Andrea Terra Pinto
Nº aluno: 2009105444

iBombeiro
iDevice to monitor and help firefighters

04/09/2013

Universidade de Coimbra

iBombeiro
iDevice to monitor and help firefighters

Andrea Terra Pinto

Coimbra, 04 September 2013

A Dissertation presented for the degree of Master of Science in Electrical and Computer Engineering

Faculty of Science and Technology
University of Coimbra

Jury:

President	:	Prof. Doutor Manuel Marques Crisóstomo
Member	:	Prof. Doutor João Pedro de Almeida Barreto
Supervisor	:	Prof. Doutor Jorge Nuno de Almeida e Sousa Almada Lobo
Co-Supervisor	:	Prof. Doutor Paulo Jorge Carvalho Menezes

"Probability theory is nothing but common sense reduced to calculation."

Pierre-Simon Laplace

Abstract

This thesis presents the development of a monitoring system for human agents, within the scope of the Chopin Project. The Chopin Project (supported by FCT-Portuguese Funding Agency) addresses the cooperation between teams of autonomous robots and human teams, by exploiting the human-robot symbiosis in search and rescue missions in small scale catastrophic urban incidents. This monitoring system is called iBombeiro.

For the development of iBombeiro a mobile platform technology iOS - iPhone 4S was chosen. The action recognition system is based on acceleration data of the user, acquired with the accelerometers of the device, and removing the gravity vector. We used a classification model based on Bayesian techniques, where inference is made based on the Maximum A Posteriori (MAP). The classifier was successfully implemented on the iphone, generating interesting results and providing a solid foundation for future development. An extended model to the system was further developed, for a future implementation, allowing a personalised iBombeiro.

The communication protocols with a reconfigurable network (MANET) were defined and implemented, in accordance with the requirements of the CHOPIN Project, to transmit the classifier results on the fly. Beyond this communication protocol solution iBombeiro also offers a webservice, allowing the system to work standalone. The iBombeiro can also be used as a "black box", recording the acquired data and enabling subsequent queries.

Although only the acceleration data and a reduced number of characteristics were used, due to computational limitations, the system has shown good results under the conditions proposed, using methods of cross-validation.

In the end some conclusions are drawn, pointing out some possible improvements for the system, and providing directions of future work, as well as showing the possible contribution of this work integrated into other projects.

Resumo

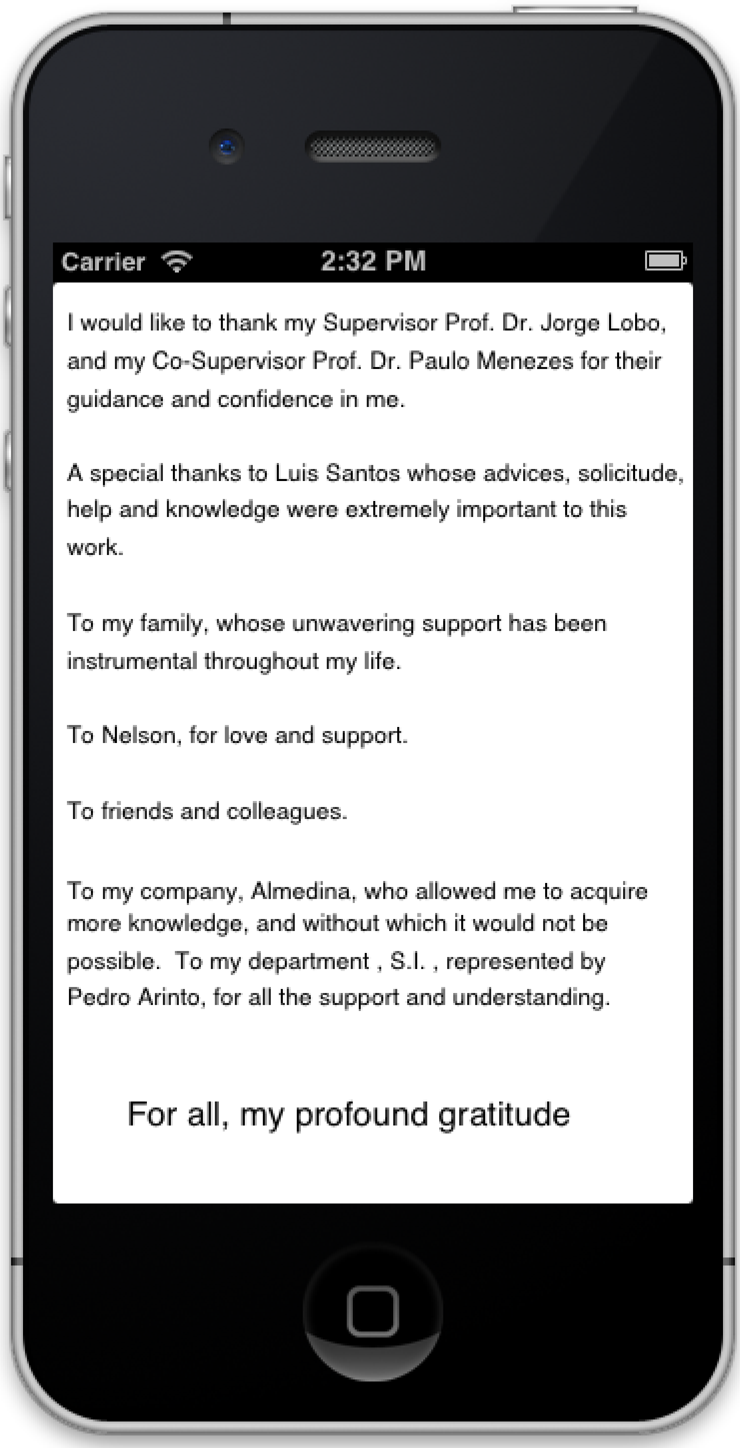
Esta tese apresenta o desenvolvimento de um sistema de monitorização de agentes humanos, e a sua integração no Projecto Chopin. O Projecto Chopin (financiado pela FCT -Fundação para a Ciência e a Tecnologia) tem como objetivo a cooperação entre equipas de robots autónomos e equipas humanas, explorando a simbiose Homem/Máquina, em missões de busca e salvamento em pequena escala, em incidentes catastróficos e urbanos. Este sistema foi denominado de iBombeiro.

Para o desenvolvimento do iBombeiro foi escolhida uma plataforma móvel, com tecnologia iOS - iphone 4S. O sistema de reconhecimento de acções é baseado nos dados de aceleração do utilizador, adquiridos com os acelerómetros do próprio dispositivo e removendo o vector gravidade. Na implementação deste sistema foi utilizado um modelo de classificação baseado em técnicas Bayesianas, cuja inferência é feita com base no *Maximum A Posteriori* (MAP). O classificador foi implementado com sucesso no iphone, gerando resultados interessantes e constituindo uma base sólida para desenvolvimento futuro. Foi desenvolvido ainda um modelo estendido ao sistema já implementado, para um futuro desenvolvimento, permitindo a personalização do iBombeiro. Foram definidos os protocolos de comunicação com uma rede de dados reconfigurável (MANET), de acordo com os requisitos do Projecto CHOPIN. Para além deste protocolo de comunicação o sistema disponibiliza também um "webservice", permitindo ao sistema trabalhar isoladamente. O iBombeiro exibe também um comportamento tipo "caixa preta", gravando os dados adquiridos e permitindo a consulta posterior dos mesmos.

Apesar de, devido a limitações computacionais, apenas se ter utilizado os dados de aceleração e um número reduzido de características, o sistema demostrou bons resultados, dentro das condições propostas, utilizando métodos de validação cruzada.

Este documento termina apresentando as conclusões e apontando algumas melhorias possíveis no sistema, e traçando rumos futuros, mostrando ainda a possível contribuição deste trabalho integrado noutros projectos.

Acknowledgments



Contents

List of Figures

List of Tables

Abbreviations

ANPC **P**ortuguese **C**ivil **D**efense **A**uthority

BSC **B**ombeiros **S**apadores de **C**oimbra

DBN **D**ynamic **B**ayesian **N**etwork

CCO **C**ommand **C**enter **O**perations

CHOPIN **C**ooperation between **H**uman and r**O**botic teams in catastro**P**hic **IN**cidents

HMM **H**idden **M**arkov **M**odels

ISR **I**nstitute of **S**ystems and **R**obotics

MANET **M**obile **A**d-hoc **NET**work

MRL **M**obile **R**obotics **L**aboratory

PASS **P**ersonal **A**lert **S**afety **S**ystem

ROS **R**obot **O**perating **S**ystem

SCBA **S**elf **C**ontained **B**reathing **A**pparatus

SVM **S**upport **V**ector **M**achines

QoS **Q**uality **o**f **S**ervice

Chapter 1

Introduction

Mobile devices are becoming increasingly sophisticated and the latest generation of smart cell phones now incorporates many diverse and powerful sensors. These sensors include acceleration sensors (accelerometers), GPS sensors, vision sensors (cameras), audio sensors (microphones), light sensors, temperature sensors, direction sensors (magnetic compasses).

The objective of this work is to have a system capable of automatically classifying the physical actions performed by a human subject. By the term physical actions, we mean either static postures, such as standing, or dynamic motions, such as walking, running, stair climbing, descend stairs.

This work is integrated in a major project, called CHOPIN Project.[1] The CHOPIN Project aims at exploiting the human-robot symbiosis in the development of human rescuers' support systems for small-scale search and rescue missions in urban catastrophic incidents, an application domain with an unquestionable beneficial impact on society. A proof of concept will be developed for innovative techniques about cooperation between teams of human agents and teams of mobile robotic agents and collaborative action recognizer.

The CHOPIN Project started on April 1st, 2012. At the moment the project team decided already to focus on two specific catastrophic scenarios in urban areas, to serve as proof of concept for CHOPIN:

- Fire in a large basement garage;
- Leakage of toxic gases in an industrial plant.

This kind of indoor scenario usually poses radio propagation difficulties to the response teams, whose members usually wear a radio emitter/transmitter to communicative by voice. Often, the communication is only possible with teammates located in line of sight; therefore,

[1] `http://chopin.isr.uc.pt/`

the communication of a first responder with the command center may require a kind of human multihop communication across multiple first responders.
In the scope of the CHOPIN Project, since a wireless communication network may be absent or damaged, human and robotic agents will have to deploy and maintain a Mobile Ad Hoc Network (MANET) for supporting interaction within the human team, within the robotic team, and between these teams.

1.1 Related Work on Action Recognition on Mobile Devices

Real-world processes generally produce observable outputs which can be characterised as signals. The signals can be discrete or continuous in nature. The original source can be stationary or non-stationary (i.e, the signal properties vary over time). The signals can be pure, or can be corrupted from other signal sources (e.g., noise) or by transmission distortions, reverberation, etc.
A problem of fundamental interest is characterising such real world signals in terms of signal models. [Rabiner, 1989]
Action recognition refers to the process of understanding and classifying meaningful movements, by studying those signals. There are many diferent works in the field of action recognition.
Researchers have already prototyped wearable computer systems that use acceleration, audio, video, and other sensors to recognize user activity.
Research has motivated activity recognition in numerous domains:

- Elderly Care. Monitoring the Activities of Daily Living estimate quality of self-care and enable assisted living. [Hristijan and Matjaz, 2011]
- Proactive Healthcare. Fighting obesity, coaching toward more active life. [Sun et al., 2010, Li et al., , Kwapisz et al., 2010]
- Psychiatry. Correlation of activities with moods, mood swings, manic depressions.
- Security / Workflow Monitoring. Tracking maintenance staff, crowds, managing accountability, emergency/rescue operations, security concerning questions.
- Groupware. Sharing activity information in groups / over social networks.

- Memory Support. Managing diaries, auto-filling journals for later accounting.

Besides the areas cited above, where the usage of the smartphones is increasing, especially in the elderly care area, there's also a growing number of applications for smartphones that are already using GPS positioning to track user's activity. The user can choose an activity from a predefined list, and with the positioning and the predefined values for each activity, concerning calories, distances, altitudes, accelerations (calculated in the distance/time basis) the smartphone can track and measure all the variables for that user/activity.

We can cite as example Sports Tracker (`http://www.sports-tracker.com/`), an old product from Nokia that now as been developed to all major plataforms. The free app is available for Android, iPhone, Windows Phone, Nokia N9 and Nokia Symbian devices. Sports Tracker started in 2004, while his developers worked at Nokia. The application was the first of its kind for mobile phones, although at that time there were no phones with GPS, smartphones or Google Maps. With Sports Tracker it is possible to track and analyze performances, monitor progress, store all training data in a personal workout diary and sharing the workouts and photos with friends and other similar-minded people via the Sports Tracker community, Facebook and Twitter.

Another great and free app is RunKeeper (`http://runkeeper.com/`). RunKeeper is available for Android and iOS and is targeted as a "personal trainer in the pocket".
RunKeeper track the user's activities as running, walking, cycling, hiking, biking and more, using the GPS in phone. Deal with stats, history, progress and nofifications. Mesure heart rate with many available sensors, among a huge list of services.

Other GPS based tracking apps can be mencioned like Edumondo (`http://www.endomondo.com/`), or My Tracks, that belongs to Google Inc. (`https://code.google.com/p/mytracks/`,

A great advantage of the smartphone's globalization is the integration with other services such as social networks and cloud sharing. For example, there exists a synchonization app for RunKeeper to Evernote (`https://evernote.com/`).
Today, everything can be easily shared (despite the security issues concerning the "online status").

A major improvement in this kind of online applications would be the integration of the accelerometer data in the system in order to process activity recognition. The users were able to perform diferent activities without redefine the activity to be monitored.

1.2 Motivation

With the increase and constant evolution of the new technologies there are a huge number of sectors that are highly benefited. One of this sectores is clearly the civil protection (Figure 1.1).

The civil protection aims to prevent collective risks and the occurrence of serious accidents or resulting disasters; to attenuate collective risks and to limit its effect; to rescue and to assist people and other living beings in danger; to protect cultural and environmental assets and other assets of high public interest and to support the reestablishment of normality in the life of people living in the areas affected by serious accidents or disasters.

Figure 1.1: Firefighter equipped to wildfires and urban fires [Castro and Antunes, 2005]

The study of the cooperative interaction between teams of autonomous robots and teams of human agents is essentially a new scientific problem that imposes new challenges to be addressed.

In the context of first responder teams, the firefigter is a key "link", highly exposed to risks when in the field. Having mixed teams with robots can have operational advantages, but the firefighter need to be instrumented so that his state can be monitored.

It's necessary then to provide the firefighter the technology, integrated in human and robotic teams, in order to take full advantage of this human-machine paradigm, whether for its own survival, whether for the success of the team, by developing a system that can provide data about his context awareness and about explicit signs, previously defined.

1.3 Our approach

We will use an iPhone to perform action recognition, a task which involves identifying the physical action a user is performing. To implement this system we collected labeled accelerometer data from users as they performed the activities and gestures previous defined. Six activities were considered: walking, standing, ascending stairs, descending stairs, running, and an explicit movement, to exemplify emergency.
With the collected data we provide a training dataset. We then used that resulting training dataset to induce a predictive model for action recognition. This predicting model was finally implemented in the iPhone, in order to perform the classification online, and in real time, sendind the firefighter status to the Command Center Operations (CCO).

This work, iBombeiro, addresses the following problems:

- Having an iPhone, how can we use the acceleration data from iPhone sensors in order to have a powerful and lightweight system capable of action recognition?
- After the identification of the action recognition, how can we do the integration of the device as a node in the MANET, created in the purpose of the CHOPIN Project, in order to send messages with the firefighter status to the CCO?

This develoved device prototype for first responders will be named as ***iBombeiro***.

In the scope of the CHOPIN project, and in a meeting with the *Bombeiros Sapadores de Coimbra*, in order to define some procedures, has been observed that there is no specific movements or gestures to signal different situations. The possibility of using one or two simple gestures used in diving for emergencies was covered, so the movement to express emergency will be the one shown at Figure 1.2.

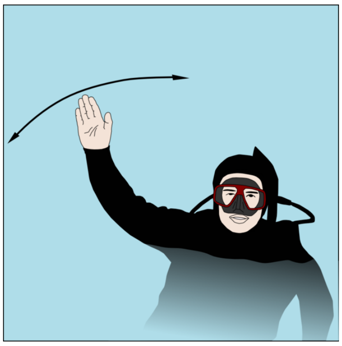


Figure 1.2: Emergency movement, used in diving[2]

The firefighters also use an emergency device, the Personal Alert Safety System - PASS (Figure 1.3).
PASS is a small device, heat and water projection resistant that the firefighter holds in the harness or in the Self-contained breathing apparatus (SCBA) and serves to alert others that he is in trouble. The device has a small beacon that indicates when connected, its proper functioning.
The emergency audible alarm can be triggered in two ways:

- Automatic - If the firefighter stay immobilized for more than 30 seconds;
- Manual - If the firefighter needs to launch an emergency alert.

Figure 1.3: Personal Security Alarm [Castro and Antunes, 2005]

In general, this equipment is suitable for work in potentially hazardous environments, especially when firefighters have to work alone.
In this work, we simulate the PASS behavior. In the automatic mode, when a firefighter is standing more than 30 seconds, a flag is activated, and the status of the firefighter changes to

[2] `http://en.wikipedia.org/wiki/Diver_communications`

"danger", and a sound is also triggered. The manual mode is simulated with the emergency movement. This work does not pretend to be a substitute of this kind of apparatus.

1.4 Thesis structure

This dissertation report is organized as follows.

In Chapter 1 we present an overview of the thesis, the motivation and the context where this work is fitted. We present also related work in action recognition, for different areas.

Chapter 2 presents the relevant background material, for the problem of online classification. After a brief introduction to the state of the art of classifiers, and after an introdution to the theory of the Dinamic Bayesian Networks, we present the features that we used in this work. Then we propose a Bayesian Model for iBombeiro, followed by the respective Bayesian Programming.
In this Chapter we also present the experimental setup that was developed in MATLAB, by explainning the algorithm used in the classifier. We finnish the Chapter by presenting the results obtained with the developed classifier, and we discuss those results.

In Chapter 3 we present the iPhone as a device for collecting data. We present the implementation of the classifier formulated in Chapter 2, in the iPhone (iBombeiro app). We also deal with the problem of the network communication, by presenting three diferent ways of showing the status of the firefighter: sending a message to the CCO via MANET, sending a message via webservice, saving the message in the iPhone database, for posterior analysis.

Chapter 4 presents a new Bayesian Model introduced in order to deal with more variables to represent a personalized implementation of the iBombeiro. Due to time constrains we present only the mathematical validation.

Finally, in Chapter 5 we end this work with our concluding remarks and provide directions for future work.

Chapter 2

The problem of online classification

What is classification? Classification is the process of using a model to predict unknown values (output variables), using a number of known values (input variables).
In order to perform classification, first it's necessary to model the relationship between the input variables and the output variables we are predicting. This process involves learning a model using data in which both the input variables and the output variables are present. This model can subsequently be used on unseen data in which only the input data is present, in order to predict the output variables.

2.1 State of the art

There exists a wide range of features and algorithms for classification of action/gesture recognition with accelerometer derived features. Commonly used methods in the context of activity recognition include Naive Bayes classifiers, decision trees, nearest neighbor methods, support vector machines (SVM), Neural Networks (NN) and Hidden Markov Models (HMM).

One of the first dynamic gesture recognition systems was developed by [Hofmann et al., 1997]. They present a method for the recognition of dynamic gestures using Hidden Markov Models (HMM). It addresses the problems of segmentation (the extraction of the beginning and ending of a gesture) and recognition of the preprocessed and segmented gesture data.The recognition took hours to complete.

In [Pärkkä et al., 2006], methods used for classification of everyday activities like walking, running, and cycling are described. The aim of the study was to find out how to recognize activities, which sensors are useful and what kind of signal processing and classification is required. A large and realistic data library of sensor data was collected. Sixteen test persons took part in the data collection, resulting in approximately 31 hours of annotated, 35-channel data recorded in an everyday environment. The test persons carried a set of wearable sensors while performing several activities during the 2 hours measurement session. Classification results of three classifiers are shown: custom decision tree, automatically generated decision tree, and artificial neural network. The classification accuracies using leaveone-subject-out cross validation range from 58 to 97% for custom decision tree classifier, from 56 to 97% for automatically generated decision tree, and from 22 to 96% for artificial neural network. Total classification accuracy is 82% for custom decision tree classifier, 86% for automatically generated decision tree, and 82% for artificial neural network.

[Sharma et al., 2011] presents the designing of a neural network for the classification of Human activity. The classification is done using only frequency domain feature. A Triaxial accelerometer sensor, housed in a chest worn sensor unit, has been used for capturing the acceleration of the movements associated. All the three axis acceleration data were collected at a base station PC via a CC2420 2.4GHz ISM band radio (zigbee wireless compliant), processed and classified using MATLAB. A fast training algorithm i.e. Levenberg-marquardt algorithm was used for training. The designed network (with 2 hidden layers) is giving a mean classification rate of 91.82 + 0.047% and 83.96 + 0.118 % for training and test data sets respectively. The mean classification rate with neural network classifier is also an improved one as compared to the previous results without the neural network classifier. The classification accuracy and use of wireless sensor node makes the system compatible to use in ubiquitous home and health environment.

[Prekopcsák, 2008] presented a real-time hand gesture recognition system, with a mobile phone, which identifies relevant parts in the continuous sensor data stream, and classifies them to the most probable gesture. He decided to create an automatic segmentation method without any buttons, which results more natural interaction. The mobile phone is used to capture accelerometer data, transmit it via Bluetooth to a nearby computer, where the data is analysed and gesture classification is performed in quasi real-time. Two distinct pattern recognition methods are compared, SVM and HMM. The results showed that both methods performed almost equally well. SVM achieved an average recognition rate of 96% and HMM reached 97.6%.

In [Just and Marcel, 2009], they address the problem of the recognition of isolated, complex, dynamic hand gestures. The goal was to provide an empirical comparison of two state-of-the-art techniques for temporal event modelling combined with specific features on two different databases. The models proposed are the HMM and Input/Output Hidden Markov Model (IOHMM), implemented within the framework of an open source machine learning library (www.torch.ch).

Accelerometer-based gesture recognition with a Nintendo Wii controller is explored by [Schlömer et al., 2008]. In their work, the sensor data is transmitted via Bluetooth from the controller device to a nearby PC, where the signal is processed. They also released the open-source Java library wiigee, which facilitates training and recognition of hand gestures, performed with a Wii controller.

[Sun et al., 2010] uses accelerometer-embedded mobile phones to monitor one's daily physical activities for sake of changing people's sedentary lifestyle. In contrast to the previous work of recognizing user's physical activities by using a single accelerometer-embedded device and placing it in a known position or fixed orientation, this work intends to recognize the physical activities in the natural setting where the mobile phone's position and orientation are varying, depending on the position, material and size of the hosting pocket. By specifying 6 pocket positions, this work develops a SVM based classifier to recognize 7 common physical activities. By introducing an orientation insensitive sensor reading dimension, they boost the overall F-score from 91.5% to 93.1%. With known pocket position, the overall F-score increases to 94.8%.

In [Niezen and Hancke, 2009] they evaluate the various gesture recognition algorithms currently in use, after which the most suitable algorithm was optimized in order to implement it on a mobile device. Gesture recognition techniques studied include hiddenMarkov models, artificial neural networks and dynamic time warping. A dataset for evaluating the gesture recognition algorithms was gathered using a mobile device's embedded accelerometer. The algorithms were evaluated based on computational efficiency, recognition accuracy and storage efficiency. The optimized algorithm was implemented on the mobile device to test the empirical validity of the study.
Initially a 4-state HMM was used, but the recognition accuracy of the HMM proved to be too low to be able to accurately compare themodel with the other algorithms. Using 8 states for the HMM increases the recognition accuracy to above 90%, which is approximately equal

to the accuracy obtained with the other algorithms.
For the Neural networks, a total of 72 of the 80 samples were correctly classified, for an overall accuracy of 90%.
The training and recognition procedures in Dynamic time warping DTW are potentially much faster than other techniques used for gesture recognition, such as HMMs and ANNs. Since there are no training steps or preprocessing required for the DTW algorithm, the raw gesture data could be evaluated directly. With the DTW algorithm, a total of 77 of the 80 samples were correctly classified, for an overall accuracy of 96.25%. This compares very well with the HMM algorithm with 8 states, and the DTW algorithm is therefore considered sufficiently accurate as to implement it. The gesture recognition algorithm was ported to the mobile device by making use of Nokia's Open C platform.

[Klingmann, 2009] implemented a gesture recognition application for the iPhone and its core features have been consolidated into a reusable Objective-C library. The iPhone' s built-in accelerometer is used to capture sensor data generated by hand movements. Discrete hidden Markov models, which work in combination with a vector quantiser, are the core of the recognition system. Various tests have been conducted in order to optimise certain application specific values, such as the codebook size and the initial prototype vectors. The final system reaches recognition results between 80 to 95 %.

LiveMove by AiLive (http://www.ailive.net/) is a commercial software product, which provides comparable functionality. It is aimed for game developers and provides gesture recognition methods for several accelerometer-equipped game controllers. Based on the Wiimote they developed a gesture recognition that employs state of the art recognition methodology such as HMM, filters and classifiers, and aim to optimise hand gesture recognition for the Wiimote.

[Yang and Ahuja, 1999] presented an algorithm for extracting two-dimensional motion fields of objects across a video sequence and classifying each as one of a set of a priori known classes. An application of the algorithm is in sign language recognition where an utterance is interpreted based on, for example, hand location, shape, an motion. The performance of the algorithm is evaluated on the task of recognising 40 complex hand gestures of American Sign Language (ASL). They used Time Delay Neural Networks (TDNN) to classify the hand gestures, based on motion trajectories.

[Kela et al., 2005] study accelerometer-based gesture recognition for controlling a television, a video recorder and lighting in a design environment. For a sensing device, they utilise a so-called SoapBox (Sensing, Operating and Activating Peripheral Box), and use HMM for the recognition.

In the research of [e Schühli, 2005], tests were done using computational methods to recognise the conducting gestures of a maestro, as timing, dynamic and expressive gestures from both hands. Taking advantage of image processing, tempo and dynamic were extracted using analysis of the vertical velocity from the right hand. The conducting gesture recognition system was implemented using Elman and T-CombNET neural network structure after an image processing stage.

[Sundaram and Mayol-Cuevas, 2009] presents a system aimed to serve as the enabling platform for a wearable assistant. The method observes manipulations from a wearable camera and classifies activities from roughly stabilized low resolution images (160x120 pixels) with the help of a 3-level Dynamic Bayesian Network and adapted temporal templates.

In [Li et al.,] a physical activity (PA) recognition algorithm for a wearable wireless sensor network using both ambulatory electrocardiogram (ECG) and accelerometer signals is proposed. First, in the time domain, the cardiac activity mean and the motion artifact noise of the ECG signal are modeled by a Hermite polynomial expansion and principal component analysis, respectively. A set of time domain accelerometer features is also extracted. A support vector machine (SVM) is employed for supervised classification using these time domain features. Second, motivated by their potential for handling convolutional noise, cepstral features extracted from ECG and accelerometer signals based on a frame level analysis are modeled using Gaussian mixture models (GMM). Third, to reduce the dimension of the tri-axial accelerometer cepstral features which are concatenated and fused at the feature level, heteroscedastic linear discriminant analysis is performed. Finally, to improve the overall recognition performance, fusion of the multi-modal (ECG and accelerometer) and multi-domain (time domain SVM and cepstral domain GMM) subsystems at the score level is performed. The classification accuracy ranges from 79.3% to 97.3% for various testing scenarios and outperforms the stateof-the-art single accelerometer based PA recognition system by over 24% relative error reduction on their 9-category PA database.

Bayesian networks are widely used to perform classification tasks, with the following advantages.

- Based on probability theory.
- Not a black box approach.
- Allows rich structure.
- Can mix expert opinion and data to build models.
- Backwards reasoning - in addition to predicting outputs given inputs, we can use output values to infer inputs.
- Support for missing data during learning and classification.

Based in some of the works cited above, and due to its particular characteristics and properties we choose Dynamic Bayesian Network models to form the basis of the pattern recognition process, for iBombeiro.

2.2 Feature Pre-Processing

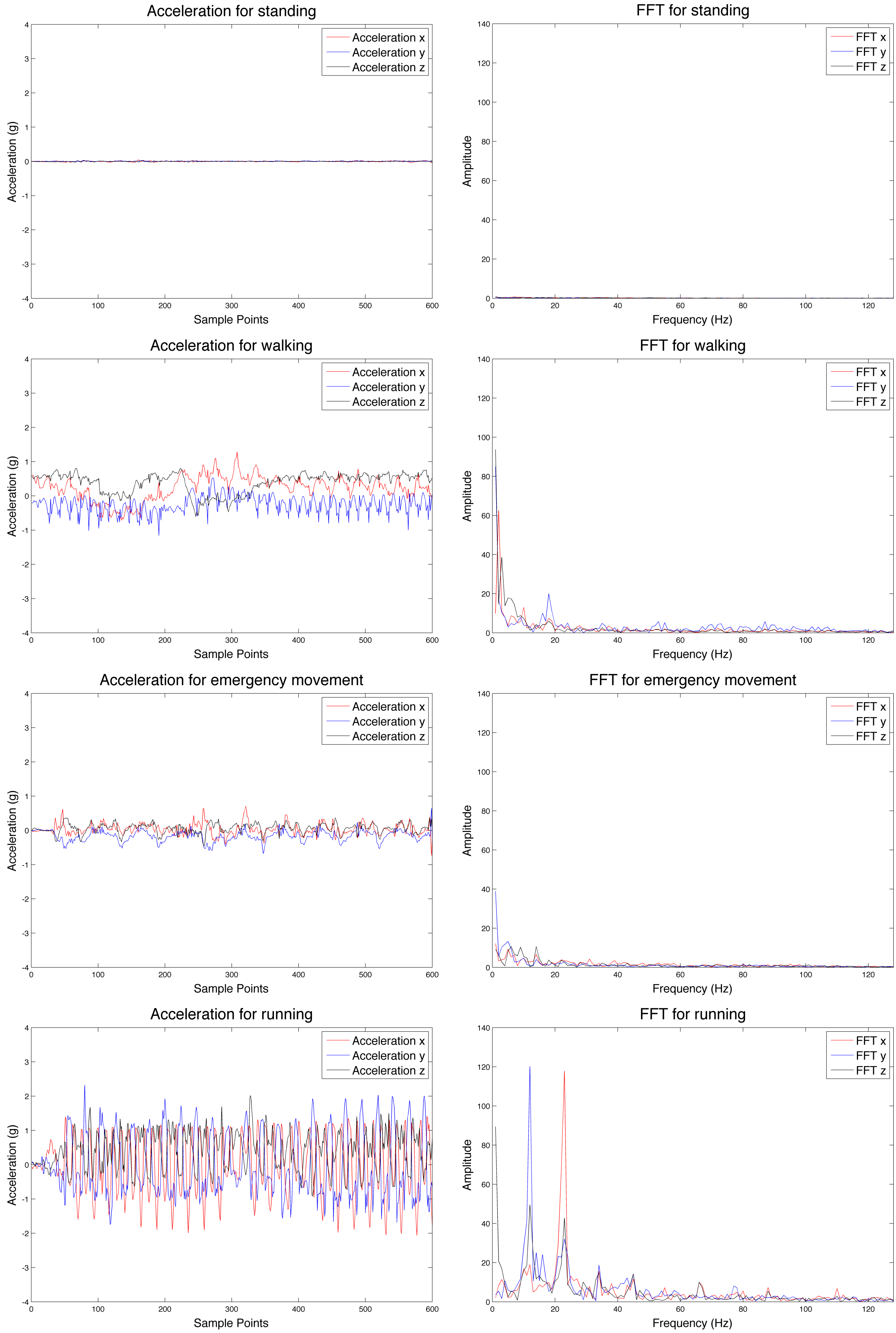
Any classification method uses a set of features or parameters to characterize each object, where these features should be relevant to the task at hand.
We consider here methods for supervised classification, meaning that a human expert both has determined into what classes an object may be categorized and also has provided a set of sample objects with known classes. This set of known objects is called the training set because it is used by the classification programs to learn how to classify objects.

In this work a number of summary statistics were derived from the accelerometer data. These statistics (henceforth referred to as "features") serve the purpose of presenting the accelerometer data in ways believed to hold greater discriminatory power between movements and context, compared to the raw acceleration values. Such features serve as the data space on which the proposed method performs the classification.
Based in previous work [Khoshhal et al., 2010, Khoshhal et al., 2012, Khoshhal and Dias, 2012], we decided to apply the fourier transform to the acceleration data, read from iPhone sensors.

The Fourier transform is among the most widely used tools for transforming data sequences and functions (single or multi-dimensional), from what is referred to as the time domain to the frequency domain. Any continuous, periodic function can be represented as a linear combination of sines and cosines. A sine is a function of the form: $Asin(2\pi\omega t + \phi)$, where A is the amplitude, ω is the frequency measured in cycles (or periods) per second, and ϕ is the phase, which is used for getting values other than 0 at t = 0. A cosine function has exactly the same components as the sine function, and can be viewed as a shifted sine (or more accurately - a sine with phase $\frac{\pi}{2}$).
The result of applying the Fourier transform to a function is called the frequency spectrum or the power spectrum of the function.
The Discrete Fourier Transform (DFT) is the equivalent of the continuous Fourier Transform for signals known only at instants separated by sample times T (i.e. a finite sequence of data).
The DFT is defined as:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-i2\pi nk}{N}}, k = 0..N-1 \tag{2.1}$$

Where $X(K)$ represents an array of complex frequency-domain data, $x(n)$ an array of complex time-domain data, n an index of time steps, k an index of frequency spectral lines and N represents the size of the data arrays (i.e. the size of the sequence).
DFT decomposes a sequence of signals into a series of components with different frequency or time intervals. This operation is useful in many fields, but in most cases computing it directly from definition is too slow to be practical.
One of the most appealing aspects of the DFT is the existence of an eficient procedure for calculating it, using $Nlog_2N$ complex operations, rather than N^2 operations required for the naive algorithm. The algorithm for fast DFT, is known as the FFT (the Fast Fourier Transform). It takes advantage of symmetry properties of the complex roots of unity, and uses repeated clever partitioning of the input sequence into two equally long subsequences, each of which can be separately (and quickly) processed. In order to take full advantage of the repetitive partitioning into equal two parts, the original sequence needs to be of length or periodicity which is a power of 2.
In Figure 2.1 the signals for the acceleration values, for each action, and their FFT is shown.



Acceleration for standing
Acceleration x
Acceleration y
Acceleration z
Acceleration (g)
Sample Points
FFT for standing
FFT x
FFT y
FFT z
Amplitude
Frequency (Hz)
Acceleration for walking
Acceleration x
Acceleration y
Acceleration z
Acceleration (g)
Sample Points
FFT for walking
FFT x
FFT y
FFT z
Amplitude
Frequency (Hz)
Acceleration for emergency movement
Acceleration x
Acceleration y
Acceleration z
Acceleration (g)
Sample Points
FFT for emergency movement
FFT x
FFT y
FFT z
Amplitude
Frequency (Hz)
Acceleration for running
Acceleration x
Acceleration y
Acceleration z
Acceleration (g)
Sample Points
FFT for running
FFT x
FFT y
FFT z
Amplitude
Frequency (Hz)

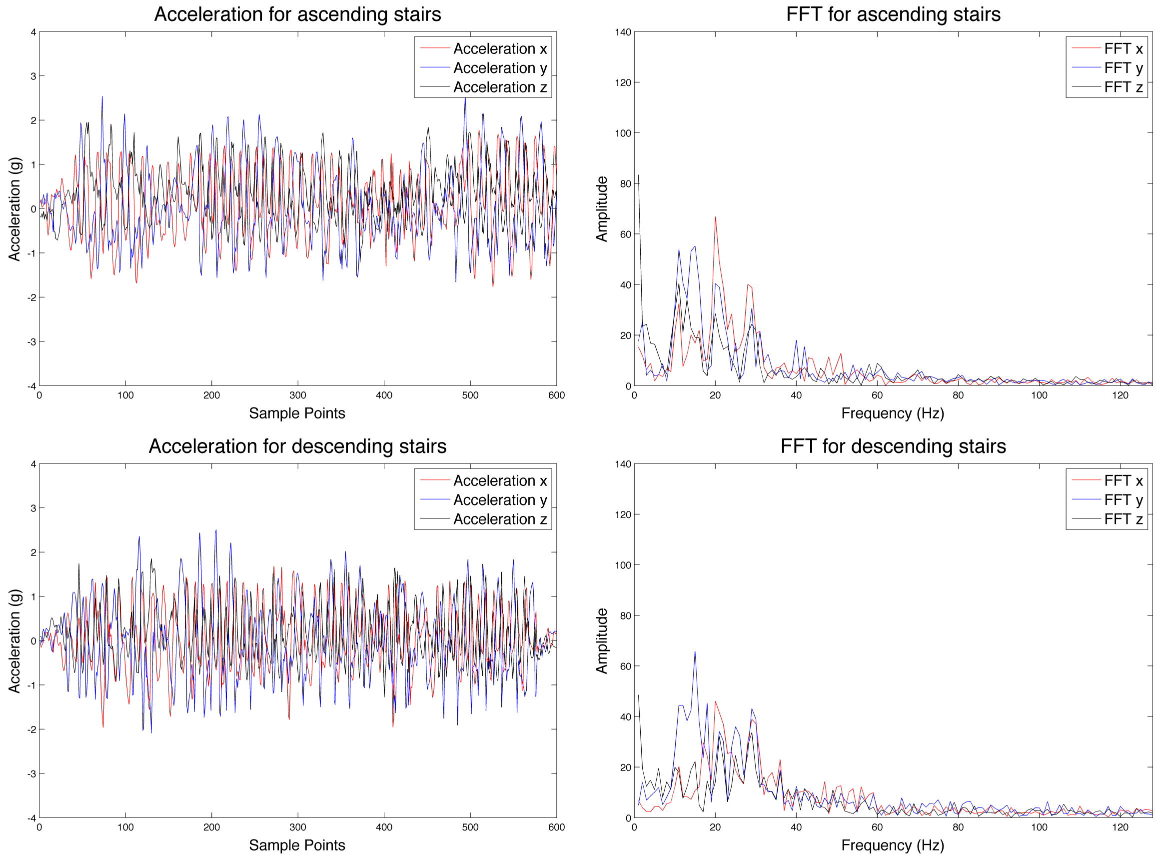
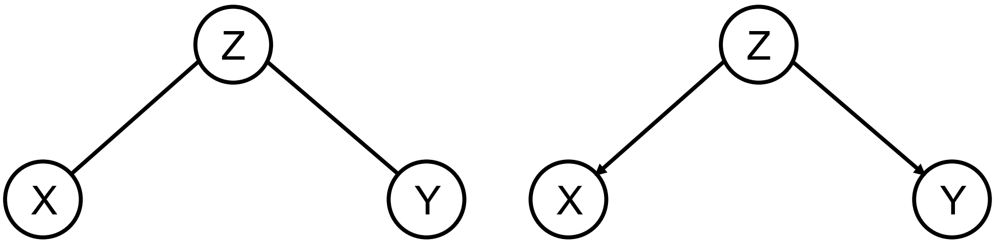


Figure 2.1: User Acceleration and FFT for specified actions

2.3 Action model

Probabilistic networks, also known as Bayesian networks (BN's), belief networks or causal networks are already well established as representations of domains involving uncertain relations among a group of random variables. A Bayesian Network is a specific type of graphical model that is represented as a Directed Acyclic Graph (DAG). Nodes in DAG are graphical representation of objects and events that exists in the real world, and are usually termed variables or states. Causal relations between nodes are represented drawing an arc (edge) between them. For any given edge between the variables (nodes) X and $X2$, if there is a causal relationship between the variables, the edge will be directional, leading from the cause variable to the effect variable. Figure 2.2(a) illustrates how X and Y are conditionally independent, given variable Z (note that this is not the same as saying that X and Y are tottaly independent. For each variable in the DAG there is probability distribution function (pdf), which dimensions and definition depends on the edges leading into the variable.



(*a*) (*b*)

Figure 2.2: User Acceleration for (a) A graphical model illustrating condicional independence between variables X and Y given the variable Z, (b) Simple Bayesian Network

In short, a graphical mode consists of variables (nodes) $V = 1, 2...K$ with a set $\mathbf{D}$ of dependencies (edges) between the variables and set $\mathbf{P}$ of probability distribution functions for each variable.
BN's can be defined as aspecial case of more general class called graphical models in which nodes represent random variables, and the lack of arc represents conditional independence assumption between variables. Figure 2.2(b) illustrates a simple Bayesian Network.
BN's do not provide direct mechanism for representing temporal dependencies. In attempting to add temporal dimension into the BN model various approaches has been suggested. Frequent names used to describe this new dimension in BN models are "temporal" and "dynamic". Dynamic Bayesian Networks should be a name of a model that describes a system that is dynamically changing or evolving over time.

Probability theory is based on three basic axioms:

1. $0 \leqslant P(X) \leqslant 1$
2. $P(X) = 1$ if and only if X is certain
3. If X and Y are mutually exclusive, then $P(X \cup Y) = P(X) + P(Y)$

and a fundamental rule of probability calculus:

$$P(X, Y) = P(X|Y)P(Y) \tag{2.2}$$

where $P(X, Y)$ is the probability of the joint event $X \cap Y$ [Mihajlovic and Petkovic, 2001]

Bayes' rule is a rigorous method for interpreting evidence in the context of previous experience or knowledge.

The general form of the Bayes' Rule can be written as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.3}$$

Where:

$P(A|B)$ is called the posterior

$P(B|A)$ is the likelihood of observations given the model.

$P(A)$ is the Prior

$P(B)$ is the evidence (used as a normalization factor).

In essence, Bayes' rule is used to combine prior experience (in the form of a prior probability) with observed data (in the form of a likelihood) to interpret these data (in the form of a posterior probability). This process is known as Bayesian inference.

Bayesian inference is not guaranteed provide the correct answer. Instead, it provides the probability that each of a number of alternative answers is true, and these can then be used to find the answer that is most probably true. In other words, it provides an informed guess.

In iBombeiro, the inicial prior (for $t = 0$) is an uniform prior. The inicial prior is a matrix of order 1 x N where all the values are $1/N$ (N is the number of states).

For $t > 0$, $prior = posterior(t = t - 1)$

2.3.1 iBombeiro Action Recognition Model

Graphical models are a marriage between probability theory and graph theory. They provide a natural tool for dealing with two problems that occur throughout applied mathematics and engineering - uncertainty and complexity - and in particular they are playing an increasingly important role in the design and analysis of machine learning algorithms. Fundamental to the idea of a graphical model is the notion of modularity - a complex system is built by combining simpler parts. Probability theory provides the glue whereby the parts are combined, ensuring that the system as a whole is consistent, and providing ways to interface models to data. The graph theoretic side of graphical models provides both an intuitively appealing interface by which humans can model highly-interacting sets of variables as well as a data structure that lends itself naturally to the design of efficient general-purpose algorithms[Jordan, 1999]

Bayesian models provide a powerful language for describing and evaluating hypotheses about perceptual behaviors.
The Bayesian model for iBombeiro will be developed using a formalism known as Bayesian Programming, consolidated by [Bessire et al., 2008].
In Bayesian Programmming the necessary information for completely specifying a probabilistic model consists of:

1. The relevant variables;
2. The conditional dependencies between these variables;
3. The parametric forms of the associated probability distributions;
4. Their associated parameter values.

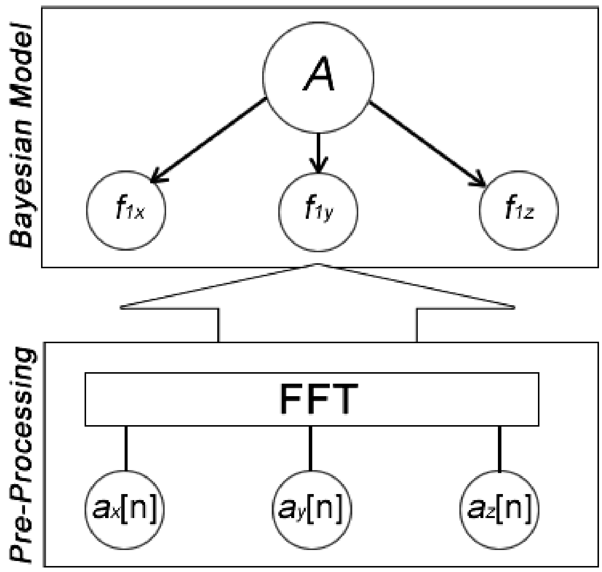


Figure 2.3: iBombeiro Bayesian Model. For each a[n] an independent FFT is computed

A Bayesian Program (BP) contains two parts:

- a description that is the probabilistic model of the studied phenomenon or the programmed behaviour;
- a question that specifies an inference problem to be solved using this model.

A description contains two parts:

- a specification part that formalizes the modeler's knowledge;
- an identification part where free parameters are learned from experimental data.

Finally, the specification is constructed from three parts:

- a selection of relevant variables to model the phenomenon;
- a decomposition, whereby the joint distribution on the relevant variables is expressed as a product of simpler distributions exploiting conditional independence between variables;
- the parametric forms in which either a given mathematical function or a question for another BP is associated with each of the distributions appearing in the decomposition.[Colas et al., 2010]

In our model, and reporting to Figure 2.4 we have a stream of data, in form of acceleration $ax[n]$, $ay[n]$ and $az[n]$, where an independent FFT is performed in each acceleration. We can treat the acceleration variables independently because we use the acceleration from the Device Motion (see section 3.1.1). It means that the acceleration does not have the component of gravity, so the values of the accelerations are not dependent (i.e. if we move the acceleration in $Y-axis$ the value of the acceleration of the $X-axis$ will not change. Computacionally this gives a major advantage, otherwize cumputacional weight would be high.

The Figure 2.4 shows the Bayesian program used in iBombeiro.

Bayesian Program: iBombeiro Action Recognition Model

program / *description* / *specification*

Variables:
$f_n \in \mathcal{F}$: First fourier coeficient variables.
A : Action variable.
Decomposition:
$P(A, f_1, \cdots, f_n) = P(f_x|A)P(f_y|A)P(f_z|A)P(A)$
Formulation:
$P(A)$: *Stochastic Matrix.*
$P(f_n|A)$: *Gaussian Distribution.*
Identification: Gaussian parameters μ and σ based on training dataset Ω.
Question: $P(A|f_n)$ answered using *Maximum A Posteriori*, a method for Bayesian inference.

Figure 2.4: Bayesian Program: iBombeiro Action Recognition Model

Reporting to the Bayesian Program in Figure 2.4 we start by defining the relevant variables for the problem.

- $f_n \in \mathcal{F}$ is a random variable that denotes the first fourier coeficient calculated independently for the acceleration measurements, for all the axis and for all the classes, and whose domain is real.

- $\zeta \equiv \{Walking, Standing, Running, Upstairs, Downstairs, Mov1\}$ is a random variable denoting the chosen actions and it can have six possible diferent states.

In the decomposition we have the expression of the joint probability distribution as the product of simpler distributions, i.e. decompose join probabilities in a product of simpler conditional probabilities where the independence of the variables is explored. Decomposition is often assisted by graphics.
In this case, the decomposition of Figure 2.4, it establishes the dependencies according to Figure 2.3.

In the formulation step, the kind of parametric probability function that is adequate to represent the distribution of the decomposition is chosen.
As the domain of f_n is real, it is common to define the distribution as a gaussian density, moreover activities are expected to be repeatable therefore a normally distributed
So, for iBombeiro Bayesian Model we will have (as shown in Equation 2.4), for each action, the following gaussian destribution.

$$P(f1_x|\xi) = \left\{ \begin{array}{ll} N(0.2015, 0.1115), for & \xi = 1 \\ N(8.8507, 4.6458), for & \xi = 2 \\ N(2.9079, 1.3355), for & \xi = 3 \\ N(19.2620, 3.9258), for & \xi = 4 \\ N(14.2720, 4.7609), for & \xi = 5 \\ N(12.9820, 3.4223), for & \xi = 6 \end{array} \right\}, P(f1_y|\xi) = \left\{ \begin{array}{ll} N(0.1441, 0.0451), for & \xi = 1 \\ N(6.3170, 3.1266), for & \xi = 2 \\ N(5.3208, 2.7935), for & \xi = 3 \\ N(20.3560, 5.1274), for & \xi = 4 \\ N(15.1590, 5.9118), for & \xi = 5 \\ N(15.6460, 4.4806), for & \xi = 6 \end{array} \right\}$$

$$P(f1_z|\xi) = \left\{ \begin{array}{ll} N(0.1023, 0.0524), for & \xi = 1 \\ N(12.4900, 5.6156), for & \xi = 2 \\ N(3.0844, 1.0834), for & \xi = 3 \\ N(15.4830, 5.0610), for & \xi = 4 \\ N(12.5210, 4.2820), for & \xi = 5 \\ N(10.1690, 3.0088), for & \xi = 6 \end{array} \right\} \tag{2.4}$$

where $1 = Standing$, $2 = Walking$, $3 = Emergency\ Movement$, $4 = Running$, $5 = Ascending\ stairs$ and $6 = Descending\ stairs$.

Figure 2.5 shows the gaussian distribution for each of the actions, where the gaussian parameters μ and σ are based on the training set.

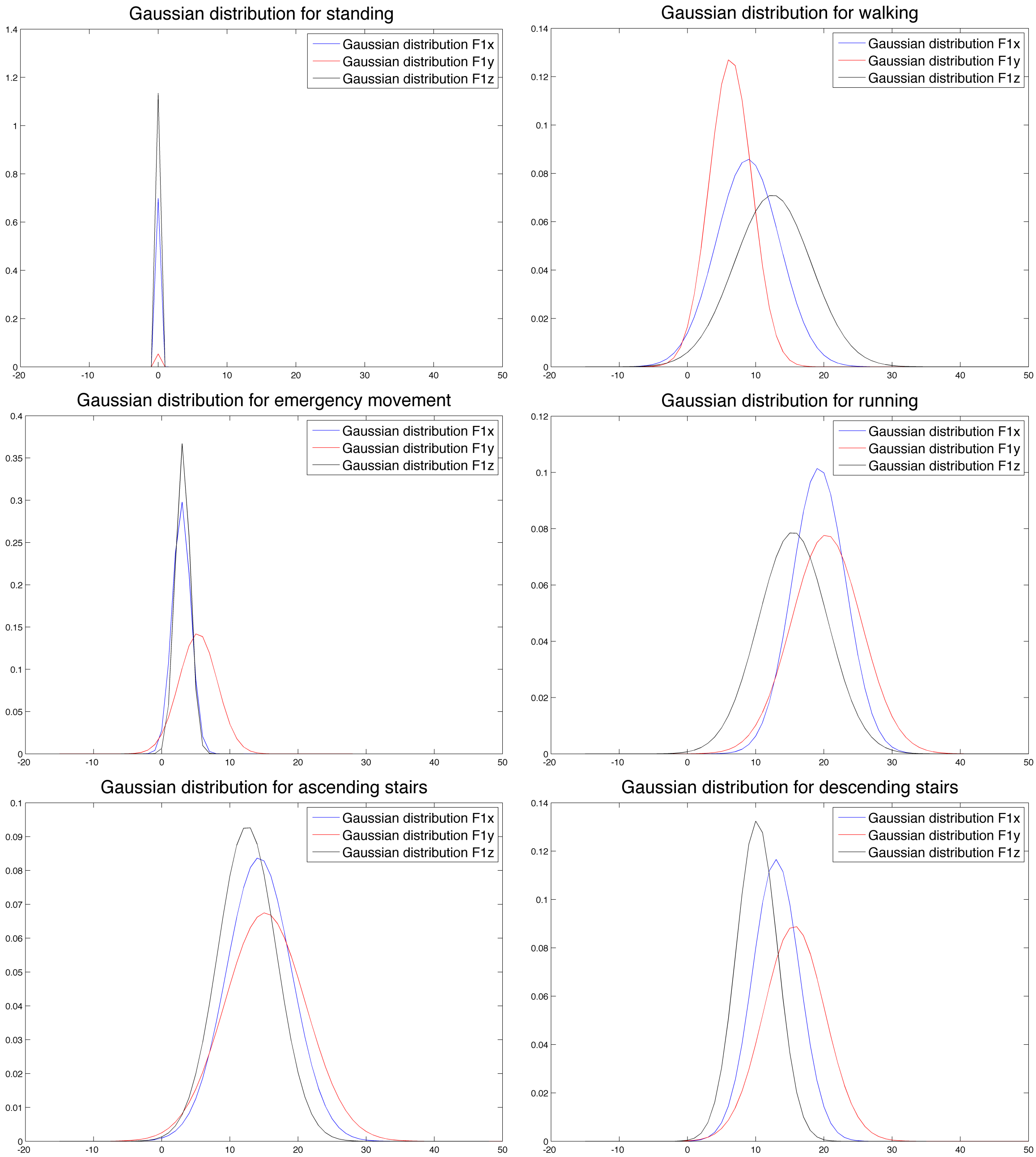


Figure 2.5: Gaussian distribution for specified actions

As for the question, "how can we maximize the posterior distribution?", it is answered wih MAP (Maximum A Posteriory).

$$\hat{A}_{MAP} = argmax_A(P(f_x|A)P(f_y|A)P(f_z|A)P(A)) \tag{2.5}$$

By replacing the prior at instant t, by the posterior calculated at instant $t = t - 1$, as the number of iterations advances, the MAP is expected to converge for a given state.

2.3.2 Extended iBombeiro Recognition Model

One major update to iBombeiro would be the development of a personalised model. With this personalisation each firefighter would have his own personal training dataset recorded on a server. The firefighter could choose an iD from the iPhone and his data would be downloaded, where the configuration file would have a record of all his personal training parameters. That personalisation of the iBombeiro app is expected to increase the acuraccy of the present results.

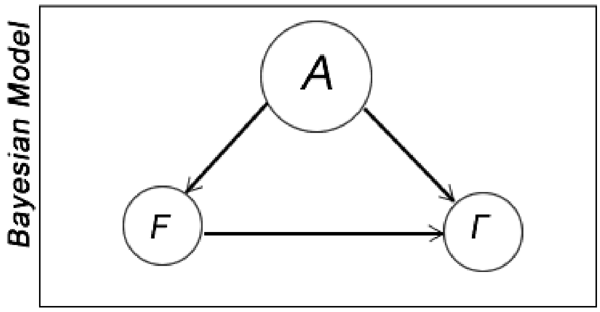


Figure 2.6: Extended iBombeiro Bayesian Model.

The Figure 2.6 presents the Bayesian model for personalised iBombeiro. The feature node it's a supernode, comprising different variables, independently distributed.
The Figure 2.7 shows the Bayesian Programming for the Extended iBombeiro Recognition Model. This model was not implemented due to time constrains, therefore in this section we only formulate the model, placing the implementation as future work. The decomposition

Bayesian Program: Extended iBombeiro Action Recognition Model

program / *description* / *specification*

Variables:
$\gamma \in \Gamma$: person identity
$f_n \in \mathcal{F}$: First fourier coeficient variables.
A : Action variable.
Decomposition:
$P(A, \gamma, f_1, \cdots, f_n) = P(F|\gamma)P(F, \gamma|A)P(A)$
Formulation:
$P(A)$: *Stochastic Matrix.*
$P(f_n|\gamma)$: *Gaussian Distribution.*
$P(f_n, \gamma|A)$: *Kernel of Gaussian Distributions.*
Identification: Gaussian parameters μ and σ based on training dataset Ω.
Question: $P(A|f_n, \gamma)$ answered using *Maximum A Posteriori*, a method for Bayesian inference.

Figure 2.7: Bayesian Program: Extended iBombeiro Action Recognition Model

of the new Bayesian program presents us with a new variable, one whose states denote each person identity. For the extended iBombeiro model, the proposed decomposition of

Figure 2.7 establishes the dependencies according to Figure 2.6, where the new learning will deal with multivariate distribution learning.

2.3.2.1 Learning Personalised iBombeiro Action Data

As for the learning of the personalised model, the firefighter could train his own set of data. The process starts with the firefighters identifying himself and the desired action on the iPhone. If the action already exists on the server, the app will ask if he wants to overwrite existing training sets. The app will wait a few seconds until detecting the firefighter is standing and ready for performing the action. Then, a beep sound is issued to indicate the beginning of the training session. The training can be done either by assigning a predefined time to the action to be done, or by applying a method to verify the training distribution convergence.

By using the latter we expect have more assertive results.

System will be capable to computing a gaussian, in specific time intervals. If the diference between the computed gaussian and the existing trained one is over a certain threshold, the learning continues, and after the specified time, a new training distribution is computed. If the diference between the gaussians is below a certain threshold the training phase can stop, an event signaled by another selective sound.

We can use Kullback-Leibler (KL) information, to infer about the convergence of the gaussians.

The Kullback-Leibler (KL) information is a fundamental quantity in probability and statistics that measures the similarity of two distributions. Suppose we have two models defined on the same sample space, one with with distribution F and a second one with distribution G.

Then the expectation, for a discrete variable

$$I^{KL}(F;G) = \sum f(x_i) log \frac{f(x_i)}{g(x_i)} \tag{2.6}$$

is the Kullback-Leibler information of G with regard to F. The new training set for that action/firefighter is uploaded and recorded to a server.

In the end of the training phase, a beep will warn the firefighter that the traning of the specified action is over.

The Bayesian program of the personalised ibombeiro, as formulated, enables not only a better accuracy by providing a personalised training dataset for each actor, but also another potentially innovative application. It will allow making inference about the identity of a given person, based on observed actions. The integration of this new system could bring enormous benefits to the network, an issue which is discussed latter.

2.4 Experimental setup in MATLAB

The experimental setup, with data collected from the iPhone (see Section 2.8), was developed in MATLAB, in order to create the classification model. First a GUI was developed in MATLAB, in order to study all the signals returned from the sensors. Then we developed a classifier in MATLAB.

2.4.1 Data Visualization in MATLAB

A Graphical User Interface (GUI) was developed in MATLAB, to perform the visualization of the data collected from the iPhone sensors (Figure 3.1).

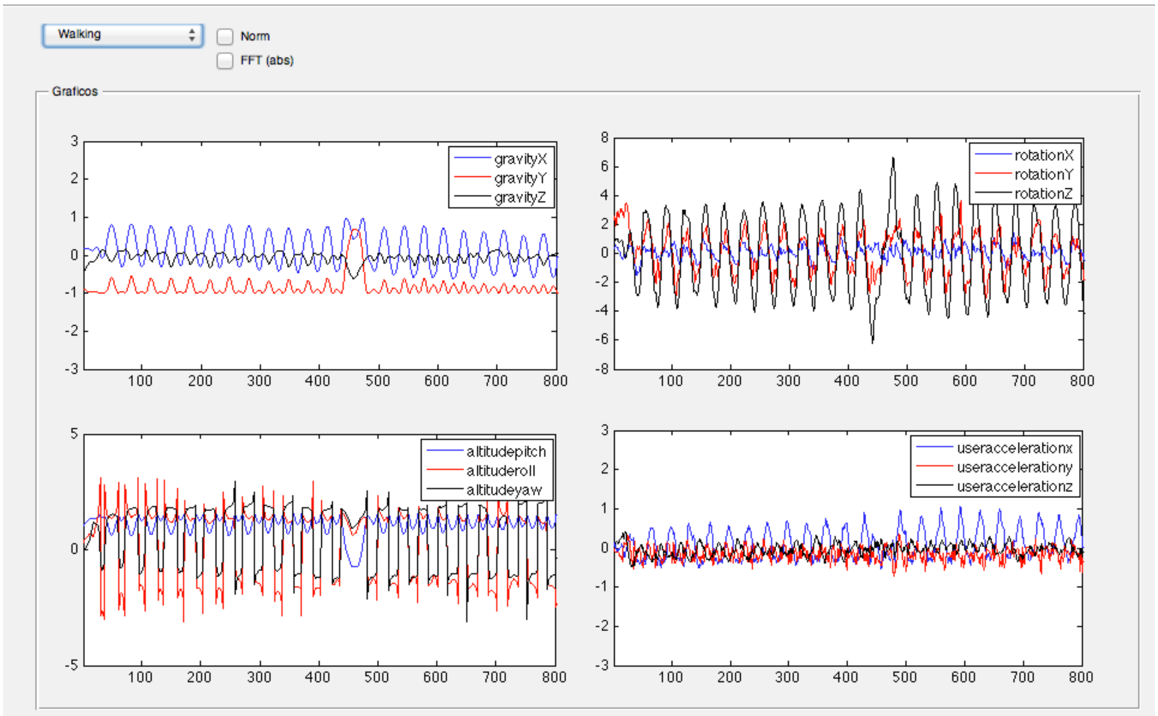


Figure 2.8: Data Visualizer in @Matlab

In the data visualizer it is possible to see:

- The gravity;
- The rotation rate;
- The altitude;

- The user acceleration;
- The FFT for each of the actions.

The GUI was prepared to show the different kinds of data.
In our case we only used the acceleration data, by interpreting this to be the most discriminatory variable. This minimization was also an objective: the reduction of the computacional load, due to the battery and CPU limitations of the device used.
However, for future work, we propose the development of techniques to assess discriminant data types, and the use of more variables in our model, in order to have a good value in the racio factor improvement versus computational load, in the classification.

2.4.2 Classifier

There are two phases to constructing a classifier, the training phase and the recognition phase. In the training phase, the training set is used to decide how the parameters ought to be weighted and combined in order to separate the various classes of objects. In the application phase, the weights determined in the training set are applied to a set of objects that do not have known classes in order to determine what their classes are likely to be.

In our work we use the first fourier coeficient, as feature.
Applying more features may bring benefits to the recognition accuracy in the case of computing on the powerful computers. However, when we are trying to implement these features inside the resource and power limited mobile phones, we should try to avoid the features that need complex computing workload, since it consumes much of computing resources and energy, which is critical to the user experience and acceptance of such application.

Our data will be validated using cross-validation.
Cross-validation is primarily a way of masuring the predictive performance of a statistical model, testing the same model on a set of data not used in estimation.

For the training phase our algorithm is the one presenting below.

```
Data: Data from iPhone's sensors
Result: training set
initialization;
N=Number of states;
for i = 0 to N do
    while not at end of data do
        for each acceleration x, y and z do
            read values from current window;
            compute fft;
            save first dynamic characterist (f1) in array;
        end
    end
    compute mean and standard deviation from array of f1;
    save the mean and the standard deviation in a file for current state (training set);
end
```

Algorithm 1: Algorithm for training phase

For the recognition phase the following algorithm is used.

```
Data: Data from iPhone's sensors
Result: Matrix of probabilities
initialization;
N=Number of states;
prior =[1/N 1/N...1/N];
for i = 0 to N do
    while not at end of data do
        for each acceleration x, y and z do
            read values from current window;
            compute fft, and calculate first fourier coeficient characterist (f1);
            compute probabilities from likelihood distributions;
        end
        posterior = distProb_x * distProb_y * distProb_z;
        normalization of the probabilities;
        prior=posterior;
    end
end
```

Algorithm 2: Algorithm for the recognition phase

2.4.3 Results and discussions

As we can see in the Table 2.1, the results are interesting, although the action with the worst results was the emergency movement. The observed confusion can be explained due to the fact the generated acceleration features from walking and this movement are very similar. More specifically, both generate back and fourth movements, which in our particular dataset are performed with similar accelerations.

When this movement is performed with a higher cadence, the confusion will translate to the gesture running.

Climbing stairs is other movement that does not have a very good recognition rate.

	Standing	**Walking**	**Emergency Movement**	**Running**	**Ascending stairs**	**Descending stairs**
Standing	1.00	0.00	0.00	0.00	0.00	0.00
Walking	0.00	0.9904	0.0096	0.00	0.00	0.00
Emergency movement	0.00	0.9474	0.0526	0.00	0.00	0.00
Running	0.00	0.0312	0.00	0.8594	0.1094	0.00
Ascending stairs	0.00	0.0625	0.00	0.00	0.9141	0.02335
Descending stairs	0.00	0.0779	0.00	0.00	0.2208	0.7013

Table 2.1: Matlab results

For dissemination purposes, we have tested yet another kind of movement, here named Test Movement. Imagining a person standing in the Y-axis, and with the Z-axis pointing to front, the test movement is performed parallel to the Z-axis, with the forearm making an angle of 90 degrees with the arm.

The following table (Table 2.2) shows the results with the introduction of this new action.

	Standing	**Walking**	**Emergency Movement**	**Running**	**Ascending stairs**	**Descending stairs**	**Test Movement**
Standing	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Walking	0.00	0.00	0.04785	0.00	0.00	0.00	0.9952
Emergency movement	0.00	0.2632	0.03	0.00	0.00	0.00	0.7068
Running	0.00	0.02343	0.00	0.8594	0.1094	0.00	0.0078
Ascending stairs	0.00	0.0391	0.00	0.00	0.9141	0.0234	0.0234
Descending stairs	0.00	0.00	0.00	0.00	0.2208	0.6753	0.1039
Test Movement	0.00	0.00	0.00	0.00	0.00	0.9186	0.0814

Table 2.2: Matlab results for one more action

With the introduction of the new action (test movement), results have become worst. Detailed analysis found that the two performances carried out for the new action gesture yields completely different signals, which naturally confuses the classifier. This problem is expected to be solved when using the identity variable, previously presented in the extended model in section 2.3.2.

Each person has its own dataset, and the identity variable works as a stamp printed in their actions. During the classification process, this discriminatory method, would expectadely improve the classification. In the next chapter it will be presented the iPhone implementation of the classifier.

Chapter 3

Iphone Implementation

In the scope of this work it will be used, for data acquisition and for online classifier, an iPhone 4S.

3.1 iPhone as a device for collecting Data

The fist approach to the problem was to collect sample data, for all the movements and physical activities, previously defined.
A primary app was developed in the iPhone platform, that consists of an operating system and a set of tools developed by Apple, using Objective-C. The iPhone platform targets portable, multi-touch devices and is currently only available on the Apple iPhone, iPod touch and iPad.
Objective-C is a programming language, which is sleeted by Apple for developing the application for iPhone and Mac systems, and it's is defined as a small but powerful set of extensions to the standard ANSI C language. The app *collectData* was able to capture the device motion data with a frequency of 30Hz.

3.1.1 Sensors

In the last few years, the fast development in sensor technology made it possible to put small accelerometer sensors into everyday devices. The availability of MEMS (Micro-Electromechanical System) 3-axis linear accelerometers allows for the design of an inexpensive mobile gesture recognition system. A single integrated tri-axial accelerometer provides better accuracy than traditional orthogonally mounted accelerometers, and reduces re-calibration frequency. Wearable inertial sensors are a low-cost, low-power solution to track gestures and, more generally, movements of a person.[Prekopcsák, 2008]

The embedded triaxial accelerometer inside the iPhone can continuously sample the experienced accelerations at each sampling interval and produce 3-D acceleration readings $A = (ax; ay; az)$, which are measures of the acceleration experienced in the three orthogonal axes: $X - axis$, $Y - axis$ and $Z - axis$ (Figure **??**)

The gyroscope measures the rate of rotation around the three axes.

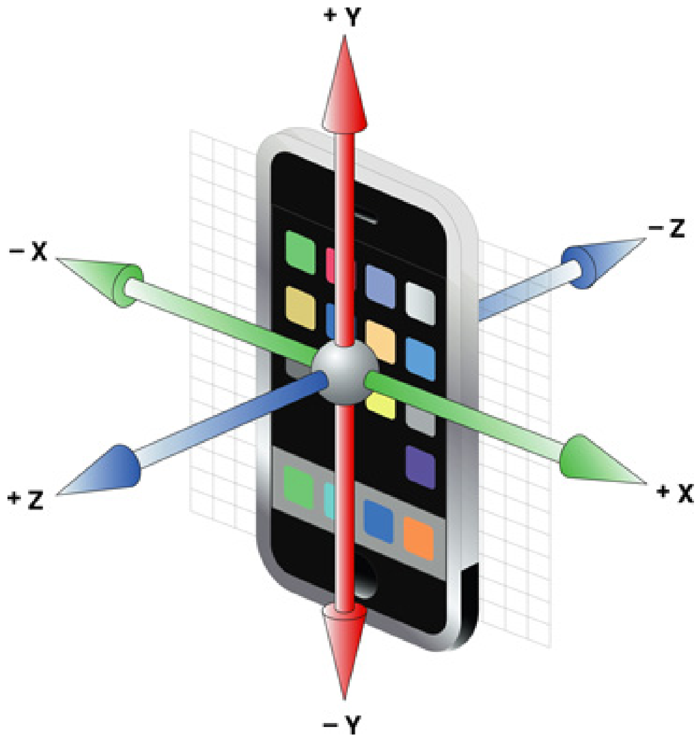


Figure 3.1: The three axes on the iPhone.

The Core Motion framework (CoreMedia.framework) provides a single set of interfaces for accessing all motion-based data available on a device. The framework supports accessing both raw and processed accelerometer data using a new set of block-based interfaces. For devices with a built-in gyroscope, it is possible to retrieve the raw gyro data as well as processed data reflecting the attitude and rotation rates of the device.

Core Motion uses unique algorithms to process the raw data it collects, so that it can present more refined information. This processing occurs on the framework's own thread. Core Motion events are represented by three data objects, each encapsulating one or more measurements. [Apple, 2013]

- A CMAccelerometerData object captures the acceleration along each of the spatial axes, from the three accelerometers.
- A CMGyroData object captures the rate of rotation around each of the three spatial axes, from the iOS device's gyroscope.

- The CMDeviceMotion object encapsulates data from the accelerometer and gyroscope.

  For iOS devices with a magnetometer, the magneticField property contains the magnetic field vector.

A CMMotionManager object is the gateway to the motion services provided by iOS. These services provide an app with accelerometer data, rotation-rate data, magnetometer data, and other device-motion data such as attitude.
Code belows shows how a CMMotionManager object is called.

```
// Start device motion updates
 if ([self.motionManager isDeviceMotionAvailable])
   {
   //update 30 times per second
       [self.motionManager setDeviceMotionUpdateInterval:1.0/30.0];
       [self.motionManager startDeviceMotionUpdatesToQueue:[NSOperationQueue mainQueue]
       withHandler: ^(CMDeviceMotion *deviceMotion, NSError *error)
        {
           CMAttitude *attitude = motionData.attitude;
           CMAcceleration gravity = motionData.gravity;
           CMAcceleration userAcceleration = motionData.userAcceleration;
           CMRotationRate rotate = motionData.rotationRate;
         // handle data here
        }
  }
```

If DeviceMotion is available we define the update interval, $tDeviceMotionUpdateInterval:$ 1.0/30.0 for performing device motion updates at 30 Hz. Then we can access the properties of the CMDeviceMotion: A CMAttitude object, that represents a measurement of attitude; a CMAcceleration object, where we can have gravity and user acceleration (the total acceleration of the device is equal to gravity plus the acceleration the user imparts to the device (userAcceleration)) and a CMRotationRate structure that contains data specifying the device's rate of rotation around three axes.

3.1.2 Parse and app for collecting data

The app for collecting data is shown in Figure 3.2. After the inicialization of the app, the user can choose the activity to perform. For each activity a file is recorded containing all the information of the sensors. After all the activities were performed the user can push the button End in order to stop the updates from the Device Motion.

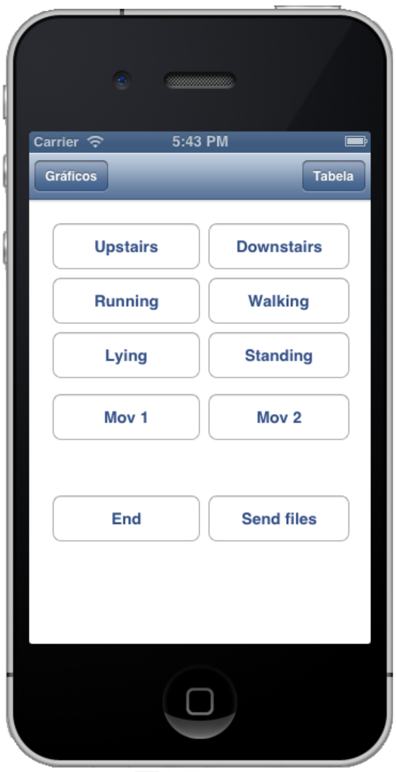


Figure 3.2: iPhone app for collecting data

Parse (`http://www.parse.com/`) is a cloud app platform for iOS, Android, JavaScript, Windows 8, Windows Phone 8, and OS X.

The Parse platform provides a complete backend solution for mobile applications, totally eliminating the need for writing server code or maintaining servers.

The Parse, in this work, removed the need of connecting the iPhone to a computer, in order to access the files, everytime a new set of data is ready.

A connection with Parse was developed, allowing the synchronization of the files with a single button push (Button send data).

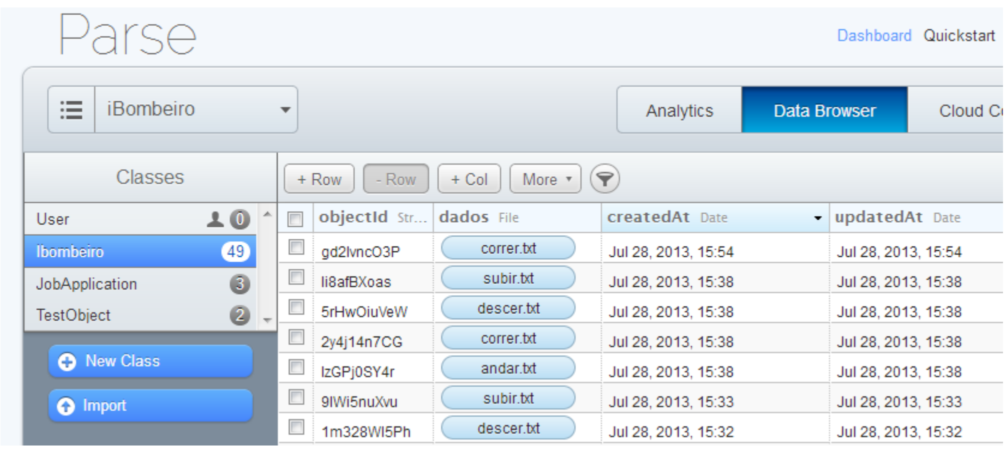


Figure 3.3: Parse cloud, where our files are stored

As we can see in Figure 3.3, all the files are available on the Parse cloud.

3.2 iBombeiro app

The iBombeiro app was then developed.
iBombeiro is continuously receiving data from device motion. The stream of data is then processed and classified. The resulted state is processed in three diferent ways that will be discussed in the next subsections.
The message for the CCO is constantly trying to be send to a near robot. When the firefighter is standing or lying for more that 30 seconds, the status becomes "danger" and an audible alarm is triggered.
If the state of the firefighter change, then a message is send to a WebService with that state (if network available), and that state is also recorded in the iPhone database (to act like a black box). To summarize:

- The iPhone will send a message to a near robot, via socket, in order to publish a topic in a ROS topic, with the state of the firefighter.
- If the state of the firefighter change the new state will be sent to a WebService, previously defined.
- If the state of the firefighter change the new state will be recorded in the iPhone database (black box approach)

In order to have a powefull app, without the system become unresponsive, it's necessary to work with multithreading. We used a core multithreading API available on iOS, Grand Central Dispatch.
Introduced in iOS 4.0, Grand Central Dispatch (GCD) is a BSD-level technology that is used to manage the execution of tasks in the application. GCD combines an asynchronous programming model with a highly optimized core to provide a convenient (and more efficient) alternative to threading.
GCD also provides convenient alternatives for many types of low-level tasks, such as reading and writing file descriptors, implementing timers, and monitoring signals and process events.
This means that if there is a failure in some part of the code, either in the database recording process, either in the webservice process or in the interaction with the MANET, the system will be capable of continue, without crash.

3.2.1 Classifier implementation in IOS

After the implementation of the classifier in MATLAB the system was portated to the iPhone. The training set obtained in MATLAB was used, so the test phase was no needed to be implemented in iPhone.

To implement the FFT in the iPhone a recent API from apple was used. The vDSP API provides mathematical functions for applications such as speech, sound, audio, and video processing, diagnostic medical imaging, radar signal processing, seismic analysis, and scientific data processing.
The vDSP functions operate on real and complex data types. The functions include data type conversions, FFTs, and vector-to-vector and vector-to-scalar operations.[1]
A sliding window technique (window size of 32 values) is applied to the buffer of acceleration values to classify with an eight first-in first-out policy.
To each sliding window an independent FFT is aplied to the values of acceleration in x, y and z. We do not perform any data-preprocessing because, as mencioned above, Core Motion uses unique algorithms to process the raw data it collects, so that it can present more refined information. The absence of the preprocessing step allows better performance of the app.
This code allows the inicialization of a class, fft, where we create the methods to compute the fast fourier transform.

```
FFT *fft;
fft=[[FFT alloc]init];
```

Then we can call the method FFT, that exists in the fft class to return the first fourier coeficient

```
// return first fourier coeficient for acceleration data in x
fourier1_x=*[fft FFT:accel_x];
```

After the calculation of the fourier coeficient the methods that will handle the probabilities, are called, in order to compute probabilities from likelihood distributions.
In order to avoid the deadlock in bayesian model, due to 0 or 1 probabilities, we also developed a method $preventFlatRegions$, to prevent that a probability is greater than 0.99% and is less than 0.005%.

[1] https://developer.apple.com

3.2.2 Comunication protocol

In its most basic form, a computer network is three or more computers connected via a communications system for the purpose of sharing data and/or resources, such as a printer. Although the communications systems used to build a computer network can vary, by far the most common types are based on the Open Systems Interconnection Basic Reference Model or OSI Model for short.

OSI Model is a standard description or reference model for how messages should be transmitted between any two points in a telecommunication network. Its purpose is to guide product implementors so that their products will consistently work with other products.

The main idea in OSI is that the process of communication between two end points in a telecommunication network can be divided into layers, with each layer adding its own set of special, related functions.
OSI divides telecommunication into seven layers. The layers are in two groups. The upper four layers are used whenever a message passes from or to a user. The lower three layers (up to the network layer) are used when any message passes through the host computer. Messages intended for this computer pass to the upper layers. Messages destined for some other host are not passed up to the upper layers but are forwarded to another host (Figure 3.4).
The OSI seven-layer model underlies every popular networking protocol.

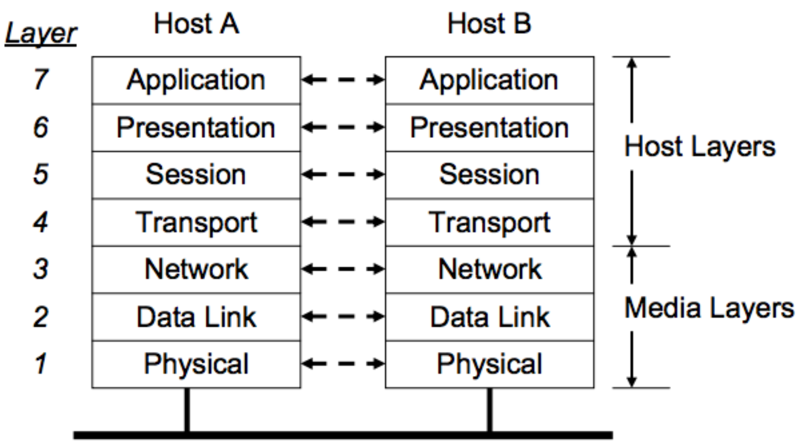


Figure 3.4: The Seven-Layer OSI Model

A MANET is an autonomous collection of mobile users that communicate over relatively bandwidth constrained wireless links. Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. The network is decentralized, where all network activity including discovering the topology and delivering messages must be executed by the nodes themselves, i.e., routing functionality will be incorporated into mobile nodes. Significant examples include establishing survivable, efficient, dynamic communication for emergency/rescue operations, disaster relief efforts, and military networks. Such network scenarios cannot rely on centralized and organized connectivity, and can be conceived as applications of MANETs.

The protocol stack of MANET consists of five layers: physical layer, data link layer, network layer, transport layer and application layer. As can be seen, the OSI model's session, presentation and application layers are merged into one section, the application layer in MANET. [Hartpence, 2011, Peacock, 2007]

In the scope of the Chopin Project a MANET was developed, so all the robot agents and all the human agents must behave like a ROS node inside the MANET.

ROS (`http://wiki.ros.org/`) is a framework for robot software development, providing libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. ROS is licensed under an open source, BSD license.
In the Chopin Project, a ROS node will be deployed for each agent, i.e. for each mobile robot and each hand-held device. Each node communicates with other nodes through Wi-Fi.

This assumption leads us to a problem: How can the iPhone be a ROS node inside the MANET?

3.2.2.1 Sockets

A way to simulate the behavior of the iPhone as a ROS node (and since this issue was not envisaged in the initial requirements) was to dispatch messages over socket to a near robot, with a predefined format, in order to communicate with the CCO. A socket is one end-point of a two-way communication link between two programs running on the network. Socket classes are used to represent the connection between a client program and a server program.
For both the Network and Transport Layers, special attention is paid to connectionless- and connection-based communication. One of the major differences between these two forms of transmission is controlling the flow of information between endpoints. It is interesting that the two Layer-4 protocols used today—TCP and UDP (User Datagram Protocol)—are

differentiated from each other in the exact same way, with TCP characterized as connection-oriented while UDP is connectionless (no session is established between hosts). TCP guarantees delivery through the use of acknowledgments and sequenced delivery of data. UDP does not guarantee or acknowledge delivery, or sequence data. UDP is fast, has low overhead requirements, and can support point-to-point and point-to-multipoint communication, while TCP is slower, has higher overhead requirements, and only supports point-to-point communication.

UDP sockets were chosen due to project constrains.

In the MANET each actor will behave like a ROS node, capable of communicating through a ROS topic, to be disseminated through all network members. The RosBridge (`http://wiki.ros.org/rosbridge_suite`) is a ROS package that emulates the messages that arrive via webSocket to the ROS network, as a ROS topic. The Figure 3.5 shows the communication structure of the different actors.

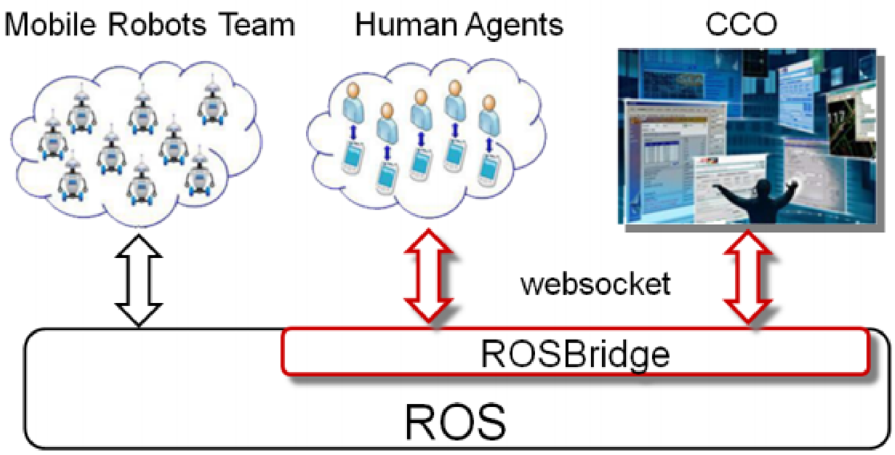


Figure 3.5: Communication structure of the different actors.

In the iPhone, when we want to send messages, the message in published in ROS topic *(/bombeiro1/datapub)*.

To receive data from CCO, the iPhone subscribe a ROS topic (*bombeiro1/datasub*).

In iOS, we used an asynchronous socket networking library, CocoaAsyncSocket (`https://github.com/robbiehanson/CocoaAsyncSocket`). CocoaAsyncSocket supports TCP and UDP. The AsyncSocket class is for TCP, and the AsyncUdpSocket class is for UDP. AsyncUdpSocket is a UDP/IP socket networking library that wraps CFSocket. It works almost exactly like the TCP version, but is designed specifically for UDP. This includes queued non-blocking send/receive operations, full delegate support, run-loop based, self-contained class, and support for IPv4 and IPv6.

As an example we show how to call the function that will send the UDP socket. The message is a string that the robot will publish in the topic *bombeiro1*.

```
    //call the function ROSPublish, and pass the firefighter status (statusString)
    [self ROSPublish:"/bombeiro1" withContent:statusString withType:"std_msgs/String"];

    //function ROSPublish
    - (void)ROSPublish:(char*)topic withContent:(char*)content withType:(char*)msg_type
{
    char end[] = "\xff";
    NSString *str = [NSString stringWithFormat:@"{\"receiver
    \":\"%s\", \"msg\": {\"data\": \"%s\"}, \"type\":\"%s\"}",topic, content, msg_type];
    NSData *strdata = [str dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:YES];
    NSMutableData *data = [NSMutableData alloc];
    [data appendData:strdata];
    [data appendBytes:end length:sizeof(end)];
    [sendSocket writeData:data withTimeout:-1 tag:2];
}
```

After defining the message, with the appropriate content, we call the function *sendSocket* that will handle with the dispatch of the UDP socket.

3.2.2.2 Webservices

In addition to the communication protocol solution developed above, within the scope of the CHOPIN Project, we also developed a webservice. Thus, it is possible to have the system to operate independently, as for the tests, as for future usage, within the project.

Web services (sometimes called application services) are services (usually including some combination of programming and data, but possibly including human resources as well) that are made available from a business's Web server for Web users or other Web-connected programs. Providers of Web services are generally known as application service providers. Users can access some Web services through a peer-to-peer arrangement rather than by going to a central server. Besides the standardization and wide availability to users and businesses of the Internet itself, Web services are also increasingly enabled by the use of the Extensible Markup Language (XML) as a means of standardizing data formats and exchanging data. XML is the foundation for the Web Services Description Language (WSDL).

In the scope of this work an webservice for the iOS app was developed (`http://www.ibombeiro.com/webservice.php`). The first step of the creation of the websevice was to create the database and the table (for this purpose only one table was needed, in the web-server (running MySQL and PHP).

The MySQL statement used to create the table was:

```
%criação da base de dados
DROP TABLE IF EXISTS ios;
CREATE TABLE ios (
    id int NOT NULL AUTO_INCREMENT PRIMARY KEY,
    status varchar(255) NOT NULL,
    timestamp varchar(20) NOT NULL
);
```

As the state of the firefighter changes, a message is send to a predefined url, via POST, i.e, the parameters are passed as part of the request body.

The status is then recorded in the mysql table.

```
if(isset($_POST['status'])){
    $status=$_POST['status'];
    $data=$_POST['data'];
    $query="insert into ios (status, data) values( '".$status."','".$data."')";
    mysql_query($query);
    mysql_close();
}
```

The div [2] that contains the status of the firefighter is refreshed second per second using asynchronous JavaScript and XML (AJAX).

Figure 3.6: Webservice

[2]div – generic flow container (`http://www.w3.org/TR/html-markup/div.html`)

In ios:

```
//function that send status to webservice
-(void) uploadDataToWebService:(NSString *) status
{
    NSString *post =[NSString stringWithFormat:@"status=%@",status];
    NSData *postData = [post dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:YES];
    NSString *postLength = [NSString stringWithFormat:@"%d", [postData length]];
    // Create the request
    NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:
        [NSURL URLWithString:postURL]];
    [request setHTTPMethod:@"POST"];
    [request setValue:postLength forHTTPHeaderField:@"Content-Length"];
    [request setValue:@"application/x-www-form-urlencoded" forHTTPHeaderField:@"Content-Type"];
    [request setHTTPBody:postData];
    NSHTTPURLResponse* urlResponse = nil;
    //process response
}
```

We created an instance of NSURLRequest, assigning it the URL of the webservice, and passing it to the webservice via POST

3.2.3 iPhone database: CoreData

Another upgrade to the system was to create a database on iPhone, in order to store the informations about the firefighter actions, for future reference. All the background is already developed enabling rapid integration with future services.

Introduced in iOS 3.0, the Core Data framework (CoreData.framework) is a technology for managing the data model of a Model-View-Controller application. Core Data is intended for use in applications in which the data model is already highly structured. Instead of defining data structures programmatically, the graphical tools in Xcode are used to build a schema representing the data model. At runtime, instances of the data-model entities are created, managed, and made available through the Core Data framework.

- managedObjectContext (of type NSManagedObjectContext) -This is the bridge between the programmer and the managed object model.
- managedObjectModel (of type NSManagedObjectModel) - This is the same concept as a schema in a database. This could represent the tables in a database or the different

types of managed objects that it is possible to create in the app database.

A Managed Object Model define the data we want to store.

  - Entity: This is an individual data object. In a database it'd be a table.
  - Atribute. This is an individual variable stored inside an entity. In a database it'd be a column.

- persistentStoreCoordinator (of type NSPersistentStoreCoordinator) - This is the bridge or the connection between the physical file that stores the data and the application. This bridge will be responsible for managing different object contexts. [Grönlund et al., 2012, Nahavandipoor, 2012]

A database model was created in iOS, to record the status of the firefighter.
Below is a code example to record data to the created database model (ibombeiro.sqlite). All The connections and configurations to the database are not shown here.

```
NSEntityDescription *entitydesc=[NSEntityDescription entityForName:@"IBombeiro"
    inManagedObjectContext:context];
NSManagedObject *newData=[[NSManagedObject alloc] initWithEntity:entitydesc
    insertIntoManagedObjectContext:context];
NSDateFormatter* dateFormatter = [[NSDateFormatter alloc] init];
[dateFormatter setDateFormat:@"yyyyMMdd HH:mm:ss"];
NSString *dateString = [dateFormatter stringFromDate:[NSDate date]];
[newData setValue:dateString forKey:@"timestamp"];
[newData setValue:newState forKey:@"coreStatus"];
NSError *error;
if (![context save:&error]) {
   //handle the error
}
```

In the code below first it's necessary to create a $NSEntityDescripton$, $*entitydesc$.

An $NSEntityDescription$ object describes an entity in Core Data. It means that we tell the managed object context what entity we want to retrieve the data from (in this case, IBombeiro entity, as shown in Figure 3.7).

Then the $NSManagedObject * newData$ is created. It's necessary associate the new managed object instance with the entity object that defines its properties and with the managed object context that defines its environment.
We can introduce new values to the table by typing $[newDatasetValue : newStateforKey :$

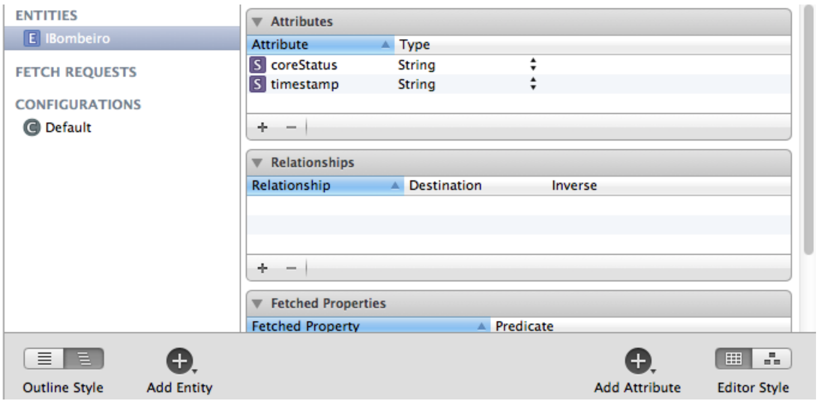


Figure 3.7: IBombeiro entity inside ibombeiro database

@"*coreStatus*"];.

newState, that is the status of the firefighter, in the current timestamp, is recorded in the IBombeiro entity, in the atribute *coreStatus*.

In order to not overcharge the iPhone (the data is processed in approximately a quarter of second) only will be saved new data when the status of the firefigter change. For the

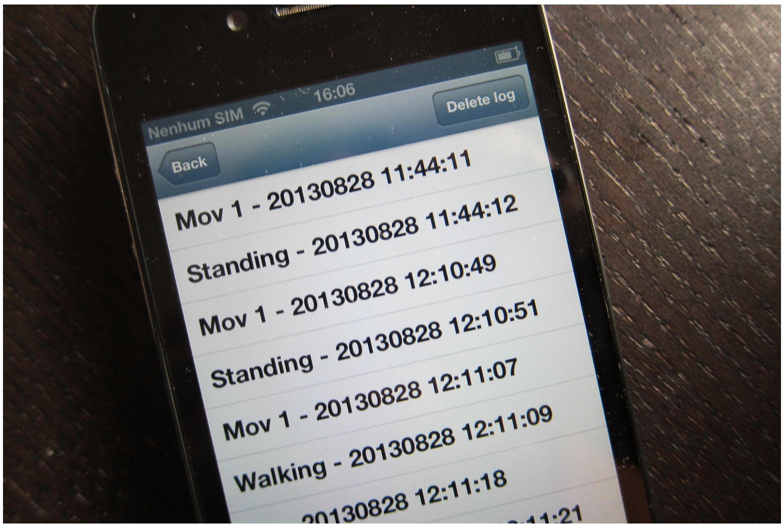


Figure 3.8: Tableview

visualization of the data, a model of a tableview was added to the app, in order to make requests to the database and show the requests in the form of a table. The *tableview* with the logs showing the diferent status of the firefighter can be acessed within the *iBombeiro* app (Figure 3.8).

There is also a button in the tableview "Delete log", that allows to delete all the records in the IBombeiro entity.

3.2.4 Experiments with an iPhone integrated in a manet

in `www.ibombeiro.com/multimedia` it is possible to see the interaction of the iPhone working as a node in the MANET (Figure 3.9).
The videos were recorded earlier so the status is only if the firefighter is standing or moving.
In the first video it is possible to see the iPhone sending a message to a ROS topic.

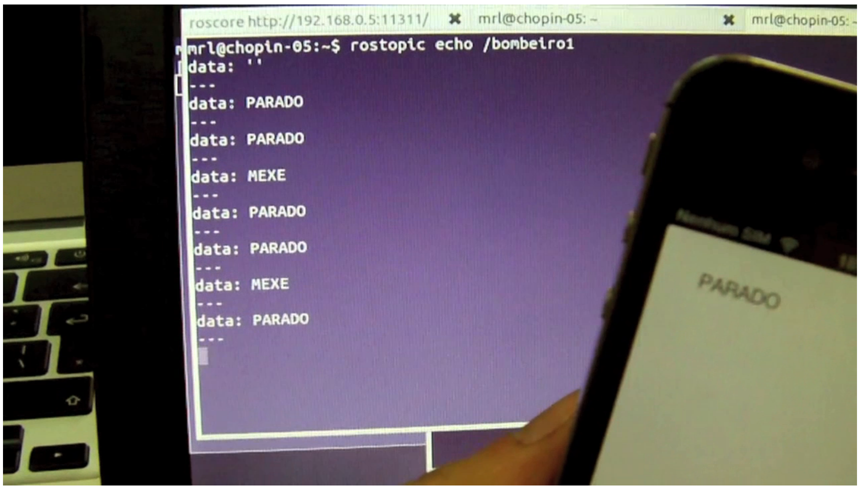


Figure 3.9: Interaction with the MANET - iPhone sending messages

The iPhone also receives mesages from CCO, via Sockets UDP. In the second video, we can see the iPhone receiving a message from the CCO (Figure 3.10).

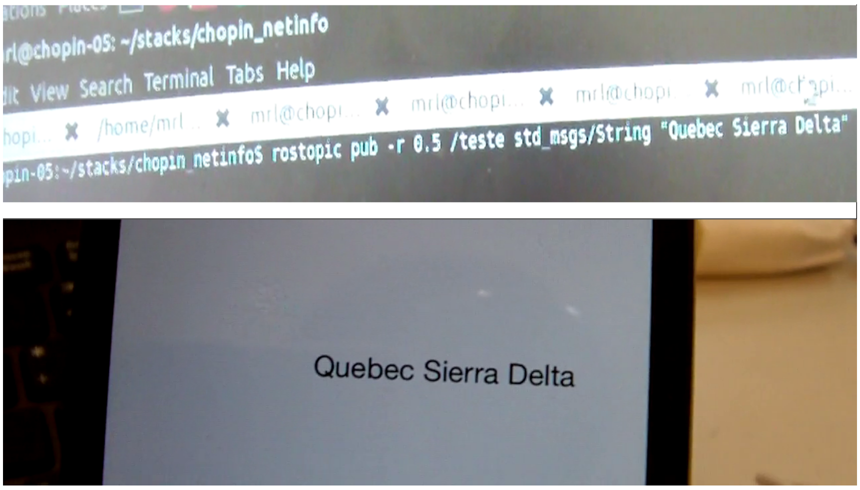


Figure 3.10: Interaction with the MANET - iPhone receiving messages

Chapter 4

Conclusions and future work

4.1 Conclusions and discussions

In this work we developed an action recognition model for implementation on a mobile system, for monitoring the state of firefighter, integrated in a mixed team of robotic and human agents. This work was developed in he scope of the CHOPIN Project.

In the system's implementation we used a classification model based on Bayesian techniques, in which inference is based on the *Maximum A Posteriori* (MAP) method.

An extended version of the model is also proposed. Time constrains alone have prevented its implementation, but given the complete formulation, this can be strongly suggested as future work, allowing a personalised version of the iBombeiro.

The model was successfully implemented in iPhone, and has generated promising results. Our approach to the problem had to be simplified, given to major restrictions: battery and CPU power autonomy. Although only the acceleration data and a reduced number of characteristics were used, due to computational limitations, the system has shown interessant results under the proposed conditions, and the models validated using methods of cross-validation. However, the device's orientation in the performance of the emergency gesture was found to have key impact during the classification.

Therefore, if device autonomy restrictions are relaxed, but not discared, and/or the models computationally optimised, we propose exploring additional features in order to improve the model robustness.

Another interesting solution is the potencial usage of an additional accelerometer, attached to the legs or hip in order to have relative accelerations and highly discriminant data, widening the range of actions to be identified.

We also developed a link between the firefighter and the MANET, where the firefighter behave like a ROS node, by sending a message to a near robot. This implementation poses

some questions, due to the project constrains. We use UPD sockets to communicate to CCO via MANET, by sending a predefined message to be published as a ROS topic. But UDP sockets are not reliable; there may not be any robots nearby; it is not possible to know whether the message reaches the destination; the robot near can be stuck in an "island". Messages from the CCO to the firefighter, that are received in the same manner (UDP socket), are also liable to not be received.
Another problem is due to the lack of tests concerning multiple robots and firefighters acting as a team, communicating to the CCO inside the MANET.
MANETs suffer from the same limitations as fixed wireless mesh networks, but also are vulnerable to additional challenges resulting from their inherent mobility. Problems will arise, that will need to be resolved.

As alternative to the MANET communications a webservice was developed allowing the system to work independently.

The possibility of the system behave like a "black box", saving all relevant information was also covered.

4.2 Future Work

As future work there are a wide range of improvements and new modules to be developed in the scope of iBombeiro, that can be added to the existing framework. The approach used in iBomeiro was a simplist one. Although the good results achieved in iBombeiro one interesting aspect would be the increase of the number of features. In this scenario rigorous tests should be conducted to conclude about the computational feasibility, for the benefits of the introduction of more features.

Another sugestion is the introduction of positioning with geolocation, taking advantages of of the iPhone features (Geolocation API). This can add another important service to iBombeiro, in areas where geolocation is avaliable.

Other important contribution for future work will be the integration of the developed Bayesian extended model, in order to take advantages of the personalisation by increasing system's accuracy. A corollary of this extended model would be the identification of persons by the way they perform their actions. The possibility of bulding a map with firefighters localization, without prior configuration, would have a great impact within CHOPIN Project.

Action recognition with wearable sensors has the potential to enhance existing applications as well as enable new ones, ranging from personal healthcare and assisted living, to industrial applications, and even entertainment and arts.

Beyond the scope of CHOPIN Project and of elderly care area, where already exists promising advances in the action recognition systems using smartphones, and as been mencioned in section 1.1 there are already smartphone applications capable of using GPS positioning to track user's activity. The integration of our work with a system like RunKeeper would allow the user to perform activities without redefining the activity in the smartphone. With an extra kit of sensors, attached to the smartphone, the accuracy of the system could be increased, and a new level of smartphone assisted personal trainer could be achieved.

Another interessant application area, would be in mountaineering (an area that was a great potencial and a personal interest).

With the excessive growth of adventure tourism enterprises, that lead unqualified persons to hazardous environments, the development of a system capable of performing action recognition to detect potentially dangerous actions (e.g. falling), integrated with GPS, temperature sensors, heartbeat sensors would be an interessant idea to explore.

Figure 4.1: A crowd of climbers slog up the Lhotse Face, heading toward Camp IV, last stop before the summit. Loose regulations and a boom in commercial guiding over the past two decades have made Everest far more accessible to experts and novices alike @Andy Bardon

Bibliography

[Apple, 2013] Apple (2013). Event handling guide for ios. Technical report.

[Bessire et al., 2008] Bessire, P., Laugier, C., and Siegwart, R. (2008). *Probabilistic Reasoning and Decision Making in Sensory-Motor Systems.* Springer Publishing Company, Incorporated, 1st edition.

[Castro and Antunes, 2005] Castro, C. F. and Antunes, J. B. (2005). *Combate a Incêndios Urbanos e Industriais.* , Escola Nacional de Bombeiros (Portuguese National School of Firefighters), Sintra, Portugal.

[Colas et al., 2010] Colas, F., Diard, J., and Bessière, P. (2010). Common bayesian models for common cognitive issues. *Acta Biotheoretica*, 58(2-3):191–216.

[e Schühli, 2005] e Schühli, E. S. (2005). Reconhecimento de gestos de maestro utilizando redes neuronais artificiais parcialmente recorrentes. Master's thesis, Universidade Federal do Paraná.

[Grönlund et al., 2012] Grönlund, H.-E., Francis, C., and Grimes, S. (2012). *iOS 6 Recipes: A Problem-Solution Approach.* Apress.

[Hartpence, 2011] Hartpence, B. (2011). *Packet Guide to Core Network Protocols.* O'Reilly Media, Inc.

[Hofmann et al., 1997] Hofmann, F. G., Heyer, P., and Hommel, G. (1997). Velocity profile based recognition of dynamic gestures with discrete hidden markov models. *Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, pages 81–95.

[Hristijan and Matjaz, 2011] Hristijan, G. and Matjaz, G. (2011). Accelerometer data preparation for activity recognition.

[Jordan, 1999] Jordan, M. I., editor (1999). *Learning in graphical models.* MIT Press, Cambridge, MA, USA.

[Just and Marcel, 2009] Just, A. and Marcel, S. (2009). A comparative study of two state-of-the-art sequence processing techniques for hand gesture recognition. *Computer Vision and Image Understanding*, 113:532–543.

[Kela et al., 2005] Kela, J., a, P. K. a., antyj arvi, J. M., Kallio, S., Savino, G., Jozzo, L., and Marca, S. D. (2005). Accelerometer-based gesture control for a design environment. *Personal and Ubiquitous Computing*, 10:285–299.

[Khoshhal et al., 2010] Khoshhal, K., Aliakbarpour, H., Quintas, J., Drews, P., and Dias, J. (2010). Probabilistic lma-based classification of human behaviour understanding using power spectrum technique. In *13th International Conference on Information Fusion*.

[Khoshhal and Dias, 2012] Khoshhal, K. and Dias, J. (2012). Probabilistic human interaction understanding - exploring relationship between human body motion and the environmental context. *Pattern Recognition Letters*, Special Issue on Scene Understandings and Behaviours Analysis:In press.

[Khoshhal et al., 2012] Khoshhal, K., Santos, L., Aliakbarpour, H., and Dias, J. (2012). Parameterizing interpersonal behaviour with laban movement analysis - a bayesian approach. In *3rd International Workshop on Socially Intelligent Surveillance and Monitoring (SISM2012) in CVPR2012*.

[Klingmann, 2009] Klingmann, M. (2009). Accelerometer-based gesture recognition with the iphone.

[Kwapisz et al., 2010] Kwapisz, J. R., Weiss, G. M., and Moore, S. A. (2010). Activity recognition using cell phone accelerometers.

[Li et al.,] Li, M., Rozgica, V., Thatte, G., Lee, S., Emken, A., Annavaram, M., Mitra, U., Spruijt-Metz, D., and Narayanan, S.

[Mihajlovic and Petkovic, 2001] Mihajlovic, V. and Petkovic, M. (2001). Dynamic bayesian networks: A state of the art.

[Nahavandipoor, 2012] Nahavandipoor, V. (2012). *iOS 6 Programming Cookbook*. O'Reilly Media, Inc.

[Niezen and Hancke, 2009] Niezen, G. and Hancke, G. P. (2009). Evaluating and optimising accelerometer-based gesture recognition techniques for mobile devices. *IEEE AFRICON 2009*.

[Pärkkä et al., 2006] Pärkkä, J., Ermes, M., a, P. K. a., antyj arvi, J. M., Peltola, J., and Korhonen, I. (2006). Activity classification using realístic data from wearable sensors. In *Cornell University*. Press.

[Peacock, 2007] Peacock, M. B. A. (2007). Connecting the edge: Mobile ad-hoc networks (manets) for network centric warfare.

[Prekopcsák, 2008] Prekopcsák, Z. (2008). Accelerometer based real-time gesture recognition. pages 1–5.

[Rabiner, 1989] Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *IEEE*.

[Schlömer et al., 2008] Schlömer, T., Poppinga, B., Henze, N., and Boll, S. (2008). Gesture recognition with a wii controller. *Proceedings of the Second International Conference on Tangible and Embedded Interaction (TEI'08), Bonn, Germany*.

[Sharma et al., 2011] Sharma, A., Lee, Y.-D., and Chung, W.-Y. (2011). High accuracy human activity monitoring using neural network. *CoRR*, abs/1107.4429.

[Sun et al., 2010] Sun, L., Zhang, D., Li, B., Guo, B., and Li, S. (2010). Activity recognition on an accelerometer embedded mobile phone with varying positions and orientations.

[Sundaram and Mayol-Cuevas, 2009] Sundaram, S. and Mayol-Cuevas, W. (2009). High level activity recognition using low resolution wearable vision. In *First Workshop on Egocentric Vision, in conjunction with the International Conference on Computer Vision and Pattern Recognition*. IEEE.

[Yang and Ahuja, 1999] Yang, M.-H. and Ahuja, N. (1999). Recognizing hand gesture using motion trajectories.