



Pedro Miguel  
Godinho Salvado

Reconstrução Dinâmica de Mapa Local para o  
*AtlasCar*





**Pedro Miguel  
Godinho Salvado**

**Reconstrução Dinâmica de Mapa Local para o  
*AtlasCar***

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestrado em Engenharia Mecânica, realizada sob orientação científica de Vitor Manuel Ferreira dos Santos, Professor Associado do Departamento de Engenharia Mecânica da Universidade de Aveiro.



**O júri / The jury**

Presidente / President

**Prof. Doutor Jorge Augusto Fernandes Ferreira**  
Professor Auxiliar da Universidade de Aveiro

Vogais / Committee

**Prof. Doutor Vitor Manuel Ferreira dos Santos**  
Professor Associado da Universidade de Aveiro (orientador)

**Prof. Doutor Paulo Miguel de Jesus Dias**  
Professor Auxiliar da Universidade do Aveiro



## **Agradecimentos / Acknowledgements**

Gostava de deixar aqui o meus mais sinceros agradecimentos a todas a pessoas que permitiram, direta ou indiretamente, a realização deste trabalho:

Ao professor Vitor Santos pelo acompanhamento, disponibilidade, orientação e motivação que transmitiu.

Um especial agradecimento às pessoas presentes no Laboratório de Automação e Robótica, Jorge Almeida e Miguel Oliveira pelo apoio e troca de opiniões essenciais ao desenvolvimento deste projeto.

Aos meus colegas que partilharam o Laboratório de Automação e Robótica ao longo destes meses, pela boa disposição e pelos momentos de descontração proporcionados.

Aos meus pais e irmã, por tudo.





## Palavras-chave

*AtlasCar*; sensor inercial;GPS; sensor *Sharp*; hodometria; ROS; nuvem de pontos; varrimento *laser*;

## Resumo

Este trabalho descreve um método para a reconstrução dinâmica de um mapa local, na vizinhança de um veículo em movimento (*AtlasCar*) com recurso a um *laser* 2D. O mapa local consiste numa nuvem de pontos 3D que é obtida através da acumulação de diversos varrimentos *laser* à medida que o *AtlasCar* navega. Para que a nuvem de pontos permita extrair informação útil do mapa, é necessário que a sua construção descreva com elevado rigor a geometria do espaço envolvente. Para tal, é essencial o desenvolvimento de um módulo de software, que forneça ao sistema o estado do veículo enquanto este se move.

Este trabalho encontra-se dividido em duas partes principais: melhoria do cálculo da posição do *AtlasCar* e a monitorização do seu estado em relação ao espaço envolvente. O processo de melhoria do cálculo da posição é feito através da introdução de um sensor inercial, que permite a medição do valor de orientação do carro. A monitorização do estado do veículo relativamente ao ambiente circundante é feito através da medição da altura do *chassis* do *AtlasCar* em quatro pontos distintos, permitindo assim a determinação da sua inclinação e a altura média em relação à estrada.

O método que se apresenta foi testado e as nuvens de pontos obtidas, mostraram ser bastante fidedignas.



**Keywords**

*AtlasCar*; IMU;GPS; sensor *Sharp*; odometry; ROS; Pointcloud; laser scan

**Abstract**

This work presents a method for the construction of a dynamic local map using a 2D laser. A local map is a 3D pointcloud that is obtained by the accumulation of several laser scans, while *AtlasCar* is moving. In order to get a pointcloud that allows the extraction of useful information from the map, it is necessary that the construction describes with a high level of precision the geometry of the surroundings of *AtlasCar*. To get a high precision pointcloud it is necessary the development of a software module that is responsible to provide the vehicle's state to the system.

This work is divided into two main parts: improvement of the calculation of the *AtlasCar* position and the development of a system that provides the information of the *AtlasCar* state relatively to the space around. The improvement of the position calculation was achieved using data from a inercial measurement unit, that measures the orientation of the car relatively to a local magnetic field. The state of the car relatively to the surroundings is obtained through measurements of the chassis height in four different points, allowing the calculation of the inclination and the mean height to the ground.

The developed method was tested and the pointclouds obtained were pretty reliable.



# Conteúdo

<b>1</b>	<b>Introdução</b>	<b>1</b>
1.1	Projeto ATLAS . . . . .	1
1.2	Apresentação do Problema . . . . .	2
1.3	Objetivos . . . . .	3
1.4	Revisão Bibliográfica . . . . .	3
1.5	<i>Robot Operating System</i> . . . . .	6
<b>2</b>	<b>Localização do <i>AtlasCar</i> e seu estado</b>	<b>9</b>
2.1	Sistemas de Coordenadas do <i>AtlasCar</i> . . . . .	9
2.2	Filtro de <i>Kalman</i> e Filtro de <i>Kalman</i> Extendido . . . . .	10
2.3	Hodometria . . . . .	12
2.3.1	Modelo Utilizado . . . . .	13
2.3.2	Aplicação do filtro de <i>Kalman</i> extendido e estimativa da posição . . . . .	15
2.3.2.1	Definição do modelo físico . . . . .	15
2.4	Determinação da orientação do <i>AtlasCar</i> . . . . .	17
2.4.1	Sensor Inercial - IMU . . . . .	18
2.4.2	Limitações da IMU e Sensores Complementares . . . . .	18
2.5	<i>Egomotion</i> . . . . .	21
2.5.1	Determinação da posição do <i>AtlasCar</i> . . . . .	21
2.5.2	Determinação das inclinações . . . . .	22
<b>3</b>	<b>Construção do Mapa Local</b>	<b>23</b>
3.1	<i>Laser Hokuyo</i> . . . . .	23
<b>4</b>	<b>Resultados Obtidos</b>	<b>27</b>
4.1	Correção de dados dos sensores <i>Sharp GP2D12</i> . . . . .	27
4.2	Melhoria no cálculo da posição do <i>AtlasCar</i> . . . . .	33
4.3	Resultados da Construção do Mapa Local . . . . .	35
<b>5</b>	<b>Conclusão e Trabalho Futuro</b>	<b>43</b>



# Lista de Tabelas





# Lista de Figuras

1.1	Robôs Atlas para condução autônoma. . . . .	1
1.2	<i>AtlasCar</i> . . . . .	2
2.1	Sistema de referenciais presentes no <i>AtlasCar</i> . . . . .	10
2.2	<i>Encoder</i> na roda esquerda do eixo traseiro. . . . .	13
2.3	Esquema com o cinemática de <i>Ackermann</i> . . . . .	14
2.4	Desvio da velocidade em relação ao valor médio . . . . .	17
2.5	Desvio da orientação em relação ao valor médio . . . . .	17
2.6	Sensor Inercial <i>Mti Xsens</i> . . . . .	18
2.7	Esquema com a posição dos sensores <i>Sharp</i> . . . . .	19
2.8	Esquema com os ângulos que o carro faz com a estrada, devido à suspensão. . . . .	20
2.9	Sensor <i>Sharp GP2D12</i> . . . . .	20
2.10	Esquema exemplificativo das diferentes inclinações presentes num carro. . . . .	21
2.11	Árvore de transformações. . . . .	22
3.1	Sensor <i>Laser Hokuyo</i> . . . . .	23
3.2	Representação de varrimento <i>laser</i> . . . . .	24
3.3	Exemplo de Reconstrução com <i>Kinect</i> . . . . .	25
3.4	Grafo de transformações utilizado por este módulo. . . . .	25
4.1	Dados de distância ao solo - <i>AtlasCar</i> parado. . . . .	29
4.2	Dados de distância ao solo - <i>AtlasCar</i> a curvar. . . . .	30
4.3	Dados de distância ao solo - <i>AtlasCar</i> a acelerar. . . . .	31
4.4	Dados de distância ao solo - Atraso do sinal filtrado em relação à medições. . . . .	32
4.5	Percurso reconstruído - Alboi . . . . .	33
4.6	Percurso reconstruído - Bairro do Liceu . . . . .	34
4.7	Representação sem informação dinâmica do <i>AtlasCar</i> . . . . .	35
4.8	Representação com a dinâmica do carro em relação à estrada. . . . .	36
4.9	Representação com a inclinação fornecida pelo sensor inercial. . . . .	36
4.10	Dados da inclinação do <i>AtlasCar</i> . . . . .	37
4.11	Escala de cores da nuvem de pontos. . . . .	38
4.12	Reconstrução com <i>AtlasCar</i> a curvar - Sem transformação . . . . .	39
4.13	Reconstrução com <i>AtlasCar</i> a travar - Com transformação . . . . .	39
4.14	Reconstrução com <i>AtlasCar</i> a curvar - Sem transformação . . . . .	40
4.15	Reconstrução com <i>AtlasCar</i> a curvar - Com transformação . . . . .	40
4.16	Representação da nuvem de pontos em análise. . . . .	41
4.17	Representação da percentagem de pontos de cada intervalo . . . . .	41



# Capítulo 1

## Introdução

### 1.1 Projeto ATLAS

O projeto *ATLAS* foi criado pelo grupo de Automação e Robótica do Departamento de Engenharia Mecânica da Universidade de Aveiro, com o principal objetivo de desenvolver novas soluções na área da condução autónoma [Atlas, 2012].

Este projeto teve início com o desenvolvimento de pequenos robôs (figura 1.1) para participação em competições de condução autónoma.



Figura 1.1: Robôs Atlas para condução autónoma.

Com o sucesso destes pequenos modelos nas competições em que participavam, a equipa *ATLAS* decidiu subir a fasquia e apostar noutras frentes. Foi a partir desse momento que, com o conhecimento adquirido, se decidiu começar a trabalhar no *AtlasCar* (Figura 1.2).

O desenvolvimento de sistemas avançados de assistência à condução e de novos métodos que asseguram uma condução mais segura, são dois dos principais objetivos a alcançar.

O *AtlasCar* é um protótipo onde são desenvolvidos e testados diversos sistemas rumo a este objetivo. Este veículo encontra-se equipado por uma série de componentes estranhos



Figura 1.2: *AtlasCar*.

à maioria dos carros dos quais se podem destacar a existência de dois computadores, câmaras, sensores *laser* 2D e 3D, entre outros. Graças a estes sensores, em conjunto com os atuadores e controladores, o *AtlasCar* tem a capacidade de detetar obstáculos e de saber onde se encontra relativamente a eles, tornando-o capaz de se movimentar de forma autónoma.

## 1.2 Apresentação do Problema

A necessidade de saber o que existe e como se encontra o ambiente que rodeia um carro é vital para que a condução assistida, ou autónoma, ocorra com o mínimo de risco. Por essa razão, a maioria das marcas de automóveis e de certos projetos de investigação têm instalado em automóveis sistemas que permitem uma condução cada vez mais controlada e segura.

No projeto em que este trabalho se encontra inserido, possuir a informação do estado do ambiente em torno de um robô é vital para o seu funcionamento.

A informação que se deseja obter através da monitorização do ambiente é bastante variada:

- deteção da geometria do espaço envolvente;
- deteção de obstáculos ou pessoas na trajetória do carro;
- deteção de lugares de estacionamento;
- identificação da estrada;
- planeamento de trajetórias;

- etc

O trabalho a desenvolver vai no sentido de permitir que seja possível ao *AtlasCar* ter acesso a essa informação numa representação útil para as tarefas enumeradas.

A obtenção desse tipo de representação será feita através da construção de uma nuvem de pontos do espaço por onde o carro circula. Esta nuvem será obtida através da acumulação de varrimentos *laser*, à medida que o carro se desloca. Para que a reconstrução do mapa local seja possível e realizada corretamente, é necessário que a posição e inclinação do carro seja determinada enquanto o *AtlasCar* se encontra em movimento.

Um dos pontos chave para uma boa reconstrução é a correta localização e caracterização do veículo. O *AtlasCar* obtém a informação sobre a sua localização através da integração da informação recolhida por um *encoder* colocado na roda esquerda traseira e pela orientação das rodas dianteiras. Apesar de este método permitir a obtenção de resultados razoáveis no que diz respeito a curtas distâncias, erros no posicionamento e orientação do veículo crescem para percursos maiores. Dessa forma, a necessidade de obter dados de outros tipos que permitam uma estimativa com um maior grau de precisão, leva à utilização de outros sensores.

### 1.3 Objetivos

Este trabalho tem como objetivo o desenvolvimento de um módulo de software que permita a reconstrução de um mapa local, com recurso ao *laser* 2D Hokuyo UTM-30LX e a informação da posição e orientação disponibilizada pelo *Egomotion* do *AtlasCar*. Este módulo será responsável pela acumulação de pontos recolhidos de forma a adquirir-se uma nuvem de pontos a três dimensões do espaço que rodeia o carro.

Para determinar a posição e orientação do carro, é necessário a criação de um pacote de software que permita fazer a fusão da informação obtida através de hodometria e dos dados de novos sensores a instalar no *AtlasCar*, nomeadamente sensor inercial e optoeletrónico.

Visto que o projeto *ATLAS* migrou para o ambiente de desenvolvimento *Robot Operation System(ROS)*, é necessária a familiarização com o seu funcionamento.

### 1.4 Revisão Bibliográfica

A utilização de robôs autónomos tem vindo a ganhar importância e irá continuar no futuro. Estes robôs têm um grande potencial de utilização em áreas que vão desde robôs para utilização doméstica até a automóveis para condução autónoma. No entanto, a sua grande limitação prende-se com o seu funcionamento num mundo em constante alteração. Para que estes robôs possam funcionar sem limitações, será necessário dotá-los da capacidade de obter informação do espaço envolvente e de forma precisa e rápida. Este tipo de informação é vital para que os robôs se possam adaptar ao ambiente envolvente evitando colisões com possíveis obstáculos que possam surgir no seu caminho.

A extração da informação do espaço que rodeia um robô é geralmente feita de duas formas, por visão ou pela utilização de *lasers* [Bailey, 2002]. A utilização de visão para a determinação a 3 dimensões do espaço é bastante exigente computacionalmente, dificultando o seu uso em tempo real, optando-se pela utilização do *laser* para fornecer essa informação.

Em [Holz et al., 2010], foi desenvolvido um sistema de percepção e navegação para ambientes domésticos. O seu principal objetivo foi o desenvolvimento de um sistema autónomo que pudesse navegar pela casa das pessoas (ambiente doméstico) evitando colisões. O sistema de percepção utilizado consiste num *laser* 2D que vai fazendo varrimentos na vertical, i.e., utiliza um mecanismo que varia continuamente a inclinação do laser em relação ao chão. Este sistema permite ao robô fazer a reconstrução a três dimensões do espaço por onde circula, permitindo detetar e evitar qualquer obstáculo com que possa colidir.

Os trabalhos [Ryde and Hu, 2007] e [Siadat et al., 1997], à semelhança do anterior têm como objetivo a percepção do espaço, variando o método de obtenção dessa informação. O primeiro utiliza dois *lasers* perpendiculares entre si que estão acoplados a um mecanismo rotacional, o segundo tem instalado um *laser* no topo de um robô, apontado para o chão imediatamente à sua frente. A informação recolhida pelo sensor é *guardada* à medida que o robô navega.

O projeto de maior interesse e que de certa forma está relacionado com o este trabalho é o vencedor do *DARPA Grand Challenge 2005* dada a sua dimensão e resultados obtidos. O *DARPA Grand Challenge* foi uma competição criada pelo *USA Defense Advanced Research Project Agency* de forma testar a capacidade de navegação de veículos não tripulados em terrenos não regulares e em ambientes fora de estrada.[Buehler et al., 2007] O vencedor da edição 2005 foi o robô *Stanley*, projeto desenvolvido pela universidade de *Stanford*. Este robô consiste num veículo (*Volkswagen Touareg R5*) constituído por unidades sensoriais, de processamento e atuação. Todos estes dispositivos tornam possível a navegação autónoma. À semelhança do *AtlasCar* e dos projetos descritos atrás, este veículo necessita de ter a informação sobre o estado do ambiente que o rodeia. No entanto, para além desta informação, necessita também de saber a sua localização e orientação em ambientes *outdoor*. A percepção no *Stanley* é feita através da utilização de diversos tipos de sensores. No seu tejadilho encontram-se instalados cinco *lasers* 2D da *SICK*, estando cada *laser* com uma inclinação diferente em relação à estrada de forma a obter o estado da mesma a diferentes distâncias. Outros sensores, como *RADAR* e câmaras permitem completar a informação sobre o estado do ambiente. O conjunto da percepção com a informação do estado do carro é que permite ao *Stanley* identificar a estrada e possíveis obstáculos, corrigindo a sua trajetória sempre que necessário [Buehler et al., 2007].

Para um robô móvel, determinar a sua localização no espaço é fundamental em navegação autónoma. Um robô que consiga acompanhar a evolução da sua posição enquanto navega é essencial para tarefas de planeamento de trajetórias, de evitar colisões com obstáculos, de exploração, entre outras. De seguida serão apresentados três métodos geralmente utilizados para determinar localização [Bailey, 2002]:

- *Dead Reckoning* - é o método mais básico de localização. Este método estima a posição do veículo por integração do movimento do robô, com informação de sensores inerciais, informação de *encoders* colocados nas rodas ou por hometria visual. A limitação deste método prende-se com o facto de a cada integração o valor estimado conter um erro associado que é acumulado ao longo de todo o processo, perdendo-se a precisão do cálculo.
- *a priori map localization* - consiste na utilização de um mapa conhecido *a priori*. Este método permite estimar a localização de um robô comparando a informação recolhida com a já existente. Este método, apesar de permitir estimar a posição

com elevada fiabilidade, tem o inconveniente de nem sempre a informação sobre o espaço envolvente estar disponível, não sendo possível a utilização deste método nessas circunstâncias

- *Simultaneous Localization and Mapping*(SLAM) - este método é responsável por guardar a informação sobre pontos de referência no espaço enquanto robô se move. Este robô depois usa essa informação para estimar a sua posição em relação aos pontos de referência. Com o desenvolvimento deste método tornou-se possível que robôs pudessem navegar em ambientes desconhecidos, identificando-os à medida que navegam.

Um dos pontos mais importantes na obtenção de um mapa prende-se com a construção do espaço envolvente de forma coerente e precisa. Cada vez mais este aspeto torna-se um desafio, pois os ambientes de navegação passam de ambientes controlados e regulares a ambiente irregulares em constante alteração.

O princípio fundamental para um boa reconstrução (*mapping*) é a caracterização do robô, i.e., o cálculo correto da sua posição e orientação relativamente ao ambiente que o rodeia. Diversos métodos têm vindo a ser desenvolvidos para melhorar o cálculo da posição em robôs. Em [Weinstein and Moore, 2010], o método utilizado para estimar a posição é feito recorrendo à informação que é possível retirar das rodas, através de um *encoder* e da orientação das rodas dianteiras (típico *Dead Reckoning*). No entanto para compensar erros durante a integração da posição, são utilizados outros sensores em conjunto, GPS e sensor inercial. O GPS tem como objetivo manter a orientação do veículo correta e o sensor inercial permite medições em sistemas com grande dinâmica. Outro projeto, [Nourani-Vatani et al., 2009], apresenta um método de cálculo através de hodometria visual. O robô utilizado é um *Toyota Prado*, sendo a hodometria realizada através da utilização de uma única câmara. A hodometria tipicamente permite estimar a posição no plano  $xy$ , no entanto, neste projeto os autores afirmam ter calculado também a variação do carro segundo a direção  $z$  com auxílio de um sensor inercial. Para o cálculo segundo  $z$ , utilizaram o valor de *pitch* enviado diretamente do sensor inercial. Não ficou esclarecido porém como é que foi feita a distinção entre o *pitch* criado pelo carro em situações de aceleração/travagem devido à suspensão, do *pitch* do carro a andar a velocidade constante em subidas/descidas. Voltando ao projeto *Stanley*[Buehler et al., 2007], só a percepção, como já foi referido, não é suficiente para tornar o veículo autónomo. Outro tipo de sensores são necessários para manter o veículo na rota desejada. O GPS permite determinar a localização global do veículo, os sensores inerciais permitem determinar a orientação com que o veículo se encontra e o *encoder* é utilizado uma vez mais para calcular o deslocamento.

Muita informação encontra-se disponível sobre projetos que aplicam diferentes métodos que permitem melhorar o cálculo da posição de robôs autónomos, no entanto no que toca à determinação da inclinação de robôs tipo carro relativamente ao ambiente, essa informação é escassa ou pouco esclarecida pelos projetos de maior relevo. Alguns projetos, [Sappa et al., 2007], [Labayrade and Aubert, 2003] e [Coulombe and L., 2002] utilizam a visão estereo para determinar os valores de *pitch* e *roll* do veículo em relação à estrada. Em todos estes projetos o desafio é o de separar o que é estrada do resto do ambiente. Estes projetos assumem que a estrada é a geometria predominante na nuvem 3D. Através dessa geometria, que varia caso o carro esteja em curva ou a acelerar/travar, determinam-se os valores dos ângulos que o carro faz com a estrada.

Uma alternativa que começa a ser adotada por projetos de maior relevo, onde a posição e orientação é obtida e processada por um único sistema, é o *Applanix POS LV*. [Joe et al., 2001] O *Applanix POS LV* é um sistema desenvolvido pela empresa *Applanix - A Trimble Company*, que permite a obtenção de informação da posição e da dinâmica de um veículo de forma bastante precisa. Este sistema é constituído por: sensor inercial; recetores GPS; sistema de processamento e software de pós-processamento, podendo englobar outros tipos de sensores dependendo da aplicação em causa. O *software* de navegação por integração inercial é o elemento principal deste sistema. Este software é responsável por calcular a posição e orientação por integração da informação recolhida pela unidade inercial (*IMU*), utilizando-se o GPS para correção da posição. O sistema de GPS instalado é constituído por dois recetores que permitem precisões no cálculo da posição na ordem dos decímetros. A utilização conjunta do GPS e do sensor inercial é possível devido à aplicação de um filtro de *Kalman*, que permite determinar a diferença entre a posição calculada por estes dois sensores e assim estimar os erros no cálculo do *IMU*. O cálculo destes erros permite que o sensor inercial seja calibrado regularmente e em tempo real [Joe et al., 2001].

Para aplicações em robôs tipo carro, o sistema é ainda constituído por um dispositivo de medição de deslocamentos (*DMI*), que permite melhorar o cálculo da posição e orientação em períodos de ausência de sinal GPS.

## 1.5 *Robot Operating System*

A descrição deste ambiente de desenvolvimento foi feita com base em [Quigley et al., 2009]. O *ROS* é um ambiente de desenvolvimento especialmente ligado a aplicações em robótica. Este ambiente foi especialmente desenvolvido para projetos de grande dimensão pois permite reduzir bastante a sua complexidade, devido à sua arquitetura modular. Este tipo de arquitetura permite reduzir um grande projeto em pequenos módulos cada um com uma aplicação específica, facilitando o *debugging* e a sua compreensão. Para além de reduzir a complexidade permite a reutilização desses módulos noutras aplicações.

Uma outra característica do *ROS* é a de permitir que para um mesmo projeto, se utilizem linguagens de programação distintas nos diversos módulos desenvolvidos. Neste ambiente a comunicação entre processos é feita através de mensagens. As mensagens em *ROS* são estruturas predefinidas.

A comunicação entre módulos é realizada através da passagem de mensagens entre si. Um módulo envia uma mensagem através da publicação de um tópico (*string* que identifica a mensagem). Em projetos com alguma dimensão é comum existirem diversos tópicos a serem publicados e subscritos por um ou diversos módulos.

Em sistemas robóticos, há a necessidade de seguir e permitir efetuar transformações no espaço entre referenciais de vários sensores e entre robôs móveis e referenciais fixos. Por esse motivo foi desenvolvido em *ROS* uma ferramenta que permite realizar e detetar essas transformações. Esta ferramenta relaciona todos os referenciais presentes num determinado sistema, não sendo necessária a definição das matrizes de transformação sempre que se deseja publicar uma transformação.

Em grandes projetos (desenvolvidos em *ROS*) por vezes torna-se complicado observar o estado de todo o sistema. Nesse sentido, foram desenvolvidas diversas ferramentas que permitem analisar o estado das comunicações entre módulos; que transformações se



encontram a ser realizadas; um osciloscópio virtual que permite observar os dados que se encontram a circular entre módulos, entre muitas outras. Estas ferramentas permitem facilitar o desenvolvimento e o *debugging* de sistemas.



## Capítulo 2

# Localização do *AtlasCar* e seu estado

Em robótica móvel, conhecer a posição do robô/veículo e a sua orientação relativamente ao ambiente que o rodeia é essencial. O *AtlasCar*, consiste numa plataforma de desenvolvimento de novos sistemas e, como tal, inclui diversos sensores e atuadores. Estes sensores, câmaras, *lasers*, sensor inercial, *encoder*, entre outros, permitem obter a informação sobre seu estado e sobre o espaço que o rodeia.

De seguida, irá ser apresentada a metodologia levada a cabo neste trabalho para a obtenção dessa informação e as melhorias aplicadas face ao que se encontrava já desenvolvido.

### 2.1 Sistemas de Coordenadas do *AtlasCar*

A colocação de sensores no *AtlasCar* não é suficiente, é necessário poder relacioná-los entre si e transmitir essa informação ao sistema responsável pela recolha e tratamento dos dados. Em *ROS*, essa informação é disponibilizada por ficheiros do tipo Unified Robot Description Format (*URDF*), que permitem descrever um determinado robô. A informação disponibilizada por este tipo de ficheiro permite definir a posição e orientação dos sensores relativamente a um referencial à escolha.[Quigley et al., 2009] A organização e atribuição dos nomes aos sistemas de coordenadas foi feita de forma a identificar em primeiro lugar de que tipo de sensor se trata (*/laser/*, */camera/*) e de seguida o nome do dispositivo ou sua função. No entanto nem todos os referenciais estão associados a sensores, como são o caso de */rear\_axis*, */center\_car\_axis* e */center\_bumper*. Estes referenciais têm como objetivo auxiliar ou possibilitar que certas transformações ou cálculos sejam feitos na posição correta em relação ao carro.

Na figura 2.1 é apresentado um esquema onde é possível observar a localização dos referenciais associados a cada sensor. A definição da posição e orientação de cada sensor, foi feita relativamente a um eixo comum a todos, o */center\_bumper*, possibilitando assim que toda a informação recolhida seja analisada não em cada referencial próprio mas sim num referencial global do carro.

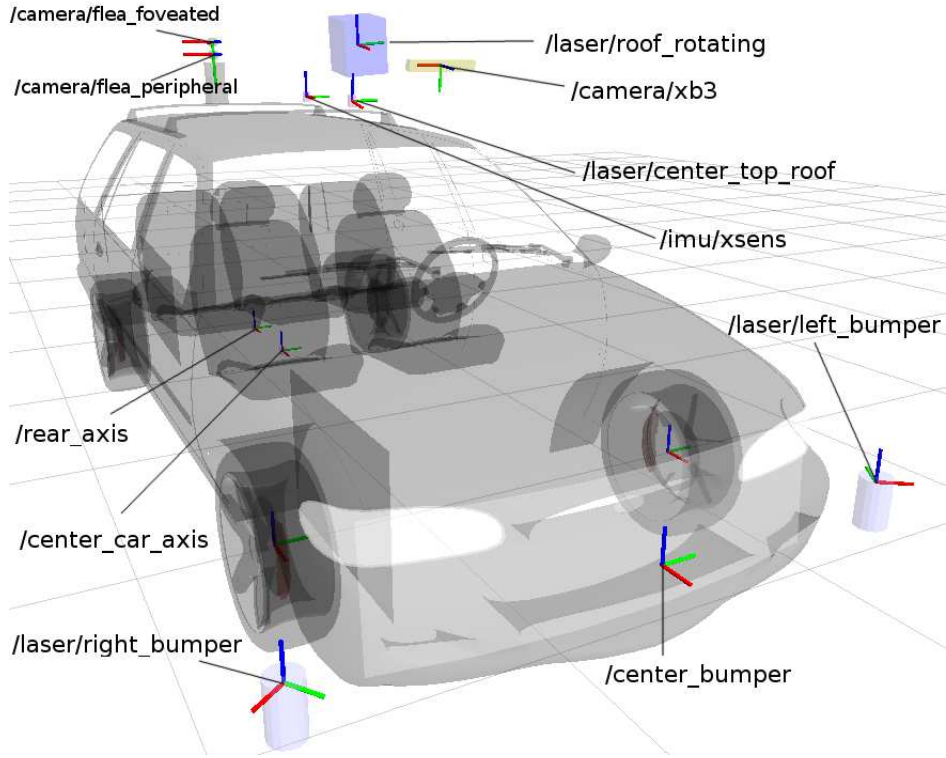


Figura 2.1: Sistema de referenciais presentes no *AtlasCar*.

## 2.2 Filtro de *Kalman* e Filtro de *Kalman* Extendido

Nos projetos em que a sua fonte de informação é constituída por diversos sensores, torna-se importante classificar e retificar a informação recolhida.

O filtro de Kalman é um algoritmo que permite, através da recolha de diversas medições ao longo de um dado intervalo de tempo, estimar o valor de uma nova medida com um grau de precisão bastante superior ao de uma outra que fosse obtida diretamente por um sensor.

Este filtro aborda o problema de forma estimar o estado  $x_k \in \mathbb{R}^n$  de um sistema discreto que seja caracterizado por equações diferenciais lineares [Welch and Bishop, 1995]

$$x_k = A_k x_{k-1} + B u_{k-1} + w_{k-1} \quad (2.1)$$

sendo as medições  $z \in \mathbb{R}^m$

$$z_k = H_k x_k + v_k \quad (2.2)$$

Onde  $x_{k-1}$  representa o vector de estado da iteração anterior e  $u_{k-1}$  o vector de controlo e  $H_k$  matriz sensibilidade à medição. As variáveis  $w_k$  e  $v_k$  representam o ruído de processo e de medição, respetivamente.

$$p(w) \sim N(0, Q) \quad (2.3)$$

$$p(v) \sim N(0, R) \quad (2.4)$$

O filtro de *Kalman* encontra-se dividido em duas partes: equações responsáveis por estimarem uma nova medida (previsão) e equações que incorporam as novas medições no estado estimado anterior (correção). De forma a estimar um novo estado  $\hat{x}_k^-$  (equação 2.5) é utilizada a estimativa à *posteriori* da iteração anterior  $\hat{x}_{k-1}$ , a matriz de estado de transição  $A$ , a matriz de controlo  $B$  e o vector  $u_{k-1}$  [Welch and Bishop, 1995].

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1} \quad (2.5)$$

Da mesma forma, a estimativa para a incerteza associada à previsão do vector  $\hat{x}_k^-$ , é dado por

$$P_k^- = AP_{k-1}A^T + Q_k \quad (2.6)$$

onde  $Q$  é a matriz de covariância do ruído do processo.

Estando a fase de estimativas concluída, de seguida procede-se à correção com a incorporação de novas medições. Para tal, pela equação 2.7, é calculado o ganho do filtro,  $K_k$  que traduz o peso relativo a atribuir às estimativas e medições, sendo o seu valor proporcional à incerteza da estimativa e inversamente proporcional à incerteza da medição.

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \quad (2.7)$$

Assim de forma a corrigir a estimativa à *posteriori*  $\hat{x}_k$ ,

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \quad (2.8)$$

sendo  $z_k$  a medida atual e  $H\hat{x}_k^-$  a previsão da medida.

O filtro de *Kalman* descrito atrás refere-se a processos que são regidos por equações lineares, no entanto na maioria dos casos tais processos não são lineares, sendo necessária a utilização de um filtro de *Kalman* extendido. Este filtro permite fazer estimativas através da linearização do estado atual, através do uso de derivadas parciais das funções que agora descrevem o processo e as medições.

O vector de estado ( $x_k \in \mathbb{R}^n$ ) é agora descrito por uma função não linear, equação 2.9

$$x_k = f(x_{k-1}, u_{k-1}, w_{k-1}) \quad (2.9)$$

e o vector de medição  $z_k \in \mathbb{R}^m$

$$z_k = h(x_k, v_k) \quad (2.10)$$

onde mais uma vez  $w_{k-1}$  e  $v_k$  representam o ruído de processo e medição, respetivamente. Em termos práticos,  $w_k$  e  $v_k$  não são conhecidos a cada instante e por esse motivo as equações 2.9 e 2.10 passam a aproximações segundo

$$\tilde{x}_k \approx f(x_{k-1}, u_{k-1}, 0) \quad (2.11)$$

e

$$\tilde{z}_k \approx h(x_k, 0) \quad (2.12)$$

À semelhança do filtro de *Kalman* original, este também se encontra dividido em duas partes: uma de previsão e outra de correção das previsões com novas medições. A fase de previsão é agora regida por

$$\hat{x}_k^- \approx f(\hat{x}_{k-1}, u_{k-1}, 0) \quad (2.13)$$

e

$$P_k^- = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T \quad (2.14)$$

onde  $A$  é agora o jacobiano de  $f_{[i]}$  em relação a  $x_{[j]}$ , equação 2.15

$$A_{[i,j]} = \frac{\partial f_{[i]}}{\partial x_{[j]}} \quad (2.15)$$

e  $W$  o jacobiano de  $f_{[i]}$  em relação a  $w_{[j]}$ , equação 2.16

$$W_{[i,j]} = \frac{\partial f_{[i]}}{\partial w_{[j]}} \quad (2.16)$$

Sendo seguida pela fase de correção da previsão

$$K_k = P_k^- H^T (H P_k^- H^T + V_k R_k V_k^T)^{-1} \quad (2.17)$$

$$\hat{x}_k = \hat{x}_k^- + K_k (z_k - h(\hat{x}_k^-, 0)) \quad (2.18)$$

$$P_k = (I - K_k H_k) P_k^- \quad (2.19)$$

onde  $H_k$  e  $V_k$  são os jacobianos associados às medições, [Welch and Bishop, 1995]

$$H_{[i,j]} = \frac{\partial h_{[i]}}{\partial x_{[j]}} \quad (2.20)$$

$$V_{[i,j]} = \frac{\partial h_{[i]}}{\partial v_{[j]}} \quad (2.21)$$

## 2.3 Hodometria

A hodometria é o método de localização mais utilizado em robótica móvel, dada a sua simplicidade de utilização e implementação. Este método permite estimar uma posição com base na integração do movimento do robô. Neste projeto foi utilizada a informação vinda das rodas para determinar a posição do *AtlasCar*.

Existem diversos tipos de robôs, cada um com a sua funcionalidade e com um sistema de locomoção adaptado. Por esse motivo, durante a navegação é necessário fazer a distinção entre dois tipos de robôs, os holonómicos e os não-holonómicos, sob risco de se incorrer em erro durante a estimativa da posição, ao longo do movimento do robô.

É o sistema de atuação de um robô que o define como sendo ou não holonómico. Para um robô holonómico, o número de graus de liberdade controlados por ele tem de ser igual ao número de graus de liberdade total. Pelo contrário, o mesmo já não acontece com os robôs não-holonómicos, visto que estes possuem um número total de graus de liberdade

superior àqueles que conseguem controlar. Dando como exemplo, um sistema holonómico tem a capacidade de para uma dada orientação movimentar-se em qualquer direção sem constrangimento.[Latombe, 1991, Almeida and Santos, 2012] Por outro lado, para robôs não holonómicos o movimento já não é independente da orientação, quer isto dizer que a direção tomada por um qualquer sistema está inerentemente associada à orientação com que este se encontra a cada instante, admitindo que não há escorregamento. O *AtlasCar* pertence à segunda configuração descrita, como é típico em automóveis.

### 2.3.1 Modelo Utilizado

O modelo utilizado para determinar a posição do carro ao longo do tempo é baseado no robô, com direção do tipo *Ackermann*. Este tipo de robô, para o *AtlasCar* assume no cálculo a existência de uma única roda exatamente a meio, para os eixos de trás e da frente, designado de *Modelo Bicicleta*. [Latombe, 1991]

O cálculo da hodometria do carro necessita da informação da velocidade com que este se encontra e da orientação das rodas dianteiras. Um *encoder* (Figura 2.3) encontra-se colocado na roda esquerda do eixo traseiro, fornecendo o deslocamento ( $\Delta l$ ) da roda através da contagem dos pulsos. Sabendo que o deslocamento foi obtido durante um dado intervalo de tempo ( $\Delta t$ ), facilmente se determina a velocidade pela equação 2.22.

$$V_{l_{wodo}} = \frac{\Delta l}{\Delta t} \quad (2.22)$$

A monitorização da orientação das rodas dianteiras é realizada com recurso a um potenciômetro que se encontra acoplado à coluna da direção.



Figura 2.2: *Encoder* na roda esquerda do eixo traseiro.

O modelo utilizado toma em consideração a orientação média das duas rodas dianteiras e a velocidade média das de trás. No *AtlasCar*, como já foi referido, sendo a medição da orientação das rodas feita através da leitura de um potenciômetro, toma-se esse valor

como o valor médio. Visto o eixo de trás só possuir um *encoder*, uma das rodas não está a ser monitorizada, impossibilitando a determinação do valor médio da velocidade desse eixo. Assim, de forma a resolver esta limitação e a determinar a velocidade no centro do eixo traseiro, foi necessário proceder à determinação do raio de curvatura instantâneo, equação 2.23 (mais uma vez através da orientação das rodas da frente) e aplicar um fator de correção à velocidade, equação 2.24 [Almeida and Santos, 2012].

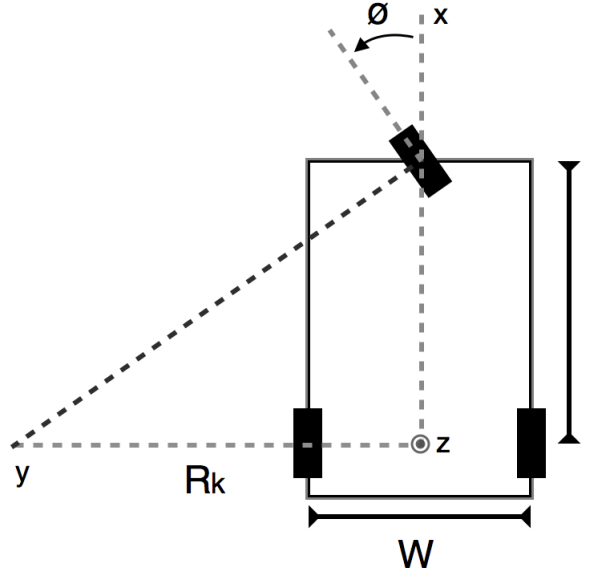


Figura 2.3: Esquema com o cinemática de *Ackermann*.

$$R_k = \frac{l}{\tan \phi} \quad (2.23)$$

$l$  - distância entre eixos

$\phi$  - orientação das rodas dianteiras

$$V_{odo} = V_{lw} \times \frac{R_k}{R_k \pm \frac{W}{2}} \quad (2.24)$$

$V_{odo}$  - velocidade no centro do eixo

$V_{lw}$  - velocidade medida

$R_k$  - raio de curva instantâneo

$W$  - distância entre rodas do eixo traseiro

Estando a determinação da velocidade e orientação feita de forma correta, procedeu-se à aplicação das formulas descritas em [Latombe, 1991]

$$\delta\psi = \sin(\phi) \frac{v}{l} \delta t \quad (2.25)$$



$$\delta x = \cos(\psi) \cos(\phi) v \delta t \quad (2.26)$$

$$\delta y = \sin(\psi) \cos(\phi) v \delta t \quad (2.27)$$

onde  $v = V_{odo}$ ,  $\delta\psi$  a variação da orientação em torno de  $z$  e  $\delta x$  e  $\delta y$  a variação das coordenadas no plano.

As equações 2.25, 2.26 e 2.27 mostram a variação segundo  $\psi$ ,  $x$  e  $y$  respetivamente. O cálculo efetivo da posição e orientação do *AtlasCar* é obtido a partir da posição anterior com a nova variação calculada, como mostram as equações 2.28, 2.29 e 2.30.

$$\psi_i = \psi_{i-1} + \delta\psi \quad (2.28)$$

$$x_i = x_{i-1} + \delta x \quad (2.29)$$

$$y_i = y_{i-1} + \delta y \quad (2.30)$$

### 2.3.2 Aplicação do filtro de *Kalman* estendido e estimativa da posição

O cálculo da hodiometria era realizado com a utilização direta dos dados fornecidos pelos sensores, não havendo qualquer controlo sobre o seu estado. Assim com a aplicação do filtro torna-se possível que este, na fase de correção da estimativa atribua um maior ou menor peso à nova medição (caso exista) de acordo com o seu grau de incerteza, permitindo assim uma redução de possíveis erros no cálculo da posição. Este filtro, utilizado no módulo responsável pelo cálculo da hodiometria, foi aplicado sobre todas as variáveis de estado presentes. A sua aplicação foi feita recorrendo a uma biblioteca, *The KFilter Project* [Zalzal, 2012].

#### 2.3.2.1 Definição do modelo físico

Aplicação do filtro de *Kalman*, implica a definição do modelo físico que descreve o comportamento do veículo enquanto este se desloca e o conhecimento das suas variáveis de estado. A definição do modelo, consiste na construção das diversas matrizes referidas na secção 2.2.

O primeiro passo passou por definir o vetor de estados  $x_{[j]}$  a utilizar. Este é constituído pelas coordenadas no plano do carro e sua orientação, e os valores de velocidade e orientação das rodas.

$$x_{[j]} = \begin{bmatrix} x \\ y \\ \psi \\ v \\ \phi \end{bmatrix}$$

O vetor de medições  $z_k$ , indica ao filtro que medidas são para ser tomadas como *input*,

$$z_k = \begin{bmatrix} v \\ \phi \end{bmatrix}$$

A utilização do filtro de *Kalman* não é possível para um modelo deste tipo, devido ao facto de não ser linear, condição essencial para a sua utilização. Assim de forma a contornar o problema, recorre-se ao um filtro de *Kalman* extendido.

A definição da matriz de estado  $A$ , é agora obtida através do jacobiano de  $f_{[i]}$  em relação a  $x_{[j]}$ , equação 2.15.

$$x_k = f(x_{k-1}, 0, 0) \quad (2.31)$$

$$A = \begin{bmatrix} 1 & 0 & -\sin(\psi)\cos(\phi)v\delta t & \cos(\psi)\cos(\phi)\delta t & -\cos(\psi)\sin(\phi)v\delta t \\ 0 & 1 & \cos(\psi)\cos(\phi)v\delta t & \sin(\psi)\cos(\phi)\delta t & -\sin(\psi)\sin(\phi)v\delta t \\ 0 & 0 & 1 & \sin(\phi)\frac{\delta t}{l} & \cos(\phi)\frac{v}{l}\delta t \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

A matriz  $H$  corresponde à relação que existe entre vetor de estados e o vetor de medição, fornecendo ao filtro a informação que variáveis de estado correspondem a medições.

$$z_k = h(x_k, 0) \quad (2.32)$$

$$H_{[i,j]} = \frac{\partial h_{[i]}}{\partial x_{[j]}} \quad (2.33)$$

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

A matriz associada ao ruído presente nas medições, orientação das rodas e velocidade, foi obtida através da análise das respetivas curvas (Figuras 2.4 e 2.5). Foi calculada a diferença entre o valor medido e o valor obtido através de um filtro de média de forma a ter um indicador da covariância relativamente a cada medição segundo a equação 2.34

$$\sigma_i = x_i - x_{imean} \quad (2.34)$$

$$R = \begin{bmatrix} R_{vel} & 0 \\ 0 & R_{steer} \end{bmatrix}$$

A determinação da matriz ruído de processo  $Q$ , foi obtida experimentalmente com alteração dos seus valores, até os resultados obtidos convergirem. O processo de determinação dos valores desta matriz foi feito de forma a encontrar um equilíbrio entre o atraso e o ruído presente nas medições filtradas.

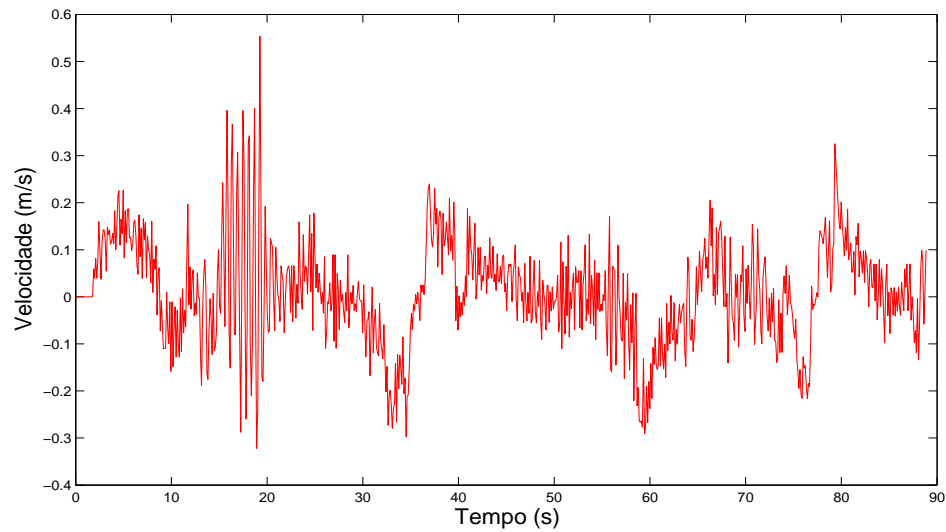


Figura 2.4: Velocidade: Desvio dos valores medidos em relação ao valor médio ( $cov = 0.01222m/s$ ).

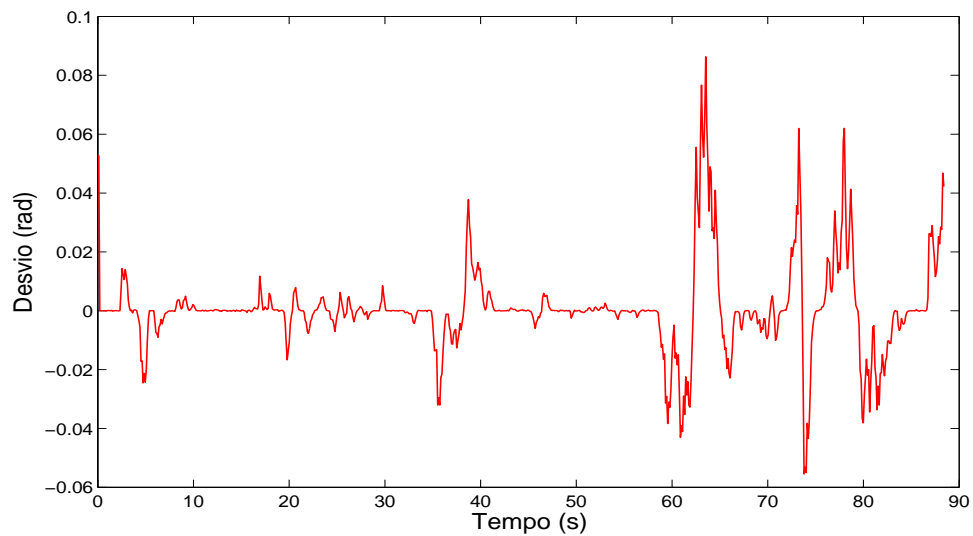


Figura 2.5: Orientação Rodas dianteiras: Desvio dos valores medidos em relação ao valor médio ( $cov = 0.000222rad$ ).

## 2.4 Determinação da orientação do *AtlasCar*

Apesar da hodiometria permitir obter a localização do carro, esta não permite saber a sua orientação relativamente ao que o rodeia, e por esse motivo houve a necessidade de colocar um sensor inercial e 4 sensores optoeletricos solidários com o chassi.

### 2.4.1 Sensor Inercial - IMU

O sensor inercial instalado é o *Mti Xsens* (Figura 2.6) . Este dispositivo é constituído por 3 magnetómetros, 3 giroscópios e 3 acelerómetros, cada um segundo as direções principais. O seu objetivo é determinar a orientação global do *AtlasCar* a cada instante. O cálculo da orientação segundo os três graus de liberdade é alcançado através da fusão da informação adquirida por cada um dos sensores, através da utilização de um filtro de *Kalman* presente no próprio dispositivo.

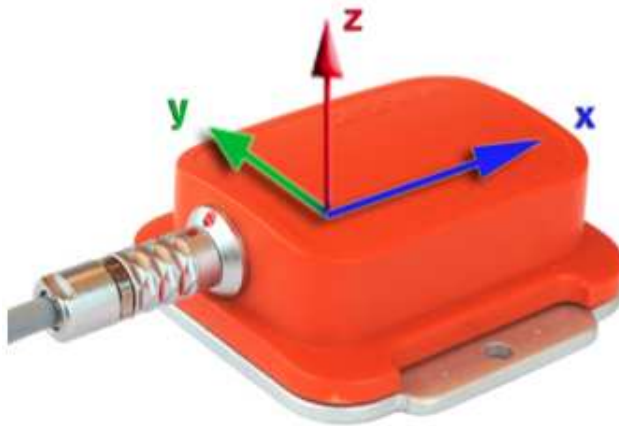


Figura 2.6: Sensor Inercial *Mti Xsens*.

O conjunto de sensores mais importante num dispositivo desta natureza são os giroscópios, pois permitem medições em sistemas com grande dinâmica. É a partir deles que a orientação é calculada por integração da velocidade angular. No entanto, este conjunto só por si não é suficiente para que a informação sobre a orientação seja fidedigna, já que a cada integração o valor utilizado para o cálculo está sujeito a erro, propagando-se com o passar do tempo. Assim, de forma a que o erro seja anulado são utilizados os outros dois conjuntos de sensores (acelerómetros e magnetómetros) que permitem fornecer ao dispositivo o valor absoluto segundo cada direção, *roll*, *pitch* e *yaw*. Os magnetómetros são os responsáveis pela correção do *yaw* ao longo do tempo. Estes funcionam como bússola, fornecendo ao sensor o valor absoluto de orientação através da medição do campo magnético local. No que toca aos restantes ângulos de inclinação, estes são estabilizados pelos três acelerómetros instalados no dispositivo, através da medição da aceleração da gravidade [Xsens, 2009]. Através dos valores existentes segundo cada componente, é possível estimar o valor de inclinação absoluta com que o dispositivo se encontra.

### 2.4.2 Limitações da IMU e Sensores Complementares

O sensor inercial instalado no carro tinha como objetivo determinar a orientação do carro em relação à estrada devido à suspensão. Este dispositivo no entanto tem diversas limitações para aplicações em robôs tipo carro. O sensor inercial não consegue distinguir o que são acelerações de gravidade das acelerações inerciais devido ao movimento do carro e não consegue distinguir as orientações medidas quando um carro se encontra a acelerar num plano horizontal de um carro a velocidade constante num plano inclinado. Assim de forma a compensar essas limitações, optou-se por fazer a monitorização da altura e

da orientação do chassi do *AtlasCar* em relação ao chão imediatamente por baixo. Dois tipos de sensores, optoeletrônicos e ultra-sons, foram pensados para essa tarefa, sendo o princípio de monitorização igual para ambos. Para a monitorização foram colocados 4 sensores, dois na parte dianteira do chassi e dois na parte traseira (figura 2.7), que vão medir a distância ao chão nesses quatro pontos. Com a informação enviada por estes sensores, torna-se possível o cálculo da altura média do carro em relação à estrada e os respectivos ângulos de inclinação (figura 2.8), como demonstram as equações 2.35, 2.36 e 2.37.

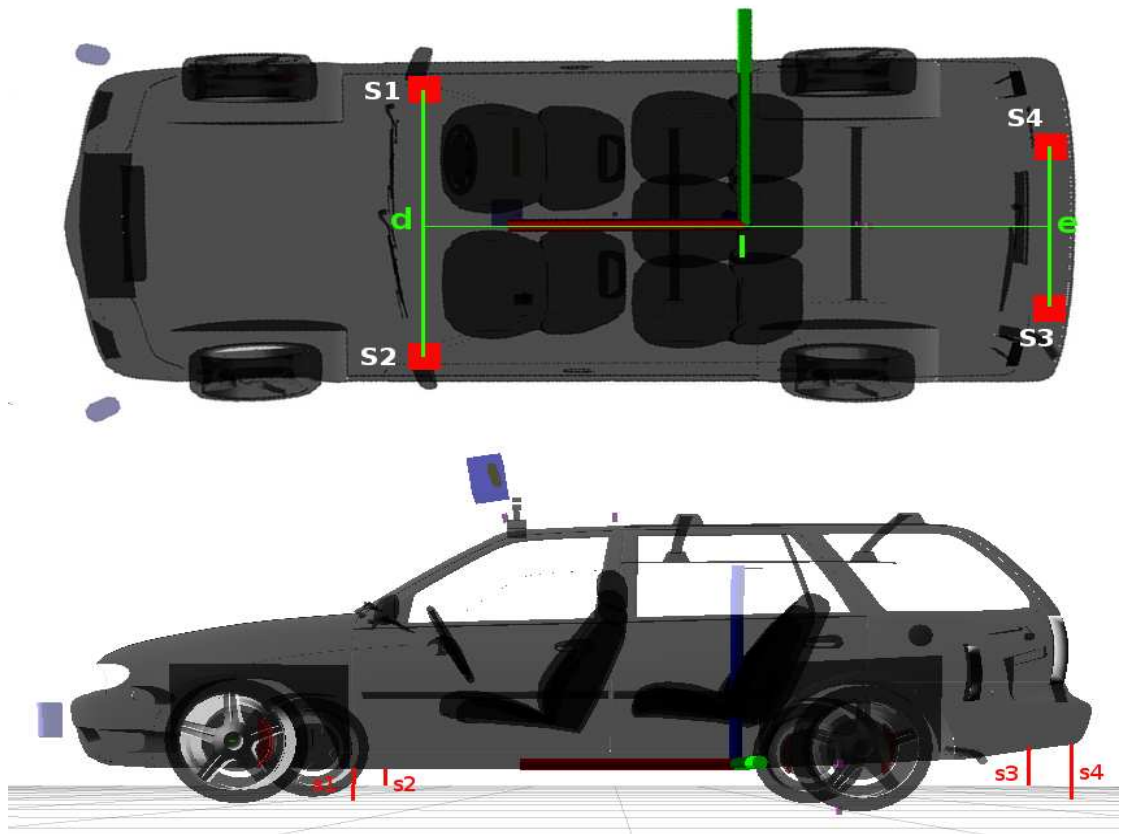


Figura 2.7: Vista de baixo e da lateral do carro-Esquema com a posição dos sensores *Sharp* no *AtlasCar*.

$$roll = \arctan \left( \frac{\frac{s1+s4}{2} - \frac{s2+s3}{2}}{\frac{d+e}{2}} \right) \quad (2.35)$$

$$pitch = \arctan \left( \frac{\frac{s3+s4}{2} - \frac{s1+s2}{2}}{l} \right) \quad (2.36)$$

$$z = \frac{s1 + s2 + s3 + s4}{4} \quad (2.37)$$

onde  $s1$ ,  $s2$ ,  $s3$  e  $s4$  são as distâncias medidas por cada sensor e  $e$ ,  $d$  e  $l$  as distâncias entre os sensores. Os ângulos,  $roll$  e  $pitch$  representam a rotação do carro segundo o eixo dos  $x$  (vermelho) e  $y$  (verde), respetivamente.

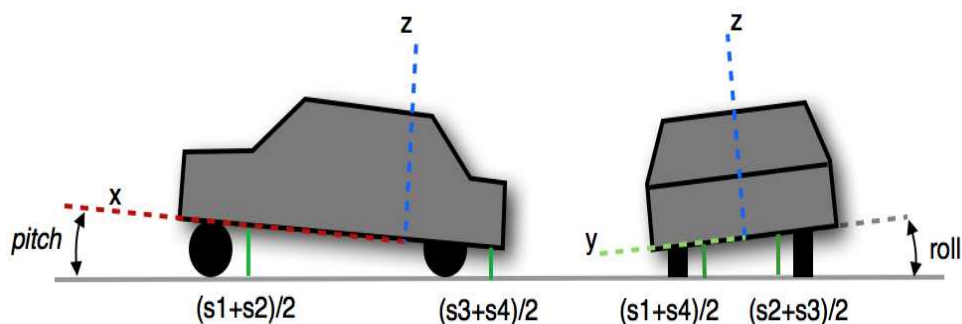


Figura 2.8: Esquema com os ângulos que o carro faz com a estrada, devido à suspensão.

A escolha recaiu sobre os sensores os optoeletrônicos *Sharp GP2D12* (figura 2.9), dada a sua facilidade de utilização, e visto que com o uso de ultra-sons, para velocidades mais elevadas do carro, os sensores colocados na parte traseira corriam o risco de não receber o eco de volta e de poderem receber os ecos dos sensores instalados na parte dianteira.

O sensor *Sharp* utilizado é um sensor analógico, que de acordo com a distância a que se encontra um determinado objeto, envia como *output* uma tensão elétrica. [Parallax, 2000] A interpretação desse valor é feita através de uma unidade de aquisição e controlo (*Arduino*), que é responsável por converter o valor da tensão numa distância. De notar que o envio do sinal desde o sensor até ao *Arduino*, é feito através de cabos *Ethernet*, que estão sujeitos a campos magnéticos e elétricos criados por outros dispositivos presentes no *AtlasCar*. Estes campos fazem com que o sinal gerado pelo sensor seja afetado, perdendo-se precisão nas medições. Por este motivo torna-se necessário a aplicação de um filtro sobre as medições de forma a remover o ruído. O filtro aplicado foi um filtro de mediana. Dada a elevada frequência de envio de medições, aproximadamente  $50\text{ Hz}$ , e das grandes quantidades de ruído, aplicou-se um filtro sobre uma vizinhança de 8 valores. Esta dimensão permitiu a eliminação dos picos presentes nas medições sem atrasar demasiado o sinal filtrado, tornando o cálculo das distâncias bastante mais suave e preciso.

Este *Arduino* é também responsável por enviar a distância calculada, através de comunicação *Ethernet* para o computador instalado no carro.

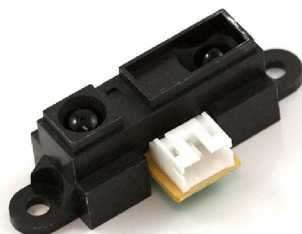


Figura 2.9: Sensor *Sharp GP2D12*.

De forma a exemplificar a razão da utilização em conjunto destes sensores, é apresentada a figura 2.10. O sensor inercial será responsável por fornecer a inclinação total do carro, i.e, o conjunto da inclinação *Carro-Estrada* ( $\beta$ ) com *Estrada-Mundo* ( $\alpha$ ), enquanto os sensores *Sharp* disponibilizarão a orientação deste em relação à estrada, inclinação *Carro-Estrada*.



Figura 2.10: Esquema exemplificativo das diferentes inclinações presentes num carro.

## 2.5 *Egomotion*

Até este ponto, toda a informação recolhida pelos diversos sensores ainda não se encontra relacionada entre si. Por esse motivo, e de forma a ter-se a caracterização do carro relativamente ao ambiente que o rodeia, tornou-se necessário fazer a fusão de toda a informação disponibilizada pelos sensores descritos anteriormente. Para tal foi desenvolvido um pacote (*atlas\_car\_egomotion*) em *ROS* e utilizado um outro previamente desenvolvido para fazer a fusão dessa informação. Este processo de fusão encontra-se dividido em duas partes principais, uma responsável pelo cálculo da posição do *AtlasCar* e outra que permite determinar a orientação do carro.

### 2.5.1 Determinação da posição do *AtlasCar*

Para o cálculo da posição do *AtlasCar* foi utilizado um pacote previamente desenvolvido em *ROS*, o *Robot Pose EKF*. Este pacote permite determinar o estado do carro a 6 graus de liberdade, através da combinação de informação de hodometria, do sensor inercial e hodometria visual/GPS, utilizando um filtro de *Kalman* estendido. Este módulo, apesar de poder receber informação das três fontes referidas atrás, não necessita que todas elas estejam presentes, podendo funcionar somente com informação de hodometria e do sensor inercial, como é o caso neste trabalho. A hodometria calculada anteriormente é responsável por fornecer a informação da posição do carro no plano  $xy$  e a sua orientação segundo  $z$ . O sensor inercial será responsável por disponibilizar a informação sobre a orientação com que o carro se encontra a cada instante.

O módulo (*Robot Pose EKF*) utilizado para o cálculo da posição não toma em consideração o valor absoluto das medições que lhe são disponibilizadas, este efetua o cálculo relativo da posição e orientação através da diferença entre os valores recebidos por cada

sensor em instantes de tempo seguidos para fazer a atualização da informação a disponibilizar ao filtro de *Kalman* do módulo.

### 2.5.2 Determinação das inclinações

Como foi referido na secção 2.4, é necessário distinguir as diferentes inclinações com que o carro se pode encontrar e para isso foi necessário a criação de 2 módulos independentes.

O primeiro módulo (*car\_pose2theground*) desenvolvido é responsável por subscrever uma mensagem com a informação das distâncias fornecidas pelos quatro sensores *Sharp* e calcular a inclinação do carro em relação à estrada (*roll<sub>car</sub>* e *pitch<sub>car</sub>*). Depois de obtidas as inclinações e altura média em relação à estrada, este módulo é responsável pela publicação da transformação respetiva entre o *chassis* do carro (*center\_car\_axis*) e a estrada (*ground*).

O segundo módulo (*ground\_pose2theworld*) é responsável pela diferenciação das duas inclinações. Este módulo, com os ângulos calculados pelo módulo anterior, vai subtraí-los aos ângulos obtidos pelo sensor inercial de forma a saber a inclinação da estrada em relação ao mundo, como mostram as equações 2.38 e 2.39

$$roll_{ground} = roll_{imu} - roll_{car} \quad (2.38)$$

$$pitch_{ground} = pitch_{imu} - pitch_{car} \quad (2.39)$$

onde *roll<sub>ground</sub>* e *pitch<sub>ground</sub>* são os ângulos da estrada em relação ao mundo e *roll<sub>imu</sub>* e *pitch<sub>imu</sub>* são ângulos medidos pelo sensor inercial. À semelhança do módulo anterior, este também é responsável pela publicação de uma transformação, agora entre a estrada e o mundo (*world*). No entanto, nesta transformação não é somente publicada a inclinação mas também a posição calculada anteriormente e a orientação segundo z.

A figura 2.11, representa o diagrama de transformações publicadas pelo pacote *atlas-car\_egomotion*.

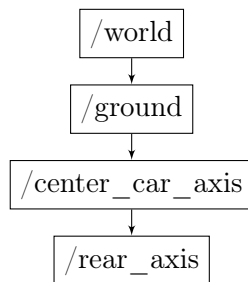


Figura 2.11: Árvore de transformações.



## Capítulo 3

# Construção do Mapa Local

A construção de um mapa local, neste trabalho, foi feita através da utilização de um *laser* que se encontra instalado na parte superior do carro (como mostra a figura 2.1). Este *laser* encontra-se com uma ligeira inclinação segundo  $y$ , de forma a que esteja apontado para a estrada à frente do carro. A inclinação foi definida de modo a que o *laser* não fosse demasiado perturbado pelo Sol e que, ao mesmo tempo permitisse obter informação com alguma antecedência sobre estado do ambiente imediatamente à frente do carro.

### 3.1 *Laser Hokuyo*

O *laser* utilizado neste trabalho tem como objetivo determinar a distância a que se encontra um objeto. Em *lasers* deste tipo, o sinal enviado é um feixe de luz. Esse feixe enviado possui uma determinada modulação e através da diferença de fase entre o feixe enviado e o recebido é calculada a distância a que se encontra o ponto.

O *laser* Hokuyo (figura 3.1) permite fazer um varrimento de  $270^\circ$  e detetar objetos num intervalo entre 10 cm a 30 m. A sua resolução angular é de  $0.25^\circ$  e cada varrimento é efetuado em 25ms. [Hokuyo Automatic CO., 2009]



Figura 3.1: Sensor *Laser Hokuyo*.

A figura 3.2 mostra um exemplo de um varrimento com a utilização do *laser Hokuyo*.

Na figura, a área a preto representa a área de varrimento na qual os pontos (a vermelho) podem ser detetados pelo *laser*.

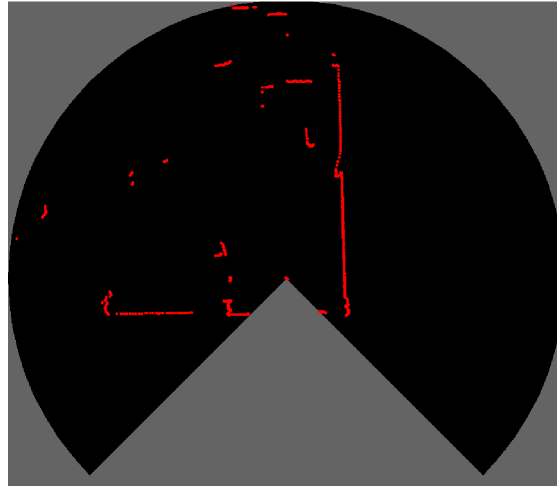


Figura 3.2: Representação de varrimento *laser*.

Para a construção do mapa local foi necessário o desenvolvimento de um módulo em *ROS*, o *pc\_accumulation*. Este módulo será responsável pela acumulação de várias nuvens de pontos à medida que o carro se desloca. Todas as operações realizadas sobre as nuvens de pontos foram feitas recorrendo a uma biblioteca já existente, a *Point Cloud Library*. [Rusu and Cousins, 2011] Esta biblioteca é constituída por diversos algoritmos e funções que permitem a aplicação de filtros, reconstrução de superfícies, identificação de objetos, entre outros. Permite também operações mais básicas como determinar a dimensão de uma nuvem de pontos, adicionar ou remover pontos, etc.

De seguida será explicado o funcionamento do módulo e as transformações necessárias para que a construção da nuvem de pontos 3D seja feita no referencial do mundo.

Inicialmente, o desenvolvimento deste módulo pressupõe somente a acumulação de varrimentos *laser*, no entanto de forma a ter uma aplicação mais geral, decidiu-se que deveria também poder subscrever a informação de outros sensores, nomeadamente das câmaras *Bumblebee XB3* e *Kinect*. A Figura 3.3, mostra um exemplo de uma reconstrução com a utilização da câmara *Kinect*, num ensaio realizado em laboratório.

A fase inicial de desenvolvimento teve como objetivo a identificação dos tipos de mensagens que poderiam ser subscritos pelo módulo. Concluiu-se que os mais comuns seriam tópicos do tipo *LaserScan* e *PointCloud*.

A informação que o sensor *laser* disponibiliza encontra-se em coordenadas polares, i.e., disponibiliza a distância a que se encontra um determinado ponto e o ângulo em que este foi medido. Visto que uma nuvem de pontos tridimensional se encontra em coordenadas cartesianas, é necessário converter a informação vinda do laser, como mostra o sistema de equações 3.1.

$$\begin{cases} x_i = r_i \cos(\theta_i) \\ y_i = r_i \sin(\theta_i) \\ z_i = 0 \end{cases} \quad (3.1)$$

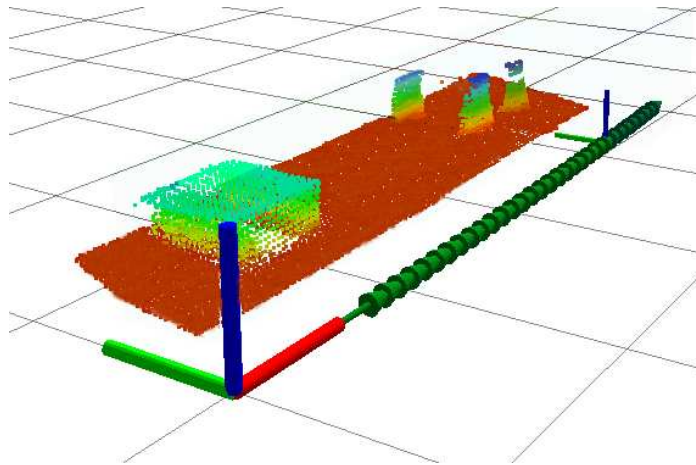


Figura 3.3: Exemplo de Reconstrução com *Kinect*.

onde  $r_i$  é distância medida e  $\theta_i$  o ângulo de varrimento de cada ponto.

De seguida, visto que os dados são obtidos no referencial próprio de cada sensor, é necessário proceder a uma transformação entre referenciais, normalmente entre o seu referencial de origem `/sensor` e o `/center_bumper`.

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{/center\_bumper} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & d_x \\ r_{21} & r_{22} & r_{23} & d_y \\ r_{31} & r_{32} & r_{33} & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{/sensor} \quad (3.2)$$

Do mesmo modo, quando se quer ter essa mesma informação vista do mundo, é necessário mais um conjunto de transformações, como mostra o grafo 3.4.

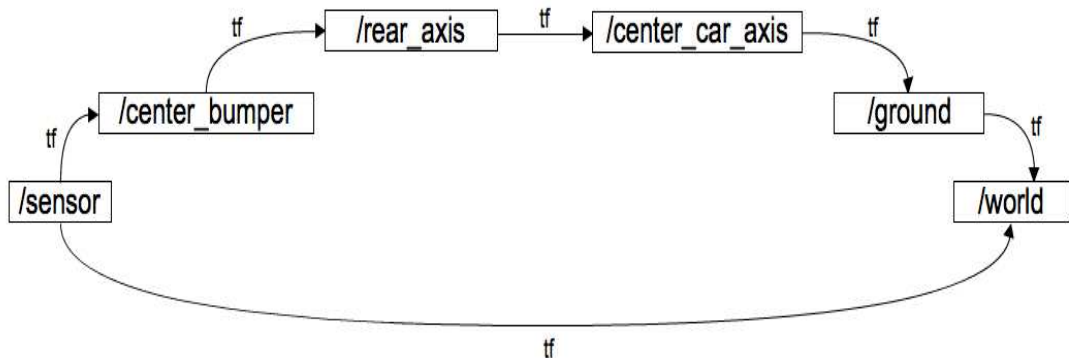


Figura 3.4: Grafo de transformações utilizado por este módulo.

Após a transformação dos dados entre referenciais, procedeu-se à acumulação da nuvem de pontos. Esta acumulação é feita de forma a que novos pontos obtidos pelos sensores sejam adicionados a uma nuvem de pontos já existente, através da concatenação de uma nuvem de pontos já existente com uma nuvem de pontos nova (operação  $+$ ). Visto que o objetivo deste módulo é permitir a construção de um mapa local, é necessário

limitar a sua dimensão. A definição da dimensão a atribuir à nuvem de pontos foi levada a cabo através do desenvolvimento de uma pequena função, cujo o objetivo é o de descartar os pontos que se encontrem a uma determinada distância ( $D$ ) de um referencial, calculada segundo a equação

$$D = \sqrt{(x_i - x_{ref})^2 + (y_i - y_{ref})^2 + (z_i - z_{ref})^2} \quad (3.3)$$

onde  $[x_i, y_i, z_i]$  são coordenadas de cada ponto da nuvem de pontos e  $[x_{ref}, y_{ref}, z_{ref}]$  coordenada do referencial escolhido. Estas coordenadas encontram-se no referencial do mundo. A distância e a escolha do referencial são parâmetros a definir sempre que se inicia o módulo.

## Capítulo 4

# Resultados Obtidos

Neste capítulo serão apresentados os resultados obtidos nas diferentes frentes que foram abordadas anteriormente. Os resultados serão apresentados conforme a evolução que o trabalho sofreu com a introdução dos novos dispositivos, mostrando a influência de cada um.

### 4.1 Correção de dados dos sensores *Sharp GP2D12*

De seguida serão apresentados diversos gráficos onde a azul estão representados o dados retirados diretamente dos sensores e a vermelho o resultado da aplicação do filtro de mediana. Os conjuntos de gráficos a apresentar, mostram situações distintas no estado do *AtlasCar*. Cada gráfico representa as medições efetuadas por cada sensor.

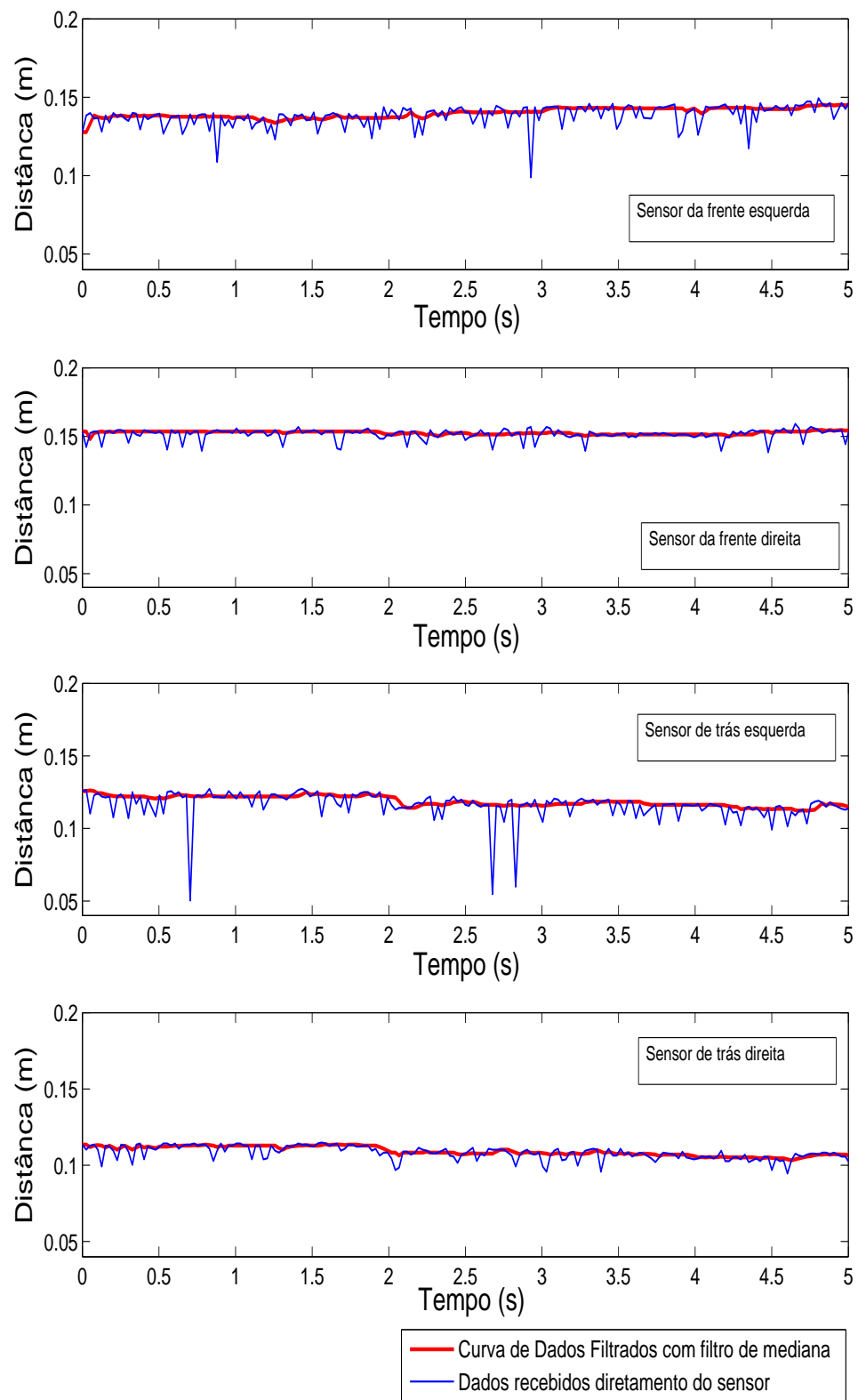
No primeiro conjunto (Figura 4.1) o *AtlasCar* encontra-se parado. É possível observar que as distâncias medidas pelos sensores se mantêm aproximadamente constantes. Nota-se porém uma diferença nos valores recolhidos pelos sensores, devido ao fato de os valores terem sido retirados com pessoas a bordo do carro, onde cada uma influencia a inclinação devido ao seu peso e devido à carga na mala do carro onde se encontram instalados o computador, o *PLC*, o inversor e a *UPS*. O filtro aplicado, como mostram os gráficos 1 e 3, compensa os picos presentes nas medições diretas.

O segundo conjunto de gráficos (Figura 4.2) mostra os valores medidos com o carro a descrever uma curva. A densidade de pontos representada é um pouco maior, dado o maior intervalo de tempo representado. É possível com a observação dos gráficos, notar a variação das distâncias medidas pelos vários sensores enquanto o carro descreve a curva,  $\Delta t = [85, 95]$  segundos. No instante representado, o carro encontra-se a curvar para a esquerda. Um carro enquanto descreve uma curva para a esquerda tende a levantar a parte esquerda e a baixar a direita em relação à estrada, como é possível observar nos gráficos representados. A variação das distâncias medidas pelos sensores da frente é mais perceptível que as medidas pelos sensores de trás, devido à sua colocação no *AtlasCar* (figura 2.7). Pode observar-se facilmente pelo gráfico referente ao sensor colocado na parte da frente à esquerda, o aumento da distância da parte esquerda do carro em relação à estrada, por outro lado pela análise do gráfico referente ao sensor da direita é possível observar uma diminuição da distância ao chão. É possível observar que o filtro aplicado permite a eliminação do ruído presente nas medições sem se atrasar demasiado em relação aos valores medidos diretamente.

De seguida mostra-se o comportamento dos sensores numa situação de arranque. Um carro quando arranca ou acelera, tende a inclinar-se para trás, i.e., a frente do carro tende a levantar e a traseira a descer. Mais uma vez é possível observar esse comportamento com a análise dos gráficos 4.3. Os dados recolhidos pelos sensores da frente mostram efetivamente um aumento da distância da parte da frente do carro em relação ao chão, enquanto os de trás como era esperado, mostram uma diminuição da distância medida em relação à estrada.

O filtro de mediana aplicado às medições, como se pôde observar, permitiu a eliminação quase total do ruído presente nas medições. Com a observação dos gráficos anteriores, nota-se que os dados dos sensores colocados na traseira do carro contêm quantidades de ruído superiores aos da frente. Este fato deve-se à maior distância que a informação tem de percorrer desde o sensor até ao *Arduino* e por estarem instalados junto à mala do carro onde se encontram o inversor e a *UPS*, dispositivos geradores de grandes campos magnéticos e elétricos.

De seguida é apresentado o atraso provocado nas medições pela aplicação do filtro (Figura 4.4). É possível observar o desfasamento entre a curva a vermelho e a curva a azul.

Figura 4.1: Dados de distância ao solo - *AtlasCar* parado.

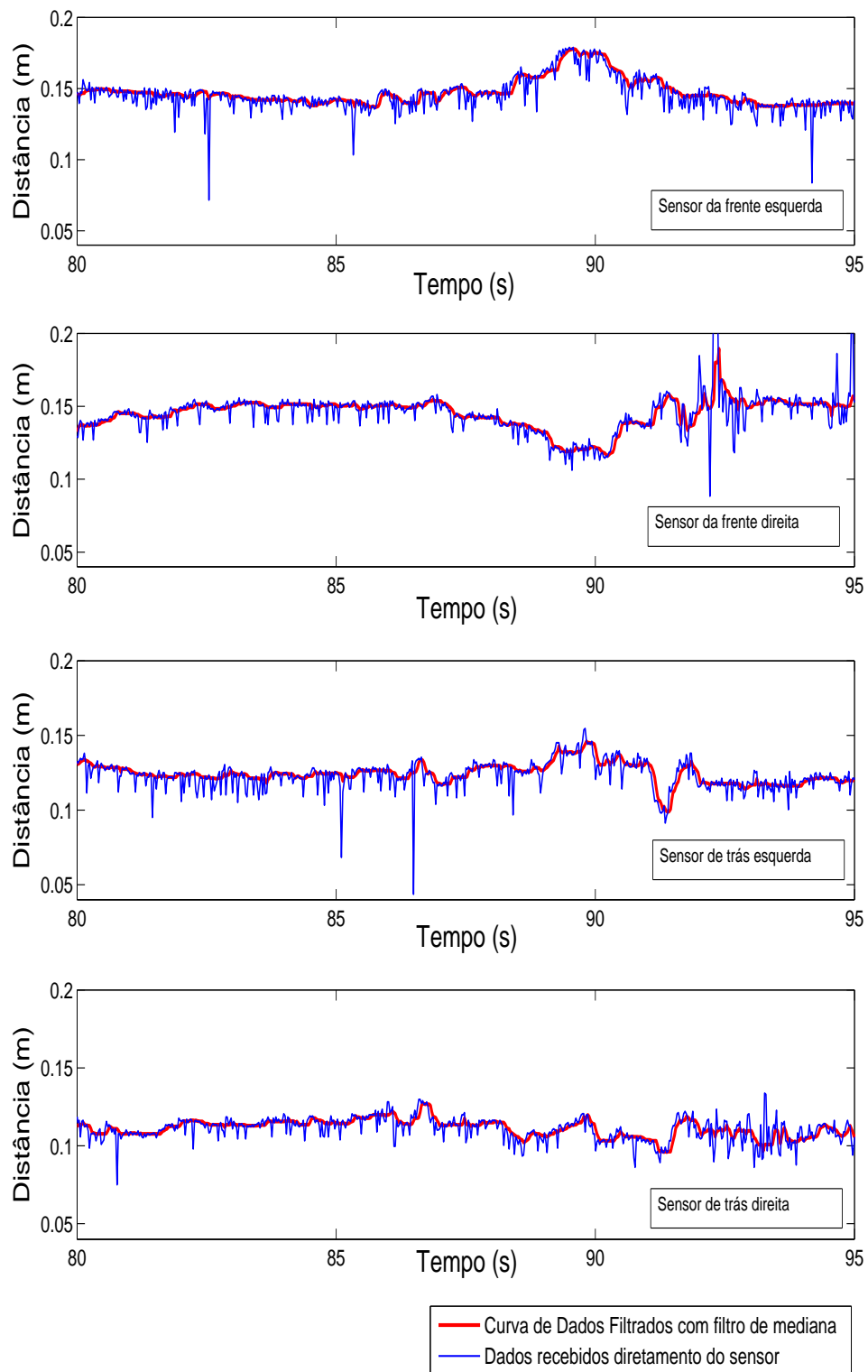
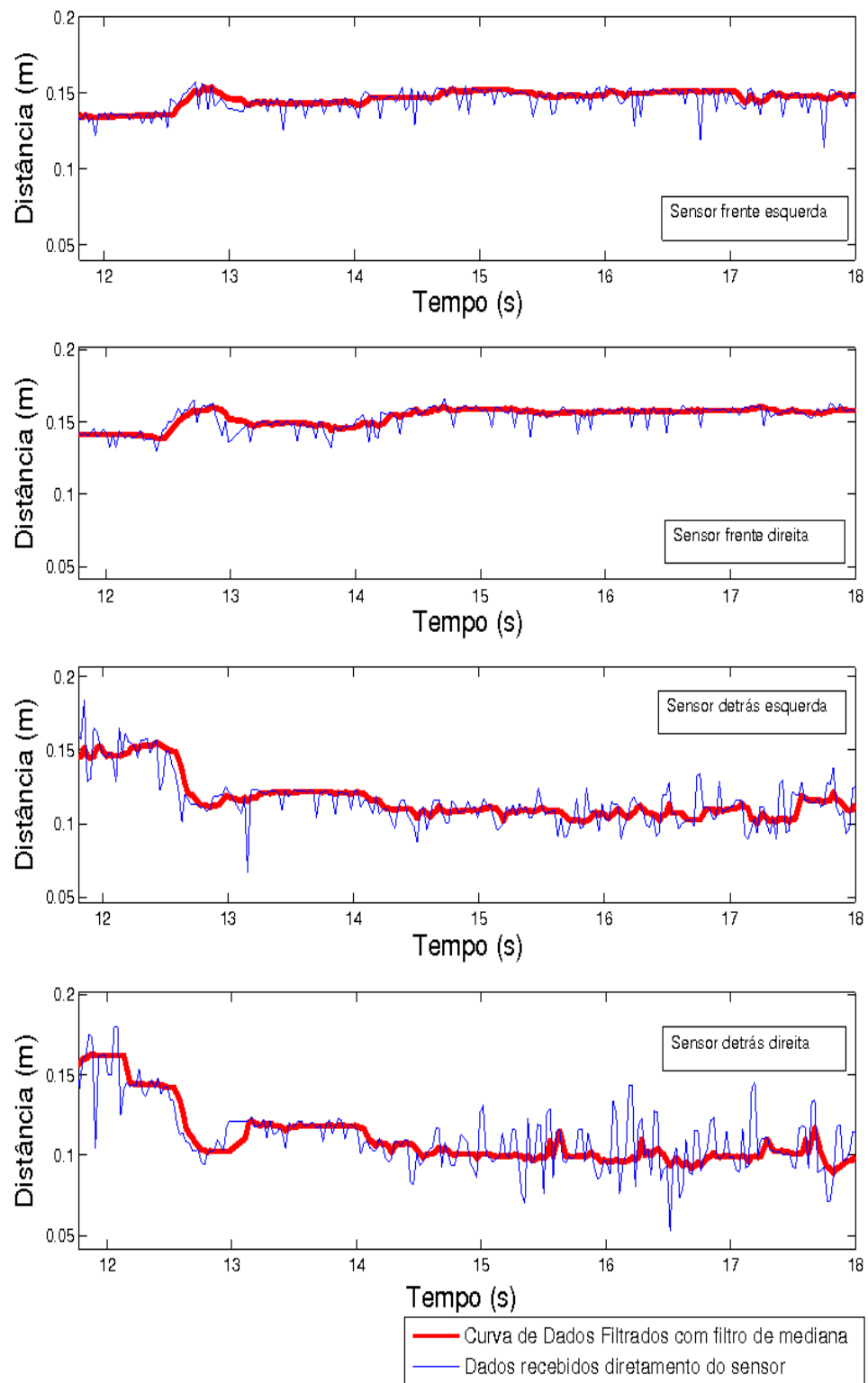


Figura 4.2: Dados de distância ao solo - *AtlasCar* a curvar.



Figura 4.3: Dados de distância ao solo - *AtlasCar* a acelerar.

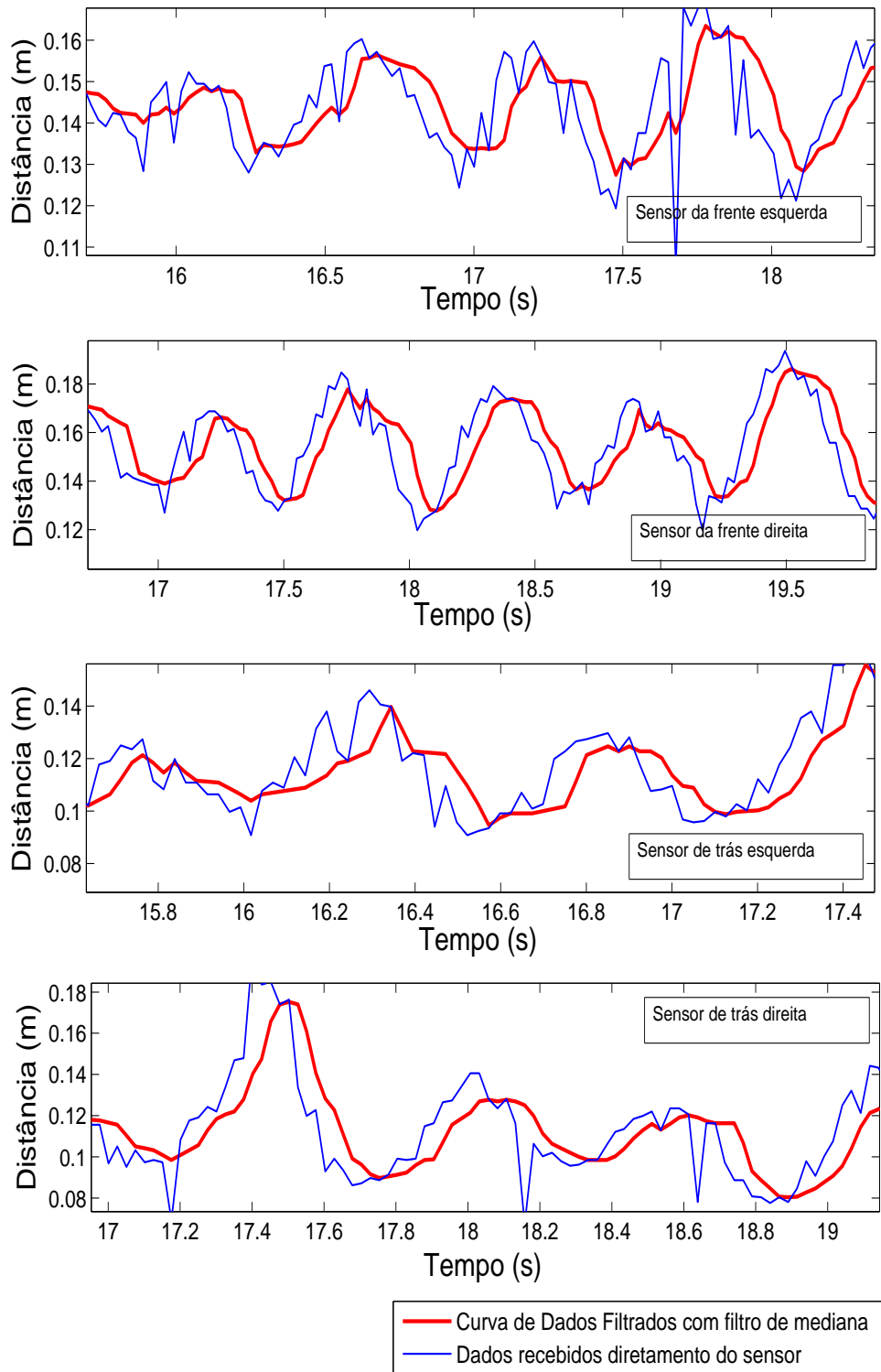


Figura 4.4: Dados de distância ao solo - Atraso do sinal filtrado em relação à medições.

## 4.2 Melhoria no cálculo da posição do *AtlasCar*

De seguida será apresentada a evolução obtida nos resultados da posição com a introdução dos métodos referidos nos capítulos anteriores. De forma a avaliar a evolução sofrida no cálculo da posição pelos diferentes módulos, recorreu-se à aplicação *Google Earth*[Google, 2012] onde é possível observar a reconstrução do caminho percorrido pelo carro.

Na figura 4.5, está representado um percurso efetuado no bairro do Alboi. A curva a azul na imagem representa a reconstrução do caminho pelo módulo responsável pelo cálculo da hodometria e a curva a rosa, o caminho calculado pelo módulo responsável pela fusão da informação de hodometria e do sensor inercial (módulo *Robot Pose EKF*). O percurso foi efetuado de forma a que este tivesse início e fim aproximadamente no mesmo ponto, para que fosse possível fechar a trajetória.



Figura 4.5: Representação da reconstrução do caminho calculado segundo o hodometria simples e segundo a fusão com o sensor inercial - Localização Alboi.

A distância total percorrida pelo carro foi de aproximadamente 650 metros. É possível observar que tanto pelo cálculo simples de hodometria (curva azul) como pelo cálculo efetuado pelo módulo *Robot Pose EKF* (curva a rosa), o erro final da posição não é significativo. No entanto, visto que o objetivo deste trabalho consiste na construção de um mapa local, é importante analisar a descrição e o comportamento do cálculo da posição destes módulos de forma mais local.

Analisando ambos os resultados obtidos, é possível reparar na constante acumulação

de erro de orientação por parte da curva azul em relação ao mapa, por outro lado, o acompanhamento do mapa pela curva rosa é mantido ao longo de todo o percurso, descrevendo de forma bastante precisa todo o caminho efetuado pelo *AtlasCar*.

De seguida é apresentado um segundo percurso (realizado no Bairro do Liceu(figura 4.6)) com características bastante diferentes, de forma a analisar a prestação do cálculo da posição. Em comparação com o percurso ilustrado anteriormente, este ultimo apresenta uma quantidade de manobras bastante inferior, sendo na sua maioria constituído por retas. A distância total percorrida foi de aproximadamente 620 metros.



Figura 4.6: Representação da reconstrução do caminho calculado segundo o hodometria simples e segundo a fusão com o sensor inercial - Localização Bairro do Liceu - Aveiro.

Na imagem a curva a azul mostra o caminho calculado por hodometria pura e a laranja pelo cálculo da posição via fusão de hodometria com sensor inercial. Pela análise da imagem, quando o *AtlasCar* muda de direção, mais uma vez é possível observar o aumento do erro da orientação por parte da curva azul. Visto que as secções retas deste percurso são bastante maiores que as do percurso anterior, o erro da posição final da curva a azul é bastante superior. Este erro acontece, pois à partida para as secções retas a orientação calculada por hodometria já possui erro associado, propagando-se o erro no cálculo da posição.

Em ambos os percursos é possível observar um constante acompanhamento do mapa por parte das curvas calculadas pelo módulo *Robot Pose EKF*, mostrando bastante precisão nas mudanças de direção tomadas pelo *AtlasCar*, permitindo uma boa caracterização

do movimento do carro em curva.

### 4.3 Resultados da Construção do Mapa Local

Todos os procedimentos que antecederam a reconstrução efetiva do espaço por onde o *AtlasCar* circula, tinham o objetivo de permitir uma construção de um mapa local de forma precisa.

De seguida serão apresentadas três imagens aproximadamente do mesmo instante, de forma a demonstrar a influência de cada módulo na construção da nuvem de pontos.

No instante representado o *AtlasCar* encontra-se a curvar para a esquerda. Um carro quando curva para a esquerda tende a inclinar-se para a direita, i.e., seguindo o sistema de referenciais presentes no *AtlasCar* (figura 2.1), dá-se uma rotação no sentido direto segundo o eixo x. Do mesmo modo, o *laser* como se encontra solidário com o chassi do carro vai rodar na mesma proporção. Assim para uma mesma faixa de estrada os pontos à direita do carro vão-se encontrar a uma distância menor que os da esquerda, sendo essa informação recolhida pelo *laser*.

A figura 4.7 representa a reconstrução obtida quando a informação da dinâmica do carro não é fornecida ao sistema. Como se pode observar a reconstrução da estrada (a verde), esta aparece inclinada quando era esperado que fosse horizontal. A reconstrução é obtida desta forma pois para o sistema o carro mantém inclinação zero em relação ao mundo, representando que a estrada é que roda e não o carro.

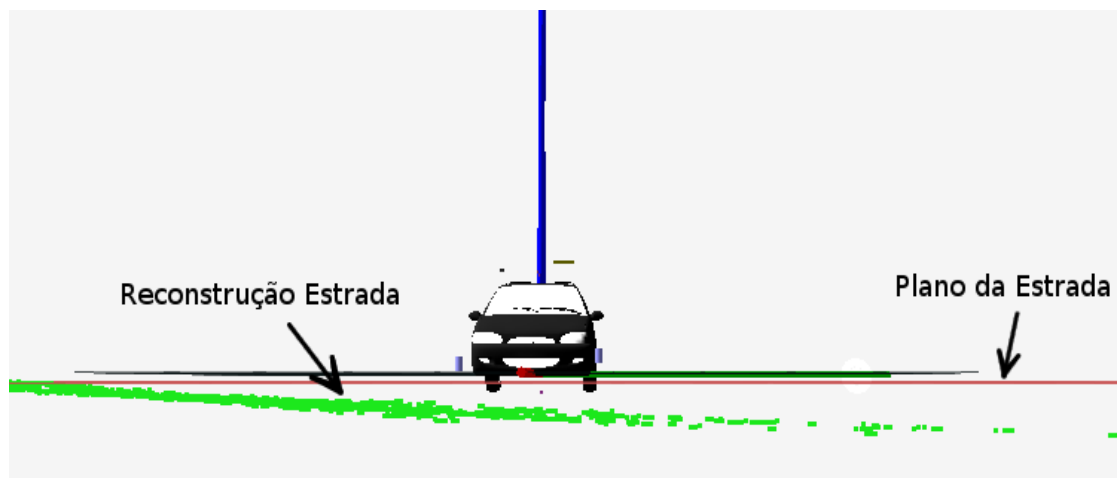


Figura 4.7: Representação sem informação dinâmica do *AtlasCar*.

De forma a corrigir a construção da nuvem de pontos foi desenvolvido o módulo que permite fornecer o estado do carro em relação à estrada. A figura seguinte mostra mais uma vez para o mesmo instante a curva efetuada pelo *AtlasCar*. Visto que já é conhecido o estado do carro em relação à estrada, i.e, são conhecidas as orientações e a altura a que o carro se encontra em relação esta, é possível observar que tanto o carro como a estrada sofreram uma rotação, tornando a representação da estrada horizontal, como acontece na realidade.

O sistema de eixos representados na imagem, mostra a efetiva rotação do carro em relação à estrada, observando que os eixos a azul já não coincidem, como acontecia na

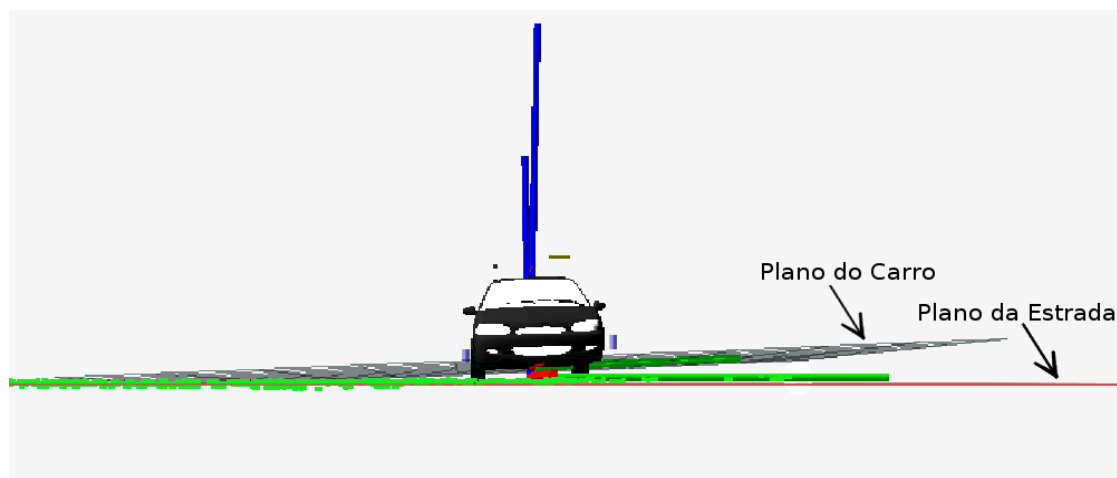


Figura 4.8: Representação com a dinâmica do carro em relação à estrada.

figura 4.7.

Como foi referido, tinha-se como objetivo a distinção da inclinação do carro em relação à estrada e desta em relação ao mundo. O local onde foi realizado esta curva é aproximadamente horizontal, sendo de esperar que a inclinação da estrada em relação ao mundo seja aproximadamente zero. A figura 4.9, representa a curva com a informação da inclinação fornecida pelo sensor inercial. Era esperado que o resultado da reconstrução fosse bastante semelhante ao representado na figura 4.8, visto que para um plano horizontal, a inclinação medida pelos sensores *Sharp* e pelo sensor inercial seria aproximadamente igual.

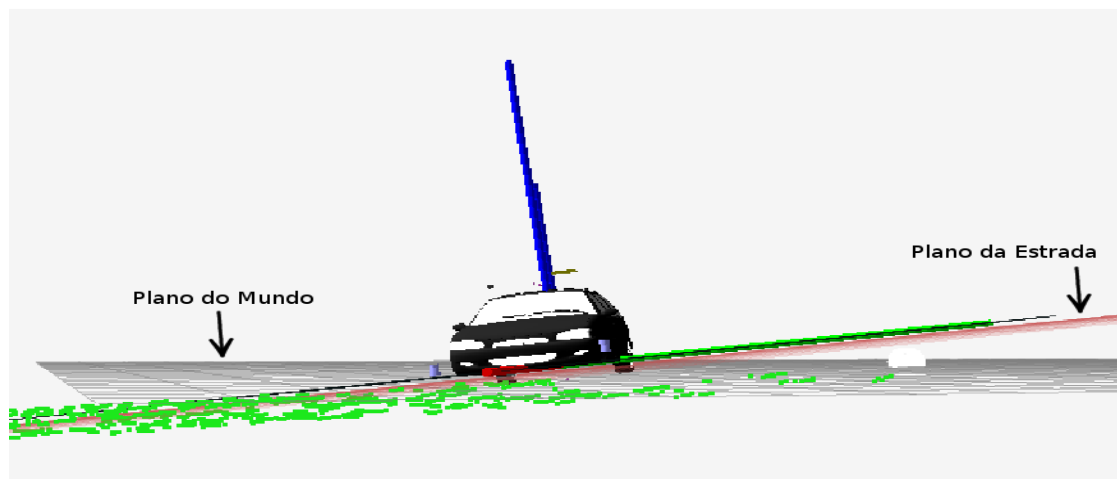


Figura 4.9: Representação com a inclinação fornecida pelo sensor inercial.

No entanto, a representação não está como esperado, observando-se uma grande rotação por parte da estrada (grelha a vermelho) em relação ao mundo (grelha a cinzento), tendo como consequência a distorção da reconstrução do mapa.

O sensor inercial quando o carro se encontra a velocidade constante e quando as mudanças de direção são suaves, permite estimar a orientação com bastante precisão.

No entanto, em regimes transitórios o sensor inercial estima rotações bastante superiores ao que acontece na realidade, como é possível observar na imagem 4.9.

Os gráficos 4.10 representam as inclinações do *AtlasCar* calculadas pelo módulo *car\_pose2ground* e as inclinações obtidas pelo sensor inercial.

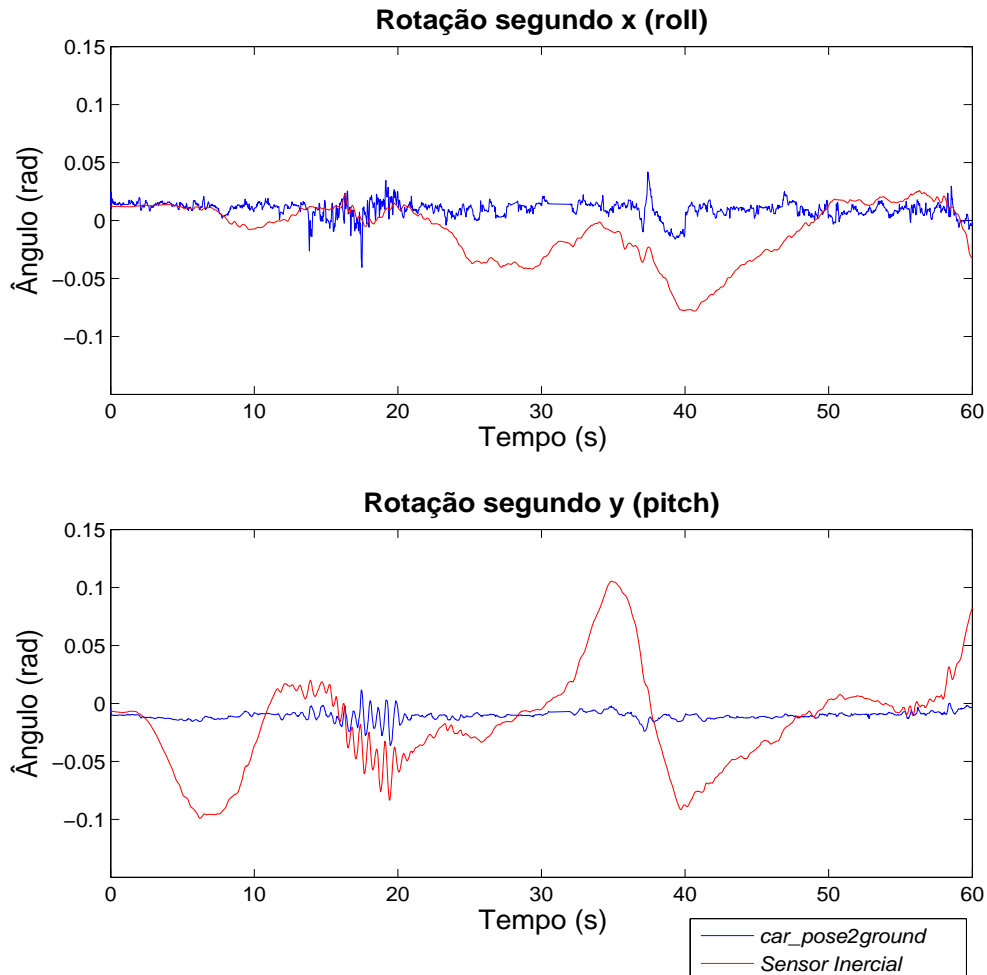


Figura 4.10: Dados da inclinação do *AtlasCar*.

Os dados foram obtidos em terreno horizontal e como se observa os valores dos ângulos de inclinação são bastante diferentes quando o *AtlasCar* se encontra em regime transitório (acelerar, travar ou a curvar).

Como foi referido anteriormente o filtro de *Kalman* presente no *Xsens* utiliza a informação dos acelerómetros para fornecer às medições um valor absoluto de inclinação. No entanto enquanto um carro se encontra em movimento, este dispositivo não distingue aceleração da gravidade, das acelerações inerciais devidas ao movimento. O filtro de *Kalman* presente, para situações em que a aceleração medida não corresponde à aceleração da gravidade, durante um intervalo de tempo, considera que a média das acelerações devido ao movimento é zero, utilizando somente as medições dos giroscópicos para o cálculo da orientação. No entanto em aplicações em robôs tipo carro esta consideração

não é válida pois muitas vezes as acelerações presentes são demasiado longas, afetando bastante o cálculo da orientação.[Xsens, 2009]

Visto que a utilização do sensor inercial para o cálculo da inclinação é bastante afetado pelas acelerações a que o carro está sujeito optou-se pela utilização somente da informação da inclinação do *chassis* em relação à estrada. Assim para a construção do mapa local, admite-se que a inclinação da estrada em relação ao mundo é constante e igual a zero.

As imagens a apresentar de seguida mostram exemplos de reconstruções obtidas através da utilização dos módulos descritos anteriormente. Será analisada a qualidade da nuvem de pontos conforme o estado da reconstrução da estrada. A escala de cores apresentada em cada imagem, mostra a variação da altura a que os pontos se encontram na nuvem de pontos. Optou-se por este tipo de representação para facilitar a distinção das formas representadas. A análise da qualidade da nuvem de pontos, vai ser realizada sobre os pontos em torno da coordenada  $z = 0$  (ao nível da estrada). Para tal, definiu-se um intervalo (figura 4.11) para a escala de cores de 0.20 metros centrado com a estrada. Os pontos fora deste intervalo, caso tenham a coordenada segundo  $z$  superior ao limite do intervalo estabelecido, são representados a roxo, por outro lado, caso os pontos estejam abaixo do limite inferior, estes são representados a vermelho.

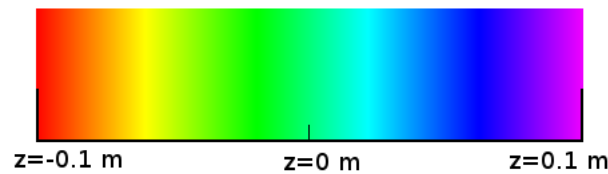


Figura 4.11: Escala de cores da nuvem de pontos.

De seguida serão apresentadas diversas reconstruções, onde se analisa a reconstrução da estrada com e sem a informação sobre o estado do carro, em relação à estrada.

As imagens 4.12 e 4.13, mostram a reconstrução da estrada para o mesmo instante. Após análise da primeira imagem é possível verificar uma maior variação da altura dos pontos na representação da estrada (mancha de pontos a azul escuro). Isto acontece devido ao fato de não ter sido disponibilizado ao sistema, a informação sobre o comportamento do carro enquanto trava, utilizando a informação vinda diretamente do sensor *laser*. Para a segunda imagem, essa variação na altura dos pontos é bastante mais suave (ausência de mancha azul escura). Essa suavização ocorre devido ao fato de o sistema saber a posição do *laser*, quando este efetua um varrimento, permitindo corrigir as coordenadas dos pontos em relação à estrada.

Uma outra situação permite claramente perceber a necessidade da presença da informação do estado do *AtlasCar*. No instante representado pelas figuras 4.14 e 4.15, o carro encontra-se a descrever uma rotunda. Observando a primeira imagem, repara-se que a reconstrução se encontra bastante inclinada, visto os pontos presentes na nuvem percorrerem toda a escala de cores. Mais uma vez, visto que o sistema não possui a informação sobre o estado do carro, os dados recolhidos pelo laser são utilizados diretamente,



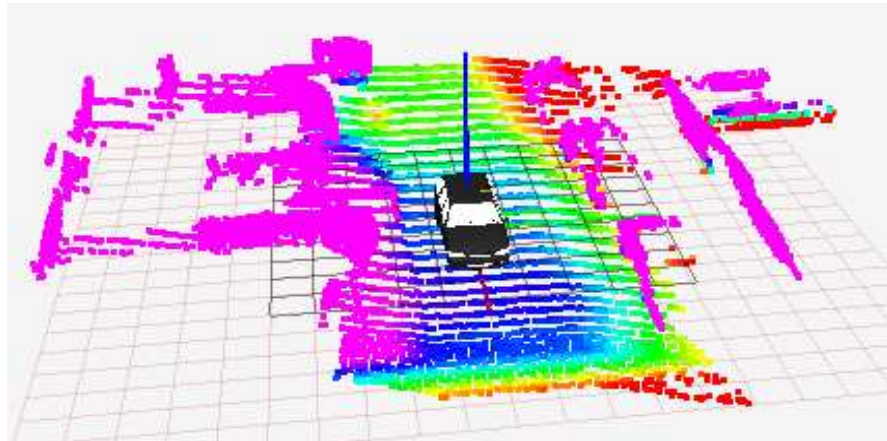


Figura 4.12: Reconstrução com *AtlasCar* a travar - Situação: Sem transformação do carro em relação à estrada.

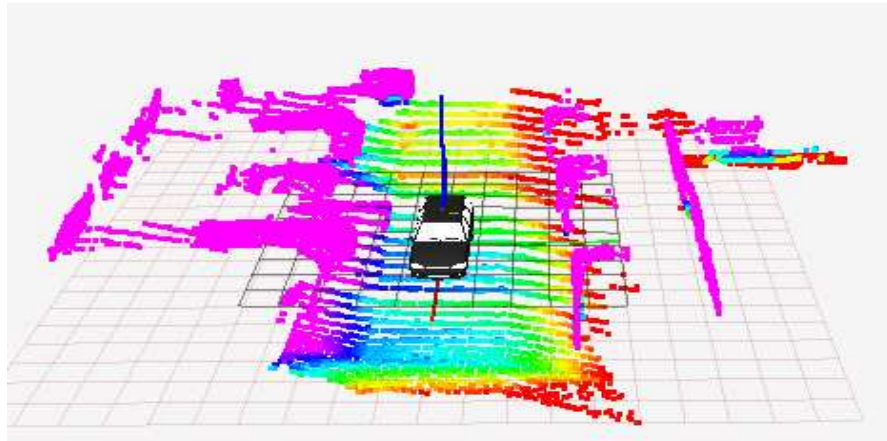


Figura 4.13: Reconstrução com *AtlasCar* a travar - Situação: Com transformação do carro em relação à estrada.

obtendo-se uma reconstrução bastante distorcida. No entanto, quando a informação sobre o estado do carro se encontra disponível, a reconstrução é bastante mais suave, como se pode observar pelas cores presentes na reconstrução (figura 4.15), que na sua maioria se mantêm à volta do valor  $z = 0$ .

De forma a analisar as reconstruções de forma quantitativa e não só qualitativa, como até agora efetuado, decidiu-se analisar pequenas porções de nuvens de pontos e averiguar a percentagem de pontos que se encontram num dado intervalo em torno de  $z = 0$ . Esta análise foi feita sobre volumes da nuvem de pontos com extensões reduzidas e imediatamente à frente do *AtlasCar*, assumindo-se que a estrada para essas extensões é plana e horizontal. A análise foi desenhada de modo a que, para um dado volume de pontos bastante reduzido, se determine o número total de pontos presentes e posteriormente se determine quantos desses pontos se encontram dentro do intervalo definido. O volume de análise definido,  $\Delta x = [0, 5]$ ,  $\Delta y = [-2, 2]$  e  $\Delta z = [-0.2, 0.2]$ , como mostra a figura 4.16.

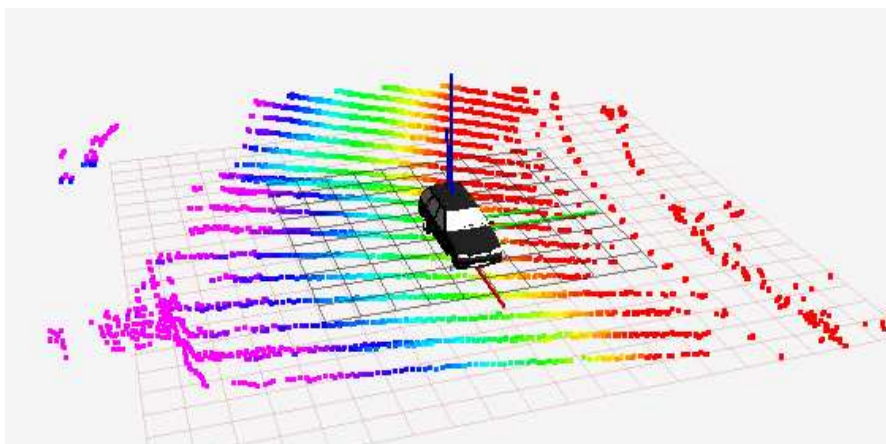


Figura 4.14: Reconstrução com *AtlasCar* a curvar - Situação: Sem transformação do carro em relação à estrada.

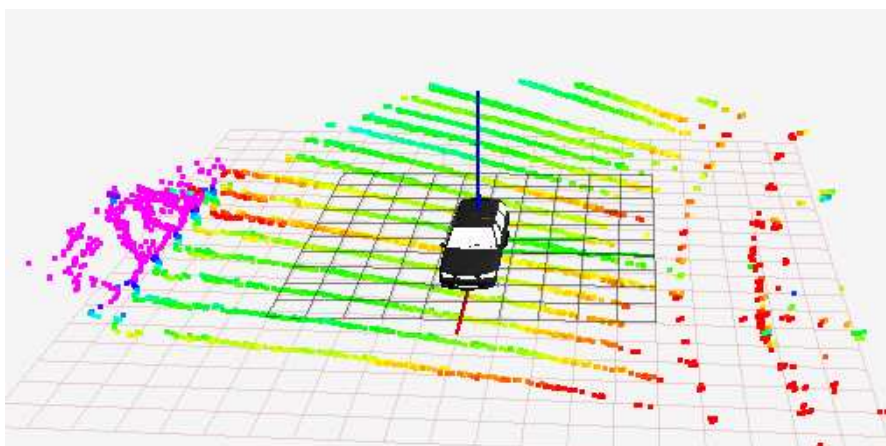


Figura 4.15: Reconstrução com *AtlasCar* a curvar - Situação: Com transformação do carro em relação à estrada.

De acordo com a equação 4.1 determina-se a percentagem dos pontos que se encontram dentro do intervalo definido.

$$\%P = \frac{n}{t} \times 100 \quad (4.1)$$

- $P$  - percentagem de pontos da porção dentro do intervalo definido
- $n$  - número de pontos da porção dentro do intervalo
- $t$  - número total de pontos da porção

A análise de seguida foi realizada com o *AtlasCar* numa secção reta. Nesta secção o *AtlasCar* encontra-se numa fase inicial a velocidade constante ( $\Delta t = [0; 3.9]s$ ) sendo seguida de uma travagem ( $\Delta t = [4.0; 5.5]s$ ) e arranque ( $\Delta t = [9; 11]s$ ).

O gráfico 4.17 permite mostrar o comportamento do sistema desenvolvido para a situação descrita atrás. As diferentes curvas representadas permitem verificar a percentagem

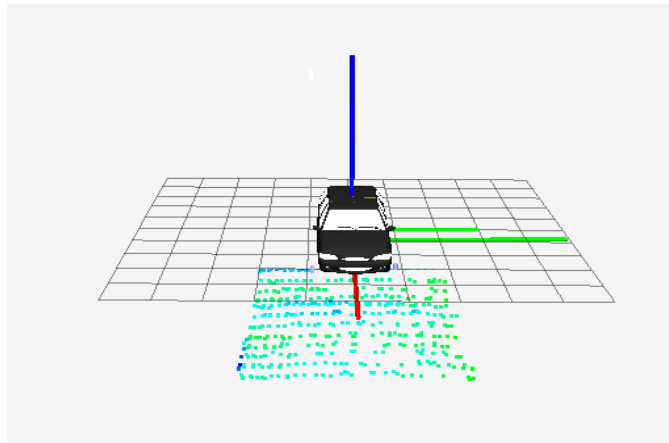


Figura 4.16: Representação da nuvem de pontos em análise.

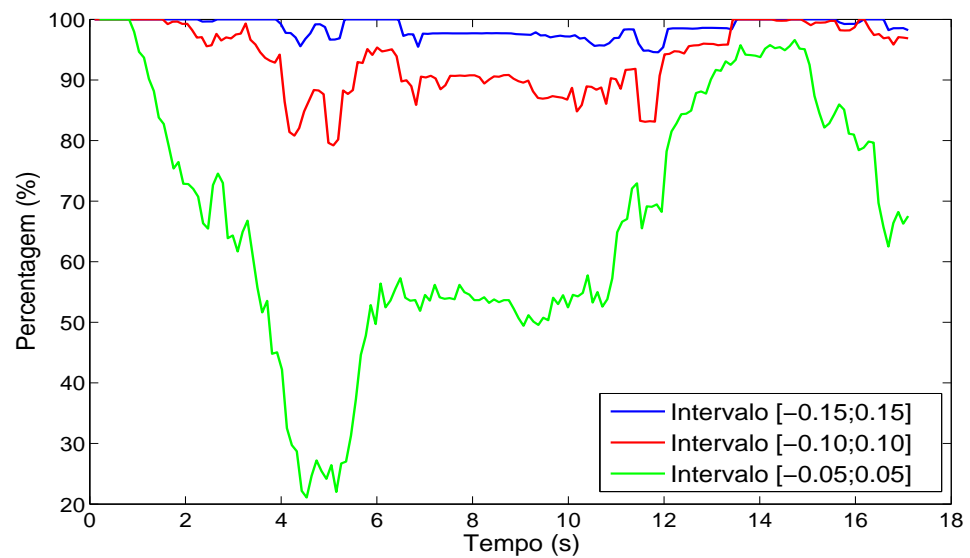


Figura 4.17: Representação da variação da percentagem de pontos presentes em cada intervalo definido.

de pontos que se encontram presentes na nuvem de acordo com cada intervalo definido, como se pode observar na legenda da figura.



## Capítulo 5

# Conclusão e Trabalho Futuro

O desenvolvimento de um método para a construção de um mapa local com recurso a um *laser* 2D foi conseguido com sucesso. Para o desenvolvimento deste método foi necessário abordar diversas áreas e melhorar algum do trabalho já desenvolvido. O ponto chave para a construção de uma nuvem de pontos fiável, foi a caracterização dinâmica do *AtlasCar*. Para tal, foi necessário melhorar o cálculo da posição do *AtlasCar* e o desenvolvimento de um módulo para a monitorização do seu estado, relativamente ao espaço envolvente.

Apesar de ainda haver certos aspetos a melhorar no cálculo da posição, a intervenção levada a cabo, com a introdução da informação do sensor inercial, permitiu manter a orientação do carro constantemente monitorizada. Este método de cálculo ainda possui algumas limitações dada a acumulação de erros durante a integração da velocidade.

O desenvolvimento do sistema de monitorização do *AtlasCar* relativamente ao espaço envolvente revelou ser um dos pontos mais importantes na construção de uma nuvem de pontos que permita fornecer informação útil sobre o estado do ambiente. Inicialmente este sistema foi pensado para fazer a distinção entre a inclinação do carro em relação à estrada e a inclinação da estrada em relação ao mundo. No entanto, verificou-se que a utilização conjunta dos valores de inclinação medidos pelo sensor inercial e os valores de inclinação determinados pelos sensores *Sharp* não era possível, dado que o sensor inercial não distingue aceleração da gravidade, das acelerações inerciais do movimento do veículo. Devido a este aspeto, optou-se pela utilização somente da informação obtida pelos sensores *Sharp*. De forma a permitir a utilização do sensor inercial, o desenvolvimento de um módulo que calcule as acelerações a que um veículo está sujeito e faça a fusão desses dados com os do sensor inercial, iria permitir fazer a distinção entre as acelerações, permitindo que a determinação das orientações se tornassem independentes do movimento do *AtlasCar*.

O método desenvolvido permite a obtenção do mapa local de forma dinâmica, i.e., só é representado o mapa da vizinhança de acordo com a dimensão definida no módulo responsável pela construção da nuvem de pontos. Esta nuvem poderá servir para a obtenção de informação sobre obstáculos (positivos/negativos) que se encontrem na trajetória do *AtlasCar*. No entanto para que seja possível obter este tipo de informação sempre de forma correta, deverão ser adotados métodos bastante mais robustos dos que foram implementados neste trabalho. Uma alternativa interessante aos sensores *Sharp* instalados, seria a utilização de *lasers* colocados ao longo de toda a parte inferior do *chassis* para monitorizar o seu comportamento. Estes dispositivos permitiriam uma precisão de dados bastante superior e uma densidade de medições que permitisse observar uma tendência,

descartando irregularidades da estrada, que no caso do sistema implementado pode afetar bastante os dados recolhidos.

Uma solução bastante mais robusta que os sistemas desenvolvidos neste trabalho, no entanto bastante dispendiosa, seria a instalação do *Applanix POS LV*. Este sistema permitiria o cálculo da posição e orientação do veículo de forma bastante fiável mesmo em condições bastante adversas onde os demais sistemas descritos neste trabalho falhariam. Essa fiabilidade é assegurada pela existência de fontes de informação redundantes e complementares que permitem compensar as limitações de cada dispositivo e devido ao software desenvolvido responsável pela fusão da informação recolhida pelos diversos sensores.

Uma alternativa ao método desenvolvido para construção do mapa local, seria a utilização de um laser 3D que mostrasse o espaço envolvente de forma rápida e em tempo real.

# Bibliografia

- [Almeida and Santos, 2012] Almeida, J. and Santos, V. M. (2012). Real time egomotion of a nonholonomic vehicle using LIDAR measurements. *Journal of Field Robotics, Submitted*.
- [Atlas, 2012] Atlas (2012). Atlas project : <http://atlas.web.ua.pt>.
- [Bailey, 2002] Bailey, T. (2002). *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, University of Sydney, Australian Centre for Field Robotics.
- [Buehler et al., 2007] Buehler, M., Lagnemma, K., and Singh, S. (2007). The 2005 DARPA grand challenge : The great robot race. *Springer Publishing Company, Incorporated*, 1 st ed.
- [Coulombe and L., 2002] Coulombe, P. and L., C. (2002). Vehicle yaw, pitch, roll and 3D lane shape recovery by vision. In *Proceedings of IEEE Intelligent Vehicles Symposium*.
- [Google, 2012] Google (2012). *Google Earth*. <http://www.google.com/earth/index.html>.
- [Hokuyo Automatic CO., 2009] Hokuyo Automatic CO., L. (2009). *Scanning Laser Range Finder*.
- [Holz et al., 2010] Holz, D., Droschel, D., Behnke, S., May, S., and Surmann, H. (2010). Fast 3D Perception for Collision Avoidance and SLAM in Domestic Environments. In Barrera, A., editor, *Mobile Robots Navigation*, pages 53–84. IN-TECH Education and Publishing, Vienna, Austria.
- [Joe et al., 2001] Joe, M. M., Hutton, J., Reid, B., and Hill, R. (2001). Gps/imu products – the applanix approach.
- [Labayrade and Aubert, 2003] Labayrade, R. and Aubert, D. (2003). A single framework for vehicle roll, pitch, yaw estimation and obstacles detection by stereovision. In *Proceedings. IEEE Intelligent Vehicles Symposium*.
- [Latombe, 1991] Latombe, J. (1991). Robot motion planning. *Boston : Kluwer Academic Publishers*.
- [Nourani-Vatani et al., 2009] Nourani-Vatani, N., Roberts, J., and Srinivasan, M. (2009). IMU aided 3D visual odometry for car-like vehicles. Australasian Conference on Robotics and Automation.

- [Parallax, 2000] Parallax, I. (2000). *Sharp GP2D12 Analog Distance Sensor*.
- [Quigley et al., 2009] Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibes, J., Berger, E., and Wheeler, R. (2009). ROS: an open-source robot operating system. *ICRA Workshop on Open Source Software*.
- [Rusu and Cousins, 2011] Rusu, R. B. and Cousins, S. (2011). 3d is here: Point cloud library (pcl). In *IEEE International Conference on Robotics and Automation (ICRA)*.
- [Ryde and Hu, 2007] Ryde, J. and Hu, H. (2007). Mobile robot 3D perception and mapping with Multi-Resolution occupancy lists. *Proc. of IEEE International Conference on Mechatronics and Automation*.
- [Sappa et al., 2007] Sappa, A., Gerónimo, D., Dornika, F., and Lopez, A. (2007). Stereo vision camera pose estimation for On-Board applications. Technical report, University of Barcelona, Computer Vision Center and Autonomus.
- [Siadat et al., 1997] Siadat, A., Kaske, A., Klausmann, S., Husson, R., and DUFAUT, M. (1997). An optimized segmentation method for a 2D Laser-Scanner applied to mobile robot navigation. In *In Proceedings of the 3rd IFAC symposium on intelligent components and instruments for control applications*.
- [Weinstein and Moore, 2010] Weinstein, A. and Moore, K. (2010). Pose estimation of ackerman steering vehicles for outdoors autonomous navigation. *Proceedings of 2010 IEEE International Conference on Industrial Automation*.
- [Welch and Bishop, 1995] Welch, G. and Bishop, G. (1995). *An Introduction to the Kalman Filter*. University of North Carolina at Chapel Hill, Chapel Hill, NC.
- [Xsens, 2009] Xsens, T. (2009). *MTi and MTx User Manual and Technical Documentation*.
- [Zalzal, 2012] Zalzal, V. (2012). The kfilter project <http://kalman.sourceforge.net>.