script* baseado no próprio método de comunicação de ROS.

Como apresentado na Figura 5.13, tem-se o primeiro *script* do módulo Python, que corresponde aos cálculos feitos no processo já explicado na captura dos dados da câmera e do laser, tem-se também o segundo *script* que captura os dados do GPS de baixo custo necessário para calcular a posição do modelo do veículo, e finalmente o terceiro *script* que captura a orientação através de uma bússola. Os resultados de cada um destes *scripts* são publicados em um nó do ROS, para que posteriormente outro *script* que se subscreva a este nó possa utilizar esta informação para novos cálculos, como aqueles de transformação de coordenadas e consultas na base de dados info_vilma.

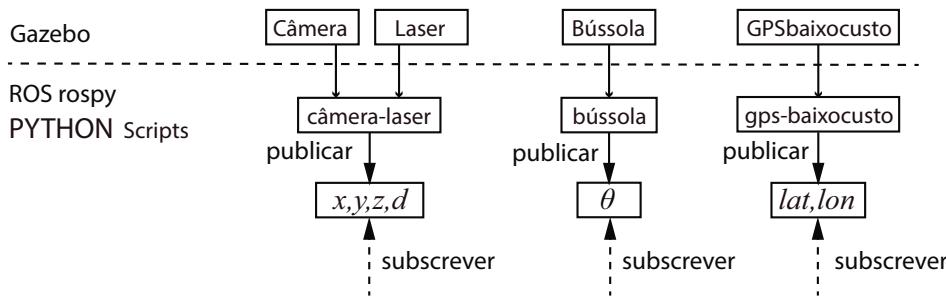


Figura 5.13: *Scripts* em Python dos cálculos de cada um dos sensores e os dados que são publicados em nós de ROS.

Então, segundo a Figura 5.13, tem-se que as informações da câmera e do laser são processadas no *script* câmera-laser e o resultado que se obtém deste processo é um ponto x,y,z com uma distância d desde o sinal de pare até o centro da câmera, e este resultado está sendo publicado em um nó. As informações do GPS de baixo custo são capturadas no *script* gps-baixocusto e os dados latitude, longitude são publicados em outro nó. E o *script* bússola vai publicar o ângulo θ que representa a orientação do modelo do veículo.

Com a estrutura feita na Figura 5.13 e segundo a Subseção 3.3.1, depois de se calcular a distância d do sinal de pare ao centro da câmera ligada ao modelo do veículo, precisa-se transformar o valor de x,y ao eixo de referência Xo,Yo . Para fazer esta transformação utilizam-se as informações dos *scripts* câmera-laser, bússola e gps-baixocusto e um novo *script* que vai fazer o cálculo, segundo a Equação 3.1, conforme representada na Figura 5.14.

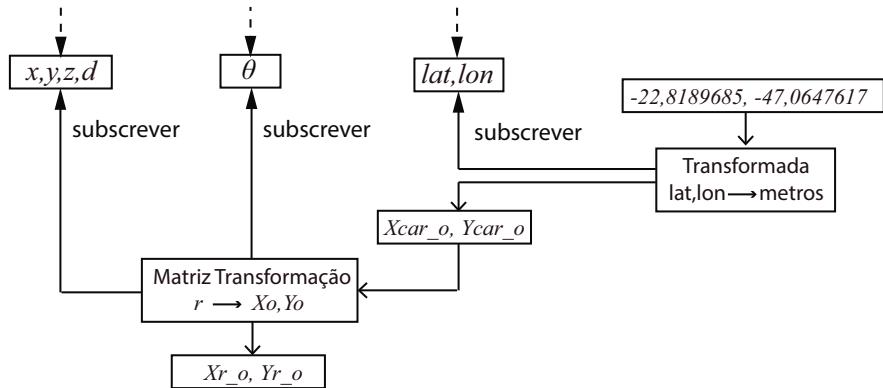


Figura 5.14: Esquema do processo para calcular a posição do sinal de pare com respeito ao mundo X_o, Y_o .

Como se mostra na Figura 5.14, primeiro é feita a transformação das coordenadas latitude, longitude do GPS de baixo custo a coordenadas planas em metros, implementando as Equações 3.10 até a 3.15, tendo como ponto de referência a coordenada $-22,8189685, -47,0647617$. Em seguida, o *script* matriz de transformação se subscreve aos dados publicados pelos *scripts* câmera-laser e bússola, junto com os dados X_{car_o}, Y_{car_o} para obter a posição do sinal de pare com respeito ao mundo X_o, Y_o , sendo o resultado igual a X_r_o, Y_r_o dado em metros, que posteriormente vai ser convertido em um ponto em coordenadas latitude, longitude.

Continuando com o processo descrito no Capítulo 3 o ponto X_r_o, Y_r_o tem que ser transformado a seu equivalente em coordenadas latitude, longitude com o objetivo de utilizar esta informação para se fazer a conexão à base de dados *info_vilma* e assim procurar entre as informações armazenadas qual é o sinal de pare que está sendo detectado.

Isto significa que, depois de encontrar a distância do sinal de pare ao centro da câmera ligada ao modelo do veículo, se faz as transformações necessárias para se encontrar sua posição com respeito ao mundo e também seu equivalente em coordenadas latitude e longitude, para poder localizar e realizar uma busca e identificação a um dos pontos dos sinais de pare que estão armazenados na base de dados *info_vilma*.

Então em um novo *script* se calcula a transformada do ponto X_r_o, Y_r_o em metros a latitude,longitude implementando as Equações 3.2 até a 3.9, tendo como ponto de referência a coordenada $-22,8189685, -47,0647617$, como se mostra na Figura 5.15.

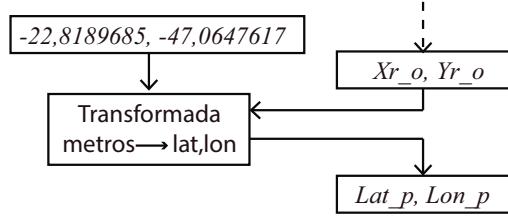


Figura 5.15: Esquema do processo para calcular a transformação do ponto Xr_o, Yr_o dada em metros a latitude, longitude.

Desta transformação tem-se o ponto Lat_p, Lon_p que corresponde à posição equivalente do sinal de pare em coordenadas latitude, longitude. Com esta informação procura-se na base de dados info_vilma, se este ponto corresponde aproximadamente a uma das posições de um dos objetos chave que estão armazenadas. Para conseguir fazer isto, precisa-se criar uma conexão com o servidor de mapas através da biblioteca Psycopg como foi visto na Subseção 3.3.2. No esquema tem-se um novo *script* que vai fazer a conexão com a base de dados info_vilma e através de consultas SQL e operações de transformação espacial PostGIS, procura-se neste caso o ponto do sinal de pare mais perto do ponto aproximado calculado. Este processo pode ser visto na Figura 5.16.

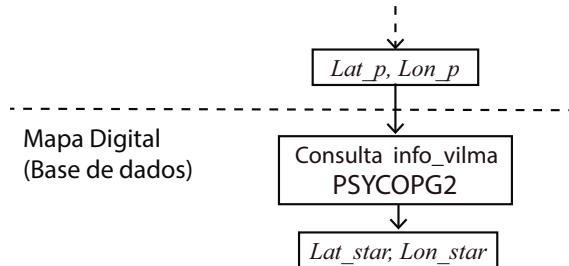


Figura 5.16: Esquema do processo para consultar na base de dados info_vilma a coordenada aproximada que foi calculada Lat_p, Lon_p .

Então os dados da posição do sinal de pare encontrado na base de dados vão ser retornados, e com o novo ponto Lat_star, Lon_star em coordenadas latitude, longitude calcula-se a posição do modelo do veículo.

De acordo com a Subseção 3.3.3, o ponto Lat_star, Lon_star que retorna da base de dados, deve passar pela função de transformação de coordenadas latitude, longitude para coordenadas planas em metros através das Equações 3.10 até a 3.15, tendo como ponto de referência a coordenada -22,8189685, -47,0647617. Com este novo ponto obtido em metros Xr_star, Yr_star calcula-se a posição do modelo do veículo através da matriz de transformação da Equação 3.16, tendo como resultado o ponto final $Xcar_m, Ycar_m$.

Pode-se ver este último processo junto com os outros processos na Figura 5.17, que apresenta o esquema geral completo das medidas do MLR, onde X_{car_m}, Y_{car_m} são as coordenadas em metros da nova posição estimada do modelo do veículo.

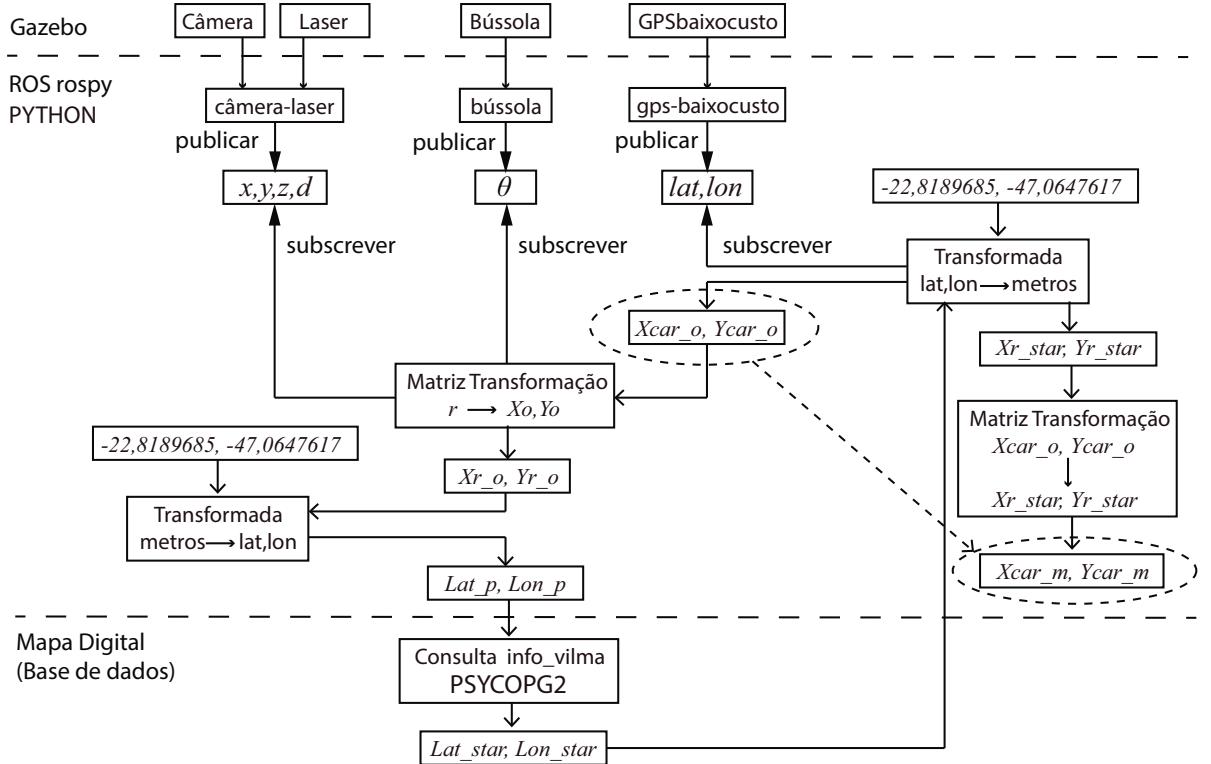


Figura 5.17: Esquema geral das medidas do método de localização referenciada.

Os testes, para verificação do resultado final da implementação, realizados nas medidas da câmera, do laser e da base de dados, representadas na Figura 5.17 que apresenta o processo de captura, de transformação e de envio de dados, obtendo uma nova medida de posição do modelo do veículo baseada na forma de localização em um ambiente através do reconhecimento de objetos chave, são apresentadas na Figura 5.18.

Na Figura 5.18 tem-se 3 arquivos de texto que durante o tempo da simulação, em que modelo do veículo faz o percurso em linha reta de um ponto ao outro, foram armazenados os dados capturados. Em paralelo, tem-se os dados da cada um dos três arquivos, o primeiro corresponde aos dados do GPS ideal *ground truth* que foram armazenados só como medida de referência, o segundo corresponde aos dados capturados pelo MLR, que correspondem à medida X_{car_m}, Y_{car_m} , como foi descrito na Figura 5.17 e o terceiro são os dados do GPS de baixo custo que correspondem à medida X_{car_o}, Y_{car_o} .

Na Figura 5.18 destaca-se que os dados obtidos pelo MLR têm valores mais perto do GPS ideal *ground truth* enquanto o GPS de baixo custo apresenta valores mais distantes e imprecisos.

gps_ideal.txt		Método localização.txt		gps_lowcost.txt	
14.2558504222	36.1741172132	14.1669635773	35.5314102173	13.6988790694	31.3986564872
14.6229758464	37.1051238362	14.9112262726	37.3730659485	14.5328938574	35.8425757994
14.9901957512	38.0361486856	14.9114093781	37.3729858398	14.0282298721	35.5759138881
15.3571444973	38.9666395796	14.9104270935	37.3734130859	16.8839998912	34.7027342552
15.7243571369	39.8975968907	16.15284729	40.1547088623	13.3362319691	34.4317373555
16.0914332263	40.8282960374	16.5128879547	41.0280723572	19.5187816739	39.8643545111
16.4586501991	41.7591548333	16.5129375458	41.028049469	12.5798187559	32.2603415734
16.8257717067	42.6899140869	16.5119113922	41.0284957886	19.3427629648	43.3176441235
17.1930859574	43.6209358187	16.5119075775	41.0284996033	10.4248958652	38.8177288916
17.5601172769	44.551394657	16.5109577179	41.0289115906	10.0082329847	43.0050837479
17.9272833427	45.4819786618	16.5107803345	41.0289878845	8.47390478861	43.2521803798
18.2944967967	46.4127983534	16.5098056793	41.0294151306	16.9888374551	39.5124318443
18.6618126727	47.3436652409	16.5096187592	41.0294952393	16.4037779453	43.8562522843
19.0289210562	48.2741494402	18.7643890381	47.6509246826	12.3027152683	46.8770323016
19.3963303779	49.2051688735	19.6529693604	49.4910736084	12.8844076111	41.4034370769

Figura 5.18: Resultado final da implementação do método de localização referenciada.

5.4.2 Teste 7. Resultados da posição estimada pelo MLR

Os resultados obtidos no Teste 6, pelo MLR, são apresentados na Figura 5.19, onde a linha de cor vermelha representa os dados capturados pelo GPS ideal e os pontos de cor azul representam os dados estimados pelo MLR.

Posteriormente aos resultados obtidos no teste apresentados na Figura 5.19, também é calculado o erro da posição de cada ponto estimado do MLR com respeito ao GPS ideal. Então tem-se na Figura 5.20 os gráficos que mostram o erro calculado para a distância x sendo Ex , o erro para distância y sendo Ey , e o erro da distância total sendo $\sqrt{Ex^2 + Ey^2}$ todos em unidades métricas ao longo do *tempo*.

Resumindo os valores do resultado do erro do MLR, apresenta-se na Tabela 5.2 os dados finais obtidos, tendo um erro máximo de 7m e um erro mínimo de 0,5m aproximadamente, sendo a media de 2m o erro do GPS de baixo custo com um desvio padrão de 1m aproximadamente.

Tabela 5.2: Erros de posição do MLR.

Erro	MLR
Distância máxima	7,6m
Distância mínima	0,5m
Distância média	2,0m
Distância desvio padrão	1,2m

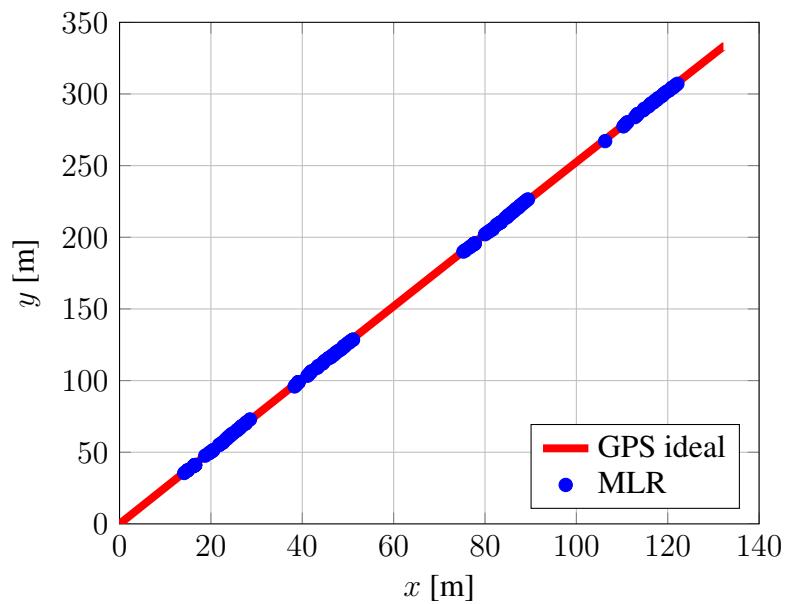


Figura 5.19: Resultados obtidos pelo MLR. GPS ideal vs. MLR.

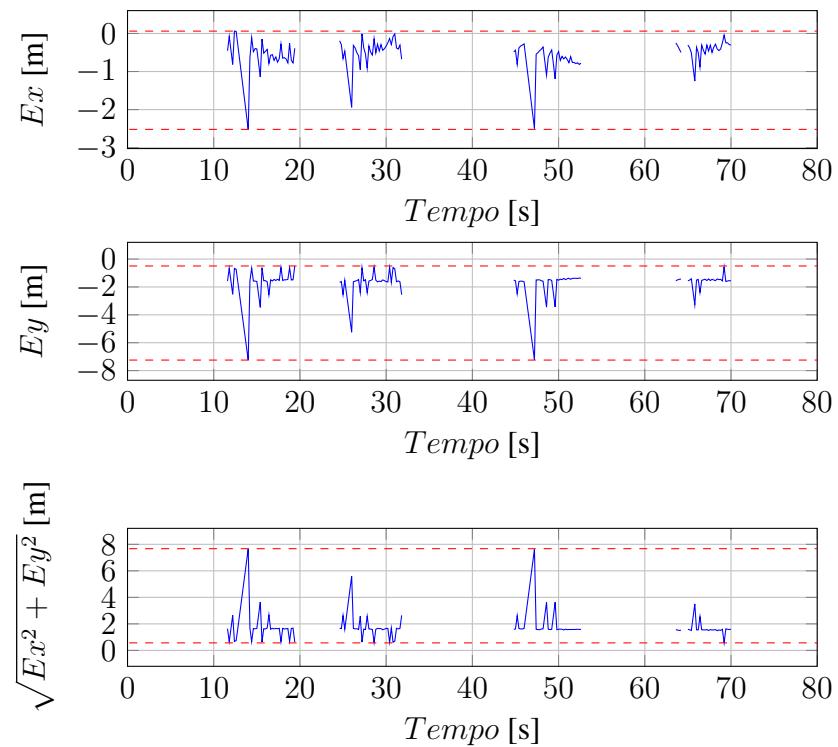


Figura 5.20: Erro da posição de cada ponto do MLR com respeito ao GPS ideal.

5.5 Localização híbrida

Fazendo uso dos dados fornecidos pelos sensores e informações disponíveis na plataforma de localização e simulação (ver Capítulo 4), junto com o modelo matemático do modelo do veículo estudado (ver Apêndice A) e o algoritmo do filtro estendido de Kalman - EKF (ver Apêndice B), é implementado o método de fusão de dados.

O alcance da implementação do EKF é estimar o estado mais provável do sistema de localização do modelo do veículo baseado nas informações medidas pelos sensores e pelo MLR. Então, o EKF faz uso da cinemática do sistema e dos sensores, do ruído aditivo no sistema e das medições, dos estados iniciais, além de todas as medições dos sensores, sem importar seu nível de precisão (NAVARRO, 2009).

Na plataforma de localização e simulação são simulados diferentes sensores e de cada um são aproveitadas suas características mais importantes, as quais vão ajudar a melhorar a medida de estimativa na localização do veículo em um ambiente urbano. Estão ligados ao modelo do veículo, uma câmera, um laser tipo Lidar, uma bússola, um GPS de baixo custo e um GPS ideal, além das informações disponíveis no mapa digital e os objetos chave.

A vantagem da utilização deste método é que mesmo sem ter todas as informações disponíveis no sistema, ele faz a estimativa com a informação que se tem. Isto porque nem sempre vão-se ter disponíveis informações do MLR.

5.5.1 Resultado 1. Fusão de dados EKF = GPS + Bússola + Modelo Matemático

O resultado 1 apresentado na Figura 5.21 se refere à implementação do EKF quando se tem somente as medições do GPS de baixo custo, da bússola e do modelo matemático. A linha da cor vermelha representa os dados capturados pelo GPS ideal e os pontos da cor azul representam os dados calculados pelo filtro EKF.

Posteriormente, a partir do resultado obtido do experimento apresentado na Figura 5.21, é calculado o erro entre a estimativa da posição feito pelo EKF somente com dados do GPS de baixo custo com respeito ao GPS ideal. Então tem-se na Figura 5.22 os gráficos que mostram o erro

calculado para a distância x sendo Ex , o erro para a distância y sendo Ey , e o erro da distância total sendo $\sqrt{Ex^2 + Ey^2}$, todos em unidades métricas ao longo do *tempo*.

5.5.2 Resultado 2. Fusão de todos os dados EKF = MLR + GPS + Bússola + Modelo Matemático

O resultado 2 apresentada na Figura 5.23 se refere a implementação da localização híbrida quando tem-se todas as medições, GPS de baixo custo, MLR, bússola e modelo matemático. A linha de cor vermelha representa os dados capturados pelo GPS ideal e os pontos de cor azul representam os dados calculados da localização híbrida.

Finalmente na Figura 5.24 é apresentado o resultado do erro obtido da estimação da localização híbrida com respeito ao GPS ideal. Então tem-se na Figura 5.24 os gráficos que mostram o erro calculado para a distância x sendo Ex , o erro para a distância y sendo Ey , e o erro da distância total sendo $\sqrt{Ex^2 + Ey^2}$, todos em unidades métricas ao longo do *tempo*.

5.6 Resultados fusionados

Apresenta-se na Tabela 5.3 os valores dos resultados da fusão de dados EKF e da localização híbrida. Na Figura 5.22 apresenta-se o erro da posição estimada com EKF dado pelo GPS de baixo custo, tendo um erro máximo de 5m e um erro mínimo de 0,2m aproximadamente, sendo a média de 2m, o erro do GPS de baixo custo com um desvio padrão de 0,9m aproximadamente. Na Figura 5.24 apresenta-se o erro da posição estimada com EKF daão pelo sistema de visão, tendo um erro máximo de 4m e um erro mínimo de 0,1m aproximadamente, sendo a média de 1,5m o erro do GPS de baixo custo com um desvio padrão de 0,6m aproximadamente.

Tabela 5.3: Erros da posição estimada, fusão de dados e localização híbrida.

Erro	EKF = GPS + Bs + MM	EKF = MLR + GPS + Bs + MM
Distância máxima	4,9m	3,9m
Distância mínima	0,1m	0,09m
Distância média	2,3m	1,5m
Distância desvio padrão	0,97m	0,6m

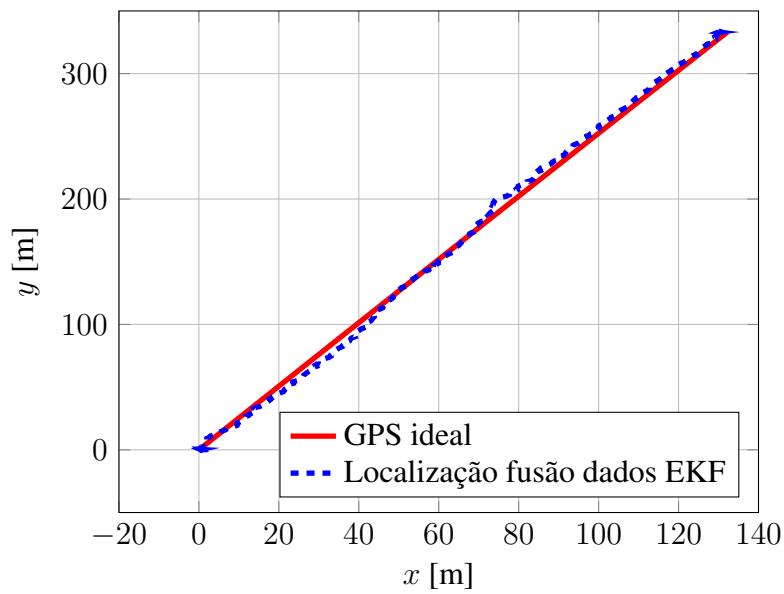


Figura 5.21: Resultados do EKF para o sistema de localização somente com dados do GPS de baixo custo.

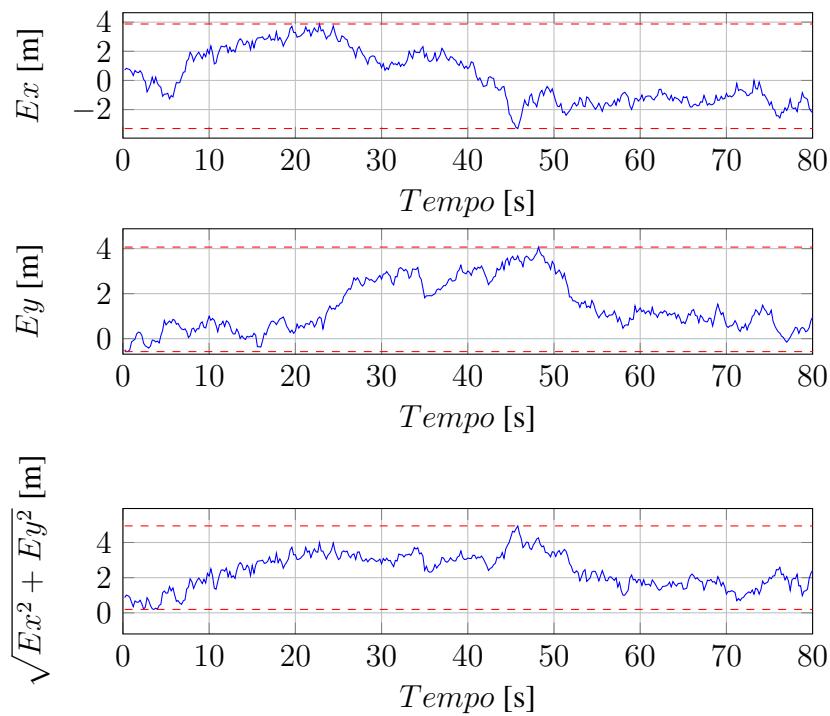


Figura 5.22: Erro da posição estimada do EKF com dados do GPS de baixo custo com respeito ao GPS ideal.

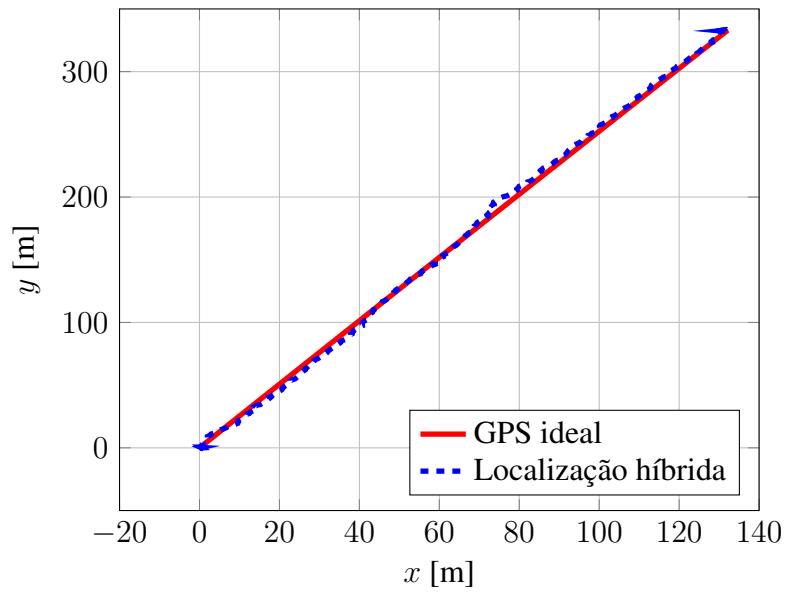


Figura 5.23: Resultados da localização híbrida. EKF = MLR + GPS + Bússola + Modelo Matemático.

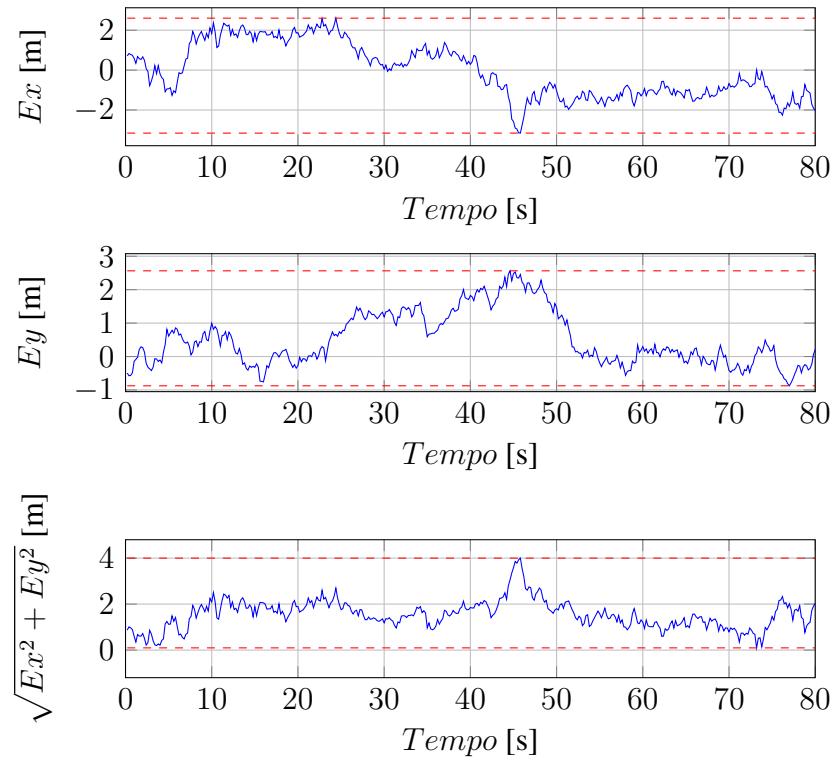


Figura 5.24: Erro da posição estimada da localização híbrida com respeito ao GPS ideal.

6 CONCLUSÕES E TRABALHOS FUTUROS

Este capítulo apresenta as conclusões e perspectivas futuras do presente trabalho. No início, as conclusões baseadas no objeto de estudos, na proposta apresentada e nos resultados obtidos. Por fim, uma breve introdução das direções sugeridas das pesquisas futuras.

6.1 Conclusões

Nos capítulos iniciais, alguns tópicos na área de robótica móvel para veículos autônomos foram apresentados. Os conceitos básicos sobre os sistemas de localização para robôs móveis foram explicados, assim como uma breve descrição dos diferentes métodos de localização foi realizada. Além disso, uma breve introdução aos sistemas de percepção e simuladores para robôs móveis foi feita.

Como principal objetivo, o método de localização referenciada baseada nas informações do mapa digital foi proposto. Esta técnica foi posteriormente implementada e o processo para obter os dados de localização foram documentados, para assim, finalmente serem usados no método de localização híbrida também proposto, realizando basicamente a fusão das informações do modelo matemático do veículo, das informações do MLR e das informações de posicionamento fornecidas pelo sensores de baixo custo.

Entre estas tarefas principais neste trabalho, a construção de uma plataforma de localização e simulação baseada em ferramentas computacionais para a área de robótica móvel foi realizada. Esta plataforma inclui o simulador de ambientes 3D que está ligado a uma ferramenta de desenvolvimento de aplicações robóticas, com o objetivo de simular a cinemática do modelo do veículo terrestre e seus sensores, e o servidor de mapas digitais que permite obter as informações geográficas disponíveis em uma base de dados, que é editável e atualizável. Entre as ferramentas que foram implementadas na plataforma tem-se o ROS, Gazebo e OpenStreetMap.

Este trabalho portanto apresentou uma contribuição aos sistemas de localização baseados em sensores de baixo custo, testado em uma plataforma de localização e simulação robusta, que integra ferramentas de robótica, visualização, base de dados geográficos, modelos 3D e servidores locais.

Ao sistema, implementa-se o método de localização por fusão híbrida que combina as informações de um GPS de baixo custo, orientação, modelo matemático do veículo e o MLR. A principal contribuição desta abordagem, foi o uso de um mapa digital que estava ligado a uma base de dados local com informações de objetos de interesse, que foram identificados posteriormente pelo MLR, e baseado na posição exata de cada objeto de interesse identificado, e, consultado na base de dados e o modelo do veículo, foi feita a correção na sua posição.

Outra importante contribuição deste trabalho foi a implementação de métodos de localização com o filtro de Kalman estendido para sistemas não lineares amplamente implementado em sistemas de localização para veículos móveis, que estima a posição e orientação do veículo. A implementação do filtro combina informações disponíveis, a fim de estabelecer uma posição estimada mais perto da realidade.

A fim de avaliar o MLR, estudado no Capítulo 3, na Tabela 5.3 foram apresentados os resultados da implementação do EKF para o caso quando tem-se somente as informações do GPS de baixo custo e para o caso quando tem-se todas as informações, onde a localização híbrida apresentou um erro 33% menor com respeito à fusão de dados EKF, que utiliza somente informações do GPS de baixo custo (valor calculado a partir das medidas de desvio padrão do EKF com GPS de baixo custo vs. localização híbrida).

A partir dos resultados obtidos, pode-se dizer que o método proposto de localização referenciada com mapas digitais na resolução da localização do modelo do veículo terrestre é eficaz, porque o uso de mapas digitais, visão e GPS quando são combinados com outras informações apresentou um resultado da estimação pelo EKF com um erro médio de 1m. Além disso, o sistema também pode manter uma boa localização na ausência dos sinais do GPS, apresentando um erro médio de 2m.

A contribuição do desenvolvimento teórico e experimental da localização em veículos terrestres neste trabalho, constitui uma aproximação ao objetivo do Laboratório de Mobilidade Autônoma (LMA) de conseguir automatizar a plataforma móvel VILMA (Veículo Inteligente do LMA), da qual faz parte esta pesquisa.

6.2 Trabalhos futuros

Como trabalhos futuros, propõe-se incrementar as informações da base de dados, com outros tipos de dados que podem ser úteis para o funcionamento do veículo autônomo, como por exemplo informação de velocidades limite. Também propõe-se aprofundar no sistema de visão, o qual possa reconhecer outros tipos de informações do ambiente através de processamento de imagens, como pedestres e carros.

Também propõe-se aprofundar no ambiente de simulação, criando mundos mais complexos para posteriormente fazer validações neles, já que neste projeto se trabalhou sómente com objetos como sinais de trânsito. Propõe-se incrementar o ambiente com prédios, ruas, etc, acarretando um ambiente mais realista.

Como as provas foram feitas em sistemas simulados quase que ideais, propõe-se fazer testes simulando terrenos com condições mais difíceis. Também propõe-se realizar os testes em um percurso mais complexo.

Uma das principais restrições encontradas foi o alto gasto computacional no cálculo do MLR, onde os cálculos de detecção do objeto chave e de sua respectiva distância retornavam grandes números de dados, o que dificultava os cálculos, tornando o sistema operacional lento e com falhas. Deste modo o sistema operacional teve que ser reduzido para retornar menos dados, e assim conseguir executar todos os processos em tempo real.

Referências

- ANDRES, B.; OBERMEIER, P.; SABUNCU, O.; SCHAUB, T. e RAJARATNAM, D. Rosoclingo: A ros package for asp-based robot control. **CoRR**, v. abs/1307.7398, 2013.
- BONNIFAIT, P.; BOURON, P.; CRUBILLE, P. e MEIZEL, D. Data fusion of four abs sensors and gps for an enhanced localization of car-like vehicles. In **Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on**, v. 2, pp. 1597–1602 vol.2. 2001.
- BROGGI, A.; BERTOZZI, M.; FASCIOLI, A.; GUARINO, C.; BIANCO, L. e PIAZZI, A. The argo autonomous vehicle's vision and control systems. **International Journal of Intelligent Control and Systems**, pp. 409–441, 1999.
- CAMPBELL, Stefan F. **Steering control of an autonomous ground vehicle with application to the DARPA Urban Challenge**. 2007. Dissertação (Mestrado). Massachusetts Institute of Technology.
- CARDENAS, Miguel. **Localização híbrida para um veículo autônomo em escala usando fusão de sensores**. 2013. Dissertação (Mestrado). Universidade Estadual de Campinas.
- DE OLIVEIRA, M. Carro sem motorista. **Pesquisa FAPESP**, 2013.
- DE PAIVA, E.; AZINHEIRA, J. e BUENO, S. Controle de trajetória para veículos terrestres de exterior. **XVIII Congresso Brasileiro de Automática**, 2010.
- DE REZENDE, E. Controle longitudinal de um veículo autônomo. 2010.
- DICKMANNS, E.D. Dynamic vision-based intelligence. **AI Magazine**, 2004.

DISSANAYAKE, M.W.M.G.; NEWMAN, P.; CLARK, S.; DURRANT-WHYTE, H. e CSORBA, M. A solution to the simultaneous localization and map building (slam) problem. **Robotics and Automation, IEEE Transactions on**, v. 17, n. 3, 229–241, Jun 2001.

DOS SANTOS, Gonçalo. **RobotTeamSim: 3D Visualization of Cooperative Mobile Robot Missions in Gazebo Virtual Environment**. 2013. Dissertação (Mestrado). University of Coimbra.

ETKIN, B. **Dynamics of Atmospheric Flight**. Dover Books on Aeronautical Engineering. Dover Publications, 2012. ISBN 9780486141657.

FIAT. **Manual de uso e manutenção. Fiat Punto**. Fiat Automóveis SA, 2008.

GALÁN, C. e LÓPEZ, R. **Sistemas de información geográfica: aplicaciones prácticas con Idrisi32 al análisis de riesgos naturales y problemáticas medioambientales**. Ra-Ma, 2003. ISBN 9788478975433.

GOEL, P.; ROUMELIOTIS, S. e SUKHATME, G. Robust localization using relative and absolute position estimates. In **Intelligent Robots and Systems, 1999. IROS '99. Proceedings. 1999 IEEE/RSJ International Conference on**, v. 2, pp. 1134–1140 vol.2. 1999.

GORNI, D.; GIANNOTTI, M.; KNOPIK, A.; BRITO, P. e RODRIGUES, M. Open source web gis sistema de informação geográfica de expedições. **XIII Simpósio Brasileiro de Sensoriamento Remoto**, 2007.

GUIVANT, J.E.; MASSON, F.R. e NEBOT, E.M. Simultaneous localization and map building using natural features and absolute information. **Robotics and Autonomous Systems**, v. 40, n. 2–3, 79 – 90, 2002. Intelligent Autonomous Systems - {IAS} -6.

URL: <http://www.sciencedirect.com/science/article/pii/S0921889002002336>

GUNES, M.; JURASCHEK, F.; BLYWIS, B. e GRAFF, C. Monotrac a mobility trace generator based on openstreetmap geo-data. In **Mobile Adhoc and Sensor Systems (MASS), 2010 IEEE 7th International Conference on**, pp. 618–623. Nov 2010.

HENTSCHEL, M. e WAGNER, B. Autonomous robot navigation based on openstreetmap geodata. In **Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on**, pp. 1645–1650. Sept 2010.

HIDE, C. e MOORE, T. Gps and low cost ins integration for positioning in the urban environment. Relatório técnico, Institute of Engineering Surveying and Space Geodesy, University of Nottingham, 2005.

JAMSHIDI, H.; LUKASZEWCZ, T.; KASHI, A.; BERGHUVUD, A.; ZEPERNICK, H. e KHATIBI, S. Fusion of digital map traffic signs and camera-detected signs. In **Signal Processing and Communication Systems (ICSPCS), 2011 5th International Conference on**, pp. 1–7. Dec 2011.

KALMAN, R.E. A new approach to linear filtering and prediction problems. 1960.

URL: <http://www.cs.unc.edu/welch/kalman/media/pdf/Kalman1960.pdf>

KILIMCI, P. e KALIPSIZ, O. Geometric modeling and visualization of moving objects on a digital map. **Geometric Modelling and Imaging**, 2007.

KOENIG, N. e HOWARD, A. Design and use paradigms for gazebo, an open-source multi-robot simulator. In **Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on**, v. 3, pp. 2149–2154 vol.3. Sept 2004.

KONRAD, M.; NUSS, D. e DIETMAYER, K. Localization in digital maps for road course estimation using grid maps. In **Intelligent Vehicles Symposium (IV), 2012 IEEE**, pp. 87–92. June 2012.

LANEURIT, J.; BLANC, C.; CHAPUIS, R. e TRASSOUDAINE, L. Multisensorial data fusion for global vehicle and obstacles absolute positioning. In **Intelligent Vehicles Symposium, 2003. Proceedings. IEEE**, pp. 138–143. June 2003.

LANEURIT, J.; CHAPUIS, R. e CHAUSSE, F. Accurate vehicle positioning on a numerical map. **International Journal of Control, Automation and Systems**, 2005.

LAUE, T.; SPIESS, K. e RÖFER, T. Simrobot – a general physical robot simulator and its application in robocup. 2005.

LEI, C. e WANG, Y. Localization of the autonomous mobile robot based on sensor fusion. In **Intelligent Control. 2003 IEEE International Symposium on**, pp. 822–826. Oct 2003.

LENG, S.S. e GRUYER, D. Merging lateral cameras information with proprioceptive sensors in vehicle location gives centimetric precision. **Proceedings of 18th International Technical Conference on the Enhanced Safety of Vehicles (ESV'2003)**, 2003.

MARÍN, Leonardo. **Navegación de un robot móvil de configuración diferencial basada en fusión sensorial**. 2011. Dissertação (Mestrado). Universidad Politécnica de Valencia.

MATTERN, N.; SCHUBERT, R. e WANIELIK, G. High-accurate vehicle localization using digital maps and coherency images. In **Intelligent Vehicles Symposium (IV), 2010 IEEE**, pp. 462–469. June 2010.

NAVARRO, Danilo Alfonzo. **Contribución a la autolocalización de robots móviles basada en la fusión de información multisensorial**. 2009. Tese (Doutorado). Universidad Politecnica de Valencia.

POSTGIS. Postgis 1.5 manual. Relatório técnico, PostGIS, 2012.

PYO, J.S.; SHIN, D.H. e SUNG, T.K. Development of a map matching method using the multiple hypothesis technique. In **Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE**, pp. 23–27. 2001.

RIFAT, M.; MOUTUSHY, S.; AHMED, S. e FERDOUS, H. Location based information system using openstreetmap. In **Research and Development (SCoReD), 2011 IEEE Student Conference on**, pp. 397–402. Dec 2011.

SHENGBO, Q.; KELIANG, D. e QINGLI, L. An effective gps/dr device and algorithm used in

vehicle positioning system. In **Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE**, v. 1, pp. 632–636 vol.1. Oct 2003.

SICILIANO, B. e KHATIB, O. **Springer Handbook of Robotics**. Gale virtual reference library. Springer, 2008. ISBN 9783540239574.

URL: <http://books.google.com.br/books?id=Xpgi5gSuBxsC>

SIEGWART, R.; NOURBAKHS, I. e SCARAMUZZA, D. **Introduction to Autonomous Mobile Robots**. Intelligent robotics and autonomous agents. MIT Press, 2011. ISBN 9780262015356.

SOARES, Sandro. **Integração GPS/INS utilizando sensores inerciais baseados em sistemas microelectromecânicos (MEMS)**. 2005. Tese (Doutorado). Universidade Federal do Paraná.

STEVENS, B. e LEWIS, F. **Aircraft Control and Simulation**. Wiley, 2003. ISBN 9780471371458.

TAO, Zui. **Vehicle Sensor-based Localization System for Autonomous Vehicle Navigation**. 2012. Dissertação (Mestrado). Université de Technologie de Compiègne (UTC).

TAO, Z.; BONNIFAIT, P.; FREMONT, V. e IBANEZ-GUZMAN, J. Mapping and localization using gps, lane markings and proprioceptive sensors. In **Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on**, pp. 406–412. Nov 2013.

TAYLOR, G. e BLEWITT, G. Virtual differential gps & road reduction filtering by map matching. **ION GPS '99**, 1999.

TEIXEIRA, A. e CHRITOFOLETTI, A. **Sistemas de Información Geográfica: dicionário ilustrado**. São Paulo, 1997.

THRUN, S.; MONTEMERLO, M.; DAHLKAMP, H.; STAVENS, D.; ARON, A.; DIEBEL, J.; FONG, P.; GALE, J.; HALPENNY, M.; HOFFMANN, G.; LAU, K.; OAKLEY, C.; PALATUCCI, M.; PRATT, V. e STANG, P. Stanley: The robot that won the darpa grand challenge. **Journal of**

Field Robotics, 2006.

TRIQUELL, Oriol. **Homes For Robots: A Rapid Prototyping Toolkit for Robotics and Intelligent Environments**. 2011. Dissertação (Mestrado). Technische Universität München.

VENKATRAMAN, K.; AMUTHA, B.; KARTHICK, K. e SANKAR, S. A hybrid method for improving gps accuracy for land vehicle navigation system. In **Emerging Trends in Robotics and Communication Technologies (INTERACT), 2010 International Conference on**, pp. 74–79. Dec 2010.

XIAOYUN, Y. e HENG, H. Gps/dr integrated navigation system based on adaptive robust kalman filtering. In **Microwave and Millimeter Wave Technology, 2008. ICMMT 2008. International Conference on**, v. 4, pp. 1946–1949. April 2008.

YANG, Q. e SUN, J. A location method for autonomous vehicle based on integrated gps/ins. In **Vehicular Electronics and Safety, 2007. ICVES. IEEE International Conference on**, pp. 1–4. Dec 2007.

YUEN, D. e MACDONALD, B. Vision-based localization algorithm based on landmark matching, triangulation, reconstruction, and comparison. **Robotics, IEEE Transactions on**, v. 21, n. 2, 217–226, April 2005.

ZHENG, J.; ZHANG, Z.; CIEPLUCH, B.; WINSTANLEY, A.; MOONEY, P. e JACOB, R. A postgis-based pedestrian way finding module using openstreetmap data. In **Geoinformatics (GE-OINFORMATICS), 2013 21st International Conference on**, pp. 1–5. June 2013.

ÍTALO LOIOLA **Modelo de navegação para robôs móveis baseado em redes de petri coloridas**. 2008. Dissertação (Mestrado). Universidade Federal do Ceará.

APÊNDICE A – Modelo matemático do veículo

Este apêndice apresenta as noções básicas relacionadas com o modelo matemático do veículo. Uma introdução ao sistema de coordenadas é feito, seguida do estudo do modelo cinemático. Por fim, o desenvolvimento do modelo e a equação do modelo simplificado do *Ackerman*, que calcula a variação da posição do robô móvel.

A.1 Introdução

O modelo matemático basicamente descreve o movimento do robô móvel terrestre usando determinados recursos, a fim de localizá-lo em um ambiente de trabalho, onde, além da capacidade de identificar por seus sensores as características de seu ambiente (sensores proprioceptivos e exteroceptivos), é necessário a definição de uma posição e orientação em relação a um sistema de coordenadas global *XY* (SIEGWART *e outros*, 2011).

A.2 Sistema de coordenadas

Considera-se que o robô móvel está em duas dimensões, isto é, os seus movimentos são representados no plano horizontal, nas coordenadas dos eixos *x,y*. Assim define-se para o robô móvel um sistema de coordenadas globais *XY*, em que o eixo *x* representa o movimento longitudinal e o eixo *y* representa o movimento lateral. A origem de coordenadas é no seu centro de gravidade (*x,y*) e da orientação do robô é descrito pelo ângulo θ .

Pode-se então definir um vector de estado que descreve a posição do robô móvel (SIEGWART *e outros*, 2011), como se mostra na Equação A.1,

$$x = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \quad (\text{A.1})$$

A.3 Modelo cinemático

A cinemática é o estudo base de como se comportam os sistemas mecânicos, neste caso, um robô móvel. Este estudo foi realizado a fim de modelar o comportamento do robô para determinadas ações (SIEGWART *e outros*, 2011).

Para o robô móvel é implementado o modelo cinemático de *Ackerman* (CAMPBELL, 2007). Este modelo é representado por uma forma retangular com quatro rodas em cada um dos seus cantos, onde as duas rodas traseiras estão fixadas ao sistema, enquanto as duas rodas dianteiras são atuadas por meio do volante do robô móvel, como é ilustrado na Figura A.1. O modelo é inserido num sistema de coordenadas globais XY , com o centro de gravidade no centro do robô x,y ; a distância entre os eixos das rodas dianteiras e traseiras é L e largura é w , o ângulo de direção é ω , e o ângulo de orientação do robô é θ em relação ao sistema de referência XY .

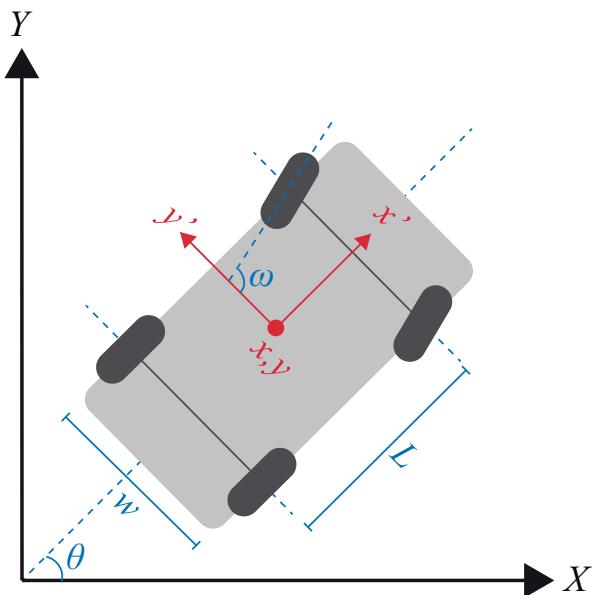


Figura A.1: Modelo de *Ackerman* para no veículo de 4 rodas.

No modelo cinemático de *Ackerman* (CAMPBELL, 2007), são feitas várias simplificações, onde o resultado é conhecido como modelo de bicicleta ou *single-track*. Este modelo é um dos mais utilizados para a análise e desenvolvimento de controladores (CAMPBELL, 2007). Entre as considerações feitas para a simplificação do modelo de *Ackerman* ao modelo de bicicleta são:

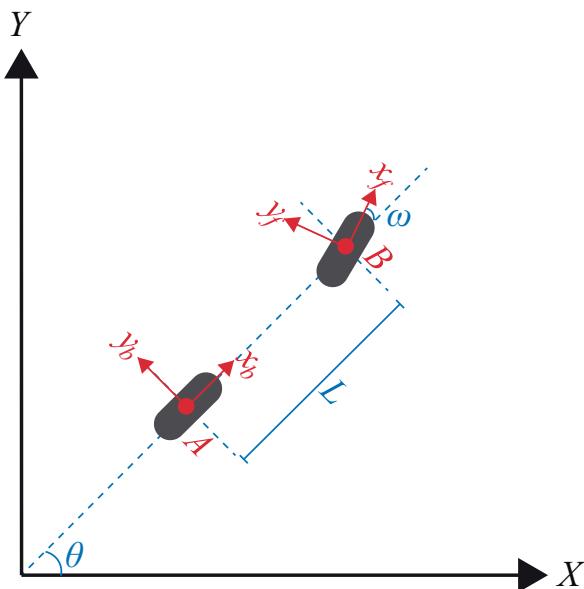


Figura A.2: Modelo simplificado bicicleta.

- (i) As duas rodas dianteiras do veículo são representadas em uma única roda dianteira no ponto B e as duas rodas traseiras são representados em uma única roda traseira no ponto A . A distância entre os eixos das rodas dianteiras e traseiras é L , o ângulo de direção é ω o ângulo e orientação do robô é θ em relação ao sistema de referência XY , como ilustrado na Figura A.2.
- (ii) Este modelo não inclui a dinâmica *roll* e *pitch*. Então, o veículo se movimenta no plano único e tem apenas ângulo de rotação *yaw*.
- (iii) O sistema de coordenadas local do veículo (x,y) coincide com o sistema de coordenadas da roda traseira (x_b, y_b) , como pode ser visto nas Figuras A.1 e A.2, respectivamente.
- (iv) θ é o ângulo formado entre o eixo X e o eixo x_b (roda traseira) e ω é o ângulo formado entre o eixo x_b (roda traseira) e o eixo x_f (roda dianteira).

Com base nas restrições acima o robô móvel é considerada não-holonômico.

A.4 Desenvolvimento do modelo cinemático

A partir do vector de estado da Equação A.1, definido acima, busca-se determinar a variação da posição em cada um dos seus componentes na Equação A.2,

$$\dot{x} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \quad (\text{A.2})$$

Para compreender o comportamento de cada uma das variáveis, é realizado análise das componentes de cada uma das rodas, para uma variação de movimento no tempo, tal como é ilustrado na Figura A.3 para a roda traseira e na Figura A.4 para a roda da frente, onde a roda é movida a partir do ponto *A* para o ponto *B*, respectivamente.

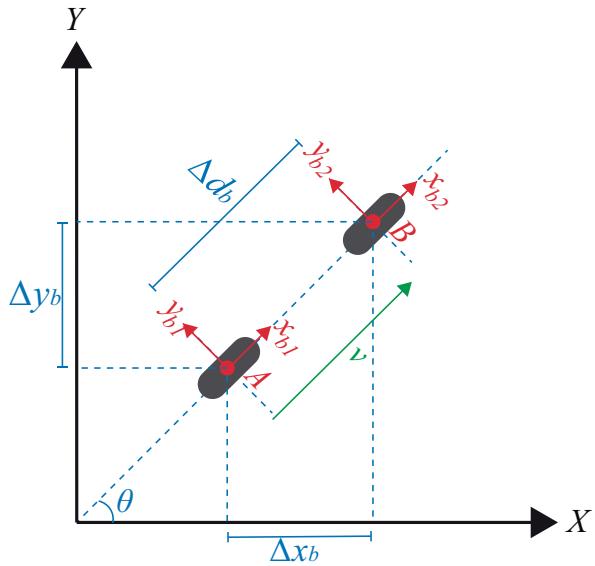


Figura A.3: Modelo cinemático bicicleta. Deslocamento da roda traseira.

No caso da deslocação da roda traseira, Figura A.3, temos que esta num sistema de coordenadas globais *XY*, com (x_b, y_b) como sistema de coordenadas local do centro da roda, o ângulo de orientação do robô é θ em relação à referência de sistema *XY*, então a variação da (x_b, y_b) para se mover de *A* para *B* a um instante de tempo é dado pelas Equações A.3, A.4 e A.5,

$$\dot{x} = \dot{d}_b \cos \theta \quad (\text{A.3})$$

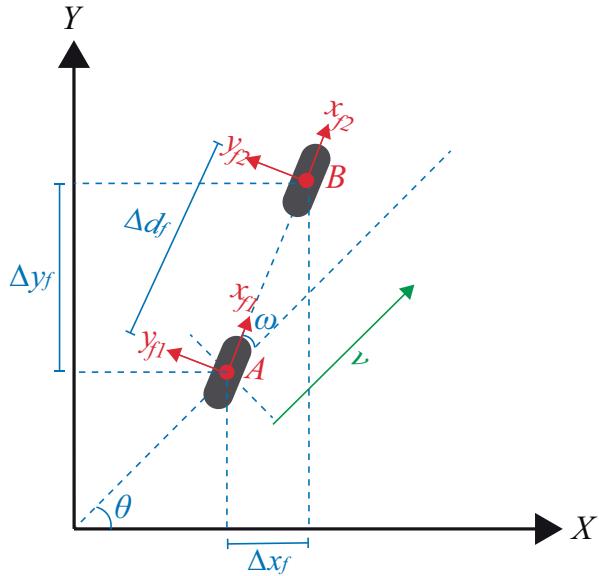


Figura A.4: Modelo cinemático bicicleta. Deslocamento da roda dianteira.

$$\dot{y} = \dot{d}_b \sin \theta \quad (\text{A.4})$$

e,

$$\dot{d}_b = v \quad (\text{A.5})$$

substituindo Equação A.5 nas Equações A.3 e A.4 temos as Equações A.6 e A.7,

$$\dot{x} = v \cos \theta \quad (\text{A.6})$$

$$\dot{y} = v \sin \theta \quad (\text{A.7})$$

onde v é a velocidade com que a roda é movida de A para B .

No caso de deslocamento da roda dianteira, Figura A.4, temos que esta num sistema de coordenadas globais XY , com (x_f, y_f) como sistema de coordenadas local do centro da roda, o ângulo de orientação do robô é θ em relação à referência de sistema XY e o ângulo da direção é ω , então a variação da (x_f, y_f) para se mover de A para B a um instante de tempo é dado pelas Equações A.8 e A.9,

$$x_f = \dot{d}_f \cos(\theta + \omega) \quad (\text{A.8})$$

$$y_f = \dot{d}_f \sin(\theta + \omega) \quad (\text{A.9})$$

igualando as Equações A.8 e A.9 temos a Equação A.10,

$$\dot{y}_f \cos(\theta + \omega) - \dot{x}_f \sin(\theta + \omega) = 0 \quad (\text{A.10})$$

Do modelo de bicicleta, precisamos que esta num sistema de coordenadas globais XY , com (x_f, y_f) como sistema de coordenadas local no centro da roda da frente e (x_b, y_b) como sistema de coordenadas local do centro da roda traseira, o ângulo de orientação do robô é θ em relação ao sistema de referência XY , o ângulo de direção é ω e a distância entre os eixos das rodas dianteiras e traseiras é L , então obtém-se as Equações A.11 e A.12,

$$x_f = x + L \cos \theta \quad (\text{A.11})$$

$$y_f = y + L \sin \theta \quad (\text{A.12})$$

derivando as as Equações A.11 e A.12, temos as Equações A.13 e A.14 temos,

$$\dot{x}_f = \dot{x} - L \dot{\theta} \sin \theta \quad (\text{A.13})$$

$$\dot{y}_f = \dot{y} + L \dot{\theta} \cos \theta \quad (\text{A.14})$$

substituindo na Equação A.10 temos a Equação A.15,

$$\dot{y} + L \dot{\theta} \cos \theta \cos(\theta + \omega) - \dot{x} - L \dot{\theta} \sin \theta \sin(\theta + \omega) = 0 \quad (\text{A.15})$$

simplificando temos a Equação A.16,

$$\dot{y} \cos(\theta + \omega) - \dot{x} \sin(\theta + \omega) - L \dot{\theta} \cos \omega = 0 \quad (\text{A.16})$$

isolando $\dot{\theta}$ da Equação A.16 temos a Equação A.17,

$$\dot{\theta} = \frac{\dot{x} \sin(\theta + \omega) + \dot{y} \cos(\theta + \omega)}{L \cos \omega} \quad (\text{A.17})$$

substituindo as Equações A.6 e A.7 em Equação A.17, temos a Equação A.18,

$$\dot{\theta} = \frac{v \cos \theta \sin(\theta + \omega) + v \sin \theta \cos(\theta + \omega)}{L \cos \omega} \quad (\text{A.18})$$

simplificando a Equação A.18 temos a Equação A.19,

$$\dot{\theta} = \frac{v}{L \cos \omega} [\sin \omega \cos^2 \theta - \sin^2 \theta] \quad (\text{A.19})$$

e temos a Equação A.20,

$$\dot{\theta} = \frac{v}{L} [\tan \omega] \quad (\text{A.20})$$

Finalmente, discretizando o modelo por Euler onde Δt é o tempo de amostragem, então temos das Equações A.6, A.7 e A.20 do modelo simplificado de *Ackerman* ou modelo de bicicleta para calcular a variação da posição do robô móvel, temos a Equação A.21,

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + v_k \Delta t \cos \theta \\ y_k + v_k \Delta t \sin \theta \\ \theta_k + \frac{v_k}{L} \Delta t \tan \omega \end{bmatrix} \quad (\text{A.21})$$

APÊNDICE B – Filtro de Kalman

Este apêndice apresenta alguns conceitos relacionados ao filtro de Kalman. No início, uma introdução ao filtro é feita. Por fim, o desenvolvimento matemático do filtro para estimar o estado mais provável de algum sistema.

B.1 Introdução

O filtro de Kalman (KF, do inglês *Kalman filter*) é um método matemático da área de estatística criado por Rudolf Kalman em 1960 (KALMAN, 1960).

O KF é um algoritmo que processa dados recursivamente, isto é, que para calcular o estado atual, se baseia em estimações e medições de um estado anterior. Seu objetivo é estimar o estado mais provável de um sistema dinâmico e fornecer uma medida de confiança da estimativa, em relação ao verdadeiro estado. Geralmente, as medições do sistema dinâmico estão contaminadas com ruído Gaussiano aditivo que geram incertezas em relação aos dados reais. Finalmente, o KF baseia-se na predição e correção, ou seja, ele primeiro calcula a incerteza de um estado através da predição baseada na dinâmica do sistema, e em seguida corrige esta predição usando medidas que entram no sistema (NAVARRO, 2009).

B.2 Filtro de Kalman Estendido

Mas muitos sistemas dinâmicos não são completamente lineares. A solução para este problema é abordar técnicas de linearização que simulam a nível local, o comportamento linear do sistema em um determinado ponto. Este processo é chamado de Filtro de Kalman Estendido (EKF, do inglês *extended Kalman filter*). Para melhores resultados, o EKF liga ao processo de linearização o último estado estimado a fim de obter uma melhor aproximação do sistema, isto é porque, se o ponto de linearização está longe do estado real do sistema os resultados podem ser inadequados (NAVARRO, 2009).

No caso dos sistemas não lineares, o sistema e o modelo de observação podem ser modelados

por uma equação não linear (NAVARRO, 2009), como se apresenta nas Equações B.1 e B.2,

$$\mathbf{x}_k = f(\mathbf{x}_{k+1}, \mathbf{u}_k) + \mathbf{w}_k \quad (\text{B.1})$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \quad (\text{B.2})$$

onde $f(\mathbf{x}_{k+1})$ é a função não-linear do sistema que relaciona o estado anterior \mathbf{x}_{k+1} , a excitação atual \mathbf{u}_k e o ruído aditivo ao sistema \mathbf{w}_k , com o estado atual \mathbf{x}_k . E $h\mathbf{x}_k$ é a função não-linear que relaciona ao estado com a observação atual ou medição, e \mathbf{v}_k é o ruído aditivo ao sistema de medição.

No caso dos sistemas não lineares, o algoritmo EKF é igual ao do KF linear, primeiro a fase de inicialização e posterior a esta, trabalham de uma forma cíclica as fases de predição e de correção (NAVARRO, 2009).

A inicialização, o EKF deve especificar o estimado do estado inicial do sistema $\hat{\mathbf{x}}_0^+$ e a incerteza \mathbf{P}_0^+ .

B.2.1 Predição

Na fase de predição, o EKF propaga o estado e a covariância para cada passo do tempo k (NAVARRO, 2009), como se apresenta nas Equações B.3 e B.4,

$$\hat{\mathbf{x}}_k^- = f(\hat{\mathbf{x}}_{k-1}^-, \mathbf{u}_k) \quad (\text{B.3})$$

$$\mathbf{P}_k^- = \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}^T + \mathbf{Q}_k \quad (\text{B.4})$$

onde \mathbf{F} é a matriz Jacobiana que contém as derivadas parciais do sistema não lineal com respeito ao estado x , \mathbf{Q} é a matriz de covariância do ruído aditivo e \mathbf{P} é a incerteza da covariância.

B.2.2 Correção

Na fase de correção, a estimativa anterior do estado é corrigida quando temos dados disponíveis de z_k (NAVARRO, 2009). Nesta fase, vão-se utilizar as Equações B.5, B.6 e B.7,

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k)^{-1} \quad (\text{B.5})$$

$$\hat{x}_k^+ = \hat{x}_k^- + \mathbf{K}_k (\mathbf{Z}_k - h(\hat{x}_k^-)) \quad (\text{B.6})$$

$$\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^{-1} \mathbf{P}_k^- \quad (\text{B.7})$$

onde z_k são as medidas, \mathbf{H}_k é a matriz das medições da função é \mathbf{K}_k e o ganho do filtro de Kalman e \mathbf{R}_k são as medições de ruído de covariância.

B.2.3 Resumo

Na Figura B.1 se apresenta em resumo o processo feito pelo EKF,

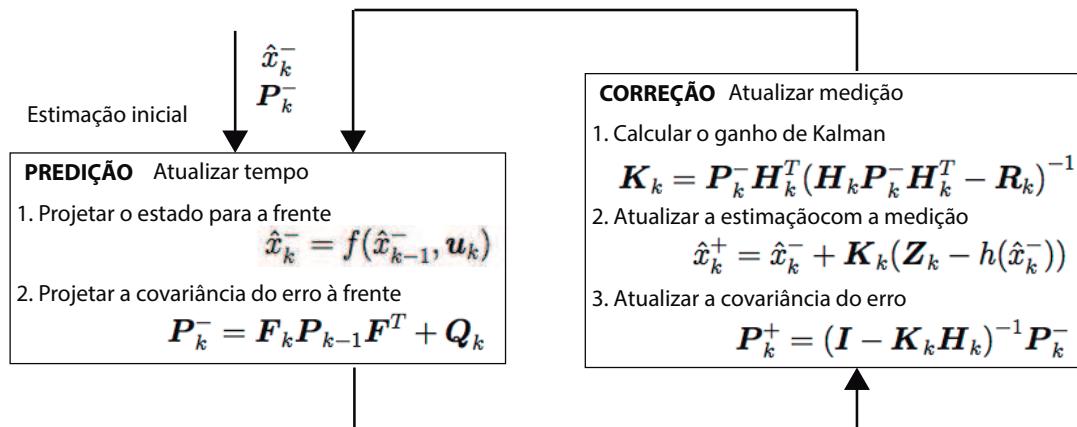


Figura B.1: Algoritmo do EKF.