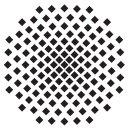

Universität Stuttgart

Institute of Parallel and Distributed Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Diploma Thesis Nr. 3749

Compressive Sensing-Based Data Uploading in Time-driven Public Sensing Applications

Na Chen

Course of Study:	Informatik
Examiner:	Prof. Dr. Kurt Rothermel
Supervisor:	Dr. Mohamed Abdelaal
Commenced:	2016-8-1
Completed:	2017-1-31
CR-Classification:	C.2.1,C.2.4

Abstract

Over the last few years, the technology of mobile phones greatly got increased. People gain and upload more and more information through their mobile phones in an easy way. Accordingly, a new sensing technology emerges, referred to as *public sensing* (PS). The core idea behind PS is to exploit the crowdedness of smart mobile devices to opportunistically provide real-time sensor data considering spatial and environmental dimensions. Recently, PS has been applied in many different application scenarios, such as environmental monitoring, traffic analysis, and indoor mapping. However, PS applications face several challenges. One of the most prominent challenges is the users acceptance to participate in the PS applications. In order to convince users to participate in the PS applications, several incentives mechanisms have been developed. However, the main two requirements – which should be met by any PS application – are the users' privacy and the energy costs of running the PS application.

In fact, there exist several energy consumers in PS applications. For example, many PS applications require the mobile devices to fix their position and frequently send this position data to the PS server. Similarly, the mobile devices waste energy when they receive sensing queries outside the sensing areas. However, the most energy-expensive task is to frequently acquire and send data to the PS server. In this thesis, we tackle the problem of energy consumption in a special category of PS applications in which the participating mobile devices are periodically queried for sensor data, such as acceleration and images.

To reduce the energy overhead of uploading lots of information, we exploit the fact that processing approximately one thousand instructions consumes energy equal to that of transmitting one bit of information. Accordingly, we exploit data compression to reduce the number of bit that will be transmitted from the participating mobile devices to the PS server. Although, the technical literature has many compression methods, such as derivative-based prediction, Cosine transform, Wavelet transform; we designed a framework based on the *compressive sensing* (CS) theory. In the last decade, CS has been proven as a promising candidate for compressing N-dimensional data. Moreover, it shows satisfactory results when used for inferring missing data. Accordingly, we exploit CS to compress 1D data (e.g. acceleration, gravity) and 2D data (e.g. images).

To efficiently utilize the CS method on "resources-taxed" devices such as the smart mobile devices, we start with identifying the most lightweight measurements matrices which will be implemented on the mobile devices. We examine several matrices, such as the random measurement matrix, the random Gaussian matrix, and the Toeplitz matrix. Our analysis mainly bases on the recovery accuracy and the dissipated energy from the mobile device's battery. Additionally, we perform a comparative study with other compressors, including the cosine transform and the lossless ZIP compressor. To further confirm that CS has a high recovery accuracy, we implemented an activity recognition algorithm at the server side. To

this end, we exploit the *dynamic time warping* (DTW) algorithm as a pattern matching tool between a set of stored patterns and the recovered data. Several experiments have been performed which show the high accuracy of both CS and DTW to recover several activities such as walking, running, and jogging. In terms of energy, CS significantly reduce the battery consumption relative to the other baseline compressors.

Finally, we prove the possibility of exploiting the CS-based compression method for manipulating 1D data as well as 2D data, i.e. images. The main challenge is to perform image encoding on the mobile devices, despite the complex matrix operations between the image pixels and the sensing matrices. To overcome this problem, we divide the image into a number of cells and subsequently, we perform the encoding process on each cell individually. Accordingly, the compression process is iteratively achieved. The evaluation results show promising results for 2D compression-based on the CS theory in terms of the saved energy consumption and the recovery accuracy.

Contents

List of Figures

Chapter 1

Introduction

In this chapter, we introduce the research problem and how we tackled it. We start with the motivating for the targeted research problem. Afterward, we present an overview of the public sensing technology. Generally, public sensing comprises several design factors. However, achieving a reasonable level of Quality of Service (QoS) is intuitively significant. Hence, we discuss some QoS matrices which are relevant to many real-world public sensing applications. Then, we dig further to discuss the source of our targeted problem, namely sensor data uploading. We explain the primary methods for data uploading between mobile devices and servers. This classification is merely important to identify the research problem. Before we explicitly explain the main objectives of this thesis, we visit the technical literature to check the related endeavours in the realm of optimizing data uploading between mobile devices and PS servers. Finally, we list the content of this thesis.

1.1 Motivation

In the recent years, during the development of the smart devices market, such as smartphones, tablets and smart glasses, people have been using different kinds of applications in smart devices for daily life. Not only for taking calls but also for tasks like checking emails, using instant messengers, entertainment, getting GPS information etc. More and more smartphones are equipped with multiple sensors like accelerometers, compasses and gyroscopes. For instance, the accelerometer sensors can be used for measuring the acceleration in all three axis. This information can be used for instance for sport applications that people can check up their daily sport and send those data to the server and to get a analysis for their health situation. Another example for using mobile devices for gathering data could be a weather database. The weather situation could be constructed with the information of the barometers of a group of mobile devices in combination with their position. Another possibility for using available GPS data of multiple devices is creating traffic information. For example, if a significant number of devices are driving more slowly then usually on a road this indicates a traffic problem.

Because of the advantages of smart devices such as that they are sensor-rich, Internet-enabled and exist everywhere, this approach could be very suitable for being a rich source of different information from multiple participants. As a result, a new technology emerges in the last few

years, referred to as *Public Sensing* (PS). The main idea behind PS systems is to opportunistically collect sensor data from multiple smart devices. The typical application for using PS could be environmental monitoring, traffic analysis, and indoor mapping.

Thanks to the recent evolution in the development of hardware components, the size of original data is getting larger and larger. For example, the typical resolution of a camera of a smartphone is rising to even more than 12 megapixels. In a similar manner, smartphones are equipped with more and more sensors where data could be gathered from. Smartphones are usually equipped with lithium-ion batteries with a range from 900mah to 3000mah. Some of the applications like GPS navigation can take a lot of battery. For the Public Sensing approach, which needs to be running in the background of the system, it is important to use data compression to transport the data between the smart devices and the server. This is necessary because even nowadays the monthly available amount of data which can be transferred is quiet limited at most cellphone contracts and the user would not accept that this data is extensively used by a background processes. A good compression algorithm can decrease the power consumption of the application and can make the battery lasting longer.

1.2 Problem Definition

The aim of this thesis is to solve the problem of energy consumption of smartphones participated in public sensing applications. The target is to develop a time-driven public sensing application for android smart phones, with which measurements are frequently reported to the server. The signals must be compressed before being sent from smartphone to the server. This reduces the amount of data which has to be transferred as well as the time needed for transmission. The shorter the time for the transmission is the longer the radio module can last in idle mode, this should reduce the battery consumption. In this application, the accelerometer is used to collect different activities from people like walking, holding, jumping and running. All those original signals will use a new highly compression method to compress, called Compressed Sensing.

In the sequel, we give an overview of the PS systems. We explain the different components and the associated QoS matrices, such as privacy, scalability, and energy consumption. Moreover, we discuss some of the well-known PS applications such as indoor mapping and real-time air quality monitoring.

1.3 Public Sensing

In the past years smartphones developed rapidly. Multi-core processors and gigabyte of flash memory bring them more and more performance. They are usually equipped with different kinds of sensors like accelerometers with multiple dimensions, a gyroscope and magnetic field sensors. Together this can be used for indoor-navigation. People can use their accelerometer sensors to collect the acceleration information and according to the orientation information can calculate the walking trace without GPS data. Another example is a fitness tracker

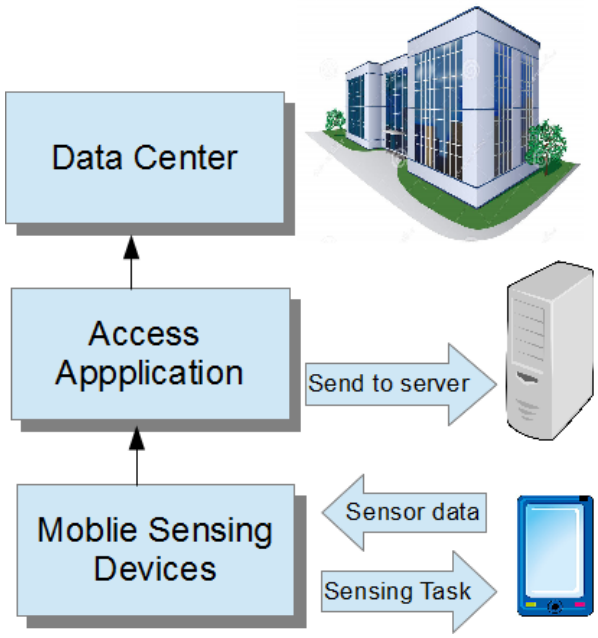

Figure 1.1: General architecture of public sensing systems

motion measurement, for this the built-in 3-axis accelerometer sensors can be used to track movements in every direction. The information from the gyroscope can be evaluated to measure the orientation and rotation. Other available sensors are: Temperature sensors, bio impedance sensors to check the temperature resistance of the user's skin and humidity sensors for instance to check the weather.

Smartphones are equipped as well with optical sensors like the high resolution front and the back camera, a light density sensor and even sometimes optical sensors to measure the pulse and heart rate. All of those collected information can be received for instance via Bluetooth by the smartphone or can be measured directly by the integrated sensors. At the smartphone side the information of the sensors can be brought to a user friendly format and be evaluated on the smartphone itself or transferred via network to another network node to evaluate the signals.

1.3.1 Overview of Public Sensing

Figure 1.1 shows the general architecture of a Public Sensing system. Public Sensing is divided into three levels. The first level is the mobile sensing device, which uses a smartphone for sampling. The second level is access application. In this level the data is uploaded to the server. The highest level is the data center, in where the captured data is utilized for data processing, for example for a health-care monitoring system.

The definition of Public Sensing is a presentation of using smart devices for gathering sensor data. The advantage of this idea is that this should be possible without having to build a

dedicated sensor network. Especially urban areas where there is a big number of smartphones available do offer here is a good playground. This gathered sensor data can be used for various applications, like measuring the level of noise or air pollution etc. The reasons for using Public Sensing are:

- In city areas, density of the smart devices is getting higher and higher. People are carrying billions of sensor nodes, generating indoor and outdoor information, depending on where the participates are.
- Because of this sensor data is recorded from privately owned smart devices, even small size companies don't need to worry about the investment cost of a large sensor network infrastructure. Commercial communities can also gather those sensor information for new services.

There are three points why building a public sensing system could be difficult:

- Quality of gathered data: Here quality means how accurate and precise the data is. Public sensing uses crowd-sensed data from different smart phone users, so that the data can and is most probably not as accurate compared to dedicated sensors, which will be installed at planed places and have some specifications for usage. For example the analysis of weather data. The positions where the sensors are mounted are chosen with specific criteria, for instance that temperature measurements are not done at places where the sensor is directly exposed to the sunlight or at a place with strong wind.

  The second point is that sensors which are used can be chosen to fit perfectly for the desired purpose. For instance a temperature range could range from -20 to 50 Degrees, and not -20 to 100 Degrees but therefor with a very high precision. When using crowd sourcing for sensor data there is also the possibility to measure even redundant values, that is because the distance between smart devices for collection of data is difficult to control.

- User acceptance: there are two main reasons why users may not accept public sensing. First, there could be some negative effects on the participant's smartphone, especially for the energy consumption. If public sensing reduces the battery level significantly this will be most probably not accepted by the user. The reason is that even without battery draining apps the battery of most smartphones lasts at maximum one or two days, depending on the model of the smartphone and the usage pattern. If the public sensing app decreases the battery life then to less than one day this will be most probably denied by the user. Secondly because of privacy. More and more people are caring about their smartphone's privacy. If a specific user can be tracked out of the created data with timestamp, position and sensor value this is not welcome by a user who is taking care about his privacy.

- Usability: Public sensing works using an application installed by the user. This type of application does get better with the amount of devices where it is installed on, so that it can collect more sensor data. The graphical user interface must be user friendly and the user should be able to experience the advantage of the data produced by others

Application Type	Function	Sensor
Environment monitoring	Noise measurement, Air quality measurement,Indoor mapping	microphone, air pollution sensor, Accelerometer
Transportation	traffic dynamic	GPS, compasses
Healthcare and Sport	Public health, Personal health, Activities recognize	GPS, Accelerometer, Pulse

Table 1.1: Public sensing applications[2]

or himself. For instance by being warned from traffic yams, etc. The location of the devices is more important than the pure number of devices.

Finally, for building a successful public sensing system, it should be quality-aware, energy-efficient, privacy-preserving and usable.

1.3.2 Public Sensing Applications

As table 1.1 shows, in recent years more and more applications have been developed on the public sensing base. Thy could be classified into different types: environmental monitoring, transportation, health-care, location services etc. Below, we briefly give an overview of some of these applications.

Indoor Mapping System

Public sensing is found to be an effective tool for indoor mapping. For instance, Damian et. al [3] introduced a model-driven public sensing approach – MapGENIE system, which is independent on GPS signals. Usually, people use equipments for creating indoor models, like laser scanners, but that is expensive to invest. Therefor, the main idea behind MapGENIE is to use smartphones to automatically generate the indoor map with a public sensing models. The candidates use their smartphones to collect the trace while they walk around the floor, which will be used as constraints in this system, then those traces will be processed into the floor map. The authors also used a new approach to improve the quality of generated plan, it is called an indoor grammar. This grammar defines the structural information about the building.

There are several levels of Indoor mapping. The first level is the trace data acquisition. User collect their daily routine data from their smartphone, then those data will sent to second level called *trace-based modeling* processes, which is aimed to improve the quality of the trace

but still might be inaccurate. The final level is called *Grammar-based Modeling*. This level will describe the rules, for instance classes of rooms, room units, relative frequent of rooms and neighboring relationship of rooms.

Real-time Air Quality Monitoring

Recently, public sensing systems have been also used for the real-time air quality monitoring. In the recent years, the air pollution become a arising problem for human being's health. Usually, the current pollution measurement systems are not fined-grained and expensive to invest. They are fixed build in some location or use mobile equipment. Using the mobile public sensing technology offers reliable real-time monitoring and reduces the cost of measurement.

This opposed vehicular-based mobile system includes two approaches. The first one is deployed on public transportation, while the other one is adopted on the personal sensing devices. For the public model a Mobile Sensing Box was developed using a micro-controller board with air pollution sensing to collection the carbon monoxide data. Using the GPS receiver and cellular modem the position is acquired and collectedly sent to the server. The person sensing devices model is working in a similar way. Different people have installed a sensor in their car which is transferring environmental information via Bluetooth to their smartphones from where the information is sent to a server. At the server side, those collected information will be processed and aggregated together and visualized as a pollution map. The users can check these maps either through mobile phone applications or through a computer browsers.[4]

DietSense

The public sensing system is also used for health-care or health-related behaviors monitoring. DietSense [5] is an application for people who want lose weight. Firstly the smartphone of the participant will automatically take photos during the day. The images include information about the participant's food selection, estimation of food weight, the GPS data and time stamps from the GPS receiver, recorded audio samples and so on will be used for clustering and to create UI components for the participant. For example: Tom is eating a pizza in a restaurant at noon. Those uploaded information are saved at the server side like a personal repository, and could be shared with the users doctors.

1.4 QoS in Public Sensing

Quality of Service (QoS), in general, describes the quality of a communication service from the users point of perspective. In general it is necessary to define a metric if something should be objectively evaluated. For QoS in public sensing this metric has be be defined depending on the use case. For instance, the authors in [6] defined the quality as a values calculated by the fraction of queries in which the QoS constraints are met compared to all queries done. These constraints can vary dependent on the use case. For a temperature measurement project

a maximum temperature deviation could be defined. For measuring the average speed of cars on a road a maximum deviation could be defined to filter out for instance pedestrians walking next to the street. Below, we discuss some of the most important QoS matrices in PS applications such as privacy and energy consumption.

1.4.1 Privacy

Privacy is an important factor for the acceptance of the user to be part of a public sensing community. If the user knows or finds out about that privacy aspects are not taken care about there is a high probability that the user will decline using the application. For instance, the authors in [7] showed possibilities to use public sensing data without allowing to track down data of a single user. Without these measures it would be possible to extract from the collected data and have the information which driver is going were at what time. It would be even possible to detect patterns and predict where the driver might want to drive at what time. If this is not known by the participant this is against the principle that everybody should be able to decide by himself which information is reviled to who. For improving and ensuring the privacy of users, PS designers should only collect the necessary data and pay much attention to the adopted data aggregation methods. This means that not the data of a single individual is used for evaluation, but aggregated data of several users. In this way it should not be possible to track down each user.

Similar work has been done at the Pennsylvania State University, the authors analyzed the existing methods and developed an new approach to handle the privacy of mobile sensing applications [8]. They have found that relying on a trusted aggregator who collects information of the smartphones and offers the collected information in an anonymous format is the optimal solution. The weak point here is to find the trusted aggregators. If it would be possible for an untrusted aggregator to collect the same information the privacy would be gone. In order to prevent this, encryption schemes are used where only the aggregator can decrypt the information. The paper mentioned above describes a way of assuring privacy even with the existence of an untrusted aggregator. This approach uses an additive homomorphic encryption and a new key management scheme based on efficient HMAC [1].

The point in here is that taking care of the privacy of the user is a quite complex topic. It is important to keep in mind the trade-off between the level of privacy, the effort on development of the system, the battery consumption of the smartphones because of encryption and the targeted level of privacy.

1.4.2 Scalability

Scalability is another important factor when designing a Multi-User infrastructure. If this factor is not taken account of there is the risk that it is necessary to re-design the whole system while the system is already up and running. Otherwise the system could get slower and slower or might not be able to cope with the number of users at all. Figure 1.2 shows

[1] Keyed-Hash Message Authentication Code

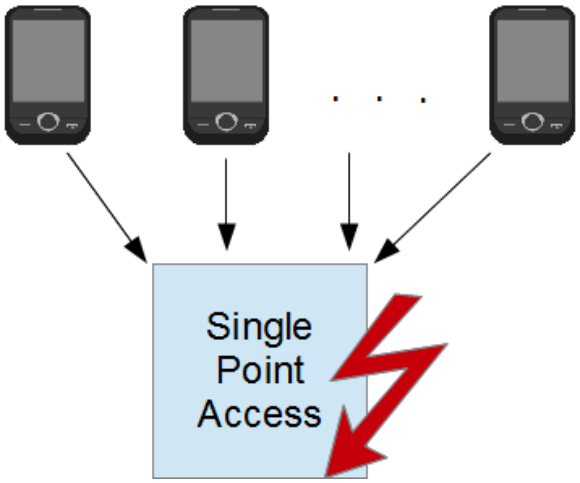

Figure 1.2: Single point of access

a server structure which is potentially not scalable. All nodes are connected to one node serving them. If the amount of user gets too high or the used bandwidth per client user rises unexpectedly this would lead to a bad system performance.

Figure 1.3 shows an approach for a scalable system using a load-balancer. The idea is the following: The work is done by multiple worker nodes. The amount of these worker nodes can be adapted to the needs, dependent on the amount and requested resources of users. A query from a client could be realized in the following way:

1. The worker nodes are continuously sending their system load to the load-balancer node. Using these information the load-balancer node is able to assign tasks to nodes with less load.
2. The smartphone is contacting the load-balancer.
3. The load-balancer is sending a link to a worker an available worker node.
4. The smartphone is using the assigned worker node to process it's request.

1.4.3 Energy Consumption

Thanks to the recent development of smartphones, people use more and more applications and functionality of them, like browsing Internet news, using navigation or playing games. The energy consumption is becoming increasingly prevalent and the energy demands play more and more an important role. Most of the available smartphones in the market use lithium-ion batteries as their energy source. A fully charged battery can keep the devices running from a couple of hours to a couple of days, dependent on the usage pattern and the apps running on the smartphone. The majority of battery consumption are typically dissipated for the display and for GSM/LTE module [9]. For example in areas which do not offer a good 3G/4G

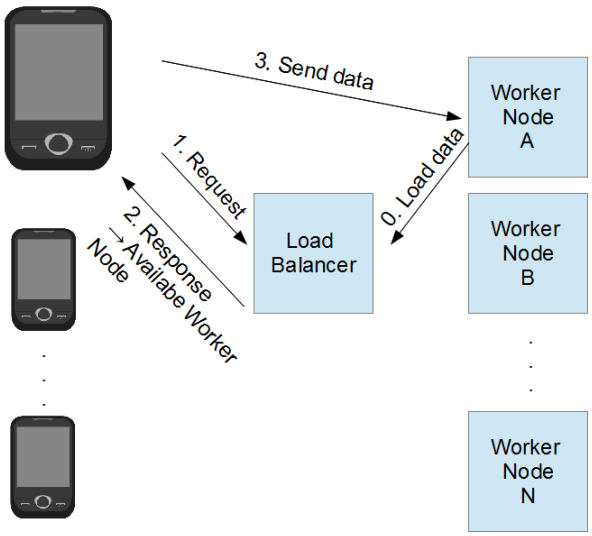

Figure 1.3: Multi node access

coverage, smartphones will always change their frequencies and search for suitable wireless channels. In this case they dissipate more energy. Additionally the selected brightness of display is a factor. In this subsection we discuss three different aspects which contribute to the energy consumption problem of smartphones when they participate in PS applications. This is the data sampling, data uploading and position measurements. Especially we target in this thesis to relieve the burden of data uploading via the utilization of compressed sensing.

Data Sampling

Recently, smartphones are heavily used in public sensing applications. It is taken advantage of having multiple available sensors to be sampled and sent to the server. One side is the smartphone sampling the data, the other side is to investigate the additional energy consumption of the smartphone needed for sampling the measurements. In this thesis we used an Android-based smartphone Samsung Note 3, which has a battery of 3200mAh.

Positioning

More and more smartphone applications are developed on the localization-base, like tracking and routing. There are three different methods for localization:

- GPS-based: The traditional location source GPS[2], it is probably built in every smartphone. For example, a real time traffic plan application requires continuous GPS data from the smartphone sent to a server. This kind of positioning is based on the client side, which can perform all necessary computations on the mobile device side. In general

[2]Global Positioning System

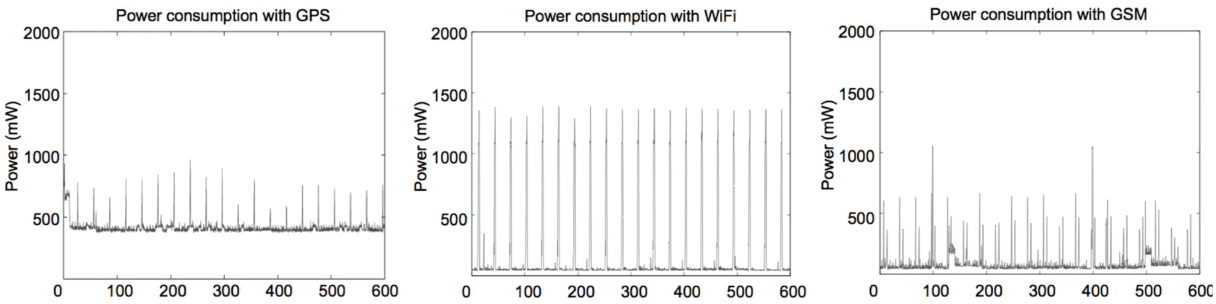

Figure 1.4: Power consumption of different location services [1]

the accuracy of GPS-based localization is about 5 to 20 meters. As figure 1.4 shows, the drawback of GPS-localization is that it uses much energy of the battery. A test showed that on average 500mW are needed for running a smartphone with active GPS which leads to 8.5 hours of run-time before the battery is empty. The other weakness of it is that it does not work in some places, for instance in buildings. This is because it is necessary to receive signals from satellites which is often not the case inside buildings.

- WiFi-based: the second method for localization is WiFi-based. In contrast to GPS-based, the WiFi-based works also for indoor positioning. The core idea behind this technique is to measure the received signal strength indication (RSSI) and the other parameters from WiFi Hotspots or wireless access point. The accuracy of this method depends on the number of positions that have been recorded in the database. Compared to the energy consumption of GPS, it take about 2.852Ws per minute, which is better.
- Cellular Network-based: Instead of WiFi using the cellular network for localization this approach is using the information about the transceiver infrastructure installed by the mobile network provider. The distance between those transceiver stations can reach between tens of meters to a few kilometers. When mobile devices users are connected through a cellular network for communication the cellphone has the information about the used station. By using this method users do not need special hardware and every smartphone can measure the signal strengths. Figure 1.4 shows the needed energy consumption. The drawback of this method is that the cellular infrastructure in some areas or within buildings can lead to bad connections, also the accuracy is as good as the other methods.

Data Uploading

By using time-driven public sensing applications, the data collected by the embedded sensors are continuously uploaded to the server. One reason why user might not accept participating is the energy consumption caused by the public sensing application. More data to be uploaded cost more time and more energy of the battery. In this thesis the different activities data are sent through WiFi to the PS server. Each activity sample consists of about 500 measurement points. We exploit the Compressed Sensing technique to sample data with reducing the

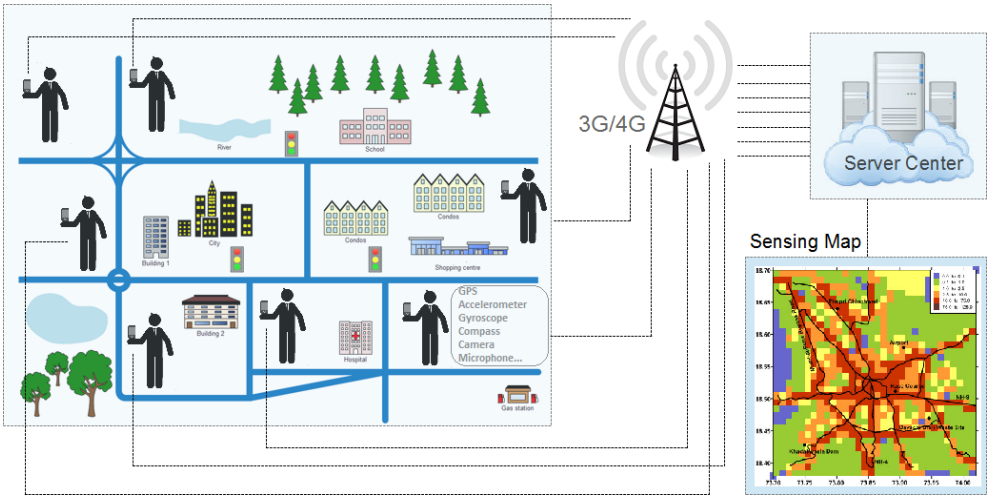

Figure 1.5: Infrastructure-based transmission

number of transmissions and comparing the accuracy of reconstructed signals to original signals. Since data uploading is the key focus on this thesis, we investigate data uploading in more detail in the next section.

1.5 Sensor Readings Uploading

Because of the low-investment and the powerful embedded sensors smartphones have been already used for applications in the health-care sector, noise level detection, traffic information system etc. There are two concepts of sensing systems: Participatory sensing and Opportunistic sensing. The participants consciously opt in to deliver data depend on the participators interest. By using this approach, the user is flexible to decide which kind of data is shared, this is ensuring the users privacy. For example there is an application like PEIR [3] from UCLA [4]. In this application people upload the information to the center server, including position, date, activities and the information of current whether. According those information the application can calculate the personal impact on the environment. Opportunistic sensing uses the opportunity to gain the data, independent on the users. In opposite to participatory sensing it is full automatically and unconscious.

1.5.1 Infrastructure-based Transmission

Infrastructure-based transmission: As figure 1.5 shows, most applications use the transmission method that the user gains and sends sensor data to a centralized server through the wireless Internet or a cellular networks. In this thesis, at the end we collect different kind of activities

[3] personalized estimates of environmental exposure
[4] University of California, Los Angeles

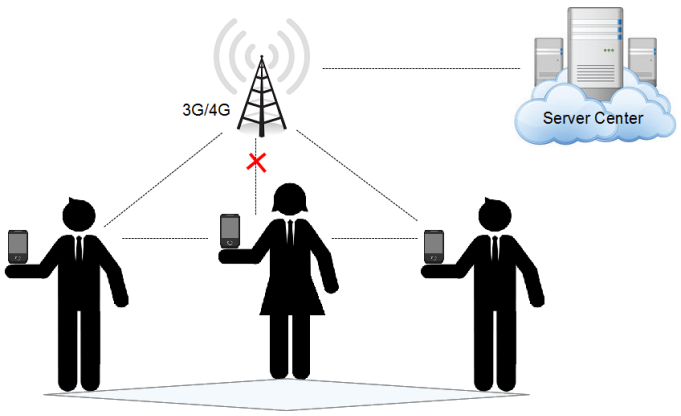

Figure 1.6: Opportunistic transmission

signals from different people for classification at the sever side, here it was suitable to use this transmission system. There are some scenarios as well that the network system might not be suitable. For example, if because of some reason the infrastructure is not work well or the sensor data is sparse, like the transfer of a weather application is getting only two information of a big distance area.

1.5.2 Opportunistic Transmission

Opportunistic transmission: the main idea of this transmission system is the application adopt between the mobile to collect or share the data through short-range radio, like WIFI or Blue-tooth. The opportunistic transmission is more extensible, because more participant are needed to expand the whole system. As figure 1.6 shows, the users connect through short-range radio. If there is a scenarios that the cellular networks do not work, the data transfer will still work between them. The Delay Tolerant Network (DTN)[10] is another example using opportunistic transmission. If the communication between the users is not reliable, then the data will firstly be sent to the server center for storage and lately delivered to the user.

1.6 Related Work

In a typical public sensing system, the limited battery life of smart mobile devices always plays an important role while designing the system. The energy management can be mainly divided into two levels: Energy conservation and energy supplying. The energy supplying consists of wireless charging and energy harvesting. The energy conservation consist of network-oriented methods, data-oriented methods and node-oriented methods. The main idea of data-oriented methods is reducing the size of data sampling. Generally, the data aggregation schemes can be divided into *event-driven methods* and *time-driven methods*. First, we introduce some of the examples of event-driven methods for solving the energy consumption problem. The first

method is called duty cycling sensing [11]. The main idea of it is, that if there is a new set of samples needed to be captured then the sensing modules are activated. This method uses dynamics of the monitored phenomenon to archive the optimization of energy management, which is time-invariant. The second method is adaptive sampling. This method improves the energy-efficiency by reducing the number of samples, in this way the size of data is also reduced.

For implementing the adaptive sensing we could use three different methods: Model-based active sensing use a computed model to reduce the data. In this model the next reading of sample will be predicted. Gedik et al. propose ASAP [12] – an adaptive sampling approach to energy efficient periodic data collection in wireless sensor network. It splits the network into clusters, and assigned the close readings to the same clusters. Mietz et al. [13] introduced another method for learning the correlation structure from past sensor data and model, which allows estimating the probability of currently outputs to sought state. The second method of implementation is adaptive sampling. According to the Nyquist theorem, for reconstruction of the original signal it needs at the least two times of the maximum frequency of the considered signal. The algorithms estimating the current maximum frequency were introduced by Alippi et al. [14]. It uses the first set of samples and detects changes, that the current maximum frequency is above or below a threshold of some consecutive samples. The third method is triggered sampling, for example used in low-power sensors for the sensing field, which do not need high accuracy. In contrast, use the energy-hungry sensors for high precision sensing fields.

Secondly, for reducing the energy cost time-driven methods can be used. The typical applications for time-driven methods are environmental monitoring, health monitoring, traffic reporting etc. Those applications upload the heavy burden sampling data regularly. Three methods have been proposed on this base: Data prediction, coding by ordering and pipeline in-network, and data compression. Data prediction implement identical predictors in the source and sink nodes to reduce the size of transmitted data [15]. The main forecasting schemes are classified into algorithmic methods, time series forecasting, and stochastic methods. In-Network Processing method is reducing the size of transmitted data through performing data aggregation, which is implemented at the intermediate nodes between the sources and the sink. The typical idea of aggregating function used in the in-network processing is application-dependent. Fasolo et al. [16] provides a comprehensive survey about in-network processing techniques. The data compression method reduces the size of data through removing data redundancy and build other more compact representation of those data. Generally, the original data could be lossless or lossy compressed. For example, the Huffman coding and ZIP format are lossless compressed. The wavelet transform and discrete cosine transform are lossy compressed. Generally, the lossy data compression can produce much smaller compressed signal than lossless method, and is still fit for some application, like mp3 or image compression [17].

In this thesis, we will introduce the method for energy consumption called Compressed sensing, which has been one of the most efficient compression technologies for increasing the speed of transmissions. It can be used in different kinds of application. Instead of obeying Nyquist-shannon sampling theorem, Candes, Romberg, Tao and Donoho developed a new theory of compression (sampling), which can keep the original information of signal by attaining the

non-adaptive linear projection. In contrast the traditional methods as "sample then compress", compressed sensing is "compress during sampling". The CS method is used for compressing data without computational burden. The main computational part is done during the signal recovery process, which could be executed on the sever side.

We could summarize some important points of these research and development efforts from two sides: The compression of one dimension data using CS and image compression with using CS. The accelerometer sensor is popularly used for sampling of the one dimension data. Those recorded signals will be used for recognizing the basic activity of humans. For example, like gesture recognition system, where the input is coming from a single three-axis accelerometer [18], and transmitted after compressed sensing process. The compressed sensing process is seen as a lightweight method compared to Nyquist. Especially for reducing the cost of transmission or storage of large volumes of data, as it was used as compression method for structural health monitoring, where the data of the patient's medical recordings might be very huge [19]. It is also tailored for the coarse sensing task of spectrum hole identification [20]. The compressed sensing approach is also used as image compression method, for example for rapid MR imaging, in which imaging speed and recovery quality both are very important.[21]

1.7 Research Objectives

In this thesis, the main goal of works is to improve the application-relevant quality of service metrics, such as energy efficiency, accuracy of signal recovery and activities signals to recognition. We target time-driven public sensing applications in which the measurements has to be frequently reported to a server. Hence, it is mandatory to efficiently convey the measurements between the source mobile devices and the server. The main idea is to compress the data at the mobile device side before to transmission. Those activities in our work consist of walking, jumping, holding by different people. This data is gained from the accelerometers sensor of a Android smartphone. In the Compressed Sensing phase using different of measurement matrices like Toeplitz, Gaussian, and random measurement matrices. Those measurement matrices are manipulated with different of compression radios. The sampling will saved on the smartphone internal memory, also through wireless sent to the server.

For data recovery, there exist different algorithms available to reconstruct the activities signal, like L1-magic, CVX etc. In this thesis will use dynamic time warping Algorithm tor activities recognition. The goals of the whole work is the evaluation of the values between compressed sensing with traditional compression about their time delay, error analyse, accuracy of recognition and also energy consumption of smartphone.

1.8 Thesis Structure

This thesis consists of six chapters. The first chapter is an introduction of the underlying research problem. This section provides the motivation for our solution, and answer the question of why we did use compressed sensing for data acquisition. Then we introduce some

necessary basic theory for support of this work, like the public sensing system, data uploading system and some another points. The second chapter will describe the system overview that is abstracted from the system requirements. It includes the Android system on the smart mobile phones and the server side to decompress and analyze the signals. In this chapter will also explained the method, that we used as transitional compression technique to compare with our work. The activities recognition system will be introduced, it will be choose for later recognition section work.

The one part main theory of our thesis is compressed sensing, that we used in this chapter is explained in the 3. chapter. The preliminaries, proposed methods and related work of Compressed Sensing will be introduced. The implementation of this compressed application is explained in chapter 4. The application system on android is presented first, contains the concrete implementation of both system. In the 5. chapter, there will reported the evaluation of our system from different aspects, like using different measurement metric, the transitional compression method compare with compressed sensing method about energy consumption for smartphone.

The core chapters of this work. First we get the original signal from the smartphone accelerometer sensor, and the compression process are based on the Android operating system in a smartphone. On the other side the server will be explained with the aspects of the decompression process and ways of analyzing data. These processes have been implemented in Matlab on the server side. In chapter Implementation the whole system will be explained. In the chapter Evaluation, we compare Compressed Sensing to compress the data and using traditional ZIP, DCT[5] compression and getting measurements analyzed regarding energy consumption and compression time. Secondly the effectiveness of different matrices of Compression Sensing with different compression ratios are compared to determine which one is the most effective one. The whole system will use different activities as input signals and analyse the accuracy of recognition. At the end of this thesis we will summarize the whole work and find some useful points for future work.

[5]Discrete Cosine Transformation

Chapter 2

System Overview

This chapter focuses on extracting requirements and presenting the system model. To fulfill all the requirements a compression processing is established. This chapter will also introduce some signal processing knowledge basics and other compression methods that are used as reference methods for comparison with the CS compression methods used in this work.

2.1 System Model

As figure 2.1 shows that the thesis consist of a smartphone and a server part. On the smartphone side firstly will read the signal from the smartphone accelerometer sensors. Different kinds of activities are used as input signal, like walking, jumping and holding. The sampling process is based on the basics of the compressed sensing theory, which is a sate-of-the-art data compression. In this process different kinds of measurement matrices will be tested. On the server side we will use the reconstruction algorithm to recover the compressed signals. At the end of the work the different activities should be recognized.

2.2 System Requirements

In this thesis we will work on the basis of public sensing, which is gathering sensor data from a crowd of mobile devices. The main idea of Public Sensing has been employed in various applications, like indoor-navigation, air pollution detection and measuring the noise level in urban areas. It is an efficient way to reduce the cost alternatively for investing in a large sensor network infrastructure. This is done by using Compressed Sensing for compressing the

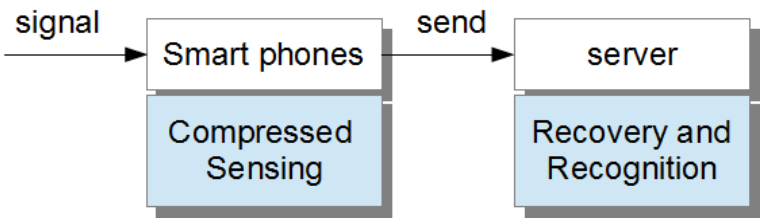

Figure 2.1: System model

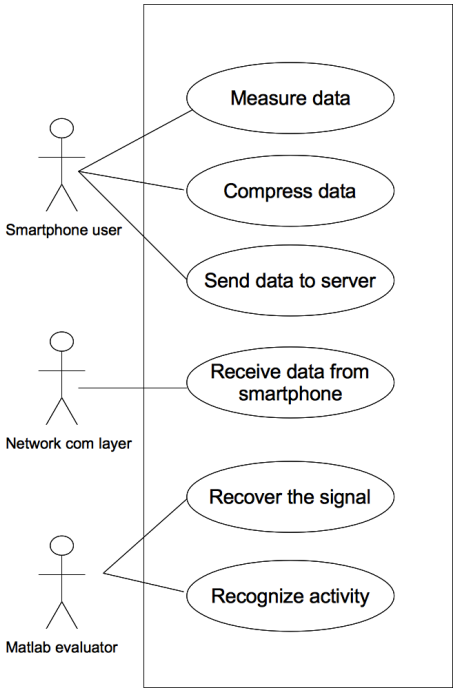

Figure 2.2: Usecase diagram

data coming from the smartphone sensors. The volume of data gathered by sampling will be reduced, also it is a better solution facing the problem of energy consumption. This approach is especially suitable for big data processing to reduce the delay time.

2.3 Application Scenarios

Figure 2.2 illustrates the actors which are building the system. The diagram consists of three actors: The smartphone user who is collecting data, the network communication layer which is responsible for receiving data from smartphones and the Matlab evaluator who is responsible of analyzing the received data. The diagram consists of the following tasks:

- Measure data: The smartphone user or respectively the app running on the smartphone should measure the information coming from the acceleration sensor and store the information locally so that they can be processed later on.
- Compress data: The smartphone user or respectively the app running on the smartphone should compress the previously measured data in an appropriate way so that only little computation is necessary and the compression ratio is still high. For this thesis the smart sensing compression should be done based on multiplications with sparse matrices.
- Send data uploading: The compressed data needs to be transferred to the node running Matlab for evaluation. In order to make this process comfortable and user friendly this

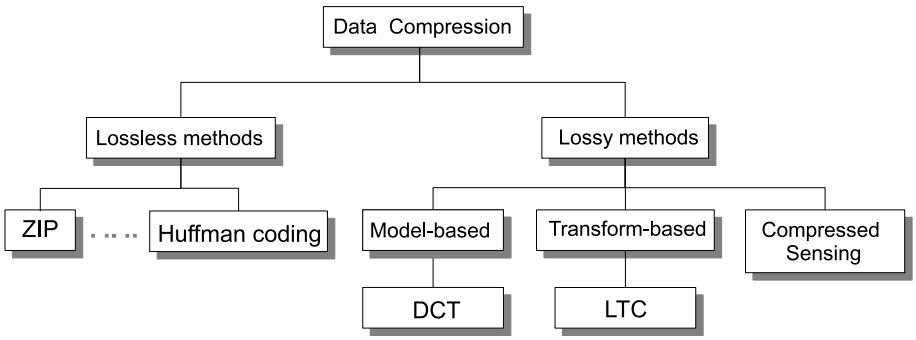

Figure 2.3: Compression methods

should be done wirelessly. After applying compressed sensing on the data collected from the accelerometers sensors the signal should be transferred to the server. This is done through wireless communication technology, which is using an FTP server running a local test machine where also the Matlab environment is located.

- Receive data from smartphone: The network communication layer is responsible of receiving the data sent by the smartphone users. Since sending data and receiving data have to be compatible to each other a common communication protocol has to be chosen here.
- Recover the signal: The Matlab evaluator has to fetch the data received by the communication layer and recover the movement data of the user. In order accomplish this task an appropriate algorithm to decompress the sparse data should be used.
- Recognize activity: The main target of this thesis should be to recognize the movements done by user. The Matlab evaluator should use the previously decompressed data and recognize the movement. This should be done by comparing the decompressed signal and samples out of a movement database by using the time warping approach.

2.4 Data Compression Methods

In this thesis, we collect the activity signals utilizing the smartphone accelerometer sensor. Using this approach results in small investment cost and it can be commonly used in many areas like gait analysis, fall detection, etc. Before we upload the activity signals to the server side for recognition the signal are compressed, because the uploading of the original signals would take more time often accompanied by take more power on the smartphone side. The main idea used for compression (sampling) is Compressed Sensing. It has been gaining great attention in the area of signal sensing and can potentially reduce the amount of data in a very efficient way. The detailed theory of CS will be introduced in the next chapter.

In this section, we introduce some traditional compression methods for comparing their results with the results gained by the CS methods used in this work. Figure 2.3 shows that data compression can be generally divided into lossless and lossy methods. The lossless methods

allow the original data to be perfectly reconstructed but with the drawback of relatively small compression ratios, for example like the ZIP format, Huffman coding, etc. Lossy methods are used to reduce the data size achieving higher compression ratios but the compressed data is only an approximation of the original data. It is well suited for application like environmental monitoring and data logging which tolerate a limited precision.

2.4.1 Lossy Data Compression

Using smartphones as a sensor cluster has the limitation that the devices are often equipped with powerful processors and large quantities of memory but they are limited at their battery life. This is why it is important to use the CPU only as little as necessary. Lossy compression methods offer the advantage that they potentially need lower resources for compression compared to lossless methods. This is because they can skip the details of the original data up to some point. Lossy compression methods can be classified into: transform-based techniques, model-based techniques and compressed sensing.

- Transform-based techniques are typically linear transformations that map data on a space, such as the discrete cosine transform. The discrete cosine transformation (DCT) was introduced first in the early seventies by Ahmed, Natarajan and Rao [22]. DCT is a member of the class of sinusoidal unitary transforms [23]. Generally people use it for transforming a signal from a spatial representation into frequency representation. DCT has been proposed in four variants: DCT-I, DCT-II, DCT-III and DCT-IV. Among of the variants DCT-II is the most commonly used form. The DCT method is suitable to be used for one dimension signal compression like the signal from an accelerometer sensor, or a speech signal from a microphone. It is also widely used for two dimension signal data, like image data.

1D Discrete Cosine Transform

In this thesis we use the DCT method for comparison with the Compressed Sensing method. For a one dimensional signal from an accelerometer sensor we could use the following form: For a data sequence C(n), n=0,1,...,(n-1) is defined as:

$$C(0) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x(n) \tag{2.1}$$

$$C(k) = \sqrt{\frac{2}{N}} \sum_{n=0}^{N-1} cos \frac{(2n+1)k\pi}{2N}, k = 1, 2, ..., (n-1) \tag{2.2}$$

In the formula 2.1 and 2.2 C(k) is represented as the k-th coefficient. If the sampling frequency is normalized to 1, C(k) is a bandpass filter with a center frequency at (2K+1)/2N. If the DCT is concentrated in a low frequency then the remaining DCT coefficients are only having a small significance to the signal.[24]

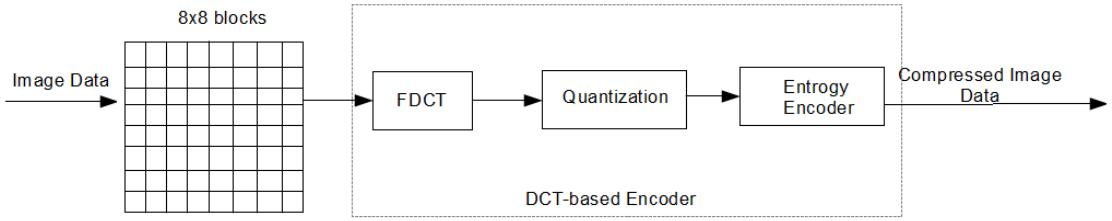

Figure 2.4: DCT image compression

2D Discrete Cosine Transform

The actual formula 2.3 and 2.4 are used for two dimensional discrete cosine transform. In this DCT equation for computing by giving an image P in the spatial domain the pixel at coordinates (x,y) is denoted as P(x,y). To transform P into an image in the frequency domain D. N is the size of partitioned image block, for example an 8x8 block with n is 8.

$$C(i)/C(j) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } u = 0 \\ \\ 1 & \text{if } u > 0 \end{cases} \tag{2.3}$$

$$D(i,j) = \frac{1}{\sqrt{2N}} C(i)C(j) \sum_{x=1}^{n-1} \sum_{y=1}^{n-1} p(x,y) cos \left[\frac{(2x+1)i\pi}{2N} \right] cos \left[\frac{(2y+1)j\pi}{2N} \right] \tag{2.4}$$

In an image the most of energy will be concentrated in the lower frequencies. The DCT compression method removes the higher frequency coefficients by the compression algorithm so that the data of image is reduced without sacrificing too much image quality.

For example figure 2.4 shows a JPEG compressor which is using DCT-based encoding in several steps: Firstly the image data source will decomposed into a 8x8 square matrix of pixels. Then each block is ran through an 8x8 2D DCT (FDCT). In the step of quantization the resulting unimportant coefficients are thrown away to reduce the size of the file. For example the JPEG picture compression algorithm is dividing the coefficients by a quantization matrix with many number of zeros. After performing this quantization a lossless method is used to compress these quantised coefficients for each of those partitioned images. In case of JPEG this loss less method can be Huffman coding or arithmetic coding.

- Model-based techniques use a simple mathematical model to approximate the samples, for example the lightweight temporal compression (LTC) [25]. It assumes that there is a small error at each reading. Dependent on the approximating line deviates from the error margin by adding the next data point a new approximation will be started. The derivative-based prediction method (DBP) is similarly built to the LTC method.

It is based on using a simple model that can capture the main data trends with linearly approximates. The difference between DBP and LTC is that the LTC method transmits the parameters of a line once it has finish its approximating and DBP sends out the line parameters immediately after the learning phase until the model in "incorrectness" state.

- Compressive sampling (CS) is the compression algorithm mainly used for this thesis. It is suited very well for compression of accelerometer data, which can be acquired and compressed simultaneous. During the process of compression the most insignificant data gets discarded. The specific description of this method will be explained in the chapter compressed sensing.

2.4.2 Lossless Data Compression

In this thesis we chose using zip compression for accelerometer data compression for comparing the performance of compressed sensing methods as figure 2.5 shows. The famous loss-less data compression is storing files in the ZIP format. It was created in 1989 by Phil Katz. The zip compression is using the end of the central directory record to be identified. All different variations of the ZIP format are built on the base DEFLATE algorithm, which is combination of the loss-less compression algorithm Lempel–Ziv–Storer–Szymanski (LZSS) and Huffman Coding.

- Lempel–Ziv–Storer–Szymanski (LZSS): was created by James Storer and Thomas Szymanski. LZSS is a variant of LZ77, using dictionary encoding technique. If a string is seen a second time and because of that an entry in the dictionary is found, it is replaced by the reference of it in the dictionary.

  There are three point where LZSS has been improved over the LZ77 algorithm. First it uses a circular queue for the look-ahead buffer; second the search buffer (the dictionary) is built as a binary search tree structure; thirdly instead of create token with three fields use only two field, which includ an offset and length, if find no match one then release the uncompressed code of the next symbol.[26]

- Huffman Coding: the main idea is using as variable-length code table as output, in which a lower number of bits to encode the data that occurs more frequently, and less common symbols are represented using higher number of bits.

2.5 Acceleration-Based Activity Recognition

In this thesis, we reconstruct the compressed signals and try to recognize the different kind of activities in Matlab. The research area of human activity recognition has been developed for many applications, such as the health-care and life-care branch.

In the past years people have been using computer vision-based techniques for activities recognition. It works well in the laboratory environment, but it is complex and not suitable

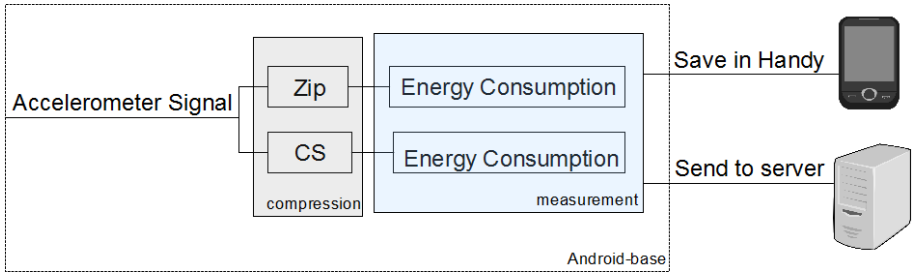

Figure 2.5: Use zip for measurement of energy consumption

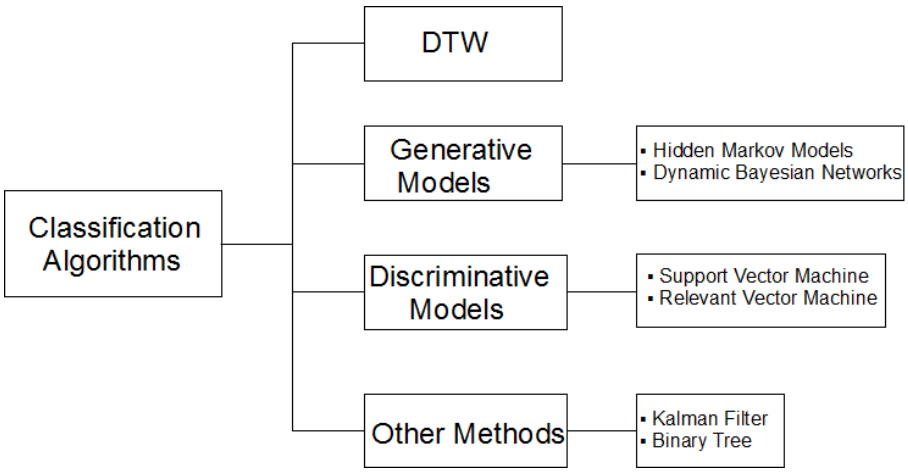

Figure 2.6: Classification algorithms

for daily life recording. In fact there have been already some works for recognition of human activities done by using smartphones. This is because of the advantages that the devices are sensor-rich, Internet enabled and available with a big number of users.

Figure 2.6 shows there there are different classification algorithms for activity recognition available: Dynamic time warping (DTW) is the most common temporal classification algorithms because it offers activity recognition in a simple way. The details of this method will be introduced in a later section. The generative models are built on the dynamic classifiers base such as the Hidden Markov Models(HMM), Dynamic Byerian Netzworks(DBN) etc. The Discriminative Models are learning a model with joint probability such as Support Vector Machine(SVM), Relevant Vector Maschine(RVM) and etc. The other methods are Kalman filter, Binary tree etc.

2.5.1 Feature Selection Method

In this section, we introduce two classification methods using the smartphone's accelerometer for recognition. One of them utilizes feature selection, which defines the feature after the analyse of collected signals. Another is using Dynamic Time Warping (DTW) which is used

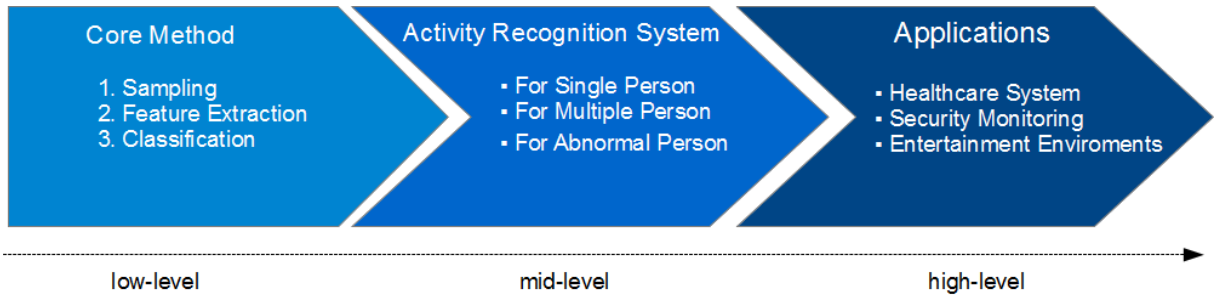

Figure 2.7: Activity recognition levels

for matching two sequences with a different temporal. In this thesis we decided to use the DTW method for activity recognition. The signal wave form of one activity and different participates might have different amplitudes but with a similar shapes. The DTW method is used with a set of template patterns to match the activity shape.

Figure 2.7 shows the three levels of activity recognition. The first level is low level which runs the core method of activity recognition. The second level is the mid-level. In this level the activity recognition system is developed tailored for different kind of participants. The highest level is the application level which is utilized for health care systems, security monitoring or entertainment environments.

We focus on the low level of activity recognition, which consist of three steps: The sampling step, the feature extraction step and the classification step. Firstly, some activity signals from different participate will be collected. Those physical activities like standing, sitting, jumping, walking, walking downstairs and upstairs are chosen because of they are typical motions in our daily life and the same activity occurs often periodically. This should make the recognition easier. Because the smartphone will deliver different raw signals dependent on the location and damping of the smartphone during the measurement, always the same location for measuring was chosen. This could be holding the cellphone in the hand or put in a pocket. The acceleration sensor delivers three acceleration vectors, one for the x-axis, one for the y-axis and one for the z-axis. Depended on the orientation of the smartphone the z-axis for example records the information about when people move forward, y about upward and downward, and x about horizontal. So when people are walking or sitting there must be different signal waves coming from the accelerometer. To increase the accuracy each participate performed the activities twice or even more often.

Secondly, because the accelerometer data is in a raw time-series format it cannot be directly used by a standard classification algorithms. That means a representation of the data as terms of feature vectors are required. The accelerometer raw data is transformed into some examples, which are achieved by dividing the raw data into fixed time segments, like 10 or 15 seconds. Each of those examples include some features which are important for the classification step. The generated features are based on the main feature types from the axis information. For example calculate each axis of accelerometer data the average values, standard deviation of each axis, average absolute difference, average resultant acceleration, time between the peaks in the sinusoidal wave, and maximum (minimum) of values for each axis, etc.

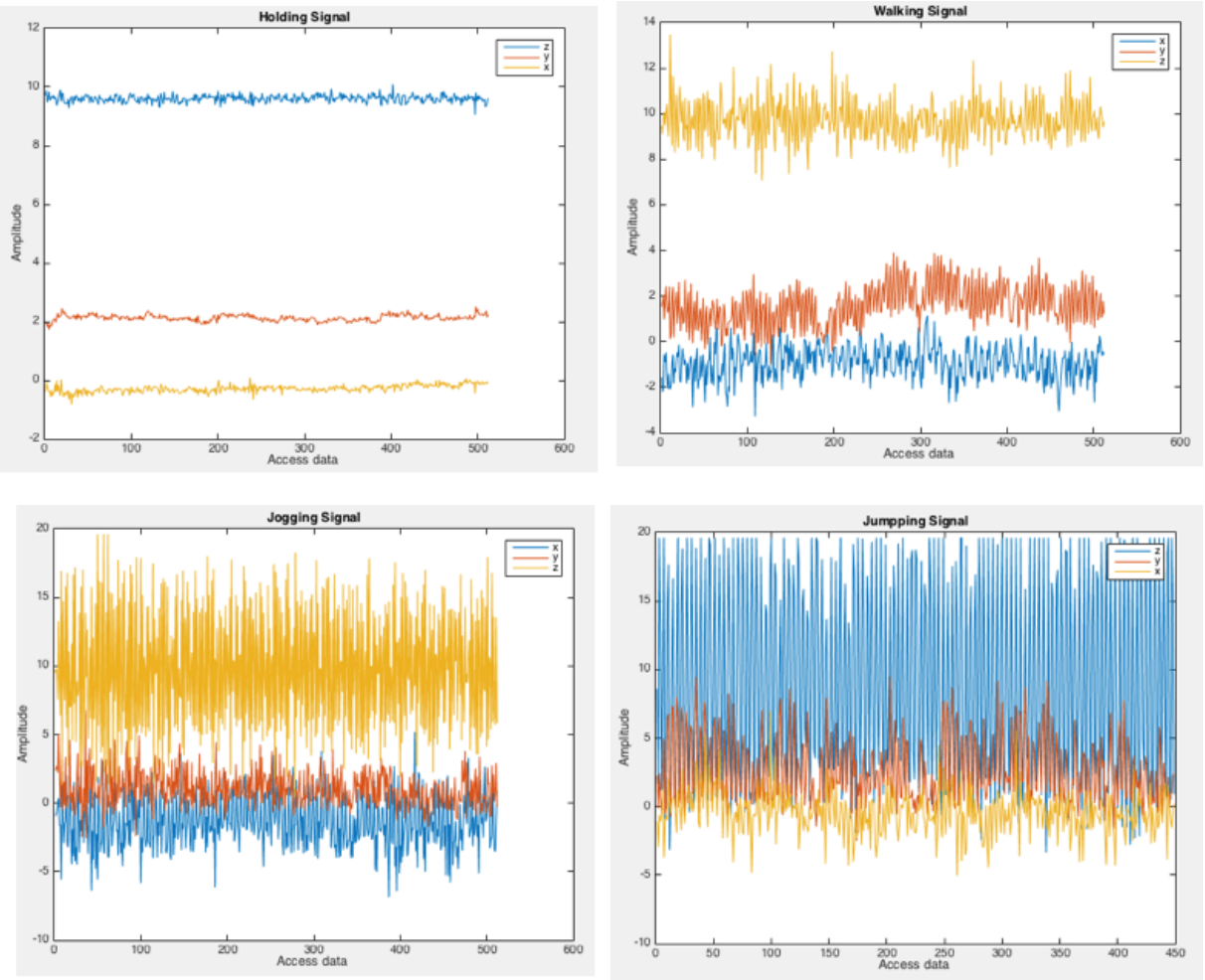

Figure 2.8: The activities signal (three axes) example

As figure 2.8 shows this data represents the activity signals like holding, jumping, walking and jogging, which were measured by the smartphone accelerometer. Each activity includes the information from three axes. For example the plot of holding seems to be constant and not period. This is different to walking, jogging or jumping where you can see a periodic behavior. The plot shows, that the most of the activities have a relative big y-value, because of it represents the gravity of earth in the accelerometer. Therefore the signal is different between sitting and standing. This information can be used to describe the features. The signals of walking, jumping and jogging are similar to each other, but it can be still differentiated between the patterns.

Thirdly, those examples will be used for the training and testing processing. The decision tree classification algorithms is one of the algorithms for classification. The main idea of this algorithm is using to answer a series of crafted questions about the some features of the test record. Every time an answer is receive, a follow-up question is asked until the class label is reached. The data structure of this method is suitable for using a decision tree, which has a

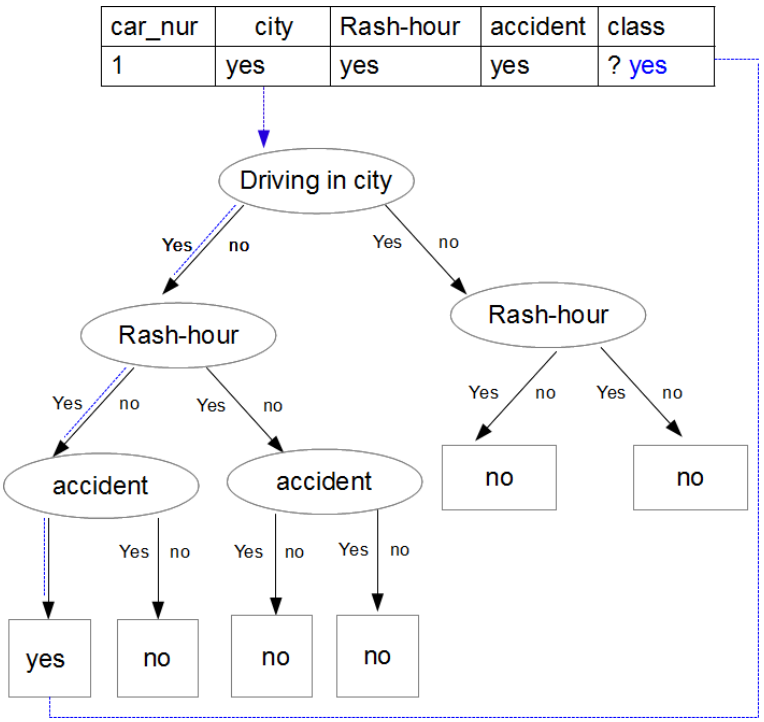

Figure 2.9: The decision tree classification example

hierarchical structure.It represents using nodes and edges. For example, as figure 2.9 shows, that we want to check whether today is might be a traffic jam. The process is starting from the root node according to the received answer until reaching the class label.

The most popular tool is `WEKA` data mining suite, it contains tools for data pre-processing, classification, regression etc, and it is also suitable for developing new machine learning schemes.[27] After using of classification algorithm, it is possible to recognize the actives. Below, we explain a novel method for detecting the activities, referred to as the *Dynamic time Warping*.

2.5.2 Dynamic Time Warping

Probably all smartphones have a tri-axial accelerometer sensor, which can measure the information of three spatial dimensions (x,y,z). It can be furthermore used for estimating velocities and displacements. In this thesis we use another method of activity recognition. This is the Dynamic Time Warping (DTW) method. It is widely used for matching two sequences. The Euclidean Distance (ED) is a frequently used algorithm for absolute distance calculation between two time series x_i and y_i. As figure 2.10 shows the ED method aligns the points in x axis on one time series with the points in y axis. Using this method two different temporal sequences can be very sensitively compare with each other.In this thesis we collect a walking signal from two people with different temporal.

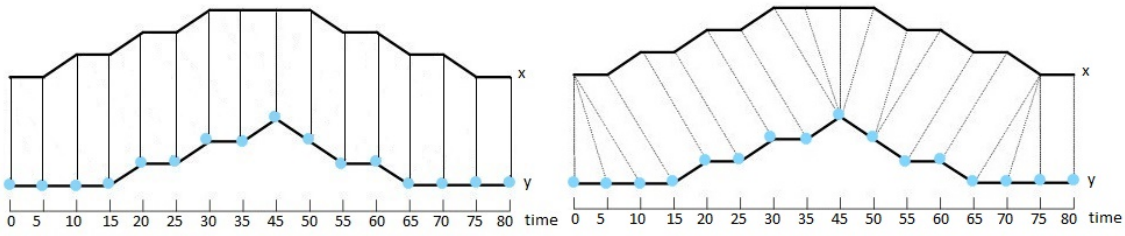

Figure 2.10: Comparison between (a) the Euclidean distance and (b) the DTW distance

The Dynamic Time Warping (DTW) algorithm is a famous mechanism to find the optimal alignment with different temporal scales of the two similar utterances. The main idea of this mechanism is find out the corresponding area or the similarity between two signals. In the recent years DTW was used for the speech recognition, where it was used for comparing two speech patterns with different speech temporal. In fact the DTW mechanism could be used for any signal that could be presented in a linear sequence, like video signals or graphics. In this thesis we try to use this algorithm for recognizing human activities. As part B of Figure 2.10 shows, DTW can be used in the warped non-linearly time path, which is different to the Euclidean Distance method.

Figure 2.11 shows that there are two time series, a sequence X with the length m and a sequence Y with the length n: $X = x_1, x_2...x_m$; $Y = y_1, y_2...y_n$. If m = n, that means those two time series have the same length. In this case measurement is easy to use the Euclidean distance method, which is an efficient metric to measure between two similarly signal with same time length. Using this method, that the each point in one time series will be directly compared to the other point in the other time series. This measurement metric is not suitable in case if only the time series is stretched or shrunk.

At first an m-by-n matrix is to be built which is filled with some elements. Those elements present the squared distance values of two points in of two time sequences. Those elements are defined as: $d(x_i, y_j) = (x_i - y_i)^2$. As figure 2.11 shows there is a path for finding out the minimizes warping cost in the matrix. If those signals are identical then the warping path will be growing as a straight diagonal line. Otherwise the warping path is going along as a non-diagonal path. That means that the following cell is increased along either by the i or the j-axis. The goal is the top right cell, then the minimal warping costs are achieved.

The warp path has tree conditions:

- Monotony, the path with index i and j never decrease.
- Continuity, the path with index i and j increase step by step.
- Boundary, the path along the bottom left of matrix and ends at the top right.

The global warp cost of two signal sequences is defined as: $\frac{1}{k}\sum_{k=1}^{k} w_i$ where w_i are elements that belong to the warping path. The basic principle of dynamic time warping uses dynamic programming algorithm, defined as:

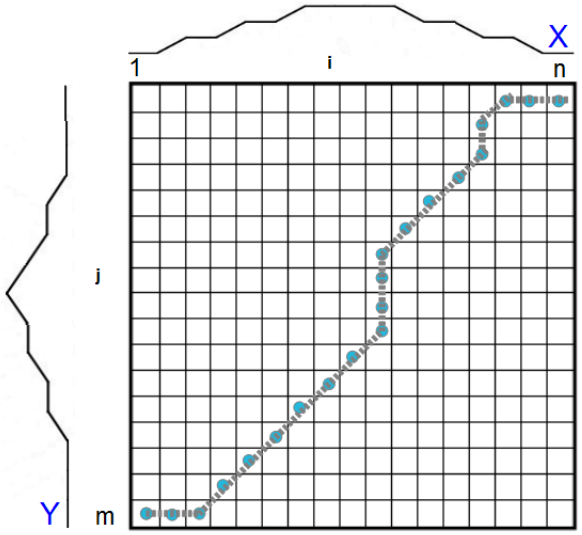

Figure 2.11: DWT warping path

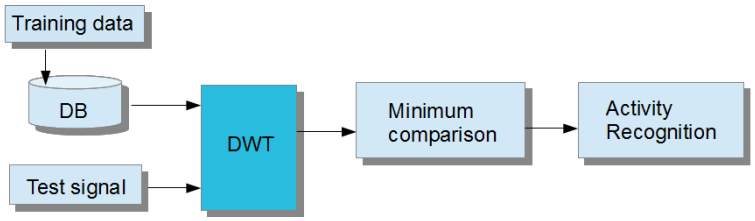

Figure 2.12: DWT-based activity recognition process

$$\gamma(i,j) = d(x_i, q_i) + min\gamma((i-1, j-1), \gamma(i-1, j), \gamma(i, j-1))$$

In this form, we use cumulative distance $\gamma(i,j)$ instead of the current distance value $d(i,j)$.

As figure 2.10 shows we utilize the DWT algorithm for the activity recognition process. It consist of several steps. At first the input data consist of the test data and training data, which is stored in the reference database. This training data is recorded from one person with different activities like holding, walking, jumping, jogging, going stairs up and going stairs down. The test signal is the unknown signal that needs to be later recognized. For each test signal the DTW method will be called to be compared with the six activities that are stored in the reference database. Using DWT we get the six optimal warping paths and then can find the optimal minimum comparison values. Those value could identify the kind of activity.

2.6 Summary

In this chapter the requirements of the proposed system are highlighted from several aspects. This system consist of three components: Data collection with the Android smartphone accelerometer sensor. The server side using Matlab for recovering the sampling data and doing activity recognition using the dynamic time warping algorithm. In order to prepare the evaluation with other compression method we also described the ZIP format and the discrete cosine transform method.

Chapter 3

Compressed Sensing

In this chapter we will focus on the details of the compression method that we used for this thesis called Compressed Sensing. We will first introduce the traditional sampling method - the Nyquist-Shannon sampling. Then we will explain some different sensing matrices like the random toeplitz matrix, the random gaussian matrix etc. At the end the specific parts of the activity signal compression process will be interpreted.

3.1 Preliminaries

In this thesis we are receiving signals from a smartphone accelerometer sensor as input data, then send them to a server. Normally the transmission of signals from the smartphone to a server takes a lot of time accompanied by a high energy consumption. This is one reason why users might refuse to use the public sensing system. We want to find a method to improve this situation by solving this problem. A new theory of compression is called Compressed Sensing or Compressive Sampling. We will use it for data acquisition and reconstruction.

3.1.1 Nyquist-Shannon Sampling

The transitional Nyquist-Shannon sampling theorem says: If the sampling frequency is at least twice of the maximum frequency contained in the signal then the band-limited continuous data can be exactly reconstructed. This theorem is used for nearly all signal acquisition protocols like audio, image etc.

Or in mathematical terms: $f_s \geq 2f_c$

In this term f_s is the sampling frequency. It defines how often samples are measured. f_c defines the highest frequency contained in the signal.

On the basis of the Shannon theory the signal processing is replaced by digital processing. The great advantage of digital processing is that the signals can be captured more easily and building a digital signal process system is more flexible and cheaper. Because of this more and more data is measured and has to be transferred. This is leading as well to a high network load. It could also be difficult for some kind of sensor to deliver such high sampling rates required by the Shannon theorem.

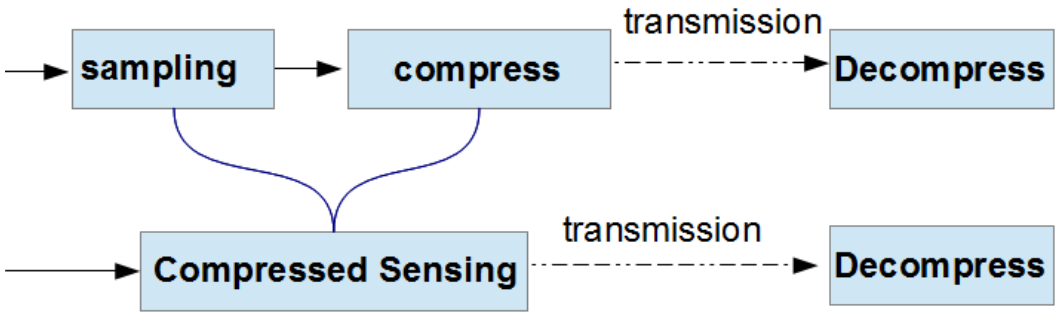

Figure 3.1: Traditional acquisition vs. compressed sensing

Compressed Sensing is a new method to process mass sampling data taking less time for compression and uploading of data. In this thesis we used this method to evaluate the energy consumption of the smartphone for the transmission of activity signals. The main idea of this method is to sample first with a normal rate, then do compression and filter out the unimportant samples. In this way the signals still contain enough information to be reconstructable on the server side after transformation. The principle of Compressed Sensing is that the most of the natural signals can be represented in a sparsely way or can be possibly changed to a suitable basis so that the sampling signals are vastly reduced.

As figure 3.1 shows we compare the data compression process using traditional acquisition and Compressed Sensing. Before transmission of compressed data, data sampling and compression will be integrated in one step by the Compressed Sensing method. The Compressed Sensing approach grew out of the the work of Candes, Romberg, and Tao and of Donoho in 2004[28]. They showed a finite-dimensional signal having a sparse or compressible representation can be recovered from a small set of linear, non adaptive measurements. The compressed sensing approach has been broadly used for signal processing within medical imaging, electrical engineering, computer science and other branches.

3.1.2 Sparse Representation

The signal x is sparse, that means x is well approximated by a linear combination of a small set of vectors form a basis or dictionary. We could understand "sparse signal" in the mathematical sense, a sparse collection of data has a small number of non-zero values. In simple words, most values are 0 and only a few contain meaningful data. Therefore the sparse signal can be appositely compressed without losing important information for reconstruction and reduce the amount of needed data transmission. At compressed sensing we suppose signal x is sparse: $X = \sum_{i=1}^{m} \theta_{n_i} \psi_{n_i}$, where M«N. ψ is the basis or dictionary, and $\psi = \psi_1, \psi_2$.... That means the signal x is M-sparse in ψ and ψ is the sparse basis. The most of signals in natural or image signals might not be sparse. They will be represented as an approximation. This means we choose a suitable basis, in this case those signals will be called compressible. We could use wavelet transformation or fourier transformation the dictionary. [28].

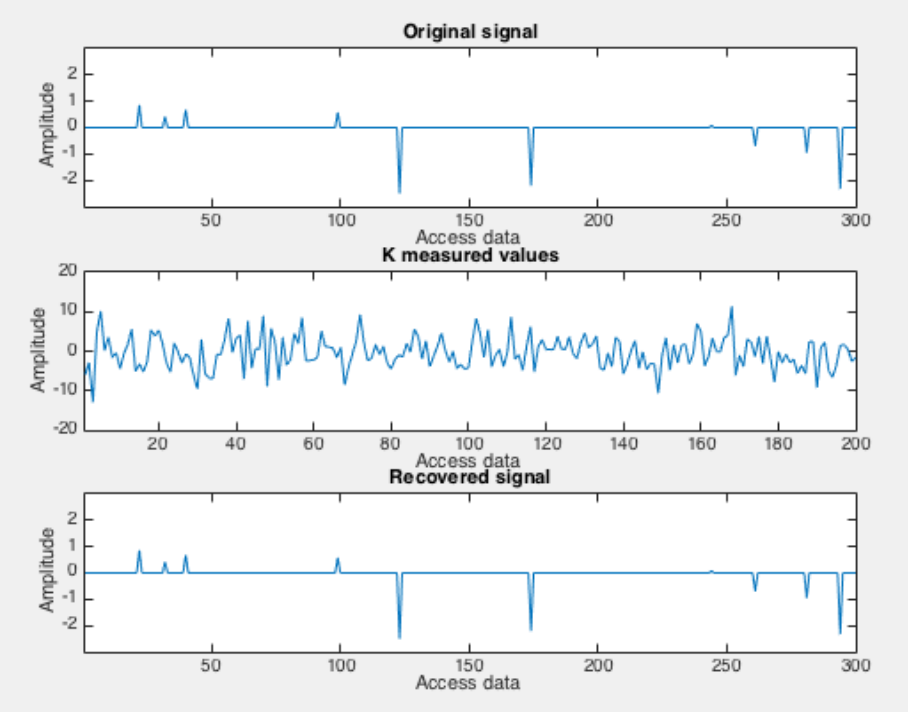

Figure 3.2: A sparse signal using compressed sensing

As figure 3.2 shows we simulate in Matlab a sparse time-domain signal with the length of 300 measurement points and nine non-zero peaks. The peaks and the amplitude are produced by a random value generator. We use a popular method, the l1-MAGIC algorithm to reconstruct the signal. As the last part of the figure shows 3.2, it perfectly reconstructed even through only 60 random measurements were evaluated. This simple example also demonstrates that the sparsity of a signal can be exploited to recover from far fewer samples.

3.1.3 Sensing Matrices

In this thesis we chose different sensing matrices in Compressed Sensing and tried to find the most effective one for our work. The goal of designing the sensing matrix Φ is to find some matrix which does not destroy any important information in the original signal by the multiplication process. In this section we will first introduce those matrices include sparse random matrices, random toeplitz matrices, and gaussian random matrices with their mathematical definition and desirable features of them.

- sparse random matrices: In our work, we chose one of the sensing matrices is a sparse random matrix introduced by Berinde et.al [29]. This matrix is generated for each column with d random values containing '1'. The value of m represents the number of measurements and must be much lower than n. Compared to the gaussian matrices

there is a high probability to have a similar effect; reducing the encoding and decoding time.

- random toeplitz matrices: A is a toeplitz matrix. If the i and j elements of A are denoted $A_{i,j}$, then we have $A_{i,j} = A_{i+1,j+1} = a_{i-j}$.

$$A = \begin{pmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{-(n-1)} \\ a_1 & a_0 & a_{-1} & \ddots & & \vdots \\ a_2 & a_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_{-1} & a_{-2} \\ \vdots & & \ddots & a_1 & a_0 & a_{-1} \\ a_{n-1} & \cdots & \cdots & a_2 & a_1 & a_0 \end{pmatrix} \tag{3.1}$$

- gaussian random matrices: There is defined a MxN matrix Φ, which is filled with random entries from a gaussian distribution of mean 0 and variance 1/M, or in the mathematical definition: $\Phi_{i,j} \sim N(0, \frac{1}{M})$

3.1.4 Reconstruction Algorithms

After the data acquisition by which the compresses sensing is achieved, the signal will be uploaded to the server. The reconstruction of the original signal can be seen as recovery of the sparse set of the coefficient sequence. This problem could be solved by following kinds of reconstruction Algorithms:

- Convex Relaxation, it solves a convex optimization problem through linear programming. The typical example is Basis Pursuit. If X' is the sparsely representation for signal x, x' = ψs'. This representation contains sorted coefficients of s, the value of the first K coefficients are preserved, and other is zero. When s' is close to s then x can be closely approximated by x'. Because of $M << N$, to solve the equation $y_j = \phi_j \psi s', j = 1, ..., M$ is a ill-pose problem. We know s' is sparse and should contain as many as zeros as possible. This underdetermined system of linear equations is solved by exactly satisfying the formula: $min\#(i|s'_i \neq 0)$ s.t. $Y = \phi_j \psi s'$. To solve this function we could use the definition of ℓ_0 norm, which requires exhaustive searches for all subset of columns of $\Psi\Phi$. An relaxation can be made with ℓ_1, ℓ_2 [28]. The convex relaxation reconstruction works with high accuracy and with a small number of measurements, but the methods are computationally complex. Therefore it is not suitable to be practically applied with huge data samples.

- Greedy Iterative Algorithm, for example like Orthogonal Matching Pursuits (OMP) [30] and Compressive Sampling Matching Pursuits (CoSaMP)[31]. Compared to the convex relaxation the recovery is faster by using low complexity of the mathematical framework. This algorithm utilizes a greedy strategy for support detection. This is done by selecting the columns of *Phi* to approximate the support set of measurements y and then solve a least square problem to recover the original signal.

Algorithms	Classification	Complexity	Minimum Measurement
Basis Pursuit [33] [34]	Convex Relaxation	$O(n)^3$	$O(s\ logn)$
OMP [34] [35]	Greedy Iterative Algorithm	$O(s\ m\ n)$	$O(s\ logn)$
CoSaMP [35]	Greedy Iterative Algorithm	$O(m\ n)$	$O(s\ logn)$
Chaining Pursuits	Sublinear Algorithms [36]	$O(s\ log^2\ n\ log^2\ s)$	$O(slog^2n)$
HHS [37]	Sublinear Algorithms	$O(s\ ploylog(n))$	$O(poly(s, logn))$
Sparse Matching Pursuits [38]	Iterative Thresholding Algorithms	$O(s\ log(n/s))$	$O(s\ log(n/s))$

Table 3.1: Complexity and minimum measurement of recovery Algorithms [39]

- Bregman Iterative Algorithms, it is used for convex optimization problems with a appealing speed, which utilizes the Bregman iterative regularization scheme to iteratively solving a sequence of unconstrained subproblems[32].

The table 3.1 shows, there are different kinds of algorithms used for recovering CS signals. These are noted with their classification, complexity and minimum measurement.

3.2 Proposed Method

In this section, we introduce the Compressed Sensing method by using one dimension signals like from the accelerometer of smartphone. Secondly we will introduce the Compressed Sensing method for two dimension signals like from image signals.

3.2.1 One-Dimension Compression

In this thesis we get the signal from the accelerometer sensor of the user's smartphone. Before transmitting those signals to the center server we proposed to investigate Compressed Sensing, which has been proved effective at energy consumption. Figure 3.3 depicts the main components used for compressing(i.e.encoding) and reconstructing(i.e. decoding) the measurements. At the mobile device compression is performed via simply multiplying the measurement vector $f \in \mathbb{R}$ by a sensing matrix Φ. The result $y \in \mathbb{R}^M$ is a compact version of the original data, where $M << N$.

At the server side, the data is recovered in two steps. First, a sparse representation of the original signal x^* is found through solving a convex optimization problem $y = \theta x^* = \Phi\psi x^*$.

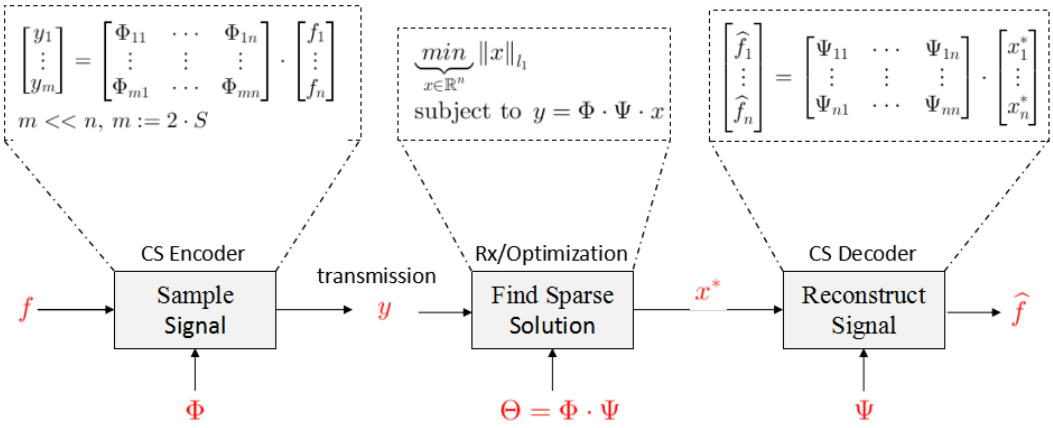

Figure 3.3: Compressed sensing accelerometer signal

Second, the original signal is recovered via converting the sparse signal x^* using a suitable transform ψ, such as Consine transform and Wavelet transform.

As it can be seen in figure 3.3 the compression process is straightforward which makes it a suitable candidate for "resources-taxed" devices such as smartphones and tablets. Furthermore the CS encoding scheme results in indirect data encryption. Using CS for encoding depends highly on their vector size M. The size of the matrix $\Phi \in \mathbb{R}^{MxN}$ determines the compression rate of the original data. The higher M is, the less the data are compressed. At the same time, as Φ is used for encryption, its size determines the complexity of guessing it. Hence, here there is a clear trade-off: the higher M gets the less data is compressed and the more secure the encryption is. Smaller data compression can save energy but at the same time makes CS-based encryption weaker.

3.2.2 Two-Dimension Compression

The image compression is a typical example for the two dimension compression. The common techniques for image compression process are Discrete Cosine Transform (DCT), Fast Fourier Transform (FFT) and Discrete Wavelet Transform (DWT).

The most of nature signals are not represented in sparse, but they are sparse in some appropriate transform domain. We could choose even discrete wavelet transformation with a suitable threshold value or discrete cosine transformation for sparse process of gray images. The threshold value controls the sparse level of the original image, so that it has a direct effect on the image reconstruction.

Figure 3.4 shows the process of a gray image using the compressed sensing method. Firstly the input image will be read as a matrix. When CS is applied for two-dimensional signals, like images we should first transform them into one-dimensional signals, for example through zig-zag quantization into a long vector. Then the vector will be multiplied with a large random matrix. The compressed signal will be reconstructed in the server side using recovery

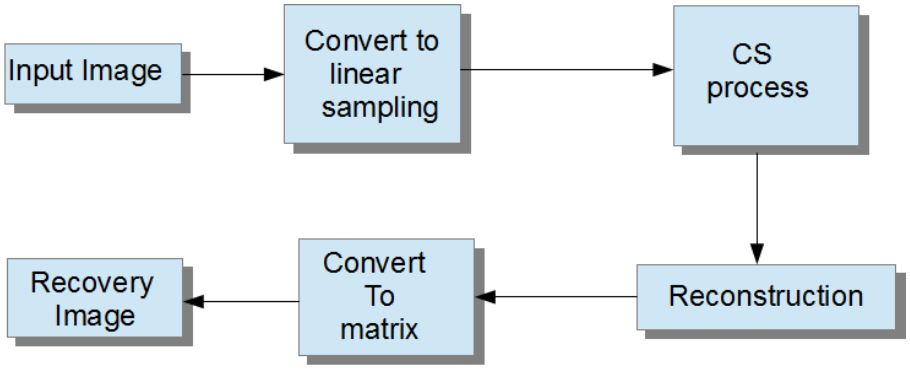

Figure 3.4: CS process for image compression

algorithms. Before getting the recovery image we still need to covert the one dimension signal back to a two dimension matrix. Alternatively, we could divide the original image into several blocks. For each of those blocks at first a discrete wavelet transform could be done. After the DWT process we get the coefficients of the appropriate transform domain. After the reconstruction step still the inverse DWT needs to be done for getting each pixel block.

Chapter 4

Implementation

The primary goal of this thesis is to collect data using mobile devices, compress the data and send the result to a PS server where the data should be decompressed and analyzed in order to find out what is the user activity took place. This activity can be walking, standing, holding, etc. In order to find this out the decompressed signals are to be compared to samples where it is known what activity took place. For the mobile device part it was chosen to use the Google Android platform and develop the necessary application for this operating system. One reason for this choice is the availability of devices for testing, another one is that this operation system does have very little limitations regarding running background processes and the general possibility to access any kind of sensor built into the device. The third reason is that it is easy to install the created application container to any android smartphone.

4.1 Sensor Data Acquisition

At first, we describe the equipments which was used to measure the power consumption in our work. The device used as a sensor sampling node was a Samsung Galaxy Note 3.Table 4.1 shows the important specification for our work. The communication function offers LTE as well as Wifi. It uses a three axis acceleration sensor for capturing the activity signals. Since the voltage change has some influence on the remaining capacity of the battery we measured the energy consumption data starting at a similar battery state each time [40].

In order to measure the acceleration values for later evaluation of the movement pattern the internal three dimensional acceleration sensor of the smartphone is used. In order to measure

Component	Specification
CPU	Quad-core 2.3 GHz
Operating system	Android 4.4.2 (KitKat)
Display	Super AMOLED, 5.7 inches, 1080 x 1920 pixels
ROM	16GB
RAM	3GB
Battery capacity	3200mAh

Table 4.1: Specification of Samsung Galaxy Note 3

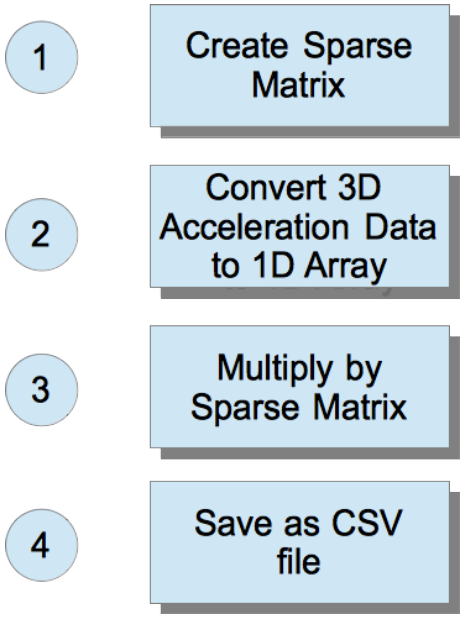

Figure 4.1: General compression sequence

these values using Android it is necessary that the class which wants to do this implements the **SensorEventListener** interface. It is then possible to use the class **SensorManager** to register to be informed when updates of the signal are available. In this case the **onSensorChanged(SensorEvent event)** method will be called by the operating system. This method must be implemented to handle the received data. When exiting the application it is necessary to call the **SensorManager** to un-register again from the service.

In order to have a better data handling it was decided to create a class representing a three dimensional acceleration vector. This class type is used for storing the received acceleration values:

$$\vec{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

There will be objects created out of this class. These can be easily processed and stored in any kind of required data structure in Java since all of these can structures can handle content having the type 'Object'.

4.2 CS-Based Compression

One of the main aspects of this thesis is to test and compare different compression algorithms. Figure 4.1 shows the general sequence of how the movement data is compressed in this project. The first step is to create a sparse matrix. For this thesis several kinds of sparse matrix have

been tested. The general idea about using a sparse matrix is that the high amount of zeros reduces the details and therefor as well reduces the file size.

The acceleration data has been measured and stored as 3D vectors consisting of a_x, a_y and a_z. For the purpose of further data reduction the vectors have been transformed to a normed vector combining all three dimensions in one:

$$a_{xyz} = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

This is done with every pattern to be recognized but it was done as well with the samples stored in the example database. In the third step the resulting 1D array out of step two gets multiplied with the created sparse matrix. These matrix multiplications were done using available source code of the Princeton University. In the last step the created data out of step three is formatted in the CSV[1] format and locally stored on the smartphone. This format was chosen because it can be easily imported wherever the data is needed. The raw data coming from the acceleration sensor is stored as well in a CSV file. In this way it is possible to later on compare the compressed signal and the raw signal to judge the quality of the compressed data. After completion of step four the data is ready to be transferred a node running Matlab for evaluation.

4.2.1 Sensing Matrices

The following matrices have been tested as sparse matrix: The random Gaussian matrix, the Toeplitz matrix, the and the random measurement matrix. The creation of the random Gaussian matrix was done as follows (cf. Algorithm 1). Two nested loops are used to fill a two dimensional array with values coming from the **nextGaussian()** method.

Algorithm 1 Computation of the random Gaussian matrix

```
1: procedure calcGauss
2:     for each integer i=0; i<384; i++ do
3:         for each integer j=0; j<384; j++ do
4:             gaussmatrix[i][j] = (int)(randomno.nextGaussian())
5:         end for
6:     end for
7:     Return gaussmatrix
8: end procedure
```

Alternatively to calculating the matrices it would be also possible to read in the values etc. from a CSV file. In this way no calculation at all would be necessary, but run-time for reading data from the file system. Since the calculations for creating the matrix are not run-time intensive the approach to dynamically create the matrix was chosen for this thesis. The Toeplitz matrix is dynamically created every time the app is started using the following Code.

[1] Comma seperated value

Algorithm 2 Computation of the Toeplitz Matrix

```
 1: procedure calcToeplitz
 2:     data[0]=1
 3:     data[1]=1
 4:     data[2]=1
 5:
 6:     for each integer i=3; i<b; i++ do
 7:         data[i] = 0
 8:     end for
 9:     for each integer i=0; i<(a-1); i++ do
10:         for each integer j=0; j<=(b-a); j++ do
11:             if j>i then
12:                 m[i][j]=data[j − i]
13:             else if j==i then
14:                 m[i][j] = data[0]
15:             else
16:                 m[i][j] = data[i − j]
17:             end if
18:         end for
19:     end for
20:     Return m
21: end procedure
```

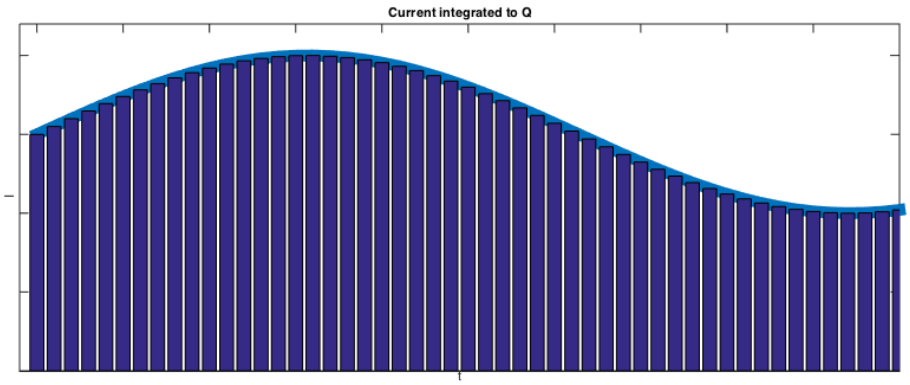

Figure 4.2: Calculation of the parameter Q

The third matrix was creating using the `la4j` library created for java. It is an open source library providing Linear Algebra primitives. Using this library it was possible to create the random measurement matrix with code: Matrix b = SparseMatrix.random(128,512,0.02,rand); The parameters handed over to the function specify the dimensions of the array to be created as well as the probability of the occurrence of a '1'.

4.2.2 Energy Measurement

There are different measurement options available to measure the energy consumption of the different compression algorithms. One approach would be to fully charge the phone and let the algorithm be running until the battery is empty. The advantage of this approach is that the battery capacity is known and that the time until the smartphone shuts down is relatively long so this should deliver relatively exact measurement values. The disadvantage of this method would be that it takes long to get results. Also it is not clear if the smartphone would not lower the CPU frequency for cooling purposes. Therefore, for this thesis another approach was taken. This is to integrate the current current with it's delta times:

$$\Delta Q = \int_{t1}^{t2} I(t)dt$$

.

The method **measure()** is called by the operation system via Intent every time there was a change at the battery status. It cannot be predicted when this will happen, because this is handled by the operation system. At this time it is possible to read information from the battery management like the current which is needed in this moment. The class **EnergyConsumption** is using this information in combination with timestamps to integrate how many mAh were needed.

The advantage of this approach is that the energy needed can be measured in "real time" and that is it not necessary to fully charge and discharge the battery to get results. The challenge

of using this method is that it might happen that the energy measurement method was not called by the operating system before and after the calculations. In this case it is necessary to repeat the measurement.

Generally it is necessary that the charger of the mobile phone is not connected while doing measurements because the current which is reported by the operating system might also be a charging current.

Algorithm 3 Computation of the energy consumption

```
1:  procedure EnergyConsumption
2:      m_ah = 0;
3:      idle_current = 0;
4:      timestamp = 0;
5:      timestamp_k1 = 0;
6:      measurementActive = false;
7:  end procedure
8:
9:  procedure startMeasuring
10:     measurementActive = true;
11:     timestamp_k1 = 0
12: end procedure
13:
14: procedure measure(float current)
15:     if measurementActive == true then
16:         timestamp = System.currentTimeMillis()
17:         if timestamp_k1 != 0 then
18:             // Integrate current
19:             m_ah+ = ((timestamp − timestamp_k1) ∗ (current_current))/(1000 ∗ 3600)
20:         end if
21:         // Last measured timestamp
22:         timestamp_k1 = timestamp;
23:     end if
24: end procedure
```

4.3 Data Transmission

For transmitting data from mobile phones to the node running Matlab a central server structure was chosen. It consists of the nodes running Android and a central server running Matlab for evaluation.

It was chosen to use the FTP[2] protocol to upload files from the client to the server. The advantage of this solution is that nearly every Unix based operating system is equipped and

[2]File transfer protocol

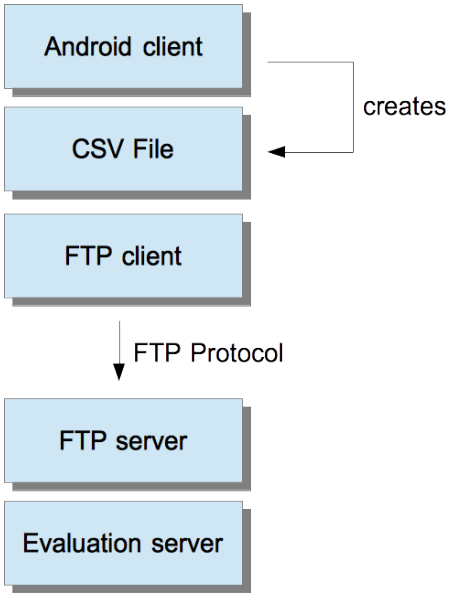

Figure 4.3: Data transmission using FTP

that any other operating system can be easily equipped with an FTP server. Also the setup of an FTP server can be easily done.

The sequence of transferring movement data from the client to the server is the following:

1. The Android client software is measuring acceleration data.
2. The acceleration data is compressed using spare matrices.
3. A CSV file is created out of the compressed information.
4. The FTP client on Android is connecting the the FTP server running on the Evaluation server.
5. The evaluation server is extracting the values out of the CSV file.
6. The movement data is decompressed.
7. An evaluation is conducted at the server to find out the activity done.

For doing a file transfer using FTP an FTP client as well as an FTP server is needed. In this thesis the client should be running on the Android system and the FTP server on the server (cf. Algorithm 4). The FTP client on the Android system is using the Java `SimpleFTP` Client Package. It is a freely available and allows the user the upload files to FTP servers. For a better handling and a better program structure, a wrapper function was created to upload files to the server.

Algorithm 4 Wrapper for uploading files

```
1: procedure UPLOADFILE(filename)
2:     ftp.connect("192.168.0.101", 21, "user", "xxxx")
3:     ftp.bin()
4:     server.ftp.cwd("/Users/Shared/CScompress_data")
5:     ftp.stor(newFile(filename))
6:     ftp.disconnect()
7: end procedure
```

4.4 Reconstruction Algorithm

At first activity signals from the smartphone are compressed and uploaded to the server side. When the data was received by the server they are decompressed and then the signals are recovered. In this thesis we are using a laptop at the server side running Matlab, version "MatlabR2014b". The reconstruction algorithms used for recovering the compressed sensing data are `l1magic` and `CVX`.

Listing 4.1: signal recovery

```
1        cvx_begin
2        variable sh(n) complex;
3        minimize(norm(sh,1));
4        subject to
5        norm(Theta*sh-y,2) < epsilon ;
6        cvx_end
7        fh = real(Psi * sh);
```

Figure 4.4 shows the original walking signal that was recorded by the smartphone accelerometer sensor. Each walking sample was recorded by 512 measurement points ranging from $\pm 8.5 m/s^2$ to $\pm 12 m/s^2$. Figure 4.5 shows the DCT coefficients of it. The original walking signal was compressed through compressed sensing mechanisms. This reduces data by multiplying the original data with a sensing matrix Φ without some complex calculation. In this example a random sparse matrix Φ from Berinde et. al. [29] was chosen. To achieve a compression ratio of 0.5 the value of M was set to 256. As the Matlab code shows 4.1, we use the CVX algorithm to recover the original walking signal. The recovered walking signal and the DCT coefficients are show as figure 4.6 and figure 4.7.

4.5 Activity Recognition Algorithm

The focus on the development of the recognition system using Matlab on the server side was to differentiate between people's motions and activities in their daily life. The activity signal was collected using the accelerometer of Android-based smartphones. The variance of those fundamental activities is depend on the personal attributes and habits. For example if the

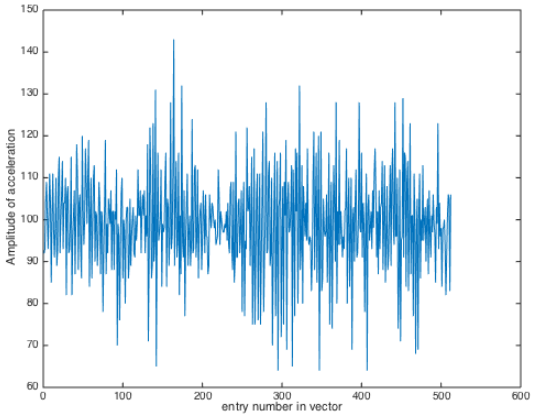

Figure 4.4: The original walking signal

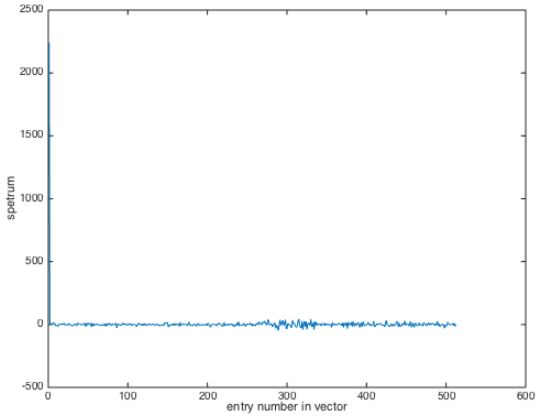

Figure 4.5: DCT coefficients of the original walking signal

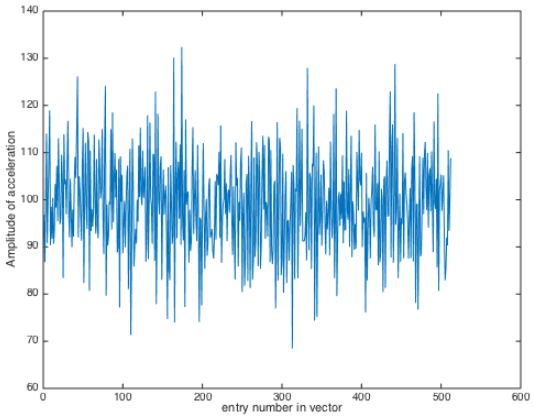

Figure 4.6: The recovered walking signal

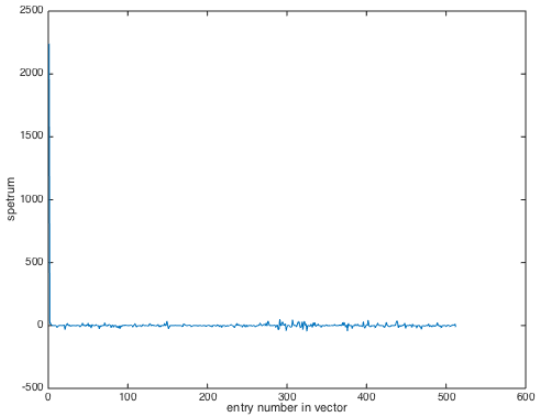

Figure 4.7: DCT coefficients of the recovered walking signal

	Training data					
	Holding	Jogging	Jumping	Walking	Stair up	Stair down
Holding						
Jogging						
Jumping						
Walking						
Stair up						
Stair down						

Table 4.2: DTW-based activity recognition

movement of children and old people are compared one difference might be that old people typically walk with slow temporal and do not jump as high as young people.

In our work we chose to use the DTW algorithm for the recognition process. We collected acceleration data of different activities of one subject in the training database. These were the following six different activities: holding, walking, jumping, jogging, going stairs up and going stairs down. Each activity was recorded by 512 acceleration measurement points. The training data and the test signal were used as input values for the DWT algorithm. The test signal was as well recorded with the same number of sampling points. Each test signal was compared to the whole training data set using the DTW algorithm. As a result of running the DTW algorithm comparing the test sample to the database samples we got six optimal warping paths. Among of them we chose the minimal value. This value has been able to classify the activity.

Table 4.2 shows the color coding used in this thesis. Table cells marked with yellow do indicate data coming from the training set. Table cells marked with green contain activity data from the test set. If the recognition of activities is correct, the cells filled with blue must be the ones with the smallest values. This indicates the optimal warp paths between two activities.

4.6 2D Compressed Sensing

In order to test compressed sensing methods for image compression we developed a small Android application processing an image. In order to have a comparison to traditional image compression methods we also compressed using the DCT algorithm. The calculation time needed is recorded as well as the energy consumption. This was done because the calculation time might be very short and therefore the resolution of the energy consumption module might be too low to deliver evaluable results.

In our work, we chose the image "lena.tif" with 50x50 pixels. This low resolution was chosen to reduce the computational workload which has to be done by the smartphone. An image with a higher resolution would be a heavy computational burden for the smartphone. The compressed sensing compression of images is working in the following way:

- In the first step the sample input picture is divided into red green and blue pixels. Only the red pixels are then stored in a two-dimensional array with the same resolution than the original picture. In order to test compressed sensing with a colorful bitmap it would be necessary to create as well an array from the green and blue pixels.
- In the second step the two dimensional image information is transformed from a matrix to an vector. Then compressed sensing is applied by multiplying this vector with a sensing matrix.
- The resulting file is then saved as a CSV file and stored in the file system of the smartphone. From there it can be copied to the computer for reconstruction using Matlab.

Chapter 5

Evaluation

In this chapter a set of tests is used to validate the system. The evaluation consist of three parts: the first part is the compressed sensing process, especially for the search on compressed sensing with different sensing matrices and different compression ratios. The delay time for uploading the data from smartphone to server also will be calculated. The results of them show which kind of sensing matrix is the most suitable one in our case. The average error is used to evaluate the effect of different compression ratios used for compressed sensing. The second part of the evaluation is the power consumption of the smartphone during the sampling time. The last part of this chapter is evaluation of the recognition system using the dynamic time warping algorithm.

5.1 CS Evaluations

In this section, we discuss the evaluation of the compressed sensing-based compression for 1D data, such as accelerometers and compasses. The evaluation has been done in terms of the energy gain thanks to applying the proposed method and the average error between the recovered signals and the original signals

5.1.1 Compression Accuracy

The first part of the evaluation is the average error of the activity signal. We implemented three different sensing matrices in the CS process. The random Toeplitz matrix, the random Gaussian matrix and the random sparse measurement matrix. The compression ratio (CR) and the normalized root mean square error (RMSE) are used for the performance evaluation as equation5.1, 5.2 and 5.3 shows.

$$Compression\ Ratio = \frac{Uncompressed\ size}{Compressed\ size} \tag{5.1}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - y_{i\ reconstructed})^2} \tag{5.2}$$

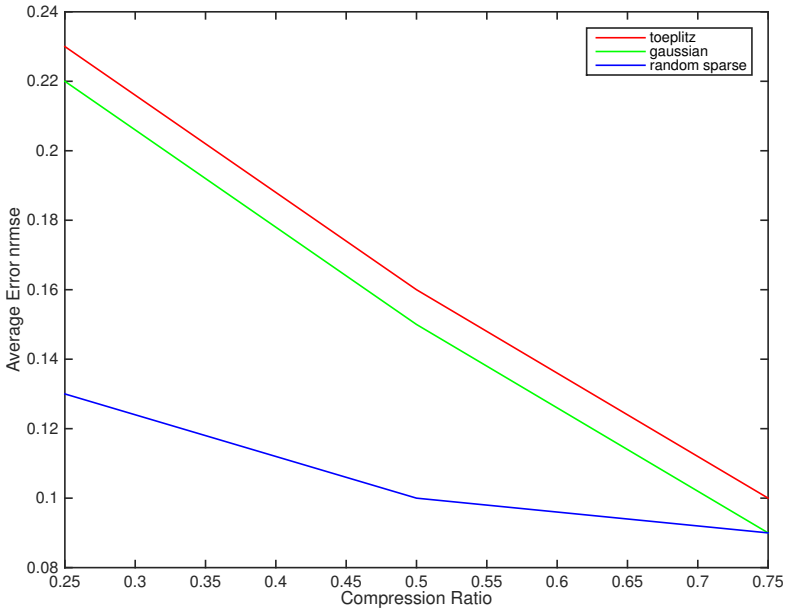

Figure 5.1: Walking signal using CS with three different matrices

$$normalized\ RMSE(NRMSE) = \frac{RMSE}{(y_{max} - y_{min})} \times 100\% \tag{5.3}$$

Figure 5.1 shows the example of the average error (nrmse) of a walking signal. If the random sparse matrix was used as sensing matrix in the CS process this was leading to get the minimal error with different compression ratios. Especially when the compression ratio is small, which means that a small value of M was chosen, the data could be compressed more. The Toeplitz matrix and the Gaussian matrix got less errors during less data was compressed. Figure 5.2 shows the delay time which resulted from using different sensing matrices in the smartphone. When the compression ratio gets larger; the delay time using the random Gaussian matrix increases linearly and faster then the other type of sensing matrices tested. Using the random Toeplitz matrix and random sparse measurement matrix the time grew nearly constant, using different compression ratios.

The figures 5.3, 5.4, 5.5, 5.6, 5.7 and 5.8 show the recovery average error of the basic activities: holding, walking, jumping, jogging, going stairs up, and going stairs down. The following compression ratios were tested: 0.25, 0.5 and 0.75. The sensing matrix chosen was the random sparse matrix. When the compression ratio is M/N = 0.5, it is possible to reconstruct the original signal with an approximate accuracy reaching from 0.10 to 0.145 NRMSE. The error of the stairs down signal is greater than those of the other activities. If more data was compressed, for example compression ratio is $M/N = 0.25$, the approximate accuracy is reaching from 0.26 to 0.16 NRMSE. The recovery of the jogging signal has the worst accuracy.

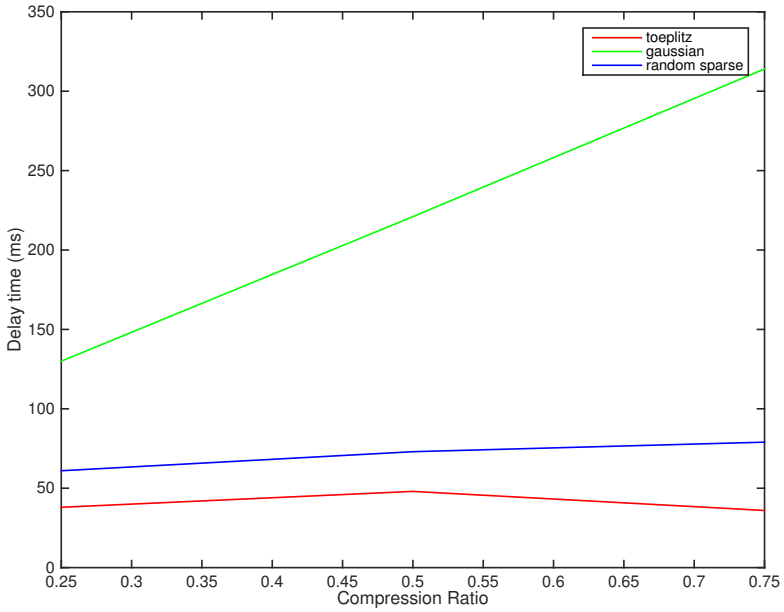

Figure 5.2: Time delay using CS with three different matrices

The signal of staying has the best accuracy (0.18 NRMSE) with a small compression ratio (0.25) and small fluctuations.

5.1.2 Uploading Energy Consumption

Figure 5.9 shows the energy consumption using three different sensing matrices: Toeplitz, Gaussian and random sparse matrix. All of those matrices increase the energy consumption linearly to the compression ratio. At a compression ratio of 0.25 the Toepliz matrix as sensing matrix in the CS process used over 10.7mAh, which is more than the other matrices. However, the toepliz matrix grows more smoothly in the energy consumption and has only small fluctuations. The other sensing matrix types grow faster when the compression ratio is increased, but those were good for using smaller compression ratios. Generally we could choose the random sparse matrix as sensing matrix because it has a well approximate accuracy with small compression ratios and the energy consumption is also good.

In our work, we also compared the time of the compressed sensing method to the time of ZIP compression method which is used for one dimension signals. The delay time of the ZIP method took 48ms for the compression of 512 samples. The delay time of the CS method took 28ms, 37ms, 39ms depend on the different compress ratios that we chosen. Generally, the delay time of compressed sensing method is much better than the ZIP method. Since the CS method has a lower time consumption for calculation than the ZIP compression method, it can

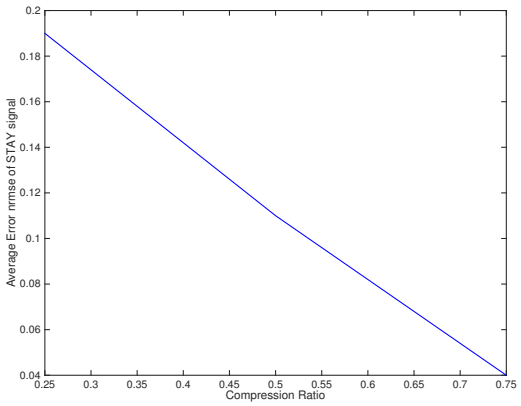

Figure 5.3: Recovery average error of the hold-ing signal

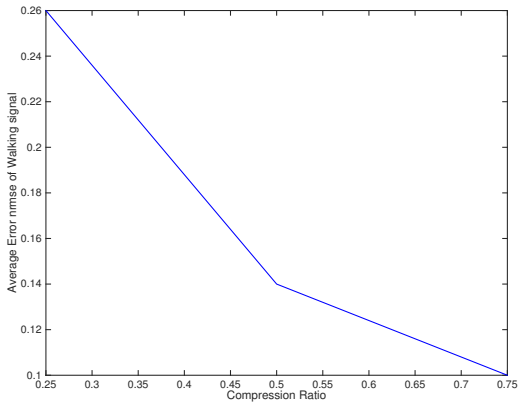

Figure 5.4: Recovery average error of the walking signal

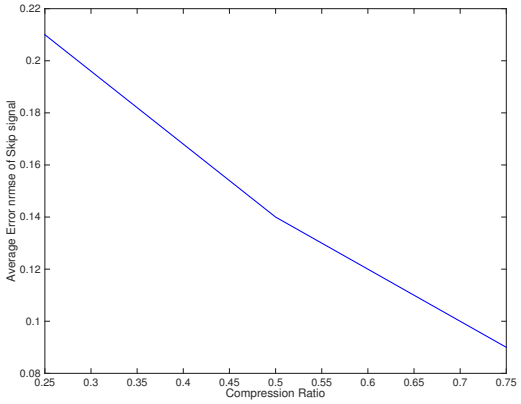

Figure 5.5: Recovery average error of the jump-ing signal

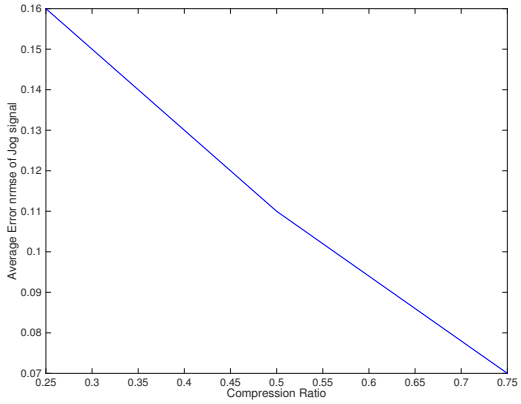

Figure 5.6: Recovery average error of the jogging signal

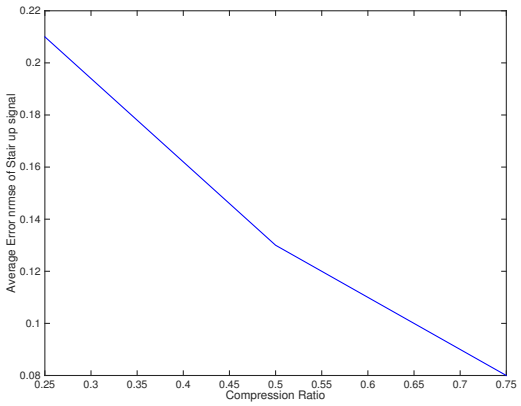

Figure 5.7: Recovery average error of the stair up signal

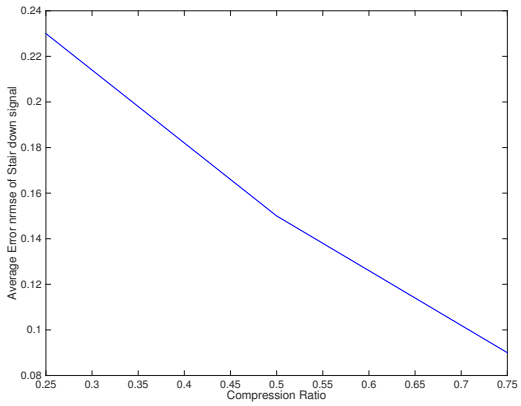

Figure 5.8: Recovery average error of the stair down signal

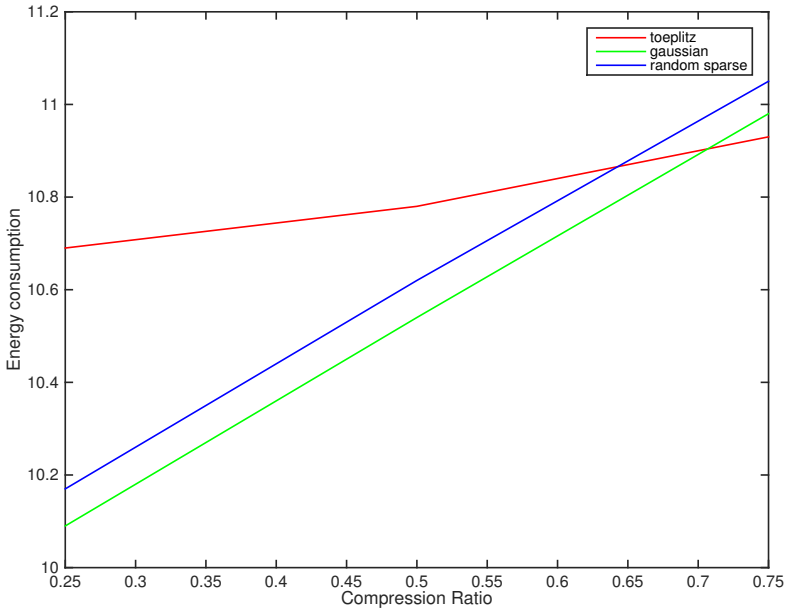

Figure 5.9: Battery consumption of the smartphone with different matrices

be also estimated that compressed sensing has a lower energy consumption for accelerometer data compression.

5.1.3 Activity Recognition Evaluations

Table 5.1 shows we used the DTW algorithm to calculate the optimal warp paths between the activity data from the training set and the data of input test data. Each test signal is compared with six activities signals. For identifying an input signal we compare it using DTW with all activities in the training data. The most minimal values in the row identifies the recognized activity. For example the first signal has the most minimal value compared to the stay signal, so that is possibly a stay signal. In our work the result shows, using the DTW algorithm for all of those input test signals (six test activities: Holding, jogging, jumping, walking, stairs up and stairs down) are well recognized.

5.2 CS-Based 2D Compression

In this section we evaluate the image compression using the compressed sensing method. Firstly figure 5.10, 5.11 and 5.12 show the original image using the CS method with different compression ratios and each least square of them. In the reconstruction step we chose the Basis Pursuit algorithm for recovering the compressed image signal, which was uploaded from the smartphone to server.

	Training data					
	Holding	Jogging	Jumping	Walking	Stair up	Stair down
Holding	366	14803	30784	4547	5858	5270
Jogging	17201	10775	17514	14639	14514	14419
Jumping	29689	20102	16879	26298	25878	25510
Walking	3473	12635	28284	3318	4050	3910
Stair up	7394	26037	11287	7394	4970	5058
Stair down	8621	24446	10937	8621	6513	6365

Table 5.1: Results of the DTW-based activity recognition

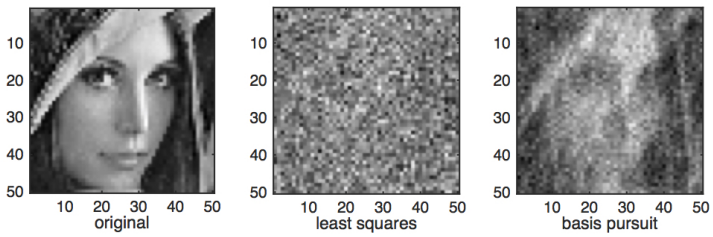

Figure 5.10: Recovery image with compression ratio 0.25

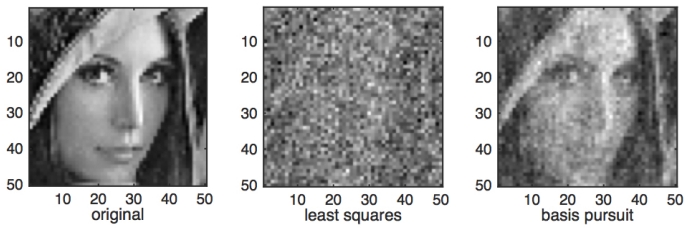

Figure 5.11: Recovery image with compression ratio 0.5

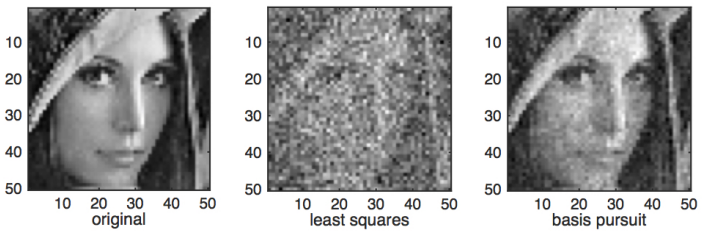

Figure 5.12: Recovery image with compression ratio 0.75

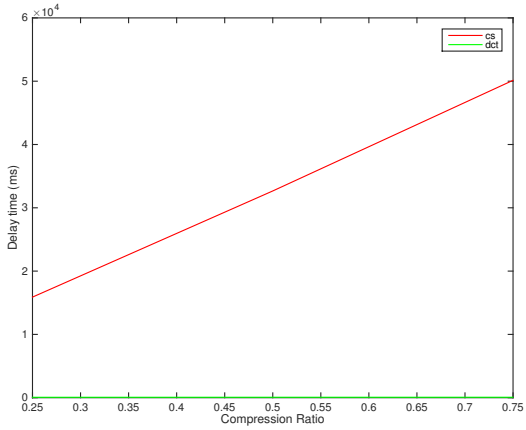

Figure 5.13: The time compare CS to DCT of image compression

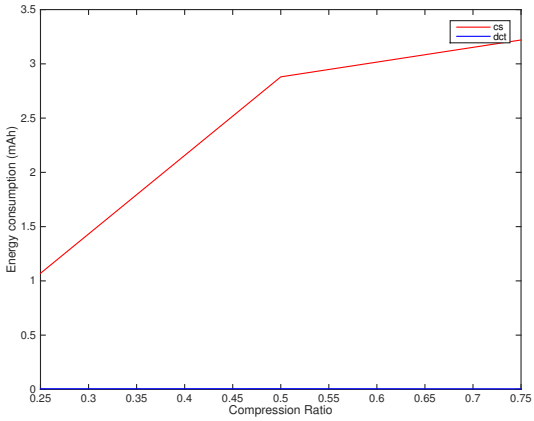

Figure 5.14: The time compare CS to DCT of image compression

Figure 5.13 shows the comparison of the delay time of CS image compression to the delay time of DCT image compression. The delay time of the DCT method seems to be independent on the selected compression quality regarding the time needed for compression. The CS method is linearly increased by using different compression ratios. The problem is that the image must firstly be converted to a long vector, then the compression process can be done by multiplying with a large sensing matrix, which possibly takes much time for calculation in the smartphone. The figure 5.14 shows, compare the energy consumption of CS image compression to the energy consumption of DCT image compression, the DCT method also seem better.

Chapter 6

Conclusions

6.1 Summary

The rapid development of mobile phones bring more opportunities for new technologies. Instead of building a transitional fix sensor infrastructure to collect data, there is now as well the possibility to develop ideas for mobile-based sensing mechanisms. The advantages of using smartphones are their flexibility, they are sensor-rich and easily scalable. The participators use their mobile phones to sample according to certain demands. Those information will then be regularly sent to a server infrastructure. The collected data can afterwards be used for a wide field of statistic researches.

From the other side this new technology can also bring some problems and challenges. The sampling and transmission process of a large quantity data could lead to a high power consumption. Since the battery run-time of nowadays smartphones is quite limited, this could be one of the reasons why people might reject the idea of participating in public sensing systems. We strive – though exploiting a recently-developed theory, referred to as the *compressed sensing* – to solve the problem of heavy transmission and computation costs at the smartphone side. It can significantly reduce the energy consumption needed for data sampling and compression at the mobile devices.

Specifically, the thesis targeted the time-driven public sensing applications which continuously acquire 1D data, such as acceleration, as well as 2D data such as images. The work is principally aimed to solve the problem of the limited available energy at the smartphones. Therefore, we used compressed sensing as compression method which showed to be suitable for this case. The CS compression was directly built in a smartphone application. Furthermore we tested different kinds of sensing matrices with different compression ratios and tried to find the most effective matrix sort for this situation regarding compression performance and energy consumption. To further examine the accuracy of the recovered signals, several activity signals were sent to the server side using wireless technology. On the server side, we used the dynamic time warping algorithm which match the received pattern with a set of stored activity patterns, trying to recognize the activities.

To validate the average error of compressed sensing we used different kinds of sensing matrices with different compression ratios. The activity recognition was validated by using recorded activity signals from different people where each sample to be tested was recorded twice. At

the end of the validation we compared compressed sensing methods with the traditional ZIP and DCT method to ensure the results. The power consumption was evaluated as well.

Our test results for one dimensional compressed sensing showed that there are no big differences of average errors at the first view which sensing matrix was used. However the random sparse matrix seemed to be the most efficient one for recovery of signals with high compression leading to a low number of input samples and it does not need much time for the compression process. The result of recovery average error of different activities showed that if the compression ratio is 0.5, the holding signal has the least recovery errors. Using compressed sensing for the activities signal could reduce the energy consumption of the smartphone, which is important for the acceptance of public sensing systems. Using an DTW algorithm makes it possible to build an activity recognition system with a good accuracy and a reasonable development effort.

The two dimensional compressed sensing method can be implemented on an Android smartphone and delivers acceptable results regarding the picture quality. On the other hand, because of it is necessary to convert the two dimensional picture into a long vector which has to be multiplied by a large sensing matrix. Therefore this calculation process could be too extensive to be executed on a smartphone. There are still some aspects where the whole system could be improved especially regarding the data set and communication principle aspect. The details of the improvement suggestions are summarized in the following section. We think these advices could be useful for the future work on this system.

6.2 Future Work

During writing of this thesis we summarized some limitations and features which can be extended in the future work.

- Activity recognition: In this thesis, we measured only activities of four people for the activity recognition. It would be possible to enhance the testing results by taking measurements and building a database with samples of more people. This would make it possible to get results which are more exact. For instance, Akimura et al. [40] explains that they used the dataset "HASC2010corpus" from N.Kawaguchi [41]. This data set contains the acceleration data of 90 people. For each person six activities where recorded: staying, walking, jogging, jumping, stairs up and stairs down. The total activity data consists of about 540 samples. Using this data either for the training data set, which is used for classification, or just as an input for testing would also make the evaluation more accurate.

- Communication principle: The communication strategy we chose for this thesis was designed for a smaller number of of participants uploading data to the server simultaneously. It was assumed that the smartphones have a reliable network connection to the server. The other point is security for transferring files to the server fixed login-data and no encryption was used.

The are several possibilities to enhanced each of these points. The first point is the general communication architecture. In opposite to the used client-server paradigm where multiple client nodes are accessing one server it would have been also possible to create for instance a P2P communication network. The scalability of such a network type would be very high, because there wouldn't be one single communication path limiting the communication throughput. This approach would as well increase the availability of the system because of one node fails the routing could be done using other nodes where the communication is working. Especially for mobile phones this would make sense because the connection the in network cannot be taken for granted. There is a lot of reasons why the communication might not be possible at least for a short time for instance because of poor reception. Because of this it would make sense to introduce some kind of mechanism that the cellphones should try from time to time to transmit it's data in case of communication limitations.

In order to tackle the security aspects there is also several possibilities how to enhance it. The first and most obvious one would be to introduce encryption between the nodes. This would make it more difficult for other people to sniff data, especially if the attacker is in the same network this would otherwise be very easy. The other point is that encryption would also make it more difficult to manipulate data of the own or other users.

Bibliography

[1] I. Constandache, S. Gaonkar, M. Sayler, R. R. Choudhury, and L. Cox, "Enloc: Energy-efficient localization for mobile phones," in *IEEE INFOCOM 2009*, pp. 2716–2720, April 2009.

[2] B. Guo, Z. Wang, Z. Yu, Y. Wang, N. Y. Yen, R. Huang, and X. Zhou, "Mobile crowd sensing and computing: The review of an emerging human-powered sensing paradigm.," *ACM Comput. Surv.*, vol. 48, no. 1, p. 7, 2015.

[3] D. Philipp, P. Baier, C. Dibak, F. Dürr, K. Rothermel, S. Becker, M. Peter, and D. Fritsch, "Mapgenie: Grammar-enhanced indoor map construction from crowd-sourced data," in *2014 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pp. 139–147, March 2014.

[4] S. Devarakonda, P. Sevusu, H. Liu, R. Liu, L. Iftode, and B. Nath, "Real-time air quality monitoring through mobile sensing in metropolitan areas," in *Proceedings of the 2Nd ACM SIGKDD International Workshop on Urban Computing*, UrbComp '13, (New York, NY, USA), pp. 15:1–15:8, ACM, 2013.

[5] S. Reddy, A. Parker, J. Hyman, J. Burke, D. Estrin, and M. Hansen, "Image browsing, processing, and clustering for participatory sensing: Lessons from a dietsense prototype," in *Proceedings of the 4th Workshop on Embedded Networked Sensors*, EmNets '07, (New York, NY, USA), pp. 13–17, ACM, 2007.

[6] D. Philipp, J. Stachowiak, F. Dürr, and K. Rothermel, "Model-Driven Public Sensing in Sparse Networks," in *Proceedings of the 10th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, LNCS, pp. 1–12, Springer, Dezember 2013.

[7] M. Wernke, F. Dürr, and K. Rothermel, "Speed Protection Algorithms for Privacy-aware Location Management," in *Proceedings of the IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2013).*, (Lyon, France), pp. 355–362, IEEE Xplore, Oktober 2013.

[8] Q. Li and G. Cao, "Efficient and privacy-preserving data aggregation in mobile sensing.," in *ICNP*, pp. 1–10, IEEE Computer Society, 2012.

[9] N. Thiagarajan, G. Aggarwal, A. Nicoara, D. Boneh, and J. P. Singh, "Who killed my battery?: Analyzing mobile browser energy consumption," in *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, (New York, NY, USA), pp. 41–50, ACM, 2012.

[10] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 27–34, ACM, 2003.

[11] C. Alippi, G. Anastasi, M. Di Francesco, and M. Roveri, "Energy management in wireless sensor networks with energy-hungry sensors," *IEEE Instrumentation & Measurement Magazine*, vol. 12, no. 2, 2009.

[12] B. Gedik, L. Liu, and S. Y. Philip, "Asap: An adaptive sampling approach to data collection in sensor networks," *IEEE Transactions on Parallel and distributed systems*, vol. 18, no. 12, pp. 1766–1783, 2007.

[13] R. Mietz and K. Römer, "Exploiting correlations for efficient content-based sensor search," in *Sensors, 2011 IEEE*, pp. 187–190, IEEE, 2011.

[14] C. Alippi, G. Anastasi, C. Galperti, F. Mancini, and M. Roveri, "Adaptive sampling for energy conservation in wireless sensor networks for snow monitoring applications," in *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE International Conference on*, pp. 1–6, IEEE, 2007.

[15] C. J. Debono and N. P. Borg, "The implementation of an adaptive data reduction technique for wireless sensor networks," in *Signal Processing and Information Technology, 2008. ISSPIT 2008. IEEE International Symposium on*, pp. 402–406, IEEE, 2008.

[16] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "In-network aggregation techniques for wireless sensor networks: a survey," *IEEE Wireless Communications*, vol. 14, no. 2, 2007.

[17] M. A. Razzaque, C. Bleakley, and S. Dobson, "Compression in wireless sensor networks: A survey and comparative evaluation," *ACM Transactions on Sensor Networks (TOSN)*, vol. 10, no. 1, p. 5, 2013.

[18] A. Akl and S. Valaee, "Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, x00026; compressive sensing," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2270–2273, 2010.

[19] Y. Bao, J. L. Beck, and H. Li, "Compressive sampling for accelerometer signals in structural health monitoring," *Structural Health Monitoring*, vol. 10, no. 3, pp. 235–246, 2011.

[20] Z. Tian and G. B. Giannakis, "Compressed sensing for wideband cognitive radios," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4, pp. IV–1357, IEEE, 2007.

[21] M. Lustig, D. Donoho, and J. M. Pauly, "Sparse mri: The application of compressed sensing for rapid mr imaging," *Magnetic resonance in medicine*, vol. 58, no. 6, pp. 1182–1195, 2007.

[22] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transfom," *IEEE Trans. Comput.*, vol. 23, pp. 90–93, Jan. 1974.

[23] K. R. Rao and P. Yip, eds., *The Transform and Data Compression Handbook*. Boca Raton, FL, USA: CRC Press, Inc., 2000.

[24] L. J. Zhenyu He, "Ieee standard for modeling and simulation (m amp;s) high level architecture (hla) - framework and rules," *IEEE Std. 1516-2000*, Oct 2009.

[25] T. Schoellhammer, B. Greenstein, E. Osterweil, M. Wimbrow, and D. Estrin, "Lightweight temporal compression of microclimate datasets," *Center for Embedded Network Sensing*, 2004.

[26] D. Salomon and G. Motta, *Handbook of Data Compression*. Springer Publishing Company, Incorporated, 5th ed., 2009.

[27] C. Sansone, J. Kittler, and F. Roli, *Multiple Classifier Systems: 10th International Workshop, MCS 2011, Naples, Italy, June 15-17, 2011. Proceedings*. Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2011.

[28] M. A. Davenport, M. F. Duarte, Y. C. Eldar, and G. Kutyniok, "Introduction to compressed sensing," *Preprint*, vol. 93, no. 1, p. 2, 2011.

[29] R. Berinde and P. Indyk, "Sparse recovery using sparse random matrices," *preprint*, 2008.

[30] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on information theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[31] D. Needell and J. A. Tropp, "Cosamp: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301–321, 2009.

[32] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, "Bregman iterative algorithms for \ell_1-minimization with applications to compressed sensing," *SIAM Journal on Imaging sciences*, vol. 1, no. 1, pp. 143–168, 2008.

[33] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Review*, vol. 43, no. 1, pp. 129–159, 2001.

[34] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE Transactions on Information Theory*, vol. 55, no. 5, pp. 2230–2249, 2009.

[35] D. Needell and R. Vershynin, "Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit," *Foundations of computational mathematics*, vol. 9, no. 3, pp. 317–334, 2009.

[36] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin, "Algorithmic linear dimension reduction in the l_1 norm for sparse vectors," *arXiv preprint cs/0608079*, 2006.

[37] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin, "One sketch for all: fast algorithms for compressed sensing," in *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pp. 237–246, ACM, 2007.

[38] R. Berinde, P. Indyk, and M. Ruzic, "Practical near-optimal sparse recovery in the l1 norm," in *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, pp. 198–205, IEEE, 2008.

[39] A. L. Pilastri and J. M. R. Tavares, "Reconstruction algorithms in compressive sensing: An overview," *DSIE| 16*, p. 127, 2016.

[40] D. Akimura, Y. Kawahara, and T. Asami, "Compressed sensing method for human activity sensing using mobile phone accelerometers," in *Networked Sensing Systems (INSS), 2012 Ninth International Conference on*, pp. 1–4, IEEE, 2012.

[41] N. Kawaguchi, N. Ogawa, Y. Iwasaki, K. Kaji, T. Terada, K. Murao, S. Inoue, Y. Kawahara, Y. Sumi, and N. Nishio, "Hasc challenge: Gathering large scale human activity corpus for the real-world activity understandings," in *Proceedings of the 2Nd Augmented Human International Conference*, AH '11, (New York, NY, USA), pp. 27:1–27:5, ACM, 2011.