Pervasive Computing and Communications Design and Deployment:

Technologies, Trends and Applications

Apostolos Malatras
University of Fribourg, Switzerland

Information Science
REFERENCE

Senior Editorial Director: Kristin Klinger
Director of Book Publications: Julia Mosemann
Editorial Director: Lindsay Johnston
Acquisitions Editor: Erika Carter
Development Editor: Hannah Abelbeck
Production Editor: Sean Woznicki
Typesetters: Jennifer Romanchak and Mike Brehm
Print Coordinator: Jamie Snavely
Cover Design: Nick Newcomer

Published in the United States of America by
Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com/reference

Library of Congress Cataloging-in-Publication Data
Pervasive computing and communications design and deployment: technologies, trends, and applications / Apostolos Malatras, editor.
p. cm.
Includes bibliographical references and index.
ISBN 978-1-60960-611-4 (hardcover) -- ISBN 978-1-60960-612-1 (ebook) 1. Ubiquitous computing. I. Malatras, Apostolos, 1979-
QA76.5915.P455 2011
004--dc22
2010040624

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Chapter 2
Context-Aware Smartphone Services

Igor Bisio
University of Genoa, Italy

Fabio Lavagetto
University of Genoa, Italy

Mario Marchese
University of Genoa, Italy

ABSTRACT

Combining the functions of mobile phones and PDAs, smartphones can be considered versatile devices and offer a wide range of possible uses. The technological evolution of smartphones, combined with their increasing diffusion, gives mobile network providers the opportunity to come up with more advanced and innovative services. Among these are the context-aware ones, highly customizable services tailored to the user's preferences and needs and relying on the real-time knowledge of the smartphone's surroundings, without requiring complex configuration on the user's part. Examples of context-aware services are profile changes as a result of context changes, proximity-based advertising or media content tagging, etc.

The contribution of this chapter is to propose a survey of several methods to extract context information, by employing a smartphone, based on Digital Signal Processing and Pattern Recognition approaches, aimed at answering to the following questions about the user's surroundings: what, who, where, when, why and how. It represents a fundamental part of the overall process needed to provide a complete context-aware service.

INTRODUCTION

Context-aware smartphone applications should answer the following questions about the device's surroundings (Dey, 2000): What, Who, Where, When, Why and How. As a consequence, in order to provide context-aware services, a description of the smartphone's environment must be obtained by acquiring and combining context data from different sources, both external (e.g., cell IDs, GPS

DOI: 10.4018/978-1-60960-611-4.ch002

coordinates, nearby Wi-Fi and Bluetooth devices) and internal (e.g., idle/active status, battery power, accelerometer measurements).

Several applications explicitly developed for smartphones will be surveyed in this chapter. In more detail, an overview of the context sources and sensors available to smartphones and the possible information they can provide is proposed in Section "Smartphones: an Overview".

The general logical model of a context-aware service composed of *i)* Context Data Acquisition, *ii)* Context Analysis and *iii)* Service Integration is introduced in Section "Context-Aware Services". A set of possible context-aware services such as Audio Environment Recognition, Speaker Count, Indoor and Outdoor Positioning, User Activity Recognition is listed in the following Sections. Further details concerning the aforementioned Context Analysis phase for specific context-aware services, which have been designed and implemented for smartphone terminals, are introduced as well.

In the specific case of this chapter, all context-aware services are based on sophisticated Digital Signal Processing approaches that have been specially designed and implemented for smartphones. The presented methods have been designed based on the principles set out by the corresponding related literature in the field, while additionally all the described solutions concern specific proposals and implementations performed by the authors.

Specifically, Audio Signal Processing-based services are introduced in Section "Audio Signal Processing based Context-Aware Services". In more detail, Environment Recognition (Perttunen, 2009) and Speaker Count (Iyer, 2006) services are described. Concerning Environment Recognition, both the architecture and the signal processing approach designed and implemented to identify the audio surrounding of the terminal (by distinguishing among street, overcrowded rooms, quiet environment, etc.) will be presented. Speaker Count services will be introduced as well. In detail, determining the number of speakers participating in a conversation is an important area of research as it has applications in telephone monitoring systems, meeting transcriptions etc. In this case the service is based on the audio signal recorded by the smartphone device. The speech processing methodologies and the algorithms employed to perform the Speaker Count process will also be introduced.

An overview of services based on the processing of signals received by smartphones' network interfaces such as GPS receiver, Wi-Fi, Bluetooth, etc. is proposed in Section "Network Interface Signal Processing based Context-Aware Services: Positioning". In particular, Indoor Positioning methods (Wang et al., 2003) have been taken into account. In this case the information required to carry out the positioning process is obtained from multiple sources such as the Wi-Fi interface (in the case of Indoor Positioning) and the GPS receiver (in the case of Outdoor Positioning). In particular, the methods suitable for smartphone implementation will be illustrated with particular emphasis on the Indoor Positioning approaches that have been implemented and tested directly on smartphones.

Finally, possible User Activity Recognition services (Ryder, 2009) that can be provided, starting from raw data acquired directly from the measurements carried out by the smartphone's accelerometer, are introduced in Section "Accelerometer Signal Processing based Context-Aware Services". In this case, additional technical details on the methods for the classification of activities such as walking, running, etc. will be described.

In all Sections where specific context-aware services will be introduced, the design and the implementation aspects of each service will also be detailed, based on the practical expertise, the employed test-beds and the results obtained during specific experimental campaigns that we have conducted. The chapter moreover will focus on the computational load and the energy consumption that is required to provide specific context-aware services in order to take into account the limited

computational capacity and energy autonomy of smartphone platforms.

CONTEXT-AWARE SERVICES FOR SMARTPHONES

Smartphones: An Overview

For the next years, several market experts predict a significant growth in the market for converged mobile devices that simultaneously provide voice phone function, multimedia access, PDA capabilities and game applications. These devices will allow expanding the current market by adding new types of consumers. In fact, they will employ these devices for activities very different with respect to classical mobile phone calls usage.

This new trend will drive both Original Equipment Manufacturers (OEMs) and Carriers to meet this growth by providing smart devices and new services for the new class of users. In more detail, in 2003 it was estimated that converged mobile devices, also termed smartphones, made up three percent of worldwide mobile phone sales volume. Nowadays, the smartphone market is continuing to expand at triple digit year-over-year growth rates, due to the evolution of voice-centric converged mobile devices, mobile phones with application processors and advanced operating systems supporting a new range of data functions, including application download and execution.

In practice, smartphones will play a crucial role to support the users' activities both from the professional and private viewpoint. Context-aware services, object of this work, are a significant example of new features available to users. For this reason, before the survey of possible context-aware services for smartphones, a brief introduction concerning the smartphone platform in terms of hardware and software architecture is provided starting from (Freescale Semiconductor Inc., 2008), as well as a survey of the available smartphone Operating Systems.

Smartphone Architecture

From the hardware viewpoint, the first generation of analog cell phones was composed of devices consisted of a discrete single Complex Instruction Set Computing (CISC)-based microcontroller (MCU) core. Its task concerned the control of several analog circuits. The migration from analog to digital technology created the necessity of a Digital Signal Processor (DSP) core.

In fact, more advanced architectures included both cores, thus creating a dual-core system consisting of an MCU and a DSP, which were integrated in a single Application Specific Integrated Circuit (ASIC). Actually, the aforementioned dual-core architectures did not support the feature requirements of converged devices because they were designed only to support communications tasks. As a result, today's smartphone architecture requires additional processing resources. Currently, a discrete application processor is included in the architecture together with the discrete dual-core cellular baseband integrated circuit. Each processor requires its own memory system including RAM and ROM, which complete the computation architecture of a smartphone. Together with the above described architecture, recent mobile devices include wireless networking interfaces, such as Wi-Fi and Bluetooth technology. Each added communication interface, in several cases also very useful to provide context-aware services, requires additional modules for each function, including radio transceivers, digital basebands, RAM and ROM components within the module. In practice, modern smartphones mount a minimum of three, or as many as six, processors, each with its own dedicated memory system, peripherals, clock generator, reset circuit, interrupt controller, debug support and inter-processor communications software. Obviously, the overall architecture requires a power supply system.

A logical scheme of the described smartphone architecture is reported in Figure 1.

Concerning the software, the Cellular Baseband block shown in Figure 1, typically divides its tasks in the same way. In particular, the DSP performs signaling tasks and serves as a slave processor to the MCU, which runs the upper layer (L2/L3) functions of the communication protocol stack. On one hand, the Layer 1 signal processing tasks, handled by the DSP, include equalization, demodulation, channel coding/decoding, voice codec, echo and noise cancellation. On the other hand, the MCU manages the radio hardware and moreover implements the upper layer functions of the network protocol stack, such as subscriber identity, user interface, battery management and the nonvolatile memory for storage for the phone book. The Application Processor block, equipped with an MCU, manages the user interface and all the applications running on a smartphone.

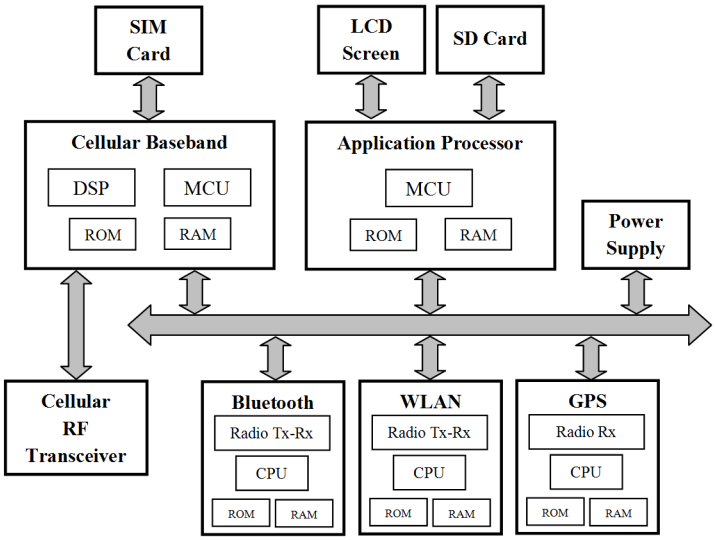
In this Hardware/Software architecture, it is worth noticing that the communication protocol tasks and multimedia workloads may lead to performance conflicts, since they share the smartphone resource. This problem, only mentioned here as it is out of the scope of the chapter, requires a sophisticated internetworking approach and, in particular, advanced inter-processor communications aimed at increasing processing availability and at reducing overheads and power consumption, which result in reduced battery life and usage time for the end user.

A possible low-cost solution to such a problem may be to merge the Application Processor and the Cellular Baseband blocks into a single ASIC consisting of two or three cores. This approach eliminates the performance conflict between the communication protocol and multimedia tasks, although the complexity of the inter-processor communication is not reduced significantly.

Operating Systems Survey

Nowadays, the software functions previously described are obtained by using dedicated Operating Systems (OSs) designed and implemented for smartphone platforms. For the sake of completeness, a list of the currently available OSs is reported below. There are many operating systems designed for smartphones, the main ones being Symbian, Palm OS, Windows Mobile (Microsoft), Android, iPhone OS (Apple) and Blackberry OS (RIM).

Figure 1. Logical scheme of the modern smartphones' hardware architecture



The most common one is Symbian, but its popularity is declining due to the recent diffusion of other OSs such as Android, iPhone and Blackberry. In fact, in recent years iPhone and BlackBerry phones have had a remarkable success, while the popularity of Android (the open source OS developed by Google) is constantly on the rise, after a very successful 2009. In more detail, *Symbian OS* is an open operating system adopted as standard by major global firms producing mobile devices (cell phones, smartphones, PDAs) and is designed to support the specific requirements of data transport generation mobile devices (2G, 2.5G and 3G). In 2005 the first version of the 9.x series (version 9.1) was released and, in particular, in 2006 version 9.3 included multimedia processing features, which are of great interest to the current applications of the smartphone platforms. *Palm OS* was developed in 1996 and released by the American Palm OS PalmSource Inc., later acquired by Japan's Access Systems which immediately reformulated the project with the aim to embrace the power and versatility of the Linux operating system. Despite having been announced for 2007, the new Palm OS at the moment is not yet available. *Windows Mobile* is the Microsoft operating system, including a suite of basic applications, dedicated to mobile devices. In this particular case, the user interface of the OS is very similar to the latest versions of the operating system for desktop and notebook PCs. In 2007 Google planned to develop a smartphone to compete directly with the Apple iPhone. Actually, in the early 2008 Google's management claimed not to be interested in the implementation of hardware, but in the development of software. In fact, Google launched a new OS called *Android*, an open source software platform for mobile devices, based on the Linux operating system. As detailed in the following Section, Android together with Symbian have been employed in our experiments to implement the described context-aware services. *iPhone OS* is the operating system developed by Apple for iPhone, iPod and iPod Touch and it is, in practice, a reduced version of Mac OS X. *BlackBerry OS* is the operating system developed by Research In Motion (RIM) and it is specifically designed for its own devices (BlackBerry).

Context-Aware Services

In this Section, a brief overview of the concept of context-aware services is provided. The overall framework presented here is based on the work reported in (Marengo et al, 2007) and in references therein.

In more detail, the real-time knowledge of a user's context is able to offer a wide range of highly personalized services. It makes services really customized because they are based on the environments, the behavior and other context factors that users themselves are experiencing when the services are provided. In practice, all the information a user may receive, for example by using a smartphone, which is also the source of context data in the case of this chapter, is based on position and geographic area in which users are, on the activities that are taking place and on the users' preferences. In practice, Context-Aware services provide useful information to users starting from the answers to the questions about the device's surroundings: What, Who, Where, When, Why and How.

The contribution of this chapter is to propose a survey of several methods to extract context information, by employing a smartphone, based on Digital Signal Processing and Pattern Recognition approaches, aimed at answering to the aforementioned questions. It represents a fundamental part of the overall process needed to provide a complete context-aware service as briefly detailed in the following.

From a more technical perspective, a system that realizes a complete context-aware service can be divided into three, successive and complementary, logic phases (or stages) listed below taking smartphone terminals as reference: *i)* Context Data Acquisition; *ii)* Context Analysis; *iii)* Ser-

vice Integration. A scheme of the overall process to provide such service in reported in Figure 2, which is a slightly revised version of the scheme proposed in (Marengo, 2007).

Stage 1: Context Data Acquisition

During this stage, context information is captured and aggregated starting from signals generated by various information sources available (typically sensors or network interfaces).

These sources can provide information concerning the employed network accesses (e.g., using the GSM network rather than UMTS or the Wi-Fi interface), concerning the terminal (e.g., battery level, idle terminal) or information obtained by signals acquired by sensors on the device (e.g., microphone, accelerometer).

At this stage, considering the very limited resources available in a smartphone, in particular from the computational and energetic viewpoint, it is important to realize Context data acquisition approaches able to collect data quickly and to integrate heterogeneous information sources.

Stage 2: Context Analysis

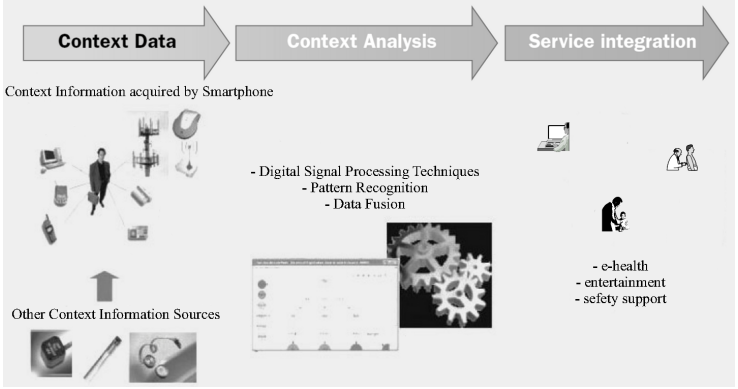
It is the stage where information previously acquired provided by smartphones' sources is processed. This level is very important because it is responsible for the process that starts from "raw" data and ends with the decision of the context in which the device is. In particular, the main functions of this level are the identification of the context of a user and the generation of complete context information, starting from lower level information, i.e. raw data, which is directly captured by the smartphone's sensors.

In this case signal processing and pattern recognition methods play a crucial role. In fact, at this stage smartphones must process the data supplied from the lower level of context and apply the algorithms needed to extract information from such data and provide higher level information.

Stage 3: Service Integration

This is the final stage where the context-aware information is exploited to provide the output of the overall context-aware service. In practice, in this phase, the services are provided to the users. For example, an e-health context-aware service may concern the tele-monitoring of long-suffering users through a smartphone terminal: signals (lower level information) from the microphone, from the accelerometer and from the GPS receiver are captured (Stage *i*); information (higher layer information) about the environment, the movement and the position of the long-suffering users is provided (Stage *ii*); possible feedbacks to the

Figure 2. Stages of a context-aware service realized with smartphones adapted from (Marengo, 2007)



long-suffering users and/or to the medical/emergency personnel of a clinic are provided directly in their own smartphones (Stage *iii*).

AUDIO SIGNAL PROCESSING BASED CONTEXT-AWARE SERVICES

In the next few years, mobile network providers and users will have the new opportunity to come up with more advanced and innovative context-aware services based on the real-time knowledge of the user's surroundings. In fact, context data may also be acquired from the smartphone's audio environment. In general, the classification of an audio environment, the correspondence between two or more audio contexts or the number and the gender of active speakers near the smartphone, together with other possible context features, can be useful information employed to obtain helpful contents and services directly by the mobile users' devices.

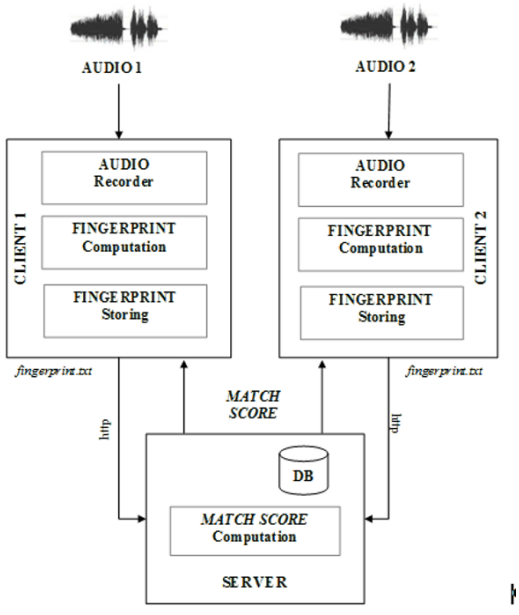
Environment Recognition

In this sub-section the implementation and the related performance analysis of an Audio Fingerprinting platform is described. The Audio Fingerprint is a synthesis of the audio signals' characteristics extracted from their power density functions in the frequency domain. The platform and its capabilities are suited to be a part of a context-aware application in which acoustic environments correspondence and/or classification is needed.

In more detail, starting from the state of the art concerning Digital Signal Processing (DSP) and Acoustic Environment Classification (AEC), we implemented a system able to analyze the correspondence among audio signals. The proposed platform, and in particular the software procedure implemented in it, produced encouraging experimental results, in terms of correct correspondence among audio environments, and the computations needed to provide audio signal correspondence, introduced below, are performed in a reasonable amount of time.

The implemented platform is a hardware/software system able to evaluate the correspondence between two or more audio signals. It is composed of N terminals, called Clients, directly connected to a centralized Server. In the proposed implementation, shown in Figure 3, $N = 2$ terminals have been used. In general, the network employed to connect the platform elements may be a Local Area Network (LAN) or other possible network typologies (e.g., Wireless Local Area Network (WLAN) as in the case of our work) may be employed without loss of functionality. All network nodes are synchronized by using the well-known Network Time Protocol (NTP) and a specific reference timing server. Synchronization is a necessary requirement. It allows performing a reliable evaluation of the audio signals correspondence. In fact, when two equal audio signals

Figure 3. Audio fingerprinting platform architecture



are unsynchronized the system might not detect their correspondence.

In more detail, the aim of the Clients is to record audio signals from the environments where they are placed, to extract the audio fingerprints, as described below, and write it in a textual file. The Clients subsequently send the file containing the fingerprints to the Server by employing the Hyper Text Transfer Protocol (HTTP). The Server node receives the fingerprints transmitted through the network and evaluates their possible correspondence as detailed in the following. To allow the fingerprint transmissions via HTTP, an Apache server has been installed on each node of the network.

The proposed architecture is suited to be used to fulfill two typical context-awareness actions: the environment classification and the environment correspondence. The former application is aimed at recognizing an environment starting from one ore more fingerprints of the recorded audio and comparing them with previously loaded fingerprints, representative of a given environment, stored in a specific Data Base (DB) in the Server node of the platform. The latter action is aimed at establishing if two or more audio fingerprints coincide.

The implemented platform, described in this sub-section, has been configured to fulfill the second action (the audio fingerprints correspondence) and in the specific case of the platform implemented by our research group, the Clients are smartphones. The considered environments, in the case of this implemented architecture, are five: quiet environment (silence); an environment where there is only one speaker; an environment where there is music; noisy environment; a noisy environment where there are also several speakers.

In the architecture shown in Figure 3, the Client part is composed of three fundamental components whose functions are:

- Audio Recording;
- Fingerprint Computation;
- Fingerprint Storing.

Furthermore, the Server component plays a crucial role. It is a typical web server where dynamic PHP pages have been implemented to exchange fingerprints among different client devices and to compute the correspondence between fingerprints. In more detail, the Server has two specific aims:

- Fingerprint Storage (in this case the stored fingerprints have been received from different Clients);
- Fingerprint Correspondence Analysis (it compares different fingerprints and establishes if they are equal as detailed in the following).

In practice, the fingerprints computed by the Clients are sent to the Server, which saves them in its local database (DB) and, finally, a specific function implemented in the Server computes the correspondence between a given fingerprint of the database (or, alternatively, the last received one) and the stored ones. It allows finding the possible correspondence among audio fingerprints.

The employed procedures for the computation of audio fingerprints and of correspondence among fingerprints are described in the following.

Audio Fingerprint Computation

The considered audio fingerprint is a matrix of values representative of the energy of the recorded audio signal computed by considering specific portions of the entire frequency bandwidth of the signal. The method applied in this paper is based on the techniques reported in (Lee, 2006). It represents a revised *Philips Robust Hash* (PRH) approach (Doets, 2006), which has been chosen in this implementation because it is suited to be applied to smartphone platforms due to its limited computational load. The basic idea is to exploit the human hearing system, which can be modeled (Peng, 2001) as a battery of 25 sub-bands

contained in the frequency interval between 0 and 20 KHz. In more technical detail, the recorded audio signal is divided into 37 ms frames and *Hann* windowing is applied to each frame. Consecutive frames are overlapped by 31/32 in order to capture local spectral characteristics. This approach follows exactly the proposal reported in (Lee, 2006; Haitsma, 2006) where, through experimental campaigns, the frame length and the overlapping fraction have been found.

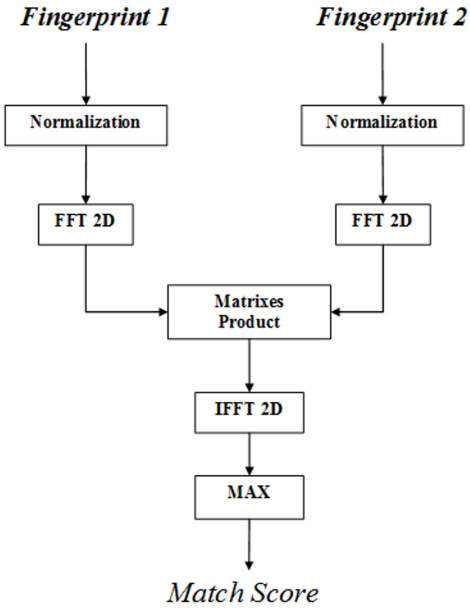
The energy of each of the above-mentioned 25 sub-bandwidths is computed for each frame in the frequency domain, by employing its Fourier Transform. The energy of a sub-bands is subtracted to the value of the previous one. The obtained results are stored in a vector of 24 components. This procedure is iterated for each frame and the final result is a matrix with 24 rows (the size of each single energy vector) and a number of columns equal to the number of frames. The final computation needed to extract the audio fingerprint requires the comparison, among columns, of the previously described matrix. In practice, for each row, each element is substituted by 1 or 0, respectively, based on whether it is greater or lesser than the following element.

Correspondence Computation

Concerning the evaluation of the correspondence among extracted fingerprints, in more detail, the considered and implemented procedure is based on the method proposed in (Lee, 2006). In practice, as reported in Figure 4, the method computes a *Match Score*, which is the measure of the correspondence between two audio fingerprints.

The computation is based on the fingerprints of short duration audio fragments. As a consequence, the employed method is implicitly granted a low computational complexity. The previously-defined fingerprint comparison is performed in the frequency domain and not in the time domain. In more technical terms, the Match Score is the maximum value of the two-dimensional cross-correlation between the fingerprint matrixes of the audio fragments. It ranges between 0, which is the value of the Match Score in case of different fingerprints, and 1, which is the value of the *Match Score* in case of comparison among two identical audio fingerprints. To reduce the number of operations needed to compute the two-dimensional cross-correlation, such procedure is carried out by computing the Discrete Fourier Transform (DFT) of the two fingerprint matrixes and by calculating their product, as schematically depicted in Figure 4. This approach has the obvious advantage of significantly reducing the number of operations required to obtain exactly the same results as in the time domain.

Figure 4. Correspondence computation functional block



Speaker Count and Gender Recognition

Speaker count is applicable to numerous speech processing problems (e.g., co-channel interference reduction, speaker identification, speech recognition) but it does not yield a simple solution. Several

speaker count algorithms have been proposed, both for closed- and open-set applications. Closed-set implies the classification of data belonging to speakers whose identity is known, while in the open-set scenario there is no available *a priori* knowledge on the speakers.

Audio-based gender recognition also has many possible applications (e.g., for selecting gender-dependent models to aid automatic speech recognition and in content-based multimedia indexing systems) and numerous methods have been designed involving a wide variety of context features.

Although for both problems many methods have produced promising results, available algorithms are not specifically designed for mobile device implementation and thus their computational requirements do not take into account smartphone processing power and context-aware application time requirements.

In this Section, on the basis of our previous practical experience, a speaker count method based on pitch estimation and Gaussian Mixture Model (GMM) classification, proposed in (Bisio, 2010) is described. It has been designed to recognize single-speaker (1S) samples from two-speaker (2S) samples and to operate in an open-set scenario. The proposed method produced encouraging experimental results and sample recognition is obtained in a reasonable amount of time. In addition, a method for single-speaker gender recognition was designed as well, and it has led to satisfying results. In this case, the employed OS is Symbian.

Many of the existing speaker count methods are based on the computation of feature vectors derived from the time- and/or frequency- domain of audio signals and subsequent labeling using generic classifiers. While performance varies based on the considered application scenario, the best results exceed 70% classification accuracy.

Audio-based gender recognition is also commonly carried out through generic classifiers, with pitch being the most frequently used feature, although many different spectral features have also been employed. Classification accuracies are better than 90% for most methods.

However, available algorithms are not designed specifically for mobile device implementation and do not take into account smartphone processing power and time requirements of context-aware applications.

Speaker Count and Gender Recognition Approach

As previously mentioned, to give a practical idea of the possibilities offered by smartphone platforms, the implemented method proposed by the authors in (Bisio, 2010) has been taken as reference in the following. As briefly described below, the basic concept of the employed method (Pitch estimation) can be employed for both Speaker Count and Gender Recognition approaches as listed below.

- ***Pitch Estimation***. For voiced speech, pitch can be defined as the rate of vibration of the vocal folds (Cheveignè, 2002), so it can be considered a reasonably distinctive feature of an individual. The basic idea of the proposed speaker count method is that if audio sample pitch estimates have similar values, the sample is 1S. If different pitch values are detected the sample is 2S. In (Bisio, 2010) a pitch estimation method based on the signal's autocorrelation was used because of its good applicability to speech and ease of implementation. Since pitch is linked to the speech signal's periodicity, the autocorrelation of a speech sample will present its highest values at very short delays and at delays corresponding to multiples of pitch periods. To estimate the pitch of an audio frame, the frame's autocorrelation is first computed in the delay interval corresponding to the human voice pitch range (50-500 Hz). The peak of this portion of autocorrelation is then detected

and the pitch is estimated as the reciprocal of the delay corresponding to the autocorrelation peak.

- ***Speaker Count***. An audio sample is divided into abutted frames and pitch estimates are computed for each frame. A given number of consecutive frames is grouped together in blocks, in order to allow the computation of a pitch Probability Distribution Function (PDF) for each block. Adjacent blocks are overlapped by a certain number of frames. The values adopted in this paper are summarized in Table 1. A block's PDF is computed by estimating the pitch and computing the power spectrum (via Fourier Transform) for each of the block's frames. For every estimate the value of the PDF bin, which represents a small frequency interval, containing that pitch is increased with the frame's power, obtained from the computed power spectrum, at the frequency corresponding to the pitch estimate.

Compared to computing PDFs by simply executing a "histogram count" (which increases by 1 the value of a PDF bin for every pitch estimate falling into such bin), this mechanism allows distinguishing higher-power pitch estimates with respect to lower-power ones. It leads to more distinct PDFs and, as a consequence, more accurate features.

In order to recognize individual blocks, a feature vector representing the dispersion of its pitch estimate PDF, composed of its maximum and standard deviation, is extracted and used by a GMM classifier to classify the block as either 1S or 2S. Once all individual blocks have been recognized, the audio sample is finally classified through a "majority vote" decision.

- ***Gender Recognition***. In addition to the speaker count algorithm, a method for single-speaker gender recognition was designed as well. It was observed that satisfying results could be obtained by using a single-feature threshold classifier, without resorting to GMMs. In this case, the chosen feature is the mean of the blocks' "histogram count" PDF. In fact, pitch values for male speakers are on average lower compared to female speakers, since pitch can be defined as the vibration rate of the vocal folds during speech and male vocal folds are greater in length and thickness compared to female ones. Individual blocks are classified as "Male" (M) or "Female" (F) by comparing their PDF mean with a fixed threshold computed based on a training set. "Histogram count" PDFs are employed because it was observed that the derived feature was sufficiently accurate. The weighted PDFs, which require the Fourier Transform of individual frames, are not used to significantly reduce the time required by the smartphone application to classify unknown samples.

Experiments and Results

In order to train the GMM and threshold classifiers an audio sample database has been employed. It was acquired using a smartphone, thus allowing the development of the proposed methods based on data consistent with the normal execution of the smartphone applications. The considered situations were 1 male speaker (1M), 1 female speaker (1F), 2 male speakers (2M), 2 female speakers (2F) and 1 male and 1 female speaker (2MF).

Table 1. Experimental setup

Sampling Frequency	22KHz
Frame Duration	2048 samples
Frames per Block	20
Block Overlap	10 frames
PDF bin Width	10 Hz

All audio samples refer to different speakers in order to evaluate classifier performance using data deriving from speakers that did not influence classifier training (open-set scenario). A total of 50 recordings was acquired, 10 for each situation. Half of the recordings was used for training, the other half for testing. The parameters used during experiments have been set to the values shown in Table 1.

Concerning Speaker Count results, different feature vectors were evaluated and comparison of test set classification accuracies was used to select the most discriminating one. A 3-dimensional feature vector performed better than some 2-dimensional ones, but adding a fourth feature does not improve classification accuracy, since it is rather correlated with the others. The feature vector ultimately used for GMM classification, as previously mentioned, comprises the maximum and standard deviation of block PDFs, and it leads to 60% test sample accuracy.

An additional set of experiments ignoring the situations that most of all led to classification errors, i.e. 2M and 2F, was carried out. In fact, these two situations can be misclassified as 1M and 1F, respectively, since same-gender speakers could have pitch estimates close enough in value to lead to 2S PDFs similar to 1S PDFs of the same gender. Therefore a new GMM classifier was designed, in order to distinguish not two classes (1S and 2S) but three classes: 1M, 1F and 2MF. Again, different feature vectors were evaluated, and for each one classification errors involved exclusively class 2MF, i.e. test sample blocks belonging to classes 1M and 1F were never mistaken one for another. The chosen feature vector consists of the mean and maximum of block PDFs, and it leads to 67% test sample accuracy. In order to compare this result with the first set of experiments, classes 1M and 1F can be considered as class 1S and class 2MF as class 2S, producing a 70% test sample accuracy.

Concerning Gender Recognition results, in order to identify the gender of single speakers the threshold on the mean of the "histogram count" pitch PDF was set to the value that led to the best training sample block classification results. The designed classifier leads to 90% test sample accuracy. While both classes are well-recognized, the totality of female samples was correctly classified.

NETWORK INTERFACE SIGNAL PROCESSING BASED CONTEXT-AWARE SERVICES: POSITIONING

Navigation and positioning systems have become extremely popular in recent years. In particular, thanks to an increasingly widespread dissemination and decrease in costs, devices such as GPS receivers are much more efficient for what concerns positioning systems. Obviously, positioning information represents very useful context information which can be exploited in several applications.

The positioning problem may be divided into two families: outdoor and indoor positioning. Several algorithms are available in the literature together with several approaches based on the use of different hardware platforms such as RF (Radio Frequency) technology, ultrasound-, infrared-, vision-based systems and magnetic fields. The RF signal-based technologies can be split into WLAN (2.4 GHz and 5 GHz bands), Bluetooth (2.4 GHz band), Ultrawideband and RFID (Gu, 2009; Moutz, 2009). Concerning the Outdoor positioning, GPS is the most popular and widely used three-dimensional positioning technology in the world. However, in many everyday environments such as indoors or in urban areas, GPS signals are not available for positioning (due to the very weak signals). Even with high sensitivity GPS receivers, positioning for urban and indoor environments cannot be guaranteed in all situations, and accuracies typically range between tens and hundreds of meters. As claimed in (Barnes, 2003), other emerging technologies obtain positions from systems that are not designed for positioning, such as mobile phones or television.

As a result, the accuracy, reliability and simplicity of the position solution is typically very poor in comparison to GPS with a clear view of the sky.

Despite this viewpoint, due to the widespread employment of smartphones, it is interesting to develop possible algorithms suited to be used with those platforms. In particular, based on our previous practical experience, we present in the following the Indoor positioning approaches, based on the Fingerprinting criteria and actually implemented on smartphones, which are introduced to give readers a tangible idea of the possible context-aware applications that such platforms may support.

Indoor Positioning

In more detail, with the increasing spread of mobile devices such as Personal Digital Assistants (PDA), laptops and smartphones, along with a great expansion of Wireless Local Area Networks (WLAN), there is a strong interest in implementing a positioning system which uses similar devices. In addition to this, with a great diffusion of the wireless communications network infrastructure and an increasing interest in location-aware services, there is a need for an accurate indoor positioning technique based on WLAN networks.

Obviously, WLAN is not designed and deployed for the purpose of positioning. However, measurements of the Signal Strength (SS) transmitted by either Access Point (AP) or station could be used to calculate the location of any Mobile User (MU). Many SS-based techniques have been proposed for position estimation in environments in which WLAN is deployed. In the following, the most common approaches in the literature have been described. There are essentially two main categories of techniques which can implement wireless positioning. The first is called trilateration, while the second fingerprinting.

Trilateration

It employs a mathematical model to "convert" the SS received by the terminal in a measure of the distance between the terminal itself and the corresponding AP. Obviously this distance does not provide any information about the direction in which the device is located. For this reason, it is assumed that the terminal is situated upon a circle centered in the considered AP, with a radius equal to the determined distance.

It is worth noticing that the SS is very sensitive to small changes in the position and orientation of the antenna of the terminal, thus making it particularly difficult to determine an analytical relationship linking SS to distance. In particular, the trilateration approach consists of two steps:

- converting SS to AP-MU distance (*Off-Line*);
- computing the location using the relationship between SS and distance (*On-Line*);

In the first step, a signal propagation model is employed to convert SS to AP-MU distance. This is the key of the trilateration approach and must be as accurate as possible. In the second step, least squares or other methods (such as the geometric method) can be used to compute the location. A positioning process ma start from the classical formulation of the propagation loss for radio communications L_F

$$L_F = 10\log(\frac{P_R}{P_T}) = 10\log G_T + 10\log G_R - 20\log f - 20\log d + K$$

where $K = 20\log\left(3 \cdot 10^8 / 4\pi\right) = 147.56$, G_T and G_R are the transmitter and the receiver antennas' gain, respectively. P_R and P_T are the receive power and the transmitted power. f and d are the frequency and the distance, respectively. In an ideal situation (i.e. G_T and G_R equal to 1), it is

possible to define the basic transmission loss (L_B) as:

$$L_B = -32.44 - 20\log f_{MHz} - 20\log d_{km}$$

where d_{km} represents distance (in *Km*) and f_{MHz} represents frequency (in *MHz*). All the equations above are referred to a free space situation, without any kind of obstacles. Thus they do not take into account all the impairments of a real environment, such as reflections, refractions and multipath fading.

In order to solve this problem and determine the mathematical relation, without considering physical properties, an empirical model based on regression was assumed. The key idea is to collect several measurements of SS at the same point, at increasing distance from the AP considered. After the acquisition phase, all measures taken at the same point are averaged with the aim to obtain a more stable reference value of SS. The obtained set of {SS-distance} pairs is then interpolated by a polynomial technique, for example by the least-squares method, producing an expression of the distance as a function of the SS. In more detail, empirical studies have proved that a cubic regressive equation results adequate to obtain a model representative of a trade-off between accuracy and computational load.

If this process is repeated for each AP used in the algorithm and the number of APs of the WLAN is at least three, this method allows estimating the distances between the device (the smartphone in the case of this chapter) and all the APs. The positioning process is then concluded by computing the intersection of three circles with radiuses equal to the estimated distances.

Fingerprinting

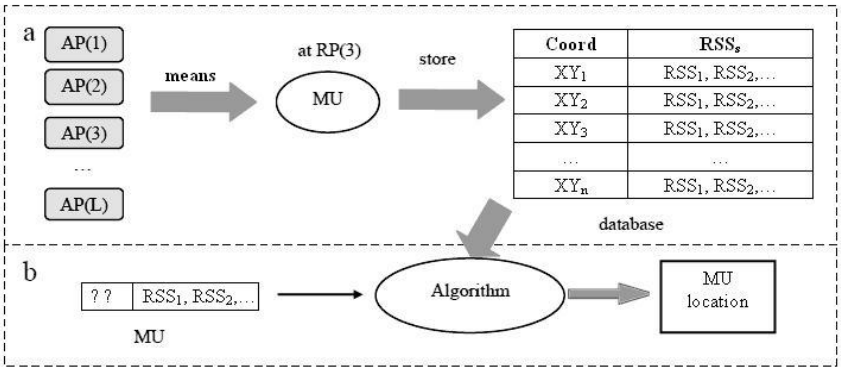
A positioning system which uses the fingerprinting approach is again composed by two phases: training (*Off-Line*) and positioning (*On-Line*). In the framework of the research activity conducted by the authors, this approach has been preferred for smartphone implementation because of its limited computational complexity. In more detail, the aim of the training phase is to build a database of fingerprints, whose meaning is clarified in the following, which is used to determine the location of the device. To generate the database, it is necessary to carefully define the reference points (RPs), i.e. the points where the fingerprints will be taken, and the concept of fingerprint itself. The RPs must be chosen in the most uniform way possible, in order to cover the entire area of interest as homogeneously as possible.

The acquisitions are made by placing the device in each RP and measuring the SS of all the APs. From all acquisitions, a distinguishing and robust feature called fingerprint is determined for each AP, taking the average value of the measured SS. This feature is then stored in the database. This process is repeated for all RPs.

During the positioning phase (*On-Line*) the device measures the SS of all the APs, i.e. it determines the fingerprint corresponding to its current position. This imprint is then compared with the fingerprints stored in the database by using an appropriate algorithm for comparison, described in the following. Obviously, the final result of this operation is the estimated position of the device. Figure 5, adapted from (Binghao, 2006), schematically summarizes the two phases of the briefly described process.

In the framework of the described fingerprinting approach there are many algorithms to determine the position of the device. The simplest of all is the so-called Nearest Neighbor (NN). This method determines the "distance" between the measured fingerprint $[s_1, s_2 \ldots s_n]$ and those which are in the database $[S_1, S_2 \ldots S_n]$. The "distance" between these two vectors is determined by the general formula below:

Figure 5. Two phases of the fingerprinting approach: (a) training phase and (b) positioning phase adapted from (Binghao, 2006)



$$L_q = \left(\sum_{i=1}^{n} \left| s_i - S_i \right|^q \right)^{\frac{1}{q}}$$

In particular, the Manhattan and Euclidean distance are obtained with $q = 1$ and $q = 2$, respectively. Experimental tests have shown that the Manhattan distance provides better performance in terms of precision. The NN method defines the Nearest Neighbor to be the RP with the minimum distance, in terms of the equation given above, from the fingerprint acquired by the device. That point is the position produced by the simple NN approach.

Another method for determining the position is to employ the K-Nearest Neighbors (KNN, with $K \geq 2$) that uses the K RPs with the minimum distance from the measured *fingerprint*, and estimates the position of the device by averaging the coordinates of the K points found.

A variant of the this method is the Weighted K-Nearest Neighbors (WKNN), in which the estimated position is obtained by making a weighted average. One of the possible strategies to determine the weights (w_i) could be using the inverse value of the distance, as shown in the equation below:

$$w_i = \frac{1}{L_q}$$

A Probabilistic Approach for the Fingerprinting Method

While the previously described deterministic method achieves reasonable localization accuracy, it discards much of the information present in the training data. Each *fingerprint* summarizes the data as the average signal strength to visible access points, based on a sequence of signal strength values recorded at that location. However, signal strength at a position can be characterized by more parameters than just the average. This led researchers to consider a Bayesian approach to WLAN localization. This had been employed with some success in the field of robot localization (Ladd, 2002). For localization, the Bayes rule can be written as

$$p(l_t / o_t) = p(o_t / l_t)\, p(l_t)\, N$$

where l_t is a position at time t, o_t is an observation at t (the instantaneous signal strength values), and N is a normalizing factor that ensures that all probabilities sum to 1.

In other words, the probability of being at location l_t given observation o_t is equal to the probability of observing o_t at location l_t, and being at location l_t in the first place. During localization, this conditional probability of being at location l_t is calculated for all *fingerprints*. The most likely position is then the localizer's output. To calculate this, it is necessary to calculate the two probabilities on the right hand side of the equation above. The quantity $p(o_t/l_t)$ is known, in Bayesian terms, as the likelihood function. This can be calculated using the signal strength map. For each *fingerprint*, the frequency of each signal strength value is used to generate a probability distribution as the likelihood function. The raw distribution can be used, but as it is typically noisy and incomplete, the data is usually summarized as either a histogram, with an empirically determined optimal number of bins, or as a discrete Gaussian distribution parameterized using mean and standard deviation.

Other representations are also possible; the Bayesian approach allows using any algorithm capable of generating a probability distribution across all positions.

In its simplest version, the Bayesian localizer calculates the prior probability $p(l_t)$ as the uniform distribution over all possible positions. This means that, before each positioning attempt, the target is equally likely to be at any of the position in the *fingerprint* map. In order to achieve higher accuracy, it is possible to compute such probability using the knowledge given by historical information such as user habits, collision detection, and anything else that affects the prior probability that can be modeled probabilistically. For example, Markov Localization (Simmons, 1995) suggests using the transitional probability between positions. This probability is described as

$$p(l_t) = \sum_{l_{t-1}} p(l_t \,/\, l_{t-1}) p(l_{t-1})$$

In other words, $p(l_t)$ is the sum of the transitional probability from all positions at $t-1$ to l_t at the current time t, multiplied by the probability of being at those locations at $t-1$. The probability $p(l_{t-1})$ is known from previous positioning attempts.

Test-Bed Description and Preliminary Results

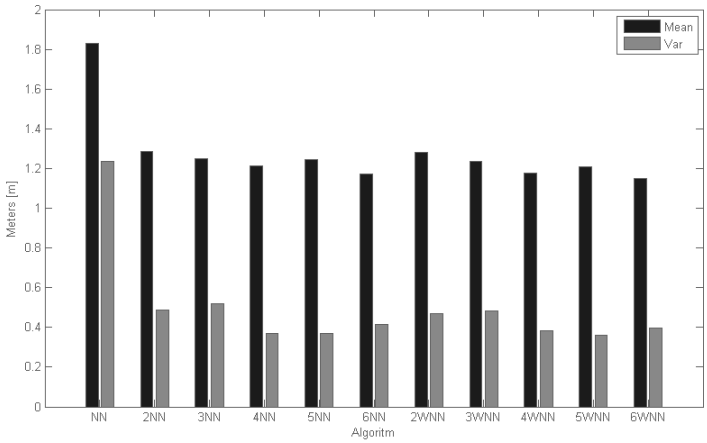
From a practical point of view, certain fingerprinting algorithms have been implemented on a smartphone platform and preliminary results are reported in the following. In more detail, to test the algorithm implementation an *ad hoc* test-bed was set up in the Digital Signal Processing Laboratory at the University of Genoa where the authors are developing their research activity. The room is approximately 8 *m* ×8 *m* in size. In the performed tests five APs have been installed, four of them in the corners of the room and one in the center. All the APs' antennas are omni-directional.

In general, the position of the APs in the room plays a crucial role because it is linked to the accuracy and precision of the system. In particular, (Wang, 2003) shows very clearly that the more APs are installed the better the performance is.

To evaluate the validity of the implemented algorithms, several tests were carried out. In particular, 30 measurements were taken in different parts of the test-bed. For each of these measurements the position was determined using the deterministic fingerprinting algorithm. In particular, all the algorithms previously described (NN, KNN, WKNN) have been compared. The database employed for all the experiments contains 121 RPs, separated 0.6 *m* one from another. The histogram below reports the obtained results.

Figure 6 shows that the simplest and quickest algorithm (i.e. NN) does not provide good results. All other algorithms have a mean error around 1.2 *m*. For this particular set of measures, 6WNN has the best performance in terms of the lowest positioning error (i.e. the distance, expressed in *m*, between the real position and the estimated

Figure 6. Mean and variance of the error for all the algorithms utilized in the fingerprinting positioning system. These results are obtained with a database of 121 RPs and 5 APs



one) and a very low variance. Empirical experiments presented in (Binghao, 2006) prove that the probabilistic approach provides slightly better performance. It indicates that probabilistic methods are relatively robust with respect to naturally occurring fluctuations.

Brief Overview of Other Approaches

For the sake of completeness other common approaches are synthetically described below, as reviewed in (Bose, 2007):

- ***Angle of Arrival*** (AOA) refers to the method by which the position of a mobile device is determined by the direction of the incoming signals from other transmitters whose locations are known. Triangulation techniques are used to compute the location of the mobile device. However, a special antenna array is required to measure the angle.
- ***Time of Arrival*** (TOA) method measures the round-trip time (RTT) of a signal. Half of the RTT corresponds to the distance of the mobile device from the stationary device. Once the distances from a mobile device to three stationary devices are estimated, the position of the mobile device with respect to the stationary devices can easily be determined using trilateration. TOA requires very accurate and tightly synchronized clocks since a 1.0 μs error corresponds to a 300 m error in the distance estimation. Thus, inaccuracies in measuring time differences should not exceed tens of nanoseconds since the error is propagated to the distance estimation.
- ***Time Difference of Arrival*** (TDOA) method is similar to Time of Arrival using the time difference of arrival times. However, the synchronization requirement is eliminated but nonetheless high accuracy is still an important factor. As in the previous method, inaccuracies in measuring time differences should not exceed tens of nanoseconds.

ACCELEROMETER SIGNAL PROCESSING BASED CONTEXT-AWARE SERVICES

The physical activity which the user is currently engaged in can be useful context information, e.g. it may be employed to support remote healthcare

monitoring (Leijdekkers, 2007), to update the user's social network status (Miluzzo, 2008) or even to reproduce inferred real-world activities in virtual settings (Musolesi, 2008).

In the following, a smartphone algorithm for user activity recognition is described. It is based on the sensing, processing and classification of data provided by the smartphone-embedded accelerometer and is designed to recognize four different classes of physical activities. It is worth noticing that the described approach has been practically implemented and tested by our research group. It represents a *proof of concept* and as such it can be considered as a useful reference for readers' comprehension.

User Activity Recognition

Four different user activities have been considered. Unless specified differently, the phone is thought to be in the user's front or rear pants pocket (as suggested in (Bao, 2004)) and training data was acquired accordingly. Furthermore, the acquisition of training data was performed keeping the smartphone in four different positions, based on whether the display was facing towards the user or away from him and whether the smartphone itself was pointing up or down. The evaluated classes are:

- ***Sitting***: the user is sitting down. Training data was acquired only with the smartphone in the front pocket, under the assumption that it's unlikely users will keep the smartphone in a back pocket while sitting.
- ***Standing***: the user is standing up, without walking. Satisfactory distinction from *Sitting* is possible due to the fact that people tend to move a little while standing.
- ***Walking***: the user is walking. Training data for this class was acquired in real-life scenarios, e.g. on streets, in shops, etc.
- ***Running***: the user is running. As for *Walking*, training data for this class was acquired in common every-day scenarios.

Sensed Data

The smartphone employed during the work is an HTC Dream, which comes with an integrated accelerometer manufactured by Asahi Kasei Corporation. It's a triaxial, piezoresistive accelerometer which returns the acceleration values on the three axes in m/s^2.

The proposed algorithm periodically collects raw data from the smartphone accelerometer and organizes it into frames. A feature vector is computed for every frame and is used by a decision tree classifier to classify the frame as one of the classes previously listed. Groups of consecutive frames are organized in windows, with consecutive windows overlapped by a certain amount of frames. Every completed window is assigned to one of the four considered classes, based on one of several possible decision policies. Such windowed decision is considered as the current user activity. Therefore, several parameters are involved in the data acquisition. First of all, a frame's duration must be set: shorter frames mean quicker feature computation, but the minimum length required to properly recognize the target activities must be considered as well. The frame acquisition rate must also be determined, i.e. how often must the accelerometer be polled. Higher frame rates imply shorter pauses between frames and a more precise knowledge of the context, but also more intensive computation. On the other hand, lower frame rates provide a less precise knowledge of the context, but also imply a lighter computational load, which is an important requirement for smartphone terminals.

The window size affects the windowed decision which determines the current state associated with the user. Small windows ensure a quicker reaction to actual changes in context, but are more vulnerable to occasionally misclassified frames.

On the other hand, large windows react more slowly to context changes but provide better protection against misclassified frames. The window overlap (number of frames shared by consecutive windows) must also be set. Employing heavily-overlapped windows provides a better knowledge of the context but may also imply consecutive windows bearing redundant information, while using slightly-overlapped windows could lead to signal sections representing meaningful data falling across consecutive windows.

Feature Computation and Frame Classification

In order to determine the best possible classifier, numerous features were evaluated and compared, among which were the mean, zero crossing rate, energy, standard deviation, cross-correlation, sum of absolute values, sum of variances and number of peaks of the data obtained from the accelerometer. The feature vector ultimately chosen is made of nine features, i.e. the mean, standard deviation and number of peaks of the accelerometer measurements along the three axes of the accelerometer.

Once a feature vector has been computed for a given frame, it is used by a classifier in order to associate the frame to one of the classes listed before. As in earlier work (Bao, 2004; Musolesi, 2008; Tapia, 2007; Ryder, 2009), the employed classifier is a decision tree. Using the Weka workbench (a tool dedicated to Machine Learning procedure), several decision trees were designed and compared based on their recognition accuracy. A decision tree was trained for every combination of two and three of the users employed in the dataset creation (see the brief performance evaluation reported below). In order to evaluate the classifiers' performance, a separate test set (made of the dataset portion not used for training) was used for each combination.

Classification Scoring and Windowed Decision

Every completed window is assigned to one of the four considered classes, based on one of several possible decision policies. The simplest of such policies is a majority-rule decision: the window is associated to the class with the most frames in the window. While it is clearly simple to implement and computationally inexpensive, the majority-rule windowed decision treats all frames within a window in the same way, without considering when the frames occurred or the single frame classifications' reliability.

Therefore, other windowed decision policies were evaluated. It must be noted that such policies are completely independent from the decision tree used to classify individual frames.

A first alternative to the majority-rule decision is the time-weighted decision. In a nutshell, it implies giving different weights to a window's frames based solely on their position in the window and assigning a window to the class with the highest total weight. This way a frame will have a greater weight the closer it is to the end of the window, under the assumption that more recent classifications should be more useful to determine the current user activity.

In order to determine what weight to give to frames, a weighting function $f(t)$ was designed according to the following criteria:

- $f\left(0\right) = 1$, where $t = 0$ represents the time at which the most recent frame occurred;
- $f\left(t\right) \geq 0$ for all $t \geq 0$;
- $f\left(t\right)$ must be non-increasing for all $t \geq 0$.

If T_f is the instant associated with a frame and T_{dec} is the instant at which the windowed decision is made, then the frame will be assigned a weight equal to $f(T_{dec} - T_f)$.

Two different weighting functions were compared, i.e. a Gaussian function and a negative

Exponential function. For each function type five different functions were compared by choosing a reference instant T_{ref} and forcing $f(T_{ref}) = p$, where p is one of five linearly-spaced values between 0 and 1.

A second kind of windowed decision policy requires assigning to each frame a score representing how reliable its classification is. As in the case of the time-weighted decision, a window is associated to the class with the highest total weight. Unlike other classifier models, the standard form of decision tree classifiers does not provide ranking or classification probability information. In the literature there are numerous approaches to extend the decision tree framework to provide ranking information (Ling, 2003; Alvarez, 2007). In our work the method proposed in (Toth, 2008) has been implemented.

Such scoring method takes advantage of the fact that each leaf of a decision tree represents a region in the feature space defined by a set of inequalities determined by the path from the tree root to the leaf. The basic idea is that the closer a frame's feature vector is to the decision boundary, the more unreliable the frame's classification will be, under the hypothesis that the majority of badly classified samples lie near the decision boundary.

This scoring method requires the computation of a feature vector's Mahalanobis distance from the decision boundary and an estimate of the correct classification probability. The Mahalanobis distance is used instead of the Euclidean one because it takes into account the correlation among features and is scale invariant. This ensures that if different features have different distributions, the same Mahalanobis distance along the direction corresponding to a feature with greater deviation will carry less weight than along the direction corresponding to a feature with lesser deviation.

The distance of a feature vector from the decision boundary is given by the shortest distance to the leaves with class label different from the label associated to the feature vector. The distance between a feature vector and a leaf is obtained by solving a constrained quadratic program.

Using separate training data for each leaf, an estimate of the correct classification probability conditional to the distance from the decision boundary is produced. Such estimate is computed by using the leaf's probability of correctly and incorrectly classifying training set samples (obtained in terms of relative frequency) and probability density of the distance from the decision boundary conditional to correct and false classification.

The classification score is finally given by the lower bound of the 95% confidence interval for the estimate of the correct classification probability conditional to the distance from the decision boundary.

The confidence interval lower bound is used instead of the correct classification probability conditional to the distance from the decision boundary estimate because the latter may remain close to 1 even for large distances. However, a large distance may not imply a reliable classification but be caused by an unknown sample located in a region of the feature space insufficiently represented in the training set. On the contrary, past a certain distance (which varies with every leaf), the confidence interval lower bound decreases rapidly.

Another windowed decision policy is given by combining the temporal weights and the classification scores into a single, joint time-and-score weight. Fusion is obtained simply by multiplying the corresponding time weight and classification score, since both are between 0 and 1.

By considering the described methods as single approaches and by mixing them it is possible to obtain six approaches. In particular, Majority decision (M), Exponential time weighting (T_e), Gaussian time weighting (T_g), classification score weighting (S), joint classification score / exponential time weighting ($S+T_e$), joint classification score / Gaussian time weighting ($S+T_g$). The performance comparison among them is briefly described below. It is worth noticing that what has been described previously is object of

ongoing research by the Authors of this chapter and further details about such solutions will be provided in the future.

Brief Performance Evaluation

The dataset employed in the experiments was acquired by 4 volunteers. Each volunteer acquired approximately 1 hour of data for each of the classes listed above, producing a total of almost 17 hours of data. The employed OS was Android. For every combination of two and three users, the dataset was then divided into a training set for classifier training and a distinct test set for performance evaluation purposes. In evaluating the performance of the proposed method, one must distinguish single-frame classification accuracy from windowed-decision accuracy: the former depends solely on the decision tree classifier, while the latter depends on what decision policy was used.

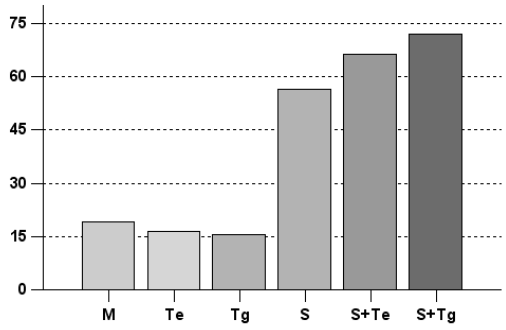
Of all the evaluated classifiers, the one with the best single-frame accuracy produced a 98% correct test set classification average. In more detail, the class with the best recognition accuracy is *Sitting* (over 99% of test set frames correctly recognized), while *Running* is the activity with the lowest test set accuracy (95.2%), with most of the incorrectly classified frames (approximately 4.6% of the total) being misclassified as *Walking*. Such results are extremely satisfactory: in particular, the implemented classifier led to an improvement of the activity recognition accuracy of more than 20% compared to (Miluzzo 2008) which considers the same classes and also uses decision tree classification, although the employed dataset is somewhat smaller.

As for the windowed decision, an additional *ad hoc* sequence, not included in the dataset used for classifier training and testing, was employed to determine the best values for the frame acquisition rate, window size, window overlap and scale parameter for the time-weighting functions briefly described above. Such sequence is made of just over an hour of data, referring to all four considered user activities executed in random order. Windowed decision was applied to the *ad hoc* sequence using 411 different parameter configurations and all six above-mentioned decision policies for each parameter combination. The results can be summed up in Figure 7. Using only the time-based frame classification weighting does not seem to improve performance compared to the majority decision, while employing classification score weighting, by itself or combined with time weighting, led to significant improvements in windowed decision accuracy. Overall, the best parameter configuration led to an 85.2% windowed decision accuracy: it was obtained using 16-second pauses between consecutive frames, 8-frame windows, single-frame window overlap and joint classification score / Gaussian time weighting

CONCLUSION

The proposed chapter is based on the authors' previous research experience and ongoing work and it is aimed at giving an idea of possible context-aware Services for smartphones considering and describing algorithms and methodologies practi-

Figure 7. Percentage of total number of evaluated parameter configurations in which each windowed decision policy gave the best correct windowed decision percentage



cally designed and implemented for such devices. In more detail, such methods have been exploited to implement particular Context-Aware services aimed at recognizing the audio environment, the number and the gender of speakers, the position of a device and the user physical activity by using a smartphone. The practical implementation of these services, capable of extracting useful context information, is the main technical objective of this work.

Starting from the open issues considered in this chapter and from the literature in the field, it is clear that context awareness needs to be enhanced with new efficient algorithms and needs to be developed in small, portable and diffused devices. Smartphones have the mentioned characteristics and, as a consequence, they may represent the target technology for future Context-Aware services. In this context, the lesson learned by the authors is that an important effort in terms of advanced signal processing procedure that exploit the smartphone feature, sensors and computational capacity need to be done and what has been presented in this chapter represents the first step in that direction.

The development of efficient signal processing procedure over smartphone opens the doors to future application of smartphone-based context-aware services in several fields. Two important sectors may be the safety and the remote assistance. In the first case, information about the audio environment, the position and the movement that the personnel dedicated to the surveillance of a sensitive area, acquired by using their smartphones, may represent a useful input for advanced surveillance systems. In the second case, remote monitoring of patients or elders that need to be monitored can be realized as well. Position, outdoor or indoor (within their domestic ambient), and movements constitute useful input for physicians to monitor the lifestyle of patients or to individuate possible emergency cases.

The evolution of the signal procession procedures for smartphones and the application of them to realize context-aware service for safety and health-care platforms constitute the future direction for the research in the presented field.

ACKNOWLEDGMENT

The authors wish to deeply thank Dr. Alessio Agneessens and Dr. Andrea Sciarrone for their precious support in the implementation and testing phase of this research work and for their important suggestions.

REFERENCES

Alvarez, I., Bernard, S., & Deffuant, G. (2007). Keep the decision tree and estimate the class probabilities using its decision boundary. *Proceedings of the International Joint Conference on Artificial Intelligence*, (pp. 654-660).

Bao, L., & Intille, S. S. (2004). *Activity recognition from user-annotated acceleration data*. In 2nd International Conference, PERVASIVE ’04.

Barnes, J., Rizos, C., Wang, J., Small, D., Voigt, G., & Gambale, N. (2003). High precision indoor and outdoor positioning using LocataNet. *Journal of Global Positioning Systems*, *2*(2), 73–82. doi:10.5081/jgps.2.2.73

Binghao, L., James, C. S. R., & Dempster, A. G. (2006). Indoor positioning techniques based on wireless LAN. In *Proceedings of Auswireless Conference 2006*.

Bisio, I., Agneessens, A., Lavagetto, F., & Marchese, M. (2010). Design and implementation of smartphone applications for speaker count and gender recognition. In Giusto, D., Iera, A., Morabito, G., & Atzori, L. (Eds.), *The Internet of things*. New York, NY: Springer Science.

Bose, A., & Foh, C. H. (2007). A practical path loss model for indoor Wi-Fi positioning enhancement. In *Proc. International Conference on Information, Communications & Signal Processing (ICICS)*.

de Cheveignè, A., & Kawahar, H. (2002). A fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, *111*(4). doi:10.1121/1.1458024

Dey, A. K., & Abowd, G. D. (2000). Towards a better understanding of context and context awareness. In *The What, Who, Where, When, Why and How of Context-Awareness Workshop at the Conference on Human Factors in Computing Systems (CHI)*.

Doets, P. J. O., Gisbert, M., & Lagendijk, R. L. (2006). On the comparison of audio fingerprints for extracting quality parameters of compressed audio. *Security, steganography, and watermarking of multimedia contents VII, Proceedings of the SPIE*.

Freescale Semiconductor, Inc. (2008). *Mobile extreme convergence: A streamlined architecture to deliver mass-market converged mobile devices*. White Paper of Freescale Semiconductor, Rev. 5.

Gu, Y., Lo, A., & Niemegeers, I. (2009). A survey of indoor positioning systems for wireless personal networks. *IEEE Communications Surveys & Tutorials, 11*(1).

Haitsma, J., & Kalker, T. (2002). A highly robust audio fingerprinting system. In *Proceedings of the International Symposium on Music Information Retrieval, Paris, France*.

Iyer, A. N., Ofoegbu, U. O., Yantorno, R. E., & Smolenski, B. Y. (2006). Generic modeling applied to speaker count. In *Proceedings IEEE, International Symposium On Intelligent Signal Processing and Communication Systems, ISPACS'06*.

Ladd, A. M., Bekris, K. E., Rudys, A., Marceau, G., Kavraki, L. E., & Dan, S. (2002). *Robotics-based location sensing using wireless ethernet*. Eighth ACM Int. Conf. on Mobile Computing & Networking (MOBICOM) (pp. 227-238).

Lee, Y., Mosley, A., Wang, P. T., & Broadway, J. (2006). *Audio fingerprinting from ELEC 301 projects*. Retrieved from http://cnx.org/content/m14231

Leijdekkers, P., Gay, V., & Lawrence, E. (2007). *Smart homecare system for health tele-monitoring*. In ICDS '07, First International Conference on the Digital Society.

Ling, C. X., & Yan, R. J. (2003). Decision tree with better ranking. In *Proceedings of the International Conference on Machine Learning (ICML2003)*.

Marengo, M., Salis, N., & Valla, M. (2007). Context awareness: Servizi mobili su misura. *Telecom Italia S.p.A. Technical Newsletter, 16*(1).

Mautz, R. (2009). Overview of current indoor positioning systems. *Geodesy and Cartography*, *35*(1), 18–22. doi:10.3846/1392-1541.2009.35.18-22

Miluzzo, E., Lane, N., Fodor, K., Peterson, R., Lu, H., Musolesi, M., et al. Campbell, A. T. (2008). Sensing meets mobile social networks: The design, implementation and evaluation of the CenceMe application. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems* (pp. 337–350).

Musolesi, M., Miluzzo, E., Lane, N. D., Eisenman, S. B., Choudhury, T., & Campbell, A. T. (2008). The second life of a sensor - integrating real-world experience in virtual worlds using mobile phones. In *Proceedings of HotEmNets '08, Charlottesville*.

Peng, W., Ser, W., & Zhang, M. (2001). *Bark scale equalizer design using wrapped filter*. Singapore: Center for Signal Processing Nanyang Technological University.

Perttunen, M., Van Kleek, M., Lassila, O., & Riekki, J. (2009). *An implementation of auditory context recognition for mobile devices*. In Tenth International Conference on Mobile Data Management: Systems, Services and Middleware.

Ryder, J., Longstaff, B., Reddy, S., & Estrin, D. (2009). *Ambulation: A tool for monitoring mobility patterns over time using mobile phones*. Technical Report UC Los Angeles: Center for Embedded Network Sensing.

Simmons, R., & Koenig, S. (1995). *Probabilistic robot navigation in partially observable environments*. In The International Joint Conference on Artificial Intelligence (IJCAI'95) (pp. 1080-1087).

Tapia, E. M., Intille, S. S., Haskell, W., Larson, K., Wright, J., King, A., & Friedman, R. (2007). Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor. In *Proceedings of International Symposium on Wearable Computers, IEEE Press* (pp. 37-40).

Toth, N., & Pataki, B. (2008). Classification confidence weighted majority voting using decision tree classifiers. *International Journal of Intelligent Computing and Cybernetics*, *1*(2), 169–192. doi:10.1108/17563780810874708

Wang, Y., Jia, X., & Lee, H. K. (2003). An indoor wireless positioning system based on wireless local area network infrastructure. In *Proceedings 6th International Symposium on Satellite Navigation Technology*.

KEY TERMS AND DEFINITIONS

Smartphones: mobile phone with a significant computational capacity: several available sensors and limited energy.

Context-Aware Services: services provided to users obtained by considering the environment in which the users are and by considering the actions that users are doing.

Digital Signal Processing: theory and methodology to process numerical signals.

Pattern Recognition: theory and methodology to recognize a pattern.

Audio Environment Recognition: methodologies obtained from both signal processing and pattern recognition approaches aimed at individuating the environment based on the audio captured by a microphone (such as the smartphones' microphone).

Speaker Count and Gender Recognition: methodologies obtained from both signal processing and pattern recognition approaches aimed at individuating the number of speakers and the related genders based on the audio captured by a microphone (such as the smartphones' microphone).

Indoor Positioning: methodologies obtained from both signal processing and pattern recognition approaches aimed at individuating the position of users in a given environment (outdoor or indoor) based on radio signals captured by radio interfaces available on a given device (such as the smartphones' Bluetooth and WiFi interfaces).

Activity Recognition: methodologies obtained from both signal processing and pattern recognition approaches aimed at individuating the number type of movement that users are doing based on the accelerometer signals generated by an (such as the smartphones' accelerometer).