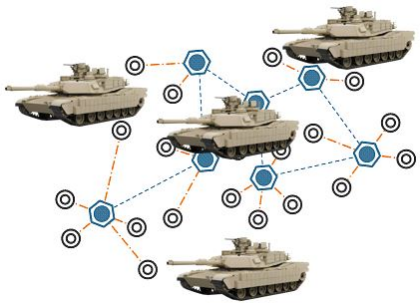
TÉCNICO LISBOA

ACADEMIA MILITAR
DULCE ET DECORUM EST PRO PATRIA MORI



Rede de Sensores Acústicos para Vigilância e Aquisição de Alvos em Operações Militares Terrestres

Gonçalo Couteiro Atanásio

Dissertação para obtenção do Grau de Mestre em

Engenharia Eletrotécnica e de Computadores

Orientador(es): Prof. António Manuel Raminhos Cordeiro Grilo
Prof. António Joaquim dos Santos Serralheiro

Júri

Orientador: Prof. António Manuel Raminhos Cordeiro Grilo
Presidente: Prof. Nuno Cavaco Gomes Horta
Vogal: Prof. José António Beltran Gerald

Outubro 2016

Esta dissertação é dedicada a toda a minha família e à minha namorada, por todo o incentivo e apoio incondicional que me proporcionaram.

Agradecimentos

Estes seis anos de trabalho culminam com a elaboração de uma dissertação baseada nos conhecimentos adquiridos ao longo de uma aliciante jornada, plena de desafios. O trabalho ora apresentado consiste no primeiro passo para um caminho que irei percorrer como Oficial de Transmissões do Exército Português.

Em primeiro lugar, quero agradecer à minha família, especialmente aos meus pais e à minha tia, por todo o apoio que me têm dado e pela constante disponibilidade para me ajudarem a ultrapassar qualquer obstáculo. Quero também agradecer à minha namorada pelo apoio incondicional que me tem dispensado em todos os momentos críticos da minha vida.

Também o apoio prestado pelos Oficiais, Sargentos e Praças da Escola das Armas, do Campo Militar de Santa Margarida e do Campo de Tiro de Alcochete envolvidos no trabalho que me encontro a desenvolver, mais concretamente na recolha de som de viaturas pertencentes ao Exército e à Força Aérea, se revestiu de grande importância.

Por último, mas não menos importante, quero expressar os meus agradecimentos ao Professor António Manuel Raminhos Cordeiro Grilo e ao Professor António Joaquim dos Santos Serralheiro, pela total disponibilidade, paciência e apoio na concretização e enriquecimento deste projeto.

Resumo

Este trabalho consiste na identificação de carros de combate utilizando uma rede de sensores sem fios que contém sensores acústicos.

O objetivo é captar som das viaturas através de sensores acústicos contidos nos nós, efetuar a classificação da viatura no próprio nó e enviar o pacote para a estação base, utilizando encaminhamento e agregação de dados.

Foi criada uma base de dados de sons de viaturas das Forças Armadas Portuguesas, treinando modelos GMM com dados que exploram as características do som, nomeadamente os MFCC e a deteção de ritmo. Através dos modelos GMM e do som recolhido torna-se possível efetuar a classificação da viatura.

Foi utilizado o simulador NS3 para simular a rede de sensores sem fios e a respetiva agregação de dados e encaminhamento. O protocolo de encaminhamento usado foi o DSDV, e este foi testado utilizando duas tecnologias distintas: o *IEEE 802.15.4* e o *IEEE 802.11g*.

Para os testes do classificador, o número de misturas Gaussianas foi otimizado e os testes foram realizados utilizando diferentes janelas de tempo para os MFCC. Chegou-se à conclusão de que o classificador é melhor sem a componente de deteção de ritmo, ou seja: utilizando apenas os MFCC é possível obter-se um F-score de 0,875 com oito viaturas na base de dados.

As simulações realizadas utilizando o NS3 indicam que é necessário verificar-se um compromisso entre os atrasos, o alcance rádio e o consumo de energia. É desejável que a rede tenha alguma autonomia, desde que o atraso não seja elevado, para que a identificação ocorra o mais próximo possível do tempo real. A tecnologia *802.15.4* com agregação seria a mais indicada no caso em estudo. Na eventualidade de se pretender cobrir áreas demasiados abrangentes, por necessidade tática, a mais indicada seria a tecnologia *802.11g*.

Palavras-chave: Rede de sensores sem fios, GMM, MFCC, Deteção de ritmo, DSDV, IEEE 802.15.4, IEEE 802.11g.

Abstract

This Master Thesis consists in the identification of combat vehicles with a wireless sensor network that contains acoustic sensors.

The objective is to gather the sound from the combat vehicles, with acoustic sensors, in the nodes of the wireless sensor network, do the classification of the car in the node, and then send the packet to sink using packet routing and data aggregation.

The data base of sound, from combat cars of the Armed Forces, was made training GMM models with data that explore characteristics of the sound, namely the MFCC and rhythm detection. With the GMM models and the sound captured it becomes possible to make the vehicle classification.

Was used the NS3 simulator, to simulate the wireless sensor network and the respective data aggregation and routing. The routing used is DSDV, and it was tested using two different technologies, IEEE 802.15.4 and IEEE 802.11g.

For the tests of the classifier, the number of Gaussian mixtures was optimized and the tests were made with different time windows for the MFCC. It was concluded that the classifier is better without the rhythm detecion, that is, using only the MFCC, and it can obtain an F-score of 0.875 with eight combat cars on the data base.

The simulations done using the NS3 simulator, indicate that there must be a trade-off between the delays, the radio range and power consumption. It is intended for the network to have some autonomy, but with a low delay, so that the identification could be close to real time. The *802.15.4* technology, with aggregation, would be the most appropriate choice to the case study or, if it's desired to cover big areas, because of tactical restrictions, the most suitable one would be the technology *802.11g*.

Keywords: Wireless sensor network, GMM, MFCC, Rhythm detection, DSDV, IEEE 802.15.4, IEEE 802.11g.

Conteúdo

Lista de Tabelas

Lista de Figuras

Lista de Blocos de Código

Lista de Símbolos

Símbolos gregos

β	Parâmetro regulador de peso atribuído à precisão ou ao *recall*
$\gamma()$	Função expetativa para cada ponto
∞	Infinito
μ	Vetor de médias
Σ	Matriz de covariâncias

Símbolos romanos

e	Função exponencial
F_β	*F-score*
$N()$	Função densidade probabilidade para Gaussiana
p	Precisão
r	*recall*
T_{tx}	Tempo de transferência de dados
X	Vetor de amostras
$p()$	Probabilidade
f	Frequência
m	Número de Mel's

Subscritos

$x, n, k,$	Índices

Sobrescritos

-1	Inversa
T	Transposta

Lista de Abreviaturas

6LoWPAN	IPv6 over Low power Wireless Personal Area Networks
AES	Advanced encryption standard
AIC	Akaike information criterion
ARP	Address Resolution Protocol
ASK	Amplitude shift keying
BIC	Bayesian information criterion
BPM	Batidas por minuto
BPSK/QPSK	Binary/Quardrature phase SK
BSS/ESS	Basic/Extended service set
CBC-MAC	Cipher block chaining message authentication code
CCK	Complementary code keying
COFDM	Coded OFDM
CRC	Cyclic redundancy check
DD	Directed Difusion
DODAG	Destination Oriented Directed Acyclic Graph
DSDV	Destination-Sequenced Distance-Vector
DSSS	Direct-sequence spread spectrum
EM	Expectation-Maximization algorithm
FN	Falso negativo
FP	Falso positivo
GEAR	Geographic and Energy Aware Routing
GFSK	Gaussian frequency SK
GMM	Gaussian Mixture Model
GPS	Global positioning system
ICMPv6	Internet Control Message Protocol Version 6
IEEE	Institute of Electrical and Electronics Engineers
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6

IP	Internet Protocol
LEACH	Low Energy Adaptive Clustering Hierarchy
LR-WPAN	Low-Rate Wireless Personal Area Network
M-QAM	M-ary quadrature amplitude modulation
MAC	Media Access Control
MFCC	Mel-Frequency Cepstral Coefficients
MLE	Maximum-likelihood Estimation
NDP	Neighbor Discovery Protocol
O-QPSK	Offset-QPSK
OFDM	Orthogonal frequency division multiplexing
RAM	Random Access Memory
RFC	Request for Comments
RPL	Routing Protocol for Low-Power and Lossy Networks
TCP/IP	The Internet protocol suite
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VN	Verdadeiro negativo
VP	Verdadeiro positivo
WPA	Wi-Fi protected access
WiFi	Wireless local area network products that are based on the IEEE 802.11 standards.

Capítulo 1

Introdução

Com a evolução da tecnologia, as redes de sensores sem fios começaram a ter aplicação nas mais diversas áreas, havendo, frequentemente, necessidade de um tráfego de dados bastante elevado com recursos de *hardware* limitados, o que leva a um consumo insustentável de energia. Desta forma, são necessários algoritmos de encaminhamento com baixo consumo de energia, assim como agregação de dados, com o objetivo de aumentar o tempo de vida útil dos nós. Estes são constituídos por componentes que permitem a aquisição, processamento e envio/receção de dados.

Quando inseridas num âmbito militar, as redes de sensores implicam determinadas preocupações e requerem certos cuidados, tais como, resistência a alterações na rede, redundância, escalabilidade, baixo consumo energético e baixa latência, para que a identificação e classificação dos veículos seja tão imediata quanto possível.

A utilização de redes de sensores sem fios apresenta algumas vantagens quando comparando com as cabladas: é possível reduzir os custos de manutenção, facilitar a instalação e, permitir minimizar o efeito de eventuais falhas, uma vez que, nas cabladas, é possível que ocorra a quebra dos cabos (Chris Townsend, 2004). No contexto militar, consoante a natureza da operação a realizar, poderá não ser possível efetuar as ligações físicas dos nós, ou poderá haver necessidade de proceder à sua rápida instalação. Consequentemente, em cenários militares, torna-se vantajosa a utilização de redes de sensores sem fios.

1.1 Motivação

Este tema traz inúmeras vantagens no que toca ao futuro das transmissões no campo de batalha. Sendo um Oficial Aluno da Arma de Transmissões, e tendo um grande interesse pelas áreas de redes de sensores sem fios e processamento de sinal, considero bastante apelativo e desafiante o estudo deste tipo de tecnologia, quando aplicado à vigilância e aquisição de alvos no campo de batalha. Esta tecnologia, uma vez que se encontra ao alcance de ambos os lados, pode ser utilizada pelos adversários. Assim sendo, a vantagem irá para aquele que melhor compreender o seu alcance e as suas limitações. É necessário saber aproveitar e entender os recursos tecnológicos, com o intuito de desenvolver formas

de superar obstáculos ou eliminar possíveis ameaças. É de grande importância tática, nos dias de hoje, ter a capacidade de detetar e classificar viaturas num contexto operacional, já que, cada vez mais, se constata a existência de uma evolução no domínio tecnológico por parte dos adversários, o que exige uma maior rapidez na aquisição de informação para que a tomada de decisão seja feita da forma mais eficiente possível. Em suma, esta tecnologia permite otimizar o comando e controlo. Deste modo, com a correta aplicação de dispositivos tecnologicamente avançados, é possível incrementar a proteção das forças militares no campo de batalha.

Mesmo com a existência de sensores acústicos no exército (de Almeida et al., 2009), o desenvolvimento deste tipo de tecnologia torna-se importante pois os atuais sistemas não permitem realizar deteção e identificação viaturas, o que, nos dias de hoje, acaba por constituir uma limitação significativa.

1.2 Objetivos do trabalho

No caso em estudo, vai ser aplicada uma rede de sensores sem fios em ambiente militar, cuja finalidade é a deteção e a classificação de veículos militares em contexto operacional. Estas redes de sensores são, normalmente, constituídas por um conjunto de nós estacionários, dispersos numa determinada área. Os objetivos da aplicação são:

- Recolha de som de viaturas;
- Classificação da viatura através de processamento no nó;
- Encaminhamento do resultado da classificação até à estação base;
- Utilização de agregação de dados em nós intermédios de forma a reduzir o consumo de energia da rede.

1.2.1 Descrição do sistema

O sistema que se pretende desenvolver encontra-se sumariamente representado na Figura 1.1. Esta figura, mostra uma viatura a aproximar-se da rede de sensores sem fios. A viatura emite um determinado som, o qual é recolhido pelos nós da rede mais próximos desta. Quando um nó recolhe um trecho de som, através de um microfone, efetua o seu pré-processamento, mediante extração de características espectrais. Cada nó compreende uma base de dados, contendo os modelos GMM que representam as várias viaturas, que é utilizada para efetuar a classificação do trecho de som recolhido.

Depois de um nó efetuar a classificação, é enviado um pacote com o resultado desta, tendo como destino a estação base. Dado que existem nós que se encontram a grandes distâncias da estação base, não tendo potência suficiente para comunicar diretamente com esta, é necessário recorrer a outros nós que efetuem o encaminhamento destes pacotes. Os nós, para além da tarefa de encaminhamento, vão também realizar agregação de dados (eliminando redundância de dados), limitando o número total de envios de pacotes, o que leva a uma redução no consumo de energia e logo uma maior autonomia da rede.

Todos estes mecanismos referidos na descrição do sistema vão descritos nos Capítulos 2 e 3.

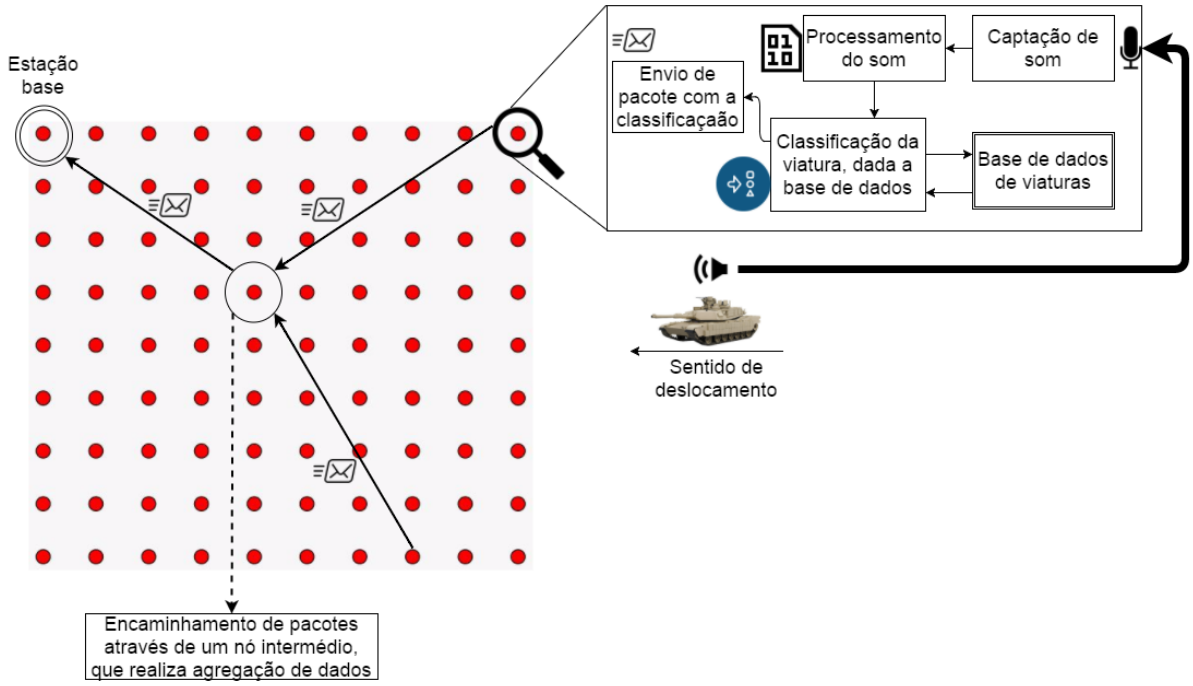


Figura 1.1: Descrição do sistema.

1.3 Contribuições

Deste trabalho resultaram as contribuições que se seguem e que poderão ser úteis para trabalho futuro:

- **Base de dados de som de viaturas** - Esta pode ser fornecida, mediante requisição e aprovação por entidades competentes;
- **Módulo DSDV, versão IPv6, para o simulador NS-3** - Este módulo inclui a modificação que permite escolher os nós que propagam a rota e encontra-se submetido para revisão como candidato a ser incluído numa próxima versão do simulador NS-3. O *patch* pode ser adquirido no repositório `<https://codereview.appspot.com/310910043/>`;
- **Código Matlab associado ao treino e classificação de viaturas** - Incluído no Apêndice A.

1.4 Estrutura do documento

Neste documento, é feita uma revisão de literatura necessária à realização deste trabalho, que está contida no Capítulo 2. Neste, o estudo de deteção e rastreio de veículos militares é iniciado com a caracterização da viatura através de uma determinada assinatura acústica, que é criada com base nas características espetrais do som emitido e nas batidas por minuto, descritas na Secção 2.1.

Nesta aplicação específica, os nós começam por adquirir amostras de som e, posteriormente, estas são processadas recorrendo aos Mel-Frequency Cepstral Coefficients (MFCC), que por si recorrem

à Transformada rápida de Fourier (Lutter, 2014) (Secção 2.1.2.1). Estas ferramentas são usadas de forma a conseguir efetuar uma análise em frequência das amostras de som captadas (Lutter, 2014). De seguida, é feita uma análise do ritmo, utilizando as batidas por minuto (BPM) (Secção 2.1.2.2).

Posteriormente, os MFCC e os BPM, são representados num conjunto de funções densidade probabilidade, denominado Gaussian Mixture Model (GMM) (Wu, 2005). Pretende-se, com estes, criar uma base de dados, na qual estão incluídas assinaturas de certos tipos de viaturas (representadas em GMM). Para isto, é necessário efetuar um treino com várias amostras, de modo a que a classificação da viatura seja feita da melhor forma possível e por comparação dos GMM (Secção 2.1.3 e 2.1.4).

Na Secção 2.2 são abordados aspetos relacionados com a rede de sensores em si. Nesta secção, estudar-se-ão os seguintes tópicos: as arquiteturas dos nós (Secção 2.2.1); algumas arquiteturas de rede existentes e suas vantagens/desvantagens (Secção 2.2.2); agregação de dados e os seus mecanismos (Secção 2.2.3); aspetos importantes relacionados com a deteção e rastreio do movimento de alvos, nomeadamente viaturas (Secção 2.2.4); tecnologias de comunicação em redes de sensores sem fios (Secção 2.2.5); e também a pilha de protocolos IP (Secção 2.2.6).

No Capítulo 3 é descrita a implementação realizada, relativamente à caracterização e classificação de viaturas (Secção 3.1) e à rede de sensores sem fios (Secção 3.2).

No Capítulo 4 são apresentados os resultados dos testes realizados com dados sintéticos e reais relacionados com a geração dos modelos e com a classificação (Secção 4.1), bem como os resultados relativos à simulação da rede de sensores com a respetiva agregação e encaminhamento de pacotes (Secção 4.2).

Por último, no Capítulo 5 são descritas as conclusões resultantes do trabalho realizado, bem como indicações de possíveis trabalhos futuros a realizar neste âmbito.

Capítulo 2

Estado da Arte

2.1 Aquisição e análise do sinal de áudio

Na situação em análise, a aquisição de um sinal de áudio é feita através de um sensor acústico contido nos nós da rede de sensores sem fios. Este sensor requer uma sensibilidade razoável para que o som seja reconhecido o mais cedo possível, o que, consequentemente, conduzirá a uma maior rapidez no processamento das amostras.

Os objetivos, no que respeita à aquisição e análise do sinal de áudio, são:

- Aquisição do sinal de áudio pelos sensores acústicos;
- Cálculo dos MFCC da amostra;
- Cálculo das batidas por minuto médias da amostra;
- Representação dos MFCC e batidas em GMM;
- Comparação com a base de dados;
- Identificação da viatura, caso exista na base de dados.

2.1.1 Componentes do som

Um som, segundo Mott (1990), é composto pelas nove componentes descritas abaixo:

- **Componentes musicais**:
  - Tom - Determinado pela frequência do som. Existem frequências baixas, médias e altas (Mott, 1990);
  - Timbre - É uma combinação entre a frequência fundamental e as harmónicas do som, que permite que certas vozes, instrumentos musicais e sons de várias naturezas possam ser classificados como únicos (Mott, 1990). É possível verificar na Figura 2.1 que existe uma grande diferença entre um espetro de uma voz e um espetro de uma viatura;

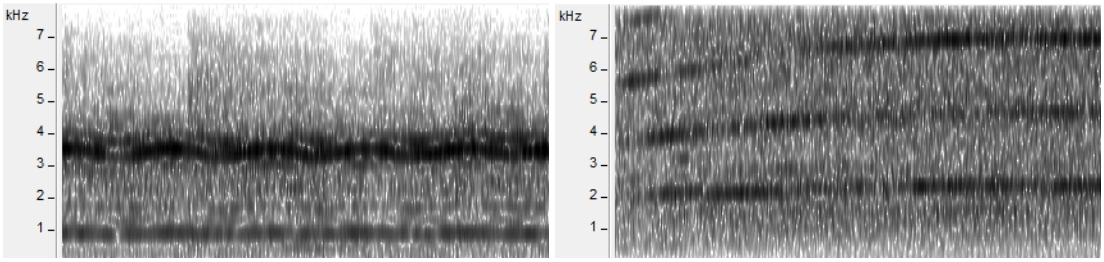
- Harmónicas - Quando existe vibração de um objeto, são propagadas ondas sonoras com uma certa frequência fundamental, da qual resultam outras frequências múltiplas denominadas harmónicas. Como referido no ponto anterior, a junção da frequência fundamental com as harmónicas, determina o timbre do som. Quantas mais harmónicas estiverem contidas num som, mais agradável se torna ao ouvido humano (Mott, 1990);
- Sonoridade - Depende da intensidade do estímulo sonoro. A sonoridade só se torna significativa quando existe comparação entre dois ou mais sons. Uma explosão de dinamite tem mais sonoridade que um tiro de pistola, pois tem a capacidade de agitar mais moléculas sonoras. Quando comparamos o disparo de uma pistola em diferentes cenários - numa estação de comboios e numa sala fechada, este pode passar despercebido caso esteja a passar um comboio na estação, mas não passar despercebido numa sala fechada (Mott, 1990);
- Ritmo - É uma das componentes mais complexas do som, uma vez que é um som recorrente que alterna entre elementos altos e baixos. Contem uma batida ou uma pulsação, um passo ou um andamento e um padrão de acentuações ou batidas mais fortes ou mais fracas (Mott, 1990).

- **Componentes da envolvente espetral** (representados na figura 2.2):
  - Ataque - O som é iniciado com o ataque e este pode ser classificado como rápido ou lento, dependendo da sua duração (Mott, 1990). Encontra-se representado na figura 2.2 com a linha a vermelho;
  - Sustentação - Após o ataque, quando o som atinge o seu pico, a sustentação vai depender da energia gerada pela fonte das vibrações. Após a fonte parar de fornecer energia, o som começa a decair (Mott, 1990). Encontra-se representado na figura 2.2 com a linha a azul;
  - Decaimento - Como referido, quando a fonte para de fornecer energia inicia-se o decaimento, que corresponde ao tempo que som leva a atingir o silêncio (Mott, 1990). Encontra-se representado na figura 2.2 com a linha a verde.

- **Componente de gravação e reprodução**:
  - Velocidade - Ao aumentar ou diminuir a velocidade de reprodução, é possível alterar as propriedades do som. Por exemplo, aumentando a velocidade de reprodução original para o dobro, consegue-se que uma explosão seja ouvida como um tiro (Mott, 1990).

Para que as classificações com os sinais de áudio sejam feitas da melhor forma, proceder-se-á ao estudo de mais do que uma componente do som.

Pretende-se efetuar uma análise do som em toda a gama de frequências – timbre - e uma análise de uma das componentes do ritmo. Para este fim, vão ser estudados os MFCC e a deteção das batidas por minuto do som ao longo do tempo.



(a) Espetrograma de uma voz humana. (b) Espetrograma de uma viatura.

Figura 2.1: Espetrograma de uma voz humana e de uma viatura

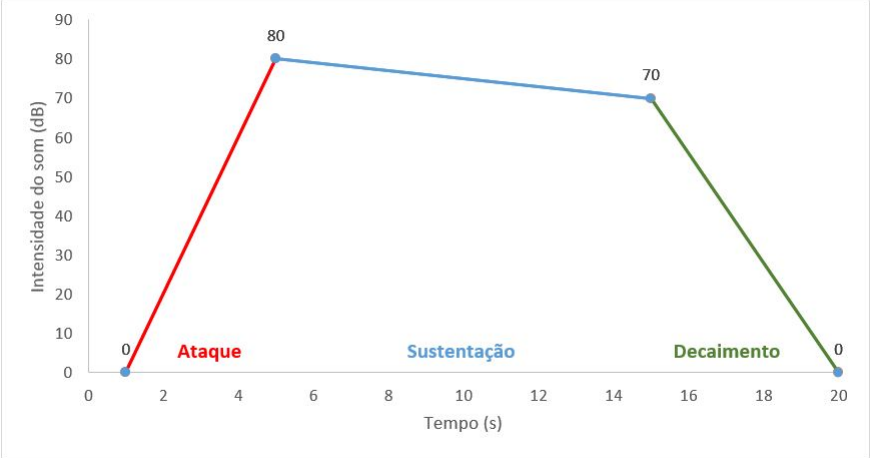


Figura 2.2: Componentes da envolvente espectral.

2.1.2 Extração de parâmetros dos sinais sonoros

Nesta secção vão ser abordados os métodos utilizados para a extração de parâmetros do som - MFCC e BPM - que estudam respetivamente o timbre e o ritmo. Estes métodos vão produzir dados que, posteriormente, vão ser utilizados pelos GMM para gerar os modelos associados a cada viatura.

2.1.2.1 *Mel-Frequency Cepstral Coefficients* (MFCC)

O conceito de *Cepstrum* foi definido no ano de 1963 num trabalho realizado por Bogert et al. (1963). Este define-se como o resultado da transformada inversa do logaritmo do espectro de um sinal (Taylor, 2009).

A escala de Mel resulta de tentativas de divisão de janelas de frequências, em secções. Desta forma, uma nova escala foi definida, onde um Mel equivale a um milésimo do *pitch* de um tom com 1kHz. O *pitch* permite que um som seja classificado em frequência como alto ou baixo (Taylor, 2009).

A fórmula seguinte converte frequência em Mel (Taylor, 2009):

$$m = 2595 log_{10}(1 + \frac{f}{700}) \tag{2.1}$$

Vários autores defendem a vantagem no uso de uma escala de frequências não-linear, uma vez que o principal objetivo é a aproximação às características auditivas humanas. Estas teorias são resultantes

do facto de o sistema auditivo humano ter uma perceção de intensidade e de tom logarítmica, ou seja, pode ser representado em logaritmo da frequência.

Os MFCC são tipicamente usados em aplicações de reconhecimento de fala, pois este método aproveita a capacidade do sistema auditivo humano para criar um vetor que contém informação sobre mensagem linguística. Os MFCC tentam copiar as características do sistema auditivo humano e, ao mesmo tempo, eliminar certas harmónicas que não são relevantes (Lutter, 2014).

Mesmo sendo usados maioritariamente em reconhecimento de fala, os MFCC podem ser adaptados a outros tipos de aplicações. A sua utilização numa componente militar mostra-se vantajosa, sobretudo no caso em estudo, onde é necessário efetuar uma análise em frequência e, posteriormente, efetuar uma classificação a partir de uma assinatura em frequência.

De uma forma geral, o cálculo dos MFCC pode ser dividido em várias etapas, que se encontram representadas na Figura 2.3 e que são:

- Dividir o sinal recolhido em pequenos segmentos de som;
- Para cada segmento, aplicar a transformada rápida de Fourier;
- Aplicar os filtros de Mel e obter o respetivo espectro;
- Efetuar logaritmo do espectro de Mel;
- Efetuar a análise cepstral (transformada de cosseno discreta);
- Efetuar a derivada para obter os MFCC.

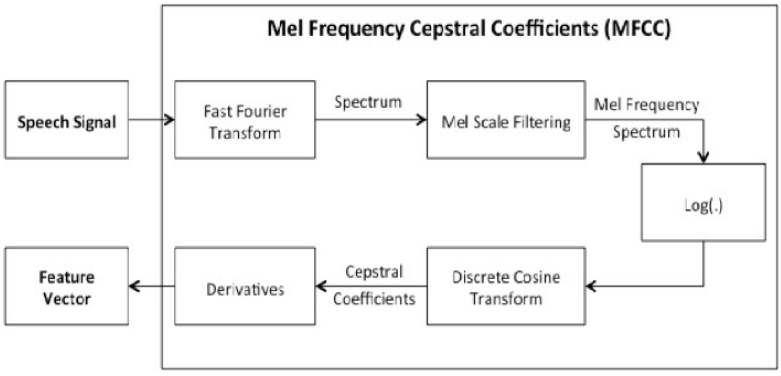


Figura 2.3: Diagrama de blocos do algoritmo dos MFCC (Lutter, 2014).

2.1.2.2 Batidas por minuto (BPM)

Segundo Ellis (2007), a fim de efetuar a deteção das batidas num som, é necessário satisfazer duas condições:

- Os instantes selecionados devem corresponder aos momentos em que uma batida do áudio é indicada, por exemplo, pelo aparecimento de uma nota gerada por um dos instrumentos (Ellis, 2007);

- Deverá existir um intervalo entre batidas localmente constante, uma vez que é este espaçamento regular entre batidas que define o ritmo musical (Ellis, 2007).

Ellis (2007) elaborou um sistema de deteção de batidas em que é inicialmente estimado o andamento global. Posteriormente, é utilizado este mesmo andamento para construir uma função de custo de transição e, por fim, utiliza programação dinâmica para encontrar o melhor conjunto de resultados que refletem o andamento e os respetivos momentos de maior "força de início". Ellis (2007) também desenvolveu código Matlab (Little and Moler, 2015) para aplicar este sistema, trabalho que se revestirá de utilidade na deteção de batidas necessárias ao estudo que irei realizar.

2.1.3 Classificação de som

Para efetuar a classificação de som recolhido, como correspondendo à viatura correta, torna-se necessária a utilização de um classificador. Nesta Secção é estudado o GMM (*Gaussian Mixture Model*), que é treinado utilizando os MFCC e/ou BPM. Este treino é descrito na Secção 2.1.4.

O GMM insere-se na classe de sistemas de reconhecimento de padrões, permitindo modelar e representar a função densidade de probabilidade das amostras observadas, utilizando misturas de densidades Gaussianas. Dado o vetor de entrada, o GMM define os pesos de cada distribuição através de algoritmos, como, por exemplo, o *Expectation-Maximization algorithm* (EM) (Wu, 2005)

A função densidade de probabilidade associada a uma gaussiana com várias dimensões é a seguinte (Bishop, 2006):

$$N(x|\mu_x, \Sigma_x) = p(x|\mu_x, \Sigma_x) = \frac{1}{\sqrt{2\pi|\sum_x|}} e^{(-\frac{1}{2}(x-\mu_x)\sum_x^{-1}(x-\mu_x)^T)} \tag{2.2}$$

Onde μ_x é um vetor de médias, Σ_x é a matriz de covariância e x o vetor de amostras que neste caso está associado aos MFCC e/ou BPM.

Para várias misturas, define-se da seguinte forma (Bishop, 2006):

$$p(x) = \sum_{k=1}^{K} \pi_k N(x|\mu_x, \Sigma_x) \tag{2.3}$$

Onde π_k é o coeficiente de mistura, ou seja, a variável que atribui o peso a cada uma das Gaussianas num conjunto destas. Verificando os pesos das misturas a restrição (Bishop, 2006):

$$\sum_{k=1}^{K} \pi_k = 1 \tag{2.4}$$

Existindo um modelo GMM que representa os dados, é possível proceder à classificação de um trecho de som dado um modelo, utilizando a Equação 2.5. Esta equação calcula a verosimilhança (log-likelihodd) associada à distribuição Gaussiana, dado um conjunto de dados. A verosimilhança é definida como sendo $-ln\, p(X|\mu, \Sigma, \pi)$ (Bishop, 2006). Desta forma, quanto maior for a probabilidade, menor é o *log-likelihood*.

É então testado um trecho de som para todos os modelos de viaturas existentes, utilizando a Equação 2.5:

$$ln\, p(X|\mu, \Sigma, \pi) = -\sum_{n=1}^{N} ln \left\{ \sum_{k=1}^{K} \pi_k N(x_n|\mu_k, \Sigma_k) \right\} \quad (2.5)$$

O classificador retorna como sendo o modelo adequado aquele que resulta de uma verosimilhança mais baixa.

Este tipo de classificação torna-se bastante vantajosa para a aplicação em estudo, pois o único elemento necessário a transmitir na rede, é o número da viatura resultante da classificação efetuada pelo nó. Posteriormente, na estação base, são avaliadas as várias classificações feitas pelos nós.

Como resultado, esta alternativa permite transmitir menos bits e gastar menos energia nas transmissões, tornando vantajosa a sua utilização.

2.1.4 Treino dos dados recolhidos

Como referido na Secção 2.1.3, são utilizados os GMM de modelos gerados para realizar a classificação. Os modelos GMM são treinados utilizando os dados dos MFCC e/ou BPM.

Para gerar modelos representativos, é necessário utilizar uma grande quantidade de dados, em comparação com os dados utilizados para teste (classificação). Para gerar estes modelos GMM, procedimento denominado de treino, é necessário calcular as médias, as covariâncias e o peso de cada mistura (coeficientes de mistura) para cada distribuição Gaussiana. Consegue-se obter estas medidas através do algoritmo EM, caracterizado como um método iterativo, com o objetivo de encontrar o *Maximum-likelihood Estimation* (MLE) (Plasse, 2013).

O MLE é um método utilizado para estimar os parâmetros de um modelo estatístico. Este método procura encontrar os valores das médias, covariâncias e coeficientes de mistura, de modo a maximizar a probabilidade dos dados amostrados (Plasse, 2013). Neste caso a função densidade de probabilidade utilizada é a associada à Gaussiana.

Pretende-se então, com este método, encontrar o mínimo *log-likelihood* e calcular as médias, covariâncias e coeficientes de mistura associadas a este.

Em cada iteração, o EM está dividido em dois passos:

- Efetua uma expetativa (E) - Calcula uma expetativa para cada ponto, através dos parâmetros de média, covariância e coeficientes de mistura.
- Efetua uma maximização (M) - Recalcula os parâmetros de média, covariância e coeficientes de mistura para os novos valores da expetativa.
- Avalia o *log-likelihood*, utilizando a Equação 2.5: se convergir, termina o algoritmo e obtém-se o *log-likelihood* mínimo, bem como os parâmetros de média, covariância e coeficientes de mistura associados a este.

A função que calcula a expetativa para cada ponto é (Bishop, 2006):

$$\gamma(z_{nk}) = \frac{\pi_k N(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x_n|\mu_j, \Sigma_j)} \quad (2.6)$$

Para calcular as novas médias é utilizada a expressão (Bishop, 2006):

$$\mu_x = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n \tag{2.7}$$

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}) \tag{2.8}$$

Para calcular as novas covariâncias é utilizada a expressão (Bishop, 2006):

$$\Sigma_x = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_x)(x_n - \mu_x)^T \tag{2.9}$$

Para calcular os novos coeficientes de mistura é utilizada a expressão (Bishop, 2006):

$$\pi_k = \frac{N_k}{N} \tag{2.10}$$

Após concluir o algoritmo de EM, obtém-se, através do vetor de médias, matriz de covariâncias e vetor de coeficientes de mistura, um GMM para cada conjunto de MFCC e/ou BPM ou seja, um modelo para cada viatura passível de ser armazenado em base dados.

2.1.5 Medidas de desempenho

Para que a classificação das viaturas seja feita da forma mais correta e eficiente, é necessário treinar, da melhor forma possível, o classificador. Concluído o seu treino, é de boa prática verificar o bom funcionamento do mesmo através de medidas de desempenho.

O desempenho de um sistema é normalmente testado usando os indicadores de precisão, *recall* e *F-score* (Goutte and Gaussier, 2005).

Para explicar estes indicadores, associar-se-á a cada amostra um nível binário *l*, que define se este está ou não correto, e um nível binário *z*, correspondente à classificação feita pelo sistema - verdadeiro ou falso.

Os resultados experimentais podem ser inseridos numa tabela de Confusão, como a Tabela 2.1. A Tabela 2.2 contém o significado dos campos representados na Tabela 2.1 (Goutte and Gaussier, 2005).

Tabela 2.1: Tabela de Confusão (Goutte and Gaussier, 2005).

		z	
		+	-
l	+	VP	FN
	-	FP	VN

Tabela 2.2: Significado dos campos representados na Tabela 2.1

+	Relevante
-	Não relevante
VP	Verdadeiro Positivo
FP	Falso Positivo
FN	Falso Negativo
VN	Verdadeiro Negativo

Através das contagens de VP, FP, FN e VN é possível calcular a precisão (*p*), representada na equação 2.11 e o *recall* (r), representado na equação 2.12 (Goutte and Gaussier, 2005).

$$p = \frac{VP}{VP+FP} \tag{2.11}$$

$$r = \frac{VP}{VP+FN} \tag{2.12}$$

Para calcular o *F-score*, é usada a média harmónica dos valores de precisão e *recall*:

$$F_\beta = (1+\beta^2)\frac{pr}{r+\beta^2 p} = \frac{(1+\beta^2)VP}{(1+\beta^2)VP+\beta^2 FN+FP} \tag{2.13}$$

Na equação 2.13 o valor de β define o peso atribuído à precisão ou ao *recall*. Ou seja, $\beta < 1$ coloca mais peso na precisão e $\beta > 1$ coloca mais peso no *recall*. Caso $\beta = 0$ só é considerada a precisão, caso $\beta = \infty$ só é considerado o *recall*.

Tanto a precisão como o *recall* têm significado em termos probabilísticos. A precisão pode ser definida como a probabilidade de um objeto ser relevante, dado que é retornado pelo sistema. O *recall* pode ser definido como a probabilidade de um objeto relevante ser retornado (Goutte and Gaussier, 2005):

$$p = p(l = +|z = +) \quad r = p(z = +|l = +) \tag{2.14}$$

Estes três indicadores podem então ser usados como medidas de desempenho para as classificações de amostras que irão ser feitas com o classificador GMM. Torna-se então possível verificar se o comportamento do sistema de classificação tem um bom desempenho e se os acertos na identificação das viaturas, utilizando as assinaturas em base de dados, estão a ser feitos numa quantidade razoável.

2.2 Rede de sensores sem fios

No geral, as redes de sensores sem fios são compostas por nós que contêm sensores com a função de recolher dados a partir da envolvente física. Estes dados são posteriormente processados e encaminhados para outros nós ou para uma estação base, como, por exemplo, através de comunicação rádio, dependendo do cenário em causa.

2.2.1 Arquitetura de Nós

Um nó pode ser de uso geral ou destinado a uma aplicação concreta. Os nós de uso geral devem ser versáteis, de forma a poderem adaptar-se a uma variedade de situações (Chris Townsend, 2004). Na aplicação em estudo, o nó deve conter, no mínimo:

- GPS - A utilização do recetor GPS permite identificar a posição, que pode ser utilizada num algoritmo de rastreio do movimento;
- Microfone - Permite a recolha de trechos de som;
- Emissor/Recetor - Enviar e/ou receber pacotes;
- Processador - Utilizado neste estudo, para efetuar o processamento de dados nos nós, antes de estes serem enviados para um outro;
- Random Access Memory (RAM) - Memória de acesso rápido que auxilia o processamento;
- Memória de armazenamento - Destinada a guardar as bases de dados das viaturas e outros dados relevantes;
- Bateria - Utilizada para armazenar energia e alimentar os componentes do nó;
- Módulo de recolha de energia (opcional) - Utilizado para recolher energia do meio ambiente, para aumentar a autonomia dos nós e, como consequência, a da rede na sua totalidade.

O nó mencionado encontra-se representado na Figura 2.4.

A energia consumida pelos nós assume uma importância relevante, uma vez que se pretende minimizar o tempo despendido na manutenção destes, para troca ou carregamento de baterias. Esta manutenção pode ser dificultada devido a constrangimentos táticos que, num caso extremo, podem inviabilizar a sua realização. Por isso, deve existir no nó um módulo que permita adquirir energia a partir do meio envolvente, como, por exemplo, através do sol - energia solar, que é a forma mais prática e a que mais se adequa ao cenário em estudo.

Para que o consumo energético seja reduzido, o processador tem de executar, com grande eficiência, determinadas tarefas:

- Gestão de dados provenientes dos sensores;
- Funções de controlo de energia;
- Interface da camada física do transmissor rádio com os dados provenientes dos sensores e gestão do protocolo da rede rádio.

O envio de dados através do transmissor rádio é a operação que consome mais energia, pelo que o seu envio terá lugar apenas em situações estritamente necessárias (Chris Townsend, 2004).

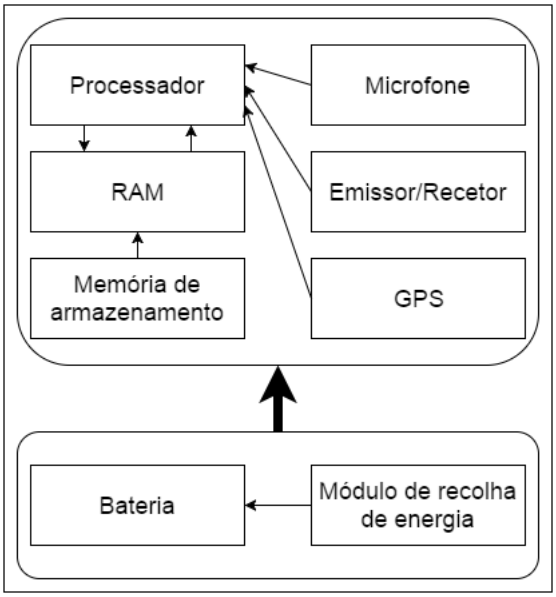


Figura 2.4: Nó tipo da rede de sensores sem fios.

2.2.2 Arquitetura de rede de sensores sem fios

As arquiteturas das redes de sensores podem ser agrupadas em três grandes grupos (Jamal N. Al-Karaki, 2004):

- Plana;
- Hierárquica;
- Baseada em localização.

Nas redes planas, todos os nós têm o mesmo papel, enquanto que nas redes hierárquicas, os protocolos têm em vista a disposição dos nós em *clusters* ou níveis, para que os nós "pai" de cada *cluster* efetuem alguma agregação e consigam reduzir os dados a transmitir, de forma a poupar energia. Os protocolos baseados em localização utilizam a posição do nó para fazerem chegar os dados aos nós pretendidos, que se encontram em determinadas regiões (Jamal N. Al-Karaki, 2004).

Existem várias topologias de rede que podem ser utilizadas nas redes de sensores sem fios em ambientes militares. Estas topologias são escolhidas conforme a aplicação pretendida.

Para a aplicação em estudo pretende-se que todos os nós tenham capacidade de efetuar agregação de dados, logo todos serão iguais. Mesmo sendo eles iguais, há a possibilidade de terem papéis diferentes, o que pode definir a rede como sendo hierárquica.

Pretende-se que durante algum tempo, alguns nós esperem por pacotes que outros enviam após recolherem som, agreguem esses dados e encaminhem um pacote com dados provenientes de vários nós, em vez de encaminharem todos os pacotes que recebem. Desta forma, qualquer nó pode receber dados de vários nós, desde que esteja contido na rota para a estação base. Esse nó, que está na rota, é automaticamente o nó pai de um determinado *cluster* de nós folha, que têm o objetivo de enviar o seu pacote para a estação base. Se as rotas mudarem, o nó pai de cada *cluster* muda automaticamente.

Com esta metodologia, que prevê a existência de apenas uma estação base, constituída por um nó diferente de todos os outros, a rede forma automaticamente uma estrutura hierárquica, dado que são eleitos nós "pai" (por pertencerem à rota para a estação base) que lhes vai permitir receber pacotes de outros nós, agregar e enviar para a estação base. Estes nós "pai" vão ser iguais a todos os outros nós, com exceção da estação base, mas vão ter papéis diferentes, o que também induz na rede uma certa hierarquia.

Em suma, a rede que se pretende desenvolver vai ser idêntica a uma árvore, onde um determinado conjunto de nós envia para um nó "pai" que, posteriormente, pode enviar para um outro nó "pai" em conjunto com outros nós, e assim sucessivamente, até os dados chegarem à estação base. Em todos estes nós "pai" poderá ser realizada agregação. A estrutura em árvore vai ser abordada com mais detalhe na Secção 2.2.3.

2.2.3 Agregação de dados

Dado que estas redes integram fontes de energia e largura de banda limitadas, pode ser necessário efetuar processamento em nós intermédios e só alguns transmitirem os dados para a estação base. Uma vez que o objetivo é minimizar as transmissões de dados ao máximo, um nó recolhe dados de vários e efetua algum processamento, eliminando algumas redundâncias. Posteriormente, os dados são enviados para a estação base ou para um outro nó. Esta metodologia é conhecida como agregação de dados (Spandan et al., 2013).

Se não existir agregação, um nó recebe pacotes de vários nós e simplesmente encaminha esses pacotes. Se forem recebidos dez pacotes num nó, este tem de encaminhar dez pacotes distintos, efetuando dez transmissões e gastando uma certa energia. Se esses dados forem todos iguais, o nó pode conter um tempo de agregação, no qual são recebidos, por exemplo, esses mesmos dez pacotes, aos quais é eliminada a redundância, sendo enviado com o mesmo destino apenas um pacote. Desta forma, apenas se gasta energia numa transmissão. Isto pode ser feito de forma simples porque, no caso em estudo, todos os nós têm como destino a estação base. Em outros casos seria necessário verificar o destino dos vários pacotes e realizar a agregação com essa condicionante.

É vantajosa a utilização de agregação de dados no caso em estudo, mesmo sendo necessário processamento adicional nos nós intermédios. Isto, porque a energia gasta com o processamento adicional é muito menor do que a gasta com um número elevado de transmissões. Assim, dado que as amostras de som recolhidas requerem algum processamento para correta identificação da assinatura, os nós que recolhem essas amostras podem efetuar o processamento e transmitir apenas os dados referidos na Secção 2.1.3.

Em muitas redes de sensores os nós existem em grande quantidade. No caso em estudo estabelece-se também como objetivo a utilização de uma grande quantidade de nós, o que torna necessária a agregação de dados (Spandan et al., 2013).

Os mecanismos de agregação de dados podem ser maioritariamente classificados em (Spandan et al., 2013):

- **Sem estrutura** - O mecanismo sem estrutura efetua agregação de dados dinâmica sem ter de gastar recursos a construir uma estrutura. Neste tipo de mecanismo é normalmente utilizada a arquitetura de cliente/servidor para comunicação entre nós da rede. Quando a rede contém muitos nós, tem de se considerar o número de ligações que vão ser necessárias entre estes e o custo, em termos de energia, que estas ligações vão implicar. Assim, em grandes redes, esta arquitetura não é muito aconselhável (Spandan et al., 2013).

- **Baseadas em estrutura** - As agregações baseadas em estrutura ou hierárquicas (Secção 2.2.2) são definidas, como referido, através de um conjunto de algoritmos que divide a rede em *clusters* ou níveis. Estes *clusters* ou níveis, gerem de forma autónoma os dados e a sua agregação. Como consequência, têm uma visão reduzida da rede. Este mecanismo tem a desvantagem de exigir um esforço extra para organizar a rede, quando instalada, e manter esta organização durante o seu funcionamento.

  O mecanismo de agregação baseado em estrutura pode ser classificado em (Spandan et al., 2013):

  – **Baseado em árvore** - Nesta estrutura, os nós são organizados em árvore. A agregação de dados é feita em nós intermédios até chegar à estação base, ou seja, cada nó tem um "pai" para onde envia os dados, onde é feita a agregação dos mesmos. Os dados começam nas folhas - nós mais a baixo na árvore, e fluem até à estação base (Spandan et al., 2013). O mecanismo baseado em árvore, encontra-se representado na Figura 2.5.

  – **Baseado em *clusters*** - Nesta estrutura a rede é dividida em vários *clusters*. Em cada *cluster* existe um nó definido como *master* do *cluster*. Estes *masters* enviam os dados dos nós do seu *cluster* para a estação base. Os restantes nós do *cluster* são considerados *slaves*. Este mecanismo leva a um controlo rigoroso do tráfego de dados, uma vez que os *slaves* não podem comunicar entre eles e nunca existe comunicação entre *clusters* (Spandan et al., 2013). O mecanismo baseado em *clusters*, encontra-se representado na Figura 2.6.

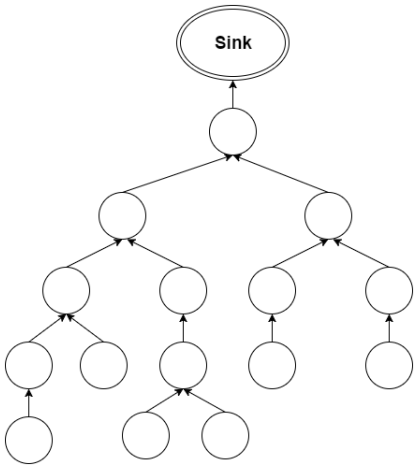


Figura 2.5: Mecanismo baseado em árvore.

Na estrutura baseada em *clusters*, a transmissão de dados é menor, pois podem ser criados *clusters* maiores. Ainda assim, existe a necessidade de a rede conter nós com capacidade de

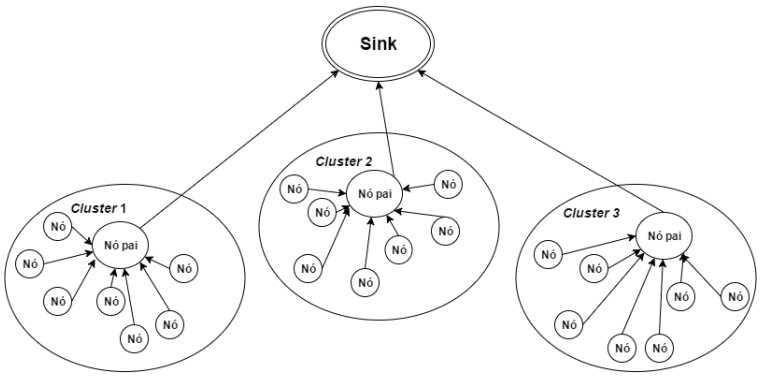


Figura 2.6: Mecanismo baseado em *clusters*.

processamento maior, de forma a que os *masters* consigam agregar mais dados. Tipicamente, neste tipo de estrutura os *masters* estão todos ao alcance da estação base, o que é uma limitação dado que se pretende implementar multi-salto mesmo pelos nós que agregam.

Na estrutura em árvore ocorrerão mais transmissões, o que implica um maior consumo energético, mas, por outro lado, pode não ser necessária a existência de nós diferentes com capacidade de processamento maior, o que é o caso da aplicação que se pretende desenvolver. Outra valência desta estrutura é o facto de os nós "pai" poderem realizar multi-salto e com isto encaminhar os dados através de outros nós, até à estação base, evitando assim a necessidade de estarem ao alcance desta.

Como referido, todos os nós vão ter a mesma capacidade de processamento e os *clusters* vão ser criados conforme as tabelas de encaminhamento para a estação base, ou seja, se um nó for utilizado para encaminhar pacotes de outros nós, este vai efetuar agregação durante um tempo e, posteriormente, vai enviar o pacote com destino à estação base, que por si pode ser novamente intercetado por outro nó que vá encaminhar o pacote. A aplicação a desenvolver pode, então, ser classificada como uma estrutura em árvore.

2.2.4 Deteção e rastreio do movimento de alvos

Um dos objetivos da aplicação em estudo, como referido anteriormente, é o rastreio de veículos em âmbito operacional. Para tal, pretende-se, numa primeira fase, identificar o tipo de viatura e classificá-la.

Num ambiente operacional podem existir, simultaneamente, diversas viaturas. Não sendo o rastreio de uma só viatura considerado trivial, realizar do rastreio de várias viaturas e sua classificação resulta numa operação ainda mais complexa. Imaginando-se duas viaturas em áreas A e B, se uma viatura se deslocar para uma outra área C, pode não ser possível identificar qual delas se deslocou, visto serem do mesmo tipo (Karl and Wilig, 2005).

2.2.4.1 Requisitos da Rede

Para dimensionar uma rede de sensores dedicada a rastreio de viaturas, é necessário ter em consideração alguns requisitos no que concerne à rede em si e ao processamento de sinal nos nós (Karl

and Wilig, 2005):

- **Deteção** - O nó é capaz de detetar a viatura através do processamento das amostras de som captadas pelo microfone. Estas são posteriormente comparadas com a base de dados, tornando possível identificar, ou não, a viatura.
- **Localização da viatura** - Para localizar a viatura é necessário existirem leituras de vários sensores. Estas leituras são combinadas para gerar uma posição da viatura. São necessárias, no mínimo, amostras de três nós, caso seja conhecida a intensidade, e de quatro, se esta não for conhecida.
- **Classificação** - Como referido no ponto anterior, é possível identificar o tipo de viatura através da análise do espetro em frequência das amostras de som recolhidas pelo microfone. Esta análise permite verificar se é efetivamente uma viatura, e qual o seu tipo (se esta estiver inserida na base de dados).

  É possível efetuar estas classificações, como já foi referido, com processamento nos nós que recolhem as amostras, ou é possível enviar as amostras para outros nós ou para a estação base, onde é efetuado o processamento destas e identificação das viaturas. As vantagens e desvantagens destas duas abordagens encontram-se na Tabela 2.3. Para a aplicação em estudo, é mais adequado efetuar algum processamento nos nós folha, pois:

  - Facilita a aquisição e o lançamento dos nós, se estes forem todos iguais;
  - Aumenta a robustez, visto os nós terem todos o mesmo papel, não existindo, assim, nós com maior importância;
  - A monitorização em tempo real torna-se mais fiável se existir menos latência na rede;
  - Se o processamento for feito nos nós folha, as transmissões são apenas os dados referidos na Secção 2.1.3. Assim, é possível reduzir a largura de banda necessária e utilizar um transcetor de menor frequência.

  É importante referir que o rastreio de movimento não foi realizado nesta aplicação, mas é uma componente que pode ser desenvolvida em trabalhos futuros e que traz muitas vantagens e utilidades de uma rede deste tipo em campo de batalha, em ações ofensivas e defensivas. Permite saber onde circula o inimigo, de forma passiva, através da captura de som pelos nós.

Tabela 2.3: Vantagens e desvantagens de processamento em nós folha ou em nós específicos (Karl and Wilig, 2005).

Processamento			
Nós Folha		**Nós Específicos**	
Vantagens	**Desvantagens**	**Vantagens**	**Desvantagens**
Necessária menos largura de banda	Perda de dados	Mais informação no nó	Necessária maior largura de banda
Nós iguais em toda a rede			Nós com maior capacidade de processamento
Menos latência na rede			Maior latência na rede

2.2.5 Tecnologias de Comunicação em Redes de Sensores Sem Fios

Nos dias de hoje, existem diversas tecnologias de comunicação e é feita uma comparação entre as que mais se adequam às redes de sensores sem fios por Lee et al. (2007). O *Bluetooth* que utiliza *IEEE 802.15.1*, o *ultra-wideband* que utiliza *IEEE 802.15.3*, o *ZigBee* que utiliza *IEEE 802.15.4* e o *Wi-Fi* que utiliza *IEEE 802.11* são quatro protocolos utilizados para comunicações sem fios de curto alcance (Lee et al., 2007).

O *ultra-wideband* destina-se a redes interiores e com distâncias de 10m entre nós, tornando-se pouco recomendável à rede que está a ser desenvolvida, pois esta tem o objetivo de ser lançada em campos de batalha e de conseguir abranger a maior área possível com um baixo número de nós (Lee et al., 2007).

O *Bluetooth* destina-se a comunicações de muito curto alcance, na ordem dos 10m, mais concretamente destinado a ligar dispositivos de redes pessoais. Foi desenvolvido para substituir os cabos dos periféricos de um dispositivo, como por exemplo, teclados ou ratos para um computador (Lee et al., 2007). Esta tecnologia só permite a criação de *clusters* até oito nós, o que é uma limitação quando se pretende desenvolver multi-salto com um elevado número de nós que necessite de abranger grandes áreas.

As tecnologias *IEEE 802.15.4* e *IEEE 802.11* são as que mais se adequam à implementação pretendida e vão ser descritas nas secções seguintes. A Tabela 2.4 sumariza as principais diferenças entre os protocolos *IEEE 802.15.4* e *IEEE 802.11g*

Tabela 2.4: Comparação entre *IEEE 802.15.4* e *IEEE 802.11g* (Lee et al., 2007).

	IEEE 802.15.4	***IEEE 802.11g***
Banda de frequência	868/915 MHz; 2.4 GHz	2.4 GHz
Taxa de transferência máxima	250 Kb/s	6 Mb/s
Distancia típica entre nós	10 - 100 m	100-140 m
Potência de transferência	(-25) - 0 dBm	15 - 20 dBm
Número de canais	1/10; 16	14
Largura de banda do canal	0.3/0.6 MHz; 2 MHz	20 MHz
Tipo de modulação	BPSK (+ ASK), O-QPSK	BPSK, CCK, OFDM
Espalhamento	DSSS	DSSS, CCK, OFDM
Célula básica	Estrela	BSS
Extensão da célula básica	Árvore de *clusters* ou Malha	ESS
Número máximo de nós por célula	>65000	2007
Encriptação	Cifra de bloco AES	Cifra RC4
Autenticação	CBC-MAC	WPA2
Proteção de dados	16-bit CRC	32-bit CRC
Tempo por bit (μs)	4	0.16667
Tamanho máximo de dados por pacote (bytes)	102	2312
Tamanho máximo para cabeçalhos (bytes)	31	58
Eficiência de código máxima (%)	76.52	97.18

2.2.5.1 Tecnologia *IEEE 802.15.4*

O protocolo *IEEE 802.15.4* foi desenvolvido para redes locais sem fios com baixo ritmo de transmissão, *Low-Rate Wireless Personal Area Network* (LR-WPAN) e é, tipicamente, usado para redes onde os nós se podem distanciar até 100 metros. Com este protocolo as redes de sensores suportam nós com funções mais complexas e nós mais simples, que apenas recolhem quantidades reduzidas de dados (Lee et al., 2007). No caso em estudo pretende-se que os nós tenham todos as mesmas características,

pelo que esta vantagem não vai ser explorada. Esta tecnologia está também associada a consumos energéticos muito baixos, o que leva a que autonomia da rede seja maior (Lee et al., 2007).

2.2.5.2 Tecnologia *IEEE 802.11*

O protocolo *IEEE 802.11* foi desenvolvido para redes locais sem fios, mas não necessariamente com baixo ritmo de transmissão. Permite que os utilizadores naveguem na Internet a velocidades altas, mesmo com bastante mobilidade por parte dos utilizadores, e oferece versatilidade no que respeita à entrada e saída de nós da rede (Lee et al., 2007). Permite ainda grandes distâncias entre nós para bandas de frequência mais baixas mas tem um elevado consumo de energia. Neste trabalho pode aplicar-se esta tecnologia, sendo possível que os sensores sejam relativamente mais caros, dada a natureza militar da aplicação, e podem beneficiar de uma bateria com maior capacidade. Pretende-se, também, utilizar nesta aplicação um mecanismo de agregação de dados que pode reduzir, em parte, o elevado consumo energético desta tecnologia.

Existem várias versões da tecnologia *IEEE 802.11* mas as mais recentes como a "g", "n" e "ac" permitem uma maior compatibilidade com os dispositivos de hoje em dia e têm maiores alcances e robustez nas comunicações. Foi preferível a utilização da versão "g", pois é compatível com a versão "n", ao contrário da versão "ac", que funciona a 5GHz.

2.2.5.3 Análise comparativa do *IEEE 802.15.4* e *IEEE 802.11g*

Analisando a Tabela 2.4, verifica-se que o protocolo *IEEE 802.11g* tem as seguintes vantagens em relação ao protocolo *IEEE 802.15.4* (Lee et al., 2007):

- Maior taxa de transferência de dados;
- Maior alcance, logo uma maior distância possível;
- Maior quantidade de dados por pacote.

O protocolo *IEEE 802.15.4* tem as seguintes vantagens em relação ao protocolo *IEEE 802.11g* (Lee et al., 2007):

- Maior número de canais (mas menor largura de banda por canal);
- Maior número de nós por célula básica.

No que se refere ao tempo de transferência de dados, este depende da taxa de transmissão de dados, do tamanho da mensagem e da distância entre nós. Este tempo pode ser calculado através da Equação 2.15 (Lee et al., 2007):

$$T_{tx} = (N_{data} + (N_{data}/N_{maxPld} * N_{ovhd})) * T_{bit} + T_{prop} \tag{2.15}$$

Onde N_{data} corresponde ao tamanho dos dados, N_{maxPld} ao tamanho máximo de dados por pacote, N_{ovhd} ao tamanho dos cabeçalhos, T_{bit} ao tempo por bit e T_{prop} ao tempo de propagação. Este último despreza-se, com objetivo de simplificar os cálculos (Lee et al., 2007).

Utilizando um *script* de *Matlab*, que inclui a equação 2.15, foi possível gerar a Figura 2.7. Esta mostra que o protocolo *IEEE 802.11g* tem sempre um tempo de transmissão menor que o protocolo *IEEE 802.15.4* para o mesmo tamanho de pacote.

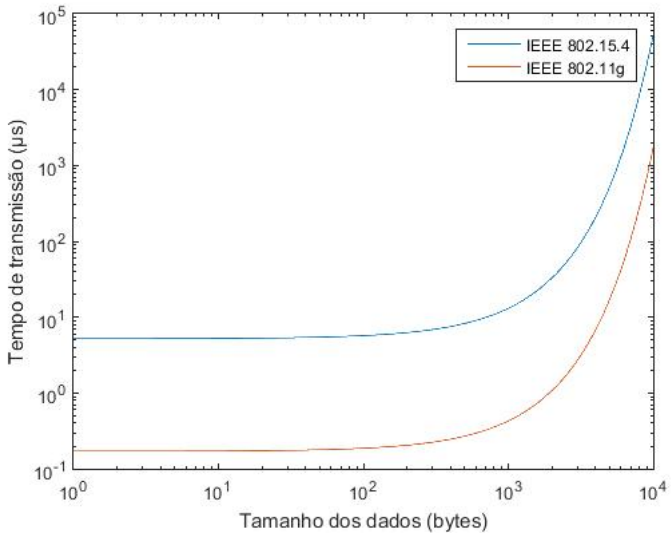


Figura 2.7: Comparação do tempo de transmissão dos protocolos *IEEE 802.11g* e *IEEE 802.15.4* dado o tamanho de dados a transmitir.

Outro aspeto que está relacionado com a transmissão de dados é a eficiência de código, que consiste na relação entre o número total de bytes usados para transmitir e os bytes da mensagem propriamente dita. Segundo Lee et al. (2007), o protocolo *IEEE 802.15.4* tem uma maior eficiência de código para tamanhos inferiores a 104 bytes, enquanto que o protocolo *IEEE 802.11g* tem maior eficiência para tamanhos superiores a este. Isto deve-se ao facto de o tamanho máximo de dados por pacote, suportado pelo protocolo *IEEE 802.15.4*, ser 104 bytes, o que torna necessária a fragmentação para tamanhos superiores a este.

Quando se pretende escolher a melhor tecnologia para uma rede de sensores sem fios que vai depender de uma fonte de energia limitada, é necessário estudar o desempenho desta a nível energético. Lee et al. (2007) refere que o protocolo *IEEE 802.15.4* é destinado a nós com comunicação de curto alcance e limitados energicamente, o que permite um baixo consumo de energia. Por outro lado, o protocolo *IEEE 802.11g* foi projetado para comunicações de maior alcance e para nós que tenham uma fonte de energia superior, pois o consumo energético pode ser substancialmente mais elevado (Lee et al., 2007). Para ser feita uma análise comparativa do consumo de energia entre estes dois protocolos, recorreu-se a dois módulos rádio que existem no mercado, CC2430 da *Texas Instruments* e VNT9485BE0CW da *VIA Embedded* (Embedded, 2013), associados, respetivamente, aos protocolos *IEEE 802.15.4* e *IEEE 802.11g*. Na Tabela 3.1 encontram-se representados os consumos de envio e receção destes módulos. Com estes dados foi possível gerar a Figura 2.8, que representa a potência consumida pelos diferentes protocolos. É visível que o protocolo *IEEE 802.15.4* consome menos potência quando comparado com o protocolo *IEEE 802.11g*. Caso se inclua a taxa de bit no cálculo da potência (normalização da energia), representada na Figura 2.9, verifica-se que o protocolo *IEEE 802.11g* tem

uma maior eficiência (Lee et al., 2007).

Através da análise dos dados, é possível concluir que o protocolo *IEEE 802.15.4* é mais indicado para aplicações em que a taxa de transferência de dados é mais baixa e as fontes de energia são limitadas, já que o consumo de energia para quantidades de dados reduzidas é mais baixo. Em relação ao protocolo *IEEE 802.11g*, este torna-se mais indicado para aplicações em que a taxa de transferência de dados seja mais alta e as fontes de energia não sejam demasiado limitadas (Lee et al., 2007).

Em suma, o protocolo *IEEE 802.15.4* será o mais indicado para a aplicação em estudo, dado que a taxa de transferência de dados não será muito elevada, uma vez que não se pretende ultrapassar os 104 bytes por pacote, e o objetivo é conseguir a maior autonomia possível.

Na secção 4.2 apresentar-se-á uma comparação entre os desempenhos dos dois protocolos na rede de sensores.

Tabela 2.5: Consumos dos módulos rádio para cada protocolo (Lee et al., 2007).

Protocolos	*IEEE 802.15.4*	*IEEE 802.11g*
Módulo rádio	CC2430	VNT9485BE0CW
VDD (volt)	3.0	3.3
Envio (mA)	27	520
Receção (mA)	27	250
***Standby* (mA)**	0.0003	8
Taxa de bit (Mb/s)	0.25	6

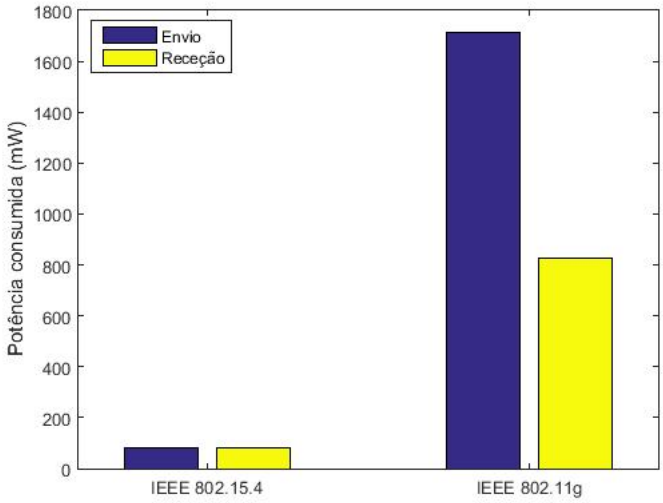


Figura 2.8: Comparação da potência consumida pelos protocolos *IEEE 802.11g* e *IEEE 802.15.4* (Lee et al., 2007).

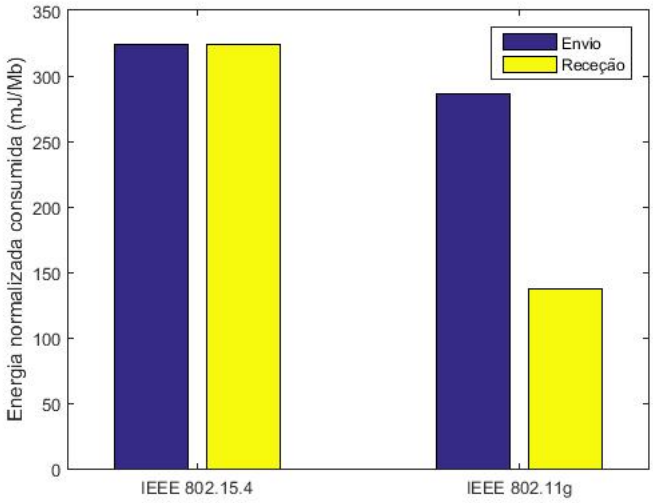


Figura 2.9: Comparação da energia normalizada que é consumida pelos protocolos *IEEE 802.11g* e *IEEE 802.15.4* (Lee et al., 2007).

2.2.6 Pilha de Protocolos IP

Vários autores defendem que o uso de IP em redes de sensores sem fios é inadequado, mas Hui and Culler (2008) e Kuorilehto et al. (2006) apresentam modelos IP para redes de sensores com baixo consumo de energia.

Segundo Kuorilehto et al. (2006), o modelo TCP/IP que deve ser utilizado para redes de sensores sem fios é o representado na figura 2.10. Este modelo é versátil e pode ser aplicado a muitas redes de sensores, mas Hui and Culler (2008) desenvolveram um modelo onde utilizam especificamente IPv6.

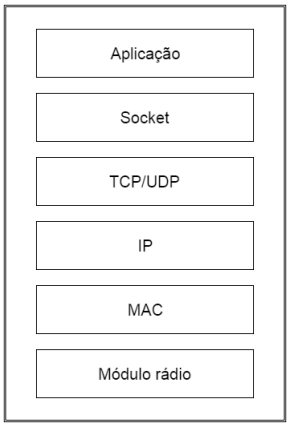


Figura 2.10: Modelo TCP/IP para redes de sensores sem fios (Kuorilehto et al., 2006).

2.2.6.1 Protocolos da Camada Rede

O IPv4 e IPv6 são duas versões do protocolo IP e estão incluídos na camada rede do modelo TCP/IP e na camada IP da Figura 2.10. Hui and Culler (2008) afirmam que ao longo dos anos as redes de sensores baseadas em IP têm vindo a evoluir, nomeadamente o IPv6 com a introdução da camada

de adaptação (6LoWPAN) para o protocolo *IEEE 802.15.4*. Devido a este facto, alguns mecanismos necessitaram de ser adaptados do IPv4 para o IPv6, o que levou a que muitos dispositivos não fossem compatíveis com esta nova implementação. Através dos resultados a que estes autores chegaram, foi possível constatar que a arquitetura baseada em IPv6 satisfaz os requisitos de uma rede de sensores com baixo consumo energético.

Cada vez mais existe uma tendência para a "*Internet of Things*", o que leva à existência de números elevados de nós com sensores que necessitam de comunicar entre eles. Atualmente, o IPv6 adequa-se mais a este tipo de sistemas do que o IPv4 e, por isso, deve ser uma prioridade no que toca ao desenvolvimento de novas aplicação em redes de sensores sem fios.

Na rede de sensores sem fios que foi desenvolvida neste trabalho, é utilizado o protocolo IPv4 com a tecnologia *IEEE 802.11g* e o IPv6 com a tecnologia *IEEE 802.15.4*. Apresenta-se uma análise de resultados na Secção 4.2.

2.2.6.2 Protocolos de encaminhamento

Como já foi referido anteriormente, as redes de sensores são compostas por um elevado número de nós. Estes, após receberem e/ou processarem os dados de interesse, têm de enviá-los para um outro nó ou para a estação base. Em muitos dos casos, os nós para os quais se pretende enviar os dados não estão ao alcance do nó que envia, pelo que se torna necessário encaminhar os dados através de outros nós. Isto acontece porque, a potência necessária para cobrir certas distâncias, de modo a todos conseguirem enviar para a estação base, seria demasiada em comparação com a capacidade que o nó tem.

Os nós dispõem de recursos energéticos limitados e capacidade de comunicação limitada. Visto isto, são necessários protocolos de encaminhamento que tenham em conta a energia disponível na rede, com a finalidade de aumentar o tempo de vida dos nós (Jamal N. Al-Karaki, 2004).

Os dados recolhidos pelos nós podem ser redundantes ou não adequados e, por isso, este fator pode ser explorado pelos algoritmos de encaminhamento, de modo a reduzir o consumo de energia nas transmissões. Na aplicação em estudo, pode ser aproveitado o facto de apenas ser necessário enviar dados, caso o som captado não seja ruído ambiente (Jamal N. Al-Karaki, 2004).

Com o avançar do tempo, foram propostos muitos algoritmos com o objetivo de colmatar problemas já existentes. Encontrar e manter as rotas por onde fluem os dados não é tarefa elementar, dado existirem restrições de energia e alterações do estado e conetividade dos nós (Jamal N. Al-Karaki, 2004).

Existem algumas técnicas, referidas nas secções anteriores, que podem ser contempladas nos algoritmos de modo a minimizar o consumo de energia (Jamal N. Al-Karaki, 2004):

- Agregação de dados (Secção 2.2.3);
- Processamento nos nós;
- *Clustering*;
- Atribuição dinâmica da função dos nós.

O desenvolvimento de protocolos de encaminhamento para redes de sensores sem fios é influenciado por diversos fatores, os quais devem ser tidos em conta de forma a tornar a comunicação entre os nós da rede o mais eficiente possível. Torna-se então relevante a abordagem de alguns desafios e problemas associados ao encaminhamento nas redes de sensores sem fios (Jamal N. Al-Karaki, 2004):

- **Implantação dos nós** - A forma como os nós estão dispostos no terreno influencia o desempenho do protocolo de encaminhamento. A implantação dos nós pode ser determinística ou ao acaso. Na forma determinística, os nós são colocados manualmente e os dados podem fluir segundo uma rota pré-definida. No caso de serem implantados ao acaso, os nós necessitam de estabelecer uma rede entre eles, tornando-se vantajosa a criação de *clusters*, uma vez que permitem uma maior conetividade e uma maior eficiência energética. Neste tipo de redes, as rotas dos dados incluem, quase sempre, vários saltos até ao destino.
- **Consumo de energia sem perda de precisão** - Como já foi referido, os nós podem conter uma fonte de energia limitada - por exemplo, as baterias, e, dado que estes têm de desempenhar função de receção e transmissão, de forma a que a rede suporte multi-salto, os algoritmos de encaminhamento têm de ser eficientes a nível energético, podendo utilizar um módulo que permita adquirir energia a partir da luz solar. É importante referir que, dada a ausência de luz solar durante a noite, é necessário que a bateria aguente o consumo energético do nó durante este período. Se um nó deixar de funcionar por falha de energia, a rede poderá ter de se reorganizar e, enviar os pacotes por outra rota.
- **Modelo de geração de dados** - A forma de tratar o envio de dados dos nós para os nós "pai", ou para a estação base, varia de acordo com a aplicação. Podem ocorrer várias situações:
  - **Envio periódico** - Os nós que contêm sensores, ativam-nos periodicamente, de forma a captar e processar os dados. Posteriormente, com os dados já processados, procede-se ao seu envio para a estação base ou nó "pai".
  - **Envio despoletado por eventos** - Os nós que contêm sensores, reagem imediatamente a mudanças drásticas no valor captado, e enviam os dados para a estação base ou nó "pai".
  - **Envio a pedido** - A estação base realiza o pedido, de modo a que, os nós recolham amostras e enviem os dados para si.
  - **Modo híbrido** - Pode existir uma combinação destes modelos.

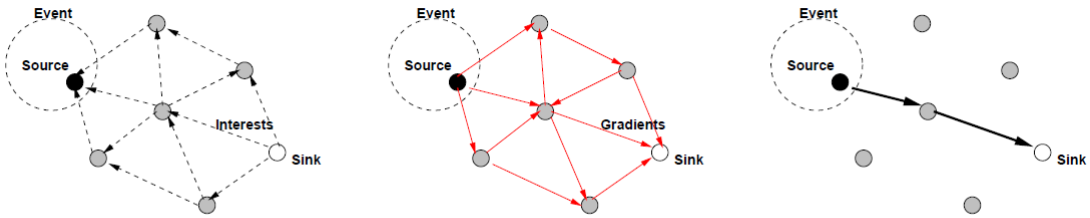
  No caso em estudo pretende-se que os nós se encontrem a dormir e, no caso de se detetar uma viatura, estes acordem, captem as amostras de som, efetuem o processamento necessário e enviem os dados para os nós "pai" que, de seguida, enviam para a estação base. Tendo em conta os modelos acima, pode afirmar-se que, o mais adequado para este estudo, é o despoletado por eventos.

Nesta implementação, pretende-se que os nós sejam todos iguais e os nós "pai" dos *clusters* tenham a mesma capacidade que os outros. Isto permite que outro nó "pai" possa ser eleito dinamicamente, sem problemas associados a heterogeneidade.

De forma a escolher a melhor possibilidade, vão ser estudados e comparados alguns algoritmos de encaminhamento já existentes. Podem identificar-se outros algoritmos em Jamal N. Al-Karaki (2004).

2.2.6.3 *Directed Difusion (DD)*

Este paradigma, ao qual podemos chamar *Directed Difusion*, é responsável pela centralização dos dados. Os dados gerados pelos sensores são compostos por pares atributo-valor. Um nó solicita dados através do envio de interesses para determinados dados. Desta forma, os dados que corresponderem ao interesse enviado, são encaminhados através dos gradientes - rotas definidas para tráfego de dados, para o nó que fez a requisição. De seguida, um ou mais desses caminhos são reforçados para o tráfego de dados. Este esquema encontra-se representado na Figura 2.11 (Intanagonwiwat et al., 2000).



(a) Interest propagation (b) Initial gradients set up (c) Data delivery along reinforced path

Figura 2.11: Esquema simplificado para *Directed Difusion* (Intanagonwiwat et al., 2000).

O DD pode adequar-se à aplicação em estudo dado que podem ser feitos pedidos de dados da estação base aos nós. Podem ser escolhidos os períodos de tempo e a frequência de envio de dados de toda a rede ou de determinados grupos de nós. Este protocolo tem a contradição de os nós não utilizarem endereços, pois, tipicamente, nestas redes existem nós em grande quantidade. Na aplicação em estudo seria um problema, uma vez que se pretende o envio do identificador de cada nó, de forma a identificar os nós que recolheram informação.

2.2.6.4 *Low Energy Adaptive Clustering Hierarchy (LEACH)*

O LEACH é um protocolo baseado em *clusters*, cuja formação destes é feita de forma distribuída. Inicialmente este escolhe, de forma aleatória, nós "pai" dentro de cada *cluster*. Posteriormente, vai elegendo outros nós "pai", de forma a distribuir a carga energética por todos os sensores da rede (Jamal N. Al-Karaki, 2004).

Os nós "pai", em cada *cluster*, recebem dados dos outros nós pertencentes ao *cluster* e efetuam agregação de dados, que enviam para a estação base. Esta metodologia tem o objetivo de reduzir a quantidade de transmissões para a estação base, o que, consequentemente, reduz a energia gasta (Jamal N. Al-Karaki, 2004).

Este protocolo é apropriado e pode ser utilizado em casos em que é necessária uma monitorização constante. O utilizador pode não precisar de dados periodicamente e, nesse caso, é possível poupar energia nos envios de dados periódicos, procedendo ao envio apenas quando os nós recolhem dados (Jamal N. Al-Karaki, 2004).

O LEACH pode adequar-se à rede a desenvolver, apresentando como única contraindicação o facto de que os nós "pai" poderiam ter de efetuar transmissões para outros nós "pai" que estivessem a longas distâncias.

2.2.6.5 *Destination-Sequenced Distance-Vector (DSDV)*

Este protocolo insere-se na classe de protocolos de encaminhamento com vetor distância. Isto é, cada nó contém uma tabela com o custo e distância dos nós vizinhos. Este custo está relacionado com a energia que vai ser a gasta ao enviar os dados por aquele nó vizinho. A tabela dá a menor distância para cada destino e a rota pela qual os dados vão ser enviados. Nas redes em que os nós enviam dados só para um nó "pai", ou para uma estação base, basta construir a rota para estes, simplificando e reduzindo o desperdício.

Para esta tabela estar sempre atualizada, podem existir pedidos de informação periódicos aos nós vizinhos. Se para um dos nós a distância mínima for alterada, o processo é repetido até todos os nós ficarem com a sua tabela atualizada (He, 2002).

Como se verifica, torna-se então necessária uma elevada troca de dados na rede e, consequentemente, um elevado gasto de energia em transmissões. Para isto, o protocolo DSDV introduz um novo atributo às tabelas de encaminhamento: um número de sequência. Ao adicionar este número, os nós conseguem distinguir a informação obsoleta encaminhada anteriormente, daquela que é a nova informação, evitando-se, assim, os ciclos de encaminhamento constantes de atualização de tabelas (He, 2002).

Pode ser feita uma modificação a este protocolo, que se torna muito vantajosa na aplicação em estudo. Esta modificação passa pela limitação das rotas existentes nas tabelas, ou seja, dado que só existe um destino (estação base), é possível manter apenas, em todas as tabelas da rede, a rota para a estação base. Esta modificação leva a uma diminuição na sobrecarga de uma rede com centenas de nós, na altura da atualização das tabelas. A estação base envia as atualizações e os nós apenas propagam esta atualização, não incluindo a rota para si. Esta modificação é explicada de uma forma mais pormenorizada na Secção 3.2.1.6 e 3.2.2.3.

Tipicamente, este protocolo está associado a uma estrutura plana, onde todos os nós têm as mesmas funções, mas, dada a modificação introduzida para limitar a propagação da rota à estação base, automaticamente este forma uma estrutura em árvore, como mencionado na Secção 2.2.3. Dada a hierarquia da rede, torna-se então possível que os nós "pai", para onde os outros nós enviam informação, efetuem agregação de dados. Não esquecendo que, como todos os nós são idênticos, todos têm a capacidade de agregar dados e, caso algum fique inoperacional, uma nova estrutura é criada, gerando, deste modo, uma maior robustez na rede.

Este protocolo, com a alteração indicada, adequa-se à rede a desenvolver, já que permite a criação de uma rota consistente para a estação base, dado que existem atualizações periódicas das rotas. Permite, também, implementar agregação em todos os nós, pois não existem nós com tarefas especificas.

2.2.6.6 *Routing Protocol for Low-Power and Lossy Networks (RPL)*

O RPL é um protocolo que faz uso do IPv6 e que específica como criar um *Destination Oriented Directed Acyclic Graph* (DODAG). Esta DODAG é criada usando uma função objetivo, e um conjunto de métricas e restrições. A função objetivo opera com um conjunto de métricas e restrições, de forma a calcular o melhor caminho (Vasseur et al., 2011).

Podem existir diferentes funções objetivos em operação e no mesmo nó. Isto porque, dependendo da colocação dos nós e da aplicação em concreto, pode ser necessário encaminhar o tráfego por diferentes caminhos. Desta forma, podem existir diferentes DODAG para situações diferentes, como por exemplo, encontrar o caminho com menor latência – métrica, e evitar nós alimentados a bateria - restrição. A função objetivo pode não especificar as métricas e as restrições, mas define regras para criar a DODAG, número de "pais", "pai" de reserva, uso de balanceamento de carga, entre outros. (Vasseur et al., 2011).

O grafo criado pelo RPL consiste numa topologia de encaminhamento lógica, criada sobre uma rede física, de forma a obedecer a determinados critérios. É possível que existam diferentes topologias de encaminhamento – grafos, ativas ao mesmo tempo, que conduzem tráfego de acordo com diferentes requisitos (Vasseur et al., 2011)

Esta topologia permite que os nós se possam juntar a um ou mais grafos, marcando o tráfego conforme as características destes. Cada grafo fornece uma determinada qualidade de serviço, e baseia-se em métricas e restrições (Vasseur et al., 2011).

Este protocolo também se adequa à aplicação em estudo, pois permite implementar todas as funcionalidades desejadas. Importa referir que, após a modificações sugeridas para o protocolo DSDV, este adquire algumas semelhanças com o RPL, no que respeita à criação de uma estrutura, mas não possui rotas alternativas, o que é uma desvantagem.

2.2.6.7 Algoritmos Geográficos

Neste tipo de protocolos, as mensagens enviadas na rede têm em conta a localização dos nós. É possível determinar a sua localização através da intensidade do sinal dos nós vizinhos - localização baseada em radiofrequência, trocando as coordenadas através de comunicação com os nós vizinhos. Em alternativa, os nós podem ser equipados com recetores GPS (*Global Positioning System*) e, ao receberem informação dos satélites, determinam e guardam a sua localização. Existem muitos algoritmos geográficos, mas o estudado debruçar-se-á apenas sobre o *Geographic and Energy Aware Routing* (GEAR) (Jamal N. Al-Karaki, 2004).

Este protocolo tira partido da localização dos nós para realizar pedidos de dados a zonas específicas da rede e baseia-se no algoritmo de DD, ou seja, envia pedidos de dados específicos para zonas específicas da rede, mas não em *broadcast*. Assim, é possível poupar mais energia nas transmissões, tanto na realização de pedido aos nós, como nas transmissões de todos os nós para a estação base (Jamal N. Al-Karaki, 2004).

Neste protocolo, os nós obtêm informação relativa ao custo de envio de dados para o destino, através dos seus nós vizinhos. Este custo está relacionado com a distância e com a energia residual do nó

(Jamal N. Al-Karaki, 2004).

Este tipo de protocolos não se adequa muito à rede a desenvolver, pois pretende-se monitorizar uma área no global, não áreas especificas cobertas por grupos de nós.

2.2.6.8 Análise comparativa dos algoritmos

Tabela 2.6: Características dos protocolos de encaminhamento.

	DD	LEACH	DSDV	RPL	GEAR
Clustering		X			
Uso da Localização					X
Pedidos de dados	X				X
Multi-caminho	X			X	
Agregação de dados	X	X		X	
Eliminação de redundancia	X	X		X	
Menor caminho			X	X	

A Tabela 2.6 representa as características dos protocolos mencionados nos pontos anteriores (Winkler and Klaus-Dieter Tuchs, 2008).

Analisando os vários protocolos de encaminhamento e as suas características, podem ser discutidas diversas modalidades. Considera-se uma abordagem simples e eficaz a utilização de um protocolo modificado, de forma a adaptar-se à aplicação em estudo.

Ao utilizar o protocolo DSDV consegue obter-se uma eficiência elevada, dado que é pretendido que os nós da rede sejam estáticos e só exista uma estação base. Ou seja, os nós só precisam de conter a rota para a estação base nas suas tabelas e, desta forma, os pedidos de dados para atualizar as tabelas sobrecarregam muito menos a rede. Sendo os nós estáticos, os pedidos de dados podem ter um período muito grande, ou, no caso extremo, as tabelas podem ser preenchidas quando a rede for lançada e não ser necessário efetuar mais pedidos de dados para as tabelas.

O protocolo DSDV em si não inclui agregação de dados e eliminação de redundância, mas é desejável e possível implementar estas características ao nível da aplicação. Os nós enviam os dados com o destino estação base e se o pacote necessitar de ser encaminhado, caso não esteja ao alcance da estação base, o nó que vai encaminhar o pacote aguarda durante algum tempo a receção de mais pacotes e, caso receba mais que um, efetua a agregação com eliminação de eventuais redundâncias e envia o pacote novamente com destino à estação base.

O protocolo RPL é, dos protocolos referidos, o mais desenvolvido e mais utilizado atualmente. Permite escolher diferentes métricas para que o encaminhamento possa ser mais eficiente, conforme as restrições encontradas, como por exemplo, métrica de "caminho mais curto para a estação base", ou métrica "caminho que garante melhor qualidade de serviço".

Dada a aplicação em estudo, existe inicialmente um DODAG com nós "pai" bem definidos, por exemplo, para a métrica "caminho mais curto para a estação base". Assim sendo, quando alguns nós detetarem viaturas, é-lhes possível capturar, processar e enviar os dados para o nó "pai", que, por sua vez, reencaminha os dados através do caminho mais curto para a estação base. Uma adaptação que pode ser feita a este algoritmo passa por, no caso de existirem duas rotas com o mesmo número de saltos, ser escolhida aquela que tiver um nó "pai" que esteja também a receber dados, de forma a que a

agregação seja feita de uma forma mais eficiente. Um problema que pode surgir com esta aplicação é a possível existência de ciclos infinitos de transmissão de dados. Assim sendo, tem de garantir-se a inexistência destes ciclos, quando é desenvolvido o algoritmo.

Perante todos os protocolos referidos nesta secção, o simulador NS-3 (NS-3, 2016) só dispõe do protocolo DSDV, sendo que o protocolo RPL se encontra atualmente em desenvolvimento e poderá ser um opção em aplicações futuras. É importante referir que o DSDV com as alterações referidas no que se refere à propagação da rota, reveste-se de algumas semelhanças com o RPL no que concerne à criação de uma estrutura, mas não possui rotas alternativas, o que é uma desvantagem.

Em suma, o protocolo usado será o DSDV, em duas versões distintas, IPv4 e IPv6, com as alterações referidas nesta secção. A implementação e os resultados associados às simulações com este protocolo encontram-se descritos nas Secções 3.2 e 4.2 respetivamente.

Capítulo 3

Descrição do projeto

3.1 Caracterização e classificação de viaturas

Nesta Secção vai ser descrita toda a implementação relativa ao desenvolvimento da base de dados de sons de viaturas e respetiva classificação. Algum Código e Figuras relativas a esta Secção, encontram-se representados no Apêndice A.

3.1.1 Aquisição dos sinais de áudio

De forma a desenvolver o código para caracterização e classificação de viaturas militares através da sua assinatura espetral, foi adquirido som, com recurso a um gravador digital, equipado com dois microfones, Zoom H6 (Nozokido, 2013), de várias viaturas militares (Exército e Força Aérea):

- Panhard M11 (Figura 3.1(a)): Veículo blindado para reconhecimento. Som gravado na Escola das Armas, em Mafra;
- Mitsubishi 4x4 L200 (Figura 3.1(b)): Veículo de transporte de pessoal. Som gravado na Escola das Armas, em Mafra;
- Leopard-2A6 (Figura 3.1(c)): Carro de combate. Som gravado no Regimento de Cavalaria nº4, no Campo Militar de Santa Margarida;
- M60A3 (Figura 3.1(d)): Carro de combate. Som gravado no Regimento de Cavalaria nº4, no Campo Militar de Santa Margarida;
- M113 (Figura 3.1(e)): Veículo blindado de transporte de pessoal. Som gravado no Regimento de Cavalaria nº4, no Campo Militar de Santa Margarida;
- Condor (Figura 3.1(f)): Veículo blindado de transporte de pessoal. Som gravado no Campo de tiro da Força Aérea, em Alcochete;
- Protect Fire (Figura 3.1(g)): Carro dos bombeiros da força aérea. Som gravado no Campo de tiro da Força Aérea, em Alcochete;

- Unimog 1300 L (Figura 3.1(h)): Viatura tática. Som gravado no Regimento de Cavalaria nº4, no Campo Militar de Santa Margarida.

Foram adquiridas amostras de cada uma das viaturas em ambiente estrada e terra batida, a diferentes velocidades. As viaturas que foram utilizadas para as gravações encontram-se representadas na Figura 3.1 e as condições do meio envolvente, em cada zona, encontram-se representadas na Tabela 3.1.

(a) Panhard M11.

(b) L200.

(c) Leopard-2A6.

(d) M60A3.

(e) M113.

(f) Condor.

(g) Protect Fire.

(h) Unimog 1300 L.

Figura 3.1: Viaturas utilizadas para as gravações.

Tabela 3.1: Condições do meio envolvente em cada zona

	Mafra	Campo Militar de Santa Margarida	Campo de Tiro da Força Aérea
Passagem de carros em autoestrada	X		
Passagem de aviões	X		X
Fala humana	X	X	X
Sons de animais	X		X
Passagem de outros veículos militares		X	
Tiros de diferentes calibres em várias carreiras de tiro			X
Muito vento	X		
Pouco vento		X	X

3.1.2 Tratamento dos sinais de áudio

As gravações realizadas com o Zoom H6 continham dois canais a uma amostragem de 48000Hz. Para facilitar a análise do som no Matlab (Little and Moler, 2015), foi necessário convertê-los para um canal com 16000Hz de amostragem. Para este efeito, foi utilizado o software *Sox* (Norskog, 1991) numa máquina virtual com Ubuntu 16.04 (Ubuntu, 2015), com o Código 3.1 (Norskog, 1991):

Código 3.1: Comando Sox

```
1 sox InputSound.WAV -c 1 -r 16000 OutputSound.raw
```

Com os ficheiros já convertidos para o formato referido, foi utilizado o *software Wavesurfer* (Sjölander and Beskow, 2006), de forma a que os sons fossem fragmentados, permitindo retirar a maioria do ruído envolvente referido. Adicionalmente, com o *software Audacity* (Mazzoni and Dannenberg, 1999), o ganho dos sons de todas a viaturas foi normalizado.

Os fragmentos de som que foram retirados - aviões, sons de animais, tiros, fala humana e carros civis, foram armazenados à parte, para que, juntamente com sons de outras viaturas, formem um modelo de *garbage*, de forma a que, no caso de a amostra de som corresponder a algum destes, não fosse erradamente classificada como uma das viaturas que se encontram na base de dados. A quantidade de ficheiros de som e a duração total de som gravado para o *garbage*, bem como para cada viatura, sem o ruído envolvente, encontram-se representados na Tabela 3.2.

Tabela 3.2: Quantidade de ficheiros de som e duração total de som gravado para cada viatura e para o *garbage*

	Número de gravações	**Tempo total de gravação (s)**
PanhardM11	11	458.4
Mitsubishi 4x4 L200	2	55.1
Leopard-2A6	5	600.4
M60A3	2	88.1
M113	9	423.0
Condor	4	202.4
Protect Fire	5	166.6
Unimog 1300 L	4	139.2
Garabage	15	478.5

Através do Código A.1 Matlab (Little and Moler, 2015), chamado a partir das diretorias correspondentes a cada viatura, os sons foram importados e compostos num só vetor para cada viatura. O vetor de som do M11, Garbage, L200, Leopard-2A6, M60A3, M113, Condor, Protect Fire e Unimog 1300 L encontram-se representados na Figura A.1.

3.1.3 Cálculo dos MFCC e geração dos modelos GMM

Como referido na secção 2.1.2.1, são calculados os MFCC dos vários vetores de amostras sonoras. Isto é feito através da função `mfcc()` de Matlab, desenvolvida por Wojcicki (2011), que retorna os MFCC e recebe o vetor de amostras como *input*, bem como outros parâmetros necessários ao cálculo dos MFCC, tais como:

- Tamanho das janelas de tempo a analisar;
- Sobreposição das janelas de tempo;
- *Pre-emphasis Coefficient* (normaliza a amplitude das bandas de frequência de modo a que estas fiquem uniformes);
- Largura de banda de frequências a analisar.

- Número de canais do banco de filtros de *Mel*.
- Número de coeficientes cepstrais.
- *Cepstral sine lifter parameter* (filtro passa-baixo com objetivo de tornar o sinal mais suave).
- Frequência de amostragem.

Após a obtenção dos MFCC, e de acordo com a secção 2.1.4, é necessário treinar um GMM que caracterize a assinatura desta viatura. Para este fim, é utilizada a função `gmdistribution.fit()` do Matlab, que recebe os MFCC como argumento e que, através do algoritmo de EM referido na secção 2.1.4, encontra o MLE e, por consequência, as médias, covariâncias e coeficientes de mistura associados a este.

O Código A.2, com a lógica representada na Figura 3.2, calcula os MFCC, bem como o modelo GMM associado a estes. Na função `gmdistribution.fit()` é possível especificar:

- O número de gaussianas pretendido (`n_mix`);
- O número de tentativas para o algoritmo EM, neste caso dez `('Replicates',10)`, ou seja, mesmo que algumas falhem e o *Log-likelihood* não consiga convergir, outras conseguem alcançar o MLE, e o modelo GMM é gerado;
- O número máximo de iterações para o algoritmo EM, neste caso mil `('MaxIter',1000)`;
- O *output* no final de cada tentativa `('final')`;
- A semente de início do algoritmo EM `(s = RandStream('mt19937ar','Seed',seed))`, usada para que os modelos gerados com os mesmos *inputs* sejam sempre iguais.

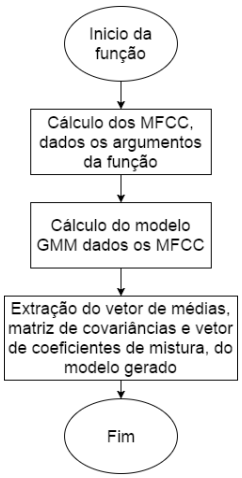


Figura 3.2: Lógica associada ao calculo dos MFCC e GMM.

Na geração dos modelos GMM os vetores de amostras usados incluem só os primeiros 85%, ou seja, são utilizados 85% para treino e os restantes 15% para teste.

3.1.4 Escolha do número de gaussianas

É necessário especificar o número de gaussianas que se pretende para cada modelo. Dado que o número de amostras é limitado e não infinito, não é possível encontrar o modelo "verdadeiro". Desta forma, recorre-se à utilização de critérios de escolha do melhor modelo, dado um conjunto de amostras (Yang, 2005).

Foram utilizados dois critérios, *Bayesian information criterion* (BIC) e *Akaike information criterion* (AIC). O BIC é consistente no sentido em que, se o "verdadeiro" modelo existir entre os candidatos, a probabilidade de selecionar o "verdadeiro" modelo aproxima-se de 100%. Por outro lado, o AIC consegue ter desempenhos semelhantes para casos paramétricos (baseados em famílias parametrizadas de funções densidade probabilidade) e não paramétricos. No geral o AIC não é consistente, enquanto que o BIC consegue-o ser no caso paramétrico (Yang, 2005).

Segundo Yang (2005), devem ser usados os dois critérios para encontrar o melhor modelo em que o BIC deve ser preferível, uma vez que, como referido acima, é mais consistente em relação aos resultados obtidos e, ainda, devido ao fato de, no caso da aplicação em estudo, estar a ser usada uma distribuição gaussiana, que é um caso paramétrico.

O AIC pode complementar o BIC dado que, em ambos os casos, quanto menor for o valor mais indicado é o modelo, e no caso de existirem vários mínimos de BIC, o AIC pode servir como critério de desempate.

Para gerar as figuras com BIC e AIC para vários modelos que vão auxiliar na escolha do melhor modelo, foi desenvolvido o Código A.3, que utiliza as variáveis BIC e AIC do modelo GMM gerado, usando a função `gmdistribution.fit()` referida anteriormente. A lógica deste código encontra-se representada na Figura 3.3.

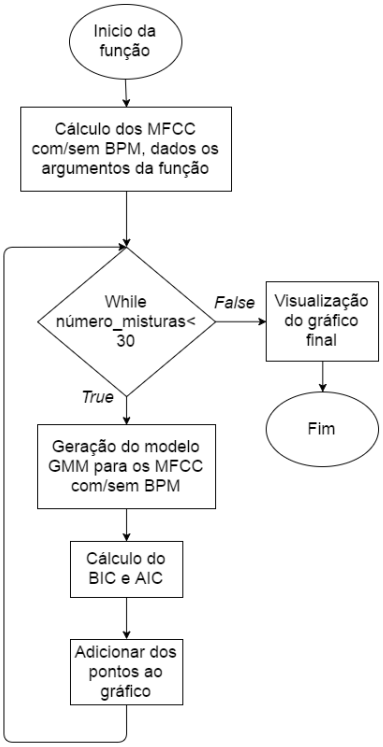


Figura 3.3: Lógica associada ao calculo do BIC e AIC para cada modelo.

3.1.5 Classificação de viaturas

É objetivo de um sensor realizar a recolha de som em janelas mais pequenas, em comparação com o vetor de amostras de som que está a ser inserido para classificação. Para aproximar à realidade, este vetor foi fragmentado em janelas de duração mais reduzida, e foram calculados os MFCC para cada uma destas. O Código A.4, com a lógica representada na Figura 3.4, foi usado para fragmentar o som e calcular os MFCC. O som foi segmentado em fragmentos de um segundo de gravação, com o objetivo de calcular os MFCC para cada um destes.

Inicialmente, são retirados os últimos 15% do som para teste, como referido na Secção 3.1.3. De seguida, são gerados MFCC para cada segundo do som de teste. No caso de o vetor chegar ao fim e existir menos de um segundo, são repetidas algumas amostras e é contabilizado um segundo a partir do fim.

No final, é obtido um vetor de MFCC com uma dimensão a mais, que é em tudo semelhante aos gerados para os modelos, mas contém adicionalmente o número de fragmentos de um segundo que aquele vetor de teste contém.

Com os modelos GMM - médias, covariâncias e coeficientes de mistura e com os MFCC associados aos vetores de teste, torna-se então possível classificar o vetor de teste como pertencente a um dos modelos gerados, usando a Equação 2.5, da Secção 2.1.3. Esta equação permite obter o logaritmo da probabilidade de um som pertencer a um determinado modelo (*log-likelihood*). Ou seja, usando os MFCC da amostra de som, dado um determinado modelo, é possível calcular a probabilidade de este pertencer ao modelo. Quanto maior for a probabilidade, menor é o logaritmo.

É então calculado *log-likelihood*, substituindo o *X* da equação pelos MFCC do som a testar e o μ, Σ e π pelas médias, covariâncias e coeficientes de mistura de cada modelo, respetivamente.

O Código A.5 representa a implementação da equação 2.5 e, como se pode verificar, existe um ciclo exterior que não se encontra representado na equação. Este tem o objetivo de iterar sobre o número de fragmentos de um segundo que o vetor de teste tem. Para cada um destes é calculado o *log-likelihood* e em seguida é gerado um vetor final com todos estes.

Este procedimento é efetuado sucessivamente para todos os modelos, ou seja, no fim são obtidos nove vetores de *log-likelihood's*, um para cada modelo (Código A.6).

Por fim, é feita a média dos vetores de *log-likelihood's*, através do Código A.7 e A.8, obtendo-se assim a média para cada modelo. No Código A.9, são concatenadas as médias dos *log-likelihood* de cada modelo num só vetor, onde através da função `min()` é achado o mínimo de todos estes, bem como o correspondente índice que vai permitir selecionar o modelo que o classificador escolheu como certo para o som que se pretendeu testar. Todo este processo de classificação encontra-se sumariamente explicado na Figura 3.5.

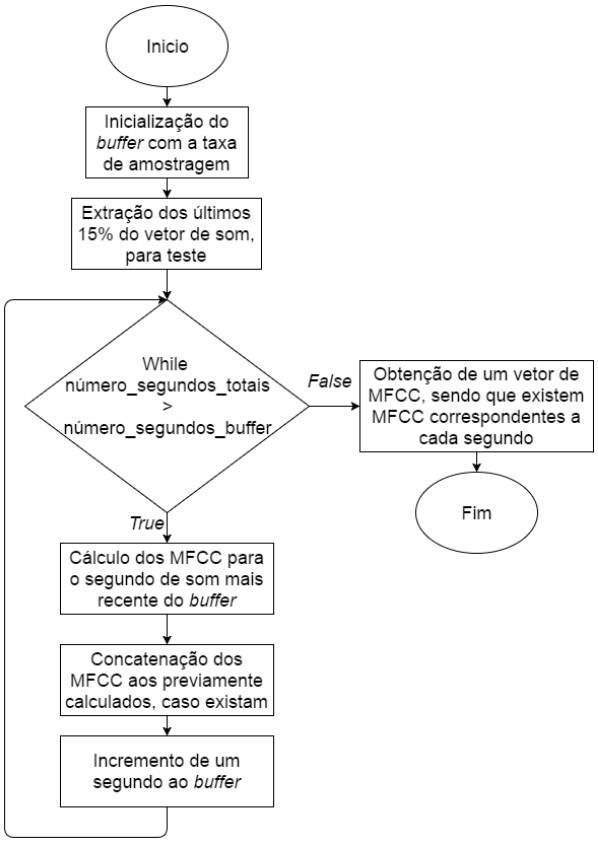


Figura 3.4: Lógica associada à fragmentação e calculo dos MFCC, do som a testar.

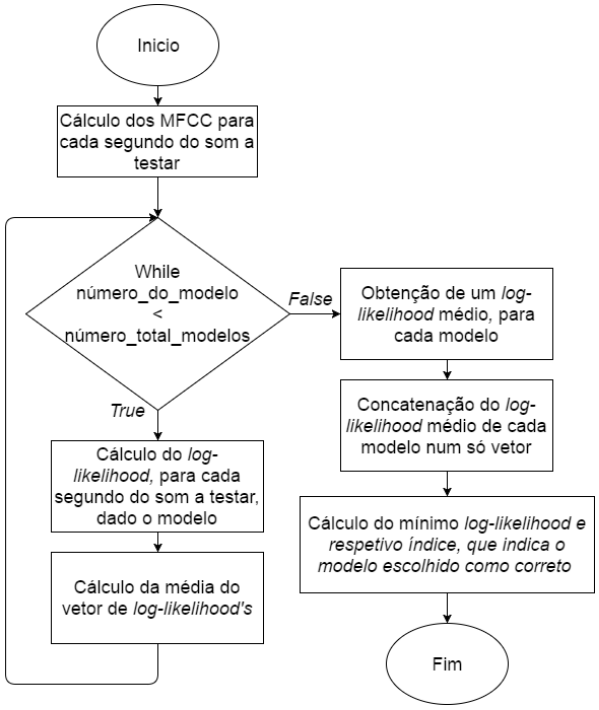


Figura 3.5: Lógica associada ao processo de classificação.

3.1.6 Cálculo e inserção dos BPM na classificação

Para o cálculo dos BPM, foi utilizada a função `tempo2()` de *Matlab*, desenvolvida por Ellis (2007). Esta função calcula o andamento de um som em BPM, dado como argumento o som e a taxa de amostragem.

Esta função retorna um vetor com três valores, que representam os BPM mais lentos, mais rápidos e um parâmetro que especifica a energia relativa dos mais lentos. Isto é, caso os BPM mais lentos tenham uma energia relativa superior a 0.5, são preferíveis aos BPM mais rápidos.

Não se pretende que o peso dos BPM na classificação seja o mesmo que o dos MFCC, pelo que as janelas de som a analisar são quatro vezes superiores às janelas utilizadas pelos MFCC. Ou seja, para quatro valores de MFCC, o valor de BPM é repetido quatro vezes e, enquanto existem oito coeficientes cepstrais nos MFCC, os BPM vão ser apenas um valor para cada janela.

Para inserir os BPM na classificação, foram pensadas duas possibilidades. A primeira passaria por fazer a classificação utilizando os MFCC e, de seguida, utilizando os BPM, onde a classificação final seria feita por escolha. Outra possibilidade seria a inclusão dos BPM nos MFCC já gerados onde a classificação seria feita como o referido na secção anterior.

Destas duas foi escolhida a última modalidade, pois é mais simples de especificar o peso que se pretende atribuir a cada componente do som e permite manter a classificação já desenvolvida.

Desta forma, os BPM foram concatenados nos MFCC, acrescentando uma coluna na matriz destes, ou seja, dado que estão a ser utilizados oito coeficientes cepstrais, a matriz que continha oito colunas será agora uma matriz com nove colunas. Esta nova coluna corresponde aos BPM calculados, não esquecendo que, para cada quatro linhas desta matriz, o valor de BPM repete-se, sendo que as janelas a analisar são quatro vezes superiores.

O Código A.10, com a lógica da Figura 3.6, contem uma função com o objetivo de inserir os BPM numa matriz de MFCC passada como argumento. A `window` que a função recebe como argumento representa a janela dos MFCC multiplicada por quatro, o `sound_vec`, o som que se pretende analisar e, o `mfcc_vec`, a matriz de MFCC. Importa referir que o `buffer`, é inicializado com metade do valor das janelas, pois pretende-se obter sobreposição de 50% no cálculo dos BPM.

O Código A.11 deve ser adicionado na linha 6 do Código A.2 para efetuar a junção dos BPM aos MFCC, bem como a posterior geração dos modelos GMM dos vários veículos.

Por fim, com os modelos gerados, torna-se necessário adicionar os BPM aos 15% finais de cada som que irão ser utilizados para teste. Para este efeito, é utilizado o Código A.12, que é em tudo semelhante ao Código A.4, à exceção de que, para cada janela, são calculados os BPM e, no fim, é adicionada a nova coluna à matriz de MFCC.

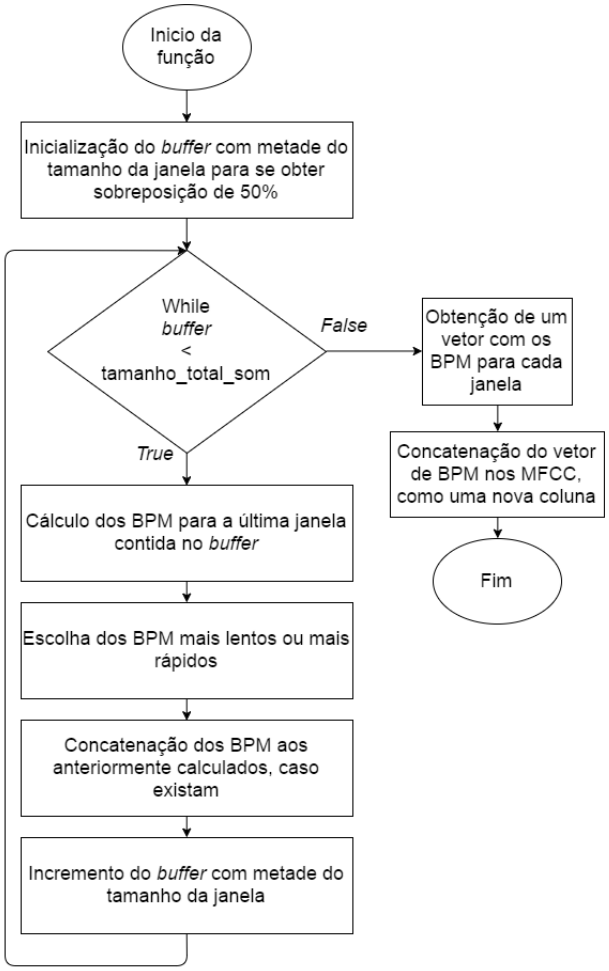


Figura 3.6: Lógica associada à concatenação dos BPM nos MFCC.

3.2 Rede de sensores sem fios

Nesta secção vão ser descritos todos os detalhes relativos à implementação da rede de sensores sem fios, com recurso ao simulador NS-3 (NS-3, 2016), que é desenvolvido na linguagem de programação C++. Todo o Código mencionado nesta secção está incluído no Apêndice C.

Como referido anteriormente, o objetivo deste trabalho é recolher amostras de som de viaturas militares, através de sensores acústicos existentes em nós da rede, efetuar a respetiva classificação e encaminhá-la até à estação base.

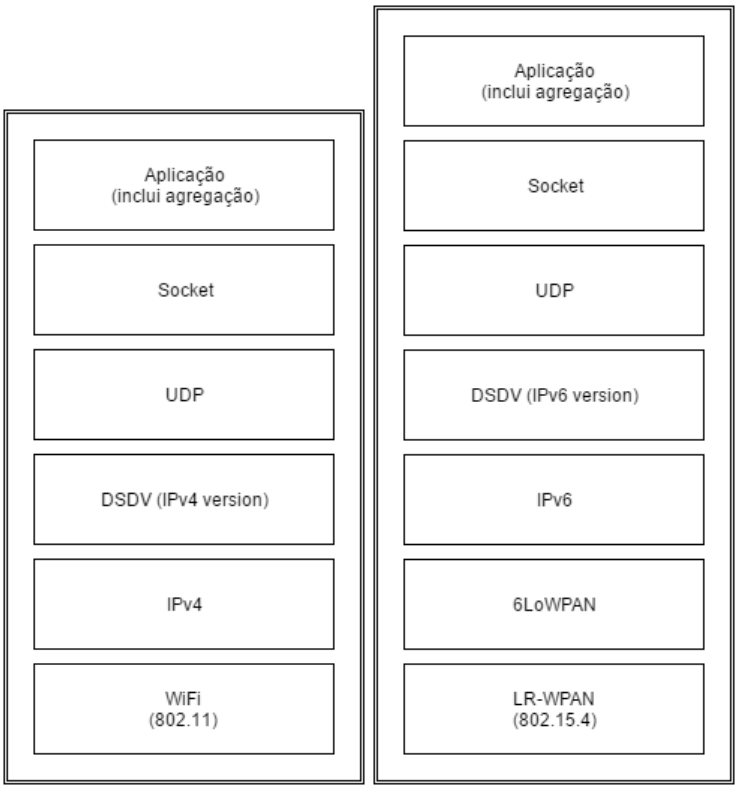
Uma vez que vai ser usado o simulador NS-3 para criar esta rede, e ainda não vai ser realizada implementação ao nível do terreno, pretende-se utilizar o Matlab para realizar todas as tarefas relativas à classificação. O NS-3 vai ser apenas utilizado para simular a rede de sensores sem fios, o que significa que estas duas componentes estão a ser desenvolvidas separadamente. Desta forma, vão ser incluídos num ficheiro valores associados ao número de uma viatura, que correspondem ao resultado da classificação realizada em cada nó.

O único objetivo da rede de sensores simulada é conseguir encaminhar os dados resultantes da classificação até à estação base, da forma mais eficiente possível, recorrendo à agregação de dados em

nós intermédios. Como referido na Secção 2.2.5, foram desenvolvidas duas versões com as principais diferenças:

- Versão 1: Utiliza o protocolo de encaminhamento DSDV, protocolo IPv4 e o *standard IEEE 802.11g* (Wifi);
- Versão 2: Utiliza o protocolo de encaminhamento DSDV, protocolo IPv6, 6LoWPAN e *standard IEEE 802.15.4* (LR-WPAN).

Utilizando a Figura 2.10 como base, as camadas protocolares associadas à Versão 1 e 2 estão representadas nas Figuras 3.7(a) e 3.7(b), respetivamente.



(a) Camada protocolar da Versão 1. (b) Camada protocolar da Versão 2.

Figura 3.7: Camadas protocolares das duas Versões.

Na camada transporte, poderia ser usado o *User Datagram Protocol* (UDP) ou o *Transmission Control Protocol* (TCP), que têm as vantagens e desvantagens representadas na Tabela 3.3 (Karl and Wilig, 2005).

Ao analisar esta tabela, verifica-se que o UDP se adequa mais à aplicação em estudo pois, é pretendido implementar multi-salto para o encaminhamento de pacotes e, além disso, também se pretende reduzir ao mínimo o consumo de energia e o atraso na rede.

Esta conclusão é apoiada por Hofmann et al. (2007), que estudam o desempenho do UDP e do TCP numa aplicação com multi-salto em conjunto com a tecnologia *IEEE 802.11g*. Estes concluem que o UDP é o que produz a melhor solução, pois introduz menos atrasos na rede. Desta forma, para a aplicação em estudo, foi escolhido o protocolo UDP para a camada transporte.

Tabela 3.3: Vantagens e desvantagens de UDP e TCP (Karl and Wilig, 2005).

UDP		TCP	
Vantagens	Desvantagens	Vantagens	Desvantagens
Cabeçalho menor, logo menos gasto de energia em trasnmissões	Não é orientado à conexão, o que pode implicar perdas.	Orientado à conexão, logo mais fiável.	Orientado à conexão, o que implica que os nós intermédios passem despercebidos.
Não é orientado à conexão, o que pode diminuir os atrasos.			Cabeçalhos maiores, logo maior gasto de energia nas transmissões.
			Não funciona bem em aplicações com multi-salto.

Na camada rede, dependendo da versão, é utilizado IPv4 ou IPv6, sendo que, neste simulador, o protocolo de encaminhamento DSDV só está preparado para IPv4 e não tem disponível agregação ou limitação de atualizações só da estação base. Por consequência, foram modificados e desenvolvidos novos módulos de forma a conseguir implementar a Versão 2. Desta forma, a Versão 1 foi a primeira a ser desenvolvida.

3.2.1 Implementação da Versão 1

Para efetuar o desenvolvimento desta versão, utilizou-se um exemplo já existente no simulador NS-3, que continha a pilha protocolar pretendida. Neste exemplo:

- São criados diversos nós, todos com mobilidade, incluindo a estação base;
- É instalada uma aplicação em todos os nós, que envia periodicamente pacotes para a estação base;
- É utilizado o protocolo DSDV para encaminhar os pacotes, sendo que, no início da simulação, existe um tempo em que os nós ainda não enviam pacotes, tempo este utilizado para o DSDV criar as tabelas de encaminhamento em cada nó.

3.2.1.1 *Script* desenvolvido para simulações

Pretendeu-se modificar o exemplo original do NS-3 de modo a que, existisse apenas uma estação base e, que todos os nós da rede fossem estáticos, excetuando um nó, criado para simular uma viatura que se irá deslocar ao longo da rede. Neste *script* são feitas as seguintes operações:

- Criar os nós da rede, incluindo a viatura;
- Adicionar os protocolos à pilha protocolar;
- Instalar a aplicação nos vários nós;
- Especificar os tempos de atualização das tabelas do protocolo DSDV;
- Especificar os tempos de simulação.

3.2.1.2 Leitura de ficheiro com informação relativa às viaturas

Como referido, foi necessário ler de um ficheiro valores correspondentes ao número de uma viatura para, posteriormente, incluir nos pacotes a enviar para a estação base. Foi então desenvolvida uma classe, cujo o objetivo é conter um método que, dado o nome do ficheiro, retorna o seu conteúdo. O conteúdo é retornado em `uint8_t`, uma vez que os valores que identificam a viatura são inteiros. O Código C.1 representa o método referido.

3.2.1.3 Mensagem desenvolvida para informação relativa às viaturas

Sendo possível obter a informação relativa às viaturas, foi então desenvolvida uma mensagem, `carDataHeader`, que contém duas variáveis:

- `cardata` - Esta variável tem como objetivo, conter o valor importado do ficheiro;
- `ID` - Este vetor de inteiros vai armazenar os identificadores de cada nó. Inicialmente é apenas o nó que envia, mas após feita a agregação, se vários nós enviarem os mesmos dados, a cada `cardata` vão ser associados vários nós, ou seja, vários ID's.

Esta mensagem irá ser utilizada na aplicação e adicionada aos pacotes, antes de estes serem enviados.

Para ser possível verificar os atrasos é adicionada uma etiqueta ao pacote, com o tempo no qual este foi enviado. Esta etiqueta é analisada nas agregações de forma a manter sempre o atraso máximo e na estação base, onde é efetivamente feito o cálculo do atraso. Estas etiquetas utilizadas no simulador NS-3, não adicionam qualquer tamanho ao pacote e não afetam o desempenho da rede. A Figura 3.8 ilustra o formato do pacote a enviar.

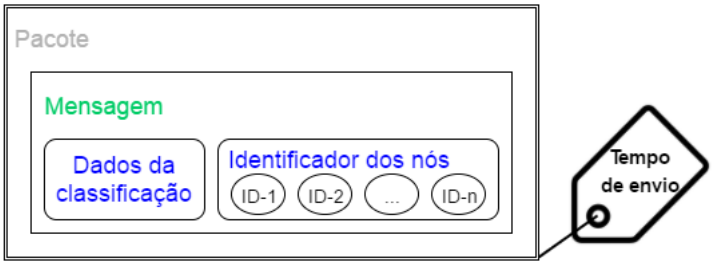


Figura 3.8: Formato do pacote a enviar pelos nós.

3.2.1.4 Alterações no IPv4 camada rede

Antes de desenvolver a aplicação, é necessário garantir que os pacotes possam ser intercetados por esta. Desta forma, foi adicionado ao módulo Internet, mais concretamente ao IPv4, um intercetor de pacotes desenvolvido pelo Professor António Grilo. O intercetor tem como objetivo adicionar à camada aplicação de um nó a possibilidade de o nó intercetar os pacotes que não se destinam a si, de forma a efetuar a agregação ao nível da aplicação. É possível que o pacote seja intercetado pela aplicação na camada IP, antes de este chegar à camada DSDV. Assim, os pacotes são intercetados

antes de serem encaminhados e são enviados posteriormente pela aplicação, após a agregação ter sido realizada. Quando a aplicação envia o pacote, este já passa na camada DSDV e o encaminhamento é feito normalmente. O Código C.2 é o necessário para integrar o intercetor na aplicação. Pode ser visualizado na Figura 3.10 o funcionamento da aplicação e a forma como esta interceção e agregação de pacotes é realizada.

3.2.1.5 Aplicação desenvolvida para os nós da rede

Com os aspetos anteriormente referidos, foi então possível desenvolver as aplicações que vão correr em todos os nós folha da rede e na estação base. Deu-se início ao desenvolvimento destas aplicações, com base numa já existente no módulo NS-3 - `onoff-application` -, cuja finalidade é o envio periódico de um certo número de pacotes para a estação base.

De forma a aproximar a simulação da realidade, foi passado como argumento à aplicação dos nós folha, o endereço do nó viatura, pois é possível, através do módulo de mobilidade do NS-3, calcular a distância a que este se encontra de um nó e, como consequência, adicionar uma condição no envio de pacotes, conforme a distância da viatura.

Em suma, um pacote só é enviado se a viatura estiver a menos de um número de metros especificado. Esta condição é implementada de forma binária, mas o ideal seria desenvolver um módulo de degradação de som em função da distância.

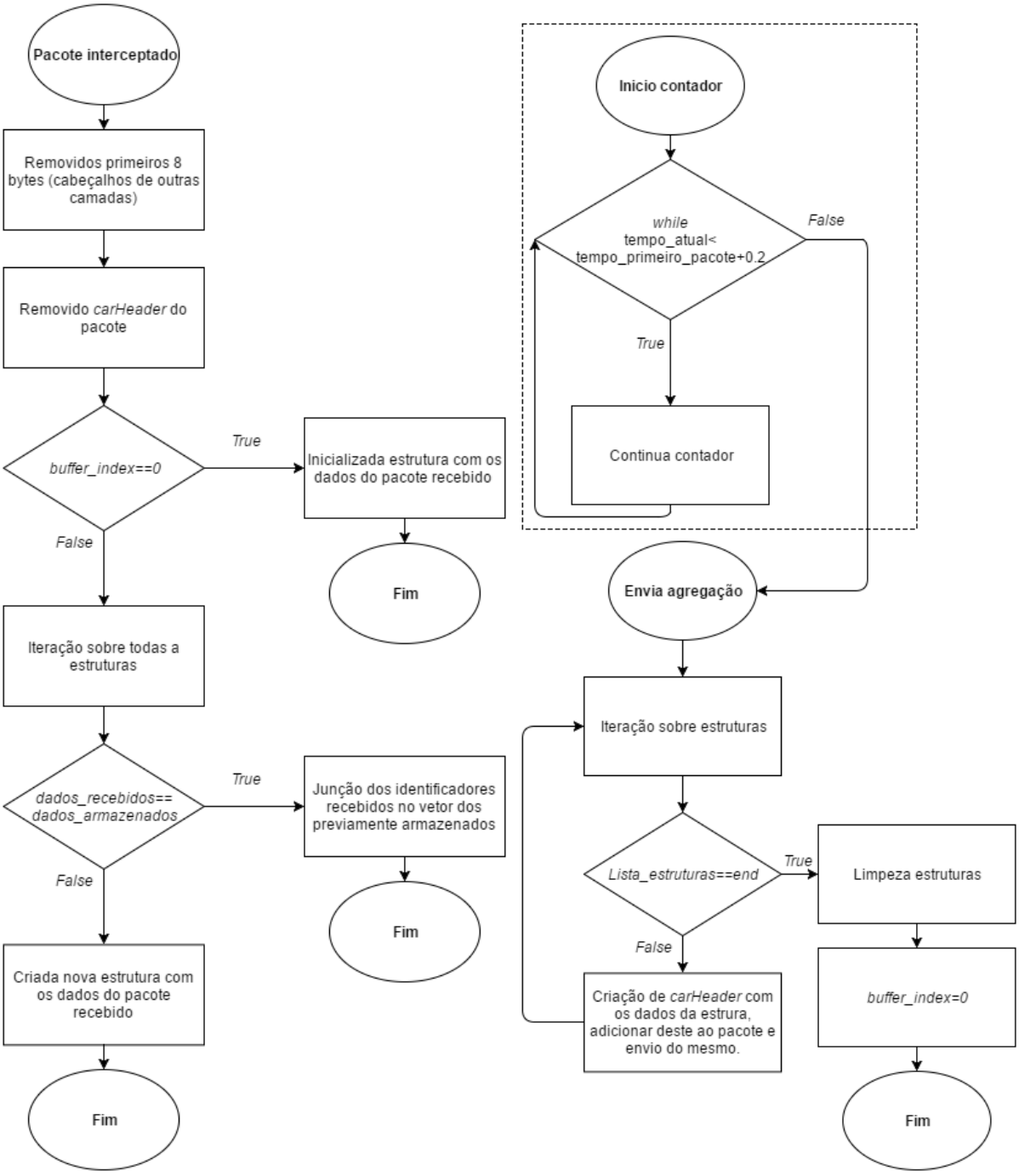
Foi então lido o ficheiro e adicionada a mensagem desenvolvida ao pacote a enviar (Código C.3).

Sendo que todas as aplicações têm instalado o intercetor de pacotes, qualquer pacote que um nó vá encaminhar, vai ser intercetado, pois existe um *callback* para uma função onde vai ser feita a agregação, representada no Código C.4. Após um tempo definido por um contador, é enviada a agregação através do Código C.5. As lógicas associadas a estes códigos, incluindo o contador, encontram-se representadas nas Figuras 3.9(a) e 3.9(b), respetivamente.

Para a agregação, é utilizada uma estrutura de dados que contém, tal como a mensagem, uma variável para os dados, um vetor para os identificadores dos nós e, adicionalmente, uma variável `time`. Cada vez que uma agregação é enviada, esta estrutura é limpa e é utilizada uma variável auxiliar que permite identificar se um nó ainda nunca intercetou nenhum pacote ou se já enviou a agregação e a estrutura se encontra limpa (`buffer_index`). De modo a controlar o contador, cada vez que é intercetado um pacote, após o envio de uma agregação, é guardado o tempo em que este foi recebido, na variável do tipo `time`, e a agregação é enviada passado um tempo especificado.

A aplicação da estação base apenas se limita a receber os pacotes e a efetuar o cálculo das medidas de desempenho descritas na Secção 4.2.

Em suma, o funcionamento da aplicação dos nós folha, encontra-se descrito na Figura 3.10.



(a) Lógica da agregação.

(b) Lógica do envio da agregação.

Figura 3.9: Lógicas da agregação e do envio da mesma.

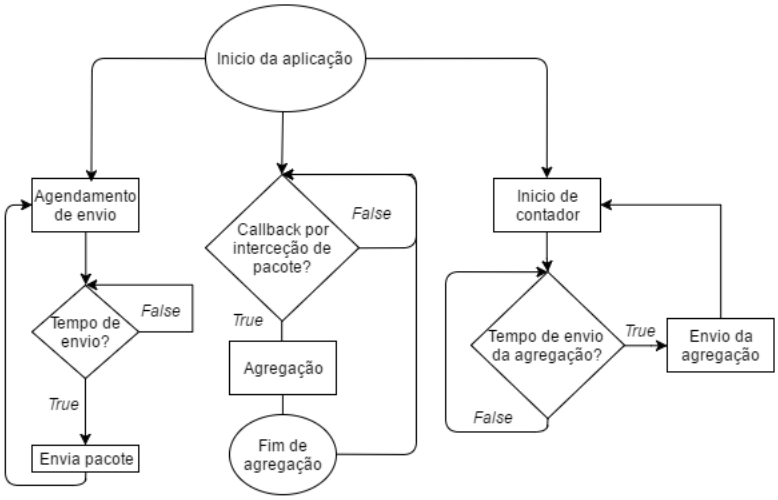


Figura 3.10: Lógica da aplicação.

3.2.1.6 Alterações no módulo DSDV

O módulo de encaminhamento - DSDV, do simulador NS-3, já é compatível com a Versão 1, mas, uma vez que todos os nós atualizam as suas tabelas de encaminhamento com as entradas dos outros nós, existe uma sobrecarga de envios e receções de pacotes em toda a rede. De modo a tornar a rede de sensores mais eficiente a nível energético e mais rápida a nível de simulação, foi desenvolvida uma modificação ao módulo DSDV. Esta modificação tem como finalidade apenas propagar o endereço das estações base para os nós da rede, o que implica que todos os nós vão apenas conter a rota para as estações base. É possível escolher quais os nós que introduzem a sua rota na atualização, tornando esta modificação adequada à aplicação em estudo, mas podendo também ser utilizada em outras aplicações que contenham mais que uma estação base.

Foi então adicionado ao módulo DSDV, um novo atributo (`GenerateUpdates`), que permite escolher se um determinado nó vai introduzir, no pacote que vai encaminhar, a rota para si. Caso o atributo seja "falso", o nó não adiciona a sua rota à tabela a encaminhar, caso seja "verdade", adiciona.

No módulo DSDV original, a procura do destino nas tabelas de cada nó não era compatível com esta nova implementação, ou seja, quando um nó enviava um pacote com o destino estação base e esta não estava ao alcance, o pacote era descartado, visto não existir na tabela de encaminhamento nenhuma rota com o destino do próximo salto. O único conteúdo da tabela é a rota com o destino estação base, em que o *gateway* é o endereço do próximo salto. Utilizando esta informação, é feita manualmente uma rota, que é retornada para a próxima camada, de modo a que o pacote siga a rota certa e chegue à estação base. O funcionamento da camada DSDV encontra-se resumido na Figura 3.11.

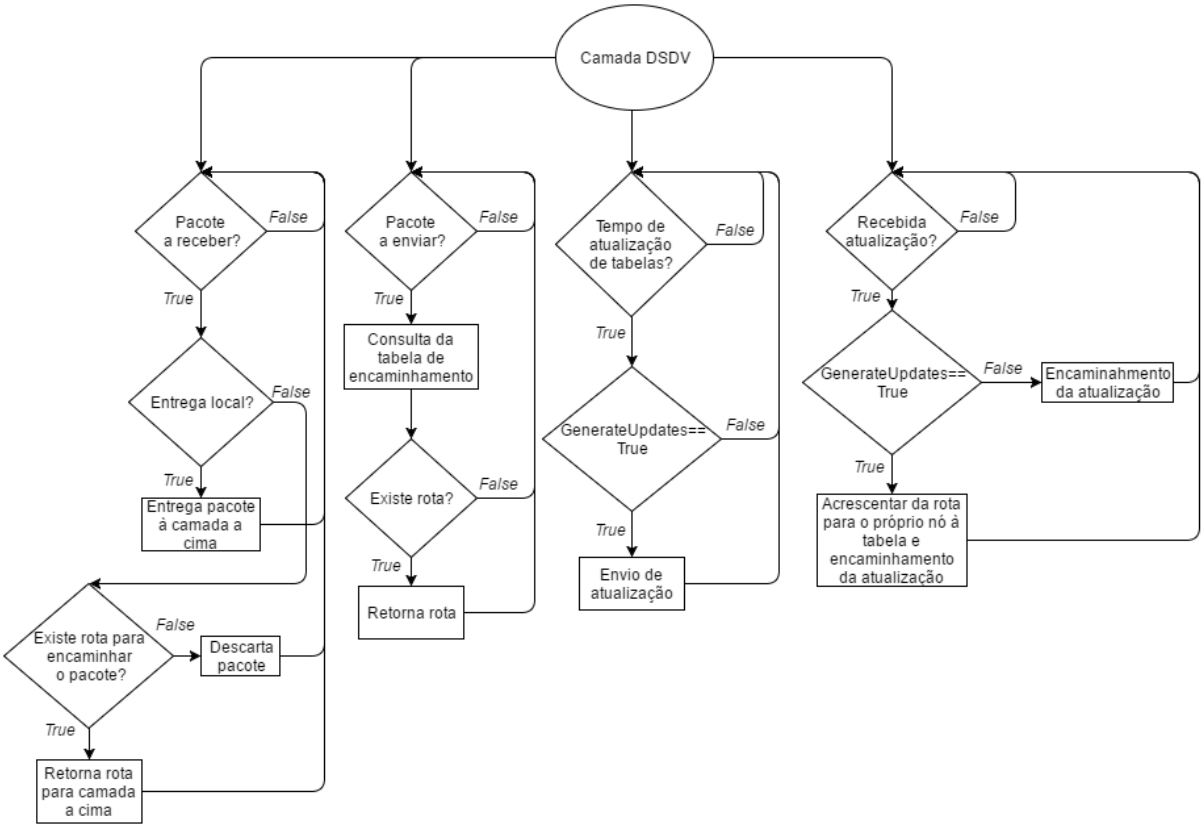


Figura 3.11: Lógica da camada DSDV.

3.2.2 Implementação da Versão 2

A abordagem ao desenvolvimento da Versão 2 foi um pouco diferente, uma vez que o módulo DSDV não era compatível com IPv6. Começou-se por colocar os módulos correspondentes às tecnologias *6LoWPAN* e *LR-WPAN (802.15.4)* a funcionar em conjunto, utilizando o exemplo `example-ping-lr-wpan` do NS-3.

De seguida, introduziu-se o UDP nesta arquitetura, substituindo a aplicação utilizada no exemplo referido, pelas aplicações `udp-client` para os nós folha, e `udp-server` para estação base. Com esta alteração, foi possível criar uma rede, com os nós ao alcance da estação base, que enviam periodicamente os pacotes, utilizando UDP.

Por último, foram feitas as devidas alterações ao protocolo DSDV, para que este pudesse ser introduzido na arquitetura desenvolvida, de forma a obter as mesmas características de simulação da Versão 1, mas utilizando o protocolo *IEEE 802.15.4* que, como referido na Secção 2.2.5, permite um menor consumo energético na rede e, consequentemente, uma maior autonomia.

É importante referir que, muito do código desenvolvido na Versão 1, é aplicável na Versão 2. A leitura do ficheiro, a mensagem desenvolvida e o código relativo à agregação e ao envio desta foram utilizados nesta nova Versão.

3.2.2.1 *Script* desenvolvido para simulações

Para o desenvolvimento deste *Script*, foi tido como exemplo o código `example-ping-lr-wpan` do simulador. A este foi acrescentada a velocidade nula para os nós da rede e uma velocidade constante para um nó viatura. Foi também adicionado à pilha protocolar o módulo DSDV (versão IPv6), para encaminhamento, e as aplicações aos nós da rede, com as devidas alterações.

Num caso real, o envio de um pacote por centenas de nós, no exato instante, é altamente improvável. Por consequência, foi introduzido um atraso no envio de pacotes entre os vários nós, de forma a evitar colisões.

3.2.2.2 Alterações no IPv6 camada rede

Foi criado um novo intercetor de pacotes baseado na versão IPv4 utilizada na Versão 1, de modo a que este possa ser integrado na camada rede do IPv6.

Este intercetor tem o mesmo objetivo e funcionalidades do desenvolvido pelo Professor António Grilo, contendo apenas conversões de classes IPv4 para IPv6 no código.

3.2.2.3 Alterações no módulo DSDV

Todas as alterações feitas anteriormente no módulo DSDV para IPv4 foram mantidas no desenvolvimento desta nova versão para IPv6.

As principais alterações foram:

- Todas as classes IPv4 foram convertidas para IPv6;
- Todos os endereços de *broadcasts* (255.255.255.255) em IPv4 foram convertidos para *all nodes multicast addresses* (FF01:0:0:0:0:0:0:1) em IPv6;
- Tabelas são preenchidas com endereços IPv6 globais e não locais.

O grande problema de incompatibilidade deste módulo com IPv6, prende-se com o protocolo de *Neighbor Discovery* (NDP), similar ao *Address Resolution Protocol* (ARP), que existe no IPv4. O NDP tem endereços específicos chamados de *solicited-node multicast addresses*, que são utilizados para fazer pedidos aos nós, de modo a obter o endereço MAC. Estes pedidos são feitos, utilizando o protocolo *Internet Control Message Protocol version 6* (ICMPv6), que no simulador funciona ao nível da camada transporte. O ICMPv6 contem diferentes tipos de pacotes, um deles (*Neighbor Solicitation*), cuja função é determinar o endereço MAC dos nós vizinhos, enviando estes pacotes para o *solicited-node multicast address*. É importante referir que no ICMPv6, são enviados primeiro pedidos de *Router Solicitation*, para endereços *multicast* de todos os *routers*, e só após receção de resposta é que são enviados os pedidos de *Neighbor Solicitation*.

Na Versão 1, a necessidade de ARP não constitui nenhum problema, uma vez que são utilizados endereços *broadcast* e existe transparência para a camada do DSDV no que se refere a estes pedidos. No caso do NDP, os endereços utilizados para estes pedidos são outros, e como tal, não é possível ignorar o encaminhamento do DSDV, ou consultar as tabelas e retornar uma rota, pois esta não existe,

ou seja, não existe transparência no DSDV para os pacotes ICMPv6, o que leva a que estes sejam descartados.

É então necessário tornar os pacotes ICMPv6 transparentes para o protocolo DSDV. Para tal, é efetuada uma verificação de endereço de destino, e retornada manualmente a rota para as camadas abaixo, como se simplesmente a camada DSDV não existisse. Na camada de transporte, o protocolo ICMPv6 trata de decidir se o nó que recebeu o pacote é o destino e realiza os procedimentos de envio de um pacote de resposta. Caso o nó não seja o destino, o pacote é simplesmente descartado.

Na prática, foi adicionado à função que trata dos envios no módulo DSDV (versão IPv6), o Código C.6, que permite efetuar a verificação do endereço de destino como sendo *solicited-node multicast address* ou *all routers multicast*, e retornar uma rota mantendo o mesmo destino e origem. Na receção de pacotes ICMPv6, no DSDV, o destino pode ser *multicast* ou o endereço global do nó. Caso tenha chegado ao destino certo, o protocolo descarta estes pacotes, pois não existem estes endereços nas tabelas de encaminhamento. Por consequência, é necessário introduzir uma condição (Código C.7) que envie para a camada acima os pacotes ICMPv6 com estes destinos, de modo a que estes sejam corretamente tratados.

Em suma, o funcionamento é idêntico à camada DSDV utilizada na versão IPv4, mas existe uma filtragem de alguns pacotes ICMPv6. A Figura 3.12 explica o funcionamento, em traços gerais, da camada DSDV para IPv6.

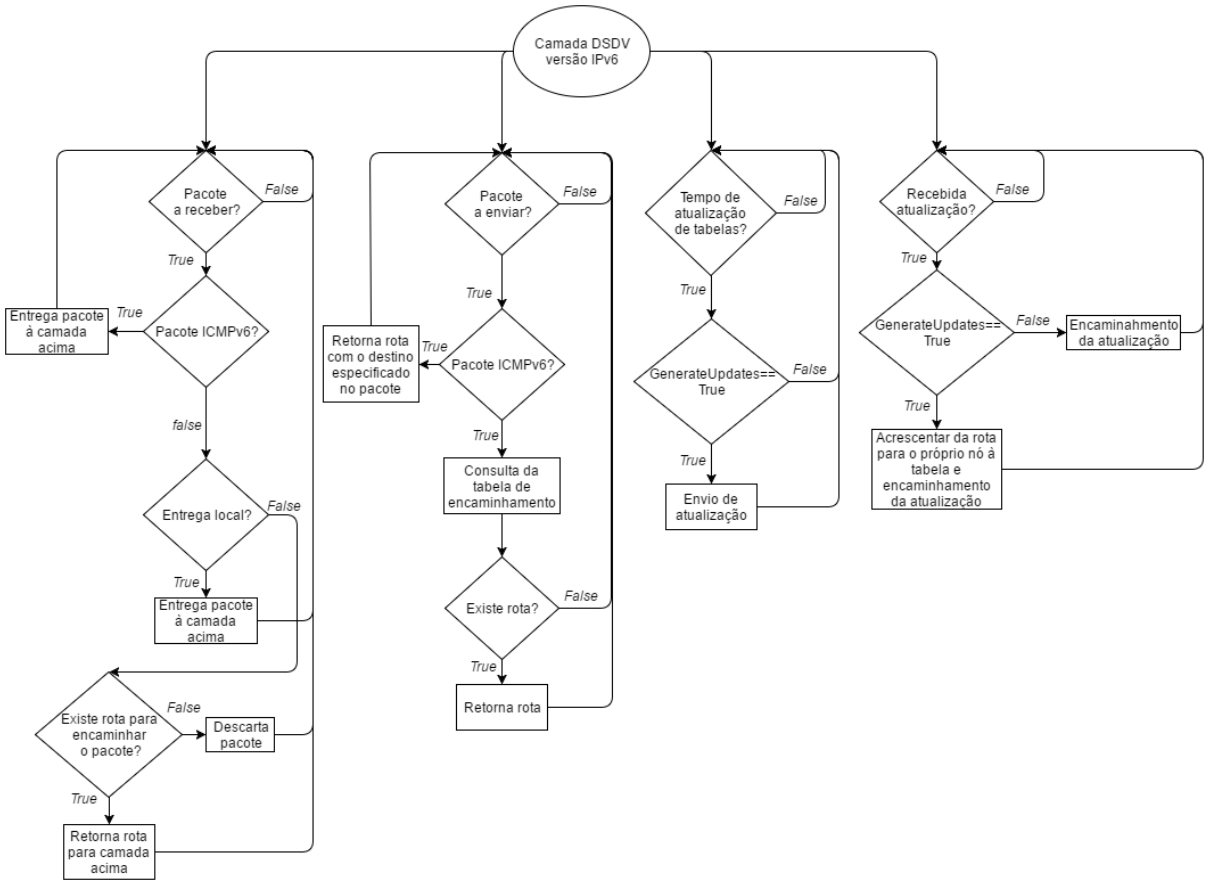


Figura 3.12: Lógica da camada DSDV versão IPv6.

3.2.2.4 Aplicação desenvolvida para os nós da rede

Na Versão 2, foram utilizadas duas aplicações, uma para a estação base – servidor, e outra para os restantes nós da rede - clientes. Estas novas aplicações `udp-client_mod` e `udp-server_mod`, são modificações das `udp-client` e `udp-server` do simulador NS-3.

Tal como na Versão 1, foi adicionado à aplicação `udp-client_mod` o intercetor de pacotes, o contador, a agregação e o envio da mesma, com as respetivas modificações ao protocolo IPv4 para IPv6.

Pretendeu-se tornar as implementações das camadas destas duas Versões o mais parecidas possível, excetuando as camadas físicas (*802.11g* ou *802.15.4*), de modo a que os resultados das simulações das redes com diferentes Versões não sejam influenciados por aspetos relacionados com a implementação.

Capítulo 4

Resultados

4.1 Caracterização e classificação de viaturas utilizando GMM

Nesta Secção são apresentados os resultados obtidos utilizando a implementação descrita na Secção 3.1, utilizando dados sintéticos, bem como os dados reiais recolhidos. Algumas das Tabelas, Código e Figuras, encontram-se representados no Apêndice B.

Todas as simulações Matlab foram realizadas no meu computador pessoal, que possui as características representadas na Tabela 4.1.

Tabela 4.1: Características do computador pessoal para realização das simulações Matlab

Características do computador pessoal	
RAM	16GB
CPU	Intel® Core™ i7-4700MQ
Utilização do CPU durante a classificação	20-23%
Número de núcleos do CPU utilizados	1
Sistema operativo	Windows 8.1 Pro (64 bits)

4.1.1 Testes de classificação com dados sintéticos

De forma a ser possível testar o funcionamento da função `gmdistribution.fit()`, foram criados conjuntos de dados aleatórios com distribuição gaussiana, que têm associada uma determinada média e desvio padrão. Aos dados foi aplicada a função referida, variando o número de Gaussianas. Desta forma, os testes realizados foram:

- Teste 1: Vetores com normas e desvio padrão iguais;
- Teste 2: Vetores com diferentes médias e desvios padrão iguais;
- Teste 3: Vetores com médias iguais e desvios padrão diferentes;
- Teste 4: Vetores com diferentes médias e desvios padrão;
- Teste 5: Vetores com diferentes médias e desvios padrão mas em quantidades diferentes.

Foram utilizados três blocos de código de *Matlab* utilizando diferente número de componentes Gaussianas na função `gmdistribution.fit()`, e diferente número de vetores em cada teste.

- Foram utilizados dois vetores e uma componente Gaussiana em todos os testes. Resultados representados na Tabela B.1;
- Foram utilizados dois vetores e duas componentes Gaussianas em todos os testes. Resultados representados na Tabela B.2;
- Foram utilizados quatro vetores e quatro componentes Gaussianas em todos os testes. Resultados representados na Tabela B.3.

Estes primeiros testes contemplam dados sintéticos e isto significa que em vez de serem utilizados sons reais, foi utilizado um gerador de números aleatórios para gerar um conjunto de observações distribuídas uniformemente com uma dada média e uma dada covariância.

Começando com o Teste 1, fez-se uma estimativa de um GMM com uma componente que representasse estas observações utilizando a função `gmdistribution.fit()` que usa o algoritmo EM. Verifica-se que após este algoritmo convergir, se obtém uma média de `-0,001` e uma covariância de `0,991`, que se encontram suficientemente próximas de `0` e `1`, como se pode verificar.

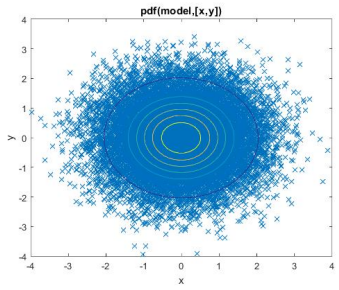
No Teste 2 foram criados novos dados sintéticos mas distribuídos segundo normais com médias diferentes e covariâncias iguais e novamente submeteu-se o GMM, com uma componente, a um processo de treino baseado no mesmo algoritmo. Verificou-se que um GMM com uma só componente, não consegue representar os dados da melhor forma, pois existem dois grupos distintos de dados. O mesmo é constatado nos restantes testes em que existem grupos de dados em número superior ao número de componentes do GMM.

No Teste 3 pode verificar-se também, que quando os conjuntos de dados têm a mesma média e os desvios padrões um pouco diferentes os conjuntos de dados não são distintos, ou seja, o treino do GMM pode não originar médias e covariâncias que se aproximem das originais.

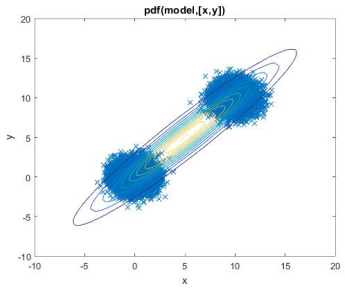
Este processo foi repetido, mas os modelos GMM foram treinados com mais uma componente e é possível verificar que, para conjuntos de dados dispersos em dois grupos distintos, os GMM já continham médias e covariâncias similares às dos dados sintéticos. Este resultado permite concluir que é necessário, pelo menos, um número de componentes igual ao número de grupos, classes ou tipos de dados distintos.

Aumentando ainda mais o número de componentes, verifica-se que não trouxe nenhuma melhoria para classificar um único grupo distinto de dados, pelo contrário, nenhuma das componentes consegue representar os dados tão bem como no primeiro caso, onde existe apenas uma componente.

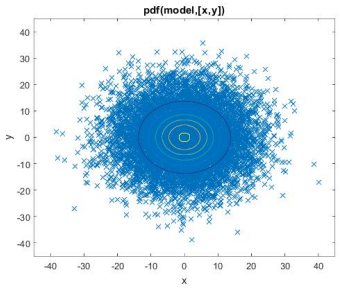
Os resultados para os testes realizados, com diferente número de componentes, podem ser observados nas Tabelas B.1, B.2 e B.3 associadas, respetivamente, às figuras 4.1, 4.2 e 4.3 com os GMM representados em duas dimensões.



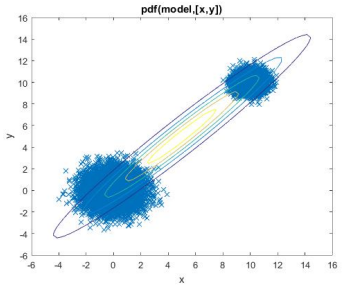
(a) Dois vetores com normas e desvio padrão iguais.



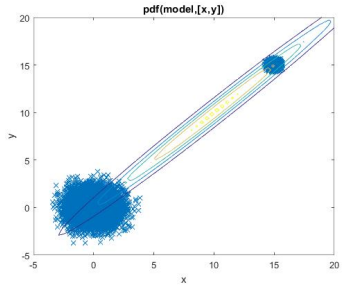
(b) Dois vetores com diferentes médias e desvios padrão iguais.



(c) Dois vetores com médias iguais e desvios padrão diferentes.

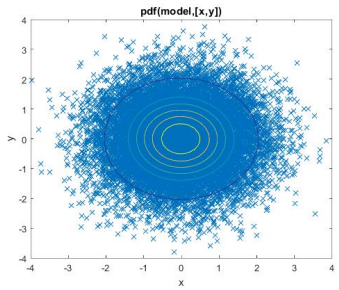


(d) Dois vetores com diferentes médias e desvios padrão.

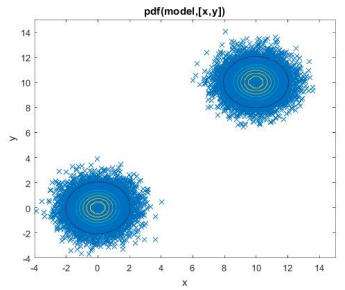


(e) Dois vetores com diferentes médias e desvios padrão mas em quantidades diferentes.

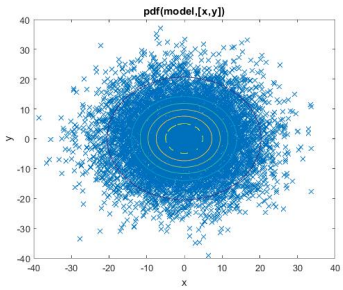
Figura 4.1: Teste de GMM com uma Gaussiana a dados sintéticos



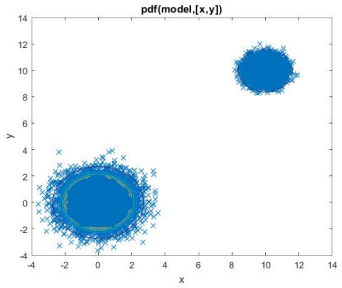
(a) Dois vetores com normas e desvio padrão iguais.



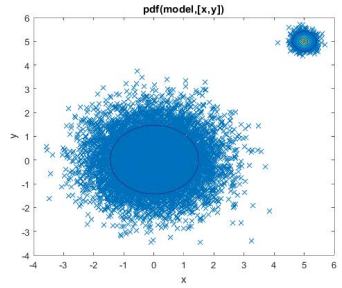
(b) Dois vetores com diferentes médias e desvios padrão iguais.



(c) Dois vetores com médias iguais e desvios padrão diferentes.

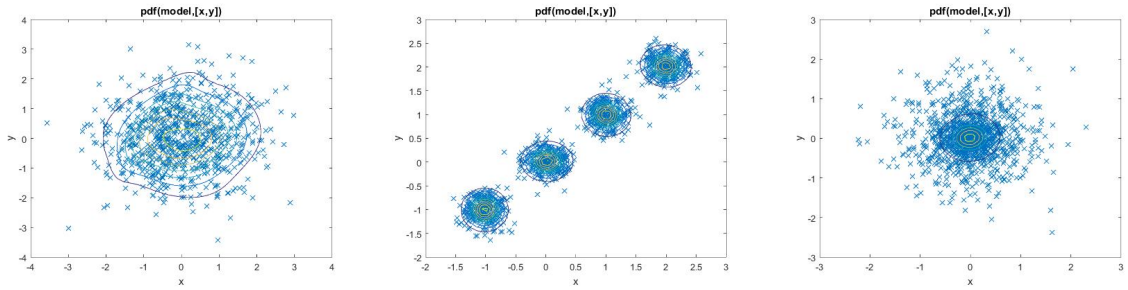


(d) Dois vetores com diferentes médias e desvios padrão.



(e) Dois vetores com diferentes médias e desvios padrão mas em quantidades diferentes.

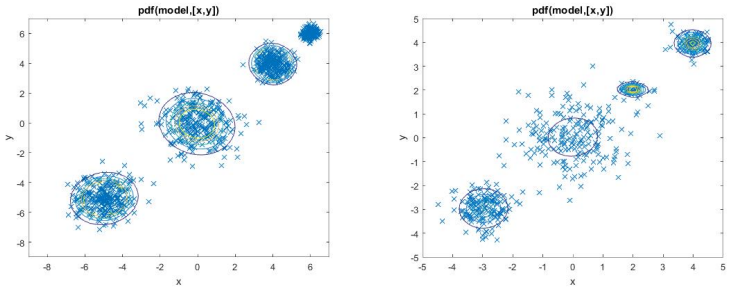
Figura 4.2: Teste de GMM com duas Gaussianas a dados sintéticos



(a) Quatro vetores com normas e desvio padrão iguais.

(b) Quatro vetores com diferentes médias e desvios padrão iguais.

(c) Quatro vetores com médias iguais e desvios padrão diferentes.



(d) Quatro vetores com diferentes médias e desvios padrão.

(e) Quatro vetores com diferentes médias e desvios padrão mas em quantidades diferentes.

Figura 4.3: Teste de GMM com quatro Gaussianas a dados sintéticos

Em suma, as conclusões foram:

- Número de componentes igual ao número de grupos distintos permite que o GMM represente bem os dados;
- Se os grupos de dados não forem distintos, aumentar o número de componentes pode não melhorar a classificação;
- Quando existem menos componentes que o número de grupos de dados distintos, as médias das componentes aproximam-se dos grupos de dados com mais amostras e dos mais densos (com menor distância entre eles) e as covariâncias aumentam;
- Se só existir um grupo distinto, um GMM com componentes a mais, pode ser redundante ou começar a representar os dados de forma errada.
- Caso existam poucas amostras, ainda que com a mesma média e desvio padrão, estas podem não conseguir definir um grupo distinto e o GMM pode não representar bem estes dados.

Com estas conclusões, e ao verificar o funcionamento desta ferramenta, torna-se possível avançar para os testes com dados reais.

4.1.2 Testes iniciais aos sons recolhidos

Após as conclusões resultantes dos testes aos dados sintéticos, foram iniciados os testes aos dados reais - sons das viaturas.

Como referido na Secção 3.1.3, existem parâmetros com possibilidade de especificação para o cálculo dos MFCC. Para os testes iniciais foram escolhidos dois coeficientes cepstrais, de forma a que os MFCC gerados contenham apenas duas dimensões e sejam comparáveis aos testes realizados aos dados sintéticos.

Em todos os testes realizados a sobreposição das janelas escolhida foi de 50% e as janelas testadas foram: 50ms, 100ms, 150ms, 250ms, 300ms, 350ms, 400ms, 450ms, 500ms, 550ms, 600ms, 800ms e 1000ms.

A situação ideal será possuir o som das viaturas com características espetrais suficientemente afastadas umas das outras para que possam ser bem representadas por modelos distintos de GMM com médias e covariâncias bem diferenciadas, como foi simulado na Secção anterior. Infelizmente, vai-se verificar mais à frente que, ao contrário daquilo que seria desejável, a sobreposição dos dados será elevada, não existindo grupos de dados distintos, o que se traduz numa deteção com erros como irá ser visto.

Para visualizar a disposição dos dados (MFCC) em duas dimensões, foram geradas figuras para cada janela, as quais se encontram representadas na Figura B.1.

É visível que a sobreposição dos dados das várias viaturas é elevada, e não existem, em muitos casos, grupos de dados distintos. Embora se verifique que grande parte dos dados se encontra sobreposta, aparece ainda uma percentagem substancial de dados suficientemente longe para poderem ser classificados corretamente. Por exemplo, os dados do M113 estão mais agrupados numa certa zona, sobrepostos parcialmente pelos dados do Panhard M11.

Se for acrescentada uma dimensão aos MFCC[1], ou seja, mais um coeficiente cepstral, verifica-se, ao rodar em três dimensões os dados no Matlab, na Figura B.2, que os grupos de dados são mais distintos. É ainda de notar que, ao aumentar cada vez mais o número de coeficientes, os grupos vão-se tornando ainda mais distintos e os GMM têm a capacidade de representar cada vez melhor os dados. É também importante referir que, segundo CMUSphinx (2016), a partir dos 12 coeficientes cepstrais não existe melhoria nos MFCC, podendo mesmo piorar a classificação. Desta forma, foram aumentados os coeficientes até ao máximo possível, dada a quantidade de dados disponível e o número obtido foi de "oito".

O facto de existir uma grande sobreposição dos grupos, deve-se, em parte, à quantidade reduzida de som de cada viatura. Ou seja, a base de dados de som que foi desenvolvida não possui tempo suficiente de gravação para existir uma separação distinta entre viaturas. No trabalho desenvolvido por Portelo et al. (2009), são utilizadas horas de gravação para a maioria dos "eventos". Na aplicação em estudo, foram recolhidos menos de uma dezena de minutos por viatura, sendo que em algumas delas o tempo foi inferior a dois minutos.

A criação de uma boa base de dados é um processo moroso, em que seria necessária a gravação de som, não só das viaturas referidas ao longo do trabalho, bem como de outras não referidas, mas também utilizadas pelas Forças Armadas Portuguesas, nos mais diversos ambientes. Uma vez que o âmbito desta tese não é a criação de uma base de dados de sons de viaturas, foram apenas recolhidas

[1] Está a ser aumentada a dimensão dos dados utilizados para gerar os modelos GMM, que corresponde ao x utilizado na Equação 2.2

amostras de som necessárias à elaboração de uma prova de conceito.

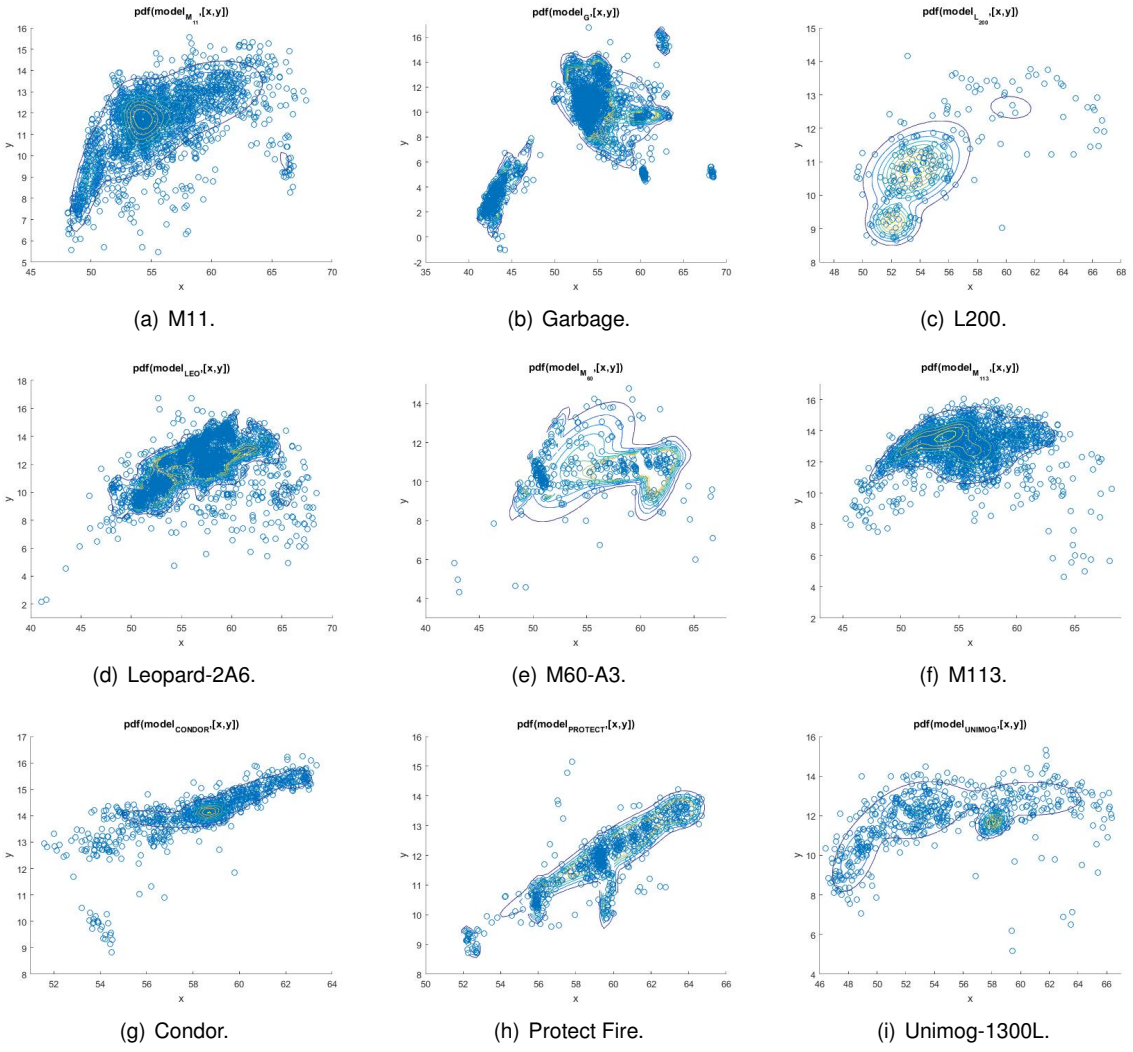
De forma a ser possível efetuar uma comparação entre os resultados obtidos e os dados sintéticos, é necessário gerar figuras para cada GMM, em cada janela. Para que tal seja possível, é utilizado o Código A.3, responsável por estimar, por cada janela e para cada viatura, qual o número ideal de componentes do GMM. São apenas incluídas na Figura B.3, como exemplo, as imagens de BIC e AIC, referentes a uma janela. O motivo desta decisão deve-se ao facto de existir um total de 351 figuras pois, para cada janela, foram geradas nove figuras - num total de treze janelas, cujo procedimento foi feito para dois coeficientes e oito coeficientes, com e sem BPM. É importante referir que, em certas viaturas para janelas mais pequenas, foi necessário testar até 60 componentes, uma vez que, como existem mais dados, o número de componentes ideal é mais elevado.

Como referido na Secção 3.1.4, na Figura B.3 encontra-se o mínimo do BIC e, caso existam dois ou mais pontos com o mesmo mínimo, o AIC é utilizado como fator de desempate, ou seja, destes pontos o que tiver menor AIC é o escolhido. Este procedimento é aplicado para todas as viaturas, em todas as janelas. Os resultados para MFCC com dois coeficientes, encontram-se representados na Tabela B.5.

Ao analisar a Tabela B.5, verifica-se que, para janelas mais pequenas, o número de componentes tende a ser maior, uma vez que existem mais MFCC, ou seja, mais dados para o GMM representar. É também possível constatar este facto, quando se analisa a Tabela 3.2 e se verifica que as viaturas que têm mais tempo de gravação, são aquelas que contêm mais componentes na Tabela B.5. Com os valores contidos nesta tabela, é possível gerar os modelos para as viaturas com o número ideal de componentes, e dado um som para testar, efetuar a respetiva classificação.

Utilizando os últimos 15% de som de cada viatura (som de teste), não incluídos na geração dos modelos, foram efetuadas as classificações. Para cada som de teste, foram aplicados os procedimentos descritos na Secção 3.1.5 e foi obtida a Tabela B.8. Nesta tabela, apenas é descrito se o som foi bem classificado (1), ou se foi mal classificado (0). Note-se que, caso um som seja bem classificado, existe um verdadeiro positivo e sete verdadeiros negativos mas, no caso de o som ser mal classificado, existe um falso positivo (classificação errada retornada pelo classificador), um falso negativo (classificação certa que deveria ser retornada pelo classificador) e os restantes são verdadeiros negativos. Com estes dados foi possível gerar a Tabela B.11, que contém o resultado das medidas de desempenho mencionadas na Secção 2.1.5. Nesta tabela, foi definido o β igual 1, para atribuir igual peso ao *recall* e à precisão.

A melhor classificação foi obtida com as janelas de 50ms, 100ms e 400ms, com um *F-score* de 0,625. Sendo que o tempo de execução da classificação é mais rápido com a janela de 400ms (Tabela B.4), apenas vão ser representadas as figuras dos GMM associadas a esta. Analisando a Figura 4.4, verifica-se que, as misturas Gaussianas tentam representar os dados, tal como nas misturas dos dados sintéticos, e dado que os grupos de dados não são tão distintos, existe um maior número de pontos mal representados.



(a) M11. (b) Garbage. (c) L200.

(d) Leopard-2A6. (e) M60-A3. (f) M113.

(g) Condor. (h) Protect Fire. (i) Unimog-1300L.

Figura 4.4: Disposição dos GMM para MFCC com dois coeficientes cepstrais

4.1.3 Testes finais aos sons recolhidos

Como referido na Secção 4.1.2, o número de coeficientes cepestrais ideal para os MFCC é oito. Desta forma, para os testes finais, foi utilizado este valor, bem como as mesmas janelas de tempo.

Os testes realizados foram:

- Teste 1: MFCC com oito coeficientes cepstrais para as várias janelas e respetiva classificação;
- Teste 2: MFCC com oito coeficientes cepstrais e BPM para as várias janelas e respetiva classificação.

Iniciou-se então o Teste 1, gerando as figuras de BIC e AIC para criar a Tabela B.6, que contém o número ideal de componentes para cada modelo. Como se pode verificar, os números de componentes nesta tabela, comparativamente à Tabela B.5, são substancialmente superiores, pois, como existem mais coeficientes nos MFCC existem mais dados e, possivelmente, grupos mais distintos. De seguida, foram feitas as classificações e geradas as Tabelas B.9 e B.12. Verifica-se que, quando comparando com as Tabelas B.8 e B.11, para oito coeficientes, existem mais acertos na classificação, e logo um

maior *F-score*. Pode-se portanto concluir que, um aumento no número de coeficientes - de dois para oito, se traduz numa melhoria na classificação.

Por último, foi realizado o Teste 2, que foi em tudo semelhante ao Teste 1, com a exceção de que nos testes BIC e AIC e na geração dos modelos foi adicionado uma nova coluna de dados, como referido na Secção 3.1.4, através da inserção dos BPM nos MFCC. Analisando a Tabela B.7, conclui-se que esta contém, no seu conjunto, um número de componentes superior à Tabela B.6. A justificação para este resultado é a mesma de quando se passa de dois para oito coeficientes: ao adicionar os BPM, existe uma nova coluna e, por conseguinte, mais dados no total.

Foram feitas então as classificações, utilizando os sons de teste e os modelos gerados, criando assim as Tabelas B.10 e B.13. Durante a classificação (no cálculo do *log-likelihood*), nas janelas de 150ms, 550ms e 1000ms, as matrizes são quase singulares, ou seja, os determinantes são perto de zero, e no caso de serem zero, não é possível efetuar a inversa da matriz.

Verifica-se que, os resultados do Teste 1 são, no geral, melhores que os do Teste 2, ou seja, ao adicionar os BPM, a classificação piora. Isto pode dever-se ao facto desta componente do ritmo não ser a melhor característica do som para distinguir as viaturas. É importante referir que, em ambos os testes é possível adquirir um *F-score* de 0,875, ou seja, uma taxa de acerto de sete viaturas em oito mas, a nível geral, a classificação no Teste 1 é melhor.

Um fator importante na escolha da melhor opção, prende-se com o tempo de execução, isto porque, no caso real, os nós têm de processar o som e classificar a viatura antes de enviar o resultado até à estação base. Quanto mais rápida for a classificação, mais rapidamente os dados são enviados e mais cedo são recebidos na estação base. Caso seja futuramente desenvolvido o rastreio de movimento, este deve ser o mais perto do tempo real possível, logo a classificação tem de ser feita da forma mais eficiente a nível de tempo de execução.

Durante a classificação, os blocos de Código que têm maior tempo de execução são os A.4 e A.5 da Secção 3.1.5. Este tempo de execução está relacionado com:

- Tamanho das matrizes que entram nos cálculos do *log-likelihood*, Código A.5;
- Tamanho das janelas dos MFCC no Código A.4.
- Quantidade de MFCC do som a testar.

Quanto menores forem as matrizes de médias e covariâncias e menos MFCC existirem, mais rápida é a classificação mas, o tempo de cálculo dos MFCC na fragmentação é maior para janelas maiores. Isto deve-se ao facto de os MFCC usarem a transformada rápida de Fourier e esta ter um tempo de computação proporcional a $N * Log(N)$ (Johnson and Frigo, 2007). Se N for substituído pelas janelas de 50ms e 1000ms obtêm-se, aproximadamente, 89 e 3000 operações de ponto flutuante, respetivamente, o que significa que para a janela de 1000ms são necessárias cerca de 33 vezes mais operações. Como consequência, tem de existir um balanço entre o tempo gasto no cálculo dos MFCC e o tempo gasto no cálculo do resto da classificação.

Os tempos de execução do Teste 1 e Teste 2, para as janelas que geram melhores resultados, encontram-se representados nas Tabelas B.14 e B.15, respetivamente. No Teste 1, é possível constatar

o referido anteriormente: para janelas muito pequenas o tempo de execução é muito elevado, pois, mesmo que a computação dos MFCC seja mais rápida, existe uma grande quantidade de dados. À medida que as janelas aumentam, o tempo de execução começa a diminuir, mas nos 500ms o tempo volta a subir (tempo de computação dos MFCC aumenta), o que indica que o balanço ideal de tempos de execução da fragmentação e do cálculo dos *log-likelihood* é nos 300ms.

No Teste 2 também é possível obter um *F-score* de 0,875 mas utilizando uma janela de 150ms, o que implica as seguintes desvantagens:

- Na janela de 150ms, como referido, existe o perigo de não ser possível inverter a matriz na classificação;
- Tem um maior tempo de execução do que a janela de 300ms do Teste 1.

Note-se que existiu muito pouco som para efetuar o treino e os testes da classificação, ou seja, em trabalhos futuros, caso exista uma maior quantidade de som para treino e teste, os GMM podem conter um número ótimo de componentes diferente do atual. O tamanho ótimo das janelas utilizadas nos MFCC para a classificação também pode variar, podendo levar a uma alteração dos atuais resultados.

Em suma, devem ser escolhidas as condições do Teste 1 com a janela de 300ms, de forma a ser possível otimizar a classificação em termos de tempo de execução e *F-score*. Não foi possível com nenhuma janela e em nenhum dos testes classificar corretamente a viatura Protect Fire.

4.2 Rede de sensores sem fios

Nesta Secção são apresentados os resultados obtidos utilizando a implementação descrita na Secção 3.2.

Existem diversas variáveis independentes com interesse para o estudo do desempenho da rede de sensores:

- Tempo que a estação aguarda para analisar os dados;
- Tempo de simulação;
- Densidade de nós no terreno;
- Número de nós no terreno;
- Disposição dos nós no terreno;
- Versões desenvolvidas com as tecnologias *IEEE 802.11g* e *IEEE 802.15.4*;
- Existência de agregação;
- Tempo de agregação;
- Distância limite de um nó à viatura, para qual os pacotes são enviados.

Dadas a variáveis independentes, foram introduzidas alterações no código, de forma a conseguir medir o desempenho da rede através das seguintes variáveis dependentes:

- **Perdas** - Para calcular as perdas da rede, dado que se podem fazer testes com e sem agregação, não é de interesse o número de pacotes perdidos, mas sim os identificadores que a estação base não recebeu e deveria ter recebido. Para este fim, os nós inscrevem na estação base o seu identificador, antes de enviarem o pacote, para que seja possível saber se um identificador não foi recebido por perda de pacote;

- **Atrasos** - Para identificar o atraso de um pacote é adicionado a este uma etiqueta, que contém o tempo a que este foi enviado, fazendo assim a diferença na estação base. Há que referir que, no caso de existir agregação, é sempre mantido o menor tempo, ou seja, é sempre calculado o atraso máximo. Dado que no caso de existir agregação é sempre mantido o atraso máximo, é calculado o atraso máximo médio com todos os pacotes, de forma a obter uma média dos atrasos. Para visualizar os intervalos de confiança foi também medido o atraso absoluto, que corresponde ao atraso máximo na receção de um pacote durante toda simulação e o atraso mínimo, que corresponde ao atraso mínimo na receção de um pacote durante toda simulação;

- **Energia** - De forma a estimar a energia gasta pela rede, é necessário saber quantos pacotes foram recebidos e enviados, incluindo as atualizações do protocolo DSDV. São assim incluídos nas camadas físicas, contadores de envio e receção que permitem, utilizando os valores especificados na Tabela 3.1, calcular os valores de energia consumida. É de interesse calcular a energia gasta em cada nó e não a energia total, pois, na aplicação em estudo podem existir nós que nunca vão enviar informação relativa a viaturas, enquanto outros podem enviar e receber bastante informação. Isto leva a que a autonomia necessária seja estimada com os nós que mais consomem e não através do consumo médio. A energia gasta na estação base não é contabilizada pois, pretende-se que no caso real, esta esteja ligada à rede elétrica, ou contenha uma fonte de energia diferente dos nós folha;

- ***Goodput*** - Este parâmetro determina o número de *bytes/s*, relativos ao som, recebidos na estação base. A contabilização dos *bytes* recebidos é feita ao nível da aplicação;

- ***Goodput* caso não existisse processamento nos nós** - Este parâmetro determina o número de *bytes/s*, necessários receber na estação base, de dados relativos ao som, caso não existisse processamento nos nós e fosse necessário enviar os dados relativos à gravação. É verificado o tamanho de um segundo de som no *Matlab*, o número de pacotes necessários enviar e, por fim, são calculados os *bytes/s* que a estação base deveria receber.

Foram então efetuados testes à rede em ambas as versões, sempre orientados a visualizar as medidas de desempenho da rede referidas.

Para que não exista um grande número de simulações, foram fixas algumas das varáveis independentes, tentando sempre manter coerentes as simulações com a realidade.

Foi fixo o tempo que a estação base aguarda para analisar os dados de modo a que a identificação e classificação de viaturas não esteja muito longe do tempo real (10s), o tempo de simulação (2000s), o número de nós (100) e a disposição de nós no terreno, colocando os nós igualmente espaçados entre eles, permitindo assim ao alterar este espaçamento, variar a densidade da rede. A disposição dos nós na rede, encontra-se representada na Figura 4.5.

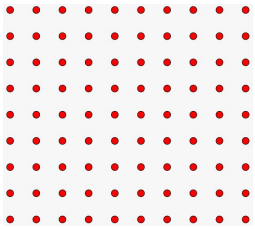


Figura 4.5: Disposição dos nós no terreno, para simulação.

A distância limite de um nó à viatura, que possibilita o envio de pacotes para a estação base, foi fixa com o valor de 300m. Como referido na Secção 3.2.1.5, o ideal seria um modelo de degradação de som dada a distância, mas os sons recolhidos não permitem extrair essa informação. Este problema foi resolvido de forma binária, ou seja, só se a viatura estiver a menos de 300m é que o nó envia, o que é uma limitação atual da implementação.

De forma a conseguir gerar as medidas de desempenho da rede, com tempo de agregação e densidade variável, as simulações foram divididas em dois grupos.

No primeiro grupo é fixo o tempo de agregação e é variada a densidade dos nós no terreno. No segundo grupo é fixa a densidade e é variado o tempo de agregação.

Em ambos os grupos são feitas simulações para as duas versões desenvolvidas, com e sem agregação e, são geradas as medidas de desempenho referidas anteriormente. No segundo grupo, dado que o tempo de agregação é variável, as simulações sem agregação geram apenas valores de referência.

Para cada densidade, versão, com e sem agregação, no primeiro grupo e para cada tempo de agregação, versão, com e sem agregação, no segundo grupo, são executadas dez simulações, sendo que, para cada uma destas existe um trajeto da viatura diferente. Estes trajetos encontram-se representados na Figura 4.6. Foram testadas seis densidades e tempos de agregação diferentes. Em suma, as simulações realizadas para o primeiro grupo têm a estrutura representada na Figura 4.7. A estrutura para as simulações do segundo grupo é em tudo igual à do primeiro grupo, excetuando a variação de densidade que passa a variação de tempo de agregação. As listas dos valores fixos e variáveis para as simulações dos dois grupos encontram-se representadas nas Tabelas 4.2, 4.3 e 4.4.

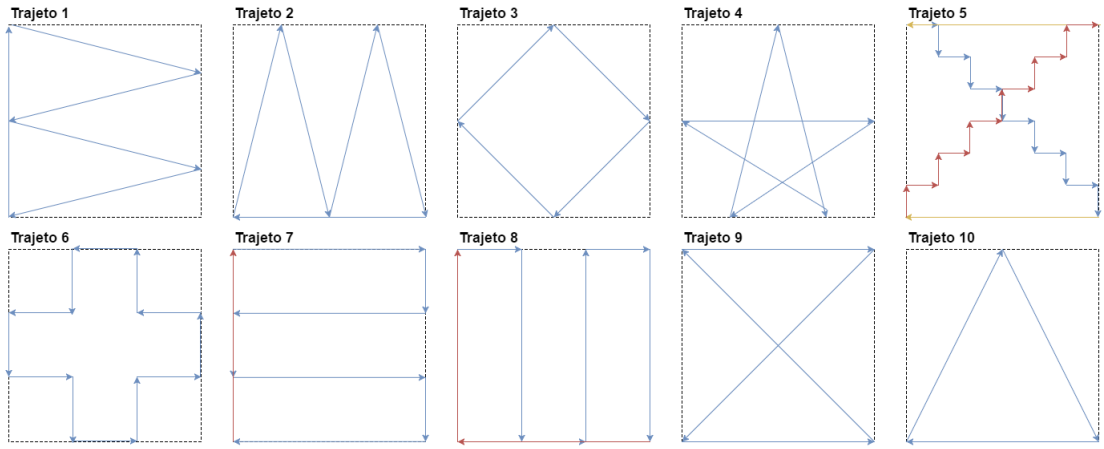


Figura 4.6: Trajetos realizados pela viatura nas simulações.

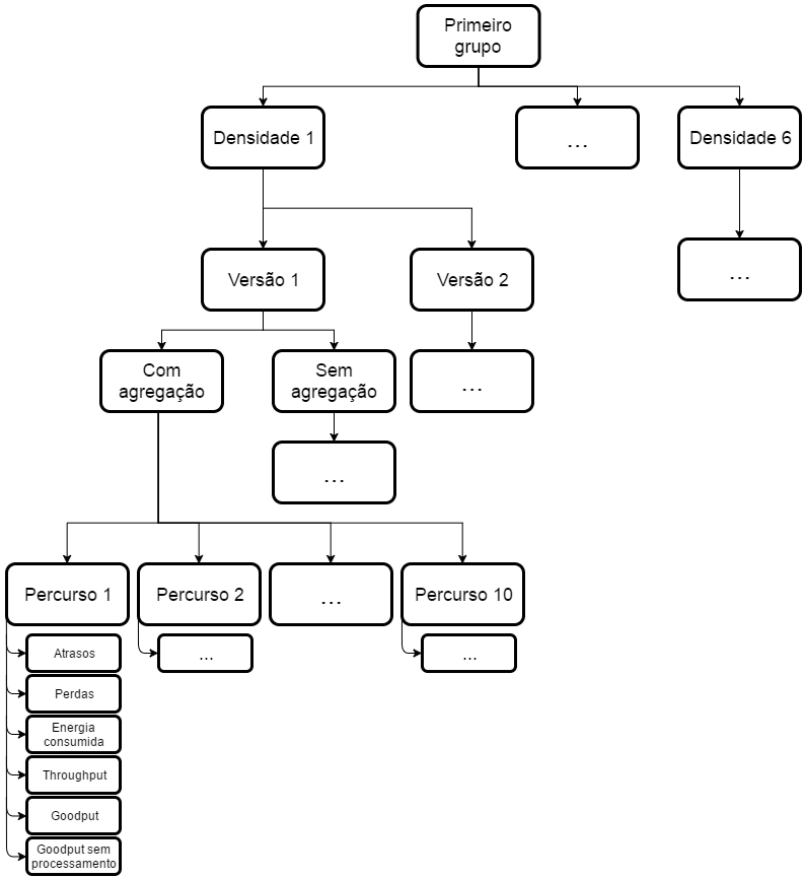


Figura 4.7: Estrutura de simulações para o primeiro grupo

Tabela 4.2: Lista de valores fixos comuns aos dois grupos.

Valores fixos comuns	
Número de nós	100
Disposição de nós para simulação	Igualmente espaçados
Distância limite de um nó à viatura, que possibilita o envio de pacotes para a estação base	300m

Tabela 4.3: Lista de valores fixos e varáveis para o primeiro grupo.

Primeiro grupo						
Valores fixos						
Tempo de agregação	0.2s					
Valores variáveis						
Versão	1	2				
Agregação	Sim	Não				
Densidade da rede (espaçamento entre nós)	150m	125m	100m	75m	50m	25m

Tabela 4.4: Lista de valores fixos e varáveis para o segundo grupo.

Segundo grupo						
Valores fixos						
Densidade da rede (espaçamento entre nós)	50					
Valores variáveis						
Versão	1	2				
Agregação	Sim	Não				
Tempo de agregação	0.2s	0.3s	0.4s	0.5s	0.6s	0.7s

É importante referir que o tempo de agregação fixo escolhido para as simulações do primeiro grupo baseou-se em certos critérios que podem influenciar o desempenho da classificação em tempo real. Se o tempo de agregação for muito elevado, o atraso dos pacotes dos nós até à estação base é muito grande, o que leva a que a classificação não consiga ser feita em tempo real, ou próximo deste. Não é de interesse tático classificar uma viatura muito tempo após ela ter passado naquela área. Importa que tudo seja feito com o mínimo de atraso possível. A agregação permite que existam menos transmissões, logo maior poupança de energia, mas tem de existir um equilíbrio entre o tempo de agregação e a energia consumida na rede, de forma a não existirem atrasos muito elevados. Outro problema, prende-se com a perda de pacotes. Se o tempo de agregação for muito elevado, são agregados mais pacotes e, caso algum destes seja perdido, por algum motivo, existe um grande prejuízo no que respeita aos dados contidos por aquele pacote, o que pode tornar menos provável a classificação correta da viatura. No caso de, em futuros trabalhos ser desenvolvido o rastreio de viaturas, o facto de os dados terem de chegar à estação base o mais perto do tempo real possível é ainda mais importante.

Em suma, é necessário existir um compromisso entre a energia gasta e o atraso introduzido pela agregação. Foram então escolhidos 0.2 segundos de tempo de agregação para efetuar a variação de densidade de nós no terreno, tendo em conta o compromisso referido.

4.2.1 Resultados das simulações do primeiro grupo

Neste grupo de simulações é importante realçar o facto de os nós só enviarem informação se a viatura se encontrar a menos de 300m deles, como pode ser visualizado na Figura 4.8. Isto implica que, para densidades menores, ou seja, maior espaçamento, vão existir menos nós a enviar informação, o que, por sua vez, significa menos congestionamento na rede.

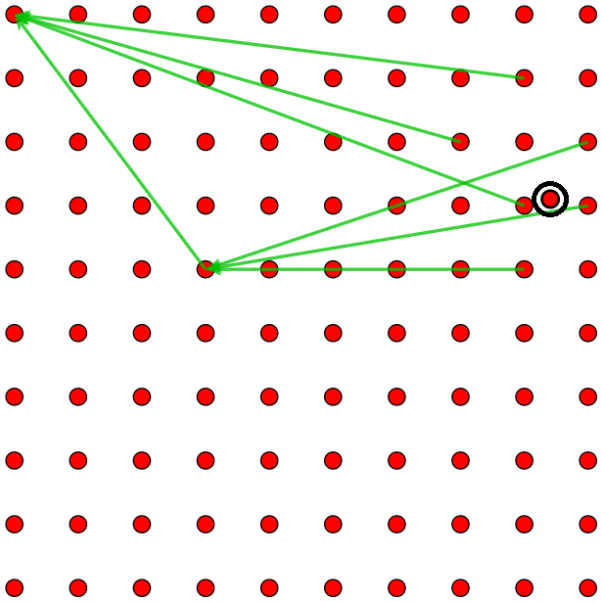
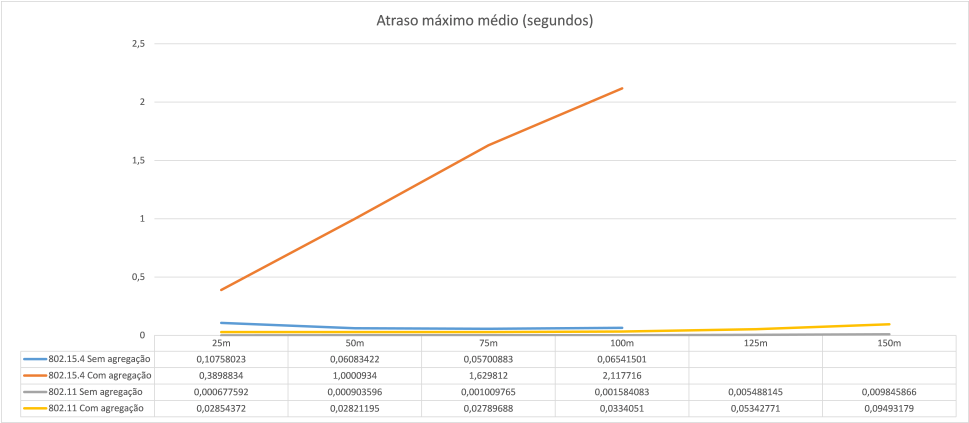


Figura 4.8: Envio de informação pelos nós que se encontram a menos de 300m da viatura.

4.2.1.1 Atrasos

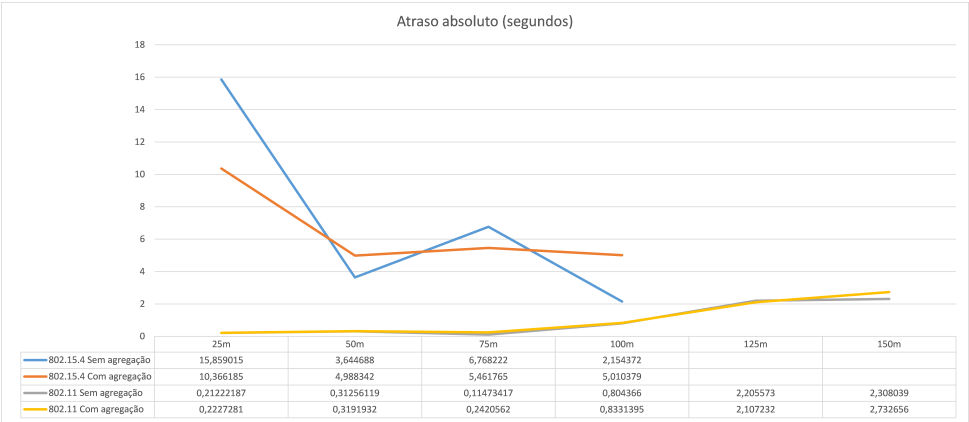
Ao analisar as Figuras 4.9, 4.10 e 4.11, que representam respetivamente, o atraso máximo médio, o atraso absoluto e o atraso mínimo em função da distância, é possível verificar que, como esperado, a agregação introduz um atraso na transferência dos pacotes, pois existe um determinado tempo em que os nós estão à espera de pacotes para poderem ou não agregar. Em ambas as tecnologias, no atraso máximo médio, a versão com agregação tem sempre maiores atrasos. No atraso mínimo, em quase todas as distâncias, os tempos coincidem, pois existem nós que estão ao alcance da estação base e existe um envio direto do pacote, em ambos os casos. No atraso absoluto, na tecnologia *802.15.4*, existe uma alternância nos valores entre a versão com e sem agregação mas estes não têm grande importância pois, apenas permitem estimar o tempo máximo que um pacote poderá demorar a chegar à estação base.

Ao comparar as diferentes tecnologias é visível que, o *802.11g* sem agregação é o que consegue ter um menor atraso máximo médio e mínimo, pois tem uma taxa de transferência de dados de 6Mb/s que comparando com 250kb/s do *802.15.4*, é mais elevada.



	25m	50m	75m	100m	125m	150m
802.15.4 Sem agregação	0,10758023	0,06083422	0,05700883	0,06541501		
802.15.4 Com agregação	0,3898834	1,0000934	1,629812	2,117716		
802.11 Sem agregação	0,000677592	0,000903596	0,001009765	0,001584083	0,005488145	0,009845866
802.11 Com agregação	0,02854372	0,02821195	0,02789688	0,0334051	0,05342771	0,09493179

Figura 4.9: Atraso máximo médio.



	25m	50m	75m	100m	125m	150m
802.15.4 Sem agregação	15,859015	3,644688	6,768222	2,154372		
802.15.4 Com agregação	10,366185	4,988342	5,461765	5,010379		
802.11 Sem agregação	0,21222187	0,31256119	0,11473417	0,804366	2,205573	2,308039
802.11 Com agregação	0,2227281	0,3191932	0,2420562	0,8331395	2,107232	2,732656

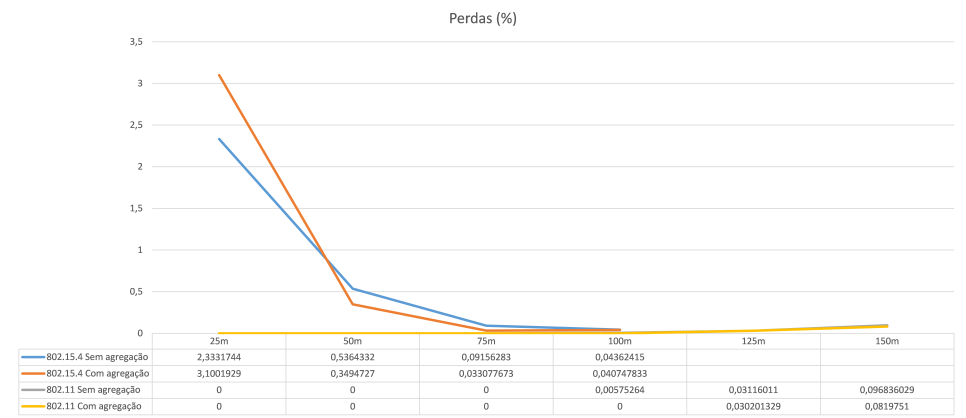
Figura 4.10: Atraso absoluto.

	25m	50m	75m	100m	125m	150m
802.15.4 Sem agregação	0,00227208	0,00227208	0,00227208	0,00227208		
802.15.4 Com agregação	0,00227208	0,00227208	0,00230425	0,031718597		
802.11 Sem agregação	0,000122083	0,000122166	0,00012225	0,000122366	0,000122527	0,000122733
802.11 Com agregação	0,000122083	0,000122166	0,00012225	0,000122366	0,00012251	0,000122733

Figura 4.11: Atraso mínimo.

4.2.1.2 Perdas

Relativamente a perdas, de acordo com a Figura 4.12, é visível que na tecnologia *802.11g* estas aumentam com a distância, pois existe uma boa taxa de transferência de dados e as perdas devem-se a interferências causadas pela distância. No caso da tecnologia *802.15.4*, a taxa de transferência de dados é mais baixa e, quando as distâncias são mais curtas, existe mais congestionamento da rede, logo, uma tendência para existirem mais perdas. À medida que a distância aumenta, as perdas no *802.15.4* diminuem, até ao espaçamento de 100m, pois a partir deste já não existe comunicação. Desta forma, o *802.11g* permite espaçamentos maiores para os quais o *802.15.4* não consegue comunicar e sempre com menores perdas.



	25m	50m	75m	100m	125m	150m
802.15.4 Sem agregação	2,3331744	0,5364332	0,09156283	0,04362415		
802.15.4 Com agregação	3,1001929	0,3494727	0,033077673	0,040747833		
802.11 Sem agregação	0	0	0	0,00575264	0,03116011	0,096836029
802.11 Com agregação	0	0	0	0	0,030201329	0,0819751

Figura 4.12: Perdas associadas aos testes do primeiro grupo.

4.2.1.3 *Goodput*

A Figura 4.13, que representa o *Goodput*, indica que a tecnologia *802.11g* sem agregação é a que contém menos *Goodput* em todas as distâncias, mas pode verificar-se que a versão com agregação obtém resultados semelhantes à que não tem agregação.

Era de esperar que a agregação levasse a um menor *Goodput* mas note-se que, para espaçamentos maiores quando existem poucos nós a enviar, dada a distância à viatura, a agregação só vai introduzir atrasos e levar a um maior envio de dados (pois é enviado o identificador do nó que agrega), do que se o pacote fosse enviado diretamente para a estação base.

Pretendeu-se estimar qual seria o *Goodput* na estação base, caso não fosse realizado processamento nos nós ou caso fossem apenas calculados os MFCC e enviados estes. Para este fim foram realizados cálculos que são meras estimativas, mas que permitem efetuar uma comparação.

Utilizando o número de pacotes enviados por cada nó e os *bytes* que ocuparia um trecho de som de um segundo, é possível saber o *Goodput* necessário.

Analisando os trechos de som utilizados para classificação, verificou-se que, como a amostragem é de 16000Hz e um tipo de dados `float` utiliza 4 *bytes*, seriam precisos 64000 *bytes* para enviar diretamente um trecho de som para a estação base, sem processamento.

Na Tabela 4.5 encontram-se representados os valores disponíveis para transporte de dados no

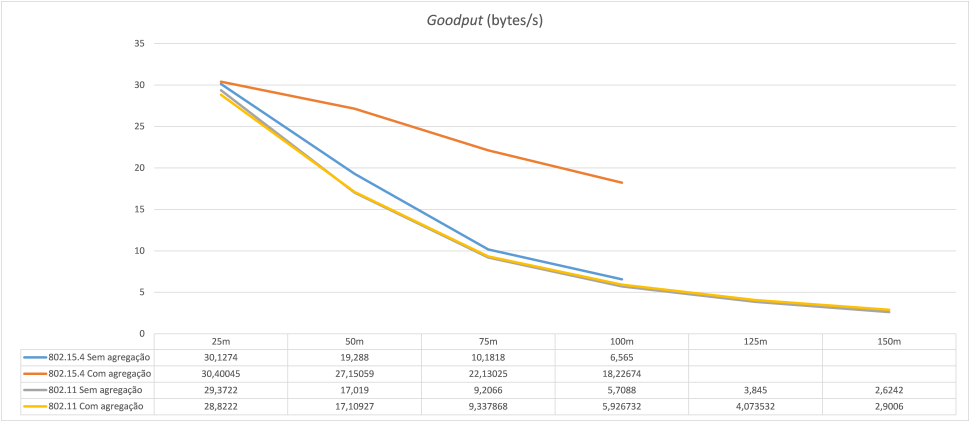


Figura 4.13: *Goodput* associado aos testes do primeiro grupo.

802.11g e *802.15.4*. Na Tabela 4.6 encontra-se representado o número de pacotes necessários enviar, por cada trecho de som de um segundo.

Tabela 4.5: Tamanhos disponíveis para dados das tecnologias *802.11g* e *802.15.4*.

Tamanho dos pacotes 802.11g	2304	**Tamanho dos pacotes 802.15.4**	127
IPv4 header	20	IPv6 header	40
FCS	2	FCS	2
UDP	8	UDP	8
MAC	23	MAC	23
Tamanho disponível para dados	2251	Tamanho disponível para dados	54

Para enviar os MFCC, é necessário contabilizar que se estão a utilizar oito coeficientes cepstrais, com janelas de 300ms sobrepostas em 50%. Sabendo que o som tem a duração de um segundo e que um `float` ocupa 4 *bytes* é possível obter os resultados associados aos MFCC representados na Tabela 4.6.

Tabela 4.6: Número de pacotes necessários para envio de um trecho de som de um segundo, para as tecnologias *802.11g* e *802.15.4*.

Envio direto de som	
802.11g	29
802.15.4	1186
Envio de MFCC	
802.11g	1
802.15.4	8

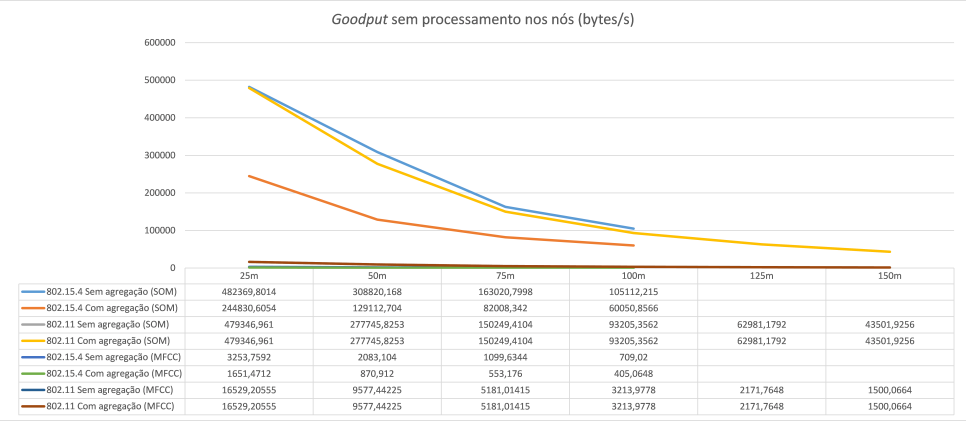
Analisando a Figura 4.14, indicativa do *Goodput* necessário na estação base, caso não existisse processamento nos nós, verifica-se que é recomendável efetuar processamento, uma vez que, comparativamente com o *Goodput* caso exista processamento, a diferença é abrupta. No caso do *802.15.4* sem agregação, teríamos aproximadamente 30,1274 *bytes/s* com processamento, 482369,8014 *bytes/s* sem processamento e 3253,7592 *bytes/s* enviando apenas os MFCC.

Enviando os MFCC consegue-se obter um *Goodput* bastante mais baixo do que enviar só o som, e podia ajudar a que a classificação fosse feita de uma melhor forma na estação base mas, já que é realizado o processamento para calcular os MFCC, que é significativo, pode ser realizada logo a

classificação no nó, o que conduz a um *Goodput* na estação base muito mais baixo.

No entanto, como os pacotes da tecnologia *802.11g* são de maior dimensão e apenas é necessário o envio de um pacote no caso dos MFCC, é possível que em trabalhos futuros esta modalidade seja testada e exequível.

Existe também a possibilidade, em trabalhos futuros, de a recolha de som ser feita a uma amostragem mais baixa e ser utilizado um tipo de dados que ocupe menos *bytes*, permitindo assim que o número de pacotes a enviar seja mais pequeno, de forma a tornar esta modalidade praticável. Na aplicação em estudo não foi testada a classificação com trechos de som mais pequenos, que seria de certa forma equivalente a ter uma amostragem mais baixa e não é possível saber o comportamento do classificador. É importante referir que trechos de som demasiado curtos podem não conter informação suficiente para a classificação e podem levar a um resultado errado.



	25m	50m	75m	100m	125m	150m
802.15.4 Sem agregação (SOM)	482369,8014	308820,168	163020,7998	105112,215		
802.15.4 Com agregação (SOM)	244830,6054	129112,704	82008,342	60050,8566		
802.11 Sem agregação (SOM)	479346,961	277745,8253	150249,4104	93205,3562	62981,1792	43501,9256
802.11 Com agregação (SOM)	479346,961	277745,8253	150249,4104	93205,3562	62981,1792	43501,9256
802.15.4 Sem agregação (MFCC)	3253,7592	2083,104	1099,6344	709,02		
802.15.4 Com agregação (MFCC)	1651,4712	870,912	553,176	405,0648		
802.11 Sem agregação (MFCC)	16529,20555	9577,44225	5181,01415	3213,9778	2171,7648	1500,0664
802.11 Com agregação (MFCC)	16529,20555	9577,44225	5181,01415	3213,9778	2171,7648	1500,0664

Figura 4.14: *Goodput* necessário sem processamento de dados, associado aos testes do primeiro grupo.

4.2.1.4 Energia consumida pela rede

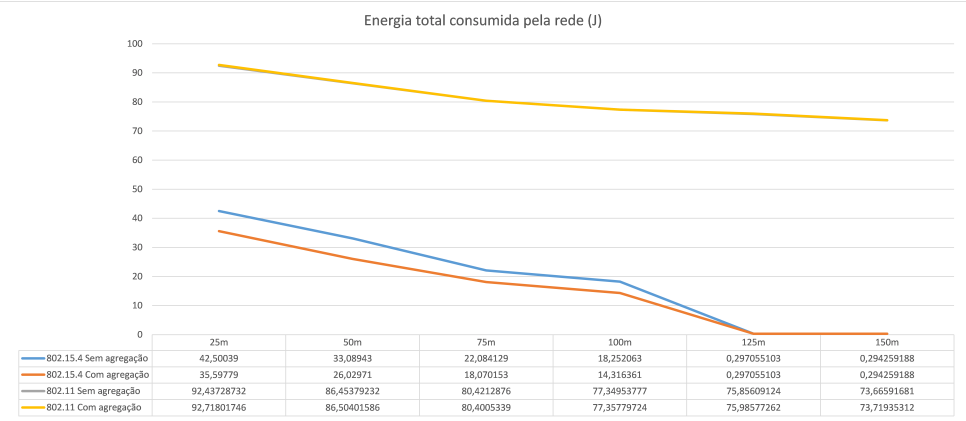
Ao analisar a Figura 4.15, verifica-se que energia consumida pela rede ao usar a tecnologia *802.15.4*, é muito mais baixa pois, o consumo nos envios, receções e *standby*, é muito mais baixo que na tecnologia *802.11g*. Isto leva a que a autonomia da rede seja muito maior, permitindo o controlo daquela área durante mais tempo.

A tecnologia *802.11g*, tem um maior consumo de energia, mas ao contrário da *802.15.4*, permite comunicações para distâncias superiores a 100m.

A propósito da comparação das tecnologias quando estas utilizam agregação de dados, verifica-se que no caso da *802.15.4*, a agregação reduz em todas as distâncias o consumo de energia, pelo que seria a melhor opção, não tendo em conta os atrasos. No caso da tecnologia *802.11g*, a agregação não produz uma diferença significativa, este resultado é apoiado pelo facto do consumo do *standby* nesta tecnologia ser muito alto, o que torna por vezes que as operações de envio e receção, numa simulação com pouca duração sejam menos significativas.

O consumo de energia diminui com o aumento da distância pois, como a rede está menos densa, não existem tantos nós que estão a menos de 300m da viatura, o que conduz a um menor envio e receção de pacotes que se traduz num menor consumo de energia.

Em suma, a tecnologia *802.15.4* com agregação é a que produz o melhor resultado em termos de consumo de energia total da rede.

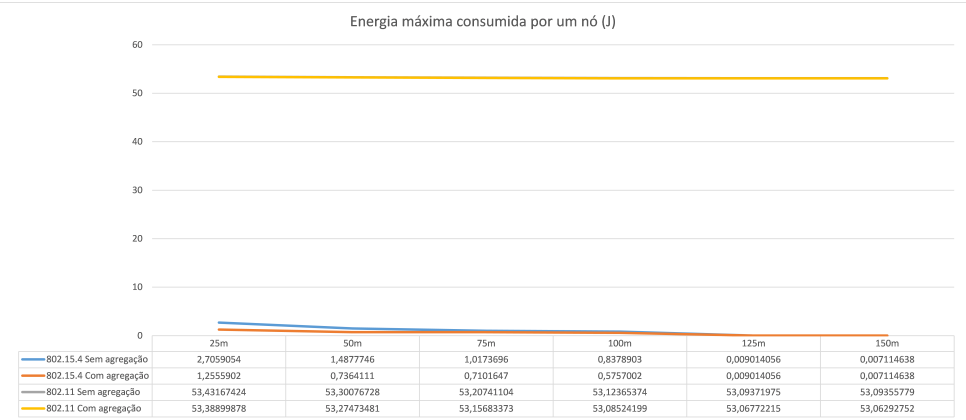


	25m	50m	75m	100m	125m	150m
802.15.4 Sem agregação	42,50039	33,08943	22,084129	18,252063	0,297055103	0,294259188
802.15.4 Com agregação	35,59779	26,02971	18,070153	14,316361	0,297055103	0,294259188
802.11 Sem agregação	92,43728732	86,45379232	80,4212876	77,34953777	75,85609124	73,66591681
802.11 Com agregação	92,71801746	86,50401586	80,4005339	77,35779724	75,98577262	73,71935312

Figura 4.15: Energia consumida pela rede, associada aos testes do primeiro grupo.

As conclusões que resultam da análise da Figura 4.16, são semelhantes às referidas anteriormente para a figura 4.15 mas, analisar o consumo de energia num nó permite efetuar uma melhor estimativa de quanta energia seria necessária por nó, para a rede funcionar na sua plenitude. Isto é, qual seria a energia que a bateria necessitaria de conter para alimentar um nó durante 2000 segundos.

A diferença entre a tecnologia *802.11g* e *802.15.4*, é imensa, pois nesta última a energia necessária é mínima quando comparada com a primeira. Sendo, assim, necessário recorrer a baterias muito menores, com a vantagem de ser muito mais fácil recarregá-las com mecanismos de obtenção de energia do meio ambiente.



	25m	50m	75m	100m	125m	150m
802.15.4 Sem agregação	2,7059054	1,4877746	1,0173696	0,8378903	0,009014056	0,007114638
802.15.4 Com agregação	1,2555902	0,7364111	0,7101647	0,5757002	0,009014056	0,007114638
802.11 Sem agregação	53,43167424	53,30076728	53,20741104	53,12365374	53,09371975	53,09355779
802.11 Com agregação	53,38899878	53,27473481	53,15683373	53,08524199	53,06772215	53,06292752

Figura 4.16: Energia consumida por um nó, associada aos testes do primeiro grupo.

4.2.2 Resultados das simulações do segundo grupo

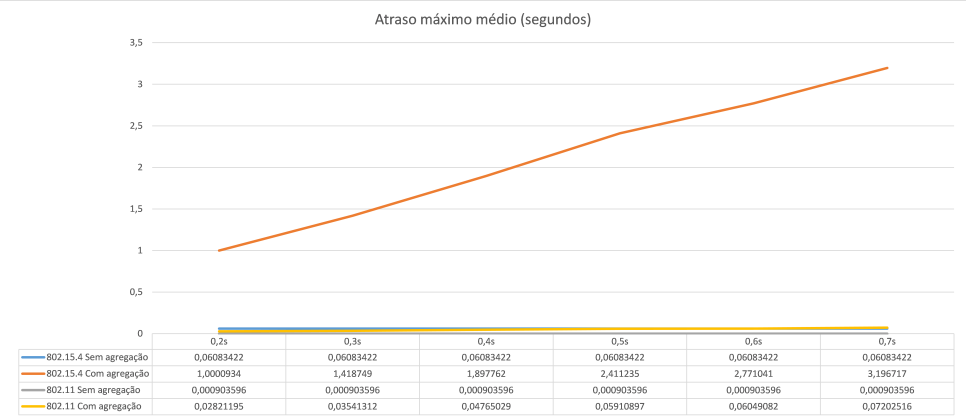
Tal como no primeiro grupo de simulações, os nós do segundo grupo só enviam informação se a viatura se encontrar a menos de 300m deles. Dado que a densidade é fixa e que na tecnologia *802.11g* o alcance dos nós é maior, vão existir claramente menos saltos do que na tecnologia *802.15.4*.

4.2.2.1 Atrasos

Ao analisar as Figuras 4.17, 4.18 e 4.19, que representam respetivamente, o atraso máximo médio, o atraso absoluto e o atraso mínimo em função da distância, é possível verificar que, como no primeiro grupo, a agregação introduz um atraso na transferência dos pacotes, pois existe um determinado tempo em que os nós estão à espera de pacotes para poderem ou não agregar. Em ambas as tecnologias, a versão com agregação tem sempre maiores atrasos, a não ser no atraso mínimo, que em ambas as tecnologias, as versões sem e com agregação, contêm tempos que coincidem, pois existem nós que estão ao alcance da estação base e existe um envio direto do pacote.

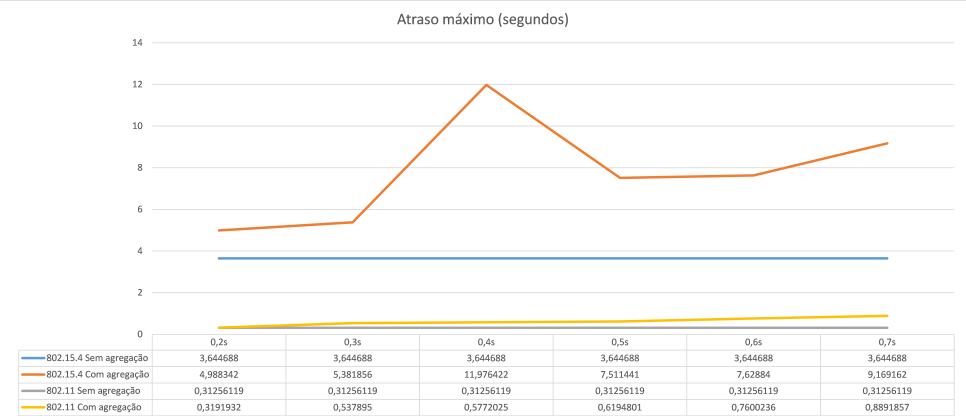
O que se pode verificar claramente é que o atraso máximo médio da tecnologia *802.15.4*, evolui muito mais com o aumento do tempo de agregação do que o atraso na tecnologia *802.11g*. Esta facto é apoiado pelo número de saltos que os pacotes vão sofrer até chegar à estação base. Dado que na tecnologia *802.11g* os alcances dos nós são maiores, existem menos saltos, ou mesmo nenhum, de forma a que o atraso introduzido pela agregação é pouco ou nenhum.

Em trabalhos futuros pode ser feita uma análise só para a tecnologia *802.11g*, com distâncias maiores, de forma a verificar que tipo de atrasos são introduzidos pela a agregação.



	0,2s	0,3s	0,4s	0,5s	0,6s	0,7s
802.15.4 Sem agregação	0,06083422	0,06083422	0,06083422	0,06083422	0,06083422	0,06083422
802.15.4 Com agregação	1,0000934	1,418749	1,897762	2,411235	2,771041	3,196717
802.11 Sem agregação	0,000903596	0,000903596	0,000903596	0,000903596	0,000903596	0,000903596
802.11 Com agregação	0,02821195	0,03541312	0,04765029	0,05910897	0,06049082	0,07202516

Figura 4.17: Atraso máximo médio.



	0,2s	0,3s	0,4s	0,5s	0,6s	0,7s
802.15.4 Sem agregação	3,644688	3,644688	3,644688	3,644688	3,644688	3,644688
802.15.4 Com agregação	4,988342	5,381856	11,976422	7,511441	7,62884	9,169162
802.11 Sem agregação	0,31256119	0,31256119	0,31256119	0,31256119	0,31256119	0,31256119
802.11 Com agregação	0,3191932	0,537895	0,5772025	0,6194801	0,7600236	0,8891857

Figura 4.18: Atraso absoluto.

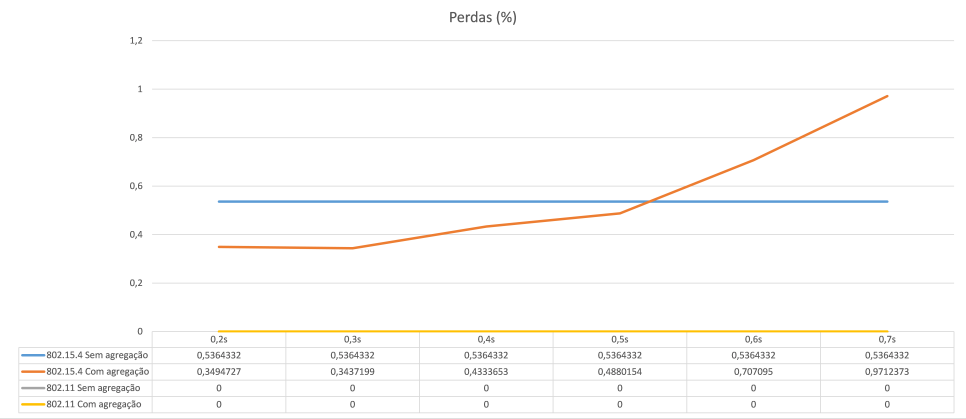
	0,2s	0,3s	0,4s	0,5s	0,6s	0,7s
802.15.4 Sem agregação	0,00227217	0,00227217	0,00227217	0,00227217	0,00227217	0,00227217
802.15.4 Com agregação	0,00227217	0,00227217	0,00227217	0,00227217	0,00227217	0,00227217
802.11 Sem agregação	0,000122166	0,000122166	0,000122166	0,000122166	0,000122166	0,000122166
802.11 Com agregação	0,000122166	0,000122166	0,000122166	0,000122166	0,000122166	0,000122166

Figura 4.19: Atraso mínimo.

4.2.2.2 Perdas

Analisando as perdas, de acordo com a Figura 4.20, é visível que, na tecnologia *802.11g*, não existem perdas, tal como era esperado após as conclusões referidas na Secção 4.2.1.2, para 50m de distância.

No caso da tecnologia *802.15.4*, tal como no primeiro grupo, existem mais perdas e verifica-se que, ao aumentar o tempo de agregação as perdas só ultrapassam a versão com agregação a partir dos 0,5s. Isto deve-se ao facto de agregação introduzir um atraso tal, que não permite os dados chegarem à estação base antes dos 10s estipulados para fazer a contabilização dos dados recebidos. Ou seja, como referido anteriormente, uma agregação com um tempo muito elevado, leva a atrasos muito elevados, o que leva a que os dados que cheguem à estação base já não sejam contabilizados e, sejam considerados perdas, pois é de interesse tático a identificação de viaturas o mais próximo possível do tempo real.



	0,2s	0,3s	0,4s	0,5s	0,6s	0,7s
802.15.4 Sem agregação	0,5364332	0,5364332	0,5364332	0,5364332	0,5364332	0,5364332
802.15.4 Com agregação	0,3494727	0,3437199	0,4333653	0,4880154	0,707095	0,9712373
802.11 Sem agregação	0	0	0	0	0	0
802.11 Com agregação	0	0	0	0	0	0

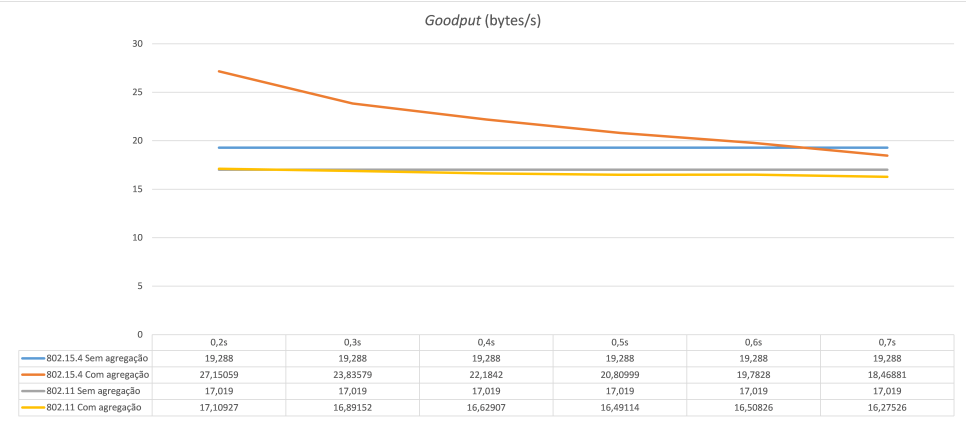
Figura 4.20: Perdas associadas aos testes do segundo grupo.

4.2.2.3 *Goodput*

A Figura 4.21, que representa o *Goodput*, mostra que a agregação na tecnologia *802.11g*, em todos os tempos, gera sempre menor *Goodput* do que a opção sem agregação. O mesmo não acontece com a tecnologia *802.15.4*, pelas mesmas razões referidas na Secção 4.2.1.3.

Na comparação das duas tecnologias, verifica-se que a *802.11g* gera um menor *Goodput* que a

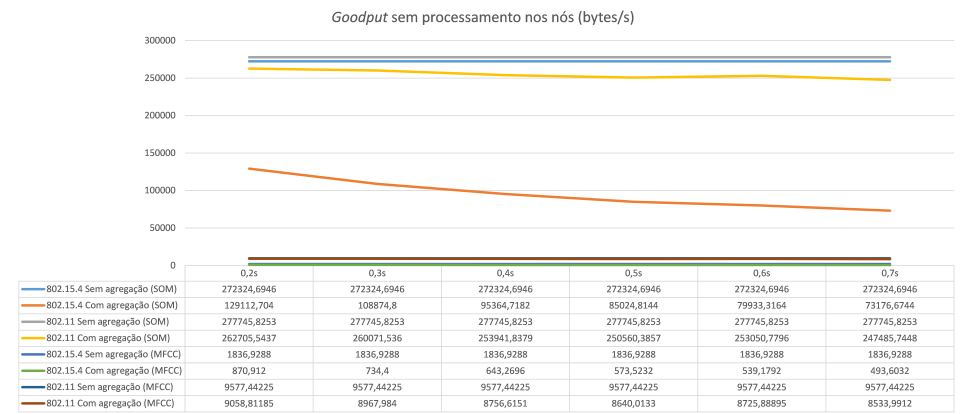
802.15.4 mas, este diminui muito mais com aumento de tempo de agregação na *802.15.4*. Dado que o tempo de agregação aumenta, são agregados mais dados e por isso, são enviados menos pacotes o que resulta numa diminuição do *Goodput*.



	0,2s	0,3s	0,4s	0,5s	0,6s	0,7s
802.15.4 Sem agregação	19,288	19,288	19,288	19,288	19,288	19,288
802.15.4 Com agregação	27,15059	23,83579	22,1842	20,80999	19,7828	18,46881
802.11 Sem agregação	17,019	17,019	17,019	17,019	17,019	17,019
802.11 Com agregação	17,10927	16,89152	16,62907	16,49114	16,50826	16,27526

Figura 4.21: *Goodput* associado aos testes do segundo grupo.

Todos os resultados dos cálculos efetuados na Secção 4.2.1.3, foram utilizados nesta secção, gerando assim a Figura 4.22. As conclusões que podem ser retiradas desta figura são as mesmas da Secção 4.2.1.3 ou seja, em comparação com o *Goodput* obtido com o processamento, o *Goodput* gerado se não existir processamento ou processamento parcial é demasiado elevado, podendo tornar esta modalidade impraticável dadas as condições atuais.



	0,2s	0,3s	0,4s	0,5s	0,6s	0,7s
802.15.4 Sem agregação (SOM)	272324,6946	272324,6946	272324,6946	272324,6946	272324,6946	272324,6946
802.15.4 Com agregação (SOM)	129112,704	108874,8	95364,7182	85024,8144	79933,3164	73176,6744
802.11 Sem agregação (SOM)	277745,8253	277745,8253	277745,8253	277745,8253	277745,8253	277745,8253
802.11 Com agregação (SOM)	262705,5437	260071,536	253941,8379	250560,3857	253050,7796	247485,7448
802.15.4 Sem agregação (MFCC)	1836,9288	1836,9288	1836,9288	1836,9288	1836,9288	1836,9288
802.15.4 Com agregação (MFCC)	870,912	734,4	643,2696	573,5232	539,1792	493,6032
802.11 Sem agregação (MFCC)	9577,44225	9577,44225	9577,44225	9577,44225	9577,44225	9577,44225
802.11 Com agregação (MFCC)	9058,81185	8967,984	8756,6151	8640,0133	8725,88895	8533,9912

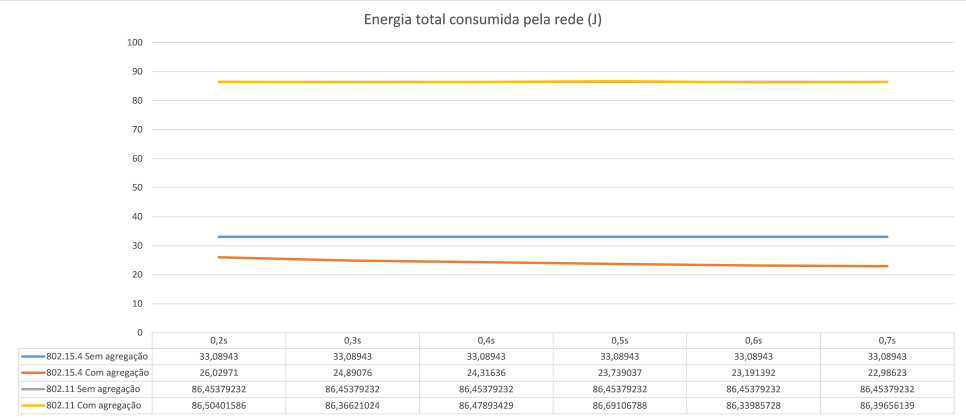
Figura 4.22: *Goodput* necessário sem processamento de dados, associado aos testes do segundo grupo.

4.2.2.4 Energia consumida pela rede

Dada a Figura 4.23 verifica-se que, tal como na Secção 4.2.1.4, a energia consumida pela tecnologia *802.15.4* é muito menor que a consumida pela *802.11g* e que a agregação conduz a uma poupança de energia, pelas mesmas razões referidas.

Adicionalmente, nesta figura, é possível verificar que, ao aumentar o tempo de agregação na tecnologia *802.15.4*, existe uma diminuição do consumo de energia, pois existe um menor número de pacotes a ser enviados e recebidos. Na tecnologia *802.11g* não existe uma diminuição gradual de

energia ao aumentar o tempo de agregação, pois, como referido anteriormente, dado que os alcances dos nós são maiores vão existir menos saltos e a agregação não se vai fazer notar da mesma forma que faz na tecnologia *802.15.4*. No que toca a consumo de energia, a melhor opção seria a tecnologia *802.15.4* com agregação.



	0,2s	0,3s	0,4s	0,5s	0,6s	0,7s
802.15.4 Sem agregação	33,08943	33,08943	33,08943	33,08943	33,08943	33,08943
802.15.4 Com agregação	26,02971	24,89076	24,31636	23,739037	23,191392	22,98623
802.11 Sem agregação	86,45379232	86,45379232	86,45379232	86,45379232	86,45379232	86,45379232
802.11 Com agregação	86,50401586	86,36621024	86,47893429	86,69106788	86,33985728	86,39656139

Figura 4.23: Energia consumida pela rede, associada aos testes do segundo grupo.

Analisando a Figura 4.24, que em termos de relação entre tecnologias é em tudo semelhante à Figura 4.23, é possível retirar as mesmas conclusões da Secção 4.2.1.4. A energia necessária um nó é muito menor na tecnologia *802.15.4*, tornando-se ainda menor caso exista agregação, o que conduz às vantagens também mencionadas na Secção 4.2.1.4.

	0,2s	0,3s	0,4s	0,5s	0,6s	0,7s
802.15.4 Sem agregação	1,4877746	1,4877746	1,4877746	1,4877746	1,4877746	1,4877746
802.15.4 Com agregação	0,7364111	0,7118553	0,662559	0,5913759	0,5946784	0,5434233
802.11 Sem agregação	53,27276152	53,27276152	53,27276152	53,27276152	53,27276152	53,27276152
802.11 Com agregação	53,27473481	53,21909821	53,25575233	53,24669995	53,24531989	53,24492218

Figura 4.24: Energia consumida por um nó, associada aos testes do segundo grupo.

4.2.3 Análise dos resultados produzidos pelos dois grupos

Em suma, as simulações do primeiro grupo, permitem verificar que a agregação introduz sempre atrasos, mas por outro lado, leva a uma poupança de energia. Quando se comparam as duas tecnologias, verifica-se que a *802.11b* tem um melhor comportamento no que toca a perdas e permite espaçamentos superiores a 100m que como consequência leva a que sejam necessários menos nós para cobrir uma maior área. A tecnologia *802.15.4* é a que conduz a um menor consumo de energia, ainda mais quando contém agregação, mas nunca consegue ter 0% de perdas e cobrir áreas tão grandes coma a tecnologia

802.11g.

As simulações do segundo grupo permitem complementar estas conclusões no sentido em que o aumento do tempo de agregação leva a um menor *Goodput* e a um menor consumo de energia, mas conduz sempre a mais perdas e a um maior atraso na entrega dos pacotes, o que contribui para que existam pacotes a serem entregues fora do tempo estipulado.

Na escolha da densidade dos nós e do tempo de agregação existe sempre um compromisso entre atrasos, alcance da rede e consumo de energia. No caso em estudo, pretende-se que a rede tenha alguma autonomia com alguma margem para perdas abaixo do 1%, desde que os pacotes não tenham muito atraso, para que a identificação ocorra o mais perto possível do tempo real. Com estas restrições e dada a diferença abrupta dos consumos de energia das duas tecnologias, a tecnologia *802.15.4* com agregação seria a mais indicada ao caso em estudo. A não ser que o objetivo seja abranger áreas demasiado grandes com poucos nós, por necessidade tática e, nesse caso, a tecnologia *802.11g* seria a mais indicada.

Capítulo 5

Conclusões e trabalho futuro

Este capítulo tem o objetivo de realizar um sumário do trabalho realizado nesta dissertação, contendo o objetivo proposto, as metodologias usadas para alcançar este objetivo, os resultados obtidos após os testes realizados, a indicação das limitações deste sistema, bem como propostas para trabalhos futuros.

O objetivo proposto foi desenvolver uma rede de sensores sem fios em ambiente militar, cuja finalidade é a deteção e classificação de veículos militares, em contexto operacional. Para este fim, foram utilizadas ferramentas para analisar diferentes componentes do som como a análise em frequência (MFCC) e análise do ritmo (BPM). Estas ferramentas permitiram gerar dados utilizados para o treino dos GMM.

Os GMM, foram utilizados para representar dados provenientes de diversas viaturas das Forças Armadas Portuguesas, levando assim à criação de uma base de dados. Com esta base dados, torna-se possível efetuar a classificação de trechos de som de viaturas, de forma a identificar qual a correta. Posteriormente, os dados resultantes desta classificação são enviados para uma estação base onde os dados são analisados. Para este fim, foi desenvolvida uma rede de sensores sem fios com o objetivo de encaminhar os pacotes até à estação base utilizando o protocolo DSDV. De forma a reduzir o consumo de energia desta rede, foram implementadas algumas modificações a este protocolo, como por exemplo, agregação de dados. Foram testadas duas tecnologias diferentes, a *802.15.4* e a *802.11g*, de forma a ser possível verificar qual a que melhor se adequaria à aplicação em estudo.

Ao iniciar o desenvolvimento desta aplicação era esperado que, ao estudar múltiplas componentes do som, a classificação tivesse um melhor desempenho. Após a análise de resultados, verifica-se que existem componentes do som, neste caso as BPM, que não produzem uma boa representação deste, mas, outros tipos de som que não sejam viaturas poderão eventualmente ser bem representados com esta análise. Os MFCC, individualmente, mostraram ser a melhor forma de analisar os sons, de modo a conseguir a melhor classificação com os GMM, sendo que permitiu obter um *F-score* de 0,875.

Quando se desenvolveu o processo de classificação, não era esperado que o tempo de execução fosse uma condicionante, dado o poder de processamento que existe nos dias de hoje. No entanto, verificou-se que o melhor tempo de execução associado ao *F-score* de 0,875 é de cerca de 57s. Este valor não é real, pois para as diferentes viaturas os 15% utilizados para teste podem representar trechos de som superiores a um segundo, o que aumenta em muito o tempo de classificação. Esta limitação

pode condicionar a identificação de viaturas em tempo real.

No envio dos pacotes que contêm a classificação, era esperado que a tecnologia *802.15.4*, juntamente com agregação, proporcionasse um menor consumo de energia, o que levaria a uma maior autonomia da rede. Este facto foi comprovado pelas simulações realizadas, a diferentes distâncias e com diferentes tempos de agregação.

Verificou-se que, para espaçamentos entre nós mais pequenos, a tecnologia *802.15.4* se comporta de forma geral melhor que a *802.11g*. É importante referir que a tecnologia *802.11g* conduz a menores atrasos e perdas e que essa diferença pode ser significativa, mesmo possuindo um elevado consumo de energia. Em cenários em que seja necessário abranger uma área muito grande, utilizando poucos nós, pode-se optar pela tecnologia *802.11g*, dado que esta permite maiores alcances.

O sistema desenvolvido contém algumas limitações, sendo que muitas podem ser colmatadas em trabalhos futuros. O som recolhido para criar a base de dados não foi suficiente. Seriam necessárias horas de gravação para cada viatura em diferentes ambientes, pelo que se recomenda o desenvolvimento de uma base de dados mais consistente. Ao desenvolver esta base dados, torna-se também importante, para aproximar as simulações da realidade, desenvolver um modelo de degradação de som com a distância. A informação para este modelo pode ser extraída dos sons captados para a base de dados, tentando sempre registar a distância a que se encontram as viaturas durante as gravações, mantendo sempre o ganho de entrada do microfone constante.

O microfone utilizado para as gravações é em tudo melhor do que os microfones que um nó vai conter. Por este motivo, seria importante que fossem realizados testes com trechos de som captados por microfones passíveis de serem utilizados em nós. Como se verificou, o tempo de execução é elevado, em parte devido ao número de amostras e, além disso, caso se pretenda enviar os dados do som sem efetuar processamento nos nós, pode ser interessante estudar a possibilidade de captar som com uma taxa de amostragem inferior a 16000Hz. O sistema desenvolvido só está preparado para identificar uma viatura diferente de cada vez, isto é, se existirem duas viaturas ou mais a serem gravadas em simultâneo, não se sabe qual o comportamento do sistema. Torna-se necessário o desenvolvimento de um mecanismo que possa ultrapassar este problema.

Inicialmente foi pretendido desenvolver o rastreio de movimento de viaturas, mas este não foi implementado. Em trabalhos futuros é possível acrescentar esta valência à rede desenvolvida, tendo sempre em atenção os atrasos introduzidos pela classificação e pela rede no envio dos pacotes. A Secção 2.2.4 é um bom ponto de partida para a implementação do rastreio de movimento.

Na rede de sensores sem fios, apenas foram testadas duas tecnologias na camada física (*802.15.4* e *802.11g*). É possível que esta rede possa obter desempenhos melhores para outras tecnologias, ou com outras versões do *IEEE 802.11*, sendo que é aconselhável em trabalhos futuros o teste de outras tecnologias.

De uma maneira geral, os resultados obtidos na realização desta dissertação, foram de encontro aos esperados e considera-se que os objetivos estipulados foram alcançados, sendo que ainda existem algumas limitações neste sistema que poderão ser supridas em trabalhos futuros.

Bibliografia

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Bogert, B., Healy, M., and Tukey, J. (1963). The quefrency alanysis of time series for echoes: Cepstrum, Pseudo-Autocovariance, Cross-Cepstrum and Saphe Cracking. In *Proc. Symp. on Time Series Analysis*, pages 209–243.

Chris Townsend, S. A. (2004). Wireless sensor networks: Principles and applications. *MicroStrain, Inc*, pages 439–449.

CMUSphinx (2016). Cmusphinx. `http://cmusphinx.sourceforge.net/wiki/faq#qwhat_speech_feature_type_does_cmusphinx_use_and_what_do_they_represent`. Consultado em 11 de Julho de 2016.

de Almeida, V. H. D. et al. (2009). A aquisição de objectivos e a artilharia na vanguarda da tecnologia. `http://www.exercito.pt/sites/EPA/Publicacoes/Documents/Boletim2009-web.pdf`. Consultado em 19 de Setembro de 2016.

Ellis, D. P. W. (2007). Beat tracking by dynamic programming. *J. New Music Research*, 2007:51–60.

Embedded, V. (2013). Vnt9485be0cw data sheet. `http://www.mouser.com/ds/2/765/VIA_VNT9485_datasheet_v130306-916691.pdf`. Consultado em 21 de Setembro de 2016.

Goutte, C. and Gaussier, E. (2005). A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *in: Proceedings of the 27th European Conference on Information Retrieval*, pages 345–359.

He, G. (2002). Destination-sequenced distance vector (dsdv) protocol.

Hofmann, P., An, C., Loyola, L., and Aad, I. (2007). Analysis of udp, tcp and voice performance in ieee 802.11 b multihop networks. In *13th European Wireless Conference*, pages 1–4.

Hui, J. W. and Culler, D. E. (2008). Ip is dead, long live ip for wireless sensor networks. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 15–28. ACM.

Intanagonwiwat, C., Govindan, R., and Estrin, D. (2000). Directed diffusion: A scalable and robust communication paradigm for sensor networks. *In Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking*, pages 56–67.

Jamal N. Al-Karaki, A. E. K. (2004). Routing techniques in wireless sensor networks: A survey. *Wireless Communications, IEEE*, pages 1–36.

Johnson, S. G. and Frigo, M. (2007). A modified split-radix fft with fewer arithmetic operations. *IEEE Transactions on Signal Processing*, 55(1):111–119.

Karl, H. and Wilig, A. (2005). *Protocols and Architectures for Wireless Sensor Networks*. Wiley, West Sussex, England.

Kuorilehto, M., Suhonen, J., Kohvakka, M., Hannikainen, M., and Hamalainen, T. D. (2006). Experimenting tcp/ip for low-power wireless sensor networks. In *2006 IEEE 17th International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–6. IEEE.

Lee, J.-S., Su, Y.-W., and Shen, C.-C. (2007). A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi. In *Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE*, pages 46–51. IEEE.

Little, J. and Moler, C. (2015). Matlab. `http://www.mathworks.com/products/matlab/`. Consultado em 3 de Janeiro de 2016.

Lutter, M. (2014). Mel-frequency cepstral coefficients. `http://recognize-speech.com/feature-extraction/mfcc`. Consultado em 20 de Novembro de 2015.

Mazzoni, D. and Dannenberg, R. (1999). Audacity. `http://www.audacityteam.org/`. Consultado em 10 de Junho de 2016.

Mott, R. (1990). *Sound Effects: Radio, TV, and Film*. Focal Press.

Norskog, L. (1991). Sox. `http://sox.sourceforge.net/Main/HomePage`. Consultado em 10 de Junho de 2016.

Nozokido, M. (2013). Zoom h6. `https://www.zoom-na.com/products/field-video-recording/field-recording/h6-handy-recorder/specs`. Consultado em 5 de Janeiro de 2016.

NS-3 (2016). Ns-3. `https://www.nsnam.org/docs/release/3.24/tutorial/ns-3-tutorial.pdf`. Consultado em 8 de Fevereiro de 2016.

Plasse, J. H. (2013). The em algorithm in multivariate gaussian mixture models using anderson acceleration.

Portelo, J., Bugalho, M., Trancoso, I., Neto, J., Abad, A., and Serralheiro, A. (2009). Non-speech audio event detection. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1973–1976. IEEE.

Sjölander, K. and Beskow, J. (2006). Wavesurfer. `http://www.speech.kth.se/wavesurfer/index2.html`. Consultado em 3 de Janeiro de 2016.

Spandan, G., Patel, A., Manjunath, C. R., and GS, N. (2013). Data aggregation protocols in wireless sensor networks. *International Journal of computational Engineering Resaerch*, pages 18–24.

Taylor, P. (2009). *Text-to-Speech Synthesis*. Cambridge University Press.

Ubuntu (2015). Ubuntu 16.04 lts. `https://wiki.ubuntu.com/XenialXerus/ReleaseNotes?_ga=1.83625942.1385163200.1466211588`. Consultado em 10 de Junho de 2016.

Vasseur, J., Fellow, C., Systems, C., Shelby, Z., Chauvenet, C., and Sas, W. (2011). Rpl: The ip routing protocol design for low power and lossy networks. *Internet Protocol for Smart Objects (IPSO) Alliance*.

Winkler, M. and Klaus-Dieter Tuchs, Kester Hughes, G. B. (2008). Theoretical and pratical aspects of military wireless sensor networks. *Journal of Telecommuniacations and information technology*, pages 37–45.

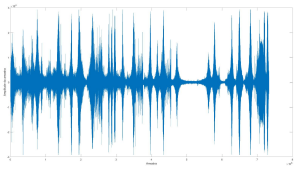
Wojcicki, K. (2011). Htk mfcc matlab. `http://www.mathworks.com/matlabcentral/fileexchange/32849-htk-mfcc-matlab`. Consultado em 3 de Janeiro de 2016.

Wu, Y.-C. (2005). Gaussian mixture model. *Connexions*, pages 1–3.

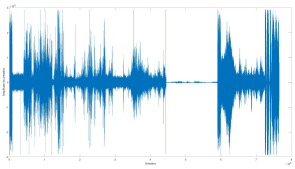
Yang, Y. (2005). Can the strengths of aic and bic be shared? a conflict between model indentification and regression estimation. *Biometrika*, 92(4):937–950.

Apêndice A

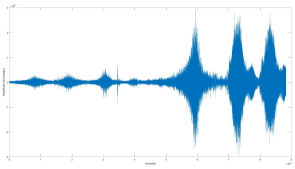
Implementação do processamento de som



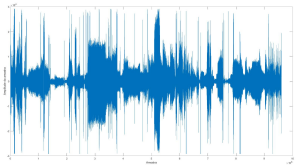
(a) Vetor de som da viatura M11.



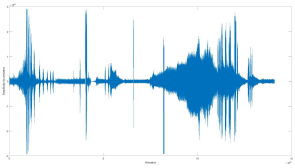
(b) Vetor de som do Garbage.



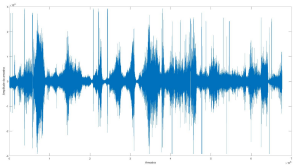
(c) Vetor de som da viatura L200.



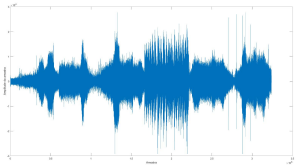
(d) Vetor de som da viatura Leopard-2A6.



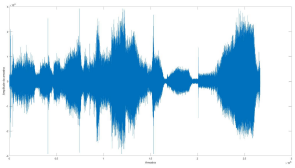
(e) Vetor de som da viatura M60A3.



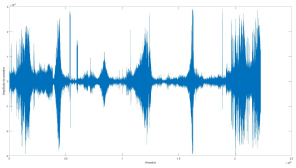
(f) Vetor de som da viatura M113.



(g) Vetor de som da viatura Condor.



(h) Vetor de som da viatura Protect Fire.



(i) Vetor de som da viatura Unimog 1300 L.

Figura A.1: Vetores de som associados às viaturas

Código A.1: Importação e composição dos ficheiros de som em um simples vetor

```
function sound_vec = sound_vec_maker()
    speech=[];
    nsamp=[];
    sound_vec=[];

    list=[dir('*.flac');dir('*.wav');dir('*.raw')];
    fid=[];

    for i=1 : size(list,1)
        fid(i)=fopen(list(i).name,'r','l');
    end

    for i=1 : size(list,1)
        [speech,nsamp(i)]=fread(fid(i),'int16');
        sound_vec=[sound_vec ; speech];
    end

end
```

Código A.2: Cálculo dos MFCC e modelo GMM dado o vetor de som

```
function [ MFCC, mean_n, cov_n,comp] = model_maker(vec, fs, Tw, Ts, alpha,
hamming, R, M, C, L, n_mix)

    [MFCC, FBEs, frames]=mfcc(vec, fs, Tw, Ts, alpha, hamming, R, M, C, L);
    MFCC=MFCC';
    rng(10);
    seed = 1;
    s = RandStream('mt19937ar','Seed',seed);
    opts = statset('MaxIter',1000,'Display','final','Streams',s);
    model = gmdistribution.fit(MFCC,n_mix,'Replicates',10,'Options',opts);
    mean_n = model.mu;
    cov_n = model.Sigma;
    comp=model.PComponents;

end
```

Código A.3: Geração de figura com BIC e AIC para vários modelos GMM desde uma a trinta misturas

```
function bic_aic_representation(vec, fs, Tw, Ts, alpha, hamming, R, M,
C, L, n)
    [MFCC, FBEs, frames]=mfcc(vec, fs, Tw, Ts, alpha, hamming, R, M, C, L);
    MFCC=MFCC';
    MFCC=add_Rhythm_Detection(vec,(4*Tw/1000),MFCC);
    rng(10);
    seed = 1;
    s = RandStream('mt19937ar','Seed',seed);
    opts = statset('MaxIter',1000,'Display','final','Streams',s);
    figure(n);
    for i=1 : 30
        model = gmdistribution.fit(MFCC,i,'Replicates',10,'Options',opts);
        AIC=model.AIC
        BIC=model.BIC
        scatter(i,AIC,[],10);
        hold on;
        scatter(i,BIC,[],7);
        hold on;
    end
end
```

Código A.4: Fragmentação do som e calculo dos MFCC para cada fragmento

```
sub_vec=[];
buffer=16001;

j=size (sample(1:floor((size(sample)*0.85))));
speech_vec4=sample(j+1:size(sample));

for i=1 : ceil(size(speech_vec4)/16000)
    if(buffer<size(speech_vec4,1))
        sub_vec=speech_vec4(buffer-16000:buffer);
        [ MFCCs3(:,:,i), FBEs3, frames3] = mfcc( sub_vec, fs, Tw, Ts,...
        alpha_v, @hamming, R, M, C, L );
        MFCCs4(:,:,i)=MFCCs3(:,:,i)';
        buffer=buffer+16000;
    else
        buffer=size(speech_vec4);
        sub_vec=speech_vec4(buffer-16000:buffer);
        [ MFCCs3(:,:,i), FBEs3, frames3] = mfcc( sub_vec, fs, Tw, Ts,...
        alpha_v, @hamming, R, M, C, L );
        MFCCs4(:,:,i)=MFCCs3(:,:,i)';
    end
end
```

Código A.5: Geração de vetor de *log-likelihood's* dados os MFCC do som a testar

```
function sum_calc = sample_given_model_X_log(mfcc,mean,cov,components)
sum_calc=zeros(1,size(mfcc,3));
sum_calc2=0;
    for k=1 : size(mfcc,3)
        sum_calc(k)=0;
        for i=1 : size(mfcc,1)
            for j=1 : size(cov,3)
                sum_calc2=sum_calc2+components(j)...
                *((1/sqrt(2*pi*det(cov(:,:,j))))...
                *exp((-1/2)*(mfcc(i,:,k)-mean(j,:)))...
                *(inv(cov(:,:,j)))...
                *transpose(mfcc(i,:,k)-mean(j,:))));
            end
            sum_calc(k)=sum_calc(k)+log(sum_calc2);
            sum_calc2=0;
        end
    sum_calc(k)=-sum_calc(k)/size(mfcc,1);
    end
end
```

Código A.6: Geração do vetor de *log-likelihood's* para cada modelo

```
log_aux3 = sample_given_model_X_log(MFCCs4,medias,...
Covariancias,components);
log_aux4 = sample_given_model_X_log(MFCCs4,medias_G,...
Covariancias_G,components2);
log_aux5 = sample_given_model_X_log(MFCCs4,medias_L200,...
Covariancias_L200,components3);
log_aux6 = sample_given_model_X_log(MFCCs4,medias_LEO,...
Covariancias_LEO,components4);
log_aux7 = sample_given_model_X_log(MFCCs4,medias_M60,...
Covariancias_M60,components5);
log_aux8 = sample_given_model_X_log(MFCCs4,medias_M113,...
Covariancias_M113,components6);
log_aux9 = sample_given_model_X_log(MFCCs4,medias_CONDOR,...
Covariancias_CONDOR,components7);
log_aux10 = sample_given_model_X_log(MFCCs4,medias_PROTECT,...
Covariancias_PROTECT,components8);
log_aux11 = sample_given_model_X_log(MFCCs4,medias_UNIMOG,...
Covariancias_UNIMOG,components9);
```

Código A.7: Função que efetua o cálculo da média dos vetores de *log-likelihood's* para um modelo

```
function dist = classifier_helper(vec)
    dist=0;
    for k=1 : size(vec,2)
        dist=dist+vec(k);
    end
    dist=dist/size(vec,2);
end
```

Código A.8: Média dos vetores de *log-likelihood's* para cada modelo

```
a1=classifier_helper(log_aux3);
a2=classifier_helper(log_aux4);
a3=classifier_helper(log_aux5);
a4=classifier_helper(log_aux6);
a5=classifier_helper(log_aux7);
a6=classifier_helper(log_aux8);
a7=classifier_helper(log_aux9);
a8=classifier_helper(log_aux10);
a9=classifier_helper(log_aux11);
```

Código A.9: Escolha do modelo correspondente ao menor *log-likelihood*

```

[min_log,index_log]=min(log_vec2);
if(index_log==1)
disp('IS M11')
elseif(index_log==2)
disp('IS GARBAGE')
elseif(index_log==3)
disp('IS L200')
elseif(index_log==4)
disp('IS LEO')
elseif(index_log==5)
disp('IS M60')
elseif(index_log==6)
disp('IS M113')
elseif(index_log==7)
disp('IS CONDOR')
elseif(index_log==8)
disp('IS PROTECT')
elseif(index_log==9)
disp('IS UNIMOG')
end
```

Código A.10: Função para inserção de BPM numa matriz de MFCC

```
function vec2 = add_Rhythm_Detection(sound_vec,window,mfcc_vec)

buffer=floor((window*16000)/2)+1;
z=1;
    for i=1 : (window/2) :ceil(size(sound_vec)/16000)
        z=z+1;
        if(buffer+floor((window*16000)/2)<size(sound_vec,1))
            sub_vec=sound_vec(buffer-floor((window*16000)/2):buffer+floor((window*16000)/2));
            t=tempo2(sub_vec,16000);
            if(t(3)>0.5)
                vec(:,z) = t(1);
            else
                vec(:,z) = t(2);
            end
            buffer=buffer+floor((window*16000)/2);
        else
            buffer=size(sound_vec);
            sub_vec=sound_vec(buffer-floor((window*16000)):buffer);
            t=tempo2(sub_vec,16000);
            if(t(3)>0.5)
                vec(:,z) = t(1);
            else
                vec(:,z) = t(2);
            end
        end
    end
    j=1;
    k=1;
    for i=1 : size(mfcc_vec,1)
        if(k<size(vec,2))
            mfcc_vec(i,9)=vec(1,k);
            j=j+1;
            if(j==5)
                j=1;
                k=k+1;
            end
        else
            mfcc_vec(i,9)=vec(1,size(vec,2));
        end
    end
    vec2=mfcc_vec;
end
```

Código A.11: Junção dos BPM aos MFCC dos vários veículos

```
MFCC=add_Rhythm_Detection(vec,(4*Tw/1000),MFCC);
```

Código A.12: Junção dos BPM aos MFCC do som a testar

```
sub_vec=[];
buffer=16001;

j=size(sample(1:floor((size(sample)*0.85))));
speech_vec4=sample(j+1:size(sample));

for i=1 : ceil(size(speech_vec4)/16000)
    if(buffer<size(speech_vec4,1))
        sub_vec=speech_vec4(buffer-16000:buffer);
        [MFCCs3(:,:,i), FBEs3, frames3] = mfcc(sub_vec, fs, Tw, Ts,...
        alpha_v, @hamming, R, M, C, L);
        MFCCs4(:,:,i)=MFCCs3(:,:,i)';
        t=tempo2(sub_vec,16000);
            if(t(3)>0.5)
                vec(:,i) = t(1);
            else
                vec(:,i) = t(2);
            end
        buffer=buffer+16000;
    else
        buffer=size(speech_vec4);
        sub_vec=speech_vec4(buffer-16000:buffer);
        [MFCCs3(:,:,i), FBEs3, frames3] = mfcc(sub_vec, fs, Tw, Ts,...
        alpha_v, @hamming, R, M, C, L);
        t=tempo2(sub_vec,16000);
            if(t(3)>0.5)
                vec(:,i) = t(1);
            else
                vec(:,i) = t(2);
            end
        MFCCs4(:,:,i)=MFCCs3(:,:,i)';
    end
end

for k=1 : size(MFCCs4,3)
    for i=1 : size(MFCCs4,1)
            MFCCs4(i,9,k)=vec(1,k);
    end
end
```

Apêndice B

Testes realizados ao som recolhido

Tabela B.1: Resultados associados à utilização de dois vetores e uma componente Gaussiana.

	Medias	**Covariâncias**	**Figura**
Teste 1	$medias1 = \begin{bmatrix} -0.0010 & -0.0010 \end{bmatrix}$	$covariancias1 = \begin{bmatrix} 0.9909 & -0.0070 \\ -0.0070 & 0.9761 \end{bmatrix}$	4.1(a)
Teste 2	$medias2 = \begin{bmatrix} 4.9930 & 4.9987 \end{bmatrix}$	$covariancias2 = \begin{bmatrix} 26.0457 & 25.0573 \\ 25.0573 & 26.0530 \end{bmatrix}$	4.1(b)
Teste 3	$medias3 = \begin{bmatrix} 0.0703 & -0.0392 \end{bmatrix}$	$covariancias3 = \begin{bmatrix} 51.4495 & 0.2023 \\ 0.2023 & 50.6367 \end{bmatrix}$	4.1(c)
Teste 4	$medias4 = \begin{bmatrix} 4.9965 & 5.0007 \end{bmatrix}$	$covariancias4 = \begin{bmatrix} 25.6604 & 25.0382 \\ 25.0382 & 25.6798 \end{bmatrix}$	4.1(d)
Teste 5	$medias5 = \begin{bmatrix} 9.9999 & 10.0003 \end{bmatrix}$	$covariancias5 = \begin{bmatrix} 50.3446 & 50.0049 \\ 50.0049 & 50.3711 \end{bmatrix}$	4.1(e)

Tabela B.2: Resultados associados à utilização de dois vetores e duas componentes Gaussianas.

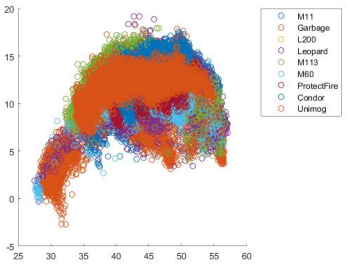
	Medias	**Covariâncias**	**Figura**
Teste 1	$medias1 = \begin{bmatrix} -0.2306 & 0.3770 \\ 0.2413 & -0.3618 \end{bmatrix}$	$covariancias1(:,:,1) = \begin{bmatrix} 0.9547 & 0.0834 \\ 0.0834 & 0.8440 \end{bmatrix}$ $covariancias1(:,:,2) = \begin{bmatrix} 0.9365 & 0.0901 \\ 0.0901 & 0.8651 \end{bmatrix}$	4.2(a)
Teste 2	$medias2 = \begin{bmatrix} 0.0108 & -0.0012 \\ 9.9896 & 9.9990 \end{bmatrix}$	$covariancias2(:,:,1) = \begin{bmatrix} 1.0111 & 0.0114 \\ 0.0114 & 0.9958 \end{bmatrix}$ $covariancias2(:,:,2) = \begin{bmatrix} 0.9814 & 0.0011 \\ 0.0011 & 0.9990 \end{bmatrix}$	4.2(b)
Teste 3	$medias3 = \begin{bmatrix} -0.0080 & 0.0085 \\ -0.0068 & 0.0108 \end{bmatrix}$	$covariancias3(:,:,1) = \begin{bmatrix} 98.5274 & 0.6250 \\ 0.6250 & 100.7026 \end{bmatrix}$ $covariancias3(:,:,2) = \begin{bmatrix} 0.9971 & -0.0023 \\ -0.0023 & 0.9902 \end{bmatrix}$	4.2(c)
Teste 4	$medias4 = \begin{bmatrix} 9.9947 & 10.0002 \\ 0.0046 & -0.0032 \end{bmatrix}$	$covariancias4(:,:,1) = \begin{bmatrix} 0.2531 & 0.0002 \\ 0.0002 & 0.2523 \end{bmatrix}$ $covariancias4(:,:,2) = \begin{bmatrix} 0.9882 & 0.0113 \\ 0.0113 & 0.9937 \end{bmatrix}$	4.2(d)
Teste 5	$medias5 = \begin{bmatrix} 4.9972 & 5.0034 \\ -0.0080 & -0.0105 \end{bmatrix}$	$covariancias5(:,:,1) = \begin{bmatrix} 0.0392 & 0.0012 \\ 0.0012 & 0.0417 \end{bmatrix}$ $covariancias5(:,:,2) = \begin{bmatrix} 0.9910 & -0.0006 \\ -0.0006 & 1.0032 \end{bmatrix}$	4.2(e)

Tabela B.3: Resultados associados à utilização de quatro vetores e quatro componentes Gaussianas.

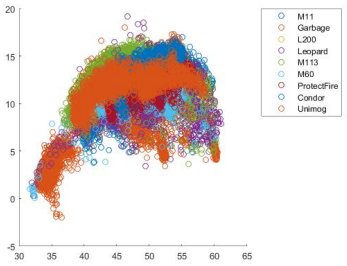
	Medias	**Covariâncias**	**Figura**
Teste 1	$medias1 = \begin{bmatrix} -0.5739 & -0.6009 \\ 0.3950 & -0.0105 \\ -0.6833 & -0.1669 \\ 1.0617 & 0.4197 \end{bmatrix}$	$covariancias1(:,:,1) = \begin{bmatrix} 0.3707 & -0.5382 \\ -0.5382 & 0.9046 \end{bmatrix}$ $covariancias1(:,:,2) = \begin{bmatrix} 0.3236 & -0.2162 \\ -0.2162 & 0.6905 \end{bmatrix}$ $covariancias1(:,:,3) = \begin{bmatrix} 0.5824 & 0.0422 \\ 0.0422 & 1.1348 \end{bmatrix}$ $covariancias1(:,:,4) = \begin{bmatrix} 0.4246 & -0.1670 \\ -0.1670 & 1.2237 \end{bmatrix}$	4.3(a)
Teste 2	$medias2 = \begin{bmatrix} -1.0053 & -0.9934 \\ 1.0055 & 1.0253 \\ 0.0162 & -0.0005 \\ 2.0170 & 2.0147 \end{bmatrix}$	$covariancias2(:,:,1) = \begin{bmatrix} 0.0422 & 0.0017 \\ 0.0017 & 0.0363 \end{bmatrix}$ $covariancias2(:,:,2) = \begin{bmatrix} 0.0367 & 0.0047 \\ 0.0047 & 0.0401 \end{bmatrix}$ $covariancias2(:,:,3) = \begin{bmatrix} 0.0420 & 0.0027 \\ 0.0027 & 0.0399 \end{bmatrix}$ $covariancias2(:,:,4) = \begin{bmatrix} 0.0376 & 0.0018 \\ 0.0018 & 0.0422 \end{bmatrix}$	4.3(b)
Teste 3	$medias3 = \begin{bmatrix} 0.5768 & 0.3204 \\ -0.1369 & -0.0744 \\ 0.0082 & -0.0161 \\ -0.1138 & 0.1152 \end{bmatrix}$	$covariancias3(:,:,1) = \begin{bmatrix} 0.1859 & -0.0620 \\ -0.0620 & 0.1348 \end{bmatrix}$ $covariancias3(:,:,2) = \begin{bmatrix} 0.2197 & -0.0947 \\ -0.0947 & 0.3730 \end{bmatrix}$ $covariancias3(:,:,3) = \begin{bmatrix} 0.0473 & -0.0032 \\ -0.0032 & 0.0498 \end{bmatrix}$ $covariancias3(:,:,4) = \begin{bmatrix} 0.7718 & 0.0347 \\ 0.0347 & 0.6582 \end{bmatrix}$	4.3(c)
Teste 4	$medias4 = \begin{bmatrix} -5.0036 & -5.0626 \\ 4.0402 & 4.0230 \\ 6.0050 & 6.0052 \\ -0.0108 & 0.0322 \end{bmatrix}$	$covariancias4(:,:,1) = \begin{bmatrix} 0.7026 & -0.0100 \\ -0.0100 & 0.6532 \end{bmatrix}$ $covariancias4(:,:,2) = \begin{bmatrix} 0.2506 & 0.0044 \\ 0.0044 & 0.2292 \end{bmatrix}$ $covariancias4(:,:,3) = \begin{bmatrix} 0.0395 & -0.0024 \\ -0.0024 & 0.0418 \end{bmatrix}$ $covariancias4(:,:,4) = \begin{bmatrix} 1.2164 & 0.0047 \\ 0.0047 & 0.9251 \end{bmatrix}$	4.3(d)
Teste 5	$medias5 = \begin{bmatrix} 0.0877 & 0.0969 \\ -2.9519 & -3.0284 \\ 4.0042 & 3.9888 \\ 1.9818 & 1.9823 \end{bmatrix}$	$covariancias5(:,:,1) = \begin{bmatrix} 0.9280 & -0.0135 \\ -0.0135 & 0.9475 \end{bmatrix}$ $covariancias5(:,:,2) = \begin{bmatrix} 0.2131 & -0.0109 \\ -0.0109 & 0.2073 \end{bmatrix}$ $covariancias5(:,:,3) = \begin{bmatrix} 0.1058 & 0.0140 \\ 0.0140 & 0.1011 \end{bmatrix}$ $covariancias5(:,:,4) = \begin{bmatrix} 0.0401 & -0.0018 \\ -0.0018 & 0.0375 \end{bmatrix}$	4.3(e)

Tabela B.4: Tempo de execução em segundos, das janelas 50ms, 100ms e 400ms, para modelos gerados com dois coeficientes cepestrais.

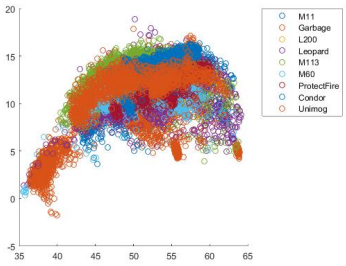
Som de teste / Janela(ms)	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Total
50ms	24,3872	4,8903	30,8484	6,7175	22,5216	11,9815	10,2611	8,8192	120,4268
100ms	14,5937	3,7761	17,2893	4,6016	12,8511	7,4618	6,405	5,7519	72,7305
400ms	13,3048	3,4334	15,3236	4,143	11,3785	6,6616	5,8098	5,2709	65,3256



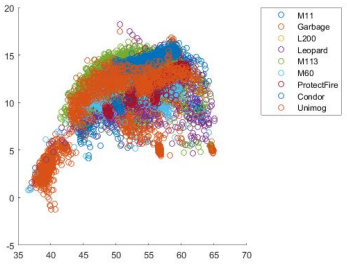
(a) Janela de 50ms.



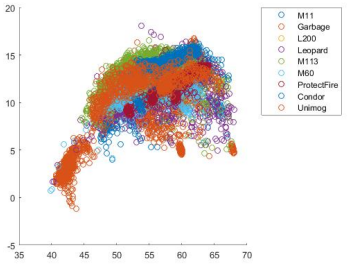
(b) Janela de 100ms.



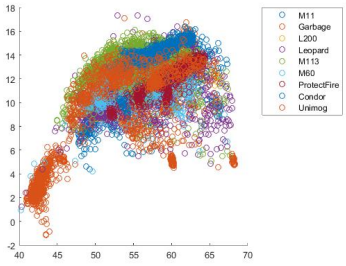
(c) Janela de 150ms.



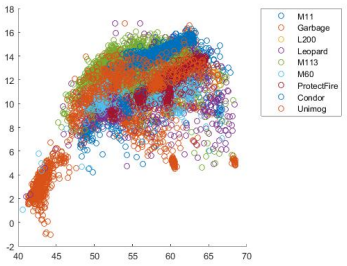
(d) Janela de 250ms.



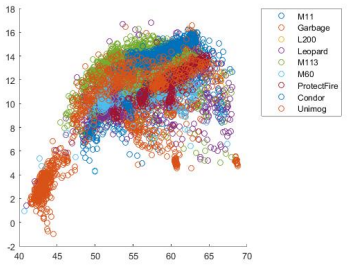
(e) Janela de 300ms.



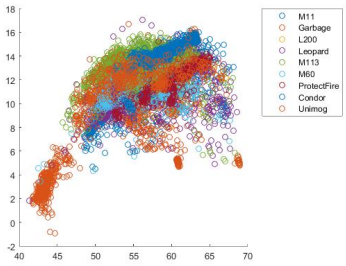
(f) Janela de 350ms.



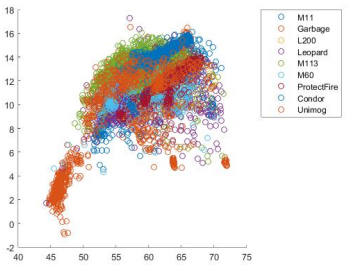
(g) Janela de 400ms.



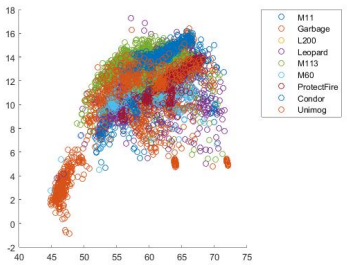
(h) Janela de 450ms.



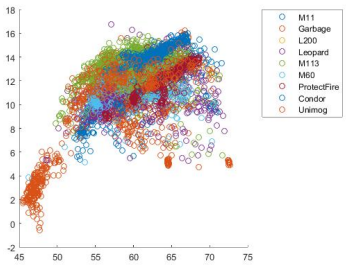
(i) Janela de 500ms.



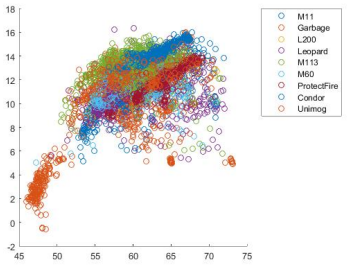
(j) Janela de 550ms.



(k) Janela de 600ms.

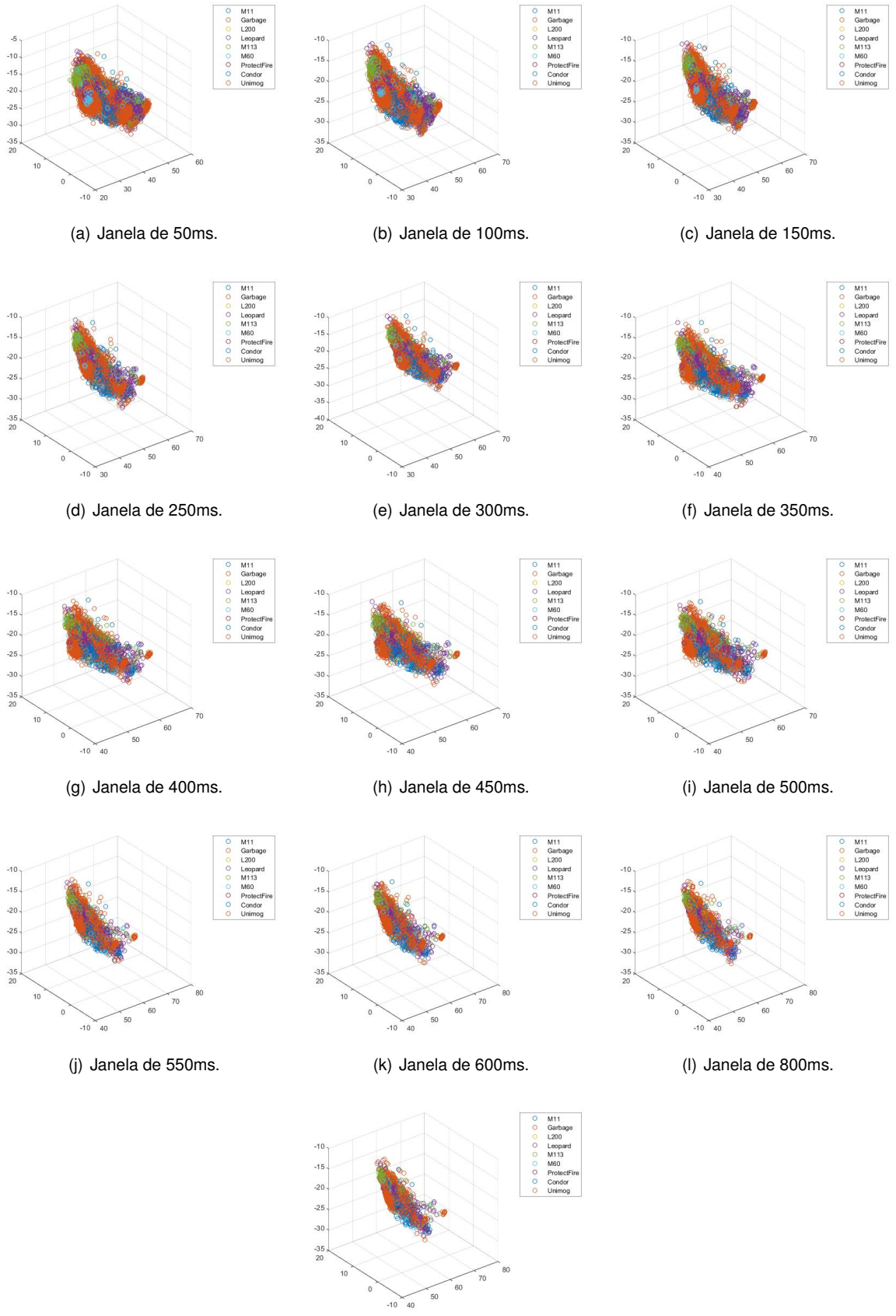


(l) Janela de 800ms.



(m) Janela de 1000ms.

Figura B.1: Disposição dos MFCC com dois coeficientes cepstrais



(a) Janela de 50ms. (b) Janela de 100ms. (c) Janela de 150ms.

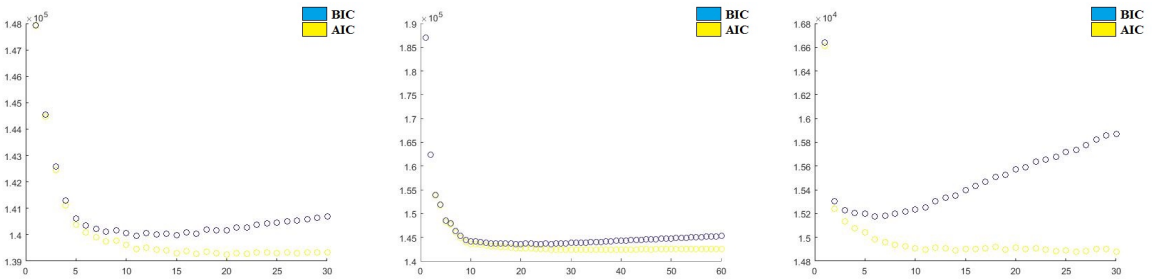
(d) Janela de 250ms. (e) Janela de 300ms. (f) Janela de 350ms.

(g) Janela de 400ms. (h) Janela de 450ms. (i) Janela de 500ms.

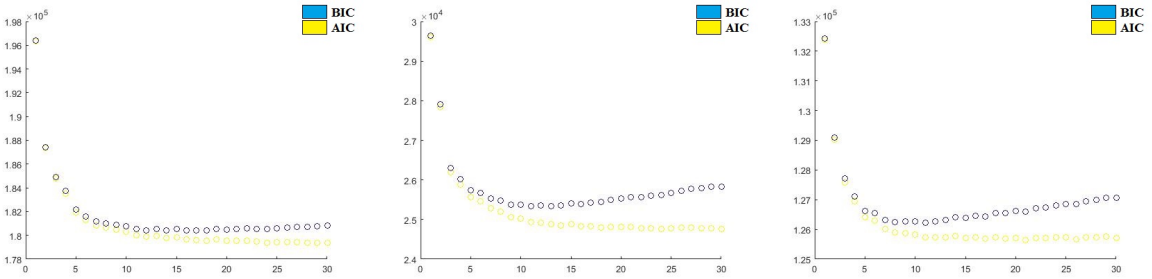
(j) Janela de 550ms. (k) Janela de 600ms. (l) Janela de 800ms.

(m) Janela de 1000ms.

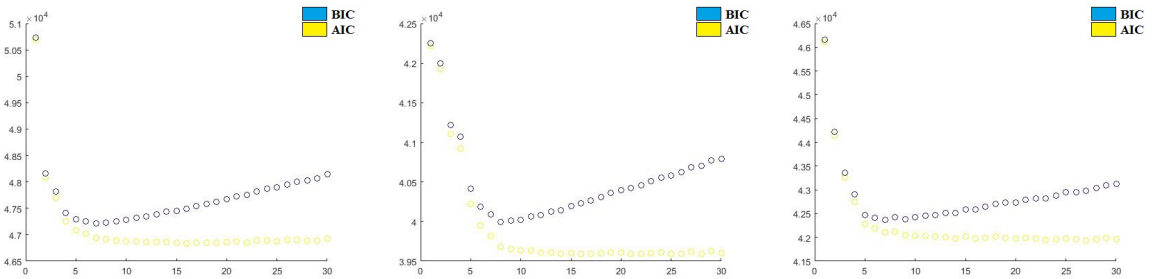
Figura B.2: Disposição dos MFCC com três coeficientes cepstrais



(a) M11. (b) Garbage. (c) L200.



(d) Leopard-2A6. (e) M60-A3. (f) M113.



(g) Condor. (h) Protect Fire. (i) Unimog-1300L.

Figura B.3: BIC e AIC para janela de 50ms

Tabela B.5: Número de componentes ideal de GMM dados MFCC com dois coeficientes.

Modelos / Janela(ms)	Panhard M11	Garbage	L200	Leopard 2A6	M60 A3	M113	Condor	Protect Fire	Unimog 1300L
50	11	26	6	18	13	11	7	8	7
100	9	21	5	19	13	8	6	8	7
150	12	17	5	23	9	8	7	9	6
250	11	18	3	15	7	8	7	7	7
300	8	15	4	16	6	7	7	7	6
350	6	16	2	16	8	6	7	8	5
400	6	15	3	17	6	7	7	8	5
450	7	13	2	11	5	5	7	6	5
500	6	11	4	10	6	5	7	7	5
550	6	11	2	8	5	5	7	7	5
600	5	11	2	9	5	5	8	7	5
800	5	8	2	7	4	5	7	7	5
1000	5	11	2	7	4	5	7	5	5

Tabela B.6: Número de componentes ideal de GMM dados MFCC com oito coeficientes.

Modelos / Janela(ms)	Panhard M11	Garbage	L200	Leopard 2A6	M60 A3	M113	Condor	Protect Fire	Unimog 1300L
50	29	37	9	37	11	19	11	10	9
100	22	28	4	34	14	18	10	8	8
150	17	29	3	27	9	17	8	9	8
250	11	18	3	22	8	14	9	8	7
300	12	17	3	20	6	9	5	6	7
350	10	16	3	19	6	11	4	6	7
400	7	13	3	18	6	9	4	5	5
450	7	14	3	17	7	7	4	6	5
500	7	13	3	15	5	9	4	6	5
550	7	13	3	15	6	9	4	5	6
600	7	11	3	15	5	6	4	6	6
800	6	13	2	10	4	7	4	3	3
1000	6	8	2	11	4	5	4	3	3

Tabela B.7: Número de componentes ideal de GMM dados MFCC com oito coeficientes e BPM.

Modelos / Janela(ms)	Panhard M11	Garbage	L200	Leopard 2A6	M60 A3	M113	Condor	Protect Fire	Unimog 1300L
50	51	27	6	32	13	14	6	20	10
100	22	29	5	41	13	16	7	24	8
150	16	24	5	21	9	14	8	9	10
250	12	18	3	21	6	11	4	7	6
300	9	15	4	25	6	9	4	8	5
350	9	17	3	17	8	10	6	13	5
400	8	16	3	15	6	7	4	5	4
450	7	13	2	20	6	7	4	6	5
500	7	7	3	13	6	6	5	5	5
550	8	10	2	20	4	7	4	6	3
600	7	11	2	9	5	7	4	4	3
800	6	7	2	8	4	5	3	3	3
1000	5	8	1	18	4	5	5	3	3

Tabela B.8: Acertos de cada som de teste dados MFCC com dois coeficientes.

Som de teste / Janela(ms)	Panhard M11	L200	Leopard 2A6	M60 A3	M113	Condor	Protect Fire	Unimog 1300L
50	1	0	1	1	0	1	0	1
100	1	1	0	1	0	1	0	1
150	1	0	0	1	0	1	0	1
250	0	1	0	1	0	1	0	1
300	0	0	0	1	0	1	0	1
350	0	0	0	1	0	1	0	1
400	1	1	0	1	0	1	0	1
450	0	0	0	1	0	1	0	1
500	0	0	0	1	0	1	0	1
550	0	0	0	1	0	1	0	1
600	0	0	0	1	0	1	0	1
800	0	0	0	1	0	1	0	1
1000	0	0	0	1	0	1	0	1

Tabela B.9: Acertos de cada som de teste dados MFCC com oito coeficientes.

Som de teste / Janela(ms)	Panhard M11	L200	Leopard 2A6	M60 A3	M113	Condor	Protect Fire	Unimog 1300L
50	1	1	1	1	1	1	0	1
100	1	1	1	1	1	1	0	1
150	1	1	1	1	1	1	0	1
250	1	1	1	1	1	1	0	1
300	1	1	1	1	1	1	0	1
350	1	1	1	0	1	1	0	1
400	1	0	1	1	1	1	0	1
450	1	1	1	0	1	1	0	1
500	1	1	1	1	1	1	0	1
550	1	1	1	0	1	1	0	1
600	1	1	1	0	1	1	0	1
800	1	1	1	0	1	1	0	1
1000	1	1	0	0	1	1	0	1

Tabela B.10: Acertos de cada som de teste dados MFCC com oito coeficientes e BPM.

Som de teste / Janela(ms)	Panhard M11	L200	Leopard 2A6	M60 A3	M113	Condor	Protect Fire	Unimog 1300L
50	0	0	0	0	0	1	0	1
100	1	0	1	0	1	1	0	1
150	1	1	1	1	1	1	0	1
250	1	1	1	1	1	1	0	0
300	1	0	1	0	1	1	0	1
350	1	1	1	0	1	1	0	1
400	1	0	1	0	1	1	0	1
450	1	1	1	0	1	1	0	1
500	1	1	1	0	1	1	0	1
550	1	0	1	1	1	1	0	1
600	1	1	1	0	1	1	0	1
800	1	0	1	0	1	1	0	1
1000	0	1	0	0	0	0	0	1

Tabela B.11: Medidas de desempenho da classificação para MFCC com dois coeficientes.

50ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	0	1	1	0	1	0	1	1	0,625	0,625	0,625
FP	0	1	0	0	1	0	1	0				
VN	7	6	7	7	6	7	6	7				
FN	0	1	0	0	1	0	1	0				
100ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	0	1	0	1	0	1	1	0,625	0,625	0,625
FP	0	0	1	0	1	0	1	0				
VN	7	7	6	7	6	7	6	7				
FN	0	0	1	0	1	0	1	0				
150ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	0	0	1	0	1	0	1	1	0,5	0,5	0,5
FP	0	1	1	0	1	0	1	0				
VN	7	6	6	7	6	7	6	7				
FN	0	1	1	0	1	0	1	0				
250ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	0	1	0	1	0	1	0	1	1	0,5	0,5	0,5
FP	1	0	1	0	1	0	1	0				
VN	6	7	6	7	6	7	6	7				
FN	1	0	1	0	1	0	1	0				
300ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	0	0	0	1	0	1	0	1	1	0,375	0,375	0,375
FP	1	1	1	0	1	0	1	0				
VN	6	6	6	7	6	7	6	7				
FN	1	1	1	0	1	0	1	0				
350ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	0	0	0	1	0	1	0	1	1	0,375	0,375	0,375
FP	1	1	1	0	1	0	1	0				
VN	6	6	6	7	6	7	6	7				
FN	1	1	1	0	1	0	1	0				
400ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	0	1	0	1	0	1	1	0,625	0,625	0,625
FP	0	0	1	0	1	0	1	0				
VN	7	7	6	7	6	7	6	7				
FN	0	0	1	0	1	0	1	0				
450ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	0	0	0	1	0	1	0	1	1	0,375	0,375	0,375
FP	1	1	1	0	1	0	1	0				
VN	6	6	6	7	6	7	6	7				
FN	1	1	1	0	1	0	1	0				
500ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	0	0	0	1	0	1	0	1	1	0,375	0,375	0,375
FP	1	1	1	0	1	0	1	0				
VN	6	6	6	7	6	7	6	7				
FN	1	1	1	0	1	0	1	0				
550ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	0	0	0	1	0	1	0	1	1	0,375	0,375	0,375
FP	1	1	1	0	1	0	1	0				
VN	6	6	6	7	6	7	6	7				
FN	1	1	1	0	1	0	1	0				
600ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	0	0	0	1	0	1	0	1	1	0,375	0,375	0,375
FP	1	1	1	0	1	0	1	0				
VN	6	6	6	7	6	7	6	7				
FN	1	1	1	0	1	0	1	0				
800ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	0	0	0	1	0	1	0	1	1	0,375	0,375	0,375
FP	1	1	1	0	1	0	1	0				
VN	6	6	6	7	6	7	6	7				
FN	1	1	1	0	1	0	1	0				
1000ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	0	0	0	1	0	1	0	1	1	0,375	0,375	0,375
FP	1	1	1	0	1	0	1	0				
VN	6	6	6	7	6	7	6	7				
FN	1	1	1	0	1	0	1	0				

Tabela B.12: Medidas de desempenho da classificação para MFCC com oito coeficientes.

50ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	1	1	1	1	0	1	1	0,875	0,875	0,875
FP	0	0	0	0	0	0	1	0				
VN	7	7	7	7	7	7	6	7				
FN	0	0	0	0	0	0	1	0				
100ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	1	1	1	1	0	1	1	0,875	0,875	0,875
FP	0	0	0	0	0	0	1	0				
VN	7	7	7	7	7	7	6	7				
FN	0	0	0	0	0	0	1	0				
150ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	1	1	1	1	0	1	1	0,875	0,875	0,875
FP	0	0	0	0	0	0	1	0				
VN	7	7	7	7	7	7	6	7				
FN	0	0	0	0	0	0	1	0				
250ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	1	1	1	1	0	1	1	0,875	0,875	0,875
FP	0	0	0	0	0	0	1	0				
VN	7	7	7	7	7	7	6	7				
FN	0	0	0	0	0	0	1	0				
300ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	1	1	1	1	0	1	1	0,875	0,875	0,875
FP	0	0	0	0	0	0	1	0				
VN	7	7	7	7	7	7	6	7				
FN	0	0	0	0	0	0	1	0				
350ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	1	0	1	1	0	1	1	0,75	0,75	0,75
FP	0	0	0	1	0	0	1	0				
VN	7	7	7	6	7	7	6	7				
FN	0	0	0	1	0	0	1	0				
400ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	0	1	1	1	1	0	1	1	0,75	0,75	0,75
FP	0	1	0	0	0	0	1	0				
VN	7	6	7	7	7	7	6	7				
FN	0	1	0	0	0	0	1	0				
450ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	1	0	1	1	0	1	1	0,75	0,75	0,75
FP	0	0	0	1	0	0	1	0				
VN	7	7	7	6	7	7	6	7				
FN	0	0	0	1	0	0	1	0				
500ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	1	1	1	1	0	1	1	0,875	0,875	0,875
FP	0	0	0	0	0	0	1	0				
VN	7	7	7	7	7	7	6	7				
FN	0	0	0	0	0	0	1	0				
550ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	1	0	1	1	0	1	1	0,75	0,75	0,75
FP	0	0	0	1	0	0	1	0				
VN	7	7	7	6	7	7	6	7				
FN	0	0	0	1	0	0	1	0				
600ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	1	0	1	1	0	1	1	0,75	0,75	0,75
FP	0	0	0	1	0	0	1	0				
VN	7	7	7	6	7	7	6	7				
FN	0	0	0	1	0	0	1	0				
800ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	1	1	0	1	1	0	1	1	0,75	0,75	0,75
FP	0	0	0	1	0	0	1	0				
VN	7	7	7	6	7	7	6	7				
FN	0	0	0	1	0	0	1	0				
1000ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	1	0	1	0	1	1	0	1	1	0,625	0,625	0,625
FP	0	1	0	1	0	0	1	0				
VN	7	6	7	6	7	7	6	7				
FN	0	1	0	1	0	0	1	0				

Tabela B.13: Medidas de desempenho da classificação para MFCC com oito coeficientes e BPM.

50ms	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Beta	Recall	Precision	F-score
VP	0	0	0	0	0	1	0	1	1	0,25	0,25	0,25
FP	1	1	1	1	1	0	1	0				
VN	6	6	6	6	6	7	6	7				
FN	1	1	1	1	1	0	1	0				
100ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	1	0	1	0	1	1	0	1	1	0,625	0,625	0,625
FP	0	1	0	1	0	0	1	0				
VN	7	6	7	6	7	7	6	7				
FN	0	1	0	1	0	0	1	0				
150ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	1	1	1	1	1	1	0	1	1	0,875	0,875	0,875
FP	0	0	0	0	0	0	1	0				
VN	7	7	7	7	7	7	6	7				
FN	0	0	0	0	0	0	1	0				
250ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	1	1	1	1	1	1	0	0	1	0,75	0,75	0,75
FP	0	0	0	0	0	0	1	1				
VN	7	7	7	7	7	7	6	6				
FN	0	0	0	0	0	0	1	1				
300ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	1	0	1	0	1	1	0	1	1	0,625	0,625	0,625
FP	0	1	0	1	0	0	1	0				
VN	7	6	7	6	7	7	6	7				
FN	0	1	0	1	0	0	1	0				
350ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	1	1	1	0	1	1	0	1	1	0,75	0,75	0,75
FP	0	0	0	1	0	0	1	0				
VN	7	7	7	6	7	7	6	7				
FN	0	0	0	1	0	0	1	0				
400ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	1	0	1	0	1	1	0	1	1	0,625	0,625	0,625
FP	0	1	0	1	0	0	1	0				
VN	7	6	7	6	7	7	6	7				
FN	0	1	0	1	0	0	1	0				
450ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	1	1	1	0	1	1	0	1	1	0,75	0,75	0,75
FP	0	0	0	1	0	0	1	0				
VN	7	7	7	6	7	7	6	7				
FN	0	0	0	1	0	0	1	0				
500ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	1	1	1	0	1	1	0	1	1	0,75	0,75	0,75
FP	0	0	0	1	0	0	1	0				
VN	7	7	7	6	7	7	6	7				
FN	0	0	0	1	0	0	1	0				
550ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	1	0	1	1	1	1	0	1	1	0,75	0,75	0,75
FP	0	1	0	0	0	0	1	0				
VN	7	6	7	7	7	7	6	7				
FN	0	1	0	0	0	0	1	0				
600ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	1	1	1	0	1	1	0	1	1	0,75	0,75	0,75
FP	0	0	0	1	0	0	1	0				
VN	7	7	7	6	7	7	6	7				
FN	0	0	0	1	0	0	1	0				
800ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	1	0	1	0	1	1	0	1	1	0,625	0,625	0,625
FP	0	1	0	1	0	0	1	0				
VN	7	6	7	6	7	7	6	7				
FN	0	1	0	1	0	0	1	0				
1000ms	**M11**	**L200**	**LEO**	**M60**	**M113**	**CONDOR**	**PROTECT**	**UNIMOG**	**Beta**	**Recall**	**Precision**	**F-score**
VP	0	1	0	0	0	0	0	1	1	0,25	0,25	0,25
FP	1	0	1	1	1	1	1	0				
VN	6	7	6	6	6	6	6	7				
FN	1	0	1	1	1	1	1	0				

Tabela B.14: Tempo de execução, em segundos, das janelas 50ms, 100ms, 150ms, 250ms, 300ms e 500ms, para modelos gerados com oito coeficientes cepestrais.

Som de teste / Janela(ms)	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Total
50ms	49,5078	8,2488	63,474	11,6104	45,3529	23,0494	19,0437	16,5658	236,8525
100ms	23,6911	5,0134	30,295	6,4708	21,1617	11,4238	9,5735	8,3554	115,9848
150ms	15,4705	3,793	18,684	4,6928	13,7458	7,7639	6,6929	5,9764	76,8191
250ms	11,8446	3,3098	13,882	3,9905	10,504	6,2517	5,4368	4,8847	60,1036
300ms	11,5124	3,2324	13,204	3,7591	9,8505	5,8557	5,1244	4,6302	57,169
500ms	17,434	4,0257	21,537	5,0523	15,5393	8,5273	7,3401	6,6518	86,1078

Tabela B.15: Tempo de execução, em segundos, da janela de 150ms para modelos gerados com oito coeficientes cepestrais e com BPM.

Som de teste / Janela(ms)	M11	L200	LEO	M60	M113	CONDOR	PROTECT	UNIMOG	Total
150ms	15,2791	3,8819	19,6155	4,8865	14,3123	8,0903	6,9334	6,1768	79,1758

Apêndice C

Implementação da rede de sensores sem fios

Código C.1: Método *read* da classe *readFile*

```
uint8_t ReadFile::read (char *name){
 uint8_t out;
 std::ifstream infile(name);
  if(!infile){
   std::cout<<"Error opening file.";
  }
 infile>>out;
 infile.close();
 return out;
}
```

Código C.2: Adicionar do intercetor de pacotes à aplicação

```
Ptr<Ipv4L3Protocol> ipv4Proto = GetNode()->GetObject<Ipv4L3Protocol> ();
 if ( ipv4Proto != 0){
  NS_LOG_INFO ("Ipv4 packet interceptor added");
  ipv4Proto->AddPacketInterceptor (ns3::MakeCallback (&ListenCarApplication::PacketIntercept, this), UdpL4Protocol::PROT_NUMBER);
  }
 else{
  NS_LOG_INFO ("No Ipv4 with packet intercept facility");
  }
```

Código C.3: Adicionar do cabeçalho ao pacote e envio do mesmo

```
Ptr<Packet> packet = Create<Packet> (m_pktSize);
CarDataHeader carHeader;
carHeader.SetCarData (send_int);
carHeader.SetCarDataID (IDs);
packet->AddHeader (carHeader);
TimestampTag timestamp;
timestamp.SetTimestamp (Simulator::Now ());
packet->AddByteTag (timestamp);
m_txTrace (packet);
m_socket->Send (packet);
```

Código C.4: Agregação de dados após interceção de pacote

```
bool
ListenCarApplication::PacketIntercept(Ptr<Packet> p, const Ipv4Header & ipHeader)
{
  bool res = false;
  p->RemoveAtStart(8);
  int z,m=0;
  CarDataHeader carHeader;
  TimestampTag timestamp;
  if (p->FindFirstMatchingByteTag (timestamp)) {
   tx = timestamp.GetTimestamp ();
  }
  if(buffer_index==0)
  {
   last_timetag=tx;
  }
  else{
   if(tx<last_timetag)
     last_timetag=tx;
  }
  p->RemoveHeader (carHeader);
  uint8_t data_aux=carHeader.GetCarData ();
  std::vector<int> ID_aux = carHeader.GetCarDataID ();
  if(buffer_index==0){
   pack.push_back(Packets());
   pack[pack.size()-1].ID=ID_aux;
   pack[pack.size()-1].data=data_aux;
   pack[pack.size()-1].time_packet=Now();
   buffer_index++;
  }
  else{
   for(std::vector<Packets>::iterator i=pack.begin(); i!=pack.end(); i++,z++)
   {
    if(pack[z].data==data_aux){
     std::sort(pack[z].ID.begin(), pack[z].ID.end());
     std::sort(ID_aux.begin(), ID_aux.end());
     std::vector<int> ID_aux_2;
     std::merge(pack[z].ID.begin(), pack[z].ID.end(),
       ID_aux.begin(), ID_aux.end(),
                 std::back_inserter(ID_aux_2));
     std::vector<int>::iterator pte = std::unique(ID_aux_2.begin(), ID_aux_2.end());
     ID_aux_2.erase(pte, ID_aux_2.end());
     pack[z].ID=ID_aux_2;
     break;
    }
    if(std::distance(i,pack.end())==1){
     pack.push_back(Packets());
     pack[pack.size()-1].ID=ID_aux;
     pack[pack.size()-1].data=data_aux;
     pack[pack.size()-1].time_packet=Now();
     buffer_index++;
     break;
    }
   }
  }
  return res;
}
```

Código C.5: Envio da agregação

```
void ListenCarApplication::SendAggregation ()
{
 for(int k=0;k<buffer_index;k++){
  pack[k].ID.push_back(int(GetNode()->GetId()));
  uint8_t send_int=pack[k].data;
  Ptr<Packet> packet = Create<Packet> ();
  CarDataHeader carHeader;
  carHeader.SetCarData (send_int);
  carHeader.SetCarDataID (pack[k].ID);
  packet->AddHeader (carHeader);
  TimestampTag timestamp;
  timestamp.SetTimestamp (last_timetag);
  packet->AddByteTag (timestamp);
  m_txTrace (packet);
  m_socket->Send (packet);
 }
 pack.resize(0);
 buffer_index=0;
}
```

Código C.6: Tratamento de pacotes ICMPv6 no envio do protocolo DSDV

```
if(header.GetDestinationAddress().IsSolicitedMulticast()||header.GetDestinationAddress().IsAllRoutersMulticast()){
    Ptr<Ipv6Route> route_aux3 = Create<Ipv6Route> ();
                route_aux3->SetOutputDevice(oif);
                route_aux3->SetGateway(address_aux);
                route_aux3->SetDestination(address_aux);
                route_aux3->SetSource(header.GetSourceAddress());
                return route_aux3;
}
```

Código C.7: Tratamento de pacotes ICMPv6 na receção do protocolo DSDV

```
if(header.GetDestinationAddress().IsMulticast()||header.GetDestinationAddress()==m_ipv6->GetAddress(1,1).GetAddress()){
  lcb (p, header, iif);
  return true;
}
```