Universidade de Aveiro Departamento de Engenharia Mecânica
2012

André João Lopes Oliveira

Sistema de Monitorização da Condução de um Automóvel

André João Lopes Oliveira

Sistema de Monitorização da Condução de um Automóvel

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Mecânica, realizada sob orientação científica do Doutor Vítor 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 Associado da Universidade de Aveiro
Vogais / Committee	**Prof. Doutor Vítor Manuel Ferreira Dos Santos** Professor Associado da Universidade de Aveiro (orientador)
	Prof. Doutor António Manuel Ferreira Mendes Lopes Professor Auxiliar da Faculdade de Engenharia da Universidade do Porto

Agradecimentos / Acknowledgements

Ao Professor Doutor Vítor Santos pela sapiência, disponibilidade e coragem transmitida em todos os momentos, pela sua excelente orientação, um verdadeiro líder de equipa.

Um abraço a todos os guerreiros do L.A.R. pelo companheirismo e espirito destemido, sempre presente ao longo desta jornada.

Aos amigos e amigas, presença constante nesta magnífica epopeia, que foi o meu curso de Engenharia Mecânica na Universidade de Aveiro.

Obrigado por tudo Américo, Conceição e Adriana. Foram cinco anos espetaculares...

"*Parece fácil!*"

Palavras-chave Monitorização da condução, medição de velocidade, AtlasCar, tempo de colisão, situações de risco, reconhecimento de condutor, Arduíno.

Resumo

O veículo AtlasCar é um protótipo desenvolvido no Laboratório de Automação e Robótica do Departamento de Engenharia Mecânica da Universidade de Aveiro, que tem como objetivo fomentar a investigação na área da condução autónoma.

Os sistemas desenvolvidos neste trabalho têm como objetivo auxiliar o condutor do AtlasCar, em situações de condução perigosa, assim como reconhecer o condutor quando realiza uma certa manobra de condução.

De modo a monitorizar a condução do AtlasCar, foram desenvolvidos e instalados sistemas hardware para a aquisição de dados dentro do habitáculo do veiculo.

A monitorização pode ser feita em tempo real ou à posteriori; em qualquer das situações é possível analisar a informação. Após a aplicação de algoritmos é mostrado um alerta sobre a situação detetada. A longo prazo, este sistema tem como objetivo ajudar o condutor a corrigir os seus erros ou deficientes práticas de condução.

Keywords Driving Monitoring, Speed Measurement, AtlasCar, Collision Time, Risk Situations, Driver Recognition, Arduíno.

Abstract

The AtlasCar vehicle is a prototype developed at the Laboratory of Automation and Robotics Department of Mechanical Engineering in University of Aveiro, whose objective is promoting research in autonomous driving.

The systems developed in this work are intended to assist the driver of the AtlasCar in situations of dangerous driving, as well to recognize the driver when performing a certain driving maneuver.

In order to monitor the AtlasCar driving, hardware systems for data acquisition have been developed and installed in the passenger compartment of the vehicle.

The monitoring can be done in real time or offline, in both situations it is possible to analyze the information, where after the application of algorithms is shown a warning signal about the situation practiced. In the long run, this system aims to help the driver to correct their bad driving practices.

Conteúdo

Lista de Tabelas

Lista de Figuras

Capítulo 1

Enquadramento

Neste primeiro capítulo será feita uma pequena introdução ao Projeto Atlas, como uma breve descrição aos sistemas de ajuda ao condutor existentes.

1.1 Projeto Atlas & AtlasCar

O projeto Atlas visa o desenvolvimento de soluções na área da robótica móvel, aplicada a sistemas de condução autónoma e de ajuda à condução. Este projeto foi iniciado em 2003, com a participação da equipa no Festival Nacional de Robótica, na prova de condução autónoma, onde ficou em quarto lugar, tendo vindo desde 2006 a vencer a prova [Project, 2011]. Na figura 1.1 são apresentados dois protótipos de robôs Atlas.

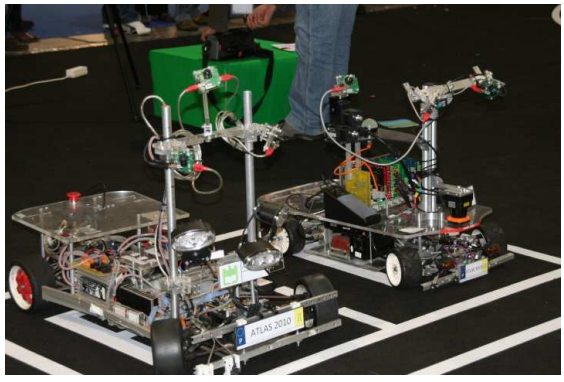

Figura 1.1: Atlasmv e Atlas Série 2000.

Depois do sucesso obtido com os vários robôs Atlas em pista e em ambiente controlado, surgiu o desafio de desenvolver um protótipo à escala real para conseguir lidar com

as exigências reais da condução: o AtlasCar. Trata-se de um protótipo de um carro autónomo, que tem como objetivo promover o desenvolvimento de novas soluções na área da condução autónoma e da robótica móvel. O AtlasCar encontra-se equipado com sensores que permitem a obtenção de dados para posterior análise. O conjunto de sensores inclui três lasers planares, um laser 3D, uma câmara estéreo, uma unidade inercial, um recetor GPS e um encoder para medição de velocidade. A "centralina" do carro é composta por dois PLC's um responsável pelo controlo do carro e outro pela ativação dos sensores do AtlasCar.

Figura 1.2: AtlasCar, modelo Ford Escort SW equipado com sensores.

1.2 Objetivos

Pretende-se com este trabalho desenvolver um módulo para deteção de algumas situações de risco durante a condução. Este módulo será capaz de mostrar ao condutor quais as situações de risco em que esteve envolvido durante a prática da condução. O módulo de situações de risco criará um ficheiro em formato de texto com toda a informação necessária para que possam ser revistas à posteriori as manobras de risco em que o condutor foi notificado.

A segunda fase deste trabalho compreende o desenvolvimento de um módulo de software capaz de reconhecer o condutor do veículo, de entre os possíveis condutores do mesmo. A manobra utilizada para a deteção do condutor foi o estacionamento em paralelo, pois é uma manobra caraterística a todos os condutores e onde é possível distinguir

as diferentes fases da mesma.

Para que estes sistemas sejam aplicados no AtlasCar, é necessário criar um sistema capaz de fazer a monitorização aos indicadores de mudança de direção, buzina, sinal de ignição, luzes e pedais. Outras variáveis necessárias para estes sistemas funcionarem são os obstáculos em redor. As rotações do motor e o ângulo da direção, já se encontram monitorizadas pelo PLC .

A velocidade do AtlasCar é uma variável já publicada pelo PLC. No entanto, as medições têm-se verificado incorretas, sendo necessário encontrar uma solução para melhorar a leitura de velocidade.

1.3 Estado da Arte

Nesta secção é feita uma pequena abordagem a alguns dos sistemas existentes.

1.3.1 Modelo de comportamento do condutor

O principal objetivo quando conduzimos é alcançar o nosso destino, para isto o nosso foco durante o exercício da condução é evitar colisões. O modelo T.C.I. - Task Capability Interface apresentado por Fuller [Fuller, 2005], tem por base a capacidade que um condutor tem em resolver os desafios apresentados ao longo da condução.

Do lado dos desafios da condução são apresentados o meio ambiente, os outros condutores, o nosso veículo, a velocidade, a posição do mesmo na via, a sua trajetória e o fator humano do condutor.

A capacidade em resolver estes desafios reside no treino que se tem em conduzir e na competência do condutor.

Desta forma quando os desafios apresentados ultrapassam a capacidade do condutor para os resolver, este nem sempre se vê envolvido num acidente, pois ainda existem ações compensatórias de outros condutores que evitam o acidente.

Fuller também defende que a capacidade do indivíduo ao longo da condução não é sempre igual, e tende a diminuir. Caso os desafios apresentados sejam também reduzidos: por exemplo, numa viagem longa onde a sonolência se vai instalando, pode levar ao adormecimento do condutor.

Uma das propostas para melhorar a capacidade do condutor é a diminuição da velocidade porque esta está diretamente relacionada com o desafio da condução.

1.3.2 Sistemas A.D.A.S. - Advanced Driver Assistance System

Os sistemas A.D.A.S. têm como objetivo reduzir ou eliminar os erros do condutor, promovendo a uma condução eficaz [Cao et al., 2010], a intervenção destes sistemas pressupõe que o condutor seja informado da sua ativação [Association, 2011] [Continental, 2011]. Destes sistemas podemos destacar aqueles que recolhem informação sobre o estado do veículo em tempo real e em caso de falha por parte do condutor entram em controlo do carro, como é o caso do programa eletrónico de estabilidade (ESP), nos travões através do sistema anti-bloqueio das rodas [Works, 2011], nos sistemas de velocidade cruzeiro [Works, 2011]. Na figura 1.3 está representado o funcionamento de um sistema que informa o condutor dos veículos que circulam em ângulo morto. O sistema de estacionamento presente na figura 1.5, ainda em fase de testes, permite estacionar o carro através de uma aplicação no telemóvel do proprietário do veículo, sem que para isso este necessite de estar dentro do mesmo. Exemplos mais recentes destes sistemas implementados em carros de série são o estacionamento automático (MERCEDES) com ajuda do condutor, a deteção de limite de velocidade (GARWIN), o seguimento do carro da frente (BOSH), como futuramente a comunicação entre os vários veículos, tendo em vista a construção de uma rede que controle todo o tráfego [Works, 2011]. A figura 1.6 diz respeito a um sistema que funciona em percursos de longo curso, onde os objetivos principais são a redução de emissões de CO2, a economia de combustível e o conforto por parte do condutor que pode fazer outra tarefa enquanto circula nestes "comboios"de carros.

A forma como estes sistemas de deteção de risco devem operar é proposta por [Cao et al., 2010] num estudo onde se chega à conclusão que para os cinco tipos de situações reconhecidas a reprodução de um *beep* seguido pela apresentação de um alerta visual é a melhor combinação para obtenção de resultados por parte do condutor. De notar que esta combinação oferece ao condutor uma liberdade na perceção dos avisos.

Existem também sistemas A.D.A.S. que conseguem detetar antecipadamente a fadiga do condutor [Krajewski et al., 2008], estilo de condução [Johnson and Trivedi, 2011], previsão do comportamento [Kuge et al., 2000b] e identificação de algumas situações durante a condução [Mitrovic, 2005], o bom funcionamento destes sistemas tem por base uma grande quantidade de dados adquiridos ao longo de várias viagens.

A recolha de dados nalguns destes sistemas é feita através de sistemas de visão, como o caso da monitorização da posição dos pés do condutor ao longo de uma viagem [Tran et al., 2012]. A utilização deste sistema com base em visão não deteta a força

Figura 1.3: Sistema de auxílio ao condutor, quando outro veículo se encontra em ângulo "morto". (BOSH)

aplicada em cada um dos pedais, pois o objetivo deste trabalho era a previsão da posição do pé do condutor [Tran et al., 2012].

Outro sistema de visão implementado em [Krajewski et al., 2008] tem por base a análise do ângulo da direção de um veículo, para que através deste consigam extrair o estado de fadiga do condutor.

Os equipamentos utilizados para deteção das diferentes manobras de condução são o GPS, acelerómetros e giroscópios [Mitrovic, 2005].

Um dos avanços recentes é a utilização de equipamentos do nosso quotidiano, como o caso do telemóvel, para recolha de dados durante a condução [Johnson and Trivedi, 2011]. A utilização deste equipamento em específico deve-se à existência de modelos que já têm equipamento GPS e acelerómetros incluídos, assim como o facto de ser proibida a utilização do telemóvel durante a condução.

Esta quantidade de dados é tratada com base em *Hidden Markov Models* (H.M.M.) [Johansson and Olofsson, 2007] e *Dynamic Time Warping* (D.T.W.) [Ko et al., 2008]. Desta forma os sistemas contêm bases de referência para cada uma das situações a detetar, para que possam ser comparadas com a informação recolhida em tempo real e a informar o condutor das situações de risco antes destas sequer ocorrerem, [Tran et al., 2012], [Mitrovic, 2005], [Kuge et al., 2000b], [Krajewski et al., 2008], [Kuge et al., 2000a] e [Johnson and Trivedi, 2011].

A maioria dos módulos apresentados utiliza sistemas de visão para aquisição de dados pretendidos, esta opção apresenta maior poder computacional. A outra opção apresen-

Figura 1.4: Sistema de suspensão ativa implementado pela primeira vez em competição 1992, no carro FW14B da equipa Williams F1.

Figura 1.5: Estacionamento automático comandado por aplicação java no telemóvel. (VALEO)

tada prende-se com a aquisição de dados através da "centralina"do carro, opção não disponível no AtlasCar.

Daí que o trabalho tenha como objetivo o desenvolvimento de Hardware específico para a recolha dos dados a utilizar, pelas razões já referidas. Como descrito na secção 1.2 os sistemas a desenvolver não têm como objetivo prever as situações futuras de situações de risco de colisão, mas sim deteta-las quando estas ocorrem, daí que não venham a ser

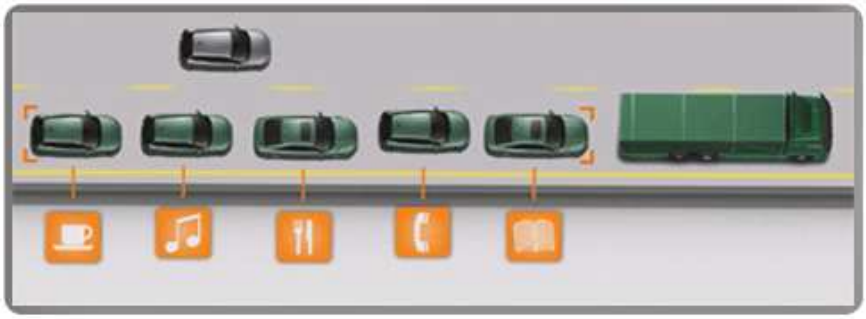

Figura 1.6: Representado sistema que permite ao condutor relaxar durante a condução. (projeto SARTRE)

utilizados os modelos de D.T.W. e H.M.M., para previsão das situações de risco.

Capítulo 2

Sistemas para Monitorização

2.1 Grandezas a monitorizar

Para fazer a monitorização da condução é necessário conhecer o estado das diferentes variáveis no interior do habitáculo do AtlasCar, isto é, todo o tipo de manípulos ou botões fundamentais ao exercício da condução. Desta forma podemos contabilizar os três pedais do condutor, os indicadores luminosos (piscas e luzes) e a buzina. Outras variáveis relevantes para análise são a velocidade e os obstáculos existentes no espaço exterior do veículo.

2.2 Hardware e software adotados

2.2.1 Controlo do AtlasCar

No caso específico do projeto AtlasCar, a "centralina" é composta por dois PLC's. O primeiro é responsável pela atuação e gestão dos diferentes módulos do carro nos modos automático e manual. O segundo, tem como tarefa o controlo da alimentação de cada sensor presente no AtlasCar, sendo possível ligar e desligar os diferentes sensores do AtlasCar fisicamente, através do uso de interruptores ou por ordem do computador.
O primeiro PLC permite o controlo dos seguintes atuadores do veículo:

- Borboleta do carburador e volante do carro - este efetua a leitura do sinal analógico proveniente de um potenciómetro colocado no eixo de rotação de cada sistema, isto é, na borboleta do carburador e no volante do carro. Quando é necessário acelerar ou virar, o PLC tem saídas que atuam o servo-motor da borboleta do carro ou a

coluna de direção. De notar que a atuação destes dois sistemas é executada de forma independente.

- Pedal do travão, embraiagem e travão de mão - para estes módulos de hardware são enviados sinais digitais do PLC e estes atuam nos respetivos componentes. O controlo de velocidade e a chegada ao fim de curso de cada módulo, são controladas por um microcontrolador PIC, presente em cada sistema. De notar que não é recebida nenhuma mensagem no PLC com a confirmação de que cada módulo atingiu o seu fim de curso.

- Ignição e Sinais luminosos do carro - para ligar estes módulos no carro o PLC atua em relés que ligarão o respetivo módulo. Não existe nenhum equipamento a monitorar o estado destes sistemas caso a atuação seja externa ao PLC, como por exemplo o próprio condutor.

O Segundo PLC, serve exclusivamente para controlo da alimentação dos sensores do carro possibilitando assim uma melhor gestão da energia dispendida. É ainda possível desligar cada um dos sensores, quando assim se entender, através dos interruptores localizados na bagageira ou pelo envio de uma mensagem, através do computador.

A rede de comunicação de dados no veículo, pode ser visualizada na figura 2.1. Podemos reparar que a maioria dos equipamentos sensoriais utilizados no carro estão conetados por protocolo Ethernet, através de um Switch (0). Sendo o caso dos dois Arduínos (2, 3), três sensores laser (4, 5, 12) e dos dois PLC's (6, 13). O outro tipo de ligação utilizado no carro é USB, sendo feita uma ligação direta ao computador (11), mais propriamente o laser scan (8), o recetor GPS (10), o recetor do comando da Xbox (9) e o sensor inercial (1). Para o caso da câmara estéreo (7), a comunicação utilizada é Firewire.

2.2.2 Placa microchip

O presente trabalho, como já referido na secção 2.1, pretende a monitorização de alguns dos atuadores do AtlasCar. Uma vez que o PLC não monitoriza os sistemas pretendidos surge a necessidade de desenvolver um sistema hardware capaz de monitorar o estado das variáveis relevantes para o trabalho.

Tendo em vista a aquisição das variáveis descritas na seção 2.1 foi então desenvolvida uma placa para monitorizar os sinais provenientes do carro, o protótipo que pode ser

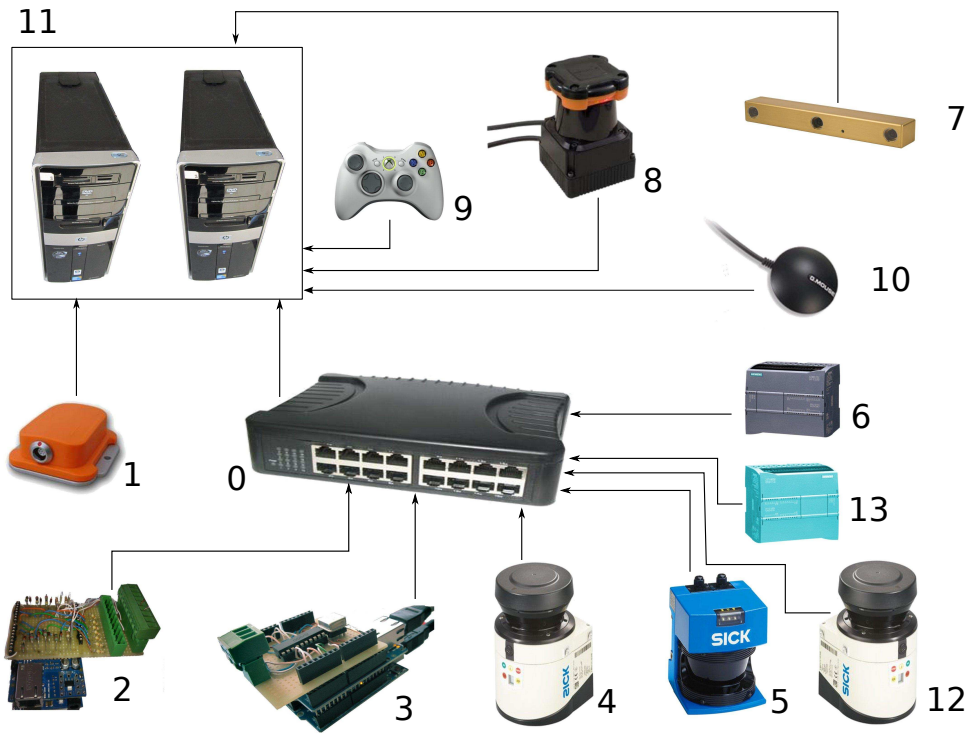

Figura 2.1: Rede local utilizada no AtlasCar.

visto na figura 2.2. Este protótipo utiliza um microcontrolador PIC18F258 da Microchip, juntamente com um conversor max232 que comunica por porta série com o computador. De notar que este hardware permite a comunicação com o utilizador através de uma interface gráfica. O funcionamento do protótipo e da interface gráfica em ambiente GTK pode ser visto no vídeo online [Oliveira, 2011].

O sistema presente na figura 2.2, revelou problemas de fiabilidade, com algumas falhas na comunicação quando submetido a testes durante várias horas. Alguns destes problemas foram resolvidos. Porém a necessidade de avançar com uma solução viável para o projeto, ditou a escolha de uma plataforma mais fiável e robusta, o Arduíno™ [Arduíno, 2012].

2.2.3 Arduíno™

O Arduíno™ é uma plataforma *opensource* que utiliza microcontroladores de 8 bits da ATmega. Esta plataforma permite a construção de projetos de controlo e monitorização de uma forma simples. A quantidade de suporte *online* existente é maior que no caso do PIC, existindo bibliotecas para fazer a comunicação com hardware. O Arduíno é

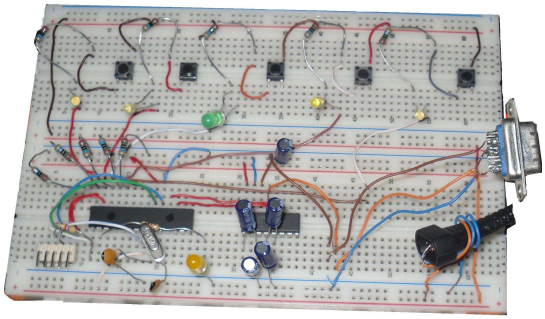

Figura 2.2: Protótipo elaborado para testes.

utilizado nos mais variados projetos de eletrónica, exemplo de um robot que recolhe bolas de ténis [Wang, 2012], monitorização de um sistema para saltar a corda [Yao et al., 2011] ou a monitorização de sistemas pneumáticos [Field et al., 2011]. A estrutura do código utilizado no Arduíno, em comparação com a programação de um PIC, é mais simplista, pois as configurações de cada um dos pinos do microcontrolador é feita através das bibliotecas internas do programa Arduíno, poupando assim ao utilizador tempo na sua configuração. De referir ainda que esta plataforma de trabalho está disponível para Linux, Windows e MacOS.

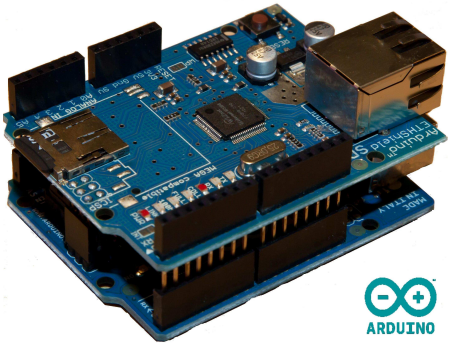

Figura 2.3: Arduíno Uno.

Como já referido, o PLC detém o controlo sobre os atuadores do carro e a monitorização de algumas variáveis, por este motivo é imperativo que todas as mensagens de estado que este envia para o computador não contenham atrasos ou valores em falta, o que não tem sido sempre o caso. Aparentemente a causa deste problema é a sobrecarga do sistema, com pedidos de comunicação. Pelas razões referidas, tornou-se fundamental

desenvolver módulos de hardware que se dediquem a uma função especifica, a solução de hardware encontrada foi o Arduíno.

2.2.4 *Shield* de ethernet

Para comunicar com o Arduíno foi utilizada a comunicação *TCP/IP*, visto que o ambiente no interior do AtlasCar induz algum ruído elétrico nos sinais. A utilização da porta USB seria problemática devido a distâncias superiores a 5 metros entre o Arduíno e a entrada usb no computador, correndo o risco de perder sinal, como se veio a comprovar com testes realizados no carro.

2.2.5 R.O.S. - Robot Operating System

No projeto AtlasCar é utilizada a arquitetura *R.O.S.* [Quigley et al., 2009]. Esta está organizada em *packages*, isto é, uma pasta principal onde podem existir vários *nodes*. Os *nodes* são instanciações de um ficheiro binário, o que permite a possibilidade de ter vários *nodes* lançados pelo mesmo binário. Este tipo de organização facilita a compreensão e organização do código. A comunicação entre *nodes* é possível através da subscrição e publicação de mensagens, os chamados *topics*. Caso seja necessário gravar informação dos *topics* do sistema operativo, podemos utilizar uma ferramenta chamada *rosbag*, responsável pela gravação e repetição de ficheiros do tipo *.bag* que contêm a informação gravada. Para analisar melhor as diferentes situações de risco é utilizada uma ferramenta chamada Rviz disponível em ambiente ROS, esta ferramenta permite ao utilizador visualizar as nuvens de pontos adquiridas aos obstáculos em redor do veículo, quando é feita a revisão dos dados gravados. Os conceitos referidos têm muita importância pois serão invocados por várias ocasiões ao longo do texto.

2.3 Placa de aquisição de dados

O sistema monitoriza o "pisca" direito e esquerdo, quatro "piscas", médios, máximos, buzina e pedais do condutor do AtlasCar em modo manual e automático (Figura 2.4). Para o desenvolvimento da placa de aquisição de dados foram tidos em conta os diferentes níveis elétricos nas entradas digitais do Arduíno que funcionam a 5 volts, e os sinais do carro que provêm da bateria a 14 volts. Os sinais a monitorizar estavam junto à coluna de direção do carro. Depois de identificados, foram acoplados a um cabo que os transporta

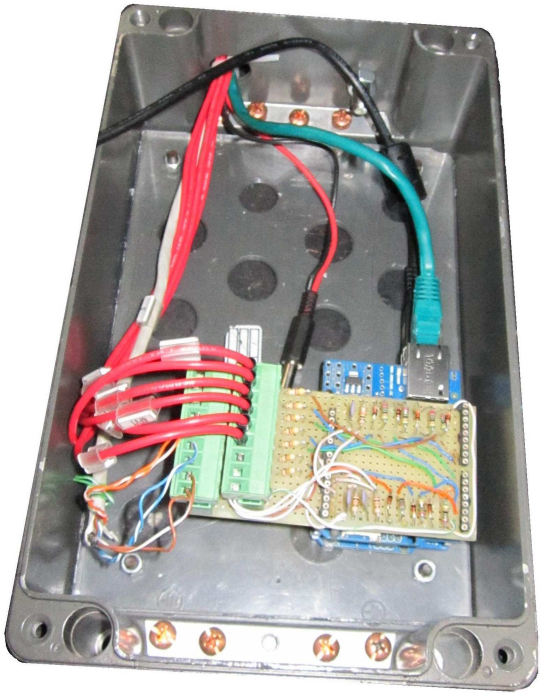

Figura 2.4: Instalação do Arduíno ™ responsável pela monitorização da condução.

até ao Arduíno colocado debaixo do banco lateral esquerdo do condutor. O esquema elétrico da placa está presente no capítulo 6, ou para uma melhor compreensão, a figura 2.7 contém representadas as portas utilizadas pelo Arduíno. A alimentação do Arduíno é efetuada por uma fonte de tensão contínua instalada na bagageira, que fornece 12 V.

2.3.1 Instalação de sensores de força nos pedais

Nos pedais do condutor foram instalados sensores de força resistivos (Figura 2.5), que permitem medir a força aplicada pelos pés do condutor. Este tipo de sensor funciona com uma tensão aplicada de 5 V no sensor que irá variar à entrada do microcontrolador consoante a força aplicada ao sensor. A escolha destes sensores deve-se ao seu baixo custo e volume ocupados depois de instalados, quando comparados a um equipamento especializado [Transducer, 2012], que utiliza uma célula de carga (Figura 2.6). Os valores de tensão provenientes destes sensores são proporcionais à força aplicada nestes. A mensagem de estados dos diferentes atuadores do habitáculo do AtlasCar pode ser vista na tabela 2.1.

Figura 2.5: Sensor instalado em cada pedal do AtlasCar. (Sensitronics)

Figura 2.6: Sistema baseado numa célula de carga para medição da força aplicada no pedal.

2.3.2 Programação do Arduíno da monitorização

O Arduíno da monitorização utiliza 7 entradas digitais e os pinos utilizados estão ilustrados na figura 2.7. É também ilustrada a utilização de três entradas analógicas, que correspondem a cada sensor instalado nos pedais do veículo.

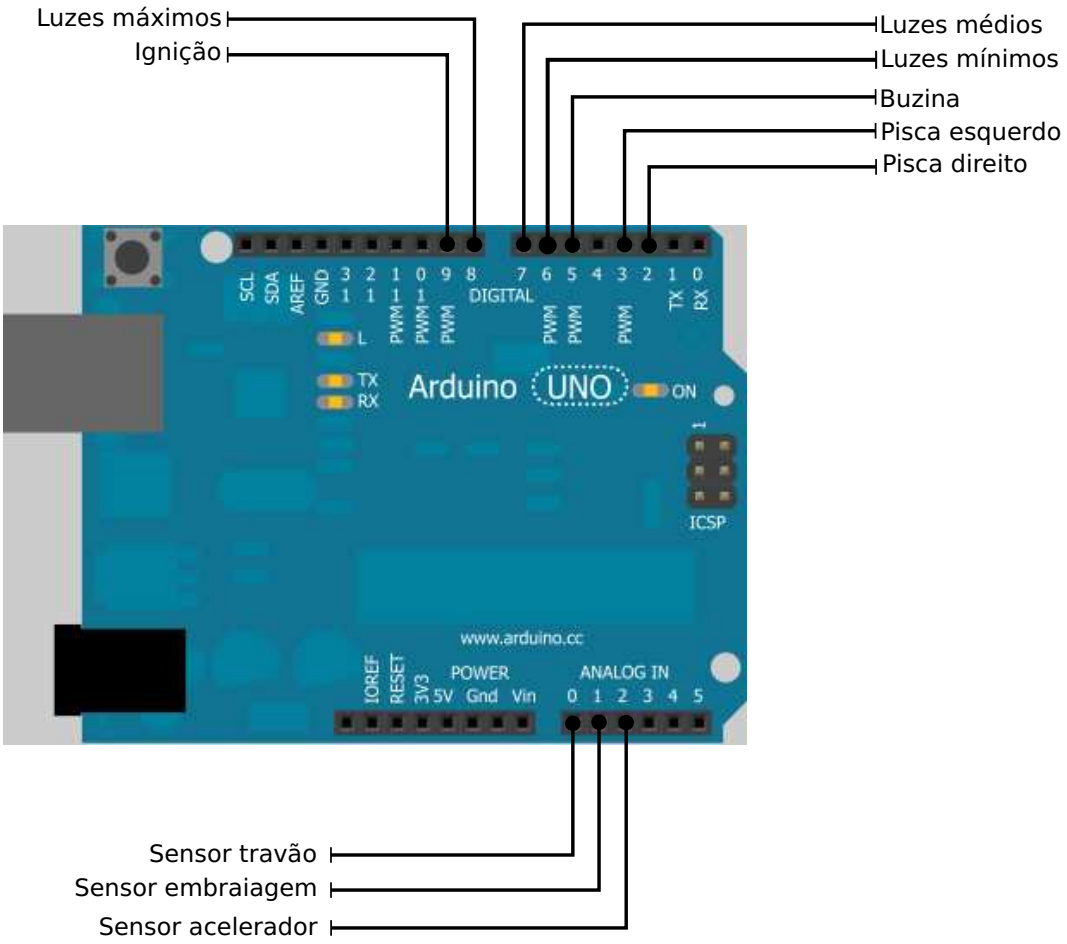

Figura 2.7: Esquema das portas utilizadas pelo Arduíno responsável pela monitorização.

Na figura 2.8 pode-se visualizar o esquema com a organização do código do programa implementado no Arduíno. Quando este liga, são configuradas as portas digitais e as analógicas que receberem os diferentes sinais assim como o protocolo Ethernet usado para comunicar com o equipamento.

Depois da configuração inicial, o módulo hardware entra num ciclo onde verifica se existe um cliente conectado por cabo ethernet e caso seja verdadeiro, envia uma mensagem com os campos descritos na tabela 2.1, contendo o estado de todas as variáveis monitorizadas.

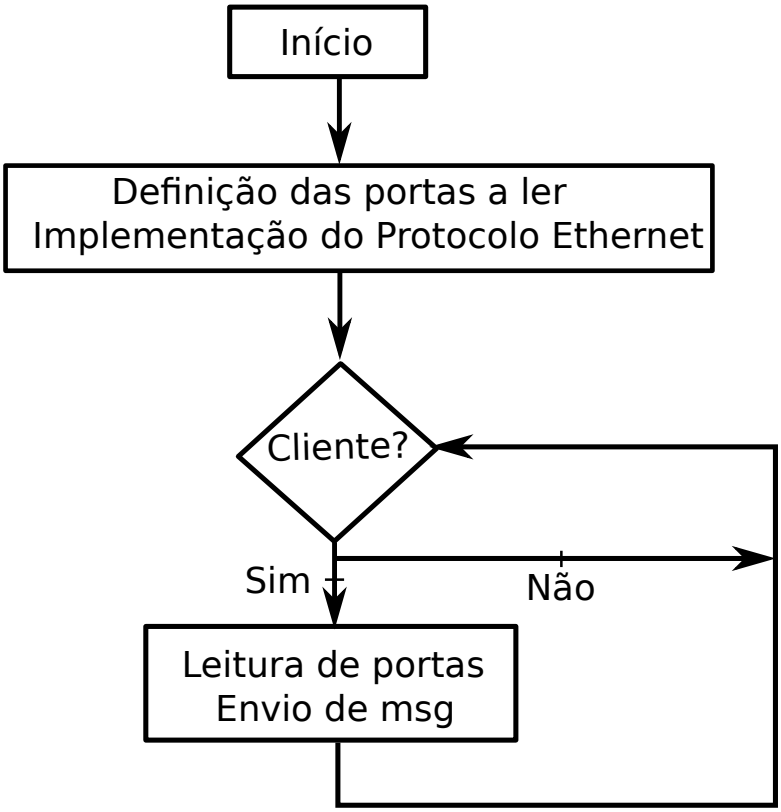

Figura 2.8: Diagrama do código utilizado para programar o Arduíno responsável pela monitorização.

Endereço de IP do Arduíno	10.0.0.30
Estrutura da mensagem	**Descrição das variáveis**
'0x02'	início de mensagem
00000000	estado das variáveis
T000	valor do sensor do acelerador
C000	valor do sensor da embraiagem
B000	valor do sensor do travão
'0x03'	fim da mensagem

Tabela 2.1: Formato da mensagem enviada com o estado do habitáculo do AtlasCar

2.4 Placa de medição da velocidade

A medição da velocidade do AtlasCar era feita inicialmente pelo PLC através de um contador de alta velocidade interno. Os contadores de alta velocidade, tal como o nome indica contam pulsos de um *encoder* a uma taxa muito superior ao ciclo do programa principal. O seu funcionamento pretende que os três canais do *encoder* sejam conetados às entradas do contador que incrementa uma unidade à contagem por cada vez que este recebe um pulso. O sistema de cálculo da velocidade do carro conta os pulsos de um *encoder* incremental [RS, 2011] com resolução de 50 pulsos por volta; este sistema já estava implementado fisicamente no carro, como pode ser visto na figura 2.9. Como referido na subsecção 2.2.3, a passagem da leitura da velocidade para um Arduíno impôs a necessidade de encontrar um contador de alta velocidade compatível com o Arduíno Uno. Essa escolha recaiu num contador de quadratura *HCTL-2022* devido ao limitado número de portas disponíveis no Arduíno, bem como a existência de uma biblioteca para comunicar com este contador que, embora fosse para o modelo Arduíno Mega, mostrou ser fácil de alterar [AVAGO, 2012].

As caraterísticas deste contador são apresentadas na tabela 2.2.

Caraterísticas	
Pinos	20
Nº de Bits	32
Tipo de contador	Up / Down e contagem
Número de eixos	1
Modo de contagem	4x
Frequência de operação	2 a 33 MHz

Tabela 2.2: Caraterísticas do contador de quadratura *HCTL-2022*.

O facto de trabalhar a 32 bits permite uma representação de grandes números, e evita a necessidade de encontrar um dispositivo externo para armazenar os valores da contagem. Porém, a leitura do valor do contador é feita através da concatenação de 4 bytes consecutivos. Para fazer a leitura dos valores basta alternar entre as diferentes combinações dos pinos SEL1 e SEL2, encarregados da seleção do buffer a ser lido (Tabela 2.3).

Sendo um contador de quadratura com modo de contagem de 4 vezes, isso implica

Figura 2.9: Suporte para encoder acoplado à roda lateral traseira do AtlasCar.

		Seleção do Buffer			
SEL 1	**SEL 2**	**1º byte**	**2º byte**	**3º byte**	**4º byte**
0	1	D4	-	-	-
1	1	-	D3	-	-
0	0	-	-	D2	-
1	0	-	-	-	D1

Tabela 2.3: Tabela com a sequência de seleção utilizada para leitura individual de cada buffer

que a cada quarto de pulso da resolução efetiva do *encoder* o contador incremente uma unidade à contagem. Sabendo que o contador lê os pulsos do canal A e B como ilustrado na figura 2.10, desta forma é incrementada uma unidade a cada transição entre os seguintes estados:

- Canal A positivo e B igual a zero (1);
- Canais A e B com sinal positivo (2);
- Canal A zero e B positivo (3);
- Canais A e B com sinal igual a zero (4).

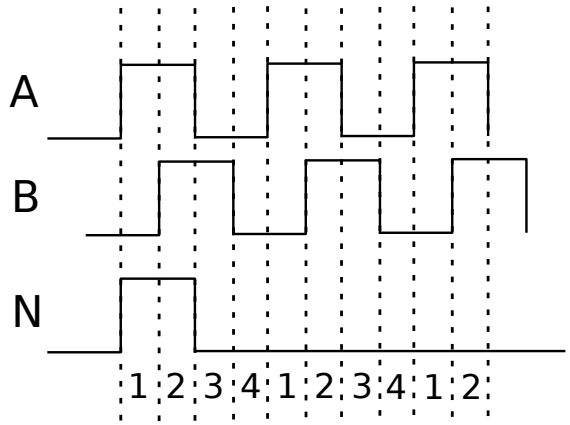

Figura 2.10: Ilustração dos pulsos que o *encoder* envia pelos três canais.

2.4.1 Programação do Arduíno da velocidade

O programa usado para calcular a velocidade do carro está ilustrado na figura 2.11. Quando o Arduíno é iniciado os seguintes comandos são executados:

- As portas utilizadas por este são definidas, como digitais.
- São incluídas as bibliotecas de comunicação do protocolo Ethernet que irá comunicar com a shield ethernet e a biblioteca que comunica com o contador de quadratura.
- Definida uma variável cuja função é garantir que o valor de velocidade enviado para o computador não é repetido.

- Por fim é associado a um *interrupt* com um tempo de ciclo de 100 *milisegundos*, uma função que lê os valores da contagem do *HCTL-2022*.

Depois da iniciação do Arduíno é então começado o loop principal, onde o Arduíno espera que um cliente se conecte via Ethernet. Quando este estiver conetado, o Arduíno envia a informação para o computador. De salientar que o interrupt implementado, está associado a uma função que a cada ciclo de 100 *ms*, executa as seguintes operações:

- Leitura do valor da contagem;
- Tempo interno do microcontrolador;
- Define a variável count=1.

O cálculo da velocidade é feito caso a variável count seja verdadeira, desta forma o cálculo da velocidade implementado no Arduíno tem por base as equações descritas na subsecção 2.4.2.

2.4.2 Cálculo da velocidade

A velocidade é calculada com base na seguinte expressão:

$$v = \frac{\triangle s}{\triangle t} \ [m/s] \tag{2.1}$$

A distância percorrida ($\triangle s$) é calculada:

$$\triangle s = \pi \times D \times \frac{P}{4} \times \frac{1}{N_{\text{Resolução do } encoder}} \ [m] \tag{2.2}$$

Onde P é o número de pulsos lidos pelo contador desde a última vez, de notar que está dividido por 4 unidades pois o modo de contagem incrementa um valor por cada quarto de pulso efetivo do *encoder*; D o diâmetro da roda, 0,52 m e N corresponde à resolução do *encoder*, 50 pulsos.

A variação do tempo $\triangle t$ é calculada pela diferença dos dois instantes:

$$\triangle t = t_f - t_i \ [s] \tag{2.3}$$

De modo a facilitar o entendimento matemático, T = $\triangle t$.

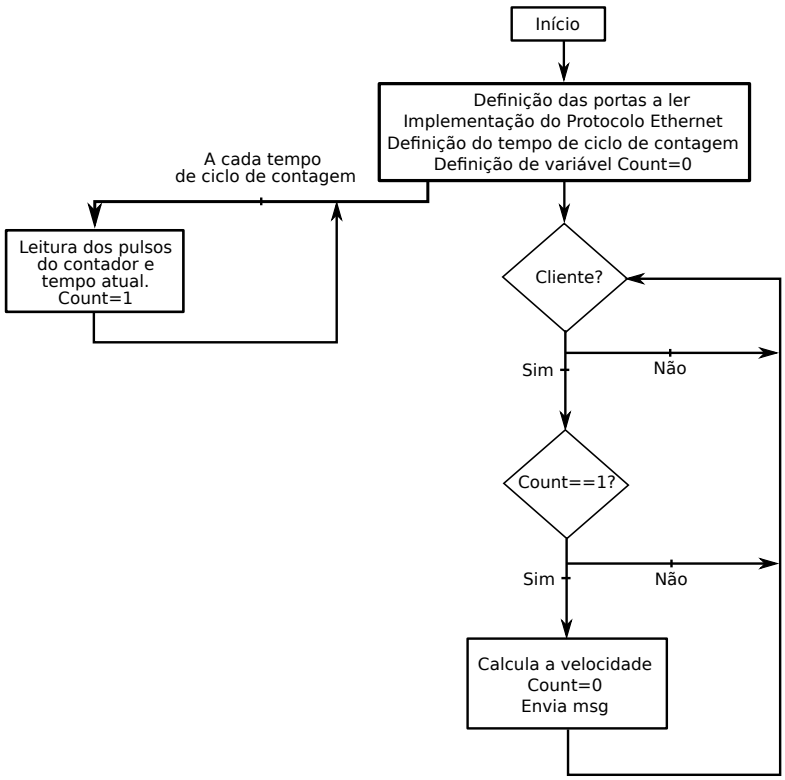

Figura 2.11: Diagrama explicativo do código utilizado na programação do Arduíno responsável pela medição da velocidade.

s

Desta forma a velocidade do carro é dada pela seguinte equação:

$$v = \frac{\pi \times D \times P}{4 \times 50 \times T} \ [m/s] \tag{2.4}$$

A resolução minima que o sistema detêm é calculada pela equação 2.4, onde após substituição dos valores de $P = 1$ pulso, $D = 0.58$m e $T = 0.1$s, temos que:

$$v = \frac{\pi \times 0.58 \times 1}{4 \times 50 \times 0.1} = 0.0911 \ [m/s] = 0.0328 \, [Km/h] \tag{2.5}$$

Para estimar a incerteza no cálculo da velocidade foi utilizada a expressão da propagação de erro:

$$(\triangle v)^2 = \sum \left(\triangle x_i \times \frac{\partial v}{\partial x_i} \right)^2 \tag{2.6}$$

$$\| \triangle v \|^2 = \left(\triangle T \times \frac{-D \times \pi \times P}{200 \times T} \right)^2 + \left(\triangle D \times \frac{P \times \pi}{200 \times T} \right)^2 + \left(\triangle P \times \frac{D \times \pi}{200 \times T} \right)^2. \tag{2.7}$$

O erro da leitura do tempo é igual à menor unidade de medida temporal do Arduíno, uma vez que este trabalha a 16 MHz a menor unidade de tempo é dada pela seguinte expressão:

$$\triangle T = \frac{1}{16000000} = 0.000000062\ [s]. \tag{2.8}$$

O erro na medição do diâmetro da roda, corresponde a:

$$\triangle D = 0.001\ [m]. \tag{2.9}$$

O erro da leitura dos pulsos é dado pela expressão:

$$\triangle P = \frac{1}{200} = 0.005\ pulsos. \tag{2.10}$$

Após substituição dos valores na equação 2.7, tem-se:

$$\| \triangle v\|^2 = \left(-6.9 \times 10^{-9} \right)^2 + \left(1.6 \times 10^{-4} \right)^2 + \left(4.4 \times 10^{-6} \right)^2 \tag{2.11}$$

Sendo que o valor do erro do cálculo da velocidade de:

$$\triangle v = 1.57 \times 10^{-4}\ [m/s]. \tag{2.12}$$

A configuração física do Arduíno da velocidade implementado no AtlasCar tem por base a informação presente, no esquema elétrico no capítulo 6. Na figura 2.12 é mostrada a configuração das portas utilizadas pelo Arduíno para que este possa comunicar com o *HCTL-2022*. A leitura do valor da contagem é feito pelos pinos D1 a D9. Os pinos OE e RST são responsáveis pela permissão de leitura do contador e pelo *reset* da contagem, respetivamente.

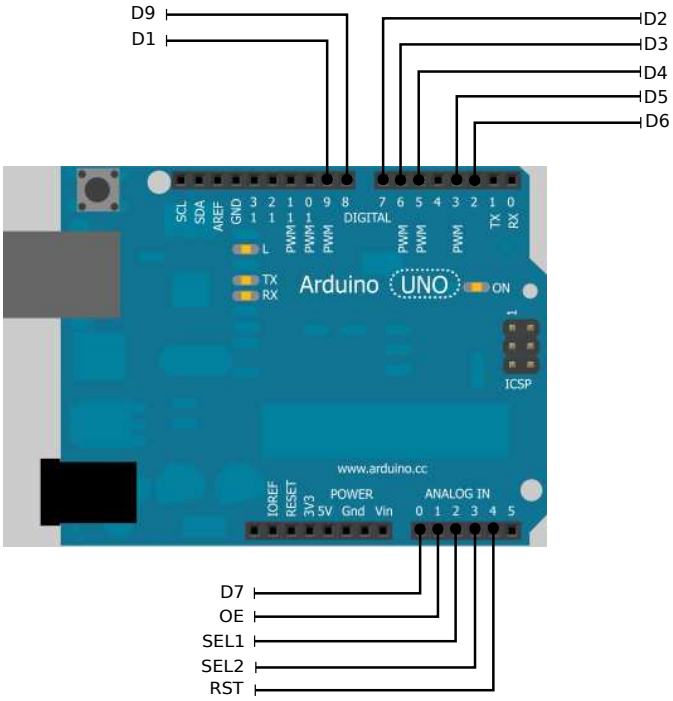

Figura 2.12: Esquema das portas utilizadas pelo Arduíno responsável pela medição da velocidade.

2.4.3 Mensagem de velocidade do carro

A mensagem enviada para o módulo de *ROS*, contém todas as variáveis que o Arduíno utiliza para calcular a velocidade, enviando assim o número absoluto de pulsos desde que o carro foi ligado, os pulsos por segundo, as rotações por segundo e por fim a velocidade em m/s do veículo. A mensagem enviada tem a estrutura presente na tabela 2.4.

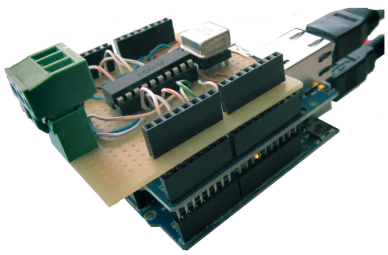

Figura 2.13: Arduíno, shield ethernet e contador de pulsos *HCTL-2022*.

Endereço de IP do Arduíno	10.0.0.31
Estrutura da mensagem	**Descrição das variáveis**
'0x02'	início de mensagem
"C 0"	número de pulsos total desde que o carro foi ligado
"P 0.00"	pulsos por segundo
"R 0.00"	rotações da roda por segundo
"S 0.0000"	velocidade em metros por segundo
'0x03'	fim de mensagem

Tabela 2.4: Formato da mensagem enviada com a velocidade do carro

Capítulo 3

Deteção de Situações de Risco

Ao longo deste capítulo serão indicados os pressupostos utilizados para a deteção das diferentes situações de risco durante o exercício da condução.

3.1 Análise das situações de risco

As situações em análise são consideradas de risco de colisão. A distância ao carro da frente, onde tencionamos medir o tempo de colisão. A atenção que o condutor tem aos obstáculos em redor do carro. Sinalizando atempadamente a mudança de direção ou ultrapassagem.

3.1.1 Pressupostos

A aquisição de dados em redor do veículo é feita através de dois detetores laser planares Sick, como na figura 3.2. Os sensores laser estão colocados nos extremos do pára-choques dianteiro. Este estudo foi restringido à deteção das situações em percurso reto. A razão desta decisão deve-se à atual impossibilidade de distinguir uma manobra de ultrapassagem de uma curva, não sendo ainda possível detetar os limites da via. Outra limitação encontrada é a de não ser possível detetar nenhum objeto na zona traseira do carro, como ilustrado na figura 3.1.

3.1.2 Deteção e seguimento de alvos múltiplos

Para fazer o seguimento dos diferentes objetos em redor do carro, foi utilizado um módulo de seguimento que tem por base o trabalho desenvolvido por Almeida & Santos [Almeida and Santos, 2010]. A vantagem deste módulo está no seguimento de objetos

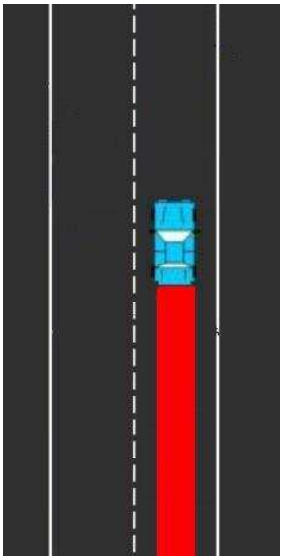

Figura 3.1: Zona vermelha, representa local onde não é possível detetar obstáculos.

Figura 3.2: Sensor laser, instalado no AtlasCar.

que deixam de ser localizados pelos lasers, durante um determinado período de tempo. O módulo utiliza filtros de Kalman, e faz uma procura numa zona elíptica posterior ao local onde o objeto foi detetado pela última vez. Esta busca tem por base a última velocidade e direção conhecidas do objeto. Alguns dos problemas de deteção dos objetos devem-se a acelerações bruscas destes durante o período em que não são detetados pelos lasers. A vantagem na utilização deste tipo de lasers está na menor carga computacional requerida para analisar os vários obstáculos, pois só se processa informação num plano. Desta forma existe uma *package mtt* em R.O.S. que publica a posição e velocidade dos

objetos em redor do AtlasCar, a informação é relativa ao referencial do carro, situado ao centro do para-choques dianteiro.

3.1.3 Formato da mensagem gravada

As situações de risco são registadas em ficheiro de texto ao longo de cada percurso, para análise posterior. Este registo permite uma melhor consulta dos ficheiros .bag guardados.

O ficheiro gerado pelo algoritmo contém, no primeiro campo o nome do ficheiro. No segundo campo a situação de risco encontrada. Os três seguintes contêm o tempo em que teve início a situação, a duração que a mesma teve e o tempo em que terminou. (Figura 3.3)

```
avenida_2.bag    object detected--> Colision Risk        1337685007.159698       1.002696        1337685008.162394
avenida_2.bag    object detected--> vehicle behind the car       1337685006.840652       7.604064        1337685014.444716
avenida_2.bag    Forgot to blink left!   1337685008.576476       5.989801        1337685014.566277
avenida_2.bag    object detected--> vehicle behind the car       1337684938.977571       4.688643        1337684943.666215
avenida_2.bag    object detected--> Colision Risk        1337684956.959087       10.687846       1337684967.646933
avenida_2.bag    object detected--> vehicle behind the car       1337684938.992711       4.691593        1337684943.684304
avenida_2.bag    object detected--> vehicle behind the car       1337684962.747304       1.212484        1337684963.959789
avenida_2.bag    object detected--> Colision Risk        1337684956.943748       10.715037       1337684967.658785
avenida_2.bag    object detected--> vehicle behind the car       1337684938.984960       19.871805       1337684958.856765
```

Figura 3.3: Excerto do ficheiro gerado pelo módulo de deteção da situação de risco.

3.1.4 Alertas visuais ao condutor

O algoritmo desenvolvido sinaliza obstáculos que estejam dentro das zonas azuis da figura 3.4, detetando assim uma possível ultrapassagem (zona 1), bem como o tempo de colisão ao obstáculo na frente do carro (zona 2).

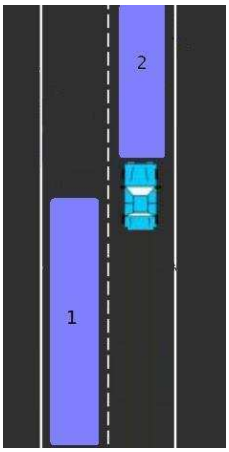

Figura 3.4: Representação das zonas de procura de obstáculos.

Assim, o módulo de deteção das situações de risco subscreve a mensagem do Arduíno da velocidade e da monitorização, do PLC e por fim dos dois lasers Sick. (Figura 3.5)

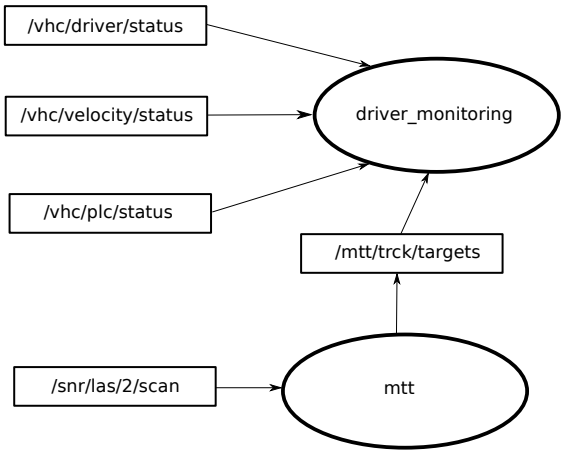

Figura 3.5: Esquema da comunicação existente para o funcionamento do módulo de deteção das situações de risco.

O programa gera alertas visuais, assinalando com um cubo os obstáculos que incorrem nas seguintes situações:

- Aviso de obstáculo: Este aviso sinaliza o obstáculo mais próximo do AtlasCar e que se encontra dentro da zona lateral ao carro. Na figura 3.6 o obstáculo detetado está assinalado com um cubo azul na zona de procura 1.

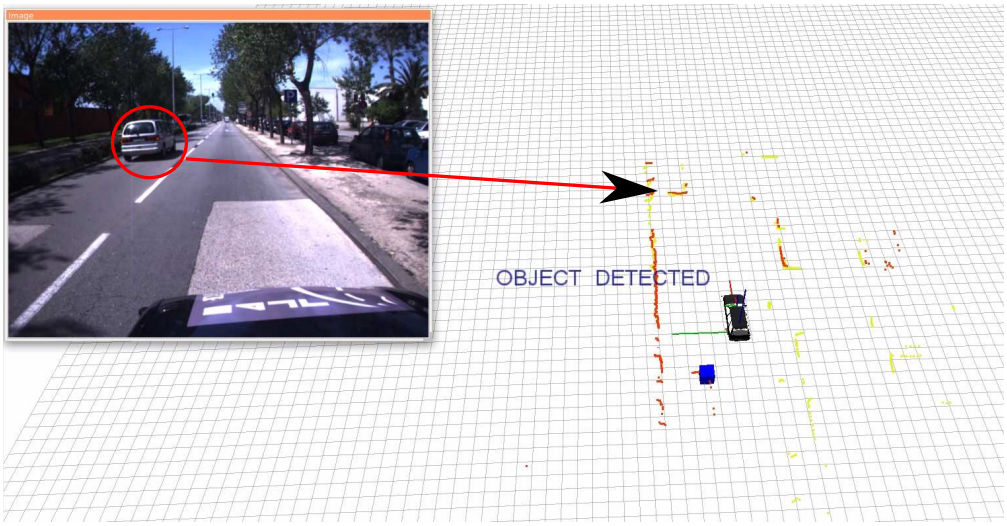

Figura 3.6: Deteção de um obstáculo na zona lateral do carro.

- Falta de sinalização: O estado dos piscas é utilizado pelo algoritmo indicando a falha do condutor, isto é, o não aviso dos outros utentes da via (3.7).

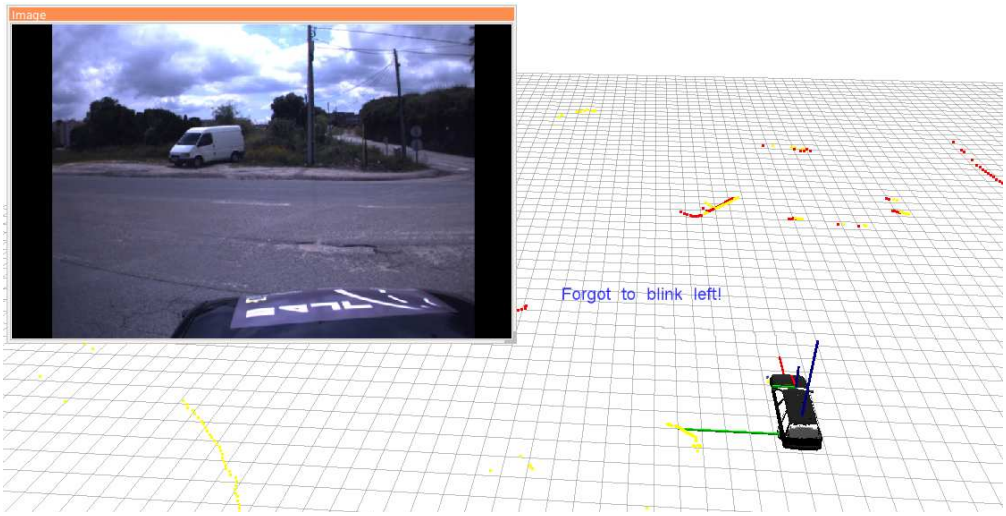

Figura 3.7: Indicação de falta de sinalização para o lado esquerdo.

- Boa manobra: Têm em conta a velocidade do carro visto circular dentro de uma localidade estando restrito à velocidade máxima de 50 km/h, o facto de fazer sinal indicador de mudança de direção (pisca) e por fim a não existência de nenhum objeto na zona lateral esquerda do carro. (figura 3.8)

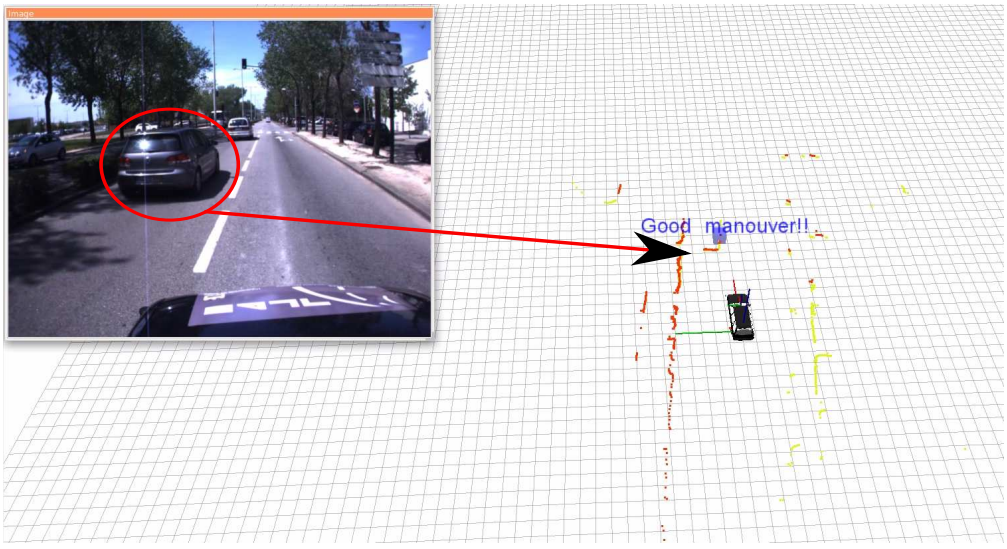

Figura 3.8: Indicação de manobra bem executada.

- Risco de colisão: Este aviso tem por base o cálculo do tempo de colisão ao obstáculo situado na zona 2 da figura 3.4. O tempo de colisão é calculado pela seguinte equação:

$$\mathrm{T}_c = \frac{D}{V} \ [s] \tag{3.1}$$

  Onde D é a distância ao obstáculo e V a velocidade que o objeto detetado tem em relação ao carro. O aviso de risco de colisão é publicado quando o tempo de colisão ao objeto mais próximo da frente do AtlasCar é inferior a 6 segundos (Figura 3.9). Foram definidos 6 segundos pois testes realizados com o AtlasCar em piso seco, definiram que este demorava a travar dos cinquenta aos zero kilómetros por hora 4.76 segundos, em média. Se tivermos em conta que o condutor demora 1 segundo a até iniciar a imobilização do carro. Teremos um total de 5.76 segundos, até à imobilização total do veículo.

- Situações de perigo: Embora não tenham sido utilizadas, a buzina e os indicadores de perigo (4 piscas) são considerados pelo algoritmo como um alerta externo, ditado pelo condutor para que seja registado como situação de risco no ficheiro.

A vantagem na utilização deste módulo, está na melhor perceção que o condutor tem das situações de risco em que é envolvido diariamente nas estradas.

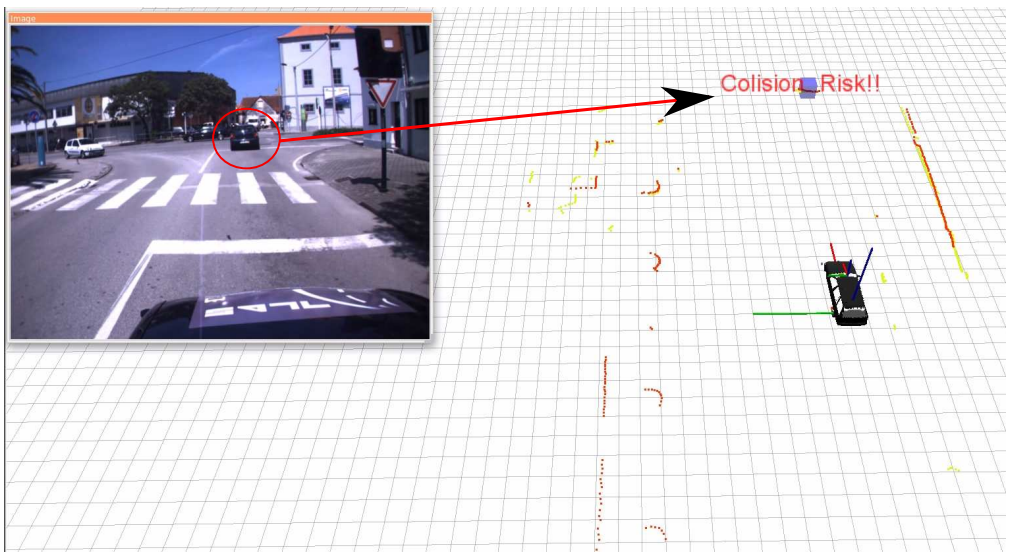

Figura 3.9: Indicação de risco de colisão.

Capítulo 4

Identificação do Condutor

Outro dos objetivos era a criação de um algoritmo que consiga reconhecer o condutor do veículo, com base num processo prévio de registo de dados da condução. Para tal ser exequível, foi necessário escolher de entre todas as manobras de condução uma que fosse característica a cada condutor: a manobra escolhida foi o estacionamento em paralelo.

4.1 Pressupostos

Para esta experiência foram selecionados três voluntários do sexo masculino, com idades compreendidas entre os 22 e 25 anos, tendo sido instruídos quanto ao tipo de manobra a fazer, bem como os estágios intermédios que a manobra envolve.
Assim, a manobra foi dividida em três estágios diferentes (Figura 4.1). No primeiro a), o condutor imobiliza o carro e inicia a marcha atrás; o segundo b), já compreende uma paragem perto do local de destino; e por fim um pequeno deslocamento para a frente c).

Esta divisão da manobra em três fases distintas permite um melhor tratamento estatístico, visto que é possível distinguir facilmente o início e fim de cada estágio da manobra, bastando para isso, verificar a alteração do valor da velocidade.

4.2 Tratamento de dados

A experiência contabilizou seis testes a cada um dos 3 indivíduos.
Todos os testes foram considerados válidos, pois cumpriam com os pressupostos indicados no início da experiência.

Tendo em vista uma melhor ferramenta para tratar os dados foi desenvolvido uma

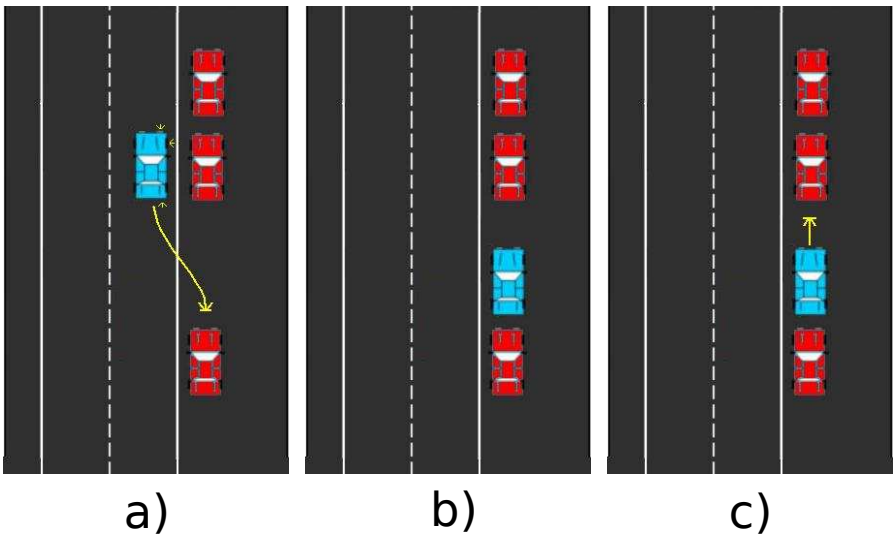

Figura 4.1: Diferentes fases da manobra.

package de *R.O.S.* que exporta a informação em formato *ascii* para posterior tratamento em *MATLAB*. Quando este módulo é executado, gera um ficheiro em formato de texto com o estado de algumas das variáveis do sistema (tabela 4.1).

Variáveis gravadas pelo modulo ROS	**Valores / unidades**
Nome do ficheiro *.bag*	
Tempo do ficheiro	segundos
Estado da ignição	0 ou 1
Estado pisca esquerdo	0 ou 1
Estado pisca direito	0 ou 1
Valor do sensor da acelerador	0 a 1023
Valor do sensor da travão	0 a 1023
Valor do sensor da embraiagem	0 a 1023
Velocidade do veículo	m/s
Rotações do motor	rotações por minuto

Tabela 4.1: Mensagem gravada para posterior tratamento em ambiente *MATLAB*.

4.3 Descritores utilizados

Para conseguir escolher de entre os vários condutores foi necessário identificar descritores para toda a manobra; os descritores potenciais considerados são mostrados na tabela 4.2.

Notação matemática	**Variáveis gravadas pelo modulo *ROS***
D_1	tempo total da manobra (s)
D_2	tempo do primeiro estágio (s)
D_3	tempo do segundo estágio (s)
D_4	tempo do terceiro estágio (s)
D_5	velocidade média no primeiro estágio
D_6	velocidade média no terceiro estágio
D_7	rotação média do motor no primeiro estágio
D_8	rotação média do motor no segundo estágio
D_9	rotação média do motor no terceiro estágio
D_{10}	média do produto do ângulo da direção com a velocidade no primeiro estágio
D_{11}	média do produto do ângulo da direção com a velocidade no terceiro estágio
D_{12}	ângulo médio da direção no segundo estágio
D_{13}	variância do ângulo da direção no segundo estágio
D_{14}	tempo que o pisca direito está ligado
D_{15}	número de vezes que o acelerador esta premido "ao de leve"
D_{16}	número de vezes que o acelerador esta premido com "leve média"
D_{17}	número de vezes que o acelerador esta premido com "média força"
D_{18}	número de vezes que o acelerador esta premido com "muita força"
D_{19}	quantidade de tempo que a embraiagem está premida
D_{20}	número de vezes que o travão esta premido "ao de leve"
D_{21}	número de vezes que o travão esta premido com "força leve média"
D_{22}	número de vezes que o travão esta premido com "média força"
D_{23}	número de vezes que o travão esta premido com "muita força"

Tabela 4.2: Representação de todos os descritores utilizados.

Alguns descritores da tabela 4.2 foram separados consoante os diferentes estágios da manobra, (figura 4.1) pois estão presentes ao longo de toda a manobra, destacando assim os tempos e as rotações do motor, $D_{2,3,4}$ e $D_{7,8,9}$, respetivamente.

Foram tidos em conta os tempos totais em que o acelerador e travão estiveram premidos durante toda a manobra de estacionamento. Nos casos do acelerador e travão foi possível dividir em quatro gamas diferentes a força aplicada, $D_{15,16,17,18}$ e $D_{20,21,22,23}$, respetivamente.

Nos casos do pedal da embraiagem, D_{19}, e do sinal indicador de mudança de direção D_{14}, foi tido em conta a quantidade de tempo que estes estão premidos. No caso particular da embraiagem, esta está sempre no seu valor máximo ou não apresenta carga, tornando-a numa variável booleana para esta manobra específica.

O descritor da velocidade foi tido em conta no primeiro e terceiro estágios, $D_{5,6}$. De notar que o carro está parado durante o segundo estágio, daí que não seja relevante ter como descritor a velocidade do veículo. Outro descritor para os estágios anteriormente descritos poderá ser o produto da velocidade do carro pelo ângulo da direção $D_{10,11}$. No segundo estágio utilizamos o ângulo médio da direção e a sua variância, $D_{12,13}$.

4.4 Escolha de descritores

Com uma lista de vinte e dois descritores seria complexo escolher o melhor descritor ou combinação de descritores que distinguissem os diferentes condutores durante o estacionamento, porque são excessivos face à baixa diversidade de amostras obtidas. Por este motivo, foi necessário desenvolver um algoritmo que fizesse teste a todas as combinações possíveis e que depois de executado desse as combinações que cumprissem as condições impostas pelo utilizador.

4.4.1 Notação utilizada

Nesta subsecção é apresentada notação matemática utilizada na formalização do algoritmo de escolha de descritores.

Notação matemática	**Significado**	**Dimensão**
$\mathbf{T}_k^x$	teste k do indivíduo x.	1×23
D_n	descritor n da tabela 4.2.	1×1
$\mathbf{B}^x$	matriz base de descritores de referência do indivíduo x.	4×23
$\overline{\mathbf{B}}^x$	média dos descritores para o indivíduo x.	1×23
$\overline{\mathbf{B}}_t$	média das bases de referência dos três indivíduos.	3×23
$\mathbf{B}_m$	valores máximos de cada descritor das bases de referência.	1×23
$\overline{\mathbf{B}}_n$	valores médios normalizados das bases.	3×23
$\mathbf{V}_d$	variância de cada descritor, presente nas três bases de referência.	1×23
P	valor definido pelo utilizador como variância mínima que os descritores têm de ter, para serem considerados como descriminantes.	1×1
k_i	número de arranjos para cada combinação de descritores. Parâmetro definido pelo utilizador.	1×1
$\mathbf{V}_{fd}$	índices dos descritores escolhidos pelo parâmetro P.	$1 \times k_i$
$\mathbf{C}$	matriz de combinações de descritores a serem testadas.	$C_p \times k_i$
$\mathbf{B}^{T^x}$	valor médio dos descritores de cada linha de $\mathbf{C}$, para cada base de referência.	$1 \times k_i$
$\mathbf{T}_5^{T^t}$	valor dos descritores de cada linha de $\mathbf{C}$, para o quinto ficheiro de cada individuo.	$1 \times k_i$
$\mathbf{E}_t^x$	distância euclideana para o teste t e base de referência do indivíduo x.	1×23
$\mathbf{F}^{k_i}$	matriz de combinações escolhidas pelo algoritmo para arranjos de k_i	$C_p \times k_i$.

Tabela 4.3: Descrição da notação utilizada.

- $\underset{k}{med}\underbrace{\left[A(n,k)\right]}_{\text{n x k}} = \underbrace{\left[medA(:,1)\ ...\ medA(:,k)\right]}_{\text{1 x k}}$
- $\mathbf{X}./\mathbf{Y} \rightarrow$ Divisão dos elementos de $\mathbf{X}$ por $\mathbf{Y}$, um a um.
- $\mathbf{X}(:, \mathbf{Y}(\mathbf{Z}, :)) \rightarrow$ Seleção das colunas de $\mathbf{X}$ cujo valor seja o valor da linha $\mathbf{Z}$ do vetor $\mathbf{Y}$.

4.4.2 Formalização do algoritmo utilizado

São dados de entrada os 22 descritores (tabela 4.2), 6 ficheiros de teste para cada condutor, num total de 18 testes. Dos seis ficheiros de cada condutor, quatro são ficheiros que serão usados para criar a base de referência, $\mathbf{B}^x$.

O quinto ficheiro de cada condutor servirá para escolher os descritores que mais se adequam a um determinado condutor, $\mathbf{T}_5^x$.

Desta forma cada teste pode ser descrito por:

$$\mathbf{T}_k^x = \begin{bmatrix} D_1 & D_2 & D_3 & ... & D_{22} \end{bmatrix},\ x = \{1,2,3\}\ k = \{1,2,...,6\} \tag{4.1}$$

Em seguida é criada uma base de referência para cada indivíduo x, utilizando apenas os 4 primeiros ficheiros de teste, $\mathbf{B}^x$.

$$\mathbf{B}^x = \begin{bmatrix} \mathbf{T}_1^x \\ \mathbf{T}_2^x \\ \mathbf{T}_3^x \\ \mathbf{T}_4^x \end{bmatrix},\ x = \{1,2,3\} \tag{4.2}$$

Será feita a média de cada descritor para cada base de referência. A média é feita ao longo da variável D_n, isto é, para todas as colunas de $\mathbf{B}^x$.

$$\overline{\mathbf{B}}^x = \underset{D_n}{med}\left(\left[\mathbf{B}^x(1:4, D_n)\right]\right),\ x = \{1,2,3\},\ n = \{1,2,...,22\} \tag{4.3}$$

O valor médio das três bases de referência é concatenado, dando origem a $\overline{\mathbf{B}}_t$.

$$\overline{\mathbf{B}}_t = \begin{bmatrix} \overline{\mathbf{B}}^1 \\ \overline{\mathbf{B}}^2 \\ \overline{\mathbf{B}}^3 \end{bmatrix} \tag{4.4}$$

Para conseguir normalizar os descritores é então calculado o máximo de cada descritor.

$$\mathbf{B}_m = \underset{D_n}{max}\left(\overline{\mathbf{B}}_t(1:3, D_n)\right),\ D_n = \{1, 2, ..., 22\} \tag{4.5}$$

Em seguida normalizam-se os valores de todos os descritores da base de referência, $\overline{\mathbf{B}}_t$, pelos valores máximos, $\mathbf{B}_m$.

$$\overline{\mathbf{B}}_n(1:3, D_n) = \overline{\mathbf{B}}_t./\mathbf{B}_m \tag{4.6}$$

Depois é calculada a variância de cada descritor nas três bases de referência.

$$\mathbf{V}_d = \underset{D_n}{var}(\overline{\mathbf{B}}_n(1:3, D_n)),\ \textit{em coluna} \tag{4.7}$$

Em seguida o algoritmo percorre todas as colunas de $\mathbf{V}_d$ e compara-os com um valor P, definido pelo utilizador. Se o valor $\mathbf{V}_d$ for maior do que o valor de P, então cria-se o vetor $\mathbf{V}_{fd}$ que contém o número dos descritores que passaram neste filtro. Esta operação tem o objetivo de excluir os descritores que variam muito pouco nas três bases de dados, logo não serão adequados como descritores dos diferentes condutores.

Após esta primeira filtragem aos descritores, será então necessário verificar as diferentes combinações para cada um dos ficheiros de teste, deste modo:

Obtém-se uma matriz $\mathbf{C}$ que, ao longo de cada linha, contém as combinações de $\mathbf{V}_{fd}$ arranjados a k_i. Onde k_i pode ter valores entre 1 e 3. O número total de combinações possíveis dos valores das colunas de $\mathbf{V}_{fd}$ é dado por:

$$C_p = \frac{n!}{k_i!(n-k_i)!},\ \text{onde } n \text{ é o número de descritores contidos em } \mathbf{V}_{fd} \tag{4.8}$$

A matriz $\mathbf{C}$ é o resultado da utilização da função *MATLAB*, *combnk*, que disponibiliza uma matriz com todas as combinações possíveis de $\mathbf{V}_{fd}$ arranjadas a k_i. Como podemos reparar, a matriz $\mathbf{C}$ tem tamanho variável, consoante os arranjos k_i que sejam feitos e a quantidade de descritores existentes.

$$\mathbf{C} = \left[C_p \times k_i\right] \tag{4.9}$$

De modo a encontrar combinações que discriminem os três condutores, para cada linha de C, isto é, para cada conjunto de descritores arranjados a k_i:

São retiradas as colunas correspondentes aos descritores de cada base de referência média.

$$\mathbf{B}^{T^x} = \overline{\mathbf{B}}^x(1, \mathbf{C}(n_i, :)),\ n_i = \{1, 2, ..., C_p\} \tag{4.10}$$

Onde $\mathbf{B}^{T^x}$ representa um vetor linha com os valores médios de cada descritor da base de referência do indivíduo x.

De mesma forma para cada quinto teste ($\mathbf{T}_5^x$) são retiradas as colunas respetivas a cada linha de C.

$$\mathbf{T}_5^{T^t} = \mathbf{T}_5^x(1, \mathbf{C}(n_i, :)),\ n_i = \{1, 2, ..., C_p\} \tag{4.11}$$

É então calculada a distancia Euclideana entre o vetor $\mathbf{T}_5^{T^t}$ e cada base de referência de cada individuo, $\mathbf{B}^{T^x}$.

$$\mathbf{E}_t^x = \sqrt{\left(\mathbf{T}_5^{T^t} - \mathbf{B}^{T^x}\right)^\intercal \times \left(\mathbf{T}_5^{T^t} - \mathbf{B}^{T^x}\right)},\ x = \{1, 2, 3\},\ t = \{1, 2, 3\} \tag{4.12}$$

O mesmo processo ocorre para os ficheiros de teste seguintes, t=2 e t=3. A combinação só é considerada válida caso o valor mínimo da distância euclideana em cada teste $\mathbf{T}_5^{T^t}$ corresponda à base de referência do indivíduo que fez o teste. Em caso de serem verificadas estas condições, o conjunto de descritores são armazenadas na matriz $\mathbf{F}^{k_i}$.

Por fim os últimos ficheiros de teste $\mathbf{T}_6^t$ são reservados para validação das combinações escolhidas pelo algoritmo. Desta forma é repetida apenas a última fase do processo anteriormente descrito, onde para cada combinação de $\mathbf{F}^{k_i}$ é verificado se o valor é mínimo para a distância euclideana daquele teste quando calculado com as bases de referência dos três indivíduos, prosseguindo desta forma até ao último ficheiro de teste.

Capítulo 5

Resultados Experimentais e Conclusão

5.1 Resultados Experimentais

Neste capítulo serão abordados os diferentes testes realizados, aos diferentes módulos e será feita uma análise crítica aos mesmos. O trajeto escolhido para testar o funcionamento destes equipamentos no AtlasCar está ilustrado na figura 5.1. Este percurso situa-se em Aveiro, foi realizado em aproximadamente 7:40 minutos e tem cerca de 3,3 km.

5.1.1 Módulo Arduíno da monitorização

Após a implementação dos dois sistemas de hardware no AtlasCar, descritos no capítulo 2, foram realizados testes para aferir o bom funcionamento dos dois módulos. Um dos objetivos deste trabalho é a monitorização do habitáculo para tal era necessário que este fosse constantemente monitorado. O módulo de monitorização apresentou uma frequência média de publicação das mensagens de 58 Hz e variância de 6.82×10^{-3} Hz. Se tivermos em conta a frequência a que este sistema publica uma mensagem, considera-se esta diferença irrelevante para o bom funcionamento dos módulos que subscrevem esta mensagem. De notar que este sistema é o que publica as mensagens com a frequência mais alta.

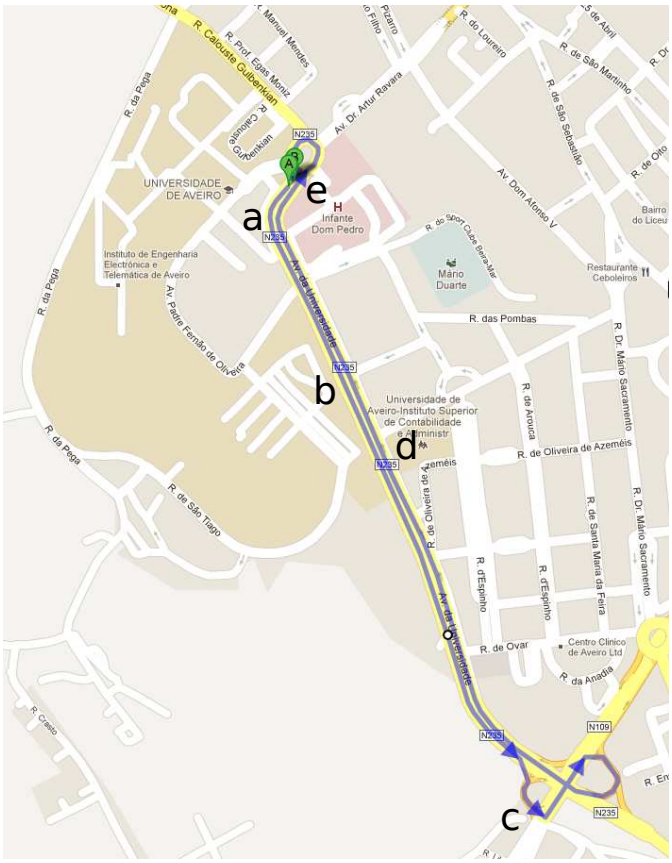

Figura 5.1: Trajeto efetuado para testes dos dois Arduínos.

5.1.2 Módulo Arduíno da velocidade

A frequência de atualização do valor da velocidade é semelhante ao tempo definido no programa do Arduíno. Como referenciado no capítulo 2, a cada 100 ms o Arduíno lê as portas do contador de velocidade, daí que a frequência de atualização da mensagem seja cerca de 10 Hz. Contudo esta frequência de atualização da velocidade demonstra ser bastante rigorosa ao longo do tempo, com uma variância de 4.5909×10^{-6}. Um dos problemas encontrados no sistema antigo, quando a leitura da velocidade era feita pelo PLC, consistia na irregularidade e falha nos valores da velocidade AtlasCar. Tal evento já não se verifica neste novo sistema. (Figura 5.2).

Para testar a robustez das comunicações de cada sistema, foi desligado o cabo Ethernet do Arduíno: este teste serviu para garantir que caso exista uma falha na comunicação, nenhum dos sistemas colapse e que o utilizador seja informado da falha de comunicação, evitando assim o colapso do sistema. Da mesma forma que quando o cabo é ligado este

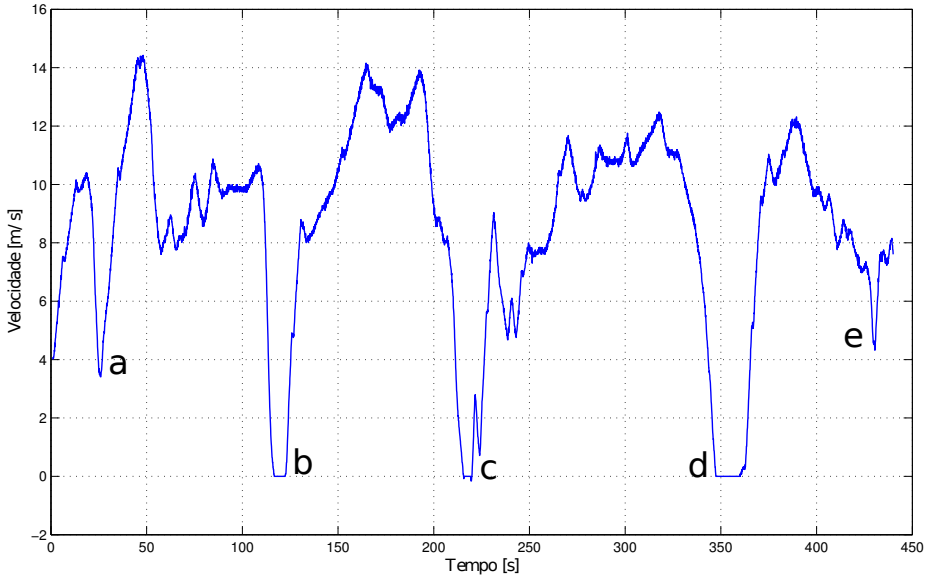

Figura 5.2: Velocidade do veículo ao longo do trajeto da figura 5.1.

volta a estabelecer ligação com o computador automaticamente. A velocidade do carro no trajeto da figura 5.1 pode ser visualizada na figura 5.2 onde a) e e) correspondem a situações típicas em que o condutor abranda a velocidade, mas não imobiliza o veículo. Nas situações b), c) e d) o carro está parado. De relevo a situação c) onde verificamos que o AtlasCar apresenta velocidade negativa, isto acontece, pois o carro estava num cruzamento onde a inclinação do terreno levou a um pequeno recuo deste.

5.1.3 Módulo de deteção das diferentes situações de risco

Como já referido no capítulo 3, o algoritmo deteta as várias situações de risco durante a condução, contudo devido a algumas imposições de hardware, não é possível:

- Detetar situações onde o condutor do Atlascar provoque uma colisão na parte traseira do veículo, devido a uma travagem brusca do mesmo. Este problema é criado pela falta de equipamento sensorial na zona traseira do veículo.
- Distinção entre um peão e uma bicicleta, ou em algumas ocasiões um camião de um carro, devido ao tipo de laser utilizado.
- Detetar os limites da faixa onde circula. Que caso fossem conhecidos não criariam

avisos ao condutor como o da figura 5.3 onde é representado como obstáculo a vegetação que separa as duas vias, induzindo o condutor em erro, e criando falsos registos no ficheiros de texto, gerado pelo módulo.

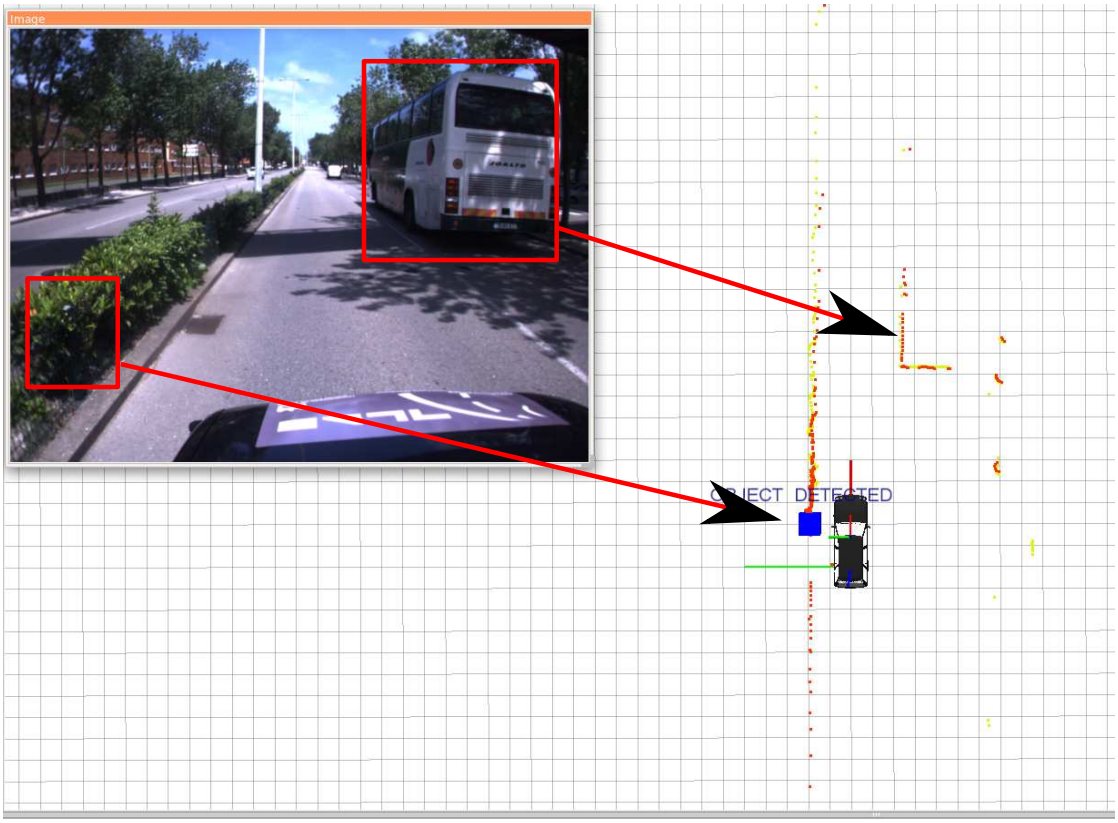

Figura 5.3: Exemplo de uma melhoria a fazer ao algoritmo.

Outro dos problemas encontrados, embora pouco frequente, é a deteção de uma mensagem de risco de colisão em situação de arranque e paragem do veículo, como ilustrado na figura 5.4. De reparar que não existe nenhum objeto na zona indicada pelo algoritmo. A resposta a estas situações está na inclinação que o veículo sofre neste tipo de situações. Apesar destas pequenas contrariedades o módulo apresentou uma eficácia superior a 88,9% no reconhecimento das situações propostas no capítulo 3, como pode ser visto na tabela 5.1

5.1.4 Módulo de deteção de diferentes condutores

Como já anteriormente explicado no capítulo 4, dos seis ficheiros existentes para cada condutor quatro são utilizados para fazer uma base de referência que contém todos os

Nome do ficheiro	Situações	Nº deteções corretas	%
avenida_1.bag	9	8	88.89
avenida_2.bag	5	5	100
avenida_3.bag	12	11	91.67
avenida_4.bag	4	4	100

Tabela 5.1: Repetibilidade do algoritmo em vários percursos.

Figura 5.4: Erro na leitura de valores do laser, devido à inclinação do carro.

descritores. O quinto ficheiro foi utilizado para a escolha dos melhores descritores.

Deste modo foram feitas combinações dos 23 descritores 2 a 2 e 3 a 3, destas combinações foram escolhidas aquelas que mais destacam o condutor escolhido dos restantes, isto corresponde ao valor máximo de P. Após a aplicação do algoritmo, chegou-se ás melhores combinações 2 a 2 e de 3 a 3 ilustradas na tabela 5.2, onde é possível reparar que os descritores 17 e 18, correspondentes ao pedal do acelerador estão sempre presentes.

Na figura 5.5, está representada a variação do número de combinações calculada pelo algoritmo com a diminuição da variância dos descritores. De notar que a quantidade

destes descritores é constante para valores de P superiores a 0,05.

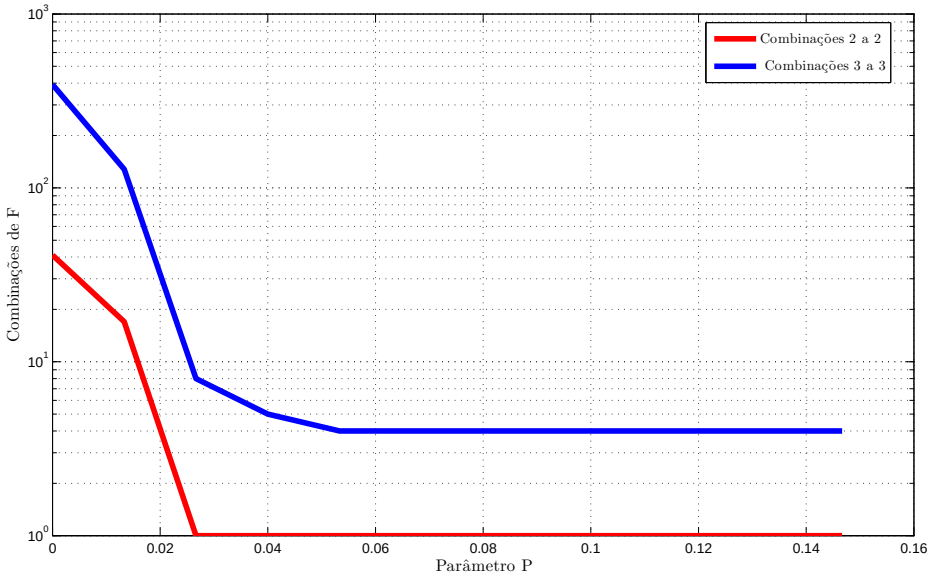

Figura 5.5: Relação entre o número de combinações F e o número P, ou seja, a variância dos vários descritores utilizados.

$k_i = 3$
10 - 17 - 18
12 - 17 - 18
13 - 17 - 18
17 - 18 - 21

$k_i = 2$
17 - 18

Tabela 5.2: Conjunto das melhores combinações de k_i= 3 a 3 e k_i=2 a 2.

Após o processo de validação das combinações escolhidas pelo algoritmo, como referido no final do capítulo 4; foi possível comprovar a veracidade da última das combinações arranjadas 3 a 3 ($k_i = 3$), descartando assim o resto das combinações, inclusive a combinação com arranjos de $k_i = 2$. Esta combinação de descritores 17-18-21, corresponde ao número de vezes que o acelerador está premido com "força média-leve" e "muita força" e ao número de vezes que o travão está premido com força "leve-média".

Estes resultados são bastante bons, contudo o reduzido número de amostras dita um processo de seleção de condutor algo limitado, pois dos 23 descritores disponíveis só foram

feitos arranjos de 3 a 3 e 2 a 2 devido à reduzida quantidade de amostras recolhidas, não sendo por isso viável fazer o estudo para arranjos de descritores superiores a 3.

5.2 Conclusões

Os dois objetivos principais eram o desenvolvimento de dois módulos de Software. Um com capacidade de identificar o condutor do veículo e outro capaz de detetar as situações de risco presentes durante o exercício da condução. Para que estes dois objetivos fossem atingidos foi necessário equipar o AtlasCar com módulos de hardware dignos de recolher dados do habitáculo do veículo, como descrito no capítulo 2.

Desta forma, foram então desenvolvidos com sucesso as duas placas para os Arduínos, bem como o código para comunicar com os mesmos em Ambiente ROS. O Arduíno de monitorização da condução e o Arduíno de medição da velocidade do carro não apresentaram problemas de comunicação ou falha de envio de dados durante o processo de testes, fazendo deles uma excelente ferramenta de desenvolvimento para o projeto AtlasCar. O seu custo associado à sua robustez fazem dele a escolha ideal para este tipo de projetos.

Os resultados no algoritmo de deteção das diferentes situações de risco foram muito bons, na deteção das falhas dos indicadores luminosos ligados, na recompensa visual ao condutor pela manobra de ultrapassagem bem feita e por fim no aviso de risco de colisão.

Os resultados obtidos no módulo de reconhecimento do condutor mostram que é possível distinguir os três condutores utilizando apenas uma combinação de três descritores, bastando para isto monitorar os pedais do acelerador e travão do AtlasCar. Apesar de existir uma limitação em termos do número de bases de dados e indivíduos, o algoritmo está parametrizado de modo a que só seja necessário definir o número de ficheiros testes e indivíduos existentes, para posterior cálculo, não havendo necessidade de alterar o código.

O desenvolvimento dos diferentes módulos em ambiente *R.O.S.* facilitou bastante a organização e integração de trabalhos desenvolvidos anteriormente por outros membros do projeto Atlas.

5.3 Trabalho Futuro

Com o desenvolvimento deste trabalho surge então uma nova frente de trabalho neste tipo de sistemas de apoio à condução;

- Desenvolvimento de módulos específicos para deteção automática de condutores do AtlasCar.
- Câmara traseira. A utilização deste sistema seria importante para fazer um levantamento dos tempos de colisão que outros condutores mantêm em média do nosso veículo.
- Deteção de todos os sinais de trânsito existentes de modo a informar o condutor para o exato cumprimento da sinalização imposta.
- Deteção dos limites da estrada, como o tipo de linhas presentes.

Capítulo 6

Anexos - Esquemas elétricos

Referências

[Almeida and Santos, 2010] Almeida, J. M. S. d. and Santos, V. M. F. d. (2010). Target tracking using laser range finder with occlusion. Master's thesis, Universidade de Aveiro. http://hdl.handle.net/10773/2533.

[Arduíno, 2012] Arduíno (2012). Site oficial do projecto arduíno. http://www.arduino.cc/.

[Association, 2011] Association, G. I. (2011). *Advanced Driver Assistance Systems, An investigation of their potential safety benefits based on an analysis of insurance claims in Germany.* German Insurers Accident Research.

[AVAGO, 2012] AVAGO (2012). Datasheet hctl-2022, quadrature decoder / counter interface ics. http://pdf1.alldatasheet.com/datasheet-pdf/view/198118/HP/HCTL-2022.html.

[Cao et al., 2010] Cao, Y., Theune, M., and Muller, C. (2010). Multimodal presentation of local danger warnings for drivers: A situation-dependent assessment of usability. In *Professional Communication Conference (IPCC), 2010 IEEE International*, pages 226 – 229.

[Continental, 2011] Continental (2011). Sistemas adas da marca continental. http://www.conti-online.com/.

[Field et al., 2011] Field, C. R., Rogers, D. A., Rose-Pehrsson, S. L., Terray, A., Hart, S. J., and Lubrano, A. (2011). Pneumatically modulated liquid delivery system for nebulizers. *Memorandum rept.*

[Fuller, 2005] Fuller, R. (2005). Towards a general theory of driver behaviour. *Accident Analysis & amp; Prevention*, 37(3):461 – 472.

[Johansson and Olofsson, 2007] Johansson, M. and Olofsson, T. (2007). Bayesian model selection for markov, hidden markov, and multinomial models. *Signal Processing Letters, IEEE*, 14(2):129 –132.

[Johnson and Trivedi, 2011] Johnson, D. and Trivedi, M. (2011). Driving style recognition using a smartphone as a sensor platform. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pages 1609 –1615.

[Ko et al., 2008] Ko, M. H., West, G., Venkatesh, S., and Kumar, M. (2008). Using dynamic time warping for online temporal fusion in multisensor systems. *Information Fusion*, 9(3):370 – 388. Special Issue on Distributed Sensor Networks.

[Krajewski et al., 2008] Krajewski, J., Sommer, D., Trutschel, U., Edwards, D., and Golz, M. (2008). Steering wheel behavior based estimation of fatigue. *Proceedings of Measuring Behavior*, 1:23–45.

[Kuge et al., 2000a] Kuge, N., Yamamura, T., Shimoyama, O., and Liu., A. (2000a). A driver behavior recognition method based on a drive model framework. *Proc. of the SAE World Congress*, 4:6–9.

[Kuge et al., 2000b] Kuge, N., Yamamura, T., Shimoyama, O., and Liu, A. (2000b). A driver behavior recognition method based on a driver model framework. *Society of Automotive Engineers, Warrendale PA*, 2:47 – 54.

[Mitrovic, 2005] Mitrovic, D. (2005). Reliable method for driving events recognition. *Intelligent Transportation Systems, IEEE Transactions on*, 6(2):198 – 205.

[Oliveira, 2011] Oliveira, A. (2011). Sistema de monitorização da condução. http://www.youtube.com/watch?v=mv9AKE_ubgY.

[Project, 2011] Project, A. (2011). http://atlas.web.ua.pt.

[Quigley et al., 2009] Quigley, M., Conley, K., Gerkey, B. P., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*.

[RS, 2011] RS (2011). Informação sobre o encoder incremental para a velocidade. http://uk.rs-online.com/web/p/products/2603718/.

[Tran et al., 2012] Tran, C., Doshi, A., and Trivedi, M. M. (2012). Modeling and prediction of driver behavior by foot gesture analysis. *Computer Vision and Image Understanding*, 116(3):435 – 445. Special issue on Semantic Understanding of Human Behaviors in Image Sequences.

[Transducer, 2012] Transducer, P. F. (2012). http://www.measure.com.au/vehicle dynamics and mapping.

[Wang, 2012] Wang, J. (2012). Ballbot: A low-cost robot for tennis ball retrieval. Master's thesis, EECS Department, University of California, Berkeley.

[Works, 2011] Works, H. S. (2011). Descrição detalhada do funcionamento dos sistemas cruise control, anti-bloqueio e pré-colisão. http://auto.howstuffworks.com.

[Yao et al., 2011] Yao, L., Dasgupta, S., Cheng, N., Spingarn-Koff, J., Rudakevych, O., and Ishii, H. (2011). Multi-jump: jump roping over distances. In *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems*, CHI EA '11, pages 1729–1734, New York, NY, USA. ACM.