Universidade de Aveiro Departamento de Engenharia Mecânica
Ano 2021

André Gonçalves Moutinho da Silva

Avaliação de Desempenho de Funcionalidades Viv (Vehicle-Infrastructure-Vehicle) nas comunicações entre um automóvel e um servidor.

Universidade de Aveiro Departamento de Engenharia Mecânica
Ano 2021

André Gonçalves Moutinho da Silva

Avaliação de Desempenho de Funcionalidades Viv (Vehicle-Infrastructure-Vehicle) nas comunicações entre um automóvel e um servidor.

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Mecânica, realizada sob a orientação científica da Professora Doutora Margarida Isabel Cabrita Marques Coelho, Professora Auxiliar com Agregação do Departamento de Engenharia Mecânica da Universidade de Aveiro e do Professor Doutor José Paulo Santos, Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro

Esta dissertação teve o apoio dos projetos:
UIDB/00481/2020 e UIDP/00481/2020 - FCT - Fundação para a Ciência e a Tecnologia;
CENTRO-01-0145-FEDER-022083 - Programa Operacional Regional do Centro (Centro2020), através do Portugal 2020 e do Fundo Europeu de Desenvolvimento Regional
MobiWise (POCI-01-0145-FEDER-016426)
DICA-VE (POCI-01-0145-FEDER-029463).

Dedico este trabalho à minha família companheiros e amigos. A todos obrigado.

o júri

Presidente	Professor Doutor João Alexandre Dias de Oliveira Professor Auxiliar, Universidade de Aveiro
Vogal – Arguente Principal	Professor. Doutor José Amaral Dias Fernandes Professor Auxiliar, Universidade de Aveiro
Vogal - Orientador	Professor Doutora Margarida Isabel Cabrita Marques Coelho Professora Auxiliar c/ Agregação, Universidade de Aveiro

agradecimentos

Em primeiro lugar quero agradecer à minha orientadora, Professora Doutora Margarida Coelho pela disponibilidade, colaboração, apoio, rigor, disciplina e visão ao longo de todo este trabalho.

Quero também deixar um agradecimento muito especial ao meu coorientador Professor Doutor José Paulo Santos, por todo a apoio dado na parte técnica, pela ajuda constante, pelos telefonemas e pelo tempo despendido nesta dissertação.
Sem contribuição de ambos, bem ou mal, este trabalho não teria ficado feito.

Deixo também um agradecimento à investigadora Elisabete Ferreira pelo apoio demonstrado não só ter-se dado ao trabalho de encomendar os sensores tão necessários para a conclusão deste trabalho, mas também por se ter sempre disponibilizado a dar apoio nas pequenas coisas, porque também fazem diferença.

Quero agradecer a todas as pessoas e entidades que de uma forma ou de outra me ajudaram a concluir esta etapa na minha vida.

Não posso deixar de agradecer aos meus pais e ao meu irmão por todo apoio e “puxões de orelhas” que me deram ao longo da vida e ao logo desta jornada. Porque todos os “anima-te tu consegues” e “tens que fazer melhor” servem todos o mesmo propósito.

Aos meus amigos e companheiros, obrigado por estarem lá, por me fazerem sentir melhor quando os dias correm mal, por celebrarem comigo quando os dias correm bem e por me fazerem sempre lembrar que mudar é sempre possível. A todos um muito obrigado.

Um agradecimento também muito especial à minha sobrinha recém-nascida, Carolina por toda a esperança que me deste. Tão jovem e já com um impacto tão grande.

palavras-chave comunicação, V2X, VIV.

resumo

O objetivo principal da dissertação consiste em implementar a comunicação *VIV (Vehicle-Infrastructure-Vehicle)*, para avaliação do desempenho nas comunicações entre um servidor e um automóvel.
Para isso é necessário a implementação e configuração de sensores, como por exemplo o *GPS* e o *LiDAR* e a implementação da comunicação entre o automóvel e um servidor remoto. O módulo *GSM* ficará encarregue da comunicação. O objetivo será enviar informações relativas à localização do automóvel e à distância entre veículos. Foi também necessário modelar uma caixa para albergar todos os equipamentos necessários. Esta caixa deverá ter a capacidade de ser fixa à parte frontal do automóvel. O centro de processamento utilizado foi o *Arduino Mega*. Para servidor foi utilizado a aplicação *MAMP* que para além de um servidor Apache vem com um servidor *MySQL*. Depois do código para o servidor e para o centro de processamento *Arduino* estarem escritos, foram feitos alguns testes com o automóvel em movimento na zona de S. João da Madeira e de Aveiro. O objetivo seria ter um tempo de comunicação abaixo de um segundo. No entanto apenas foi possível ter um tempo de comunicação de sensivelmente cinco segundos.
Apesar dos vários desafios encontrados o caminho está traçado para se puder melhorar e modernizar este projeto.

keywords

Comunication, V2X, VIV.

abstract

The main objective of the dissertation is to implement VIV (Vehicle-Infrastructure-Vehicle) communication, to evaluate the performance in communications between a server and an automobile.
For this, it is necessary to implement and configure sensors, such as GPS and LiDAR, and to implement communication between the car and a remote server. The GSM module will be in charge of the communication. The goal will be to send information regarding the car's location and the distance between vehicles. It was also necessary to model a box to house all the necessary equipment. This box should be able to be fixed to the front of the car. The processing center used was the Arduino Mega. For the server, the MAMP application was used, which in addition to an Apache server comes with a MySQL server. After the code for the server and the Arduino processing center were written, some tests were made with the car moving in the S zone. João da Madeira and Aveiro. The goal would be to have a communication time below one second. However, it was only possible to have a communication time of approximately five seconds.

Despite the various challenges encountered, the path is set to improve and modernize this project.

Índice

Índice de figuras

Índice das Tabelas

Lista de abreviaturas/Siglas

GEE - Gases de Efeito de estufa
RPM - Rotações por minito
ITS - Inteligent Transportation Systems
V2I - Vehicle-to-Infrastructure
DSRC - Dedicated Short-Range Communication
SAE - Sociaty of Automotive Engineers
VIV - Vehicle-Infrastructure-Vehicle
GPS - Global Positioning System
LiDAR - Light Detection and Ranging
GSM - Global System for Mobile
V2X - Vehicle-to-Everything
V2V - Vehicle-to-Vehicle
V2N - Vehicle-to-Network
V2P - Vehicle-to-person
WAVE - Wireless Access in Vehicular Environment
IEEE - Institute of Electrical and Eletronics Engineers
ETSI - European Telecommunications Standards Institute
CAMs - Cooperatives Awareness Messages
ms - milisegundo
DENM - Decentralized Environmental Notification Messages
FDM - Frequency Division Multiplexing
RSU's - Road Side Units
LTE - Long Term Evolution
FDMA - Frequency-Division Multiple Access
RB - Resource Blocks
TB's - Transport Blocks
PSCCH - Physical Sidelink Shared Chanel
SCI - Sidelink Control Information
PSSCH - Physical Sidelink Control Chanel
3GPP - Third Generation Partnership Project
CMOS - Complementary Metal-Oxide Semiconductor
CCD - Charged Couple Device
BSD - Blind Spot Detection
LCA - Lane Changing Assistant
SONAR - Sound Navigation Ranging
RADAR - Radio Detection and Ranging
MMW - Milimeter Wave RADAR
FCTA - Forward Crossing Traffic Alert
MEMS - Microelectomechanical Mirrors
GNSS - Global Navigation Satellite System
EUA - Estados Unidos da América
NMEA - National Marine Electronics Association
TCP/IP - Transmission Control Protocol/Internet Protocol
ADAS - Advanced Driver-Assistance Systems
IDE - Integrated Development Environment
MySQL - My Structured Query Language
GLOSA - Green Optimal Speed Advisory
DGNSS - Diferential GNSS
CO_2 - Dióxido de Carbono
VRU - Vulnerable Road User
RV - Remote Vehicle
HV - Host Vehicle
PWM - Pulse Wide Modulation
SMS - Short Message Service
GPRS - General Packet Radio
DARPA - Defense Advanced Research Projects Agency
HTTP - HyperText Transport Protocol
LED - Light Emiting Diode
I2C - Inter Integrated Circuit
SDA - Serial Data
SCL - Serial Clock
MAMP - Mac OSX Apache MySQL PHP
HTML - HyperText Markup Language
mA - milliamperes
RS232 - Recomended Standard 232
EIA - Electronic Industries Association
5V - Cinco volts
RMC - Recomended Minimum
GMT - Greenwich Mean Time
USB - Universal Serial Bus
TTL - Transistor-Transistor Logic
Gbps - Giga bits por segundo
kbps - kilo bits por segundo
PHP - Perl or Python
km – Quilómetros
5G - Quinta geração
2G - Segunda geração
CV - Veículos Conectados
AV - Veículos Automatizados
NR - New Radio
kHz - kilo Hertz
ToF - (Time of flight)
GHz - Giga Hertz
nm - nanómetros
UE - União Europeia
C-V2X - Cellular V2X
5GAA - 5G Automotive Association

Capítulo 1 Introdução

1.1 Motivação

A eficiência da rede de transportes é um assunto que preocupa governos e outros setores da sociedade, seja por motivações económicas ou ambientais. Além da variação do preço dos combustíveis derivados do petróleo, as externalidades associadas à emissão de poluentes são cada vez mais fonte de preocupação global.

Adicionalmente, os congestionamentos frequentes no acesso a cidades são origem de fenómenos de ineficiência energética e aumento das emissões de poluentes, além de causarem prejuízos económicos significativos. As emissões provenientes do tráfego urbano encontram-se intrinsecamente ligadas ao tipo de condução e às interrupções de tráfego que o veículo tem de ultrapassar. Uma condução que seja caracterizada por constantes variações de velocidade do veículo provoca um aumento tanto do consumo de combustível, como das emissões de poluentes para a atmosfera. O elevado impacto das emissões de poluentes, em particular nos grandes centros urbanos, faz com que a avaliação de políticas de gestão de tráfego ao nível energético e de emissões seja atualmente um tema de relevância científica e de discussão em todo o mundo.

Nesse sentido, pretende-se monitorizar a volatilidade da condução e percecionar de forma mais apurada como se pode prevenir e mitigar comportamentos anormais de condução e prevenir, desta forma, acidentes rodoviários, evitar consumos excessivos de combustível e evitar picos de emissões de poluentes. Tal conduzirá a uma melhoria da mobilidade e da qualidade de vida dentro dos centros urbanos.

Ao analisar as estatísticas apresentadas na figura 1 pode-se ver que os principais setores responsáveis pela emissão de gases poluentes são as Indústrias Produtoras de Energia (37.05%), o setor do Transporte (27.17%) e as Indústrias de Manufatura e Construção (14.75%) (dados de 2017)[1].

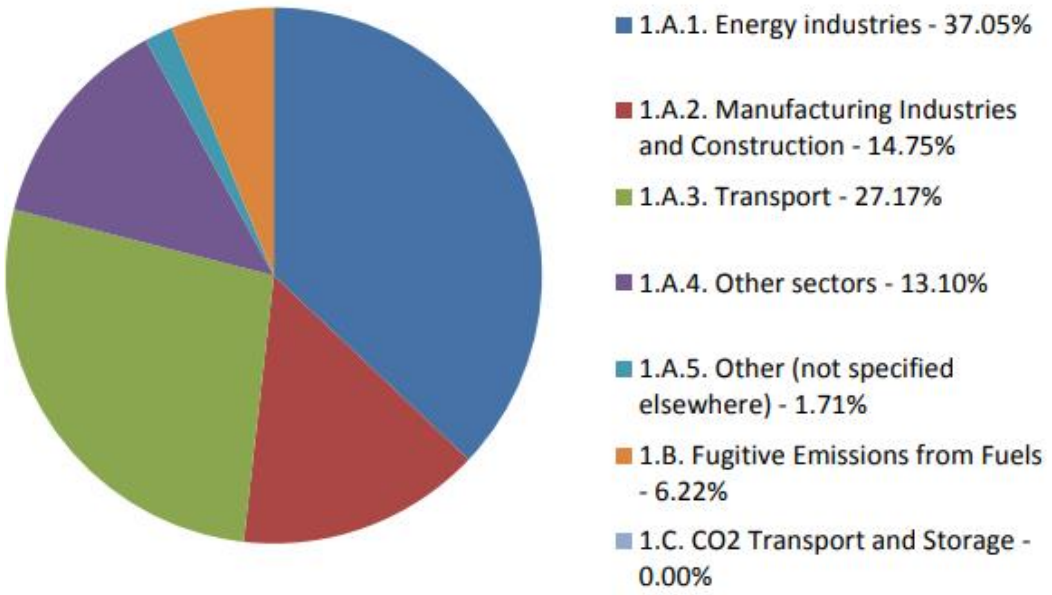


Figura 1: Emissões de gases de efeito de estufa por setor de atividade no planeta. Dados de 2017[1].

As figuras seguintes são mais direcionadas ao que o setor dos transportes diz respeito. A Figura 2 regista o aumento/diminuição das emissões de *GEE* entre 1990 e

2018 dentro da UE dividida pelos vários meios de transporte. Ao analisar-se esta figura pode-se ver que o setor da aviação regista o maior aumento nas emissões de gases de efeito de estufa. Mas também se conclui que o total das emissões para o setor dos transportes atingiu um pico no ano 2008 e tem vindo a diminuir desde então.

A Figura 3 mostra a divisão atual das emissões de *GEE* dentro da UE a 28 no que o setor dos transportes diz respeito (dados de 2019). Nesta figura vê-se que o transporte por terra vai largamente destacado com 71.7% das emissões.

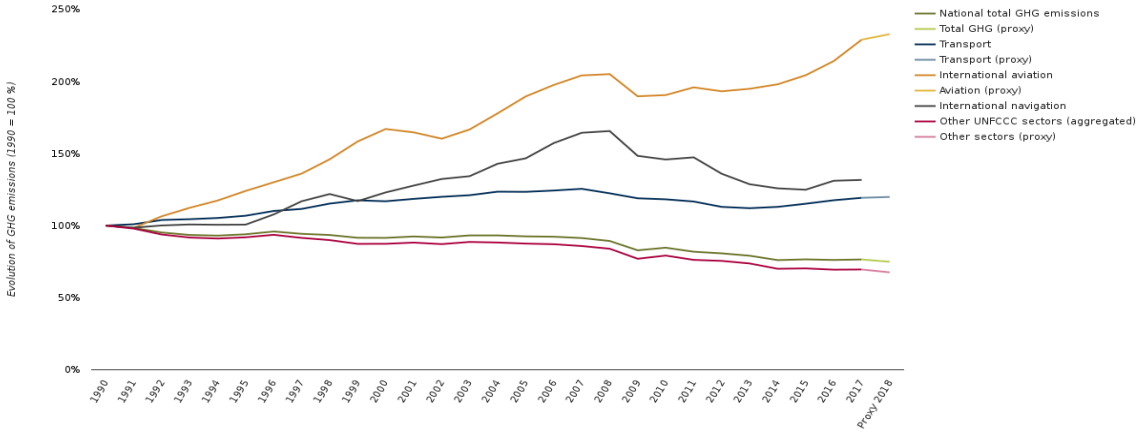


Figura 2: Evolução da emissão dos gases de efeito de estufa na União Europeia[2].

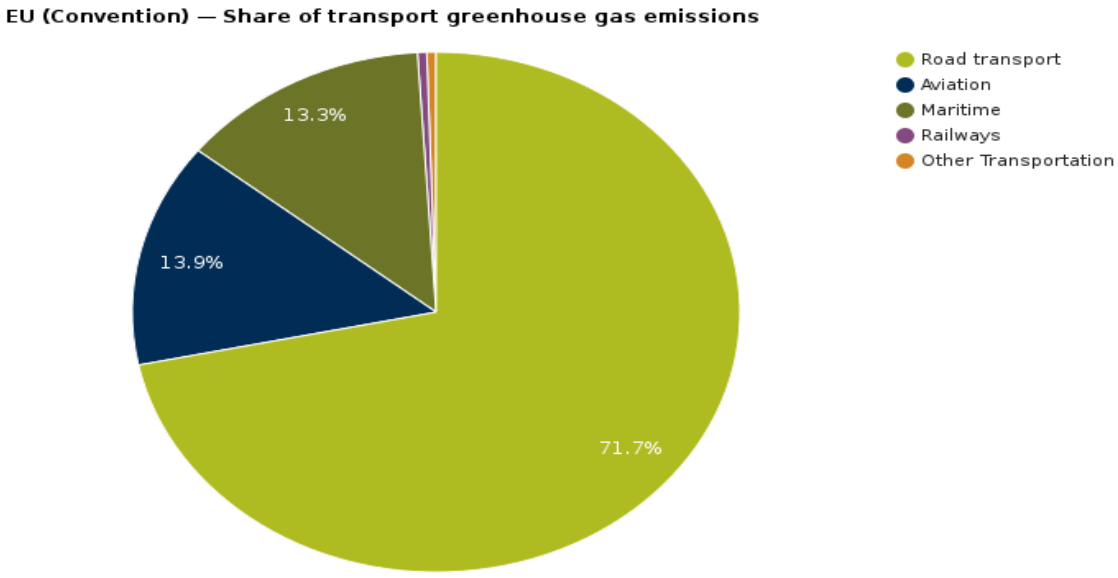


Figura 3: Emissão dos GEE por meio de transporte na EU a 28. Dados de 2019 [3] [4]

Uma das formas de minimizar o impacto negativo das emissões provocadas pelo transporte terrestre é a implementação de técnicas de Ecocondução por parte dos condutores. A poupança estimada pode chegar aos 20% [1].

Ao nível operacional Ecocondução define um conjunto de regras com vista à alteração de hábitos de condução que levam a uma condução mais eficiente. Estas são algumas técnicas utilizadas [5] [6] [7]:

- utilizar a própria inércia do veículo, acelerar e desacelerar mais suavemente;

- manter uma velocidade o mais constante possível ao longo da viagem;
- trocar as mudanças a baixas RPM's, antecipar o comportamento do transito;
- Planear um trajeto mais eficiente. Ou seja, com a mesma viagem, completar vários objetivos;
- Ter o carro com menos carga possível;
- Encher apenas meio depósito e manter o depósito pelo menos a um quarto da capacidade;
- Evitar usar o travão. O uso deste provoca a dissipação de energia já produzida no motor.

Todas estas estratégias levam a um aumento da eficiência do consumo de combustível, a uma redução de acidentes e redução do ruído provocado devido à adoção de padrões de condução mais suaves [5]. O estudo referido em [5] ajudará a perceber quanto será a diminuição esperada com a adoção de técnicas de condução mais suaves.

1.1.1 ITS (Inteligent Transportation Systems)

Os Sistemas de Transporte Inteligentes são conjuntos de aplicações que visam proporcionar serviços inovadores relacionados não só com o transporte rodoviário, mas também o transporte ferroviário, aéreo e marítimo. Incluem sistemas de navegação e podem ser utilizados por passageiros e para transporte de carga[8].

Incluem a telemática e todas as tecnologias de comunicação em veículos, entre veículos (carro-carro, por exemplo), e entre veículos e locais fixos (carro-infraestrutura).

Num sistema de transportes do tipo ITS, sensores V2I (*Vehicle-to-Infrastructure*) capturaram informação, em tempo real, vinda de algum edifício existente nas proximidades com o intuito de avisar o condutor do veículo acerca do estado da via, trânsito ou congestionamentos, acidentes, zonas de obras, ou até mesmo a existência de zonas de estacionamento [9]. A comunicação V2I é tipicamente sem fios e bidirecional. As infraestruturas presentes na via, como marcações na via, sinais de trânsito, semáforos, servem de veículo para a informação ser enviada e recebida, desde que estejam devidamente equipados com os dispositivos necessários [10].

Da mesma forma, os sistemas de supervisão de tráfego, usam a informação vinda tanto de infraestruturas como de veículos, de modo a alterar limites de velocidade, ou a ajustar sinalização de trânsito. Todo o equipamento utilizado para este tipo de sistemas é uma parte integrante de todas as iniciativas com veículos autónomos. Algumas tecnologias necessárias são [11]:

- Marcações de via Avançadas, são marcações equipadas com sensores tornando-se assim visíveis para condutores e dispositivos;
- Sinais de trânsito inteligentes equipados com matérias retrorefletores e dispositivos de Inteligência artificial aplicada a algoritmos auto-adaptativos e com capacidade de comunicação para receberem dados vindos tanto de infraestruturas como de veículos na via;
- Comunicação sem fios, do tipo DSRC ou 5G.

1.1.2 CV e AV

Veículos conectados (CV) e veículos automatizados são duas tecnologias muito distintas que podem trabalhar cooperativamente e que estão a emergir muito rapidamente e com o potencial de transformar comunidades e a maneira de como estas se deslocam [12].

Tecnologias CV permitem a que todo o tipo de veículos "dialoguem" quer com outros veículos, quer com edifícios e peões próximos, desde que tenham algum hardware

compatível com a tecnologia. O objetivo é partilhar informações importantes para a segurança e mobilidade [13].

A comunicação é feita em meio wireless como redes de telemóvel e WI-FI. Tipicamente, para comunicações de segurança, são utilizadas Dedicated Short-Range Communication. É uma tecnologia semelhante ao WI-FI, mas foi otimizada para ser rápida, segura e pouco vulnerável a interferências.

Outros tipos de comunicações podem ser transmitidos por via DSRC. É importante referir que a informação transmitida não identifica o condutor, ou a viatura. Foram tomadas precauções de ordem técnica para prevenir o rastreamento de veículos e adulteração maliciosa da informação. Os principais benefícios deste tipo de tecnologia sumarizam-se em [12]:

- Melhorias na prevenção de colisões;
- Capacidade de detetar/alertar peões ou ciclistas de veículos em aproximação. Desde que estes peões/ciclistas estejam devidamente equipados;
- Capacidade de interação com sinais de trânsito (mudança de estado, pedido de ativação);
- Capacidade de os utilizadores de transportes públicos solicitarem a paragem dum autocarro ou de serem notificados quando um autocarro está próximo, desde que devidamente equipados,

Os veículos automatizados (AV's) operam ao nível de funções críticas do veículo (direção, aceleração, travagem etc), que ocorre sem a intervenção do condutor. Av's podem utilizar sensores disponíveis no próprio veículo comummente encontrados nos veículos modernos ou também podem estar conectados.

Existem vários níveis de automação de veículos. Segue um quadro resumo desses níveis publicado pela SAE International [14]:

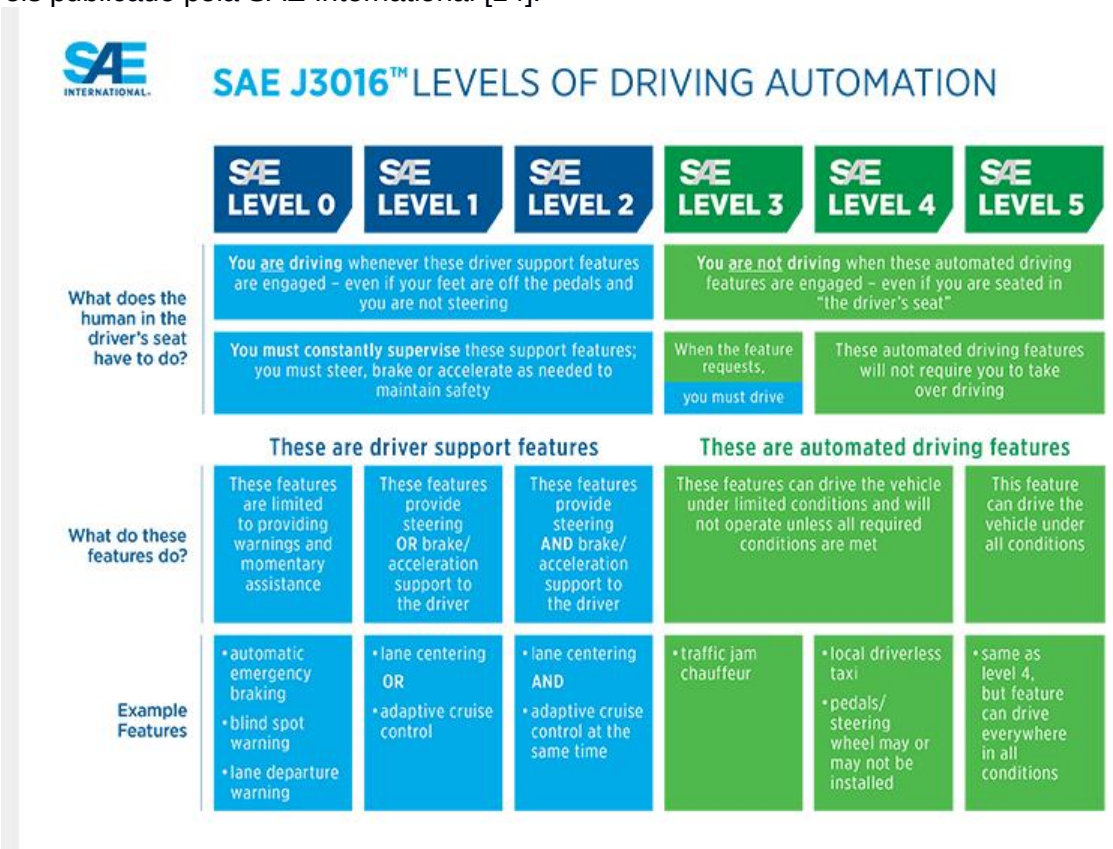


Figura 4: Níveis de automação de veículos motorizados [14].

Ao implementar a tecnologia de automatização de veículos são esperados vários benefícios, quer ao nível da segurança, quer ao nível da eficiência energética, mesmo em baixos níveis de automação.

Os benefícios esperados pela automatização de veículos são:

- Redução de acidentes provocados pela desatenção do condutor;
- Veículos automáticos de nível 4 ou 5 podem dar mobilidade a pessoas que não podem conduzir;
- Redução de custos e aumento da fiabilidade em táxis e outros serviços de transporte;
- Redução da necessidade de estacionamento. Se os veículos, depois da sua viagem, puderem ser despachados imediatamente para outro serviço deixa de existir a necessidade de estacionamento:
- Impacto ambiental reduzido devido à aplicação de uma condução mais suave;
- Mobilidade acrescida devido à maior coordenação e resposta do tráfego.

1.2 Objetivos

O objetivo principal da dissertação consiste na implementação de comunicações VIV (Vehicle-Infrastructure-Vehicle) entre um automóvel e um servidor como prova de conceito.

Para isso é necessário a implementação, a configuração de sensores, como por exemplo o *GPS* e o *LiDAR* e a implementação da comunicação entre o automóvel e um servidor remoto. O módulo *GSM* ficará encarregue da comunicação. As etapas mais importantes desta tese serão o envio de informações relativas à localização do automóvel e à distância entre veículos, a modelação de uma caixa para albergar todos os equipamentos necessários e a fixação desta caixa à parte frontal do automóvel.

1.3 Estrutura

Com o objetivo de melhorar a experiência do leitor e garantir uma melhor perceção do trabalho efetuado esta secção dá a conhecer todos os capítulos que compõem este documento. Esta dissertação está dividida em cinco capítulos principais:

O capítulo 1 é a introdução à dissertação realizada. Está subdividida numa primeira parte que é a motivação que levou ao desenvolvimento deste trabalho, depois desta são apresentados os objetivos e a estrutura do documento que acompanha a dissertação.

No capítulo 2 é apresentada a revisão bibliográfica relevante para o tema em estudo. Para além dos sensores mais utilizados e distinção entre sensores ativos e passives, nesta dissertação são também apresentados os novos paradigmas de comunicações que acompanham a tecnologia bem como alguns casos de estudo elaborados com o objetivo de testar a tecnologia em cenários o mais próximo possível a situações reais.

No capítulo 3 é exposta a metodologia adaptada aos objetivos previamente citados, focando-se um pouco mais na apresentação das principais características do hardware e o programa para o servidor, bem como um cronograma e um fluxograma da dissertação.

O capítulo 4, é a implementação da solução proposta. A primeira secção retrata um resumo do sistema com especial atenção aos protocolos de comunicação utilizados por cada dispositivo, na segunda secção é abordado em pormenor o programa escrito para o *Arduino,* na terceira secção é descrito o programa escrito para o servidor também em pormenor. Nas duas secções seguintes fala-se de como o sistema foi montado mecanicamente e é apresentado o esquema elétrico do sistema. Na secção final são apresentados os testes feitos à solução.

Finalmente, no capítulo 5 apresenta-se as conclusões, como se pode melhorar e sugere-se alguns trabalhos futuros com o intuito de dar seguimento a este projeto.

Capítulo 2: Revisão bibliográfica

O foco deste capítulo será a tecnologia existente no mercado e que seja pertinente para o tema desta Dissertação.

2.1. Comunicações e Conectividade.

A possibilidade de os veículos partilharem informações entre si e entre infraestruturas dedicadas ao controlo de tráfego faz com que exista mais informação para os veículos autónomos interagirem com o ambiente em segurança. O contrário também é verdade, também existe mais informação para sistemas de controlo enviarem a melhor informação possível aos veículos [15].

Outra função importante da conectividade será a atualização regular de softwares e mapas. Estas atualizações só serão possíveis se os veículos estivem ligados a um servidor, ou cloud, melhorando assim a experiência de conetividade e mais tarde, a experiência de autocondução.

A digitalização é algo que terá um forte impacto tanto em veículos individuais como em sistemas públicos de transporte, no controlo de tráfego, em serviços de emergência e em sistemas de informação e entretenimento. Este novo paradigma pode ser sintetizado pelo termo "*Vehicle-to-Everything*" (V2X). Este termo engloba/encapsula uma série maior de comunicações. "Vehicle-to-Vehicle" (V2V), "Vehicle-to-Network" (V2N) e "Vehicle-to-person" (V2P). A Figura 5 traduz uma representação esquemática deste novo paradigma onde a conectividade entre condutores, peões, infraestruturas e restantes utilizadores é algo constante.

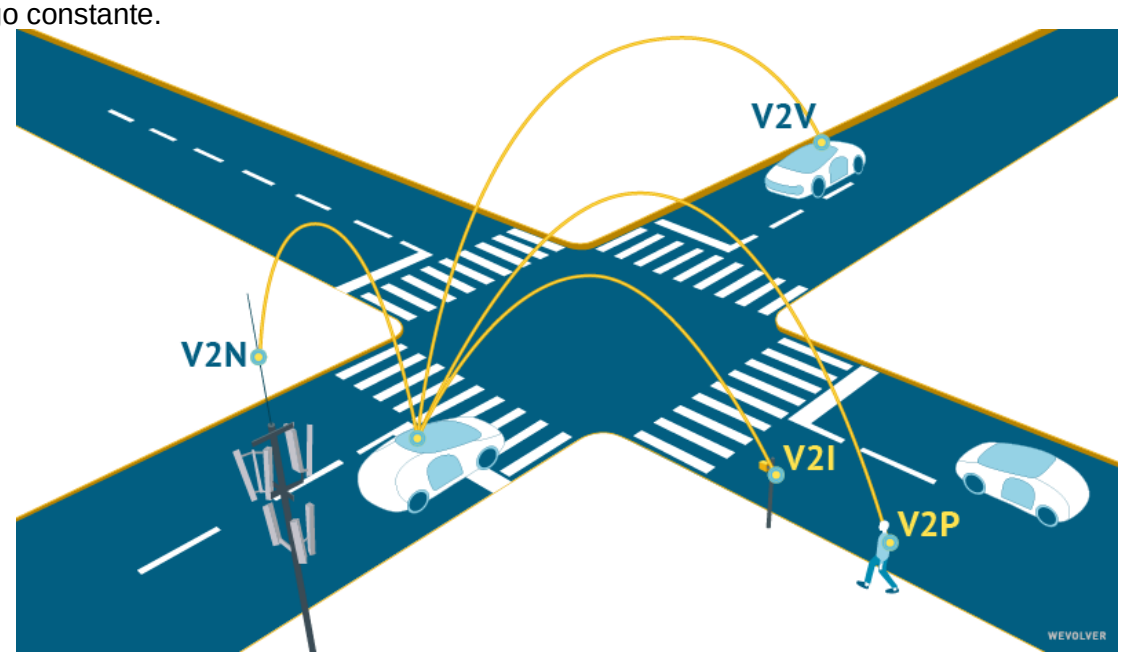


Figura 5: V2X [15]

Os intervenientes, que estão ligados por ligações sem fios, sejam estes peões, carros ou infraestruturas trocam informações entre si, em tempo real, com vista a prevenir situações potencialmente perigosas.

O principal desafio deste tipo de aplicações é a viabilidade da rede. Desta forma, a comunicação V2X é principalmente dominada por dois standards, *DSRC* e *Cellular V2X* [16].

Dedicated Short-Range Communication (DSRC), baseado no protocolo IEEE 802.11p especificado para o setor automóvel. Usa a banda nos 10MHz e nos 5.9GHz. DSRC, é uma família de comunicações standard, designada por WAVE (*Wireless Access in Vehicular Environments*), com o intuito de suportar as comunicações V2X. Tem se visto um grande esforço por parte de grandes organizações, como a IEEE (*Institute of Electrical and Eletronics Engineers*) e pela ETSI (*European Telecommunications Standards Institute*) juntamente com a SAE (*Sociaty of Automotive Engineers*) no que toca à elaboração de Standards para o protocolo DSRC. Informações como localização, velocidade, aceleração, direção entre outras, são passados por dois tipos de mensagens:

- Mensagens cooperativas, *Cooperatives Awareness Messages (CAMs)*, que são mensagens periódicas de estado do veículo com uma latência máxima de 100 ms;
- Mensagens desencadeadas por eventos, *Decentralized Environmental Notification Messages (DENM)*.

Ao usar esta tipologia são esperadas algumas desvantagens na performance devido a colisões de estações perdidas/pontos de acesso (*Access Point)* vulgarmente designadas por *Hidden Node Colisions*. A potência de ligação (*link budget*) é reduzida devido à utilização de código convoluto sem a utilização de um esquema FDM (*Frequency Division Multiplexing*). Para além disso a colocação de *RSU's (Road Side Units)* pode ser dispendiosa. Por estes motivos que foram citados, não existe um caminho claro na evolução desta tecnologia [16].

Cellular V2X (C-V2X) pode ser dividida numa tecnologia mais antiga baseada em LTE e na mais recente 5G New Radio (5G-NR) que está a ser estandardizada. No entanto, NR V2X é considerada um complemento à tecnologia LTE eV2X já existente [17].

Tal como a tecnologia DSRC, a 5G tem uma latência baixa, é utilizada tanto para as comunicações essenciais e de segurança como para informação e entretenimento, módulos rádio e *Spectrum*, com o intuito de expor problemas técnicos do veiculo aos fabricantes [17].

LTE-V2X é uma tipologia de comunicações versátil, que pode ser melhorada para habilitar serviços do tipo V2X. Foi vastamente utilizada como LTE-V no programa chinês de telecomunicações para veículos para atingir a performance desejada e para responder às oportunidades do crescente mercado das comunicações para veículos. O standard LTE-V2X introduz duas novas interfaces de rádio denominadas *Uu*, que é responsável pela comunicação V2I (*Vehicle-to-Infrastructure*) e a interface PC5 que suporta e comunicação entre veículos (V2V) [18]. Esta tecnologia utiliza um esquema FDMA (*Frequency-Division Multiple Access* ou Múltiplo Acesso por Divisão de Frequência) em onda única e suporta canais de 10-20MHz. Cada canal é dividido em *subframes*, *RB (Resource Blocks/*Blocos de Recursos*)* e subcanais. Os *subframes* são de 1 ms, os RB's são a unidade de frequência mais pequena que pode ser alocada pelo utilizador. Cerca de 180kHz (12 *subcarriers* de 15kHz). O protocolo LTE-V define subcanais como um grupo de *RB's* em cada *subframe*. O número de *RB's* em cada subcanal pode variar. Os subcanais são usados para transferir dados e informação de controlo sendo que os dados são transmitidos por *TB's ("Transport Blocks")* por *PSCCH (Physical Sidelink Shared Chanel)* e a informação de controlo *SCI (Sidelink Control Information)* a partir de *PSSCH (Physical Sidelink Control Chanel)*. Deste modo, uma transmissão de dados *TB* tem que estar associada o respetivo *SCI* [18].

A maior vantagem do 5G em relação ao DSRC é que o 5G pode utilizar infraestruturas existentes, desde que sejam feitos alguns melhoramentos, ao contrário do DSRC que necessita de novas RSUs. Isto faz com o preço de instalação do DSRC suba [17].

Cellular V2X é essencialmente dividida em dois modos. Modo 3 e o Modo 4.

No Modo 4, os veículos podem operar sem rede celular e é considerado a base das comunicações V2V porque as aplicações responsáveis pela segurança não podem ficar dependentes da existência de rede. Porém, quando os veículos estão com rede celular, a rede decide como configurar o canal V2X e informa os restantes veículos através de *Sidelink*. A mensagem de configuração inclui a frequência do canal V2X, a *resource pool* escolhida, o número de *subchanels* por *subframe* o número de *Resource Blocks (RB)* por subcanal entre outros. Quando os veículos não estão sob rede celular, utilizam um conjunto de parâmetros previamente configurados muito embora o standard em vigor (*Release 14*) não especifica um valor concreto para cada parâmetro, são usadas *resource pools* para configurar a comunicação. Estas *resource pools* indicam o número de *subframes* em cada canal que são utilizados para a comunicação V2X sendo que o resto dos *subframes* podem ser utilizados por outros serviços. Este standard, também dá a opção de dividir os *resource pools* existentes em áreas geográficas. Esta técnica é denominada *Zooning* [18].

No Modo 3 a rede celular escolhe e gere os recursos usados por veículos para as comunicações V2V, continuam a utilizar o *Sidelink* ou a comunicação V2V, no entanto, a seleção de subcanais é feita pela estação base ou pelos *eNodeB*. Por esta razão, o Modo 3 está apenas disponível quando o veículo tem cobertura. As melhorias necessárias para suportar V2X foram definidas pelo 3GPP (*Third Generation Partnership Project*), uma delas é a função de controlo V2X que é utilizada pela rede com o intuito de gerir e controlar os recursos da mesma. O funcionamento do Modo 3 é semelhante ao Modo 4, ou seja, uma mensagem tem de ser composta pelo par SCI/TB e tem de ser enviado no mesmo *subframe*. Ao contrário do Modo 4, os standards não especificam um algoritmo de gestão de recursos, cada operador pode implementar o seu e, normalmente cai numa destas categorias:

- *Dynamic Schedulling* – o veículo “pede” subcanais ao *eNodeB* para cada pacote de transmissão.
- *SPS* – o *eNodeB* reserva subcanais para as transmissões periódicas de cada veículo.

Uma área largamente estudada é o *platooning*. Onde vários veículos, tipicamente veículos pesados, autónomos ou semiautónomos, se movimentam numa espécie de comboio. A distância entre veículos é bastante mais curta que a distância de segurança, permitindo assim uma poupança de combustível de 7 a 15% [19].

Outra aplicação da tecnologia V2X demonstrada recentemente pela *Fiat Chrysler Automobiles, Continental e Qualcomm*, na ocorrência duma travagem brusca, é enviada uma mensagem aos carros que vêm atrás. Notificando-os assim duma possível situação perigosa [16].

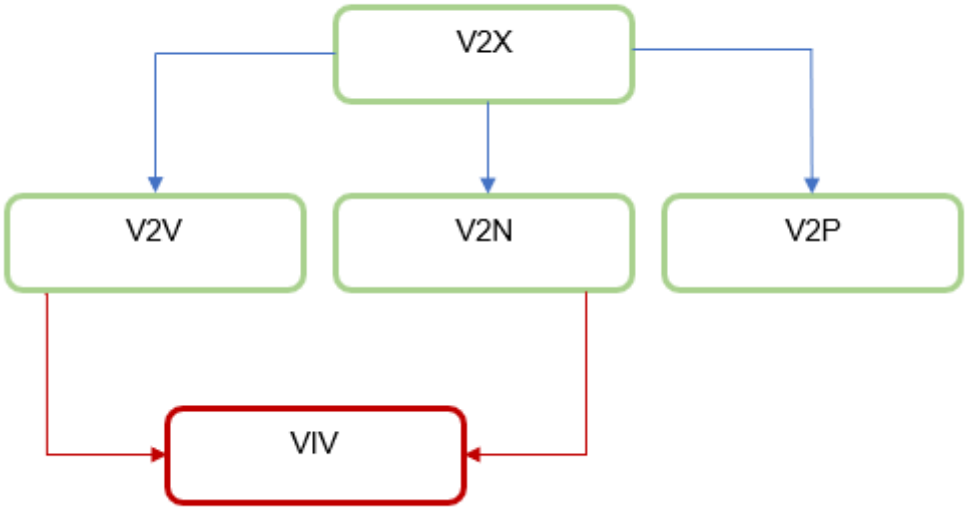


Figura 6: Fluxograma V2X

Na Figura 6 podemos ver um fluxograma da tecnologia V2X. Esta tese aborda a temática VIV que se insere entre as tecnologias V2V e V2N.

2.2. Sensores

Com o objetivo de percecionar o ambiente em redor, têm sido utilizados sensores de deteção de objetos. Os sensores podem ser divididos em sensores ativos e sensores passivos.

Os sensores passivos são aqueles que medem a energia existente no sistema como radiação, convertendo essa energia em corrente elétrica. Os sensores ativos são aqueles que necessitam de ter a sua própria fonte de energia para interagir com o sistema e convertem a energia refletida numa corrente elétrica [20].

2.2.1 Sensores Passivos

O princípio de funcionamento deste tipo de sensores reside na tradução da radiação que chega ao sensor num sinal elétrico. As câmaras digitais podem usar dois tipos de sensores, os CMOS e os *CCD* [20]*:*

- *CMOS (Complementary Metal-Oxide Semiconductor);*
- *CCD Charged Couple Device.*

As câmaras VIS (câmaras a cores) processam o sinal de radiação que "cai" no sensor dentro do intervalo do espectro do visível, ou seja, entre os 400 e os 780 nm [21]. As câmaras a cores são usadas para operações BSD (*Blind Spot Detection* ou, Deteção do Ponto Cego*),* controlo lateral, gravação de acidentes, identificação de objetos, LCA (*Lane Changing Assistant,* ou Assistência de Mudança de Via*) e* deteção de sinais de trânsito. Uma câmara de infravermelhos é usada para se obter uma imagem térmica, pode ser especialmente útil na deteção de peões ou animais na via, ou para situações onde ocorre um pico de iluminação, como por exemplo, no final de um túnel [21].

O princípio de funcionamento de ambos os sensores é o mesmo, um fotão colide com um átomo de silicone, que é um semicondutor. Quando isto acontece o átomo de silicone é levado para um estado maior de energia. Isto faz com o eletrão tenha liberdade para se movimentar no meio de outros átomos de silicone. Os pixels atuam como reservatórios de fotoeletrões, quanto mais fotões "entrarem" nestes pixels, mais energia o pixel tem. A

analogia do balde de água da chuva pode ser usada para este caso (Figura 7). Quanto mais chove, mais cheio estará o balde [22].

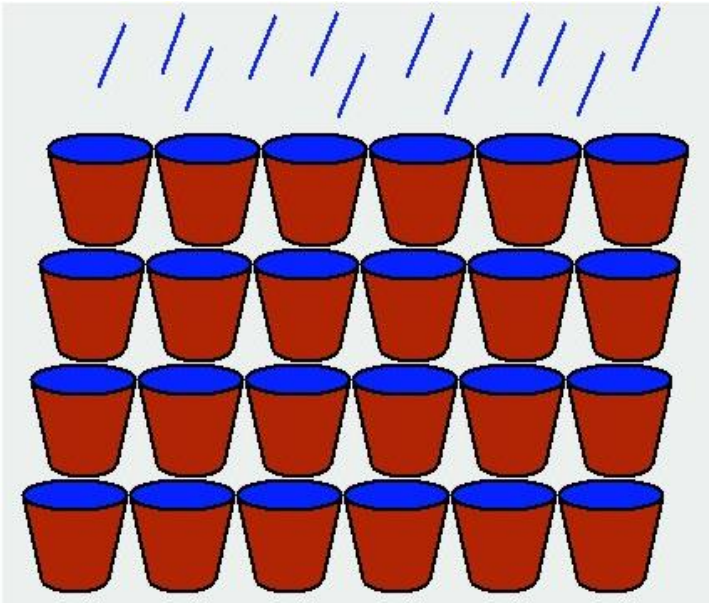


Figura 7: Analogia do balde usada para exemplificar o funcionamento dos sensores óticos[22]*.*

A diferença entre um sensor CCD e CMOS reside no modo de como medem a energia existente no pixel. Durante a leitura os sensores CCD transportam os fotoeletrões, pixel a pixel, para uma zona específica do chip onde é efetuada a amplificação e leitura do sinal. Isto faz com que cada pixel seja lido da mesma forma, tornando o CCD menos suscetível a ruído, e não haja desperdício de espaço. A desvantagem é que, o processo de leitura é mais lento e a produção deste tipo de equipamento é mais dispendioso. A leitura da informação por parte dos sensores CMOS é feita em cada pixel ao mesmo tempo, ou seja, cada pixel tem um amplificador de leitura e um conversor analógico-digital. Isto faz com que a leitura do todo o conjunto seja mais rápida comparativamente aos sensores CCD. No entanto, isto faz com que os transístores localizados em cada pixel ocupem bastante espaço o que causa uma perda de sensibilidade, como se pode observar na Figura 8 [22].

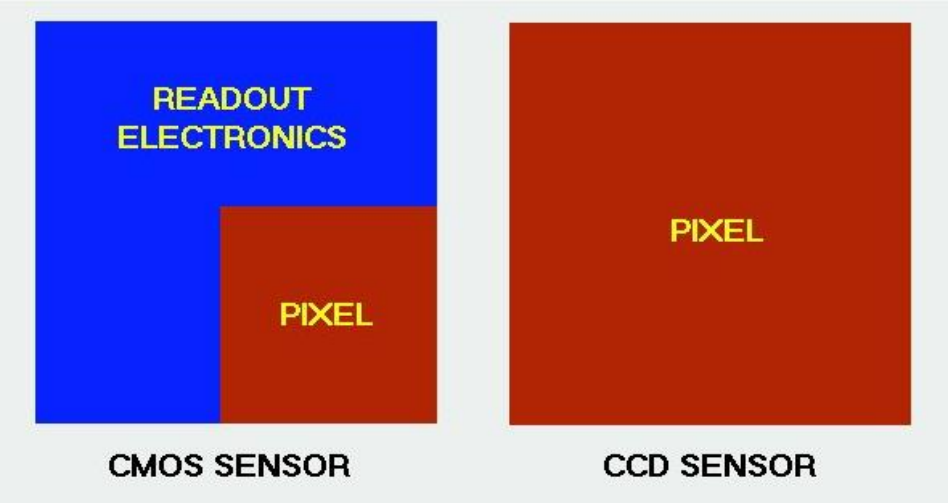


Figura 8: Comparação dos pixels de um sensor CMOS e de um sensor CCD [22].

2.2.2 Sensores ativos

Os sensores ativos baseiam-se no princípio de ToF (*Time of flight)* para percecionar o ambiente. Uma radiação infravermelha/rádio/onda sonora é emitida, quando esta embate num objeto é refletida de volta para o sensor. O sinal refletido demora um certo tempo a chegar ao sensor, este tempo é utilizado para calcular a distância do sensor ao objeto da seguinte forma:

$$d = \frac{c\ x\ ToF}{2}$$

- *d* – distância
- *c – velocidade da luz/som*
- *Tof – tempo de voo.*

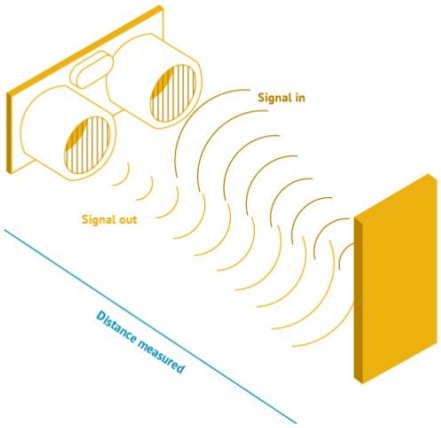


Figura 9: Diagrama TOF [15].

A frequência do sinal determina a energia utilizada pelo sistema bem com a sua precisão. Portanto, determinar o comprimento de onda correto determina a precisão do sistema.

2.2.2.1 SONAR:

Sound Navigation Ranging usa ultrassons no intervalo dos 20 kHz aos 40 kHz produzidos por uma membrana magneto-resistiva [21]. Estes sensores são utilizados para medir alturas de armazenamento de matérias-primas, em veículos são utilizados como sensores de parqueamento ou para medir curtas distâncias a velocidades reduzidas. É a tecnologia mais antiga e menos dispendiosa dos sistemas abordados. As desvantagens destes sistemas é a tendência de produzir falsos positivos quando ocorrem ressaltos e a existência de uma "zona cega" localizada entre o elemento emissor e o alcance mínimo [21].

2.2.2.2 RADAR:

Radio Detection and Ranging, trabalham em comprimentos de onda na ordem dos milímetros. É usada em aplicações militares e civis, como por exemplo na deteção de ameaças terrestres ou aéreas, sistemas balísticos, em aeroportos e em sistemas meteorológicos. No setor automóvel é utilizado o *MMW (Milimeter Wave RADAR)* que trabalha com frequências na ordem dos 24/77/79 GHz [21] . Este tipo de sistemas é capaz de percecionar a distância entre o veículo a vários objetos, a direção e a velocidade dos objetos. Este novo tipo de RADAR usa um conjunto de várias microantenas capazes de gerar um conjunto de *lobes* de deteção melhorando assim a deteção de objetos e medição da distância, algo representado a figura 8b. A Figura 10 representa um sistema RADAR disponível no mercado.

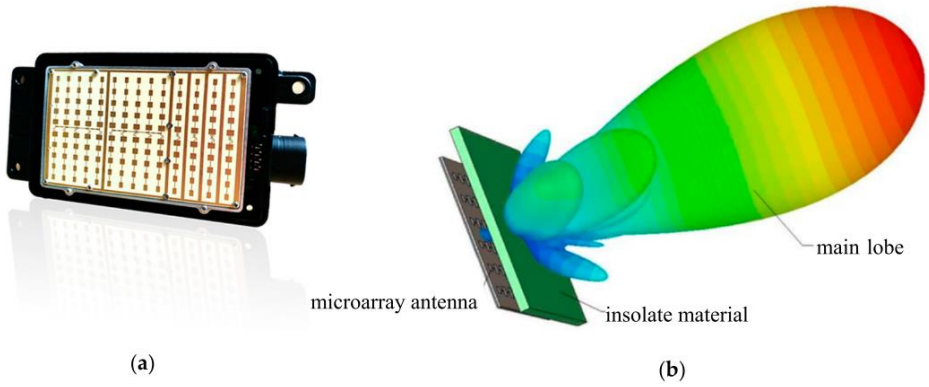


Figura 10: Sistema Milimetre-Wave RADAR CAR70 da Nanoradar. (a) Array de microantenas. (b) representação dos multi-lobes [21].

Range Milimetre-Wave RADAR é usado na deteção de zonas cegas (*BSD, Blind-Spot Detection,* ou Deteção do Ponto Cego*),* é usado na assistência à alteração de via (*LCA, Lane Crossing Assistant,* ou Assistência de Mudança de Via*),* na *FCTA (Forward Crossing Traffic Alert,* ou Alerta de trânsito em cruzamento frontal*)* ou em sistema de fusão RADAR-Vídeo. Isto porque as ondas rádio tem uma elevada penetrabilidade em todo o tipo de condições climatéricas e conseguem detetar com acuidade objetos a pouca distância a qualquer um dos lados do veículo. Algumas desvantagens deste equipamento residem na falta de precisão (embora seja um equipamento exato), reduzido campo de visão e o facto de produzir falsos positivos devido a ressaltos do sinal [21].

Outra tecnologia interessante utilizada no ramo dos carros autónomos é o *Imaging RADAR.* Esta tecnologia utiliza ondas na ordem dos 77-79GHz o que permite ao RADAR efetuar um scan de 100 graus em torno do veículo até uma distância de 300m. Esta tecnologia elimina limitações de resolução e gera uma imagem 4D de elevada resolução [15].

2.2.2.3 LIDAR

O LIDAR, *Light Detection and Ranging,* emite um sinal de luz pulsada. Estes dispositivos são capazes de emitir 50.000-200.000 pulsos por segundo com o intuito de cobrir uma área e compilar os sinais de retorno numa nuvem de pontos 3D.

Existem várias abordagens a esta tecnologia:

Mechanical Scanning LIDARS, usa um sistema de rotação para rodar o laser. Isto proporciona um largo campo de visão, porém é uma tecnologia dispendiosa e ocupa bastante espaço [23].

Microelectomechanical Mirrors (MEMS), às vezes denominados LIDAR de estado sólido ou *"LIDAR-on-a-Chip"* são sistemas baseados em LIDAR que distribuem os pulsos emitidos através de vários pequenos espelhos cuja inclinação é controlada a partir da voltagem que lhes é aplicada. Ao substituir o sistema mecânico de rotação por um sistema eletromecânico, MEMS LIDAR pode conseguir uma defleção do laser eficiente e menos dispendiosa que a alternativa exclusivamente mecânica [15].

Flash LIDARS, são um tipo de LIDAR em estado sólido onde um laser difuso é emitido iluminando toda a cena com um único flash. A luz refletida é capturada por uma rede de pequenos sensores. O maior problema desta tecnologia é a sua acuidade [15].

Phased Array LIDARS, é outro tipo de LIDAR de estado sólido onde um laser é passado numa fila de emissores que mudam a velocidade e fase do laser que está a passar por estes. O laser é apontado ao ajustar incrementalmente a fase do sinal de um emissor para o seguinte. Outra tecnologia interessante é a aplicação de metamateriais para com o objetivo de controlar a velocidade e fase do laser. Estas tecnologias ainda estão em fase de desenvolvimento [15].

O desempenho deste tipo de sensores é seriamente afetado pela humidade das superfícies onde o laser irá incidir. Outro fator importante é a interferência de outras fontes de iluminação incluindo outros LIDAR's. Temos visto uma mudança no comprimento de onda de 900 nm para 1500 nm [23], aumentado assim o alcance do *LiDAR* para 150-250m, outra desvantagem importante é o preço desta tecnologia [20].

Os *LiDAR's* têm o benefício de ter um campo de visão relativamente largo com uma cobertura potencial de 360º 3D, para além disso é um sistema mais exato que sensores passivos baseados em câmaras e é menos exigente computacionalmente [15].

De entre os sensores baseados no paradigma de *ToF* o *SONAR* é aquele que é utilizado em situações de elevada proximidade devido ao curto alcance das ondas rádio. *RADAR* é incapaz de decifrar formas complexas, no entanto é bastante eficiente em situações climatéricas adversas. Com o *LIDAR* é possível decifrar formas complexas, porém tem um alcance curto e é mais afetado pela luz ambiente ou proveniente de outros *LIDAR's* bem como nevoeiro ou poeiras. Normalmente são utilizados um ou mais sensores em conjugação e sensores ativos são utlizados muitas vezes, em conjunto com sensores passivos [15].

2.2.2.4 GNSS

Global Navigation Satellite System, GNSS, é a tecnologia mais utilizada para localização e posicionamento de veículos em terra, mar ou ar. *GNSS* calcula a posição absoluta de um recetor com base na posição de um conjunto de satélites que orbitam a aproximadamente 20,000km da superfície Terra. Estes emitem sinais com informação da sua posição, parâmetros de órbita, etc. Os recetores destes sinais utilizam esta informação para extrair a sua posição, velocidade e hora exata. O sistema mais

conhecido é o *GPS, Global Positioning System* desenvolvido pelos *EUA* nos anos 70, no entanto há vários sistemas semelhantes como demonstra a seguinte tabela.

Tabela 1: Sistemas de Posicionamento Global [23].

	GPS	*GLONASS*	*Galileo*	*Beidou*
Satélites	24	24	30	30 + 5*
Precisão	7.8 m, civil 5.9 m, militar	7.4 m, civil 4.5 m, militar	1.0 m, civil 0.01 m, *advantage*	10 m, civil 0.1 m, militar
Cobertura	Global	Global	Global	Território Chinês
Período	11 h 58 min	11 h 15 min	14 h	12 h 53 min
Altitude	26,650 km	19,100 km	23,222 km	21,150 km
Dono	EUA	Rússia	União Europeia	China

*Satélites Geoestacionários

O princípio de operação baseia-se na medição do tempo de voo do sinal emitido pelo satélite. O sistema é capaz de obter detalhes de posição e tempo com o mínimo de 4 satélites visíveis. Os sinais de satélite são condicionados por vários erros:

- Sincronização de relógios atómicos;
- Transmissão de sinal pela ionosfera e troposfera;
- Ruido nos recetores;
- Efeito de *multipath* devido a reflecções de sinal;
- Incertezas geométricas.

Diferential GNSS (DGNSS) foi desenvolvido com o intuito de aliviar erros que afetam a medição do sinal. Esta tecnologia consiste em dois recetores GNSS, uma estação terrestre e uma estação móvel conhecida com *ROVER*. A estação terrestre conhece a sua posição exata e comunica continuamente as correções do sinal para a estação móvel isto proporciona uma precisão de 0.7m a 3m para aplicações civis [21].

De seguida é abordado, um pouco mais ao pormenor, um estudo de aplicação de técnicas de Ecocondução.

O estudo de avaliação de potencialidades de Ecocondução [5] foi dividido em três partes. Foram selecionados dois carros diferentes, um Fiat 500 a gasolina e um Opel Astra a gasóleo, duas rotas e doze condutores de diferentes perfis, (idade, sexo, anos de condução). É preciso ter em atenção que este estudo foi realizado nos arredores de Madrid.

De seguida, os condutores foram levados a conduzir em turnos durante doze horas com o intuito de reunir informação suficiente.

Depois deste primeiro período de testes, os elementos do estudo foram submetidos a uma formação de 6h em técnicas de Eco-Condução. Esta formação foi levada a cabo por instrutores de uma escola de condução local, e incluía aulas teóricas e práticas.

Depois desta formação, os sujeitos foram levados a conduzir as duas mesmas rotas, sob condições idênticas.

A redução média nos consumos foi 6.3%. Devido à sua relação direta podemos concluir que a redução nas emissões de CO_2 foi de 6.5%. A discrepância de resultados entre o carro a gasolina (redução de 7.6%) e o carro a gasóleo (redução de 4.7%) pode ser explicada pelo facto de serem de segmentos ligeiramente diferentes.

Podemos também tirar algumas ilações quanto ao tipo de rota. A redução mais significativa no consumo de combustível foi numa estrada de grande capacidade (redução de 8%) devido à ocorrência de maiores velocidades.

Apesar das óbvias melhorias no combustível continuam a existir alguns desafios:

- Não se consegue garantir que, após um determinado período os condutores que receberam esta formação ou semelhante não voltem não voltem aos velhos hábitos. Alias, a redução no consumo de combustível imediatamente após uma formação em Eco-Condução pode chegar aos 25%, sendo que a longo prazo pode situar-se nos 10% [24].
- Os condutores têm mais dificuldade em aplicar as novas técnicas aprendidas quando estão sujeitos a ambientes com uma maior velocidade, como é o caso de Autoestradas.

É neste tipo de desafios que as novas tecnologias de automação e comunicação de automóveis podem entrar em cena. Um automóvel equipado com um conjunto de sensores que determinam a velocidade (GPS por exemplo) poderá ser útil para ajudar o condutor a manter uma velocidade mais estável ao longo do seu trajeto, atingindo assim uma diminuição nos consumos de combustível e consequente diminuição das emissões de GEE. Ao equipar-se um automóvel com tecnologias de comunicação VIV ou V2V, ao que este sistema se propõe ser um pequeno passo neste sentido, ajudará ao desenvolvimento de um veículo autónomo de alto nível (nível 4 ou nível 5) deixando de ser necessário a contribuição do condutor. Ao eliminar este fator humano, as diminuições descritas acima serão mais constantes. Com o advento de tecnologias de automação automóvel de alto nível também se prevê melhorias ao nível na segurança. Utilizando técnicas de alarmística com o intuito de alertar o condutor em caso de perigo (quando a distância entre veículos é demasiado curta por exemplo), ou evitando a ocorrências dessas mesmas situações. A Ecoconducao é um conjunto de técnicas que leva a uma diminuição dos consumos de combustível. É um paradigma que deve ser aplicado às novas tecnologias de comunicação e automação para tornar o exercício de condução mais eficiente. O avanço das novas tecnologias de automação e comunicação de veículos é algo que este trabalho se propõe ser um pequeno contributo.

Outra estratégia que permite uma diminuição da pegada ecológica no setor dos transportes é o “eco-routing”, que se caracteriza pela escolha de uma rota que leva ao menor consumo de combustível ou que leva a menores emissões. As escolhas são feitas por softwares dedicados para o efeito [25].

A Figura 11 é uma representação esquemática da integração de alguns dos sensores abordados neste capítulo que compõem um sistema *ADAS (Advanced Driver-Assistance Systems* ou, Sistemas Avançados de Assistência ao Condutor). Ou seja, a junção/implementação de todos ou de alguns destes sistemas, juntamente com uma unidade de processamento num automóvel ou qualquer outro veículo leva à criação de um sistema *ADAS.* Tipicamente os sistemas *ADAS* são utilizados em aplicações que promovem a segurança do veículo e seus ocupantes, no entanto já existem trabalhos em que estes sistemas são aplicados na combinação de Estratégias de Gestão de Energia (*Optimal Energy Managment Strategy)* com técnicas de Eco-Condução.

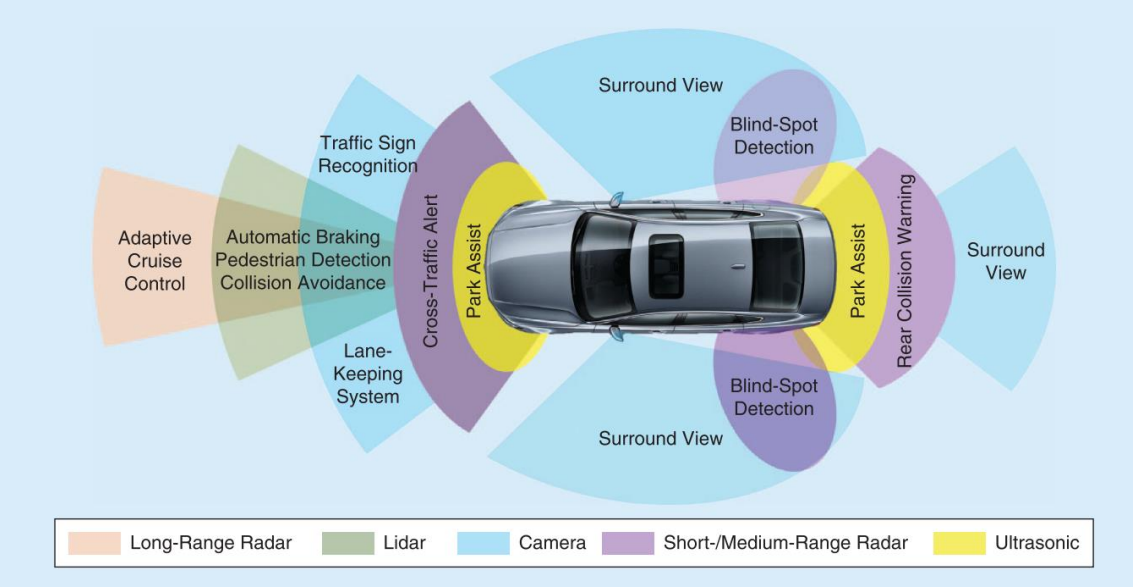


Figura 11: Carro moderno [10].

2.3 Casos de Estudo

De seguida, serão abordados alguns casos de estudo da aplicação da tecnologia V2X.

A 5G Automotive Association (5GAA) [26] identificou uma série de cenários e metodologias com vista à implementação e estudo da tecnologia C-V2X. A 5GAA define casos de estudo como "uma descrição de um conjunto de ações que um sistema tem que fazer e que produz um resultado observável e de valor para um ator". Os casos de estudo podem ser divididos em sete grupos [26]:

- Segurança;
- Gestão de Operações dos Veículos;
- Conveniência;
- Condução autónoma;
- *Platooning;*
- Eficiência de tráfego e Impacto ambiental;
- Sociedade e Comunidade.

Segurança: Este grupo inclui casos de estudo que melhoram a segurança do veículo e do condutor. Incluem, a Travagem de Emergência, Assistência de Interseção, Alerta de Colisão ou Assistente de Troca de Via. Muitos destas tecnologias são aplicáveis quer em carros autónomos quer em carros não autónomos. É esperado que muitos destes casos de estudos tenham de ser regulados e standardizados para garantir um funcionamento consistente entre diferentes fabricantes.

Gestão de Operações dos Veículos: Este grupo inclui casos de estudo que providenciam valor de gestão operacional ao fabricante. Os casos de estudo neste grupo incluem monitorização de sensores, atualização de software, suporte remoto etc. Estes casos de estudo podem ser providenciados pelo fabricante com o intuito de melhorar a eficiência de manutenção e monitorização do veículo. Não deverá ser necessário standardização.

Conveniência: Inclui casos de estudo em informação e entretenimento, Navegação Assistida, *Smart Parking* etc.

Condução autónoma: Este grupo foca-se em casos de estudo relevantes à problemática da condução autónoma (níveis 4 e 5 da SAE). Alguns exemplos são, a Tele-Operação com suporte em Realidade Aumentada para um condutor remoto, manuseamento de Mapas Dinâmicos (*update/download*), alguns casos de estudo em Segurança necessitam de interação cooperativa entre veículos para serem eficientes e seguros.

Platooning: este grupo dirige-se a casos de estudo que sejam relevantes para o *Platooning.* Desde a Gestão do Pelotão, por exemplo quando um pelotão se forma e é estabilizado, determinação da posição dentro do pelotão, dissolução do pelotão, gerenciamento da distância entre cada veículo dentro do pelotão, controlo do pelotão quando este está estável, pedidos de passagem pelo pelotão. Esta temática será mais do interesse de empresas transportadoras.

Eficiência de Tráfego e Impacto ambiental: este grupo foca-se em casos de estudo que acrescentam valor a infraestruturas ou habitantes de cidades onde os veículos irão operar. Alguns exemplos para este grupo de trabalho são, *Green Optimal Speed Advisory (GLOSA),* informações relativas a engarrafamentos e *Smart Routing.*

Sociedade e Comunidade: este grupo é focado em casos de estudo que tragam valor e interesse para a sociedade e o público em geral, por exemplo, a proteção do Utilizador Vulnerável da Via (*VRU*), serviços públicos como ambulâncias, polícia, bombeiros, emergências, governos, autoridades rodoviárias. Alguns exemplos de casos de estudo são a Aproximação de Veículos de Emergência, monitorização de pacientes e relatórios de acidentes.

De seguida, vão ser expostos alguns casos de estudo de situações que acontecem no dia-a-dia.

A descrição dos casos de estudo é da autoria da 5GAA WG1. Estes casos/exemplos podem ser compostos por vários cenários.

No primeiro caso de estudo aborda-se a Assistência de Viragem à Esquerda (Segurança, Condução Autónoma). Quando o veículo a azul, denominado *Assisted Host Vehicle* pretende virar à esquerda numa interseção com trânsito a aproximar-se vindo da via da direita (Figura 12) ou da via da esquerda (Figura 13).

Os possíveis embates estão assinalados com um círculo a vermelho. O objetivo será sempre de evitar o embate. Admite-se dois cenários possíveis.

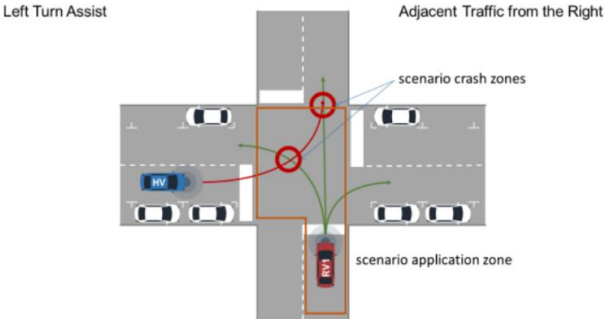


Figura 12: Viragem à esquerda. Caso de estudo 1 [26]

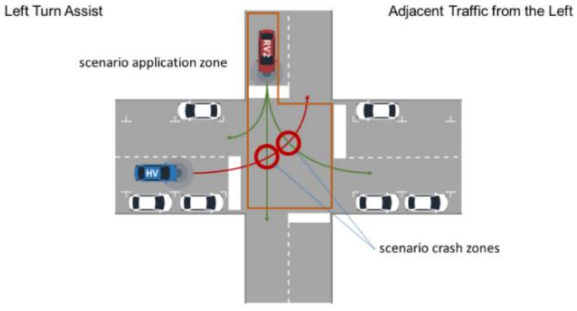


Figura 13: Viragem à esquerda. Caso de estudo 2 [26]

Tabela 2: Descrição geral dos cenários do caso de estudo Viragem à Esquerda [26].

Cenário 1	Veículos autónomos trocam mensagens cooperativas (CAM's). Não são trocadas informações à cerca de trajetórias futuras. Em vez disso o risco de colisão é calculado a partir da informação recolhida no passado e no presente. Um aviso ao condutor é mostrado em caso de perigo.
Cenário 2	Neste cenário, níveis mais altos de automação são considerados. Veículos autónomos trocam trajetórias futuras previamente planeadas. Com esta troca de informação, é possível fazer estimativas mais exatas à cerca de possíveis colisões.

Com base nestes dois cenários, requisitos de nível de serviço são redigidos.

Tabela 3: Descrição detalhada do cenário 1. Assistência de Viragem à esquerda [26].

Nível de serviço	Unidades	Valor	Explicação/Raciocínio/Background
Alcance	[m]	350	É assumido o máximo alcance de comunicação. O que permite 5s de tempo de reação à velocidade (ver secção da velocidade).
Informação solicitada/gerada	Qualidade/ necessidade de informação	300 bytes/ mensagem	Baseado em mensagens cooperativas. (CAM's)
Latência	[ms]	100	Latência associada à CAM's
Fiabilidade	%	90	Para mensagens cooperativas sem retransmissão, uma fiabilidade de 90% é suficiente para garantir uma probabilidade >5% de falharem duas mensagens consecutivas.
Velocidade	[m/s]	33.3	Situações críticas podem ocorrer em interseções rurais onde o *Remote Vehicle* pode transitar a 120km/h, e o *Host Vehicle,* desacelera a partir dos 120km/h. 120km/h é a velocidade máxima.
Densidade de veículos	[veículos/km^2]	1500	Mais provável de acontecer em áreas metropolitanas pouco densas, onde a velocidade nas interceções ronda os 50km/h e onde existem semáforos.
Posicionamento	[m]	1.5 (3σ)	O cenário mais provável situar-se-á em zonas rurais onde são

			praticadas velocidades mais elevadas.
Interoperabilidade/ Regulamentação/ Standardização	[Sim/Não]	Sim/Sim/Sim	Com o intuito de posicionar o veículo na sua via de forma precisa, uma precisão de 1m deve ser obrigatória

Tabela 4: Descrição detalhada do cenário 2. Assistência de Viragem à esquerda [26].

Nível de serviço	Unidades	Valor	Explicação/Raciocínio/Background
Alcance	[m]	350	É assumido o máximo alcance de comunicação. O que permite 5s de tempo de reação à velocidade (ver secção da velocidade).
Informação solicitada/gerada	Qualidade/necessidade de informação	1000 bytes/ mensagem	As trajetórias pretendidas têm de ser enviadas aos RV's porque é desta maneira que se determina se há a probabilidade de uma colisão eminente. Logo, será necessária uma carga maior de transmissão de dados.
Latência	[ms]	10	Dependendo da implementação podem ser utilizadas mensagens de perigo poucos instantes antes da manobra de viragem. Por este motivo, é que é necessária uma latência tão baixa.
Fiabilidade	%	99.9	É necessária uma fiabilidade tão elevada para garantir erro-zero na transmissão.
Velocidade	[m/s]	33.3	Situações críticas podem ocorrer em interseções rurais onde o *Remote Vehicle* pode transitar a 120km/h, e o *Host Vehicle,* desacelera a partir dos 120km/h. 120km/h é a velocidade máxima.
Densidade de veículos	[veículos/km^2]	1500	Mais provável de acontecer em áreas metropolitanas pouco densas, onde a velocidade nas interceções ronda os 50km/h e onde existem semáforos.
Posicionamento	[m]	1.5 (3σ)	Com o intuito de posicionar o veículo na sua via de forma precisa, uma precisão de 1m

			deve ser obrigatória
Interoperabilidade/ Regulamentação/ Standardização	[sim/não]	Sim/Sim/Sim	Interoperabilidade entre diferentes fabricantes é necessária. Deve ser regulado que periodicamente todo o veículo tem de tornar a sua presença conhecida (como um *Broadcast)*. A standardização do formato das trajetórias é necessária.

Na *Figura 14* aborda-se a Assistência no Movimento de Interseção (Segurança, Condução Autónoma). O *Host Vehicle* que aparece a azul está parado no Stop. Este veículo é alertado quando é inseguro prosseguir a marcha na interceção (por exemplo, o HV recebe alertas quando existe trânsito a aproximar-se vindo da esquerda ou da direita).

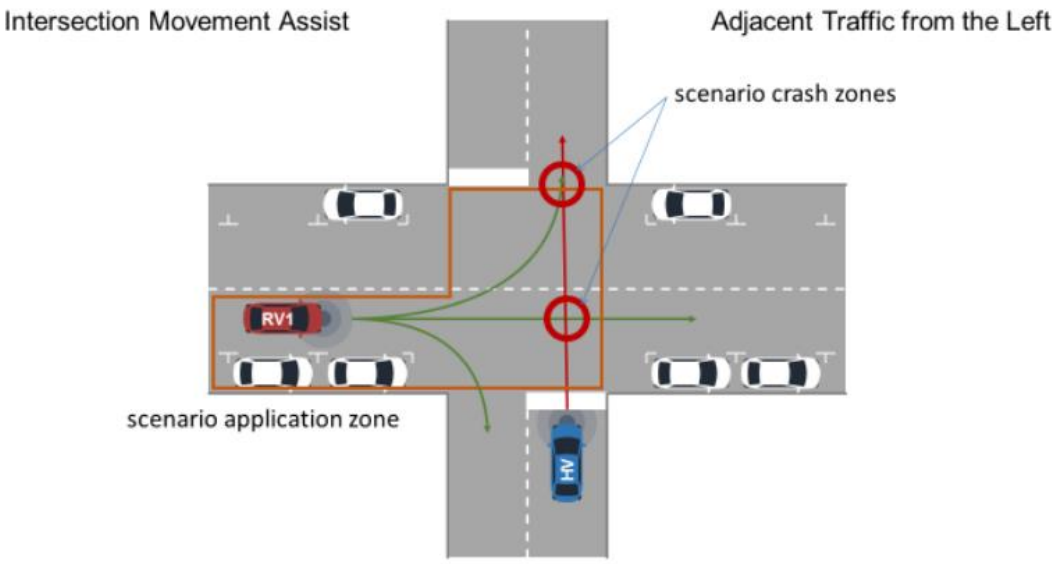


Figura 14: Assistência no movimento de interseção [26].

Cenário 1: dois veículos estão a aproximar-se duma interseção. Os veículos determinam o risco de colisão baseado na trajetória estimada dos veículos.

Tabela 5: Descrição detalhada. Assistência no movimento de interseção [26].

Nível de serviço	Unidades	Valor	Explicação/Raciocínio/Background
Alcance	[m]	Min 100	Distância de travagem a 0.4g a partir de 100km/h.
Informação solicitada/gerada	Qualidade/ necessidade de informação	Mensagens cooperativas (CAM's) 300 bytes/ mensagem	Calcula a trajetória baseada na troca de dados. Alterações na cinemática dos veículos pode fazer com que esta informação seja partilhada periodicamente. Respeitando sempre os limites de Latência.

Latência	[ms]	100	Deve ficar abaixo dos 100ms. Semelhante a outros sistemas ADAS.
Fiabilidade	%	Alta / 99.9	Tem de ser fiável o suficiente para permitir o cálculo da trajetória que evite colisões.
Velocidade	[m/s]	33.3	Assume uma velocidade máxima de 120km/h
Densidade de veículos	[veículos/km^2]	12000	Densidade máxima assumida em áreas urbanas.
Posicionamento	[m]	1.5 (3σ)	Necessário para cálculo da trajetória e estimativa do risco de colisão.
Interoperabilidade/ Regulamentação/ Standardização	[sim/não]	Sim/Sim/Sim	Interoperabilidade entre diferentes fabricantes é necessária e deve ser garantida através da standardização.

Para além do desenvolvimento de algo parecido com um Sistemas *ADAS,* é também necessário que o sistema seja testado num meio parecido ao real. Estes são alguns cenários que a 5GAA considera importantes. Um ponto de partida para esta dissertação foi teste com um automóvel em trânsito. Não se aplicaram nenhum destes cenários, mas considera-se útil serem aplicados em trabalhos futuros. Com base nas temáticas abordadas nesta revisão bibliográfica foi considerado que um sensor *GPS*, um *LiDAR* seriam os sensores que ficariam encarregues de percecionar o ambiente ao redor do automóvel. O *LiDAR* mediria a distância ao veículo da frente, quando o carro se aproximasse demasiado, seria enviado um alerta, por meio de um sinal luminoso. O *GPS* é útil para saber onde essa situação perigosa tinha ocorrido. O centro de processamento seria o microcontrolador do tipo *Arduino.* Apesar de não ter um processador muito potente (comparativamente aos computadores domésticos por exemplo) seria suficiente para tratar as informações vindas dos sensores. A tarefa de comunicação com o servidor ficaria encarregue do módulo SIM900. Quando o centro de processamento detetar uma situação perigosa, deverá emitir um alerta e alertar o servidor deste facto. Estes cenários deverão ficar guardados em base de dados.

Capítulo 3 Metodologia

Neste capítulo, será exposta a tecnologia utilizada nesta dissertação.

Para melhor entender as fases deste trabalho foi elaborado um fluxograma, que se pode ver na Figura 15, e um cronograma exposto na Tabela 6.

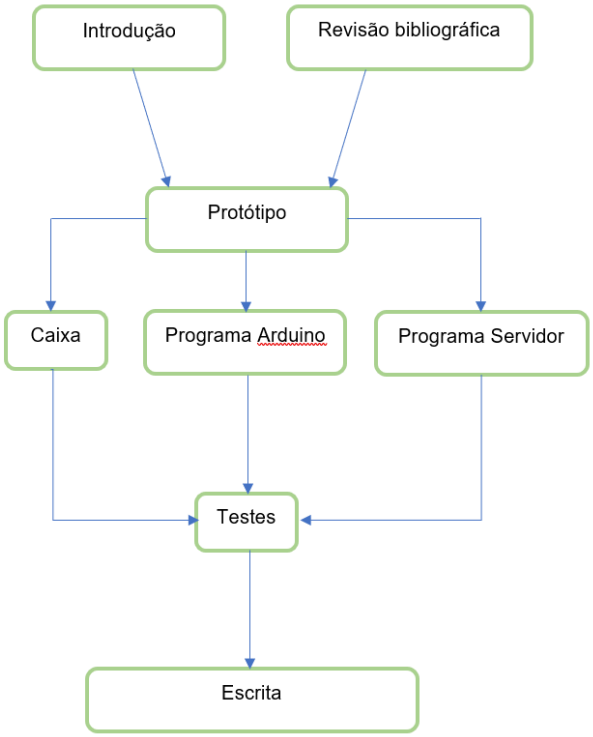


Figura 15: Fluxograma da dissertação.

Tabela 6: Cronograma.

Tarefa	Janeiro 2020	Fevereiro 2020	Março 2020	Abril 2020	Maio 2020	Junho 2020	Julho 2020	Agosto 2020	Setembro 2020	Outubro 2020	Novembro 2020	Dezembro 2020	Janeiro 2021
1													
2													
3													
4													
5													
6													

Tarefa 1: Revisão bibliográfica e introdução
Tarefa 2: Configuração, programação e implementação de sensores.
Tarefa 3: Configuração do servidor/firewall
Tarefa 4: Testes
Tarefa 5: Solução proposta. E Implementação da Solução proposta
Tarefa 6: Escrita.

3.1 Descrição geral do sistema

O objetivo principal desta dissertação consiste na implementação dum protocolo do tipo VIV como prova de conceito, onde são trocadas informações entre um servidor e um automóvel, nomeadamente a distância do automóvel ao veículo da frente.

Para isto o seguinte hardware foi selecionado:

- Arduino Mega do tipo 2560;
- Sensor *GPS* do tipo *Shield* da *Adafruit*;
- Módulo de comunicação *GSM, SIM900*;
- Sensor *LiDAR Lite V3* da *Garmin*.

3.2 Descrição geral do Arduino Mega

O *Arduino Mega 2560* é um microcontrolador baseado no chip *ATMega2560*. Está equipado com 54 pinos digitais *input/ouput* (destes pinos 15 podem ser utilizados como outputs PWM), 16 entradas analógicas 4 portas série, um cristal oscilador de 16Mhz e uma porta USB [27]. Para este trabalho foi também utilizado o *Arduino Uno*, que constitui um microcontrolador menos potente que o *Arduino Mega 2560*. Este teve de ser substituído devido ao programa ocupar demasiado espaço destinado a variáveis globais. Depois da substituição do microcontrolador, o programa teve de ser adaptado para o *Arduino Mega*.

Fui utilizado uma versão compatível do *Arduino Mega 2560*, denominado *Funduino Mega*, como o que está representado na Figura 16, por este ser mais económico e estar imediatamente disponível. A Figura 17 representa o esquema de ligações internas dum *Arduino Mega 2560*.

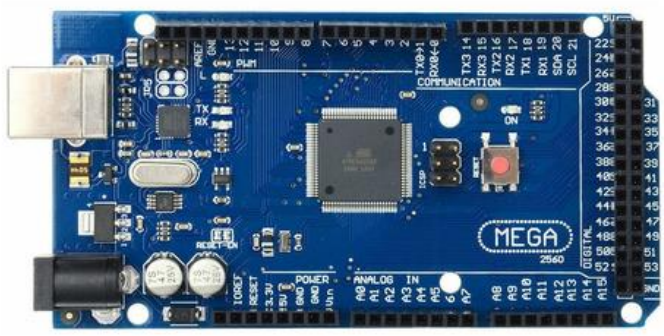


Figura 16: Funduino Mega [28].

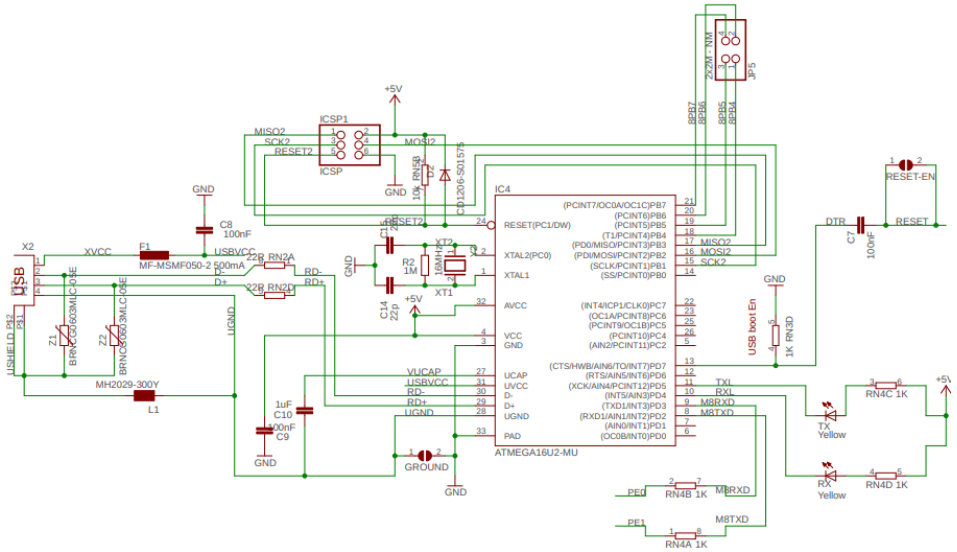


Figura 17: Diagrama de ligações do Arduino Mega [27].

3.3 Descrição geral do Módulo GPS

O *GPS Shield da Adafruit* está preparado para trabalhar com microcontroladores do tipo *Arduino UNO* ou *Leonardo*, mas funciona bem com o *Arduino Mega* utilizado neste projeto. Está desenhado para escrever dados num cartão SD (não implementado).

Principais características [29]:

- Sensibilidade -165dBm, updates de 10Hz e 66 canais;
- Consumo de 20mA;
- *Shield* testado para microcontroladores do tipo *Arduino*;
- Equipado com uma slot para cartão de memória *MicroSD*;
- Equipado com uma bateria do tipo *RTC*, o que dá ao *Shield* uma capacidade de backup de dados até 7 anos;
- Registo de dados para memória flash;
- Output PPS quando tem um *fix*;
- Equipado com uma antena interna mais um conector *u.FL* para ser utilizado com uma antena externa (não necessário para este projeto dado que é para ser utilizado ao ar livre);
- Equipado com *LED's* indicadores de *Fix, Power* e Pin 13;
- Grande área de prototipagem.

As figuras seguintes representam o aspeto geral do módulo *GPS* com uma antena externa (Figura 18) e o esquema de ligações do mesmo módulo (Figura 19).

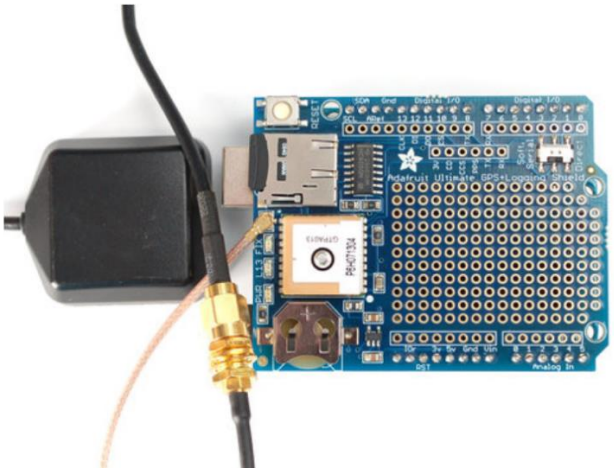


Figura 18: Adafruit GPS Shield com Antena externa [29].

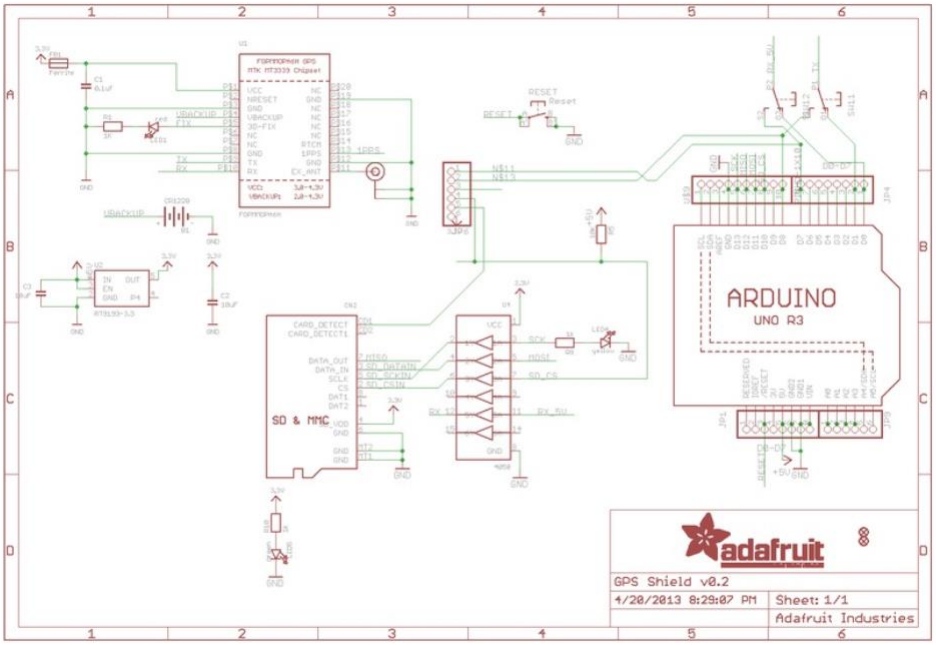


Figura 19: Esquema de ligações do GPS Shield [29].

3.4 Descrição Geral do Módulo SIM900

O *SIM900* é um modem do tipo *GSM/GPRS* com a capacidades semelhantes de um telemóvel normal. Consegue fazer e receber telefonemas, mandar e receber *SMS* e conectar-se à internet por via *GPRS* ou *TCP/IP*. Este dispositivo suporta *Quad-Band* fazendo com que possa ser utilizado em qualquer parte do mundo.

Algumas das características mais importantes deste equipamento vão ser enumeradas em baixo [30]:

- Suporta *Quad-Band*: *GSM850, EGSM900, DCS1800 e PCS1900*;
- Consegue conectar-se a qualquer rede *GSM* global com qualquer cartão *SIM 2G*;
- Faz e recebe chamadas utilizando um auricular e um microfone;
- Envia e recebe *SMS*;
- Envia e recebe dados *GPRS* (*TCP/IP, HTTP* etc);
- Baseado em comandos *AT* enviados por porta série;
- Possui um conector *u.FL* e *SMA* para serem utilizados com uma antena externa;
- Aceita um cartão *SIM* de tamanho normal;
- Possui três *LED's* indicadores de estado:
    - *PWR*, se este *LED* estiver ligado o modem está a ser alimentado de forma correta;
    - *Status*, se este *LED* estiver ligado o modem está em modo de trabalho;
    - *Netlight*, este *LED* pisca com diferentes cadências dependo do estado da ligação à rede:
        - Se estiver desligado, o chip não está a trabalhar;
        - 64ms ligado, 800ms desligado, o *SIM900* está a trabalhar, mas não está ligado a nenhuma rede;

▪ 64ms ligado, 3 segundos desligado, o *SIM900* está registado a uma rede celular pode enviar/receber dados de voz ou *SMS* ou *GPRS*;

▪ 64ms ligado, 300ms desligado, a ligação *GPRS* está ativa.

As figuras seguintes representam o aspeto geral do módulo SIM 900. Vista dianteira (Figura 20), vista traseira (Figura 21).

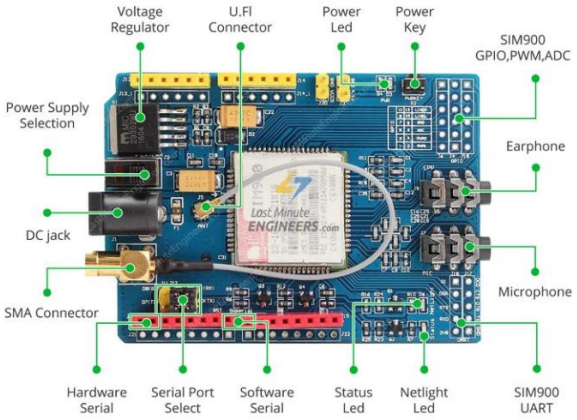


Figura 20: SIM900 [30]*. 1*

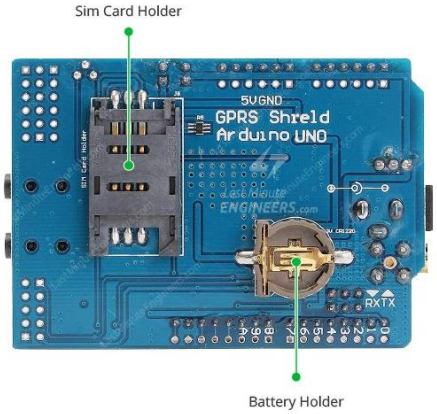


Figura 21: SIM900 [30]*. 2*

3.5 Descrição geral do LiDAR

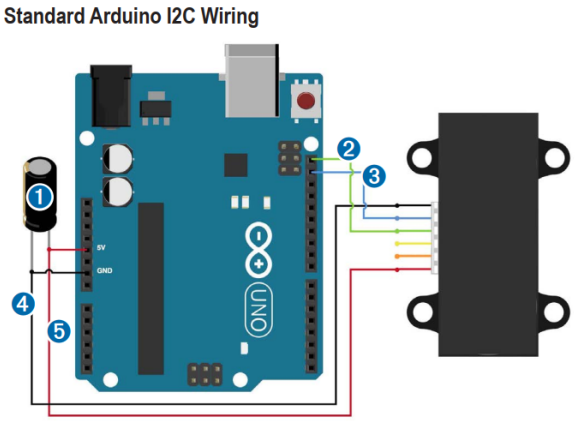
O sensor *LiDAR Lite V3* mede a distância através do cálculo do tempo de atraso entre a transmissão de um feixe de luz infravermelha e a sua receção após o sinal ser refletido num alvo, dado que o feixe de luz viaja à velocidade da luz. Depois do sinal ser processado, a distância calculada pode ser transmitida ao centro de processamento (neste caso o *Arduino Mega*) de duas formas [31]:

- Protocolo I^2C;
- *Pulse With Modulation, PWM* (não implementado).

O I^2C é um protocolo do tipo *master/slave* onde são utilizados dois fios para estabelecer uma conexão e enviar e receber dados. Os dados são transmitidos numa sequência de nove pulsos do relógio através da linha *SDA* quando a linha *SCL* está a *HIGH*. O nono pulso é o bit de *acknowledge*.

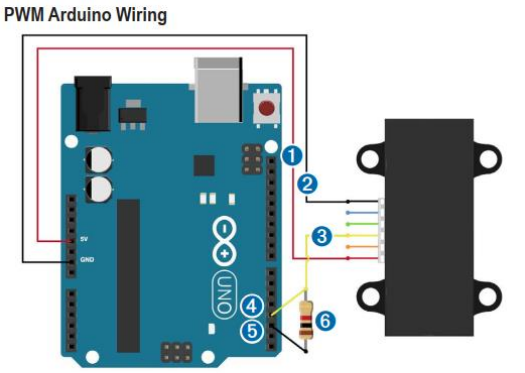
O *PWM* é uma técnica de transmissão de dados onde a largura dos próprios pulsos corresponde a valores específicos, codificados numa extremidade e descodificados noutra extremidade. A inclusão de um *clock* não é necessária [31].

A Figura 22 representa o esquema de ligações necessário para estabelecer uma ligação por I^2C. A Figura 23 representa o esquema de ligações necessário para estabelecer uma ligação por *PWM.*



Item	Description	Notes
1	680µF electrolytic capacitor	You must observe the correct polarity when installing the capacitor.
2	I2C SCA connection	Green wire
3	I2C SDA connection	Blue wire
4	Power ground (-) connection	Black wire
5	5 Vdc power (+) connection	Red wire The sensor operates at 4.75 through 5.5 Vdc, with a max. of 6 Vdc.

Figura 22: LIDAR I²C [31].



Item	Description	Notes
1	5 Vdc power (+) connection	Red wire The sensor operates at 4.75 through 5.5 Vdc, with a max. of 6 Vdc.
2	Power ground (-) connection	Black Wire
3	Mode-control connection	Yellow wire
4	Monitor pin on microcontroller	Connect one side of the resistor to the mode-control connection on the device, and to a monitoring pin on your microcontroller.
5	Trigger pin on microcontroller	Connect the other side of the resistor to the trigger pin on your microcontroller.
6	1kΩ resistor	

Figura 23: LIDAR PWM [31]

3.6 Descrição Geral do MAMP

Para este trabalho foi utilizado um servidor com a capacidade de escrever dados numa base de dados em *MySQL*, o *MAMP*. O *MAMP* é um pacote de instalação dum servidor *Apache* e/ou *Nginx* com um servidor *MySQL*. Várias linguagens de *web development* são permitidas [32]. Neste trabalho foram utilizadas *HTML* com *PHP* e *MySQL* para lidar com a base de dados. Para o servidor Apache a porta 8080 foi utilizada, para o servidor *MySQL* a porta 3306 foi utilizada (porta standard da ferramenta). A Figura 24 indica a painel inicial do *MAMP,* a Figura 25 indica o painel de configuração de portas.

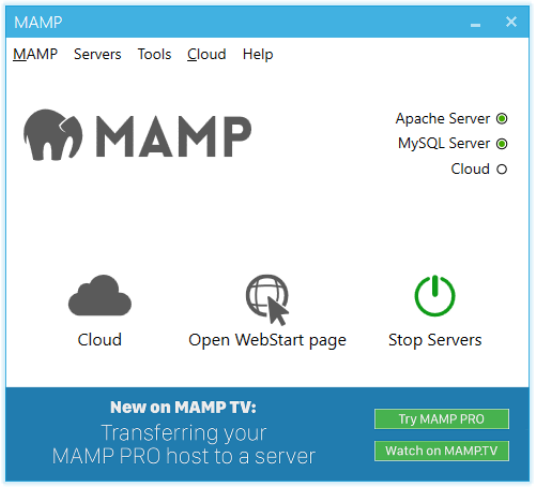


Figura 24: MAMP, página inicial.

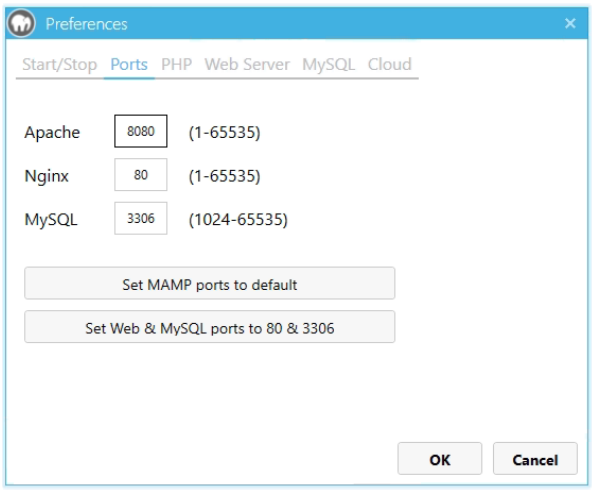


Figura 25: MAMP, configuração de portas.

Foi também adquirida uma powerbank de 2500mA para alimentar o sistema.

Capítulo 4: Implementação da solução proposta

Neste capítulo, será exposta a implementação da solução com o hardware.

Para melhor entender a metodologia deste trabalho foi elaborado um fluxograma desta dissertação que será apresentado na figura seguinte.

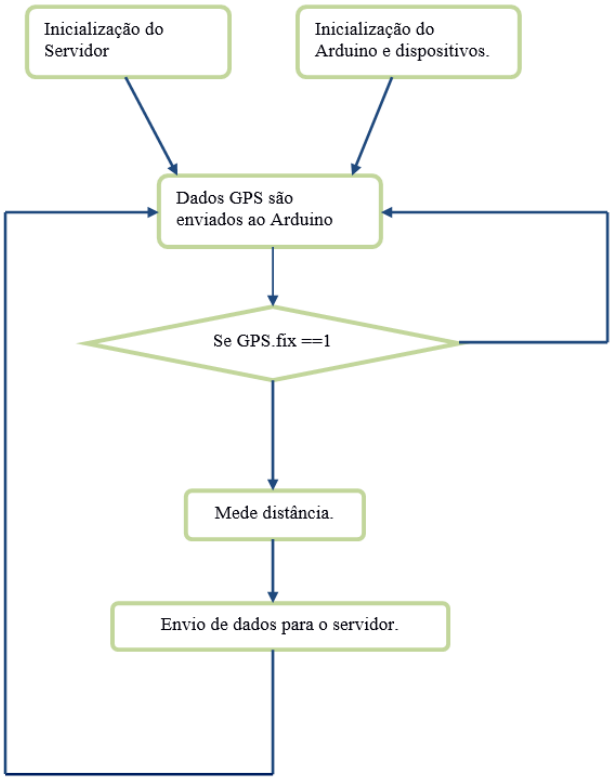


Figura 26: Fluxograma.

4.1 Resumo do sistema

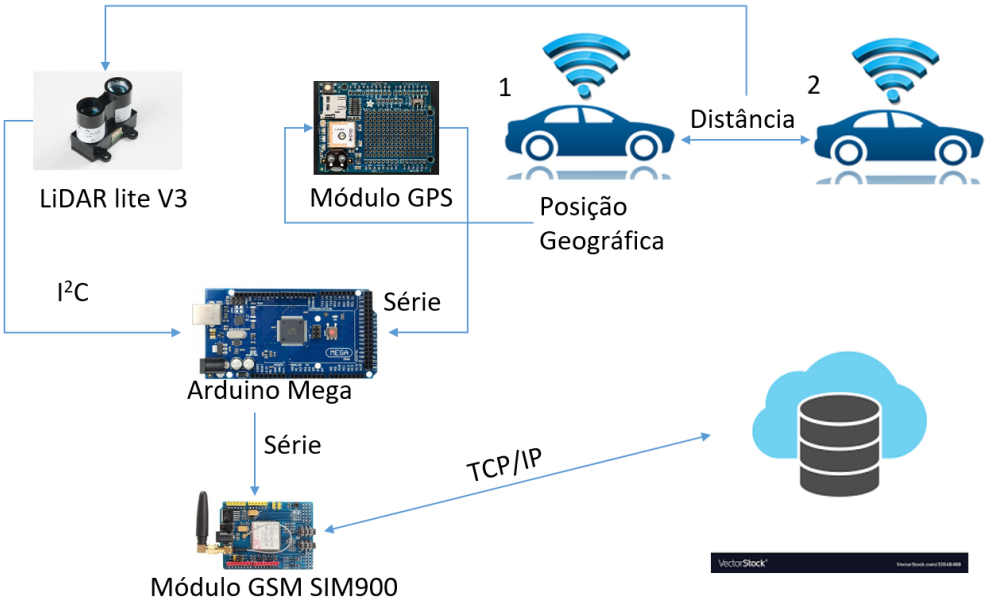


Figura 27: Resumo do sistema

O objetivo deste trabalho é de medir a distância de carro a carro, tirar a posição geográfica e enviar esta informação para um servidor. A posição geográfica é obtida a partir do *Módulo GPS* juntamente com outras informações. Como a altitude, o número de satélites, a velocidade e o ângulo de viagem.

As informações vindas do *GPS* são enviadas para o *Arduino* por porta série (comunicação *RS232*) para serem processadas. A comunicação *RS232* tinha por objetivo permitir a ligação de equipamentos digitais a redes públicas analógicas. Este protocolo foi aprovado pela *Electronic Industries Association – EIA,* e recebeu a designação *EIA232C* que tem sofrido várias atualizações ao longo dos anos [33].

Os pinos *Tx* e *Rx* são usados para enviar e receber dados respetivamente. A comunicação RS232 pode ser síncrona ou assíncrona. Na comunicação assíncrona o sinal gerado pelo emissor depende apenas do seu relógio interno. Numa comunicação síncrona tanto o emissor como o recetor dependem do mesmo sinal de relógio para determinar os instantes da transição do sinal de dados e dessa forma saberem em que instante de tempo amostrar o sinal. Neste modo é necessário utilizar mais dois pinos. O *TXClock e o RxClock* para sincronizar o relógio de ambos os dispositivos. Neste projeto foi utilizada a comunicação série assíncrona. Este tipo de comunicação só permite a ligação entre dois pontos apenas e são utilizados fios de cobre para completar a ligação. Os sinais são gerados a partir de tensões *TTL* (*Transistor Transistor Logic)*, só podem assumir valores entre os 0 e os *5V*. A mensagem é composta por um *start bit* seguido de até 8 bits de dados, um bit de paridade (facultativo) e um *stop bit.* É possível escolher entre vários valores de *baudrate (bits/s).* Quanto maior for esse *baudrate* mais rápida será a mensagem. As taxas de transferência: 150, 300, 1200, 4800, 9600, 19200, 38400, 57600, 115200 bits/s entre outras estão previstas para este protocolo [33].

Depois dos dados do *GPS* terem sido recolhidos é necessário tirar a distância carro a carro com a ajuda do *LiDAR*. O *LiDAR* comunica com o *Arduino* por *I^2C Inter Integrated Circuit.* Este protocolo foi proposto pela *Philips Eletronics* para permitir a troca de dados entre circuitos integrados. Este protocolo utiliza dois condutores para ligar dois ou mais dispositivos. A linha *Serial Clock - SCL* usada para transmitir o sinal de relógio e o condutor *Serial Data – SDA* para enviar e recebe dados (bidirecional). As linhas *SDA* e *SCL* apresentam uma tensão de *5V*. Quando um dispositivo quiser enviar um bit tem de forçar a linha *SDA* a zero durante um intervalo de tempo. A Figura 28 representa um esquema do protocolo *I^2C.*

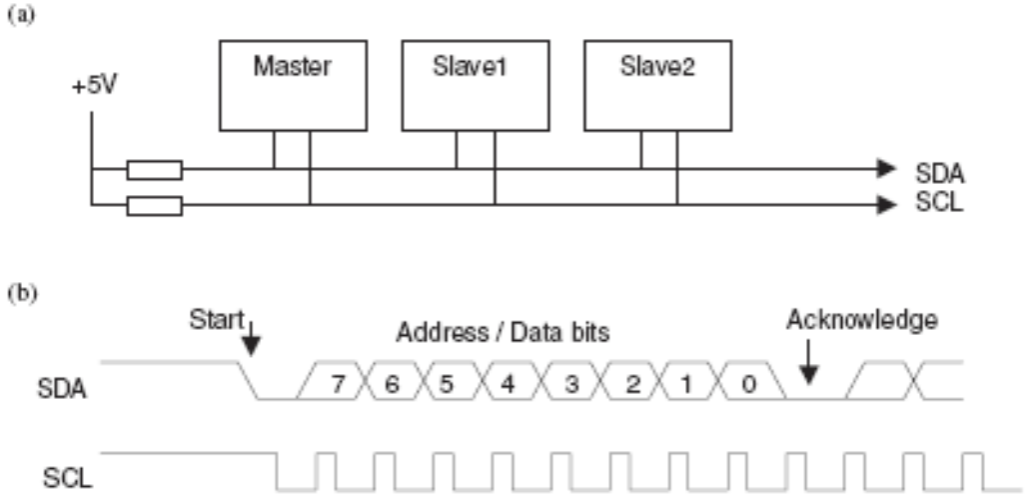


Figura 28: Esquema I^2C [34]

O tipo de topologia é em barramento. Compete ao *Master:*

- Gerar o sinal de relógio;
- Controlar a linha de dados, *SDA*, para enviar ordens e dados para os outros dispositivos *Slaves*
- Enviar ordens aos *Slaves* para que respondam com determinados dados. Neste último caso, depois de fazer o pedido, o *Master* mantém o sinal de relógio ativo o tempo necessário para que o *Slave* lhe possa responder. Durante a resposta é o "*Slave*" que controla a linha de dados SDA.
- O *Slave* pode forçar o sinal de relógio a zero fazendo dessa forma que o *MASTER* suspenda o envio de dados "*Controlo de fluxo*".
- Cada *Slave* tem um endereço de 7 bits.

A tensão aplicada à linha de dados *SDA,* só pode ser alterada, se, e quando o sinal *SCL* tiver a tensão nula, exceto no início e o fim da mensagem. Se a tensão à linha de dados passar a zero, enquanto o sinal de relógio tiver uma tensão positiva, identifica o início da mensagem *Start.* A mudança de tensão na linha de dados *SDA* para 5V, enquanto o sinal de relógio tiver uma tensão positiva, identifica que a mensagem terminou *Stop.*

Depois de todas as informações úteis ao projeto tiverem sido capturadas são enviadas pelo módulo SIM900 para um servidor externo. A ligação é feita por *TCP/IP*. Nesta ligação são enviados comandos HTML para gravar as medições numa base de dados *MySQL*.

O protocolo *TCP – Transmission Control Protocol* foi desenvolvido pela *DARPA – Defense Advanced Research Projects Agency.* O *TCP/IP* representa um conjunto de protocolos que permitem que diversos equipamentos que constituem uma rede possam comunicar entre si. É um protocolo estruturado por camadas na qual cada camada utiliza e presta serviços às camadas adjacentes. Cada camada apenas trata das informações que correspondem à sua função.

Existem cinco camadas distintas que formam o *TCP/IP* [35]:

- Aplicação. Esta camada é formada por um conjunto de protocolos que permitem o correto funcionamento dos diversos Serviços/Aplicações do modelo *TCP/IP*. Esta camada não possui um padrão comum para todas as aplicações, ou seja, consoante o serviço em questão irá depender também o protocolo que o vai atender;
- Transporte. O *TCP* é responsável pelos serviços de transporte que garantem a entrega sequencial de dados sem erros e sem falhas. O acesso das aplicações a esta camada é feito através de portas;
- Camada *IP* ou Internet. Responsável pelo endereçamento, roteamento e controlo de envio e receção dos dados;
- Acesso à Rede. Esta camada tem como principal função a adaptação do Modelo *TCP/IP* aos diversos tipos de redes (*X.25, ATM, FDDI, Ethernet, Token Ring, Frame Relay, PPP e SLIP*). É a camada de abstração de hardware e devido à enorme variedade de tecnologias de rede possíveis, é uma camada não normalizada pelo modelo *TCP/IP*. É possível a interligação e interoperação com redes heterogéneas. Nesta camada são utilizados *gateways* ou *routers*.
- Física. Esta camada descreve as características físicas da comunicação tais como a natureza do meio usado para a comunicação (cobre, fibra-óptica ou links de rádio) e todos os detalhes relacionados com os sinais (modulações, comprimentos de onda, níveis de sinal, sincronizações, distâncias máximas, etc).

4.2 Programa *Arduino*

O código para o *Arduino* foi escrito com o software próprio denominado *Arduino IDE*.

A área de código é dividida em duas áreas importantes. O *Setup* e o *Loop*. No Setup é colocado todo o código que só vai correr uma vez. Tipicamente é a preparação de todos os sensores e dispositivos presentes no projeto. No *Loop* o código é avaliado recursivamente onde os sensores captam e enviam a informação necessária. Pode-se ainda acrescentar uma terceira zona situada antes do *Setup*. Onde são declaradas todas as variáveis globais e onde se importa todas a bibliotecas necessárias para o projeto.

Na área antes do Setup do programa *get_full_program_AS_V12_mega* escrito para este projeto são importadas as bibliotecas *LIDARLite.h e AdafruitGPS.h* que servem para controlar o *LiDAR* e o *Módulo GPS* respetivamente. Foi também gerado um ficheiro de código à parte denominado *functions_sim900.ino* onde foram escritas todas as funções necessárias para usar o módulo *SIM900*. A figura seguinte demonstra a importação de todas as bibliotecas necessárias para o projeto e a declaração de todas as variáveis globais, algo que se pode ver na figura seguinte.

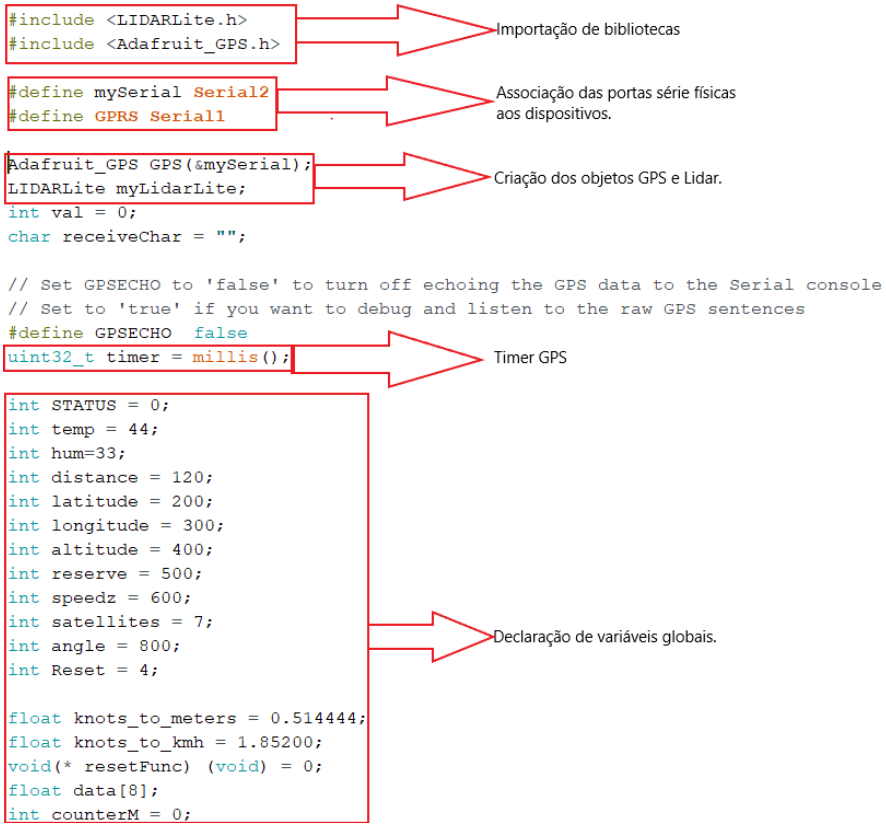


Figura 29: Excerto de código. Pré Setup

O objeto *GPS* é criado e associado à segunda porta série do *Arduino Mega* (*Serial2)* na linha:

- *Adafruit_GPS GPS (&mySerial);*

O Timer GPS também tem especial importância que será explicado mais à frente.

Na criação do objeto referente ao Lidar:

- *LIDARLite myLidarLite;*

Não é necessário associar uma porta série porque este comunica por I^2C.

De entre as variáveis globais, a linha, *void(* resetFunc) (void) = 0,* é especialmente útil. Ao chamar a função *resetFunc,* permite que o Arduino entre em Reset. Evita-se assim

que o programa pare quando ocorre algum erro de comunicação (comunicação série entrar em *Timeout* por exemplo).

De seguida entra-se no Setup. A Figura 30 demonstra a implementação das funções que compõem o *Setup.*

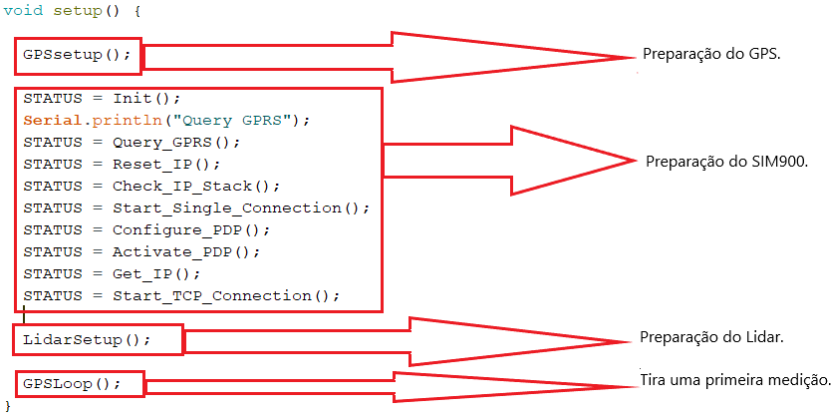


Figura 30: Excerto do código. Setup

Na função *GPSsetup* a porta série do *GPS* é inicializada e são dadas instruções relativas às frases *NMEA* a serem utilizadas bem como a cadência das mesmas. As informações chegam ao módulo por frases *NMEA* (National Marine Electronics Association). Há várias formas de obter informação a partir dos satélites, no Setup do programa o GPS foi configurado para trabalhar com atualizações de 1Hz e para ficar à escuta das frases RMC (*Recomended Minimum)* e *GGA* como ilustra a imagem seguinte.

```
// uncomment this line to turn on RMC (recommended minimum) and GGA (fix data) including altitude
GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);
// uncomment this line to turn on only the "minimum recommended" data
//GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCONLY);
// For parsing data, we don't suggest using anything but either RMC only or RMC+GGA since
// the parser doesn't care about other sentences at this time

// Set the update rate
GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ);   // 1 Hz update rate
// For the parsing code to work nicely and have time to sort thru the data, and
// print it out we don't suggest using anything higher than 1 Hz
```

Figura 31: Excerto de código do Setup do GPS.

Na função *LidarSetup* o *LiDAR* foi configurado para trabalhar com o protocolo I^2C a 400kHz e no modo *default*. A *Figura 32* representa a função de *setup* do *LiDAR* onde este é configurado.

```
myLidarLite.begin(0, true); // Set configuration to default and I2C to 400 kHz

/*
  configure(int configuration, char lidarliteAddress)

  Selects one of several preset configurations.

  Parameters
  -------------------------------------------------------------------------
  configuration:  Default 0.
    0: Default mode, balanced performance.
    1: Short range, high speed. Uses 0x1d maximum acquisition count.
    2: Default range, higher speed short range. Turns on quick termination
        detection for faster measurements at short range (with decreased
        accuracy)
    3: Maximum range. Uses 0xff maximum acquisition count.
    4: High sensitivity detection. Overrides default valid measurement detection
        algorithm, and uses a threshold value for high sensitivity and noise.
    5: Low sensitivity detection. Overrides default valid measurement detection
        algorithm, and uses a threshold value for low sensitivity and noise.
  lidarliteAddress: Default 0x62. Fill in new address here if changed. See
    operating manual for instructions.
*/
myLidarLite.configure(0); // Change this number to try out alternate configurations
```

Figura 32: Excerto de código. Setup Lidar.

No *setup* do módulo *SIM900* é necessário enviar vários comandos *AT* por meio de porta série. Como é um procedimento bastante extenso foi escrito num documento à parte denominado *functions_sim900.ino.* A importação deste ficheiro é feita de forma automática pelo *IDE*.

Na função *Init()* deste documento é enviado o comando *AT*. Depois do envio, o programa fica à espera da resposta “OK” vinda do *SIM900* para puder prosseguir para a próxima função. Essa espera é feita pela função *waitforFullString(int timeout)*. Esta função fica à escuta (função *Listen())* até surgir o caractere de nova linha, vulgo “\n”.

```
incoming_char =GPRS.read();
Incoming_String +=incoming_char;
```

Figura 33: Incremento da String. Função Listen().

A função *Listen()* recebe, caractere a caractere, as mensagens vindas do *SIM900* e incrementa numa *String* denominada *Incoming_String,* como se pode ver na Figura 34*.* A condição de paragem é a “chegada” do caracter “\n” ou quando o tempo de *timeout* é ultrapassado (Figura 36). Quando isto acontece o programa faz *Reset* ao *Arduino* começando tudo de novo. Para enviar o comando *“AT”* e receber a resposta adequada não é necessário ter tantos cuidados, no entanto este tipo de controlo é bastante útil para os comandos que são posteriormente enviados. Quer na parte do *Setup,* quer na parte do *Loop.*

```
if(incoming_char == '\n')
{
  //Serial.println("incoming char = //n");
  break;
}
if(counter > timeout)
{

  issueTimoutWarning();
  break;
}
counter++;
```

Figura 34: Condição de paragem. Função waitforFullString.

A função Query_GPRS envia o *comando* "AT+CGATT?" este comando informa o SIM900 para se ligar ao serviço de GPRS permitindo ao sistema ter acesso à rede móvel. Depois de enviar o comando o Arduino entra na função isEcho(String order). Esta função

avalia se a String rececionada é igual à ordem enviada, vulgo echo. Este módo é especialmente útil para quando se está a fazer debug do sistema, no entanto o sistema em trabalho não recebe estes echos. O código foi escrito de modo a trabalhar com a receção de echos e sem a receção de echos. Ao ser determinado que foi rececionado um echo, a função entra noutra denominada waitforResponse(int timeout, String order). Esta função fica à espera da resposta que seja adequada à ordem enviada. Neste caso aguarda por "+CGATT: 1\r\n". Como se pode ver na Figura 35.

```
void waitforResponse(int timeout, String order)
{
    Incoming_String = "";
    String response = "";
    Serial.print("order = ");
    Serial.println(order);
    if(order=="AT")
    {
      response = "OK\r\n";
    }
    else if((order == "AT+CGATT?\r\n") || (order == "AT+CGATT?")|| (order == "AT+CGATT?\r"))
    {

      response = "+CGATT: 1\r\n";
    }
    else if((order == "AT+CIPSHUT\r\n") || (order == "AT+CIPSHUT")|| (order == "AT+CIPSHUT\r"))
    {

      response = "SHUT OK\r\n";
    }
    else if((order == "AT+CIPSTATUS\r\n") || (order == "AT+CIPSTATUS")|| (order == "AT+CIPSTATUS\r"))
    {

      response = "STATE: IP INITIAL\r\n";
    }
    else if(order.startsWith("AT+CIPSTART"))
    {

      response = "CONNECT OK\r\n";
    }
```

Figura 35: Excerto de código. waitforResponse(int timeout, string order).

Se a resposta não é rececionada antes do *timeout,* o *Arduino* entra em *Reset* e começa tudo de novo.

As funções seguintes comportam-se de maneira semelhante, portanto serão descritas por meio de uma tabela.

Tabela 7: Setup SIM900.

Função	Comando	Comentário	Resposta
Reset_IP	*"AT+CIPSHUT"*	Desactiva o *GPRS*	*"SHUT OK\r\n"*
Check_IP_Stack	*"AT+CIPSTATUS"*	Avalia o estado da conecção	*"STATE: IP INITIAL\r\n"*
Start_Single_Connection	*"AT+CIPMUX=0"*	Activa uma conecção multi-IP	*"OK\r\n"*
Configure_PDP	*"AT+CSTT ="APN", "USER_NAME", "PASSWORD""*	Activa o serviço móvel *GPRS*	*"OK\r\n"*

Activate_PDP	*"AT+CIICR"*	Activa ligação *Wireless* com o *GPRS*	*“OK\r\n”*
Get_IP	*"AT+CIFSR"*	Aguarda pelo *IP*	Aguarda pelo IP
Start_TCP_Connection	*"AT+CIPSTART=”TCP”, “HOST”, “PORT””*	Inicia uma coneção *TCP* com o servidor com o endereço *HOST* e a porta *PORT*.	*"CONNECT OK\r\n"*

Para a função Configure_PDP:

- *String APN = "internet";*
- *String USER_NAME = "guest";*
- *String PASSWORD = "guest".*

Para a função Start_TCP_Connection:

- *String HOST = "xxx.xxx.xxx.x"* (endereço de IP do servidor, censurado por questões de privacidade)*;*
- *int PORT = 8080*.

Depois do *Setup* estar concluído o programa entra na parte recursiva que é o *Loop*. A Figura 36 mostra como são chamadas as funções nesta parte do código.

```
void loop() {
  // put your main code here, to run repeatedly:
  if(counterM == 200)
  {
    GPSLoop();
    counterM = 0;
  }
  else
  {
    counterM++;
  }
```

Figura 36: Loop do Arduino.

No Loop apenas é chamada a função GPSLoop. Nesta função são retiras todas as informações úteis do GPS e a distância calculada no LiDAR. Esta função é chamada uma vez em cada duzentas iterações para evitar problemas de comunicação.

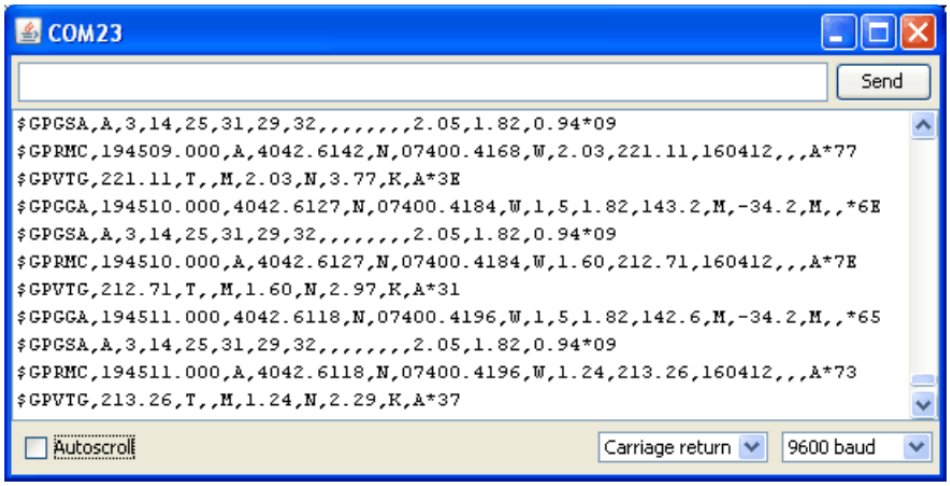


Figura 37: Output do GPS.

A figura 35 ilustra o tipo de dados que é dado pelo *Módulo GPS*. Ao analisar a linha *"$GPRMC,194509.000,A,4042.6142,N,07400.4168,W,2.03,221.11,160412,,,A*77"* pode-se tirar as informações úteis para esta etapa do projeto. A primeira parte da frase *"194509.000"* indica a hora a que a mensagem chegou ao módulo *GPS* no fuso horário *GMT* (*"Greenwich Mean Time"*). Os números "*19"* indicam a hora os dois números seguintes indicam os minutos, "*45"*, e os dois números seguintes indicam os segundos, "*09"*. Os três restantes indicam os milissegundos. O *GPS* desconhece o fuso horário que está. Se estiver noutro fuso horário é necessário proceder a cálculos para determinar a hora correta. A segunda parte indica o estado, se for um V significa que temos um tipo dados *"Void"*, não tem informação de posição. Se for um A significa que está *"Active",* o *GPS* tem um *fix* na posição e é possível retirar esses dados com exatidão. Os quatro segmentos seguintes indicam a posição. É importante notar que o *GPS* não apresenta os dados de geolocalização em graus decimais. Estes dados são apresentados no seguinte formato para a latitude *DDMM.MMMM*. Os primeiros dois caracteres são os graus os dois seguintes são os minutos e os quatro últimos carateres são parte decimal dos minutos. Desta forma, para a latitude temos a seguinte informação, *"4042.6142,N"*. O que significa *"40 graus e 42,6142 minutos Norte"*. Ou, *"40º e 42,6142' Norte"*. Para a latitude, os dados estão no formato *DDDMM.MMMM*. Os três primeiros caracteres são os graus, os três seguintes os minutos e os quatro últimos a parte decimal dos minutos. Desta forma, para a latitude têm-se *"07400.4168,W"*, ou seja, *"074 graus e 0,4168 minutos Oeste"*. Ou *"74º e 0,4168' Oeste"*.

Os dados seguintes dão a informação da velocidade em nós. *2,03 nós*. Depois têm-se a informação da direção de viagem. Este cálculo é feito tendo em conta a posição anterior. De seguinte têm-se a informação da data, *"160412"*, o que significa 16 de Abril de 2012. A última parte, *"*XX"*, é o *"checksum"* da mensagem.

A partir deste ponto seria necessário fazer o *"parsing"* da mensagem que é chegada. No entanto a biblioteca disponibilizada pelo fabricante já está preparada para o efeito. Como se pode ver Figura 37 o objeto *GPS* já possui as informações necessárias. A Tabela 8 resume a forma de como as informações são tratadas pelo Arduino:

Tabela 8: Descrição dos objetos disponíveis na biblioteca.

Variável	Comentário
GPS.fix	Indicação de fix
GPS.hour	Indicação da hora.
GPS.minute	Indicação do minuto
GPS.seconds	Indicação do segundo
GPS.milliseconds	Indicação do milissegundo

GPS.latitude	Indicação da latitude
GPS.lat	Indicação de Norte ou Sul
GPS.longitude	Indicação da longitude
GPS.lon	Indicação de Este ou Oeste
GPS.speed	Indicação da velocidade
GPS.angle	Indicação do ângulo de viagem
GPS.altitude	Indicação da altitude
GPS.satellites	Indicação do número de satélites

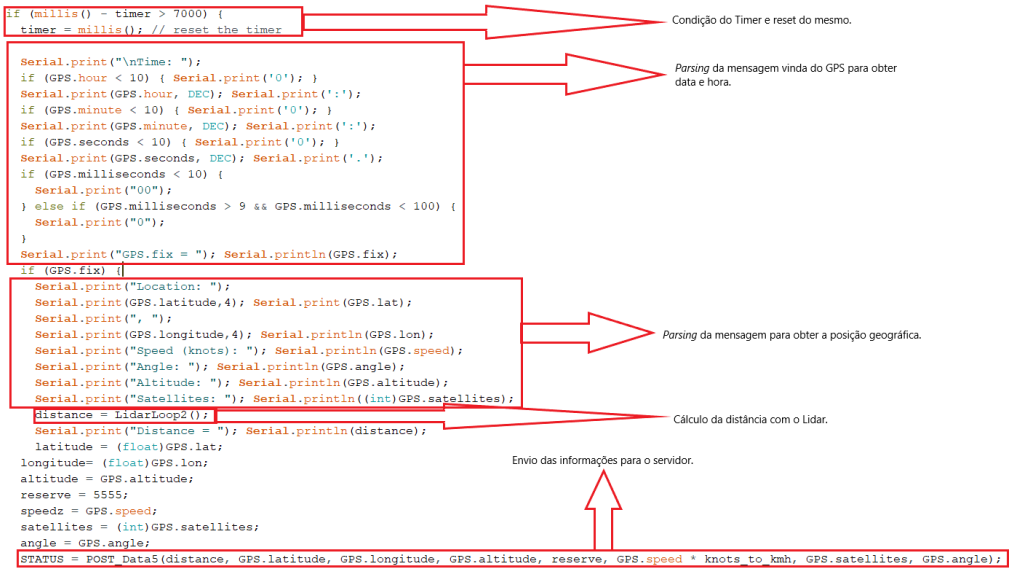


Figura 38: Excerto do código Arduino. GPSLoop.

A distância é medida com o *LiDAR*, que é um sensor do tipo *TOF*. Na função *Lidarloop2* é tirada uma média de nove medições e é retornada a distância em formato *float.* O valor da distância é passado ao *Arduino Mega* por I^2C. A Figura 39 representa um excerto do código que foi escrito para determinar a distância.

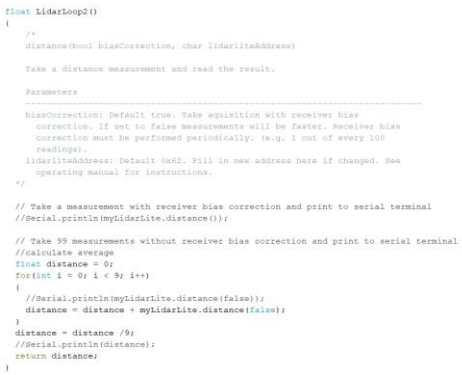


Figura 39: Excerto do código Arduino. LidarLoop2.

Como o esquema da Figura 27 indica, as informações vindas do *Módulo GPS* chegam ao *Arduino* por comunicação série. Os dados são enviados na função *POST_Data5(float distance, float latitude, float longitude, float altitude, float reserve, float speedz, int satellites, float angle).* É enviado o comando *"AT+CIPSEND"*, depois de receber o caracter *">"* do *SIM900* é enviado o comando *HTML "POST /add_V3.php HTTP/1.1\r\n"* seguido das informações previamente recolhidas da seguinte forma;

“POST_DATA = POST /add_V3.php HTTP/1.1
Host: xxx.xxx.xxx.x
Content-Type: application/x-www-form-urlencoded
Content-Length: 97
dist1=Distance&lat1=Latitude&long1=Longitude&alt1=Altitude&res1=Reserve&speed=Speed&sat1=Satellites&ang1=Angle”.

De seguida a função espera pela mensagem "SEND OK\r\n" para poder prosseguir. Se o *SIM900* enviar a resposta "CLOSED\r\n" significa que o servidor fechou a ligação, depois disto, o *Arduino* corre a função *Start_TCP_Connection* com o intuito de voltar a reconectar-se com o servidor. Como isto demora algum tempo o *Arduino* retorna da função e volta ao início do *loop* para tirar nova medida.

Se o envio for bem-sucedido (receção da mensagem "SEND OK\r\n") a função espera pela chegada da mensagem *"Content-Length"*. Isto por norma indica que a medição foi guardada na base de dados e que o programa pode prosseguir. A função é retornada e o programa volta ao início do *loop* para tirar nova medição.

4.3 Programa do Servidor

Para programar o servidor optou-se por escrever o programa em linguagem *PHP* e *HTML*. Para isso foi usado o editor *Sublime* disponível online de forma gratuita.

Para abrir o servidor para a rede externa foi necessário colocar uma regra no router que permitisse fazer o roteamento de ligações externas para o servidor local. O servidor foi aberto para qualquer endereço de IP desde que a comunicação fosse feita na porta 8080. Após a receção do pedido de ligação vindo do exterior, este é reencaminhada para o servidor local com o endereço 192.168.1.67 como é mostrado na Figura 40.

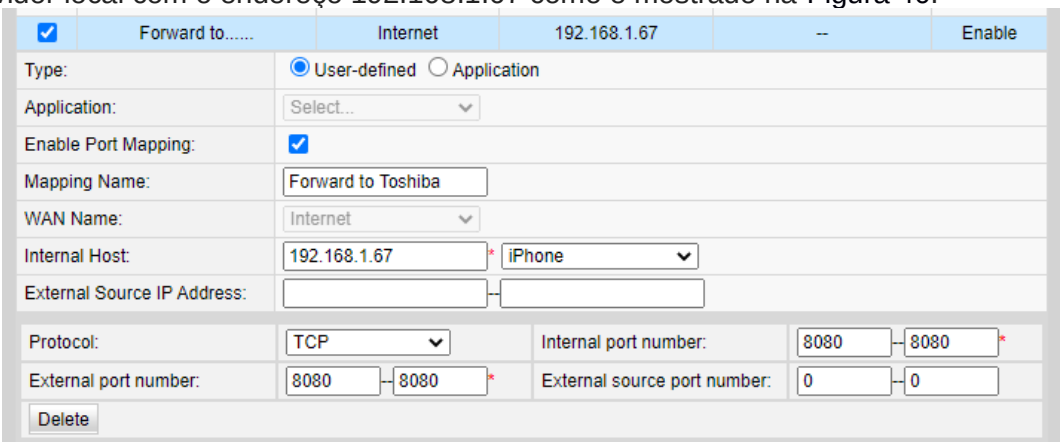


Figura 40: Regra de roteamento.

No lado do servidor é mostrada a seguinte página web. Para aceder basta digitar o endereço de IP seguido da porta em qualquer *browser.*

VIV Test

Timestamp	Distance	Latitude	Longitude	Altitude	Speed	Angle	Satellites	Reserve
2020-12-18 11:40:38.521	50.220	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:40:34.219	50.220	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:40:29.568	50.670	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:40:25.245	50.110	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:40:21.018	50.890	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:40:16.884	50.780	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:40:12.780	51.890	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:40:08.640	51.220	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:40:04.402	49.780	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:40:00.157	49.890	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:56.039	50.110	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:51.701	49.000	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:47.603	51.560	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:43.324	50.440	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:39.218	50.440	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:35.030	50.780	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:30.846	49.440	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:26.777	50.670	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:22.706	51.440	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:18.509	50.780	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:14.361	50.780	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:39:10.195	49.890	4042.7261	833.2983	15.000	52.040	322.370	7	5555
2020-12-18 11:38:13.070	49.440	4039.1609	837.2348	47.200	40.690	35.610	9	5555
2020-12-18 11:38:08.940	51.780	4039.1609	837.2348	47.200	40.690	35.610	9	5555
2020-12-18 11:38:04.690	51.780	4039.1609	837.2348	47.200	40.690	35.610	9	5555
2020-12-18 11:38:00.425	51.670	4039.1609	837.2348	47.200	40.690	35.610	9	5555

Figura 41: Página web do servidor.

A tabela que é demonstrada na Figura 41 foi criada com recurso a uma base de dados *MySQL*. A base de dados denomina-se *"viv_v1.0.1"*, nessa base de dados foi criada uma tabela com as colunas *Timestamp*, Distância (que vem diretamente do Lidar em centímetros), Latitude, Longitude, Altitude, Velocidade (vem de GPS em nós mas depois é convertida para km/h, sendo gravada em km/h), Ângulo de Viagem, Número de Satélites e uma coluna de reserva que acabou por não ser utilizada. A conexão com a base de dados é feita na função *Connection* dentro do ficheiro *connect.php.* Depois o objeto *$mysqli* é retornado sendo este o objeto da conexão. A Figura 42 representa o excerto de código da função *Connection.*

```php
<?php
    function Connection()
    {
        $server="localhost";
        $user="root";
        $pass="root";
        //$db="arduino_db";
        //$db="viv_v1.0.0";
        $db="viv_v1.0.1";

        $mysqli = new mysqli($server, $user, $pass, $db);
        if (!$mysqli)
        {
            //"!!Database connection failed: "
            echo "Database connection failed:";
            die("!!Database connection failed: " . mysqli_error($mysqli));
            echo "</br>mysqli_error($mysqli)";
        }
        $db_select = mysqli_select_db($mysqli, $db);
        if (!$db_select)
        {
            echo "</br>!!!Database selection failed: ";
            die("</br>!!!Database selection failed: " . mysqli_error());
        }
        return $mysqli;
    }
?>
```

Figura 42: Excerto de código do Servidor. connect.php

No ficheiro principal do servidor *index.html* foi criada uma função *loadXMLDoc* que fica à escuta de comandos HTML. Dentro desta função é criado o objeto *XMLHttpRequest* que fica encarregue de trocar dados com o servidor em *background* tornando possível a atualização de excertos da página web sem a necessidade de recarregar manualmente a página. Dentro desta função é aberto o ficheiro *getdatabase_V3.php* que amostra os dados. Os dados são recuperados da base de dados através dum *Query* do tipo *Select* depois a tabela com os dados da Figura 41 é criada e amostrada. O ficheiro *add_V3.php* é chamado pelo Arduino quando este envia dados para o servidor. Os dados são passados por variável global do tipo *post* e são gravados na base de dados por intermédio dum *Query* do tipo *Insert Into* como se pode ver na Figura 43.

```php
<?php

    include("connect.php");
    $link=Connection();
    //echo "first </br>";
    $lat1 = $_POST['lat1'];
    $long1 = $_POST['long1'];
    $altitude = $_POST['alt1'];
    $distance = $_POST['dist1'];
    $reserve = $_POST['res1'];
    $speed = $_POST['speed'];
    $satellites = $_POST['sat1'];
    $angle = $_POST['ang1'];




    $sql = "INSERT INTO `coords_Table_v3` (`TimeStamp`, `Location`, `Altitude`, `Distance`, `Speed`, `Angle`, `Satellites`,`Reserve`)
        Values(NOW(3), Point('".$lat1."', '".$long1."'), '".$altitude."', '".$distance."', '".$speed."', '".$angle."', '".$satellites."
        ', '".$reserve."')";


    mysqli_query($link,$sql);
```

Figura 43: Excerto do código do Servidor. add_V3.php

4.3 Montagem do Sistema

Após a escolha do hardware que aparentasse ser mais adequado e a programação do servidor e do *Arduino* foi desenhada uma caixa em *Solidworks* para albergar todos os dispositivos. Originalmente a caixa teria o aspeto que se pode ver na Figura 44:

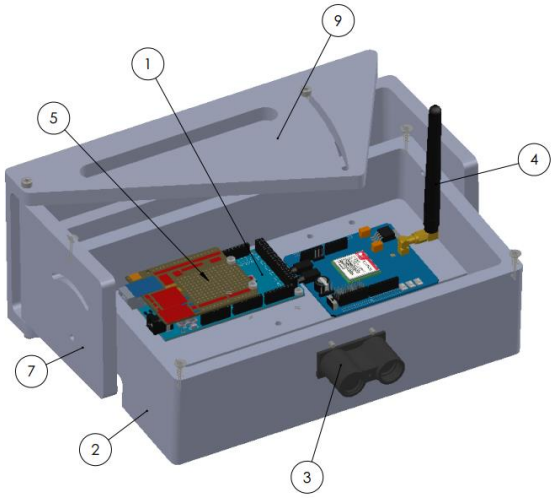


Figura 44: Assembly V6

Tabela 9: Material Utilizado.

ITEM NO.	PART NUMBER	QTY.
1	Arduino Mega ATmega 2560	1
2	box_V2	1
3	Garmin Lidar Lite V3	1
4	GSM Arduino Shield SIM900	1
5	Adafruit GPS Logger Shield	1
6	top	1
7	lever_mega	1
9	lever2_mega	1

A modelação dos dispositivos *Arduino Mega*, *Modem SIM900*, *LiDAR Lite V3* e *Adafruit GPS Shield* já estava feita. Foi apenas necessário fazer download dos mesmos numa biblioteca disponível online denominada *GrabCad*.

No entanto, devido a alguns erros no desenho do conjunto foi necessário fazer algumas alterações para que o material estivesse todo acomodado. Com a disposição mostrada na imagem anterior, a totalidade do material não iria caber devido a quatro fatores:

- Não foi deixado espaço suficiente para a ficha *USB* encaixar no *Arduino*;
- O encaixe da antena do *SIM900* é ligeiramente maior que aquele que está modelado no desenho;
- Não foi deixado espaço suficiente para a *Powerbank*;
- Não foi deixado espaço para o encaixe da ficha do *LiDAR Lite V3*.

Assim, alterou-se a disposição do material, encostou-se o *Arduino* a uma das paredes da caixa e prendeu-se o *Arduino* e o *Adafruit GPS Shield* com um parafuso. Deixou-se algum espaço para a ficha do *LiDAR* e o *SIM900* e a *Powerbank* foram presos com braçadeiras de fivela.

A *Figura 45* representa o aspeto atual da caixa:

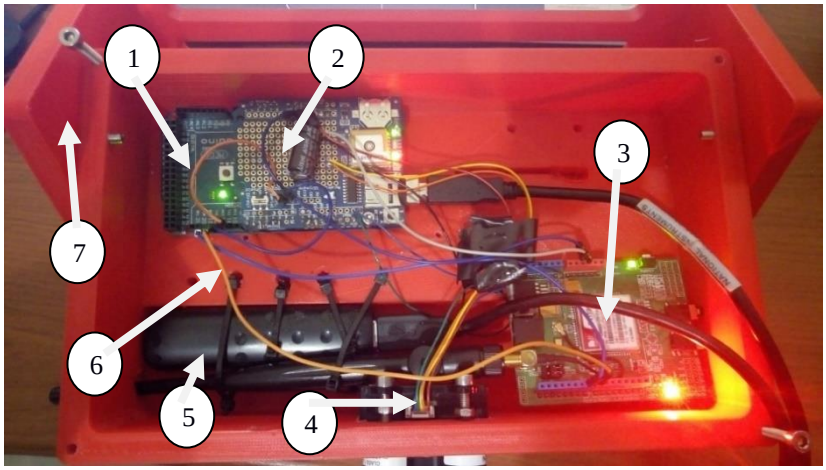


Figura 45: Caixa. Aspeto real.

Tabela 10: Material utilizado.

1	Arduino Mega
2	Adafruit GPS Shield
3	Modem SIM900
4	Lidar LITE V3
5	Powerbank
6	*box_V2*
7	*lever_mega*

A caixa é presa à grelha do automóvel também com braçadeiras de fivela. Os rasgos angulares são necessários para que o *LiDAR* possa estar bem orientado com o pára-choques dos carros em frente.

Figura 46: Montagem da caixa no carro.

4.4 Esquema elétrico

A Figura 47 seguinte representa o esquema elétrico do protótipo elaborado num *Software* próprio para o efeito denominado *Fritzing.*

Para a comunicação em série foram utilizadas as portas físicas disponíveis no *Arduino Mega.* O *Módulo SIM900* foi ligado ao *Arduino Mega* pela porta *Serial1*, pino 18 (*Tx1*) liga ao pino 7(Rx) do *Módulo SIM900,* a ligação é feita por *SoftwareSerial* do lado *SIM900.* O pino 19 (*Rx1*) do *Arduino Mega* liga ao pino 8(*Tx*) do *Módulo SIM900.* A alimentação é ligada de forma direta, os *5V* do *Arduino* ligam aos *5V* do *SIM900*, as massas também são ligadas desta forma.

O *Arduino* comunica com o *Módulo GPS* através da porta série física *Serial2,* desta forma, o pino 16 (*TX2*) do *Arduino* liga ao pino *RX* do *GPS* e o pino 17 (*Rx2*) liga ao pino *Tx* do *GPS.* A alimentação é ligada por pinos físicos que encaixam o *GPS* ao *Arduino*. A ligação entre o *Arduino* e o *GPS* é feita por *SoftwareSerial.*

O *LiDAR* comunica com o *Arduino* por I^2C, desta forma, os pinos *SDA* e *SCL* são ligados diretamente. A alimentação tem de ser ligada a um condensador, posteriormente são ligados os *5V* e a massa. Os pinos *Mode* e *Power Pin* não são utilizados, de modo que são fixos ao *GPS Shield* com solda, mas não estão ligados a nada.

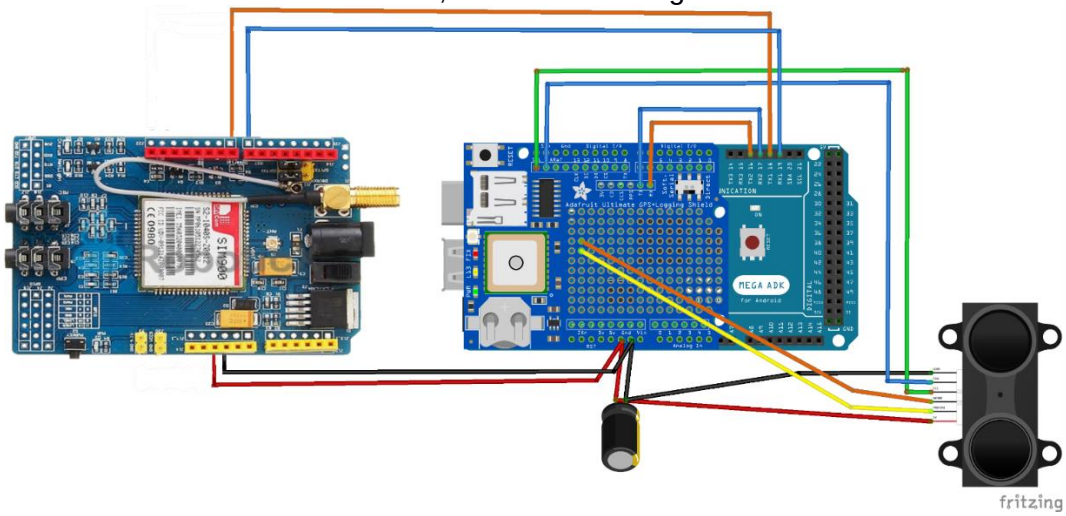


Figura 47: Esquema elétrico.

4.5 Descrição dos testes práticos.

Os testes para este trabalho foram realizados por etapas.

Primeiro decidiu-se testar os sensores em separado. Ligou-se o *LiDAR* ao *Arduino* (na altura seria um *Arduino UNO)* escreveu-se o código *Arduino* necessário para pôr o dispositivo a funcionar. Este código era em todo semelhante ao que foi descrito nas secções anteriores, ao ligar o *Serial Monitor* disponível no *Arduino IDE* viu-se que o *LiDAR* estava a transmitir a distância ao *Arduino* sem nenhum problema. Os testes ao Módulo *GPS* foram um bocado mais complicados. Primeiro testou-se a ligação por *Hardware Serial* direta desde módulo ao *Arduino.* As mensagens que chegavam ao *Arduino* eram semelhantes às descritas na secção 4.2 desta dissertação. A partir daqui concluiu-se que ambos os sensores funcionavam da forma que era pretendida. No entanto não é possível ligar o *GPS* à porta *Serial0* do *Arduino UNO,* optou-se por ligar o *GPS* ao *Arduino* por *Software Serial.* O pino 7 do *Arduino* comportar-se-ia como pino *Tx* e o pino 8 como pino *Rx.* O módulo *GPS* tem um seletor que permite a configuração da comunicação para *Software Serial* bastando apenas selecionar o lado correto. Estando a ligações prontas, correu-se um programa de teste disponibilizado na biblioteca do fabricante que já estava preparado para comunicar por *Software Serial* e também estaria preparado para fazer o *parsing* da mensagem como está descrito na secção 4.2. Este

módulo está preparado para gravar informações num cartão de memória. A comunicação seria feita por *SPI.* No entanto foi uma etapa que não se conseguiu concluir. Por algum motivo o *Arduino* não estava a reconhecer o cartão. Tentou-se usar vários cartões e formatar os mesmo de várias maneiras diferentes, mas sem sucesso. A partir daqui foi necessário avançar para a etapa seguinte.

Depois dos sensores estarem devidamente configurados e ambos os programas escritos, seria a altura de configurar e programar o módulo *SIM900* para fazer as comunicações que são o objetivo central deste trabalho. Primeiro ligou-se ao *SIM900* um adaptador USB para TTL para fazer ligação direta com o computador e enviou-se comandos simples ao *SIM900* (como por exemplo envio do comando *AT,* ligação *TCP* ao servidor da Universidade de Aveiro, entre outros*).* Viu-se que este dispositivo respondia sem problemas. Depois destes pequenos testes optou-se por escrever um pequeno código que enviaria uma *SMS* para um telemóvel, esta tarefa ficou concluída com sucesso.

Depois destes pequenos testes foi necessário escrever o código para que o *SIM900* se ligue ao servidor pessoal, e que envie os dados para este mesmo servidor. Primeiro, o router teria de ser configurado para receber comunicações vindas do exterior (secção 4.3), depois seria necessário escrever o código do servidor que rececionasse os dados e os escrevesse numa base de dados e dados *MySQL* (secção 4.3) e um código *Arduino para* enviar esses dados. Tanto o código do *Arduino* como o do servidor foram baseados em [36].

Este documento explica como como enviar os dados vindos de um sensor de temperatura e um sensor de humidade para um servidor que depois escreve numa base de dados. Ao experimentar este código viu-se rapidamente que funcionava como era previsto. No entanto, após enviar a primeira mensagem, demorava perto de 30 segundos a enviar a segunda. O problema residia no facto de o programa esperar pelo fecho da ligação *TCP* vinda do lado de servidor para depois voltar-se a conectar. Este processo demora tempo. O problema só foi solucionado mais tarde.

Após a escrita do código *Arduino* e do código do Servidor os primeiros testes que foram feitos ocorreram em casa, num ambiente controlado. Aí, o *Arduino* estaria permanentemente ligado ao computador por porta Série. O centro de processamento para este projeto seria um *Arduino Uno*. No entanto este não era capaz de albergar todo o código necessário por falta de espaço para as variáveis globais. Optou-se por trocar este dispositivo por um controlador *ESP8266MOD Ver 0.1*. No entanto, por algum motivo, o código não funcionava como era suposto. Optou-se por trocar este dispositivo por um *Arduino Mega 2560*. No entanto o código também não funcionava como era suposto. Teve-se de adaptar o código do *Arduino*. Depois de algumas afinações, o tempo de transmissão situava-se nos 5 segundos e não se registava problemas na comunicação.

Depois destes problemas estarem solucionados e da montagem do protótipo tal como está representado na Figura 46 foram feitos alguns testes para averiguar a robustez do protótipo. Ao montar o protótipo como se vê na Figura 46 reparou-se de imediato que haveria problemas com vibrações uma vez que a caixa não estava bem segura. A maneira de como a caixa é fixa à grelha não é a melhor. São necessários pelo menos mais dois pontos de fixação. Para solucionar este problema atou-se duas tiras de fita isolante à grelha do automóvel (como se pode ver na mesma figura). Tirou-se de imediato a conclusão que seria impossível fazer testes em autoestrada.

Ao fazer-se alguns testes numa estrada em meio urbano numa via principal da cidade de S. João da Madeira até uma zona da cidade vizinha de Santa Maria da Feira (Figura 48) reparou-se que o sistema deixava de transmitir dados passado algum tempo. O problema é que a comunicação Série do *SIM900* entrava em *Timeout* ficando o código "preso" num *loop* infinito. A solução encontrada foi fazer-se *Reset* ao código do *Arduino* por intermédio da função *void(* resetFunc) (void)*. Quando o código entra na rotina de *Timeout,* em vez de o código entrar num *loop* infinito, esta função é chamada e o *Arduino* recomeça do zero. Esta solução resolveu o problema de o código ficar preso, no entanto

é algo que eleva o tempo de transmissão. Quando isto acontece, o tempo de transmissão aumenta para 10 segundos.

Figura 48: Rota 1

A distância percorrida nestes testes foi de 11km com uma velocidade média de 52.3 km/h.

Depois deste problema ficar solucionado, foram feitos mais testes na zona da Universidade de Aveiro. Devido ao problema das vibrações exposto anteriormente optou-se então por se fazer testes no parque de estacionamento da Universidade de Aveiro. O objetivo seria medir a variação da distância e tirar as coordenadas geográficas do automóvel em andamento. No entanto o tempo de transmissão é demasiado grande, cerca de 20 segundos. Optou-se por configurar o código para tirar apenas a distância, assim o tempo de transmissão desce para sensivelmente 5 segundos. Viu-se que o *LiDAR* consegue medir a distância de um modo bastante satisfatório.

Capítulo 5: Conclusões e Sugestões de trabalhos futuros.

O primeiro objetivo desta tese é a comunicação de informações vindas do *LiDAR* e do *GPS.* As informações são enviadas por comandos *HTML* através duma ligação *TCP* como foi descrito no capítulo 4. Este objetivo foi concluído embora com problemas relativos ao tempo de comunicação. As informações seriam enviadas ao servidor por via de um módulo *GSM,* o *SIM900.* Depois da configuração de sensores e programação do microcontrolador e do servidor, o protótipo foi preso à grelha dum automóvel. De seguida o automóvel seguia a sua viagem pela rota escolhida, descrita na secção 4.5. Esta rota foi escolhida por ser de fácil acesso, por ter algum tráfego automóvel, ficando assim mais fácil a medição de distâncias, e por ter boa cobertura de sinal *GPRS* e de sinal *GPS.* Desta forma seria simples testar todas as fases deste projeto.

A nível de tempo de comunicação, o mais rápido que se conseguiu foi um tempo de ciclo de cinco segundos. O que é francamente insuficiente para as espectativas e necessidades deste projeto. Ainda assim, este tempo de ciclo acarretou outro tipo de problemas na aquisição de valores vindos do *GPS*. Por algum motivo ainda desconhecido, se se utilizasse a configuração que leva a este tempo, o *GPS* estaria sempre a enviar a mesma informação para o servidor. Ou seja, seriam enviados sempre os mesmos dados de latitude, longitude, altitude, velocidade, número de satélites e ângulo de viagem. De notar que esta informação não está incorreta, simplesmente está atrasada.

A solução encontrada para este problema foi o atraso da rotina que recebe os dados vindos do *GPS*. Para isso, aumentou-se o tempo do *timer* descrito na secção 4.3. Antes estava a dois mil milissegundos (dois segundos) aumentou-se para sete mil milissegundos (sete segundos) isto levou a um aumento bastante substancial no tempo de transmissão, mas resolveu o problema de estar sempre a enviar a mesma mensagem.

O tempo real de transmissão situa-se na ordem dos 20 segundos. Isto porque quando o *Arduino* envia uma mensagem para o servidor, depois de receber a resposta e tentar enviar a próxima, é tempo suficiente para o servidor fechar a conexão. O *Arduino* entra em *timeout*, é reinicializado e tem de começar o programa do zero. Pelas experiências realizadas isto leva aos tais 20 segundos.

O segundo objetivo prende-se com o alerta a ser gerado em caso de perigo. Este objetivo não ficou concluído. Devido aos atrasos com a configuração da comunicação, não houve tempo para tratar desta importante etapa desta dissertação. No entanto o caminho está traçado para se puder melhorar o trabalho já realizado.

Para além destes objetivos, tentou-se também implementar a escrita de dados num cartão *SD.* No entanto este trabalho teve de ser suspenso porque não se conseguiu fazer com que o *Arduino* reconhecesse o cartão. Tentou-se utilizar vários cartões *SD* e formatá-los de formas diferentes, mas sem sucesso. Tentou-se também utilizar uma app para controlar a comunicação de dados por meio de um botão *Start.* O desenvolvimento desta *app* teve de ser interrompido porque o *SIM900* não suporta estar ligado a dois dispositivos diferentes.

Com base nos resultados adquiridos neste projeto é recomendado que se faça um upgrade à solução proposta. Será necessário usar um dispositivo com comunicação 5G como por exemplo o módulo SIM8202G-M2, ou o SIM8200EA-M2, ou SIM8300G-M2 ou SIM8202G.

O SIM8202G-M2 da SIMCom é um módulo multi-banda 5G NR/LTE-FDD/LTE-TDD/HSPA com um factor de forma M2 e uma velocidade de transmissão de dados de 2.4Gbps. Possui interface com Sim Card, USB, PCM e I^2C. É recomendável utilizar a interface I^2C dado que foi esta que deu menos problemas.

Os módulos com tecnologia 2G apresentam taxas de transferência na ordem dos 64kbps e uma largura de banda de cerca entre os 30 e os 200Khz. Ao passo que os

módulos com tecnologia 5G apresentam uma velocidade de transmissão de até 10Gbps e uma largura de banda de 30 a 300GHz[37]. Com estas melhorias em hardware é espectável uma melhoria nos tempos de transmissão.

É recomendável também utilizar outro tipo de microcontrolador que suporte programação multi-core/ programação paralela. Como *Teensy 3.0* já equipado com um processador Arm, ou um ESP32. Com a programação multi-core é possível fazer várias tarefas ao mesmo tempo. Neste caso, quando os dados estão a ser enviados para o servidor, uma outra thread poderá ocupar-se de estar a recolher dados novos que depois seriam enviados. Isto também fará com que o tempo de comunicação desça. Pode-se mesmo utilizar um Raspberry PI 4 já equipado com placa gráfica e com uma porta Gigabit Ethernet. Poderá ser interessante se se quiser acoplar uma câmara ao projeto para fazer alguns testes com algoritmos de Visão Industrial. É também importante ter atenção à montagem do sistema. É recomendável que em vez de se colocar os sensores no exterior do veículo se coloquem dentro do mesmo para evitar a interferência por vibrações como foi realizado em [38].

Depois de todo o hardware estar configurado é recomendável que se aplique alguns casos de estudo como os referidos na secção **2.3 Casos de Estudo**. Sempre com o intuito de testar a precisão e exatidão dos sensores, e de testar a performance das comunicações.

Referências

[1] G. Growth, “20% fuel savings from eco-driving | Green Growth from Business Growth Hub.” https://www.green-growth.org.uk/article/20pc-fuel-savings-eco-driving (accessed Jun. 21, 2020).

[2] E. E. Agency, “Evolution of GHG emissions from transport in the EU-28 — European Environment Agency.” https://www.eea.europa.eu/data-and-maps/daviz/evolution-of-ghg-emissions-in-2#tab-chart_1 (accessed Jun. 21, 2020).

[3] “Share of transport greenhouse gas emissions — European Environment Agency.” https://www.eea.europa.eu/data-and-maps/daviz/share-of-transport-ghg-emissions-2?fbclid=IwAR3qdC6Jz9XijY1e2Pn4S_LNQA21oOkWDBp85wl5ZEJpanB2CerEPZ6-FZs#tab-chart_1 (accessed Jun. 21, 2020).

[4] “National emissions reported to the UNFCCC and to the EU Greenhouse Gas Monitoring Mechanism — European Environment Agency.” https://www.eea.europa.eu/data-and-maps/data/national-emissions-reported-to-the-unfccc-and-to-the-eu-greenhouse-gas-monitoring-mechanism-15 (accessed Jun. 21, 2020).

[5] Y. Wang and A. Boggio-Marzet, “Evaluation of eco-driving training for fuel efficiency and emissions reduction according to road type,” *Sustain.*, vol. 10, no. 11, 2018, doi: 10.3390/su10113891.

[6] “Eco-Driving Habits Could Save Drivers 20 Percent On Fuel.” https://cleantechnica.com/2013/07/17/drivers-could-save-20-per-cent-on-fuel-by-embracing-eco-driving/ (accessed Jun. 21, 2020).

[7] V. Heije, N. Ligterink, and U. Stewagen, “Potential of eco-driving Deliverable 45.1,” *Udrive Eur. Nat. Driv. Study*, vol. 7, 2017, doi: 10.26323/UDRIVE_D45.1.

[8] “ITS - ITS Portugal.” https://www.its-portugal.com/its (accessed Jun. 21, 2020).

[9] M. Rouse, “What is vehicle to infrastructure (V2I or v2i)? - Definition from WhatIs.com.” https://whatis.techtarget.com/definition/vehicle-to-infrastructure-V2I-or-V2X (accessed Jun. 21, 2020).

[10] I. Rasheed, L. Zhang, and F. Hu, “A privacy preserving scheme for vehicle-to-everything communications using 5G mobile edge computing,” *Comput. Networks*, vol. 176, 2020, doi: 10.1016/j.comnet.2020.107283.

[11] “What is Vehicle-to-Infrastructure (V2I) Communication and why do we need it?” https://www.3m.com/3M/en_US/road-safety-us/resources/road-transportation-safety-center-blog/full-story/~/what-is-vehicle-to-infrastructure-v2i-communication-and-why-do-we-need-it/?storyid=021748d7-f48c-4cd8-8948-b7707f231795 (accessed Dec. 30, 2020).

[12] E. Zlotchenko, “Environmental Justice Considerations for Connected and Automated Vehicles,” 2016. Accessed: Jun. 23, 2020. [Online]. Available: https://www.transportation.gov/AV.

[13] S. Murtha, “Autonomous vs connected vehicles – what’s the difference? – Atkins.” https://www.atkinsglobal.com/en-gb/angles/all-angles/autonomous-vs-connected-vehicles-whats-the-difference (accessed Jun. 21, 2020).

[14] SAE, “SAE International Releases Updated Visual Chart for Its ‘Levels of Driving Automation’ Standard for Self-Driving Vehicles.” https://www.sae.org/news/press-room/2018/12/sae-international-releases-updated-visual-chart-for-its-“levels-of-driving-automation”-standard-for-self-driving-vehicles (accessed Jun. 21, 2020).

[15] A. Nassari, A. Schiphorst, N. di Palo, J. Sotudeh, F. Chaudry, and J. Horne, “2020 Autonomous Vehicle Technology Report.” https://www.wevolver.com/article/2020.autonomous.vehicle.technology.report (accessed Jun. 21, 2020).

[16] S. Chen *et al.*, “Vehicle-to-Everything (v2x) Services Supported by LTE-Based Systems and 5G,” *IEEE Commun. Stand. Mag.*, vol. 1, no. 2, pp. 70–76, 2017, doi: 10.1109/MCOMSTD.2017.1700015.

[17] X. Krasniqi and E. Hajrizi, “Use of IoT Technology to Drive the Automotive Industry from Connected to Full Autonomous Vehicles,” *IFAC-PapersOnLine*, vol. 49, no. 29, pp. 269–274, 2016, doi: 10.1016/j.ifacol.2016.11.078.

[18] R. Molina-masegosa and J. Gozalvez, “T 30 |||,” no. December, 2017.

[19] V. Vukadinovic *et al.*, “3GPP C-V2X and IEEE 802.11p for Vehicle-to-Vehicle communications in highway platooning scenarios,” *Ad Hoc Networks*, 2018, doi: 10.1016/j.adhoc.2018.03.004.

[20] “Passive vs. Active Sensing | Natural Resources Canada.” https://www.nrcan.gc.ca/maps-tools-publications/satellite-imagery-air-photos/remote-sensing-tutorials/introduction/passive-vs-active-sensing/14639 (accessed Jun. 21, 2020).

[21] F. Rosique, P. J. Navarro, C. Fernández, and A. Padilla, “A systematic review of perception system and simulators for autonomous vehicles research,” *Sensors (Switzerland)*, vol. 19, no. 3. MDPI AG, Feb. 01, 2019, doi: 10.3390/s19030648.
[22] “CCD versus CMOS: Which is Better? - Astronomy & Scientific Imaging Solutions.” https://diffractionlimited.com/ccd-versus-cmos-better/ (accessed Jun. 21, 2020).
[23] Gondek, “Optical optimalization of organic solar cell wyth BH,” *Opto-Electronics Rev.*, vol. 22, no. 2, pp. 77–85, 2010, doi: 10.2478/s11772.
[24] DfT, “Efficient Driving A Rapid Evidence Assessment for the Department for Transport A Rapid Evidence Assessment for the Department for Transport,” 2016.
[25] J. M. Bandeira, P. Fernandes, T. Fontes, S. R. Pereira, A. J. Khattak, and M. C. Coelho, “Exploring multiple eco-routing guidance strategies in a commuting corridor,” *Int. J. Sustain. Transp.*, vol. 12, no. 1, pp. 53–65, 2018, doi: 10.1080/15568318.2017.1328545.
[26] “1 5Gaa P-180106,” pp. 1–77.
[27] “Arduino Mega 2560 Rev3 | Arduino Official Store.” https://store.arduino.cc/arduino-mega-2560-rev3 (accessed Jan. 05, 2021).
[28] “Funduino Mega2560 (Arduino Compatible).” https://grobotronics.com/funduino-mega2560-rev3-arduino-mega-compatible.html?sl=en (accessed Jan. 05, 2021).
[29] Ada, “Adafruit Ultimate GPS Logger Shield,” *Adafruit*, 2016, [Online]. Available: https://learn.adafruit.com/adafruit-ultimate-gps-logger-shield/downloads.
[30] “In-Depth: Send Receive SMS & Call with SIM900 GSM Shield & Arduino.” https://lastminuteengineers.com/sim900-gsm-shield-arduino-tutorial/ (accessed Jan. 05, 2021).
[31] Garmin, “Lidar Lite v3 Operation Manual and Technical Specifications Laser Safety,” p. 14, 2016, [Online]. Available: http://static.garmin.com/pumac/LIDAR_Lite_v3_Operation_Manual_and_Technical_Specifications.pdf.
[32] “MAMP & MAMP PRO - your local web development solution for PHP and WordPress development.” https://www.mamp.info/en/windows/ (accessed Jan. 05, 2021).
[33] J. Paulo Santos, “3 PROTOCOLO DE COMUNICAÇÃO SÉRIE EIA232, Informática Industrial,” pp. 13–24, 2011.
[34] J. Paulo Santos, “Capítulo 10 i2c, Informática Industrial,” pp. 134–150, 2011.
[35] J. Paulo Santos, “Capítulo 9 tcp, Informática Industrial,” pp. 78–86, 2011.
[36] A. O. Guide and L. Wijesinghe, “Online HTTP Information Server and SMS Alerts System for Mobile Weather Station through GSM Wireless Communication,” pp. 1–42, 2017.
[37] RantCell, “Comparison of 2G 3G 4G 5G | 2G vs 3G vs 4G vs 5G | Rantcell.” https://rantcell.com/comparison-of-2g-3g-4g-5g.html (accessed Jan. 12, 2021).
[38] L. Vasconcelos, L. Neto, S. Santos, A. B. Silva, and Á. Seco, “Calibration of the gipps car-following model using trajectory data,” *Transp. Res. Procedia*, vol. 3, no. July, pp. 952–961, 2014, doi: 10.1016/j.trpro.2014.10.075.