8-2015

Compressed Sensing in Resource-Constrained Environments: From Sensing Mechanism Design to Recovery Algorithms

Shuangjiang Li
University of Tennessee - Knoxville, sli22@vols.utk.edu

To the Graduate Council:

I am submitting herewith a dissertation written by Shuangjiang Li entitled "Compressed Sensing in Resource-Constrained Environments: From Sensing Mechanism Design to Recovery Algorithms." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Computer Engineering.

Hairong Qi, Major Professor

We have read this dissertation and recommend its acceptance:

Russell Zaretzki, Qing Cao, Husheng Li

Accepted for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Compressed Sensing in Resource-Constrained Environments: From Sensing Mechanism Design to Recovery Algorithms

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Shuangjiang Li

August 2015

This dissertation is dedicated to my loving wife Ruoning

and my cute son Isaac Li (李溪根 *).*

Acknowledgements

I would like to thank all the individuals who have inspired, encouraged, and advised me in the preparation of this dissertation.

First and foremost, I would like to thank my advisor, Dr. Hairong Qi. Her willingness to support my work and her guidance throughout my studies has allowed me to develop my skills as a researcher within a supportive team environment. Her openness and determination gave me tremendous encouragement during my research. I thank her for that opportunity. Also, I would like to thank Dr. Russell Zaretzki, Dr. Husheng Li and Dr. Qing Cao. Their advice and counsel have been of equal importance. I greatly appreciate their time and input to this dissertation.

Within the AICIP Lab, I owe many thanks to my fellow graduate students. I enjoyed the many conversations and discussions that have had a great impact on my research and myself as a person. Wei Wang, Jiajia Luo, Rui Guo, Zhibo Wang, Zhifei Zhang, Liu Liu, Sangwoo Moon, Mahmut Karakaya, Dayu Yang, Yang Bai, Bryan Bodkin, Alireza Rahimpour, thank you very much.

Last but not the least, I express my deepest appreciation to my parents and my sister for their unconditional love, support and encouragement. I would not be able to achieve anything without their never-ending love and support. I feel very lucky and privileged to have them.

The fear of the LORD is the beginning of wisdom: and the knowledge of the holy is understanding. (Proverbs 9:10)

Abstract

Compressed Sensing (CS) is an emerging field based on the revelation that a small collection of linear projections of a sparse signal contains enough information for reconstruction. It is promising that CS can be utilized in environments where the signal acquisition process is extremely difficult or costly, e.g., a resource-constrained environment like the smartphone platform, or a band-limited environment like visual sensor network (VSNs). There are several challenges to perform sensing due to the characteristic of these platforms, including, for example, needing active user involvement, computational and storage limitations and lower transmission capabilities. This dissertation focuses on the study of CS in resource-constrained environments.

First, we try to solve the problem on how to design sensing mechanisms that could better adapt to the resource-limited smartphone platform. We propose the compressed phone sensing (CPS) framework where two challenging issues are studied, the energy drainage issue due to continuous sensing which may impede the normal functionality of the smartphones and the requirement of active user inputs for data collection that may place a high burden on the user.

Second, we propose a CS reconstruction algorithm to be used in VSNs for recovery of frames/images. An efficient algorithm, NonLocal Douglas-Rachford (NLDR), is developed. NLDR takes advantage of self-similarity in images using nonlocal means (NL) filtering. We further formulate the nonlocal estimation as the low-rank matrix approximation problem and solve the constrained optimization problem using Douglas-Rachford splitting method.

Third, we extend the NLDR algorithm to surveillance video processing in VSNs and propose recursive Low-rank and Sparse estimation through Douglas-Rachford splitting (rLSDR) method for recovery of the video frame into a low-rank background component and sparse component that corresponds to the moving object. The spatial and temporal low-rank features of the video frame, e.g., the nonlocal similar patches within the single video frame and the low-rank background component residing in multiple frames, are successfully exploited.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

We are in the midst of a digital revolution that is driving the development and deployment of new kinds of sensing systems with ever-increasing fidelity and resolution. The theoretical foundation of this revolution is the pioneering work of Kotelnikov, Nyquist, Shannon, and Whittaker on sampling continuous-time band-limited signals Kotelnikov (1933); Nyquist (1928); Shannon (1949); Whittaker (1915). Their results demonstrate that signals, images, videos, and other data can be exactly recovered from a set of uniformly spaced samples taken at the so-called Nyquist rate of twice the highest frequency present in the signal of interest. Capitalizing on this discovery, much of signal processing has moved from the analog to the digital domain and ridden the wave of Moore's law Wikipedia (2014). Digitization has enabled the creation of sensing and processing systems that are more robust, flexible, cheaper and, consequently, more widely used than their analog counterparts. As a result of this success, the amount of data generated by sensing systems has grown from a trickle to a torrent Economist (2010). Unfortunately, in many important and emerging applications, the resulting Nyquist rate is so high that we end up with far too many samples. Thus, despite extraordinary advances in computational power, the acquisition and processing of signals in application areas such as imaging, video, medical imaging, remote surveillance, spectroscopy, and genomic data analysis continues to pose a tremendous challenge Davenport et al. (2011).

In recent years, Compressed Sensing/Compressive Sampling (CS) Candès et al. (2006); Baraniuk (2007); Donoho (2006); Candes et al. (2006) has attracted considerable attention in areas of applied mathematics, computer science, and electrical engineering by suggesting that it may be possible to surpass the traditional limits of sampling theory Davenport et al. (2011). CS builds upon the fundamental fact that we can represent many signals using only a few non-zero coefficients in a suitable basis or dictionary. Nonlinear optimization can then enable recovery of such signals from very few measurements.

It is promising that CS can be utilized in the environment where the signal acquisition process is very hard or costly. Typically this would be in a resource-constrained environment, e.g., smartphone platform, or bandwidth-limited environment, e.g., Visual Sensor Networks (VSNs). There are several challenges to perform sensing due to the characteristic of the these platforms. For example, needing active user involvement, computational and storage limitations and lower transmission capabilities.

1.1 Mobile Phone Sensing

Todays smartphone not only serves as the key computing and communication mobile device of choice, but it also comes with a rich set of embedded sensors, such as an accelerometer, digital compass, gyroscope, GPS, microphone, and camera. Collectively, these sensors are enabling new applications across a wide variety of domains, such as social networking Miluzzo et al. (2008); Wang et al. (2011), health care Consolvo et al. (2008), location based services (LBS) Jiang et al. (2011); Zhuang et al. (2010), environmental monitoring Mun et al. (2009), and transportation Li et al. (2011); Thiagarajan et al. (2009); Mohan et al. (2008), and give rise to a new area of research called mobile phone sensing Lane et al. (2010).

Sensor enabled smartphones have revolutionized the way "sensing" can be performed Eagle and Pentland (2006); Cioffi-Revilla (2010). Depending on how much the user should be actively involved during the sensing activity, mobile phone sensing can be divided into participatory sensing Burke et al. (2006); Rana et al. (2010); Mun et al. (2009);

Das et al. (2010) where the user actively participates in the data collection activity (i.e., the user manually determines how, when, what, and where to sample), or, alternatively, opportunistic sensing Campbell et al. (2006) where the data collection state is fully automated with no user involvement Lane et al. (2010).

Although opportunistic sensing lowers the burden placed on the user, it is relatively less studied, perhaps due to a main challenge, the phone context problem. For example, the application wants to only take a few minutes of samples when the user is changing from the "running on a greenway" activity to the "reading in the library" activity. These types of context issues can be solved by using the phone sensors. However, that would require the phone sensors to continuously function to capture such transient events. Designed mainly for bursty user interaction, current smart phones rely on its main processors to control the sensors directly. Continuous sensing implies that the main processor has to stay on all the time, which may consume hundreds of milliwatts when they are active even with the screen and radios turned off. As a result, continuous sensing applications drastically reduce battery lifetime into a few hours, jeopardizing the usability of the phone. While there has been some previous efforts that took a hardware approach to solve this problem by processing sensor data on a dedicated low-power processor Priyantha et al. (2011a) or by customized OS support Roy et al. (2011), it brings challenges on application development in terms of complexity and portability.

Participatory sensing, on the other hand, places a higher burden on the user, who has to manually select what data to collect and when the data can be collected. Although it leverages human intelligence in resolving the context problem, the drawback is that the quality of data is dependent on participant enthusiasm to reliably collect sensing data and the compatibility of a person's mobility patterns to the intended goals of the application.

1.2 Visual Sensor Networks (VSNs)

Wireless Sensor Networks (WSNs) are becoming a mature technology after a decade of intensive worldwide research and development efforts. WSNs primary function is

to collect and disseminate critical data that characterize physical phenomena around the sensors Chong and Kumar (2003); Yick et al. (2008); Li and Qi (2013a). Availability of low-cost CMOS cameras has created the opportunity to build low-cost Visual Sensor Network (VSN) platforms able to capture, process, and disseminate visual data collectively Soro and Heinzelman (2009); Yang and Hairong (2010). The application of VSNs has spanned a wide spectrum of domains, including environmental surveillance, human behavior monitoring, and object tracking and recognition Sankaranarayanan et al. (2008); Obraczka et al. (2002)

VSNs are capable of collecting large volumes of data about monitored scenes but are constrained with the available node resources and network bandwidth. Designing and implementing VSNs is thus faced with several challenges. First, robust visual data processing are needed to produce useful data and reduce the amount sent over the network but are typically restricted with the node resources (memory and power). Second, since raw images are processed locally so only partial useful data is sent to the central station for further analysis or other nodes for collaborative processing, then how to collaboratively analyze the data for certain applications, e.g., object detection, surveillance, is very challenging.

1.3 Motivations

In order to address these challenges in sensing and processing on the resource-constrained smartphone platform and VSNs. With the advances in CS, we first try to study the problem of energy-efficient smart sensing through CS guided by proper models of human behavior to achieve opportunistic participatory sensing. That is, we achieve participatory sensing through learned knowledge of human behavior without the users involvement. We then propose an CS image recovery algorithm which could reliably reconstruct the visual data with fewer measurements needed but with high fidelity compared with existing CS recovery algorithms. The proposed CS image recovery algorithm can be used in various application

in VSNs, for example, surveillance video processing and medical imaging processing as well.

1.4 Contribution

The dissertation work presents a set of solutions based on CS theory to the aforementioned challenges in the resource-constrained environment. To this end, the current contributions include:

- An opportunistic sensing mechanism using pattern-based activity model for smartphone sensing. This is a traditional sensing scheduling problem Tian and Georganas (2003) in the smartphone sensing context to address the challenge on *when to sense* automatically.

- An energy-efficient randomized compressed phone sensing scheme using sparse binary measurement matrix for activity sensing. This addresses the problem on *how to sense* on the resource-constrained smartphone, where both the storage and computation burdens are relieved.

- An evaluation of the opportunistic sensing mechanism with randomized compressed phone sensing based on a case study on driving activity sensing.

- A new compressed sensing recovery algorithm, **NLDR** (NonLocal Douglas-Rachford splitting), that exploits the self-similarity within the image captured in VSNs. Thus, to achieve the same recovery performance, the proposed algorithm requires much less measurements compared to existing state-of-the-art CS recovery algorithm.

- A new algorithm named **rLSDR** (recursive Low-rank and Sparse estimation through Douglas-Rachford splitting) for segmentation of background by recursively estimating low-rank and sparse components in the reconstructed surveillance video frames from CS measurements. Experimental results on real surveillance videos showed that

NLDR performs best for CS frame recovery and rLSDR could successfully recovery the background and sparse object with less resource consumption.

1.5 Dissertation Outline

The dissertation is organized as follows: Chapter 2 provides a literature survey on the CS background, state-of-the-art recovery algorithms and CS applied on the smartphone sensing as well as VSNs. Chapter 3 explains the compressed phone sensing framework, the sensing scheduling and sample scheduling components. Chapter 4 discusses a case study on driving activity sensing which explains the evaluation metrics for CS sensing mechanism design, as well as the experimental results. Chapter 5 presents the NonLocal Douglas-Rachford splitting (NLDR) CS recovery algorithm for VSNs as well as the experimental results. Chapter 6 discusses the proposed recursive Low-rank and Sparse estimation through Douglas-Rachford splitting (rLSDR) algorithm for segmentation of background by recursively estimating low-rank and sparse components in the reconstructed surveillance video frames from CS measurements. Chapter 7 concludes the dissertation with accomplished and future works.

Chapter 2

Literature Review

2.1 A Review on Compressed Sensing

The Shannon/Nyquist sampling theorem tells us that in order to not lose information when uniformly sampling a signal we must sample at least two times faster than its bandwidth. Consider a real-valued, finite-length, one-dimensional, discrete-time signal x, which we view as an $n \times 1$ column vector in $\mathbb{R}^n$ with elements $x[n], n = 1, 2, \cdots, n$. We treat an image or higher-dimensional data by vectorizing it into a long one-dimensional vector. More often, people use transform coding in data acquisition systems like digital cameras where the number of samples is high but the signals are compressible. In this framework, we acquire the full n-sample signal x; compute the complete set of transform coefficients $\{s_i\}$ via $s = \Psi^T x$; locate the k largest coefficients and discard the $(n - k)$ smallest coefficients; and encode the k values and locations of the largest coefficients. (In practice, we also convert the values and locations to digital bits.) Unfortunately, the sample-then-compress framework suffers from three inherent inefficiencies: First, we must start with a potentially large number of samples n even if the ultimate desired k is small. Second, the encoder must compute all of the k transform coefficients $\{s_i\}$, even though it will discard all but k of them. Third, the encoder faces the overhead of encoding the locations of the large coefficients.

Instead of thinking in the traditional way, compressed sensing promises to recover the high-dimensional signals exactly or accurately, by using a much smaller number of non-adaptive linear samplings or measurements. In general, signals in this context are represented by vectors from linear spaces, many of which in the applications will represent images or other objects. However, the fundamental theorem of linear algebra, "as many equations as unknowns," tells us that it is not possible to reconstruct a unique signal from an incomplete set of linear measurements. As we said before, many signals such as real-world images or audio signals are often sparse or compressible over some basis, such as smooth signals or signals whose variations are bounded. This opens the room for recovering these signals accurately or even exactly from incomplete linear measurements.

2.1.1 CS formulation

The original CS formulation is as follows:

$$\min_x \|x\|_{\ell_0} \text{ subject to } \Phi x = y, \tag{2.1}$$

where the ℓ_0-norm is the number of non-zero elements. This should allow the perfect reconstruction from only $2K$ sampling points. Unfortunately, the problem is computationally intractable ("NP-hard") and impossible to solve in practical cases. Here the CS theory comes into play, with its central theorem stating, that with overwhelming probability $\ell 1$-norm minimization gives the same result as ℓ_0-norm minimization Candès et al. (2006); Candès and Tao (2006).

$$\min_x \|x\|_{\ell_1} \text{ subject to } \Phi x = y, \tag{2.2}$$

The number of samples required is slightly higher $\mathcal{O}(Klog(N/K))$, but the minimized function becomes convex and the task can be solved in a reasonable time with available algorithms. Importantly, the approach works also for nearly-sparse signals (or signals less

sparse than assumed) and signals with noise Candès and Wakin (2008), where the Eq. (2.2) has to be reformulated as:

$$\min_x \|\Phi x - y\|_{\ell_2}^2 + \lambda \|x\|_{\ell_1}, \tag{2.3}$$

where λ is a regularization parameter chosen to balance consistency with the data and sparseness. The main advantage of the ℓ_1-norm based search is that it can be performed using relatively simple and fast algorithms. One of them, Iterative Soft Thresholding (IST) is particularly popular

2.1.2 Matrix properties: RIP

Candès and Tao Candes and Tao (2005) introduced the following isometry condition on matrices Φ and established its important role in CS. Given a matrix $\Phi \in \mathbb{R}^{m \times n}$ and any set T of column indices, we denote by Φ_T the $m \times \#(T)$ (i.e., $m \times |T|$) matrix composed of these columns. Similarly, for a vector $x \in \mathbb{R}^n$, we denote by x_T the vector obtained by retaining only the entries in x corresponding to the column indices T. We say that a matrix Φ satisfies the *Restricted Isometry Property (RIP)* of order k if there exists a $\delta_k \in (0, 1)$ such that

$$(1 - \delta_k)\|x_T\|_2^2 \leq \|\Phi_T x_T\|_2^2 \leq (1 + \delta_k)\|x\|_2^2 \tag{2.4}$$

holds for all sets T with $\#T \leq k$ (i.e., $|T| \leq k$). The condition (2.4) is equivalent to requiring that the Grammian matrix $\Phi_T^t \Phi_T$ has all of its eigenvalues in $[1 - \delta_k, 1 + \delta_k]$ (here Φ_T^t denotes the transpose of Φ_T).

2.1.3 Recovery algorithm

The principle of **Basis Pursuit (BP)** Chen et al. (2001) is to find a representation of the signal whose coefficients have minimal ℓ_1 norm. Formally, one solves the problem

$$\min\|\alpha\|_1 \text{ subject to } \Phi\alpha = s. \tag{2.5}$$

BP is an optimization principle, not an algorithm.

Greedy

One popular class of sparse recovery algorithms is based on the idea of iterative greedy pursuit. The earliest one include the matching pursuit (MP) by G. Mallat, et al.Mallat and Zhang (1993), later advanced by Y. Pati, et al.Pati et al. (1993) and G. Davis, et al.Davis et al. (1994). Matching pursuit is related to the field of compressed sensing and has been extended by researchers. Notable extensions are Orthogonal Matching Pursuit (OMP)Tropp and Gilbert (2007), Stagewise OMP (StOMP)Donoho et al. (2012), and compressive sampling matching pursuit (CoSaMP) Needell and Tropp (2009).

Algorithm 1: Matching Pursuit

Input:

▶ Measurement matrix $\Phi \in \mathbb{R}^{m\times n}$.

▶ Observation vector $y \in \mathbb{R}^m$.

Output:

▶ An estimate $\hat{x} \in \mathbb{R}^n$ of the ideal signal x.

1: $\hat{x}_0 = 0$, $r^{(0)} \leftarrow y$, $i = 0$ ▷ Initialization
2: while *halting criterion false* do
3: $i \leftarrow i + 1$
4: $\phi_i \leftarrow \underset{\phi_i\in\Phi}{\operatorname{argmax}}|\langle r^{(i-1)}, \phi_i\rangle|$ ▷ The column of Φ that is most correlated with $r^{(i-1)}$
5: $\hat{x}_i \leftarrow \langle r^{(i-1)}, \phi_i\rangle$ ▷ From residual new signal estimate
6: $r^{(i)} = r^{(i-1)} - \phi_i\hat{x}_i$ ▷ Update residual
7: end while
8: return $\hat{x} \leftarrow \hat{x}_i$

Matching Pursuit was originally introduced in the signal-processing community as an algorithm "*that decomposes any signal into a linear expansion of waveforms that are selected from a redundant dictionary of functions*" Mallat and Zhang (1993). It is a general, greedy, sparse function approximation scheme with squared error loss, which iteratively adds new functions (i.e., basis functions) to the linear expansion.

The essence of matching pursuit, Algorithm 1 is that, for a given vector x to be approximated, first choose the vector from the dictionary on which x has the longest projection. Then, remove any component of the form of the selected vector from x, i.e., orthogonalize x with respect to the selected dictionary vector, and obtain the residual of x. The selected dictionary vector is in fact the one that results in the residual of x with the smallest energy. Repeat this process for the residual of x with the rest of dictionary vectors until the norm of the residual becomes smaller than the threshold ε.

In matching pursuit, after a vector in the dictionary is selected, one may remove any component of its form not only from x, but also from all other dictionary vectors before repeating the process. This version of the method is called orthogonal matching pursuit and is computationally more expensive than the nonorthogonal version, but typically gives significantly better results in the context of coding.

The reconstruction complexity of these algorithms (OMP, StOMP, ROMP) is around $\mathbb{O}(KMN)$, which is significantly lower than the BP methods. However, they require more measurements for perfect reconstruction and they lack provable reconstruction quality. More recently, greedy algorithms such as the subpace pursuit(SP) Dai and Milenkovic (2009) and the compressive sampling matching pursuit (CoSaMP) Needell and Tropp (2009) have been proposed by incorporating the idea of backtracking. They offer comparable theoretical reconstruction quality as that of the LP methods and low reconstruction complexity. However, both the SP and the CoSAMP assume that the sparsity K is known, whereas K may not be available in many practical applications.

Algorithm *sparsity adaptive matching pursuit (SAMP)* Do et al. (2008), could recover signal without prior information of the sparsity. Which make it promising for many practical applications when the number of non-zero (significant) coefficients of a signal is not available.

2.2 Compressed Sensing on the Smartphone Platform

Although there has been rich literature on the application of compressed sensing in wireless sensor networks (WSNs) Luo et al. (2009); Chou et al. (2009), we emphasize the unique characteristics of mobile phone sensing, especially with human-centric sensing. First, mobile phones are tightly coupled with their users, following them to almost every single activity they engage during the course of the day. There usually is no backup phones in case the battery is drained or the phone is damaged. For the purpose of activity sensing, the phone is the only sensing platform (although multiple sensors can be built-in on the same platform) functioning. On the contrary, in WSNs, multiple sensing platforms are deployed to sense the same environment with high redundancy. The sensors are also only loosely coupled with the environment. Second, since human-centric mobile phone sensing targets at human activity understanding, the sensing schedule can be made to correlate with human activity pattern, in other words, mobile phone sensing should take advantage of the context. The same is not true for sensing in WSNs. Because of these differences, the compressive sensing problem on mobile phones requires fundamentally different approaches compared to sensor networks.

The author in Yang and Gerla (2010) studied an energy-efficient accelerometer data transfer of human body movement using CS. The characteristics of human body movements were investigated, and the advantage of the CS framework in terms of energy saving was examined when it is applied to the wireless accelerometer data transfer system. The experimental results showed that the movement data collected by accelerometers on several parts of human body is sparse enough to be compressed by CS and the CS framework can save up to 40% of energy in the sensing unit, compared with the traditional data compression scheme. In Akimura et al. (2012), the CS theory was applied to activity sensor data gathering for smartphones. The experiments were done on the iPhone platform by an application that continuously samples, compresses, and sends acceleration data to a server using CS. Evaluation of the reconstruction error and the recognition accuracy of 6 basic human activities using the acceleration data corpus of 90 test subjects were provided. The

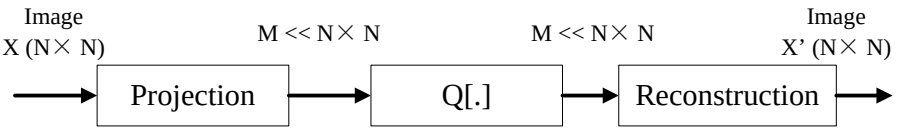


Figure 2.1: Simulated system block diagram Barakat et al. (2008).

results showed that CS scheme reduced power consumption by 16% as compared to the traditional ZIP compression method with the recognition accuracy of the 6 basic activities over 70%.

2.3 Compressed Sensing for VSNs

The authors in Barakat et al. (2008) studied the performance of CS for VSN images in terms of complexity and quality of reconstruction. In order to assess the performance of CS, the authors implement the block diagram shown in Fig. 2.1, where X is the input image of $N \times N$ pixels, and M is the number of measurements. The projection is performed onto a measurement matrix whose elements are generated by gathering 256 samples of the Fourier coefficients of the input image X along each of r radial lines in the frequency plane as explained in Barakat et al. (2008). The authors show that it is possible to operate at very low data rates with reduced complexity and still achieving good image quality at the reception. Based on CS, an image representation scheme for VSN is proposed in Han et al. (2010). The target image was firstly divided into two components through a wavelet transform: dense and sparse components. The former is encoded using JPEG or JPEG2000, while the latter is encoded using CS. In order to improve the rate distortion performance, the authors suggested leveraging the strong correlation between dense and sparse components using a piecewise autoregressive model. In general, the proposed work reduces the number of random measurements needed for CS reconstruction and the decoding computational complexity, compared to traditional CS methods.

In Wakin et al. (2006), the authors suggested algorithms and hardware implementation to support CS. In fact, they used a camera architecture, called single-pixel camera (which

is detailed in Baraniuk (2008)), which employed a digital micromirror device to carry out optical calculations of linear projections of an image onto pseudorandom binary patterns. Its main characteristic is the ability to acquire an image with a single detection element. This can significantly reduce the computation and the power required for video acquisition and compression. In Gan et al. (2008), the authors proposed a sparse and fast sampling operator based on the block Hadamard transform. Despite its simplicity, the proposed measurement operator requires a near optimal number of samples for perfect reconstruction. From the practical standpoint, the block Hadamard transform is easily implemented in the optical domain (e.g., using the singlepixel camera Baraniuk (2008)) and offers fast computation as well as small memory requirement. The suggested algorithm seems very efficient to be applied in power-constrained applications such as VSN. The unique work adopting CS paradigm in the context of VSN is that one developed in Chen et al. (2008), where both CS and JPEG are used for compression purpose. No details about the CS scheme are furnished in Chen et al. (2008).

Chapter 3

Compressed Phone Sensing

In this chapter we present an unobtrusive, energy-efficient approach to human activity sensing through the *intelligent scheduling* of built-in sensors on smartphones and *light-weight compressed sensing (CS)*. We refer to this framework as compressed phone sensing (CPS) where two challenging issues are studied, the energy drainage issue due to continuous sensing which may impede the normal functionality of the smartphones and the requirement of active user inputs for data collection that may place a high burden on the user. The proposed CPS framework consists of two components, the sensing scheduling and the sample scheduling, where CS-based techniques are applied to "sample" the temporal dimension to effectively control the "on" and "off" of the sensing and sampling activities of built-in sensors such that "smart"-phones can truly possess the "smart" sensing capability. In the sensing scheduling component, the pattern-based activity is first defined and modeled based on the "pattern matrix". We then propose a weighted model-based and CS-based sensing scheduling approach to turn on/off smartphone sensors. When the sensors are turned on, in the sample scheduling component, CS is adopted again to schedule how to sense samples. We propose a light-weight randomized CS scheme based on the sparse binary measurement matrix, that results in only addition operations for the resource-limited smartphones. With CS being its core and applied in two layers of activity sensing at different scales, the CPS framework has hence cohesively achieved energy efficiency

without sacrificing sensing accuracy. In Chapter 4, a case study on driving activity sensing shows that CPS framework can have, on average, the sensing scheduling accuracy about 83.92% but with 62.86% less overall energy consumption as compared to the continuous sensing.

3.1 Introduction

With the widespread popularity of smartphones and the rich set of built-in sensors (e.g., GPS, accelerometer, temperature, etc.), smartphones have revolutionized the way "sensing" can be performed. Collectively, these sensors have made available a wide variety of applications across different domains, including, for example, social networking Miluzzo et al. (2008); Wang et al. (2011), health care Consolvo et al. (2008), location based services (LBS) Jiang et al. (2011); Zhuang et al. (2010), environmental monitoring Mun et al. (2009), and transportation Li et al. (2011); Thiagarajan et al. (2009); Mohan et al. (2008), among which human centered smartphone activity sensing has gained more and more attention.

Activity sensing is not a trivial task. There exist quite some uncertainties in a given person's activities across varying timescales. Therefore, *continuous sensing* has been demanded for maintaining signal accuracy and integrity. While smartphones continue to provide more advanced capabilities in computation, memory, storage, battery, and sensing, the phone is still a resource-limited device. Therefore, *participatory sensing* Lane et al. (2010), which allows users to manually decide when, where, how, and what to sense, has been emerging recently. Although the user can control the frequency and duration of the sensing task as well as intelligently making usage of the battery, this would place extra burden on the user and is also impractical for sensing tasks that could last for several months or large-scale community based activities sensing where minimal human intervention would be necessary.

Ideally, modern "smart"-phones should also possess the capability of "smart" sensing, the ability to "intelligently" predict when and how to sense. Since human centered activity

sensing targets at individual's activity, the sensing schedule can be made to correlate with one's activity pattern. In other words, smartphone sensing should take advantage of the *context*. Human behavior modeling or pattern discovery has been well studied by engineers as well as human science researchers Kim et al. (2010); Lane et al. (2011). While many works have been focused on how to recognize or discover unknown human activities by analyzing the raw data, little work has been conducted on how activity pattern can facilitate the sensing task.

In this dissertation, we explore a new direction where we study how to intelligently perform data acquisition on smartphone without jeopardizing normal phone usage or putting extra burden on the user. We propose a compressed phone sensing (CPS) framework for human centered activity sensing and solve two common issues in any smartphone activity sensing; **when to sense** (i.e., sensing scheduling to decide when to turn on/off the sensor) and **how to sense** (i.e., sample scheduling to decide which samples to/not to acquire when the sensor is on) in order to maximize the battery life. To that end, data analysis such as activity recognition, discovery or understanding is out of the scope of this work.

Compressed Sensing (CS) Donoho (2006); Candès et al. (2006) is the fundamental building block of the CPS framework. CS is itself an effective sampling approach. By exploiting the sparsity property of a signal, CS can reconstruct the signal with far fewer samples than required by the Shannon-Nyquist sampling theorem. In this chapter, we apply the idea of compressed sensing or compressive sampling to the area of sensor scheduling. That is, instead of continuous sensing where the sensor is kept on all the time, the temporal dimension is "sampled" using the CS technique. That is, only when a certain time slot (e.g., 5-min) is "sampled", the sensor needs to be turned on; otherwise, the sensor should stay in the "off" mode to save energy. In addition, when the sensor is turned on, CS is applied again to "sample" the temporal dimension in finer detail, such that only a subset of samples need to be acquired. These two levels of application of CS schematically "samples" the temporal domain at two different scales to achieve the best tradeoff between energy efficiency and accuracy of activity sensing. By accuracy, we mean the samples acquired are samples

when the activity actually occurs. We refer to these two levels of scheduling as sensing scheduling and sample scheduling, where CS is incorporated in both levels.

The proposed CPS framework mainly consists of two components, i.e., Sensing Scheduling (*SenS*) and Sample Scheduling (*SamS*), as shown in Fig. 3.1. The *Sensing Scheduling* component tries to solve the problem regarding "*when to sense*". The major goal of the *SenS* component is to alleviate the phone from continuous sensing in order to save battery life.

In *SenS*, the concept of "probability of sensing" is introduced and its value is jointly determined by two different scheduling approaches: the model-based scheduling and compressed sensing (CS) based scheduling. The *SenS* component generally consists of two stages, *training stage* and *prediction stage*.

For the model-based scheduling, during the training stage, the "probability of sensing" vector is randomly initialized indicating the probability that an activity actually occurs in each of the 5-minute time slot during a day. When this probability is over certain threshold, the *SamS* component will be activated to perform the sensing task. Meanwhile, a *pattern matrix* is constructed that records the actual activity occurrence of a person on a daily base in each 5-minute interval. After accumulating sufficient training data, a Gaussian mixture model will be built based on the pattern matrix. During the prediction stage, the probability of sensing vector is initialized based on the Gaussian mixture model derived from the training stage such that it has a higher probability to catch the activity.

For the CS-based scheduling, a Measurement Scheduling Matrix (MSM) is designed that specifies a measurement scheduling policy: it contains a "1" in the (m, n) position, if the m-th measurement is taken at time n. The MSM servers as two purposes: a scheduling policy matrix to turn the sensors on/off as well as being a CS *measurement matrix* that applied on the actual pattern matrix to obtain a measured sequences (i.e., CS measurements). During the training stage, each day, we obtain a projected sequence of the actual pattern matrix through MSM and then, based on the CS recovery algorithm, to reconstruct for the actual pattern matrix. During the prediction stage, the probability

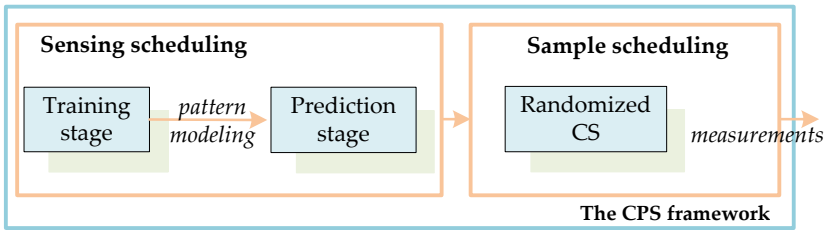


Figure 3.1: The CPS framework.

of sensing vector can be calculated based on the actual pattern matrices which will be discussed in Sec 3.4.2.

In both stages, adaptive update of the probability of sensing vector is conducted based on theory of learning automata. *SenS* then uses the weighted combinations as the final probability of sensing value to make the sensing decisions. The reason why we use two different scheduling approaches will be discussed later in Sec 3.4.2.

The *SamS* component deals with the issue on "*how to sense*" when the sensors are turned on. A method called Randomized-CS (RCS) is developed. Traditionally, CS is achieved by directly projecting the raw activity signal to a lower dimensional signal, referred to as the *measurements* and thus involves a projection matrix which is often called the measurement matrix. When applying CS on the resource-limited smartphone platform, we propose a specifically designed sparse binary measurement matrix, such that only "additions" are performed thus reducing the computational burden placed on the smartphone side during the sensing process. It is designed with the goal that both the computational complexity and sensing energy cost are minimized.

The main contributions of this chapter are three-fold: First, we propose to use activity pattern model and CS-based scheduling to intelligently schedule the sensing task such that sensors can be alleviated from the burden of continuous sensing in order to save energy. Second, we propose a light-weight randomized compressed sensing scheme where instead of using traditional CS measurement matrix, we use the sparse binary measurement matrix for sensing to reduce the energy consumption as well as computation burden of smartphones. Third, we prototype the proposed CPS framework on the Android-based

smartphones and conduct a case study on the driving activity sensing which demonstrates the effectiveness through real-life experiment.

The rest of this chapter is organized as follows. We discuss some related works in Section 3.2. In Section 3.3 we introduce the background of CS. In Section 3.4, we present the Sensing Scheduling (*SenS*) component. In Section 3.5, we elaborate on the sample scheduling *(SamS)* component.

3.2 Related Work

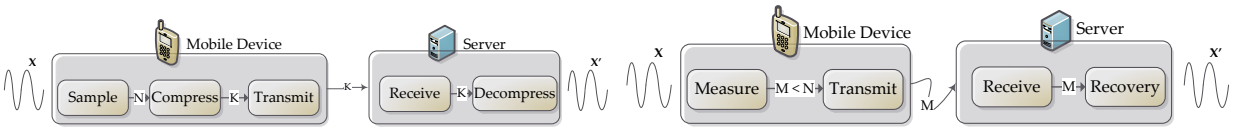
To the best of our knowledge, no other prior work incorporates human activity patterns into smartphone sensing using the compressed sensing technique. CPS, as we have presented, combines human activity models and compressed sensing for energy efficient smartphone sensing. We now describe how some related works have been conducted on each of these separate aspects related to CPS.

The work from Gonzalez et al. Gonzalez et al. (2008) published in *Nature* tried to study the basic laws that govern human motion using smartphones. It turned out that the individual travel patterns collapse into a single spatial probability distribution, indicating that, despite the diversity of their travel history, humans follow simple reproducible patterns. Huynh et al. Huynh et al. (2008) used topic models to enable the automatic discovery of such patterns in a smartphone user's daily routine with results that showed the ability to model and recognize daily routines without user annotation. Darwin Miluzzo et al. (2010) presented a sensing system that combines machine learning techniques to reason about human behavior and context on smartphone where an evolve-pool-collaborate model was designed to provide a foundation for many context and sensing applications. The classifier evolution method can update the models over time such that the classifiers are robust to the variability in sensing, which is similar to our activity model training process. Most recently, a startup called *Behav.io* behavio (2012), tried to turn smartphones into smart sensors of people's behaviors and surroundings, such that the smartphone will be able to understand trends and behavioral changes in individuals as well as entire community. In

Nawaz and Mascolo (2014), the authors tried to mine a user's significant driving routes by time warping angular speeds solely from a smartphone's gyroscope and accelerometer signals. By avoiding the energy-hungry GPS sensor, it achieves energy savings of an order of magnitude over the GPS sensor.

Regardless of the various perspectives of smartphone sensing, the biggest challenge is on battery life. Little Rock Priyantha et al. (2011b) took a hardware approach to solve this problem where a new sensing architecture is proposed to offload sampling and, when possible, processing of sensor data to a dedicated low-power process. Zhuang et al. (2010) proposed an adaptive location sensing framework for location based services with four design principles, substitution, suppression, piggybacking and adaptation, where smartphone battery can be saved through smart utilization of these four design principles into the applications. ACE (Acquisitional Context Engine) Nath (2012) exploited the relationships among various context attributes for efficient and continuous sensing of user's context in a smartphone. Liu et al. (2012) addressed the problem of energy consumption in GPS receiving by splitting the GPS location sensing into a device part and a cloud offloading part to assist various smartphone applications that require location sensing. Similar to ACE, Jiang et al. proposed `CARLOG` Jiang et al. (2014), a programming system for automotive apps. `CARLOG` allowed programmers to succinctly express fusion of vehicle sensor and cloud information, a capability that can be used to detect events in automotive settings. It contains novel optimization algorithms designed to minimize the cost of predicate acquisition.

Yang et al. Yang and Gerla (2010) applied compressed sensing framework to the wireless accelerometer data transfer system, where results confirmed the feasibility that the CS framework can reduce energy consumption more than the traditional compression methods without increasing the implementation complexity in various real applications. In Akimura et al. (2011), CS was used for human activity accelerometer data compression at the mobile side and sending a minimum amount of data over the wireless link for recovery at the server side. This reduced power consumption by 16% as compared to the ZIP compression scheme.



(a) Traditional "sensing and compression" scheme (b) Compressed phone sensing scheme

Figure 3.2: Comparison of two mobile sensing schemes.

3.3 Background on Compressed Sensing (CS)

The fundamental idea behind CS is that rather than first sampling at a high rate and then compressing the sampled data, one could directly sense the data in a compressed form, i.e., at a low sampling rate. Figure 3.2 shows how the traditional "Sensing and Compression" scheme and the "Compressed Sensing" scheme work in the mobile sensing scenario. Let x be the activity signal of interest, e.g., accelerometer, gyroscope or magnetic signal. Assume that N units of data is necessary for the analysis. In the traditional case, the smartphone sensor is required to record N units of data. Then the mobile device may compress and transmit $K, K < N$, units of data to the server in order to save energy. In the CS case, instead of capturing N units of data, only $M \ll N$ units of data is sampled, which is then directly sent to the server. The server side reconstructs the data to its original signal x. The quality of the reconstructed signal x' is closely related to (1) the sparsity of the original signal, (2) the size of M (i.e., the number of measurements), and (3) the reconstruction method.

3.3.1 Sparsity

In theory, CS works well on sparse signals. A signal is said to be $K\text{-}sparse$ if the signal is expressible with a linear combination of the basis functions. Let $\Psi = \{\psi_1, \psi_2, \cdots, \psi_N\}$ be a set of basis vectors, then the activity signal x of length N can be represented as:

$$x = \sum_{i=1}^{K} c_i \psi_i, \tag{3.1}$$

where $K \ll N$ and c_i is the coefficient with respect to the basis Ψ. Typically, we will be dealing with signals that are not themselves sparse but would admit a sparse representation in some basis Ψ. In this case, we still refer to x as being $K\text{-}sparse$, with the understanding that we can express x as $x = \Psi c$, where $\|c\|_0 \leq K$ ($\|\cdot\|_0$ indicates the number of nonzero entries in a vector). It has been demonstrated in Yang and Gerla (2010); Akimura et al. (2011) that the sensed daily walking signals and the human body movement accelerometer signals possess the sparse nature.

3.3.2 Measurements

According to the CS theory, a non-adaptive $M \times N$ *measurement matrix* Φ can be explicitly constructed. We directly acquire the measurements y, which is obtained by projecting the original signal x on to the measurement matrix Φ. Note that, we use the original signal x only for measurement projection purpose and will not store or transmit it, only the measurements y of length M will then be sent to the server. This is expressed in Eq. (3.2).

$$\begin{aligned} x &= \Psi c \\ y &= \Phi x = \Phi \Psi c \end{aligned} \tag{3.2}$$

Here, $x \in \mathbb{R}^{N\times 1}$ is the original activity signal, it is sparse represented under basis matrix $\Psi \in \mathbb{R}^{N\times N}$ with coefficients $c \in \mathbb{R}^{N\times 1}$. In the measurement process, Φ should be incoherent with Ψ. This incoherence requires that the rows of Φ cannot sparsely represent the columns of Ψ (and vice versa). However, the incoherence can be achieved with high probability simply by selecting Φ as a random matrix with Gaussian entries or scrambled Fourier coefficients Baraniuk (2007). There is also research on using sparse measurement matrix Berinde et al. (2008); Gilbert and Indyk (2010).

3.3.3 Reconstruction

The original signal x, can be recovered or reconstructed using the observation vector or the measurement, $y = \Phi x$. Often, signal x can be recovered as a solution for the following ℓ_1-norm minimization problem by linear programming.

$$x^* = \arg\min_{x'} \|x\|_1 \qquad \text{s.t.} \qquad y = \Phi x' \tag{3.3}$$

Eq. (3.3) is usually known as the *Basis Pursuit (BP)* problem Chen et al. (1998), which is a convex optimization problem and can be solved easily by many software packages (e.g., ℓ_1-magic Candes and Romberg (2005)).

3.4 Sensing Scheduling (*SenS*)

In this section, we elaborate on the Sensing Scheduling (*SenS*) component in the CPS framework. We first define the pattern-based activity followed by the probability of sensing vector that associated with an activity. Then we discuss and compare the two approaches of *SenS* as well as how the probability of sensing vector is initialized in both the training and the prediction stage in the *SenS* component.

3.4.1 Definitions

We now discuss the definition of pattern-based activity and its modeling as well as the mathematical definition of sensing scheduling problem. We then introduce the concept of the probability of sensing.

Pattern-based Activity

We define the *human activity* (e.g., walking, driving, jogging) as a sequence of meaningful actions Nardi (1996); Ryder (2012); Tian et al. (2011), intended to achieve certain goals. We assume an activity is different from an action or operation, where the latter usually

lasts for very short duration while an activity generally consists of a sequence of actions. *Activity pattern* is related to an individual's activity but different from the activity itself. We define that an activity is pattern-based if: (1) it is a frequently occurring event, and (2) it tends to have certain characteristics over a long period of time. For example, for most people in the US, driving is a daily activity. However, the time one drives and how long the driving activity takes place are different from person to person. This is one's driving pattern.

In order for the smartphone to be able to predict the activity and thus perform sensing tasks, two pieces of information are essential:

The granularity of an activity: We assume an activity is meaningful if its duration is longer than certain period of time. In *SenS*, we define this period as 5 minutes. Accordingly, we divide a day into 5-minute time slots, yielding totally $\frac{24\times 60}{5} = 288$ time slots.

The temporal information of the previous activity: The temporal information of the previous activity consists of two components, including the time the previous activity starts and the duration of the activity. This is very useful for building the activity model and predicting future activities.

Here, we use a so-called *Pattern Matrix (PM)* to incorporate both the above information.

Definition 1 (Pattern Matrix)**.** *A Pattern Matrix is a binary matrix with each column j representing a different day and each row $i, (i = 1, \cdots, 288)$, representing a time slot during that day, such that,*

$$PM(i,j) = \begin{cases} 0 & \textit{the activity does not occur} \\ 1 & \textit{the activity occurs}. \end{cases}$$

Therefore, each column of the PM records the temporal information of an activity happened during that day using binary indicators. After certain amount of days (referred to as the *training stage* in *SenS*), the PM will be expanded with multiple columns. We can then visualize the PM in the sense of histogram of the activity occurrence with the *x-axis*

as the time slots (bins) over a day and the *y-axis* as the total number of activity occurrences (or frequencies) over a number of days, as shown in Fig. 3.3.

Sensing Scheduling Problem Definition

We first begin by defining mathematically the sensing scheduling problem and then introduce the concept of "probability of sensing" in our two *SenS* approaches.

Denote $\mathbf{x} = \{x_t, t = 1, 2, \cdots, N\}$ as an actual realization of the activity sensing process. A *measurement policy* is given by a sequence of sampling times: $T^\pi = \{t_1, t_2, \cdots, t_n\} \in \{1, 2, \cdots, N\}$. Assuming perfect measurements (no error or noise), this policy induces the following *sampled sequence* $\mathbf{x}^\pi = \{x_{t_1}, x_{t_2}, \cdots, x_{t_n}\}$. An *estimation policy* λ then takes this sampled sequence and produces estimates of the original sequence $\hat{\mathbf{x}}^\lambda = \{\hat{x}_t, t = 1, 2, \cdots, N\}$, where $\hat{x}_t = x_t$ if $t \in T^\pi$, and $\hat{x}_t = \hat{x}^\lambda(\mathbf{x}^\pi)$ otherwise, for some estimation function $\hat{x}_t^\lambda()$.

The objective is to select the best measurement and estimation policies so as to minimize the estimation error subject to a requirement on the average sampling rate being no more than a certain desired level:

$$\min_{\pi,\lambda} \text{Err}(\mathbf{x}, \hat{\mathbf{x}}^\lambda(\mathbf{x}^\pi)) \text{ s.t. } n/N \leq r, \tag{3.4}$$

where $\text{Err}(\cdot)$ is certain error measure, e.g., the mean-squared error, and r is the requirement on sampling or measurement rate.

The Probability of Sensing

We take a probability approach in the realization of the *measurement policy* and introduce the *probability of sensing*. At the sample time t_i, the value of P_{t_i} indicates the probability of taking action (i.e., sensing) based on certain threshold τ. When a sensor conducts sensing and results in actually capturing an activity of interests, it is considered a *success*, otherwise, a *failure*. We denote here the probability of sensing as P_i where i is the time slot of a day throughout the rest of the chapter.

3.4.2 Two approaches of *SenS*

In this section, we describe and then compare the two approaches of *SenS*, i.e., model-based scheduling and CS-based scheduling, and how the probability of sensing vector is initialized in these two approaches during the training stage and the prediction stages.

Model-based Scheduling

For model-based scheduling, we propose to use the Gaussian Mixture Model (GMM) to model the activity pattern. Generally speaking, the user tends to perform the activity around one particular time of the day and then the frequency of occurrence gradually decreases as the time of the activity deviates from that favorite time. Thus, we make a hypothesis that the activity occurrence time has a normal distribution $N(\mu_W, \sigma_W^2)$ where μ_W is the mean and σ_W^2 is the variance which can be calculated by Eqs. (3.5) and (3.6), respectively.

$$\mu_W = \frac{1}{T}\sum_{i=1}^{T}\omega(i) \tag{3.5}$$

$$\sigma_W^2 = \frac{1}{T}\sum_{i=1}^{T}(\omega(i) - \mu_W)^2 \tag{3.6}$$

where the random variable W represents the occurring time slot of an activity, T is the total number of time slots in a day ($T = 288$), and $\omega(i)$ is the total number of the activity occurred at the i^{th} time slot. The estimated probability density function (pdf) of W is given by Eq. (3.7).

$$g(\omega|\mu_W, \sigma_W^2) = \frac{1}{\sqrt{2\pi\sigma_W^2}}e^{-(\omega-\mu_W)^2/2\sigma_W^2} \tag{3.7}$$

When taking into account the day variances over one's activity behavior, we model this kind of activity as a Gaussian mixture model, shown in Eq. (3.8).

$$p(\omega) = \sum_{n=1}^{M} c_n g(\omega|\mu_{W_n}, \sigma_{W_n}^2) \tag{3.8}$$

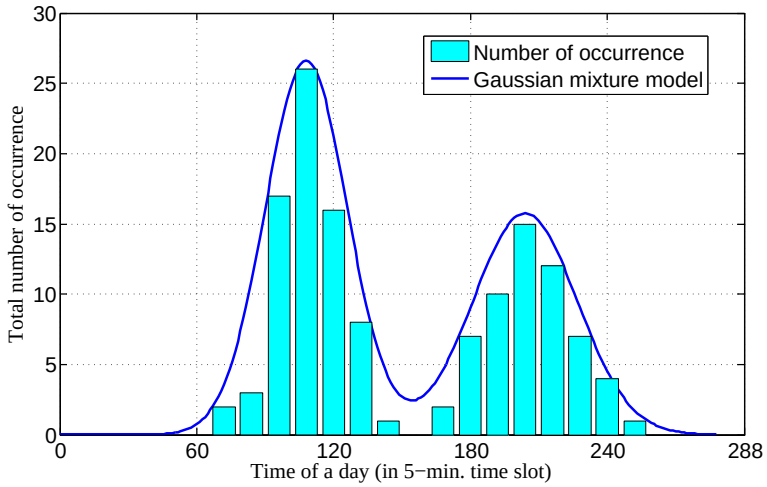


Figure 3.3: An example of the driving activity pattern whose total number of occurrence is modeled using a Gaussian mixture model.

where M is the number of components (or modes), $g(\omega|\mu_{W_n}, \sigma^2_{W_n})$ is the component Gaussian density, and $c_n, n = 1, \cdots, M$, is the mixing weight, satisfying the constraint that $\sum_{n=1}^{M} c_n = 1$. Figure 3.3 shows an example of the driving pattern whose time of occurrence is fitted with the Gaussian mixture model.

Based on the GMM model, the probability of sensing value of an activity taken place between the i^{th} and the $(i+1)^{th}$ time slot is given by

$$P_i^{Model} = Pr\{i \leq W \leq i+1\} = \int_i^{i+1} p(\omega)\, \mathrm{d}\omega \tag{3.9}$$

where $p(\omega)$ is the Gaussian mixture model from Eq. (3.8).

CS-based Scheduling

Recall that each column of the PM records the temporal information of an activity happened during that day using binary indicators. The PM vector is generally sparse which makes CS-based scheduling plausible as CS requires that the signal to be measured is sparse under certain basis. CS-based scheduling tries to realize the scheduling process through a measurement scheduling policy based on a so-called measurement scheduling matrix

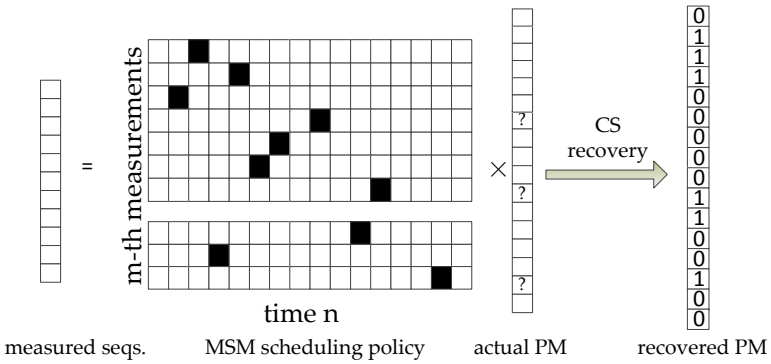


Figure 3.4: An illustration of the CS-based scheduling policy. Dark cells indicate when the sensing process takes place.

(MSM) that contains a "1" in the (m, n) position, if the m-th measurement is taken at time n. The physical nature of the instrument is such that only a single measurement is taken at any scheduled time, i.e., upon actuation, the sensor takes one measurement of the activity signal at the time of actuation; the same point in the process cannot be measured more than once due to causality. This implies, regardless of the schedule, MSM contains one and only one "1" element in any row, and at most one "1" in any column, and "0" everywhere else. In *SenS*, the MSM is realized based on randomization which is generated using certain probability distribution with an average sampling rate of $\frac{m}{n}$. The MSM further servers as two purposes: a scheduling policy matrix to turn the sensors on/off as well as being a CS measurement matrix that applied on the actual pattern matrix to obtain a measured sequences (i.e., CS measurements). Fig. 3.4 illustrates the scheduling process when the MSM is applied on the actual PM.

During the training stage, each day we will obtain a measurement which is the measured sequence of the actual PM. We then reconstruct the PM vector using CS recovery algorithm (e.g., ℓ_1-magic Candes and Romberg (2005)). Denote the reconstructed PM record vector of each day as $\mathcal{A}_i, i = \{1, 2, \cdots, 288\}$. Then during the prediction stage, the probability of sensing values is calculated using $P_i^{CS} = \frac{c \times \mathcal{A}_i}{\sum_{i=1}^{288} \mathcal{A}_i}$, where c is a scalar.

Model-based vs. CS-based Scheduling

We explain here the reason why we adopt two different approaches in *SenS*. The problem of activity sensing belongs to the larger class of sensor scheduling problems. There are two general approaches. The first is a closed-loop approach that makes a measurement decision using past observations and decisions. This typically requires the knowledge of prior statistics of the underlying random process to be monitored, gained either through assumption or training, and is also sometimes referred to as the Bayesian approach, for example Meier et al. (1967); Evans and Krishnamurthy (2001); Li et al. (2009b). Our model-based scheduling falls into this category. The second is an open-loop approach whereby measurement decisions are made independent of past observations and decisions (e.g., Wu and Liu (2012)). CS based measurement falls under this category.

Since our model-based scheduling uses GMM which will captures the underlying pattern of a subject's activity, the model usually falls short when some "outlier" activities took place. For example, a person may get sick and need to drive to hospital in the midnight. This driving activity might be missed in model-based PM records. While this activity could be easily identified by CS-based scheduling. In *SenS*, we use the weighted "probability of sensing" output from both the model-based and CS-based scheduling.

Training and Prediction Stage

We explain here the details of the two stages: *training stage* and *prediction stage* in the *SenS*.

During the training state of the model-based scheduling, the probability of sensing vector, P_i^{Model}, $(i = 1, \cdots, 288)$, is *randomly* initialized at the beginning of each day. We set $P_i^{Model} \in [0.1, 0.9]$. We limit the lower bound to 0.1 to avoid very small sampling chance, which would potentially lead to missing an activity. We also limit the upper bound to avoid a too aggressive sampling, which would reduce the battery life. In addition, based on the common sense that human beings are more active during day time, we place higher P_i^{Model} values in the daytime slots (i.e., non-uniform randomization). After this random

initialization, when P_i^{Model} exceeds a threshold τ at a certain time slot i, the sensor will start the sensing process. The result of this sensing, either *success* or *failure* will, first, be used to update the probability of sensing values of the subsequent time slots as detailed in Sec. 3.4.3. It will then be used to update the PM value to 1 (*success*) or 0 (*failure*) at the time slot i. Further, if the sensing activity is a success, that is, the activity of interest is actually happening, then the *SamS* component will be activated for activity sensing.

For CS-based scheduling, the MSM will be randomly initialized each day to perform sensing tasks. The measurement then could be obtained at the end of the day and will be used to reconstruct the estimated PM records accordingly. Note that during the training stage, it is highly possible that certain time slots will be activated in both Model-based and CS-based scheduling policy. In other words, the total number of time slots that will be activated to perform sensing task will not exceed the total number of time slots defined, i.e., 288.

At the end of the training stage, the PM will be used to construct the Gaussian mixture model using the Expectation Maximization (EM) algorithm Dempster et al. (1977). Also, the reconstructed PM from CS-based scheduling will be used to generate the average probability of sensing, P_i^{CS}. That is, after each reconstructed PM vector is obtained at the end of the day, P_i^{CS} will be calculated as discussed in Sec. 3.4.2. Then the P_i^{CS} from all training days will be averaged to generate the final P_i^{CS}.

During the prediction stage, instead of randomly generating the probability of sensing vector, P_i, the probability is initialized based on the weighted GMM model constructed from the training stage using Eq. (3.9) and the CS reconstructed outputs as shown in Eq. (3.10), where γ controls the weight.

$$P_i = \gamma P_i^{Model} + (1 - \gamma) P_i^{CS} \tag{3.10}$$

It is hypothesized that the P_i vector initialized based on GMM and CS reconstructed output have a higher success rate than random initialization as in the training stage. This will be validated through comprehensive evaluations in Sec. 4.2. In both stages, the

probability of sensing value is adapted based on the theory of learning automata, as will be discussed in the following.

3.4.3 Sensing Adaptation

The *SenS* component uses a learning technique based on the theory of learning automata to control the on/off of the smartphone sensors. In particular, we use the *linear reward-inaction* Kaelbling et al. (1996) algorithm. Learning automata based techniques are defined in terms of *actions*, *probability* of taking these actions, and their resulting *success* or *failure*. In *SenS*, the only action taken is sensing from a sensor. The decision whether to sense or not at a time slot i is based on the probability of sensing, P_i, which is thus dynamically adjusted according to their previous success or failure action, as formulated below:

$$P_\Lambda = \begin{cases} P_\Lambda + \alpha(1 - P_\Lambda) & \text{action is a success} \\ P_\Lambda - \alpha P_\Lambda & \text{action is a failure} \end{cases} \tag{3.11}$$

where Λ is the index set ranging from $[i, i+1, \cdots, i+n]$, P_Λ denotes each of the n probability of sensing values after P_i, $\alpha \in [0, 1]$ controls the rate of the probability updates and n is the number of subsequent probability values affected by P_i.

3.5 Sample Scheduling (*SamS*)

When *SenS* decides to turn on the sensor (i.e., the probability of sensing is over certain threshold τ), the *SamS* component will be activated to perform the actual sensing task. Three strategies are integrated for light-weight sensing without degrading the fidelity of the signal. First of all, the theory of compressed sensing is utilized to directly project (through a so-called measurement matrix) the raw signal to a lower-dimensional measurement signal such that significantly fewer samples need to be collected and transmitted without any additional compression process. Second, a sparse binary measurement matrix is developed

that reduces the projection from involving the matrix-vector multiplication to only addition operations. This significantly reduces the energy consumption related to computation on smartphones. Finally, a randomized CS scheme is proposed that randomly selects a short "on" period during the 5-minute activation time slot, to further reduce the energy consumed during sensing.

3.5.1 Sparse Binary Measurement Matrices

From Section 3.3.2, we see that a matrix-vector multiplication would be needed for generating measurement signal y for recovery. Under a high sampling rate, a short duration of sensing would consume a lot of computational resource on the smartphone. Instead of using a traditional way of constructing dense random measurement matrix, e.g., random Gaussian matrices or scrambled Fourier matrices, *SamS* uses a sparse binary measurement matrix. The benefit is that it reduces the matrix-vector multiplication to only addition operations on smartphones.

The sparse binary measurement matrix used in the CPS framework is based on the adjacency matrix of the high-quality expander graph, with guaranteed CS recovery Berinde et al. (2008). That is, the recovered signal x^* satisfies $\|x-x^*\|_2 \leq C \min_{x'} \|x-x'\|_2$, where x' ranges over all $K\text{-}sparse$ vectors for any constant C.

A sparse binary measurement matrix Φ of M rows and N columns is generated in two steps:

- **Step 1:** For each column, randomly generate d integers whose values are between 1 and M and place $1's$ in those rows indexed by the d integers,
- **Step 2:** If the d integers are not distinct, repeat the first step until they are (this is not really an issue when $d \ll M$).

With some proper value of d (e.g., $d = 8$ in our experiments), we see that such matrix is the adjacency matrix of an expander graph of degree d with high probability. For detailed information, see Berinde et al. (2008); Li and Qi (2013a).

3.5.2 Randomized CS

Upon activating the sensors within a 5-minute time slot, instead of keeping the sensors on for the entire 5 minutes, in order to further save energy, CPS chooses to randomly start the sensors and keep them active for a small amount of time t where $t < 5$ min), which we refer to as the randomized CS. The smaller the t, the more energy saved.

When the sensing scheduling component decides a certain 5-minute time slot should be "on" (i.e., active sensing), *SamS* randomly picks a *t*-minute interval when the CS measurements are actually generated by projecting the raw sensor signals to the binary measurement matrix.

Chapter 4

Case study: Driving Activity Sensing

In this chapter, we conduct a case study on driving activity sensing to evaluate the performance of the P-CPS framework. The P-CPS software architecture and implementation issues are provided and explained in Section 4.1. In Section 4.2, we discuss the experimental results and the overhead of P-CPS. Finally, Section 4.3 concludes this chapter.

4.1 Software Architecture and Implementation

We have implemented the proposed CPS framework on a Nexus S Android Developer Phone with OS version 4.1.1 Jelly Bean. All the application interfaces and service components are implemented in Java inside the Android framework. We have chosen Driving Activity Sensing (DAS) as a case study to demonstrate the effectiveness of CPS in energy saving. Although the software architecture and implementation is based on DAS, extensions to other applications can be easily done.

4.1.1 Background and Motivation

New York Times has reported a growing phenomenon of "user-based insurance", in which your insurance rates are based on your actual driving behavior. By installing a small wireless gadget in the car, the user's driving data will be collected and sent to an insurance

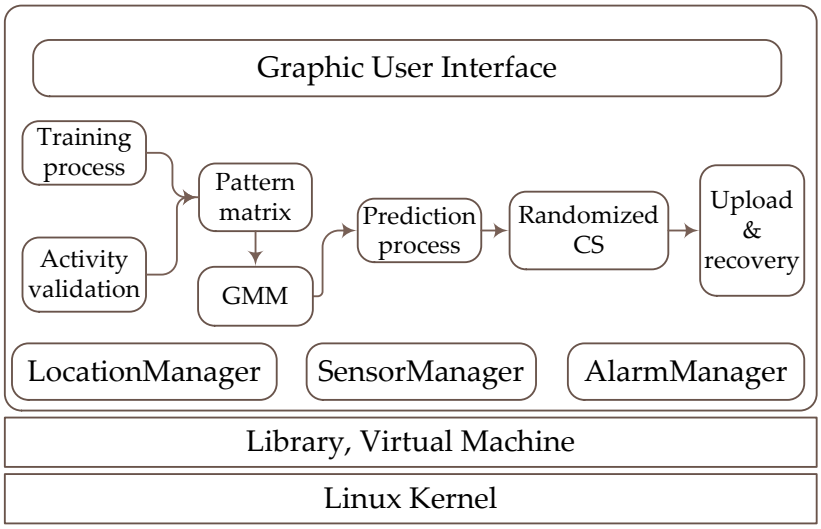


Figure 4.1: A typical software architecture of the CPS framework based on the Android platform.

company and analyzed. More drivers, seeking discounts on auto insurance, are voluntarily doing this Times (2012). Motivated by this phenomenon and also taking advantage of the rich set of sensors on the smartphone, we conduct a case study on driving activity sensing (DAS) using smartphones. DAS utilizes two built-in sensors generally available on a commercial smartphone, the accelerometer sensor and the GPS sensor. Correspondingly, two types of driving activity signals are collected, the accelerometer signal associated with the accelerometer sensor and the location signal (i.e., latitude and longitude coordinates) associated with the GPS sensor. Note that DAS can also capture the speed information through the GPS sensor. This speed data is used most often for analyzing the "user-based insurance". Generally speaking, the accelerometer sensor consumes much less energy as compared to the GPS sensor, which is a power hungry sensor and could reduce the battery life to few hours when turned on. We intentionally select these two sensors with different energy consumption characteristics to evaluate the proposed CPS framework.

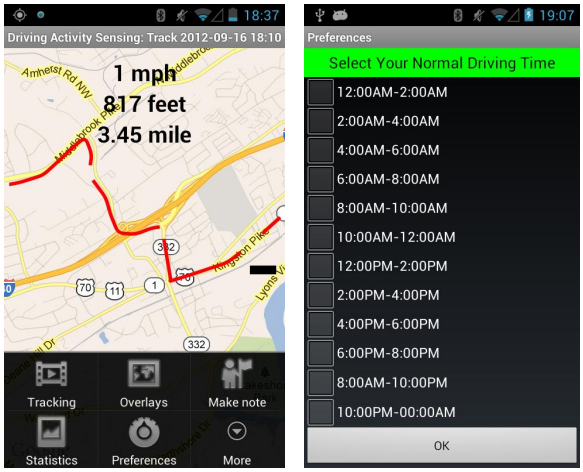


Figure 4.2: The DAS GPS map view (left) and driving preference input (right).

4.1.2 Software Architecture

The software architecture of the DAS application is illustrated in Fig. 4.1. There are three layers in the software architecture. At the bottom is the Android SDKs and Kernel. The proposed CPS framework serves in the middle layer. The two major components of CPS, Sensing Scheduling (*SenS*) and Sample Scheduling (*SamS*) are both implemented as the service components and work closely with several existing components such as *LocationManager*, *SensorManager* and *AlarmManager* in the Android Framework. At the top, DAS contains the Graphic User Interface (GUI) which allows users to input their preferred driving time periods and to view their existing driving routines. Figure 4.2 shows the interface, where the map view is implemented using Google Map APIs. The selected daily driving time is used as the activity preference. Note that, in general, CPS does not need any user input during the training stage. This activity preference can be used in order to expedite the training stage, where the first column of the PM is initialized randomly but with a high probability of sensing values within these selected time slots.

4.1.3 Implementations Issues

There are four major issues related to the CPS framework implementation of DAS, namely, how to construct the GMM, how to validate if the driving activity is actually occurring, how

to handle the daily prediction failures, and how to upload the measurements and perform activity signal recovery.

GMM estimation: We implement the GMM estimation process on a server using the Expectation Maximization (EM) algorithm. The performance of the EM algorithm can be highly dependent on how the algorithm is initialized. Although the best method to use for initialization remains a somewhat controversial topic, in DAS, we use random initialization. We set here the number of modes in GMM equal to 3 to avoid a too complicated model for DAS.

Driving activity validation: To validate if a driving activity is actually occurring during one of the active sensing time slots is not a trivial task. Existing algorithms take accelerometer and GPS data as input, and determine whether or not the user is currently in a vehicle Reddy et al. (2010). DAS adopts the idea of using low-energy sensors to trigger more power-hungry sensors. Several sources of data have been used to validate the driving activity in DAS, including accelerometer, RF radios (e.g., WiFi, Bluetooth), and GPS. DAS first uses the low power sensor, i.e., accelerometer, to detect the motion. For every new accelerometer sample, we compute the standard deviation of the magnitude of the acceleration over a sliding window of w samples. The window slides one sample at a time. If the standard deviation in a window exceeds a threshold a, a movement has been detected. When the standard deviation is within the threshold for s consecutive sliding windows, the smartphone is thus stationary. Based on the heuristic experiments in Ravindranath et al. (2011), we manually set $w = 5, a = 0.15\ m/s^2$, and $s = 10$. We also listen to the possible network changes (i.e., with/without WiFi connections, cell tower changes, etc.) through Android *BroadcastReceiver*. Often, the smartphone network changes would indicate possibly physical location changes of the user, e.g., driving. If any changes were detected, the smartphone will use Bluetooth sensor to scan the nearby devices. We assume that the user's smartphone has already been paired with the vehicle. If the car Bluetooth device was found by the smartphone scanning, the power-hungry GPS sensor will be turned on and further speed and location information will be captured. If the

speed is greater than 0 or if there is any location change during the 1-minute sensing time, the driving activity validation will return true.

Handling daily prediction failures: Due to the randomness in the CPS training stage where random initialization is performed to obtain the probability of sensing values, the prediction failure cases are unavoidable. Three metrics are calculated each day in order to better handle daily prediction failures, including accuracy, positive predictive value (PPV), and specificity. *Accuracy* measures the percentage of the number of successfully sensed driving activities over the actual total number of driving occurrences. *Positive predictive value* measures the percentage of the number of successfully sensed driving activities over the total number of driving validations (i.e., the number of times when P_i exceeds the threshold τ). *Specificity* measures the percentage of the actual total number of driving occurrences over the total number of driving validations. It is the PPV divided by accuracy. While the accuracy is related to the initialization of P_i, PPV is related to the threshold τ that applied on the probability of sensing values P_i. If PPV is too small, we need to increase the value τ accordingly and vice versa. The specificity value could be either between zero and one or greater than one. Ideally, we want the specificity value to be close to one. If it is larger than one, we should lower the threshold τ and vice versa.

Uploading measurements and performing activity signal recovery: In DAS, the smartphone uploads the sensed driving activity signals (i.e., CPS measurements) to a dedicated sever. On the sever side, we use the popular ℓ_1-magic Candes and Romberg (2005) MATLAB package for CS recovery used in both *SenS* and *SamS*..

4.2 Experimental Results

In this section, we evaluate the CPS framework based on our case study on the DAS application. We first discuss the experimental data as well as the performance metrics, followed by the detailed results and discussions.

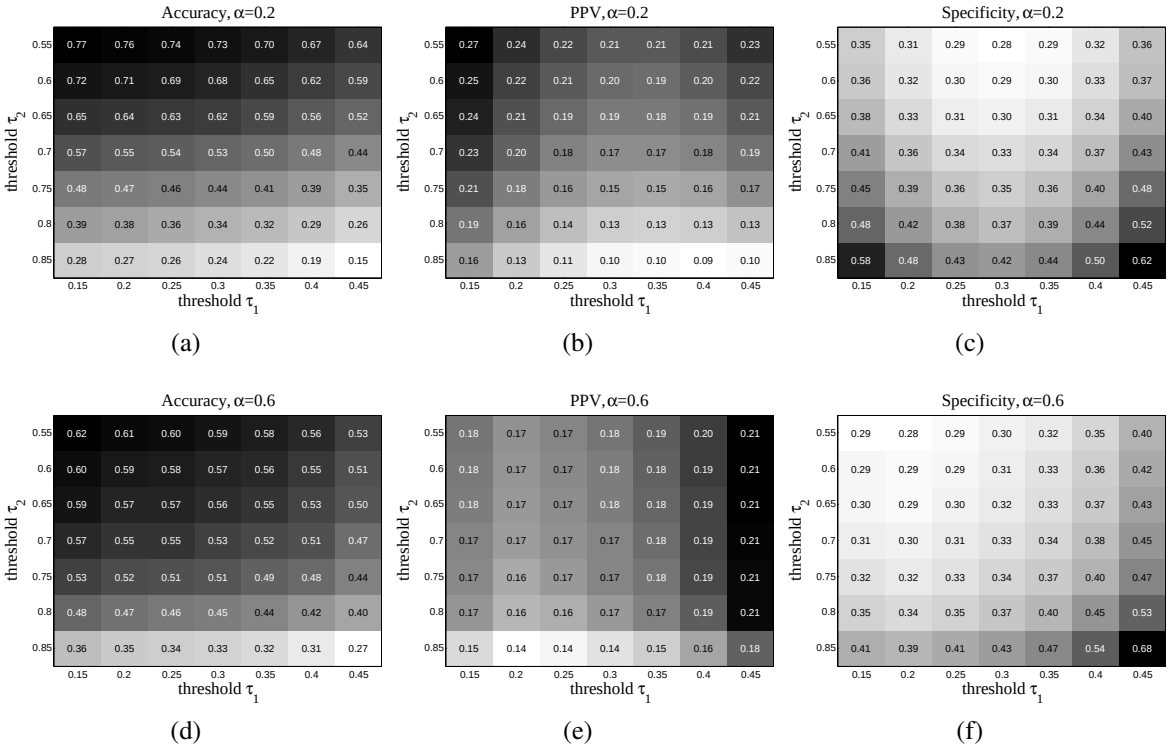


Figure 4.3: Averaged training stage accuracy, PPV and specificity values when $\alpha = 0.2$ (upper) and $\alpha = 0.6$ (lower).

4.2.1 The DAS Data and Performance Metrics

The data used in the experiments were collected by the DAS application from 6 volunteers across various time frames (e.g., school time, weekend, summer time, etc.). Among the 6 volunteers, four of whom are college students, one is self-employed, and one office employee. The data collection process lasts from 4 months to 8 months. We set the DAS training stage as 3 months and treat whatever remaining days as the prediction stage.

We use three metrics to evaluate the effectiveness of the proposed CPS framework: *accuracy*, *energy*, and *recovery error*. The accuracy and the related PPV and specificity follow the same definition as in Sec. 4.1.3. The energy consumption is the amount of energy consumed by CPS. For evaluation purpose, we compare the energy consumption of the accelerometer sensor using three sensing schemes, (1) continuous sensing, (2) *SenS* without *SamS*, and (3) *SenS* with *SamS*, i.e., the the proposed CPS framework. In *SenS*, we further

evaluate the effect of model-based vs. CS-based scheduling. We make certain that the mobile phone is under the same baseline energy consumption (i.e., under the same settings, sensors and applications). The recovery error is measured based on the Normalized Root Mean Square Error (NRMSE) of the GPS signal. We show the recovery performance at the server side and discuss the recovered GPS tracks. The overhead of the CPS framework is also analyzed at the end of this section.

Accuracy

Training stage: We first experiment on how the value of α, the number n in Eq. (3.11) as well as threshold τ will affect the accuracy of DAS in the training stage. Based on the simple heuristic that human beings are more active during the day time, we randomly initialize the P_i values to be in the range $[0.1, 0.5]$ for the night time slots (i.e., $i \in [1, 72]$ and $i \in [253, 288]$) and $[0.5, 0.9]$ for the day time slots (i.e., $i \in [73, 252]$). Accordingly, we choose τ_1 and τ_2 as the threshold for the night time and day time threshold.

We show in Fig. 4.3 the accuracy, positive prediction values (PPV) and the specificity values with τ_1 ranging from 0.15 to 0.45 and τ_2 ranging from 0.55 and 0.85 while α equal to 0.2 and 0.6, respectively. We observe that DAS gets higher accuracy when the τ_1 and τ_2 values become closer to their lower bound. Also smaller α value gives higher accuracy and PPV. For the specificity, the best performances happened either when τ_1 is at its lower bound and τ_2 at its upper bound or both at their upper bounds. It is true since the specificity value is the PPV divided by accuracy.

We then choose evenly nine pairs of (τ_1, τ_2) where τ_1 ranges between 0.15 and 0.45, and, τ_2 0.55 to 0.85. We set $\alpha = 0.2$ in the training stage. Also for the neighborhood number n, which is related to the DAS driving time, we test various values and set $n_1 = 3$ during night time and $n_2 = 5$ during day time which gives the best accuracy. The weighted probability sensing parameter γ is set as 0.8. Figure 4.4 shows the false alarm ratio versus accuracy (i.e., detection ratio) given the nine pairs of threshold values, where the false alarm ratio is defined as the number of false detected driving activities over the

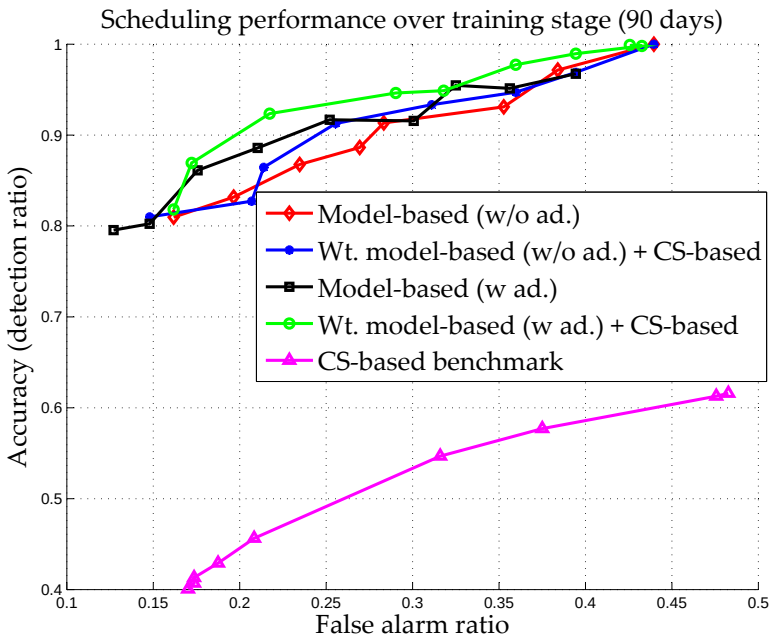


Figure 4.4: The false alarm ratio versus accuracy during training stage. The approaches compared are model-based scheduling with sensing adaptation (w ad.) and without sensing adaptation (w/o ad.) as well as weighted (Wt.) model-based and CS-based scheduling with adaptation. The CS-based scheduling alone is shown as a benchmark.

total number of time slots (i.e., 288). From the result, we see that under the same false alarm ratio, the combination of weighted model-based and CS-based scheduling with sensing adaptation gives the best accuracy. Sensing adaptation also improves the accuracy when compared using the model-based scheduling. The CS-based scheduling alone is shown as a benchmark which gives the worst result as CS-based scheduling excels in capturing "outliers" activity only.

Prediction stage: In the prediction stage, we first model the total number of driving occurrences versus time of day over the training stage as a 3-component Gaussian mixture. In Figure 4.5, we show the generated three Gaussian mixtures based on PM using the EM algorithm. Then in the prediction stage, each day we initialize the probability of sensing values P_i based on the combination of GMM and CS-based scheduling output. We also set the γ value as 0.8 which gives us the best result. One example of these probability values is plotted over the time of day in Figure 4.6. The comparison of different combinations of the *SenS* approaches is shown in Figure 4.7. We also experiment on all the 6 subjects'

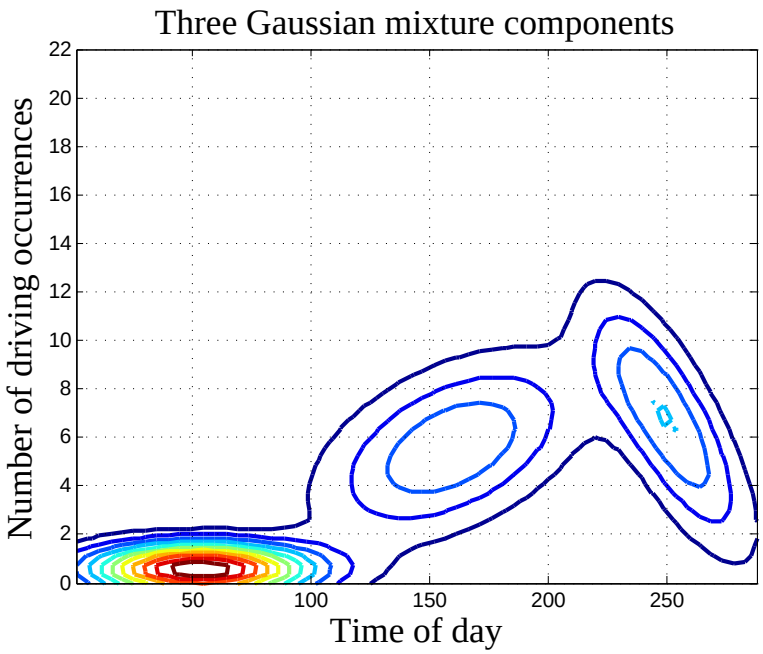


Figure 4.5: Generated 3 Gaussian mixtures using the training stage PM.

data. Table 4.1 lists the accuracy during the training and the prediction stages. We select the threshold parameters τ_1, τ_2 such that the false alarm ratio is under 0.2 and set $\alpha = 0.2$. We set the number of GMM components as 3 and using random initialization on the EM algorithm as well as random initialization of the CS-based *MSM*. We set the neighborhood number $n_1 = 3$ and $n_2 = 5$ for both the training and the prediction stage. We see that the highest accuracy during prediction stage can reach as high as 92.33% with averaged accuracy being 83.92%.

Energy

The energy consumption of the DAS application is evaluated using the Android *Battery-Manager* API. We record the battery level information every 10 minutes during each day of the DAS experiments. Figure 4.8 demonstrates the smartphone energy consumption using different sensing schemes, where we observe that the weighted model-based and CS-based scheduling with adaptation can save 62.86% of energy as compared to continuous sensing. The model-based scheduling scheme with sensing adaptation is the most energy efficient

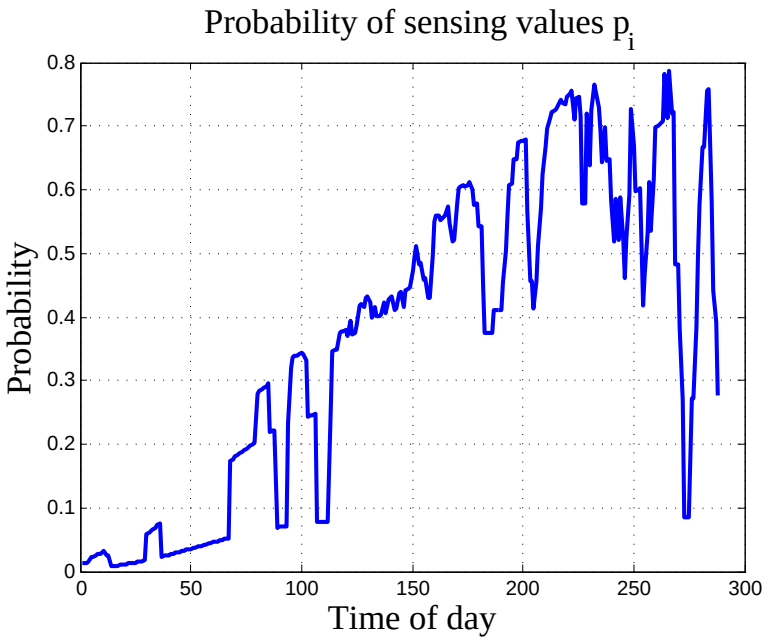


Figure 4.6: The probability of sensing values P_i.

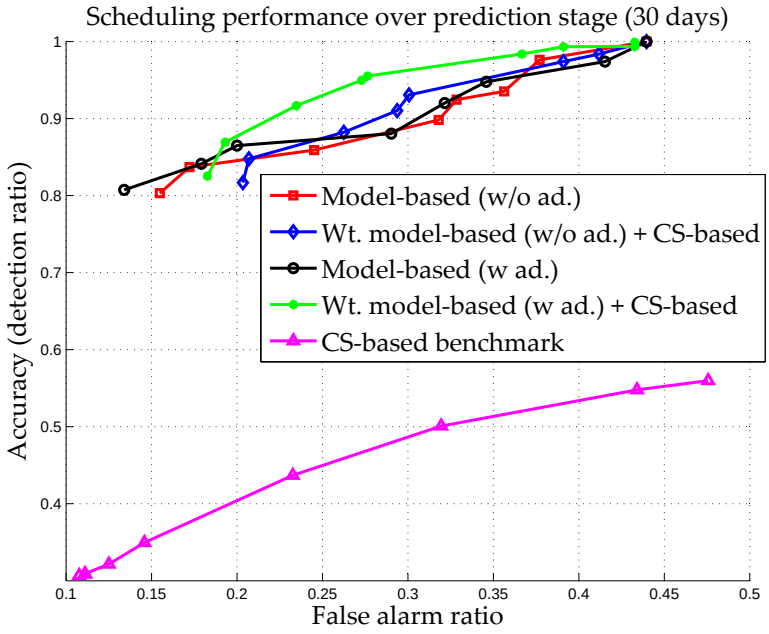


Figure 4.7: The false alarm ratio versus accuracy during prediction stage. The approaches compared are model-based scheduling with sensing adaptation (w ad.) and without sensing adaptation (w/o ad.) as well as weighted (Wt.) model-based and CS-based scheduling with adaptation. The CS-based scheduling alone is shown as a benchmark.

Table 4.1: Averaged CPS training and sensing stage performances on 6 subjects using GMM with three mixture components over 90 days of training stage, 3 days of prediction stage, with $\alpha = 0.2$ and false alarm ratio < 0.2.

Subj.	**Training stage**			**Prediction stage**		
	Accu.	PPV	Spec.	Accu.	PPV	Spec.
1	75.28%	28.42%	37.75%	92.33%	42.62%	46.16%
2	72.87%	36.23%	49.72%	86.45%	52.23%	60.42%
3	69.90%	37.28%	53.33%	82.44%	53.35%	64.71%
4	63.27%	28.46%	44.98%	81.75%	53.44%	65.37%
5	70.26%	27.19%	38.70%	83.30%	47.03%	56.46%
6	67.47%	36.32%	53.83%	77.23%	51.28%	66.40%
avg.	69.84%	32.32%	46.39%	83.92%	49.99%	59.92%

sensing scheme, while the weighted model-based with adaptation combined with CS-based scheduling has a close energy consumption level as compared with model-based scheduling without adaptation. The energy consumption performance of the *SamS* component is also evaluated in the experiments. In Figure 4.9, we can see that the addition of the *SamS* component saves up to 39.62% of energy when using model-based scheduling and 9.26% when compared with weighted model-based and CS-based sensing scheme. The results also show that for continuous sensing the battery level reduces more drastically, which has a negative impact on smartphone's normal usage.

Figure 4.10 shows the energy consumption results of the CPS framework using the sparse binary measurement matrix in the randomized CS process as compared to using the traditional dense random Gaussian and scrambled Fourier measurement matrices. We observe that energy consumption using the proposed sparse binary measurement matrix presents more stable profile which is especially true as the time of the day prolongs.

Recovery Error

During the randomized CS process, the smartphone GPS sensor will collect the longitude, latitude, altitude and speed information. In DAS, we use the randomized CS in continuous 1-minute time interval for each 5-minute time slot. In the following, we will first give the

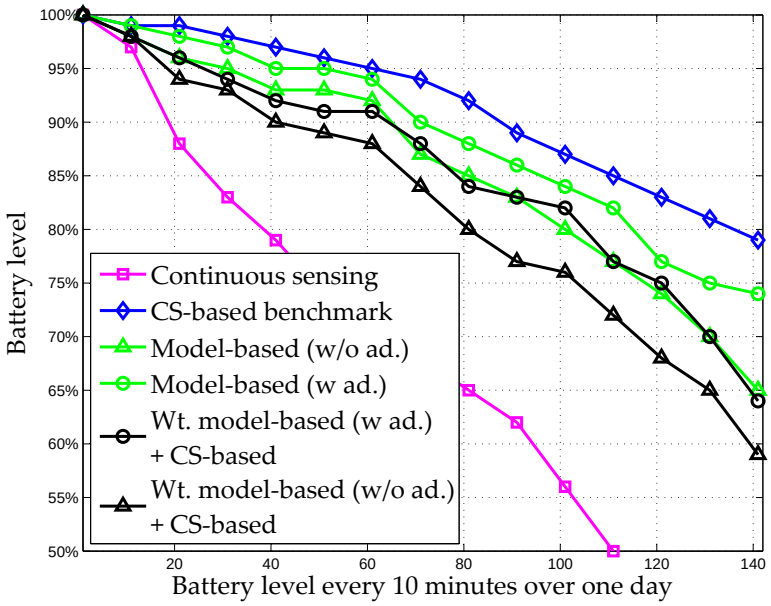


Figure 4.8: Energy consumption comparison of the sensing adaptation. The *SamS* component is included in all the sensing schemes except for the continuous sensing and the CS-based benchmark.

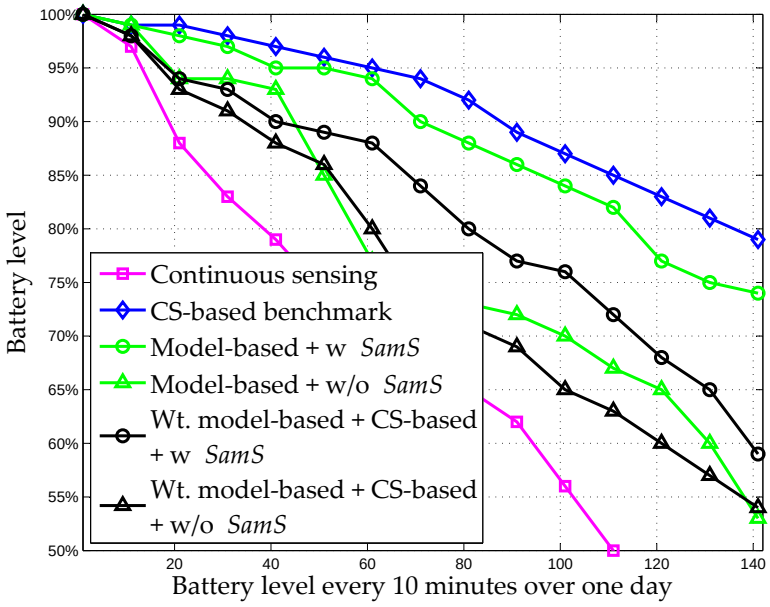


Figure 4.9: Energy consumption comparison of the *SamS* component. The sensing adaptation is adopted in all the sensing schemes except for the continuous sensing and the CS-based benchmark.

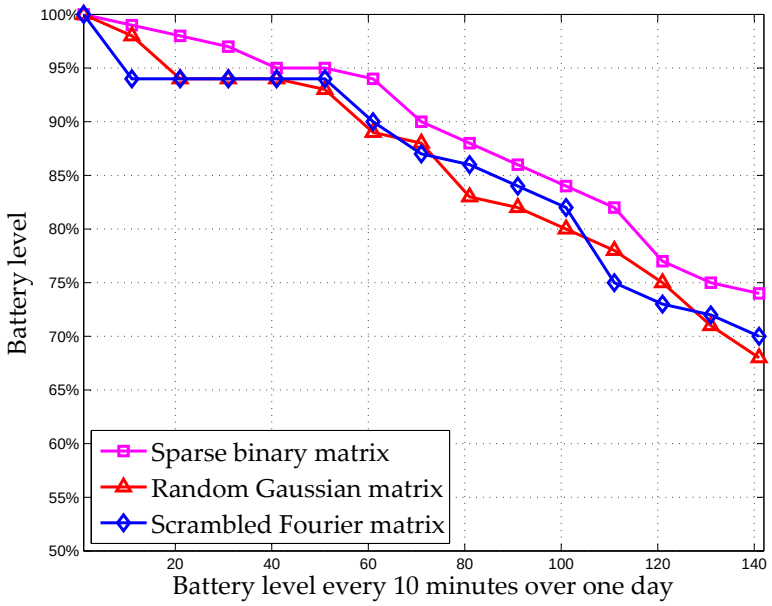


Figure 4.10: Energy consumption using proposed sparse binary measurement matrix versus traditional dense measurement matrices.

results of the recovery error, then we will show that randomized CS can actually be used to recover the driving tracks with very high precision.

Recovery results: The DAS first preprocesses the GPS location signal for compression and efficient storage purpose. The proposed signal preprocessing scheme exploits the high correlation that typically exists between consecutive samples collected by a GPS sensor. We consider only the difference between newly acquired latitude and longitude signal and the previously acquired sample.

Figure 4.11: The DAS tracks.

We use $loc = \{(lg, lat), (dl_{lg}, dl_{lat}), \cdots, (dn_{lg}, dn_{lat})\}$ as the data tuple to record the location signals. Here, the first pair of the original longitude and latitude signals is used for later restoration purpose. We observe that in practice different sensors often report data with different decimal degrees depending on the idiosyncrasy of the particular sensor. Thus we multiply a common factor $c = 10^3$ to the acquired GPS location signal. Figure 4.11 shows the DAS tracks, generated by converting our sensed GPS coordinates to the Keyhole Markup Language (KML) file format that could be displayed by Google Maps*. We select three driving activity tracks each within a 5-minute time slot for CS recovery purpose. *Track 1* is on a local road, *Track 2* is on the Interstate 40 and *Track 3* is on a hybrid road that contains both interstate and local road. These three tracks are represented together by a 1024×2 matrix with the first column representing longitude and the second column the latitude signal. During randomized CPS, the Discrete Cosine Transform (DCT) is used as the sparse basis (i.e., Ψ in Eq. (3.2)) for the GPS signal and both the longitude and latitude signal will be multiplied with a sparse binary matrix (i.e., Φ in Eq. (3.2)) of size $M \times N$ (in this case $N = 1024$). These measurements will be stored and uploaded to the sever for later recovery of location tracks. On the server side, we recover the latitude and longitude separately via ℓ_1-magic Candes and Romberg (2005).

We choose a measurement ratio (i.e., $\frac{M}{N}$) of 0.48 to recover the 1024-dimensional GPS location signal, thus our sparse binary measurement matrix is of dimension 500×1024. This is to balance the recovery performance as well as the computational complexity. The NRMSE for the longitude signal recovery is approximately 0.0147 ± 0.007, and 0.0075 ± 0.004 for the latitude signal. As shown in Figure 4.12, the signals are reconstructed more accurately as the number of measurements increases. When the number of measurements $K = 500$, about 99.34% of samples will be successfully recovered with accuracies up to the 4th decimal place, which is equivalent to a distance accuracy about 11.1m. When $K = 900$, 75.65% of times, the recovery accuracy will reach the 5th decimal place (i.e., 1.11m distance accuracy). For various location based applications, 1.11m distance accuracy is

*https://maps.google.com

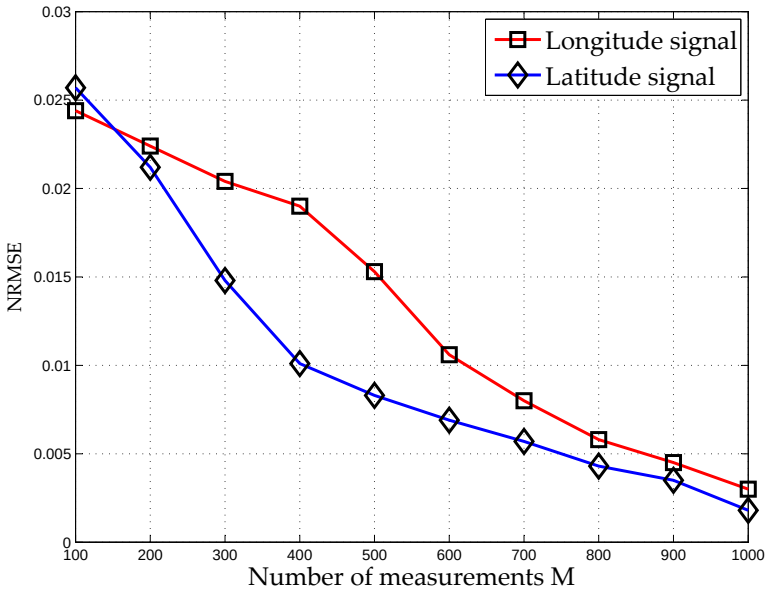


Figure 4.12: Randomized CPS recovery error with different number of measurements.

acceptable which in turn also requires significantly large number of measurements as shown in the experiments.

The DAS recovered tracks: We show the recovered DAS Tracks in Figure 4.13. The first row contains the full track which is recovered using 500 measurements to 1024 GPS coordinates. While we still can identify all the three tracks, *Tracks 2* seems to be more smooth along the road, *Track 1* has some deviations at the turn, for the hybrid road *Track 3*, when reaching the complex urban areas, we have some perturbations in the recovered route. We also apply a postprocessing step on the recovered location signal by removing certain duplications and sorting the coordinates. The second row of Figure 4.13 shows the recovered tracks with only 193 GPS coordinates. By automatically sorting some of the GPS coordinates, we see a reduction in the zigzag routes.

4.2.2 Overhead of the CPS Framework

The major overhead introduced by the CPS framework is on the PM matrix construction and maintenance, the randomized compressed sensing, and the extra energy consumed by frequently turning on/off mobile sensors. Figure 4.8 has already compared the total energy

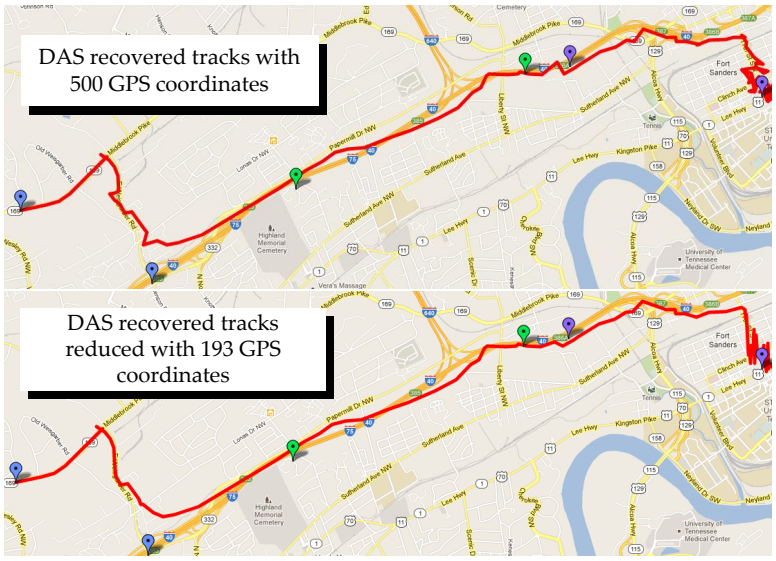


Figure 4.13: The DAS recovered tracks

consumption. Here, we provide a detailed CPU and memory usage on the smartphone during the entire CPS process. We mainly focus on the resource usage during the PM construction (**PM-Con**) (i.e., activity PM modeling in both model-based and CS-based scheduling approaches), the PM update (**PM-Up**) (i.e., sensing adaptation), and the randomized compressed sensing (**R-CS**) in *SamS* component as well as compared with the continuous sensing (**Cont**) scheme. Figure 4.14 demonstrates the CPU and memory usage on the smartphone. For continuous sensing, the averaged CPU usage is about 31%. The CPU usage increases to 64% during the time of the PM construction and rises up to 71% when the randomized compressed sensing is performed on the smartphone. For the memory usage, we observe that the CPS framework does not introduce much overhead. Actually for the randomized compressed sensing, the DAS consumes less memory when compared with that of continuous sensing, which validates the benefit of using the sparse binary measurement matrices.

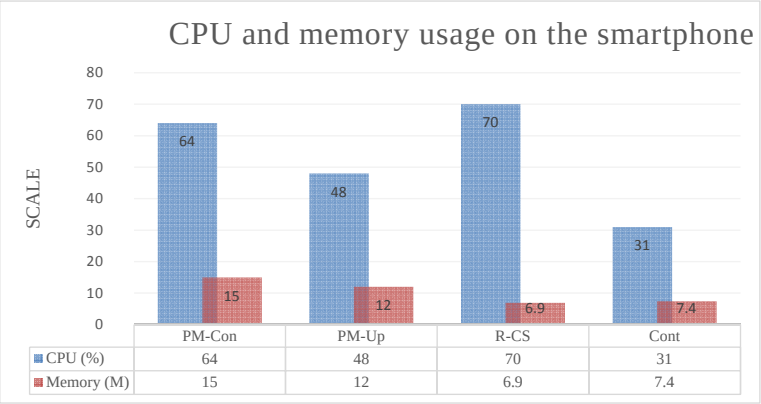


Figure 4.14: CPU and memory usage on the smartphone.

The energy consumption calculation on switching sensor on/off has been introduced in Misra and Lim (2011) for smartphone sensors. Based on Eq. (4.1)

$$E_s = \begin{cases} P_{idl} * (\frac{N}{f} - \frac{N*S}{B}) + P_a * \frac{N*S}{B} \\ \quad +E_{switch} \qquad \text{if } \frac{N}{f} - \frac{N*S}{B} > T_{idl} \\ \\ P_a * \frac{N}{f} \qquad\qquad\qquad \text{otherwise} \end{cases} \tag{4.1}$$

where P_a and P_{idl} are the power consumed in active mode and 'idle' mode, respectively, we consider the case of a generic sensor, operating under a sampling frequency of f Hz with a sample size of S bits and B being the transmission bandwidth (bps). Typically, for an IEEE 802.11 radio, $P_a = 947$mW, $P_{idl} = 231$mW, $B = 54$Mbps, $E_{switch} = 14\mu$Joule, $T_{idl} = 100$ms. Simple calculation reveals that, the overhead introduced by switching on/off sensor is 10^6 less than the energy consumed by continuous sensing.

4.3 Summary

In the first component of the dissertation research, we investigated the smartphone activity sensing based on human activity patterns. An energy efficient CPS framework was proposed that utilized activity pattern to combine both model-based and CS-based

scheduling for the sensing task such that sensors can be alleviated from the burden of continuous sensing in order to save energy. CPS also developed a lightweight compressed sensing scheme where instead of using traditional CS measurement matrix, the sparse binary measurement matrix was designed for sensing to reduce the energy consumption as well as computation burden of smartphones. We implemented the proposed CPS framework on an Android-based smartphone for driving activity sensing (DAS). Evaluation results on the implementation showed that CPS had an averaged sensing scheduling accuracy about 83.92% on 6 subjects over 4 months while the smartphone energy consumption using CPS is 62.86% less than that of continuous sensing and can have a distance accuracy within 11.1m on almost every recovered GPS location samples using only 48% of samples as compared to traditional sensing.

Chapter 5

A Douglas-Rachford Splitting Approach to Compressed Sensing Image Recovery using Low-rank Regularization

In this chapter, we study the Compressed Sensing (CS) image recovery problem. The traditional method divides the image into blocks and treats each block as an independent sub-CS recovery task. This often results in losing global structure of an image. In order to improve the CS recovery result, we propose a nonlocal estimation step after the initial CS recovery for denoising purpose. The nonlocal estimation is based on the well-known nonlocal means (NL) filtering that takes advantage of self-similarity in images. We formulate the nonlocal estimation as the low-rank matrix approximation problem where the low-rank matrix is formed by the nonlocal similarity patches. An efficient algorithm, NonLocal Douglas-Rachford (NLDR), based on Douglas-Rachford splitting is developed to solve this low-rank optimization problem constrained by the CS measurements. Experimental results demonstrate that the proposed NLDR algorithm achieves significant performance improvements over the state-of-the-art in CS image recovery.

5.1 Introduction

Compressed Sensing (CS) has drawn quite some attention as a joint sampling and compression approach Donoho (2006); Candès et al. (2006). It states that under certain conditions, signals of interest can be sampled at a rate much lower than the Nyquist rate while still enabling exact reconstruction of the original signal. CS-based approach has an attractive advantage that the encoding process is made signal-independent and computationally inexpensive at the cost of high decoding/recovery complexity. Usually, the CS measurement is acquired through projecting the raw signals on to a pre-defined random sampling operator. Thus, CS is especially desirable in some image processing applications when the data acquisition devices must be simple (e.g., inexpensive resource-deprived sensors), or when oversampling can harm the object being captured (e.g., X-ray imaging) Lustig et al. (2007), among which the compressive sensing Magnetic Resonance Imaging (CS-MRI) is most promising as it significantly reduces the acquisition time of MRI scanning. When applied to 2D images, CS faces several challenges including a computationally expensive reconstruction process and huge memory required to store the random sampling operator Mun and Fowler (2009). Several fast algorithms have been developed for CS reconstruction Zhang et al. (2012b); Mun and Fowler (2009); Li et al. (2009a). The memory challenge was first addressed in Gan (2007) using a block-based sampling operation, which later on became the most common method in CS image recovery.

Block-based compressed sensing (BCS) has made the CS image recovery practical since it reduces the recovery cost, where image acquisition is conducted in a block-by-block manner through the same compressed sensing (CS) measurement operator. However, manually dividing the image into blocks and treating each image block as an independent sub-CS recovery task would inevitably lose some global properties of the image. Thus it would often require some filtering technique (i.e., Wiener filter Mun and Fowler (2009)) to generate good visual recovery result. Nonetheless, the recovered image still suffers a low PSNR. Aside from BCS, another class of popular methods is based on the total variation

(TV) model Zhang et al. (2012b); Dahl et al. (2010), which exploits the prior knowledge that a natural image is sparse in the gradient domain. TV based algorithms often suffer from undesirable staircase artifacts and tend to over-smooth image details and textures Louchet and Moisan (2011).

In this chapter, we propose **NLDR**, a CS image recovery algorithm based on the BCS scheme. We overcome the aforementioned BCS problems by introducing a new nonlocal estimation step after the initial CS reconstruction to further remove noise. The nonlocal estimation process is built on the well-known nonlocal means (NL) filtering that takes advantage of self-similarities in images, which preserves certain global structure. We formulate the nonlocal estimation into the low-rank approximation problem where the low-rank matrix is formed by the nonlocal similarity patches. Furthermore, by using a deterministic annealing (DA) approach, we incorporate the CS measurement constraint into the low-rank optimization problem. We propose an efficient algorithm based on Douglas-Rachford splitting (DR) to solve the low-rank matrix approximation problem combined with the CS measurement constraints, the solution to which is the final CS recovery output. The proposed NLDR algorithm effectively reduces the staircase artifacts that introduced in BCS and TV by utilizing the nonlocal similarity patches while preventing over-smoothness by recursively incorporating the initial CS measurement constraint.

The rest of the chapter is organized as follows. Section 5.2 provides a brief review of the CS image recovery problem as well as some related works. Section 5.3 discusses the nonlocal estimation and Douglas-Rachford Splitting method. We conduct experiments in Section 5.4 on both standard test images and MRI images. Section 6.6 concludes the chapter.

5.2 Background and Related Works

5.2.1 CS Image Recovery Problem

Mathematically, the sparse representation model assumes that a signal $x \in \mathcal{R}^n$ can be represented as $x = \Psi\alpha$, where $\Psi \in \mathcal{R}^{n\times n}$ is a sparsifying basis or dictionary, and most entries of the coding vector α are zero or close to zero. This sparse decomposition of x can be obtained by solving a relaxed convex ℓ_1-minimization problem in the following Lagrangian form:

$$\min_{\alpha}\{\|x - \Psi\alpha\|_2^2 + \lambda_\alpha\|\alpha\|_1\}, \tag{5.1}$$

where constant λ_α denotes the regularization parameter.

In CS image recovery, we consider an image $I \in \mathcal{R}^{\sqrt{n}\times\sqrt{n}}$. By representing the image I in vector format, denoted as x, what we observe is the projected measurement y via $y = \Phi x + \nu$, where $\Phi \in \mathcal{R}^{m\times n}(m < n)$ is the measurement operator and ν is the additive noise vector. To recover x from y, first y is sparsely coded with respect to the basis Ψ by solving the following minimization problem

$$\hat{\alpha} = \arg\min_{\alpha}\{\|y - \Phi\Psi\alpha\|_2^2 + \lambda_\alpha\|\alpha\|_1\} \tag{5.2}$$

and then x is reconstructed by $\hat{x} = \Psi\hat{\alpha}$.

This can be easily extended to the block-based scenario, as stated in Elad and Aharon (2006). Let $x_i = R_i x$ denote an image patch extracted at location i, where R_i is the matrix extracting patch x_i from x at pixel location i. Given a basis Ψ, each patch can be sparse represented and solved by Eq. (5.1). Then the entire image x can be represented by the set of sparse code using $\{\Psi\alpha_i\}$. The patches can be overlapped to suppress the boundary artifacts.

Similarly, in order to reconstruct the image x from the measurement y, we can adopt the same block-based CS recovery by solving α_i from Eq. (5.2). The whole image x is then

reconstructed as $\hat{x} = \Psi\hat{\alpha} = (\sum_i^N R_i^T R_i)^{-1} \sum_i^N (R_i^T \Phi \hat{\alpha}_i)$ as proved in Elad and Aharon (2006).

The Iterative soft thresholding (IST) algorithm Daubechies et al. (2004) can be very efficient in solving the problem in Eq. (5.2). In the $(k+1)$-th iteration, the solution is given by $\alpha^{(k+1)} = \mathcal{S}_\tau(\alpha^{(k)} + \Phi^* y - \Phi^* \Phi \Psi \alpha^{(k)})$, where $\mathcal{S}_\tau(\cdot)$ is the classic soft-thresholding operator Daubechies et al. (2004). In this paper, we use a slightly modified IST algorithm Daubechies et al. (2008), where the solution in each iteration is called the *projected Landweber iteration* with the adaptive descent parameter $\beta^{(k)} > 0$,

$$\alpha^{(k+1)} = \mathcal{P}_\mathcal{R}[\alpha^{(k)} + \beta^{(k)} \Phi^* (y - \Phi \Psi \alpha^{(k)})], \tag{5.3}$$

where $\mathcal{P}_\mathcal{R}$ is the ℓ_2 projection of α on the ℓ_1 ball with radius $\mathcal{R}$. The adaptive descent parameter $\beta^{(k)}$ can be selected using the greedy strategy as follows,

$$\beta^{(k)} = \frac{\|\Phi^*(y - \Phi\Psi\alpha^{(k)})\|_2^2}{\|\Phi\Phi^*(y - \Phi\Psi\alpha^{(k)})\|_2^2} \tag{5.4}$$

This is an accelerated version of IST that converges faster than the original IST. Readers may refer to Daubechies et al. (2008) for details.

5.2.2 Other Related Works

Buades et al. introduced in Buades et al. (2005) the *nonlocal means* (NLM) filtering approach to image denoising, where the self-similarities between rectangular patches are used as a prior on natural images. The idea of nonlocal means has recently received much attention in image processing Peyré et al. (2008); Yang and Jacob (2013); Zhang et al. (2012a); Shu et al. (2014); Dong et al. (2014b); Chierchia et al. (2014); Dong et al. (2014a). For example, Peyre et al. Peyré et al. (2008) proposed to use the Total Variation (TV) prior and nonlocal graph to solve the inverse problem with application in CS. The same idea was also adopted in Yang et al. Yang and Jacob (2013). Zhang et al. Zhang et al. (2012a) proposed TVNLR which improves the conventional TV approach by adding a nonlocal

regularization to the CS recovery problem and solved the problem using the Augmented Lagrangian Method (ALM). Shu et al. proposed the NLCS algorithm Shu et al. (2014) and tried to group similar patches through NLS (nonlocal sparsity) regularization. The authors in Chierchia et al. (2014) proposed a nonlocal total variation structure tensor (ST-NLTV) regularization approach for multicomponent image recovery from degraded observations, leading to significant improvements in terms of convergence speed over state-of-the-art methods such as the Alternating Direction Method of Multipliers (ADMM). Dong et al. proposed the nonlocal low-rank regularization (NLR-CS) method Dong et al. (2014a) which explored the structured sparsity of the image patches for compressed sensing. In order to explore the low-rank structure of the image patches, a smooth but non-convex surrogate function for the rank estimation is adopted as objective function. Zhang et al. proposed nonlocal TV regularization (NLVT) Zhang et al. (2010) for CS image recovery. NLTV is based on the Bregman iteration Osher et al. (2005), namely Bregmanized Operator splitting (BOS).

In this work, we adopt the nonlocal means filtering idea and introduce a new nonlocal estimation step after the initial CS reconstruction to further remove noise. It differs from Peyré et al. (2008) as we use the ℓ_1-norm based sparsity of the image and result in solving a convex optimization problem using the projection method. In Peyré et al. (2008) the nonlocal graph is similar to the nonlocal weights between patches as used in our paper. The main difference is that the author further imposed that these weights correspond to a probability distribution and that the graph only connects pixels that are not too far away. While in Yang and Jacob (2013), the nonlocal weights may be improved using a different distance metric (i.e., robust distance metric) to promote the averaging of similar patches while minimizing the averaging of dissimilar patches. In this paper, we only aim to find similar patches to form low-rank matrix and thus differ from these methods. In Dong et al. (2014a) instead of using the nuclear norm for low-rank approximation, the authors proposed to use non-convex surrogate function and subsequently solved the optimization problem via ADMM.

In Shu et al. (2014), two non-local sparsity measures, i.e., non-local wavelet sparsity and non-local joint sparsity, were proposed to exploit the patch correlation in NLCS. It then combines with the conventional TV measure to form the optimization objective function and use the ADMM method to solve the CS recovery problem. It differs from our algorithm in that their search for similar patches is incorporated in the objective function while NLDR directly adopts the nonlocal means filtering approach to find the similar patches and then conducts low-rank approximation. After getting the non-local low-rank estimation, we further incorporate the initial CS measurement constraint into the low-rank optimization problem, using a deterministic annealing (DA) approach to further improve the recovery result. Additionally, compared to the traditional ADMM method, we propose to use Douglas-Rachford splitting method to effectively solve the combined optimization problem.

In Candès and Tao (2010), Candès and Tao proposed to solve the matrix completion problem using low-rank regularization through convex optimization. Later in Dong et al. (2013) Dong et al. first combined the nonlocal image representation and low-rank approach for image restoration and achived state-of-the-art performance in image denosing. Ji et al. (2010) also incorporated the low-rank matrix completion in video denoising.

To summarize, the main contribution of this chapter is three-fold: First, we propose to incorporate the nonlocal similarity patches searching step after the initial CS image recovery task. By searching and incorporating the nonlocal similarity patches the traditional block based CS recovery artifacts could be resolved. Second, we propose to estimate the grouped similarity patches matrix as a low-rank matrix completion problem, referred as nonlocal low-rank estimation. The idea is that, by searching the nonlocal similarity patches we could resolve the block and staircase artifacts, while using low-rank estimation we can further denoise the grouped similarity patches. Third, we incorporate the initial CS measurement constraint into the low-rank estimation optimization problem. By using a deterministic annealing (DA) approach, the Douglas-Rachford splitting effectively solves the reformulated optimization problem.

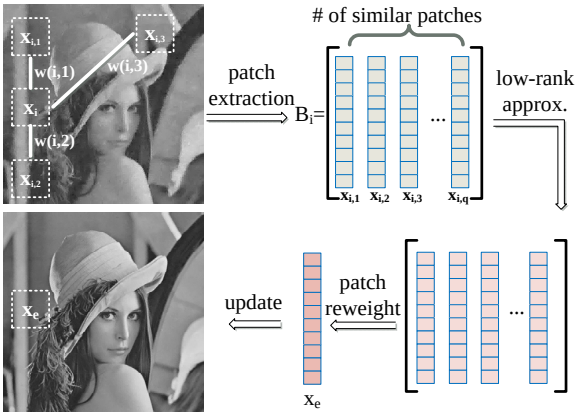


Figure 5.1: An illustration of nonlocal estimation and similar patches denoising using low-rank matrix approximation.

5.3 Nonlocal Low-rank Regularization and Douglas-Rachford splitting

In this section, we present the idea of nonlocal low-rank regularization, followed by the proposed Douglas-Rachford splitting method. We refer to the algorithm as the Nonlocal Douglas-Rachford splitting (NLDR) algorithm.

5.3.1 Nonlocal Low-rank Regularization for CS Image

An example to illustrate the nonlocal estimation step is shown in Fig. 5.1. The *Lena* image in the first row is obtained from the IST CS recovery algorithm. Then the nonlocal similar patches are searched across the entire image. We denote the nonlocal similar patches of x_i as $x_{i,1}, x_{i,2}, x_{i,3}, \cdots x_{i,q}$. These extracted patches then form the matrix B_i where the low-rank approximation is conducted to yield the resulting denoised patch matrix, as shown in the second row. We apply patch reweight to obtain the estimated patch x_e to update the original patch x_i. After iterating over the entire image, the much cleaner *Lena* image is shown leftmost in the second row.

Nonlocal Similarity Patches

The basic idea of nonlocal (NL) means filtering is simple. For a given pixel u_i in an image x, its NL filtered new intensity value, denoted by $NL(u_i)$, is obtained as a weighted average of its neighborhood pixels within a search window of size w.

In our work, we extend the pixel-wise nonlocal filtering to the patch-based filtering. Specifically, we search for the nonlocal similar "patches" $x_{i,j}, j = 1, 2, \cdots, q$, to the given patch x_i in a large window of size w centered at pixel u_i. Here, q is the total number of similar patches to be selected. The weight of patch $x_{i,j}$ to x_i, denoted as ω_{ij}, is then computed by

$$\omega_{ij} = \frac{1}{c_i} \exp\left(\frac{-\|x_i - x_{i,j}\|_2^2}{h^2}\right), \; j = 1, \cdots, q \tag{5.5}$$

where h is a pre-determined scalar and c_i is the normalization factor. Accordingly, for each patch x_i, we have a set of its similar patches, denoted by Ω_i. Then the nonlocal estimates of each patch $\hat{x}_i$ can be computed as $\hat{x}_i = \sum_{j \in \Omega_i} \omega_{ij} x_{i,j}$. Further, this can be written in a matrix form as

$$\hat{x}_{nl} \doteq \mathbf{W} \sum_{i=1}^{p} \hat{x}_i, \; \mathbf{W}(i,j) = \begin{cases} \omega_{ij}, \text{ if } x_j \in \Omega_i \\ 0, \text{ otherwise.} \end{cases} \tag{5.6}$$

where p denotes the number of all patches in the entire image and $\hat{x}_{nl}$ is the nonlocal estimated image output.

Patch Denoising by Low-rank Approximation

Although we can use Eq. (5.6) to remove noise in the IST recovered image $\hat{x}$ to a certain degree, this is based on a weighted average of patches in $\hat{x}$, which are inherently noisy. Thus, it is imperative to apply some denoising techniques before the nonlocal similarity patch reweight using Eq. (5.6) to prevent the noise from accumulating. By rewriting the nonlocal similarity patches into the matrix format, we have $B_i = [x_{i,1}; x_{i,2}; , \cdots; x_{i,q}]$,

where each column of B_i is a vector representation of $x_{i,j}, j = 1, 2, \cdots, q$ for patch x_i. Since all columns of B_i share similarity with patch x_i, the columns of B_i should bear a high degree of similarity between each other. In other words, we can safely treat B_i as a low-rank matrix. We thus formulate the nonlocal patch denoising problem into the low-rank matrix approximation problem Candès and Tao (2010) as follows,

$$\min_{\hat{B}_i} \frac{1}{2}\|B_i - \hat{B}_i\|_2^2 + \lambda_{B_i}\|\hat{B}_i\|_*, \tag{5.7}$$

where $\|\hat{B}_i\|_*$ is the nuclear norm of the low-rank approximated patch matrix $\hat{B}_i$, defined by $\|\hat{B}_i\|_* \triangleq \text{trace}(\sqrt{\hat{B}_i^T \hat{B}_i}) = \sum_{r=1}^{q} \sigma_r$, and σ_r's are the singular values of $\hat{B}_i$.

In addition, since the columns of B_i (or the patches) are also a subset of the reconstructed image from IST recovery algorithm, it should be subject to the CS measurement constraint $y = \Phi x$. Therefore, multiplying Eq. (5.7) with $\mathbf{W}$, we reformulate the denoising problem of Eq. (5.7) into

$$\min_{x} \frac{1}{2}\|x - \mathbf{W}B_i\|_2^2 + \lambda_x\|x\|_* \text{ s.t. } y = \Phi x. \tag{5.8}$$

In what follows, we discuss in sec. 5.3.2 how to solve Eq. (5.8) with the CS measurement constraint using the method referred to as the Douglas-Rachford splitting method.

5.3.2 Douglas-Rachford Splitting

The Douglas-Rachford splitting method was originally proposed in Douglas and Rachford (1956) for solving matrix equations. Later on it was advanced as an iterative scheme to minimize the functions of the form,

$$\min_{x} F(x) + G(x) \tag{5.9}$$

where both F and G are convex functions for which one is able to compute the proximal mappings $\mathsf{prox}_{\gamma F}$ and $\mathsf{prox}_{\gamma G}$ which are defined as

$$\mathsf{prox}_{\gamma F}(x) = \arg\min_{y} \frac{1}{2}\|x - y\|_2^2 + \gamma F(y) \tag{5.10}$$

The same definition applies to $\mathsf{prox}_{\gamma G}$ Combettes and Pesquet (2011). In order to solve Eq. (5.8), we have $F(x) = \iota_{\mathcal{C}}(x)$ and $G(x) = \|x\|_*$, where $\mathcal{C} = \{x : y = \Phi x\}$ and $\iota_{\mathcal{C}}$ is the indicator function.

Given that $F(x) = \iota_{\mathcal{C}}(x)$, the solution to Eq. (5.10) is the same as projections onto convex sets (POCS), and does not depend on γ. Therefore, we have

$$\mathsf{prox}_{\gamma \iota_{\mathcal{C}} F}(x) = \mathsf{prox}_{\iota_{\mathcal{C}} F}(x) = x + \Phi^{+}(y - \Phi x), \tag{5.11}$$

where $\Phi^{+} = \Phi^{T}(\Phi\Phi^{T})^{-1}$ is the pseudoinverse of Φ. The proximal operator of $G(x)$ is the soft thresholding of the singular values

$$\mathsf{prox}_{\gamma G}(x) = U(x) \cdot \rho_{\lambda_x}(S(x)) \cdot V(x)^{*} \tag{5.12}$$

where $x = U \cdot S \cdot V^{*}$ is the singular value decomposition of the matrix x and $S = \mathrm{diag}(s_i)_i$ is the diagonal matrix of singular values s_i, and $\rho_{\lambda_x}(S)$ is defined as a diagonal operator.

$$\rho_{\lambda}(S) = \mathrm{diag}(\max(0, 1 - \lambda_x/|s_i|)s_i)_i \tag{5.13}$$

We can then solve the problem in Eq. (5.7) using the Douglas-Rachford iterations given by

$$\tilde{x}_{k+1} = (1 - \frac{\mu}{2})\tilde{x}_k + \frac{\mu}{2}\mathsf{rprox}_{\gamma G}(\mathsf{rprox}_{\gamma F}(\tilde{x}_k)) \tag{5.14}$$

and the $(k+1)$-th solution $\hat{x}_{k+1}$ is calculated by $\hat{x}_{k+1} = \mathsf{prox}_{\gamma F}(\tilde{x}_{k+1})$. Here the reversed-proximal mappings is given by $\mathsf{rprox}_{\gamma F} = 2\mathsf{prox}_{\gamma F} - x$ for $F(x)$ and in the similar fashion to G(x).. The parameters are selected as $\lambda_x > 0$ and $0 < \mu < 2$ which guarantee $\hat{x}$ to be a solution that minimizes $F(x) + G(x)$ based on the proof in Combettes and Wajs (2005).

5.3.3 The NLDR Algorithm

Algorithm 2 provides a pseudo-code for the proposed Nonlocal Douglas-Rachford splitting (NLDR) algorithm. Given the observation y (i.e., compressed measurements), the NLDR algorithm first outputs an intermediate reconstruction result $\hat{x}_{IST}$ through the IST algorithm. This soft-thresholding output is then used to calculate the nonlocal estimated image $\hat{x}_{nl}$, which is used to initialize the low-rank optimization problem in Eq. (5.7) where the Douglas-Rachford splitting method will be carried out iteratively based on Eq. (5.14).

Algorithm 2: Nonlocal Douglas-Rachford Splitting (NLDR) Algorithm

Input:

- ▶ Measurement matrix $\Phi \in \mathbb{R}^{m\times n}$
- ▶ Basis matrix $\Psi \in \mathbb{R}^{n\times n}$
- ▶ Observation vector $y \in \mathbb{R}^m$.
- ▶ Number of IST iterations iter, number of nonlocal estimation iterations J, DR splitting iterations K

Output:

- ▶ An estimate $\hat{x} \in \mathbb{R}^n$ of the original image x.

Initialize $\alpha^0 \leftarrow \mathbf{0}$
for $k = 1, \cdots,$ iter do
  (a) Select $\beta^{(k)}$ based on Eq. (5.4)
  (b) Update $\alpha^{(k+1)}$ using Eq. (5.3)
end for
for $j = 1, 2, \cdots, J$ do
Step 1: Nonlocal Estimate
  (a) Calculate nonlocal weights ω_{ij} via Eq. (5.5)
  (b) Obtain low-rank patch matrix B_i via Eq. (5.7)
Step 2: Douglas-Rachford Splitting to solve Eq. (5.8)
  for $k = 1, 2, \cdots,$ K do
  (a) Calculate $\mathsf{prox}_{\gamma F}(x)$ via Eq. (5.11)
  (b) Calculate $\mathsf{prox}_{\gamma G}(x)$ via Eq. (5.12)
  (c) Calculate $\tilde{x}_{k+1}$ via Eq. (5.14)
  end for
end for
return $\hat{x} \leftarrow \tilde{x}_{k+1}$

As for calculating the nonlocal estimates of the image, the NLDR algorithm obtains the averaged result based on J nonlocal estimation iterations. For the IST algorithm, we

empirically set the penalty parameter $\lambda_\alpha = 1.8$ and soft-thresholding parameter $\tau = 1.2$, respectively.

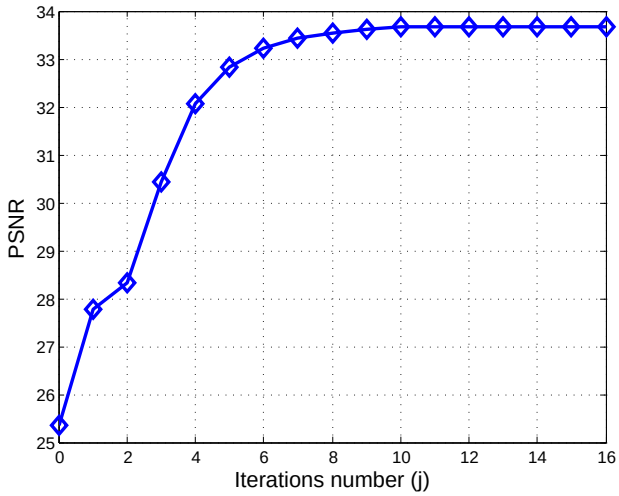


Figure 5.2: CS recovery results on *Lena* image with 10% measurements at iteration j.

5.4 Experiments

In this section, we evaluate the NLDR algorithm for CS image reconstruction where both standard test images and MRI images are used. The reason for choosing MRI images for evaluation purpose is due to the significant impact of CS on the clinical practice of MRI, where long acquisition time has been one of the primary obstacles. We implement the algorithm using *Matlab* 2013b on a 2.20GHz laptop computer. BCS-SPL Mun and Fowler (2009) is a block-based CS image recovery method solved using a smoothed version of projected Landweber (SPL) algorithm. The smoothing process is done by the Wiener filter. We further compare our result with one of the state-of-the-art algorithms for image CS recovery, known as TVAL3 Li et al. (2009a). TVAL3 tries to minimize the image total variation norm using augmented lagrangian and alternating direction algorithms. Several TV-based methods are also compared. The TV benchmark method denoted as TV which is implemented based on l1magic (2006), TVNLR Zhang et al. (2013) and NLTV Zhang et al. (2010). We also compare NLDR performance with other nonlocal based approaches,

e.g., NLCS Shu et al. (2014) and NLR-CS Dong et al. (2014a). Finally, to evaluate the potential of NLDR as a standalone denoising method, we compare its performance with the state-of-the-art BM3D Dabov et al. (2007) method for noise removal purpose.

5.4.1 CS Recovery on Standard Image Dataset

We present the experimental results for noiseless CS measurements and then report the results using noisy CS measurements.

Noiseless Recovery

We first test the NLDR algorithm in noiseless settings using standard test images of size 512×512. The block-based image patch is of size 6×6. We set the number of similar patches q in the nonlocal estimation step as 45. We use the scrambled Fourier matrix as the CS measurement operator Φ and DCT matrix as the basis Ψ to represent the original image in the initial IST recovery. The parameters are selected as $\mu = 1$ for DR iteration and $\lambda_x = \frac{c_i}{\max(s_i)}$ for each iteration where $c_i = C_0 * \epsilon, 0 < \epsilon < 1$ and C_0 is a constant. For the number of iterations in the outerloop, we find that the recovery result gradually converges when J reaches 12 for all the image datasets. Fig. 5.2 shows one example on *Lena* image using 10% of measurements. Note that at iteration 0, we use the initial IST recovery result.

Table 5.1 compares PSNR with different measurement ratios (i.e., $\frac{m}{n}$). We see that the NLDR algorithm considerably outperforms the other methods in all the cases, with PSNR improvements of up to 11.38dB and 13.68dB, as compared with BCS-SPL and TVAL3, respectively. Furthermore, the average PSNR gain by NLDR over BCS-SPL is 6.18dB and 5.17dB over TVAL3. For the other nonlocal based methods, we see that NLDR also outperforms them, with average PSNR gain over NLCS by 2.19dB, 5.41dB over TVNLR, 2.79Db over NLR-CS and 4.28dB over NLTV.

Since originally NLDR is calculated on top of the IST recovery algorithm with an extra nonlocal estimation step, in order to perform a fair comparison among the BCS-SPL and TVAL3 algorithms, we use the result image from BCS-SPL and TVAL3 algorithm as the

(a) Original Barbara (b) NLDR (c) BCS-SPL (d) TVAL3

(e) TVNLR (f) NLCS (g) NLR-CS (h) NLTV

Figure 5.3: CS Reconstructed image *Barbara* with 30% measurement ratio. (a) Original image; (b) proposed **NLDR** recovery, PSNR=37.30dB; (c) BCS-SPL recovery Mun and Fowler (2009), PSNR=25.92dB; (d) TVAL3 recovery Li et al. (2009a), PSNR=24.79dB; (e) TVNLR recovery Zhang et al. (2013), PSNR=25.35dB. (f) NLCS recovery Shu et al. (2014), PSNR=31.65dB; (g) NLR-CS recovery Dong et al. (2014a), PSNR=34.26dB; (h) NLTV recovery Zhang et al. (2010), PSNR=31.79dB.

(a) NLDR (b) NLCS (c) TVNLR (d) NLR-CS (e) NLTV

Figure 5.4: *Boat* image with cropped character patch using 20% measurements. (a) proposed **NLDR** recovery, PSNR=32.48dB; (b) NLCS recovery Shu et al. (2014), PSNR=30.66dB; (c) TVNLR recovery Zhang et al. (2013), PSNR=28.02dB; (d) NLR-CS recovery Dong et al. (2014a), PSNR=29.07dB; (e) NLTV recovery Zhang et al. (2010), PSNR=27.97dB.

input to the NLDR algorithm. By doing this, we would be able to quantify how much improvement NLDR has gained. Also, since the initial image from IST output is noisy, we further apply the state-of-the-art denoising algorithm - BM3D on top of the IST recovery result to denoise the result image in order to compare with the NLDR result.

In Table 5.1, the column TVAL3+NLDR denotes applying NLDR on the TVAL3 resulting image, the column BCS-SPL+NLDR denotes NLDR applied on top of the BCS-SPL output, and IST+BM3D denotes BM3D applied on top of the IST output. Note, we also generate the sole IST algorithm output in the first column. From the table, we can see that the columns correspond to TVAL3+NLDR, BCS-SPL+NLDR and NLDR yield similar PSNR. This result indicates the generalization capability of NLDR, that it actually gives the best available denoised recovery result no matter what the initial input is. That is, NLDR has the great potential of serving as a stand-alone denoising algorithm.

Some visual results of CS reconstructed image *Barbara* with 30% measurement ratio are presented in Fig. 5.3. Obviously, NLDR generates much better visual quality than those from BCS-SPL and TVAL3, where both BCS-SPL and TVAL3 have blurred artifacts. When compared using Table 5.1, we see NLDR outperforms the other two algorithms largely in PSNR. The reason is that the image *Barbara* itself has a lot of texture patterns (i.e., nonlocal similar patches), which had been successfully exploited in the NLDR algorithm. Fig. 5.4 demonstrates the *Boat* image with cropped character patch using 20%

(a) Original Lena (b) IST + NLDR (c) TVAL3 + NLDR (d) BCS-SPL + NLDR

Figure 5.5: Part of *Lena* image with 200% magnification using 20% measurements. (a) Original image; (b) reconstruction using proposed NLDR with IST, PSNR=36.33dB; (c) TVAL3 + NLDR, PSNR=36.35dB (d) BCS-SPL + NLDR, PSNR=36.35dB.

Table 5.1: PSNR Performance in dB.

Images		**Lena**											
Algorithms		IST	TV	TVAL3	BCS-SPL	IST+BM3D	NLCS	TVNLR	NLR-CS	NLTV	NLDR	TVAL3+NLDR	BCS-SPL+NLDR
m/n	0.1	25.41	22.75	29.02	28.31	25.93	31.74	28.62	29.58	25.94	33.67	**33.81**	33.80
	0.2	29.57	24.44	31.56	31.37	30.42	34.78	30.98	32.95	29.73	36.33	**36.35**	**36.35**
	0.3	32.05	25.47	32.99	33.50	32.91	36.67	33.52	34.73	31.73	37.82	**37.83**	**37.83**
	0.4	34.07	27.88	35.03	35.20	34.72	38.22	35.48	36.56	35.39	**39.02**	**39.02**	**39.02**
	0.5	35.89	30.73	36.26	36.79	36.34	39.66	36.94	38.77	37.90	40.16	**40.17**	40.16
		Barbara											
m/n	0.1	21.18	20.10	21.31	22.85	21.34	24.34	22.55	26.90	23.13	29.48	**31.14**	31.01
	0.2	24.35	21.66	21.60	24.33	24.80	28.17	24.30	30.87	28.29	**35.28**	35.21	35.22
	0.3	26.96	23.61	24.79	25.92	27.73	31.65	25.35	34.26	31.79	37.30	37.30	**37.32**
	0.4	29.38	25.32	28.55	27.68	30.29	34.32	26.85	36.14	34.44	**38.95**	**38.95**	**38.95**
	0.5	31.73	26.62	31.08	30.15	32.82	36.63	28.02	39.36	36.23	40.50	**40.51**	**40.51**
		Peppers											
m/n	0.1	24.30	21.98	29.69	28.88	24.77	31.85	26.25	29.18	25.78	32.91	**33.11**	33.07
	0.2	29.16	23.47	32.70	31.44	29.86	33.99	31.64	32.38	29.35	**34.74**	34.70	34.69
	0.3	31.54	25.57	34.02	32.89	32.16	35.27	34.35	33.73	32.85	**35.78**	35.70	35.70
	0.4	33.29	27.56	34.98	34.06	33.57	36.41	35.62	35.29	34.82	**36.91**	36.75	36.75
	0.5	34.73	29.67	36.08	35.18	34.57	37.52	36.74	37.03	36.96	**38.13**	37.99	37.99
		Mandrill											
m/n	0.1	19.68	19.79	19.01	20.21	19.66	20.89	20.37	20.63	19.95	21.15	**21.89**	21.80
	0.2	21.02	21.17	19.22	21.09	20.69	22.78	21.93	21.89	22.25	23.43	**24.08**	23.93
	0.3	22.34	22.81	19.70	21.80	21.79	24.44	23.11	23.21	24.64	25.49	**25.85**	25.73
	0.4	23.68	24.79	20.20	22.92	23.13	26.17	24.26	25.20	26.95	27.47	**27.91**	27.81
	0.5	25.16	26.71	22.73	24.50	24.72	27.87	25.42	27.16	29.37	29.40	**29.71**	29.66
		Goldhill											
m/n	0.1	24.96	22.56	27.45	26.96	25.08	28.53	25.20	28.92	23.58	28.94	**29.66**	29.57
	0.2	27.81	23.93	29.86	28.95	27.87	30.84	28.92	31.60	26.90	32.10	**32.26**	32.24
	0.3	29.53	25.88	31.62	30.56	29.49	32.55	31.29	33.37	30.08	33.99	**34.02**	**34.02**
	0.4	31.23	27.73	33.21	32.09	30.92	34.13	33.08	34.36	32.66	35.61	**35.63**	35.62
	0.5	32.76	29.44	33.29	33.61	32.18	35.67	34.55	36.41	35.01	37.20	37.20	**37.21**
		Cameraman											
m/n	0.1	23.86	22.05	28.50	26.36	24.38	33.26	25.53	30.72	29.71	36.83	**36.97**	36.85
	0.2	26.97	24.11	34.12	30.07	31.66	37.49	30.67	35.29	33.95	41.49	**41.56**	41.55
	0.3	33.66	26.55	38.16	32.82	35.44	40.71	34.34	37.79	37.65	43.92	**43.96**	**43.96**
	0.4	34.85	29.23	40.42	35.48	38.28	43.40	37.56	41.65	40.83	45.96	**46.01**	45.95
	0.5	36.26	34.02	43.01	37.85	40.71	45.92	39.95	45.56	43.67	47.90	**47.92**	47.90
		Boat											
m/n	0.1	23.77	21.48	25.76	24.65	24.16	27.74	24.03	26.41	23.94	28.69	**29.52**	29.24
	0.2	27.01	23.18	28.94	27.02	27.38	30.66	28.02	29.07	27.97	32.48	**32.68**	32.63
	0.3	29.10	24.84	31.09	28.94	29.61	32.64	30.80	30.65	30.89	34.41	**34.47**	34.43
	0.4	30.91	26.93	32.68	30.59	31.24	34.26	33.06	32.48	33.45	35.77	35.81	**35.86**
	0.5	32.68	29.19	33.53	32.19	32.76	35.73	34.66	35.87	36.04	37.24	37.24	**37.25**

measurements. Also, we show in Fig. 5.5 the result of original NLDR using IST as well as TVAL3+NLDR and BCS-SPL+NLDR. They all have similar visual results as compared to the original image. This is consistent to the observation made based on Table 5.1 that their recovery PSNRs are very close.

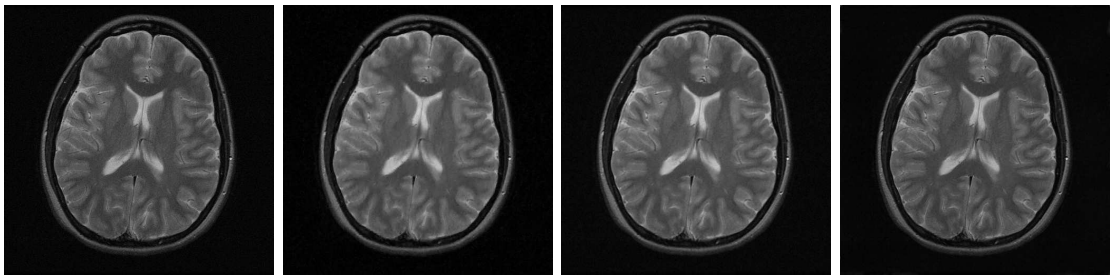
Noisy Recovery

In this experiment, the robustness of the NLDR algorithm to noise is demonstrated. In practice, CS measurements consist mostly of linear operations, thus the Gaussian noise corrupting the signal during the signal acquisition is approximated as the Gaussian noise corrupting the compressed measurement vector. In our experiments, we simply corrupt the compressed measurement vector by different levels of Gaussian noise measured by Signal-to-Noise Ratios (SNRs). We use all seven standard test images and add different SNRs $(5, 10, 15, 25, 35)$ to their 20% CS measurements and report the PSNR values of the final CS recovered image in Table 5.2.

From Table 5.2, we see that by adding 5dB of Gaussian noise on the CS measurements, all the TV-based algorithms' (i.e., TV, NLTV, TVAL3 and TVNLR) recovery performance suffer in terms of PSNR as compared with their original noiseless recovery settings. When the noise SNR reaches 35, the recovery result is close to its noiseless case. It also demonstrates that the recovery performance degrades on both BCS-SPL and NLCS when noise is added while NLDR is affected much less by the noise in all SNR cases. We see that the NLR-CS algorithm is also robust on noise with only less than 1dB PSNR decrease as compared with its noiseless settings for all the testing images. For BM3D, as a denoising algorithm, we see that the recovery result is not affected much with different noise dB levels. However, NLDR still outperforms NLR-CS and BM3D in the noisy CS recovery case.

Table 5.2: CS noisy recovery results on standard test images with 20% measurements.

	Algorithm								
	NLDR	**TV**	**NLTV**	**BCS-SPL**	**TVAL3**	**TVNLR**	**NLCS**	**NRL-CS**	**BM3D**
SNR	**Lena**								
5	36.24	21.27	25.94	30.50	28.82	28.14	32.45	32.55	30.32
10	36.29	21.63	27.66	30.51	28.93	28.43	33.13	32.65	30.31
15	36.29	22.19	28.34	30.52	30.94	29.23	33.44	32.76	30.31
25	36.29	23.63	29.01	30.52	31.18	30.96	34.01	32.90	30.34
35	36.29	24.34	29.50	30.52	31.18	30.98	34.57	32.95	30.34
Noiseless	36.33	24.44	29.73	31.37	31.56	30.98	34.78	32.95	30.42
	Barbara								
5	35.15	19.03	25.11	24.40	19.45	23.22	27.73	30.39	24.74
10	35.16	19.34	25.94	24.44	19.80	23.56	27.86	30.50	24.75
15	35.16	19.87	26.37	24.45	19.94	24.17	28.02	30.76	24.77
25	35.21	21.05	27.35	24.45	20.03	24.04	27.94	30.87	24.77
35	35.27	21.32	28.04	24.46	20.07	24.28	28.01	30.87	24.80
Noiseless	35.28	21.66	28.29	24.33	21.60	24.30	28.17	30.87	24.80
	Peppers								
5	34.61	20.11	26.21	30.77	31.33	29.87	32.11	32.01	29.79
10	34.61	20.49	26.73	30.77	31.71	30.01	32.49	32.17	29.73
15	34.68	21.61	27.01	30.92	31.99	30.55	33.41	32.30	29.76
25	34.68	23.20	28.35	30.92	32.66	31.60	33.37	32.38	29.80
35	34.58	23.44	29.22	30.92	32.68	31.60	33.89	32.38	29.80
Noiseless	34.74	23.47	29.35	31.44	32.70	31.64	33.99	32.38	29.86
	Mandrill								
5	23.41	19.35	19.76	21.31	16.27	20.22	20.41	21.45	20.55
10	23.39	19.74	20.01	21.29	16.35	20.94	21.33	21.60	20.62
15	23.41	20.20	20.69	21.31	17.07	21.77	22.01	21.76	20.62
25	23.42	20.67	21.27	21.33	17.67	21.90	22.48	21.80	20.62
35	23.42	20.99	22.03	20.81	18.21	21.93	22.60	21.84	20.63
Noiseless	23.43	21.17	22.25	21.09	19.22	21.93	22.78	21.89	20.69
	Goldhill								
5	32.04	21.09	24.81	28.36	28.54	28.02	28.77	31.27	29.79
10	32.07	21.86	25.03	28.37	28.84	28.43	29.03	31.30	29.81
15	32.10	22.44	25.84	28.37	28.96	28.89	30.48	31.44	29.81
25	32.06	23.50	26.40	28.37	29.70	28.92	30.33	31.59	29.81
35	32.06	23.61	26.66	28.37	29.75	28.92	30.67	31.60	29.81
Noiseless	32.10	23.93	26.90	28.95	29.86	28.92	30.84	31.60	29.86
	Cameraman								
5	41.20	21.01	31.13	30.07	33.02	29.17	36.77	34.98	31.57
10	41.40	21.27	31.36	30.19	33.21	29.83	36.98	35.20	31.62
15	41.48	22.11	32.07	30.06	33.42	30.53	37.40	35.26	31.63
25	41.49	22.98	33.24	30.09	33.44	30.47	37.37	35.29	31.64
35	41.49	23.67	33.86	30.20	33.95	30.64	37.44	35.29	31.64
Noiseless	41.49	24.11	33.95	30.07	34.12	30.67	37.49	35.29	31.66
	Boat								
5	32.39	20.15	25.46	27.00	27.65	27.14	28.83	28.67	27.29
10	32.44	20.44	25.63	27.01	27.78	27.45	28.97	28.75	27.32
15	32.44	21.21	26.29	27.02	28.08	27.93	29.25	28.97	27.32
25	32.44	21.99	26.99	27.02	28.21	28.02	29.76	29.05	27.32
35	32.44	22.78	27.68	27.02	28.57	28.00	30.49	29.05	27.34
Noiseless	32.48	23.18	27.97	27.02	28.94	28.02	30.66	29.07	27.38



(a) Brain image (b) SparseMRI (ℓ_l + Wavelet) (c) DLMRI (d) IST + NLDR

Figure 5.6: Axial T2 Weighted Brain image CS recovery using 4 fold downsampling (25% measurements). (a) Original image; (b) reconstruction using SparseMRI, PSNR=31.84dB; (c) DLMRI, PSNR=34.75dB; (d) NLDR (IST), PSNR=34.86dB.

5.4.2 Recovery Performance on MRI Data

In this experiment, the performance of the proposed NLDR algorithm is demonstrated on the real MRI Brain image data with a variety of undersampling factors. The image used is *in vivo* MR scans of size 512×512 from ARS (2009). The CS data acquisition is simulated by downsampling the 2D discrete Fourier transform of the Brain image. Our result is compared with a leading CS MRI method by Lustig et al. Lustig et al. (2007) (denoted as SparseMRI) and the dictionary learning based recovery algorithm called DLMRI Ravishankar and Bresler (2011). The SparseMRI method is to minimize both the l_1 norm and the TV norm of the image in the wavelet domain. The DLMRI uses K-SVD dictionary learning methods and tries to find the best sparse representation of the image for CS recovery. We adopt the same 2D random sampling scheme as in Ravishankar and Bresler (2011) with $2.5, 4, 6, 8, 10, 20$ fold downsampling. Here, for the k fold downsampling, it is equivalent to the measurement ratio (i.e., $\frac{m}{n}$) of $\frac{1}{k}$.

In Fig. 5.6, we present the CS recovery result on the Brain image with 4 fold downsampling. We observe that NLDR (based on IST) gives the best recovery result in PSNR which is 34.86dB. The DLMRI method also has a close PSNR of 34.75dB. We also demonstrate in Fig. 5.7 the comparison with various downsampling factors. When the downsampling factor is within 10 fold, the NLDR performance is comparable to that of

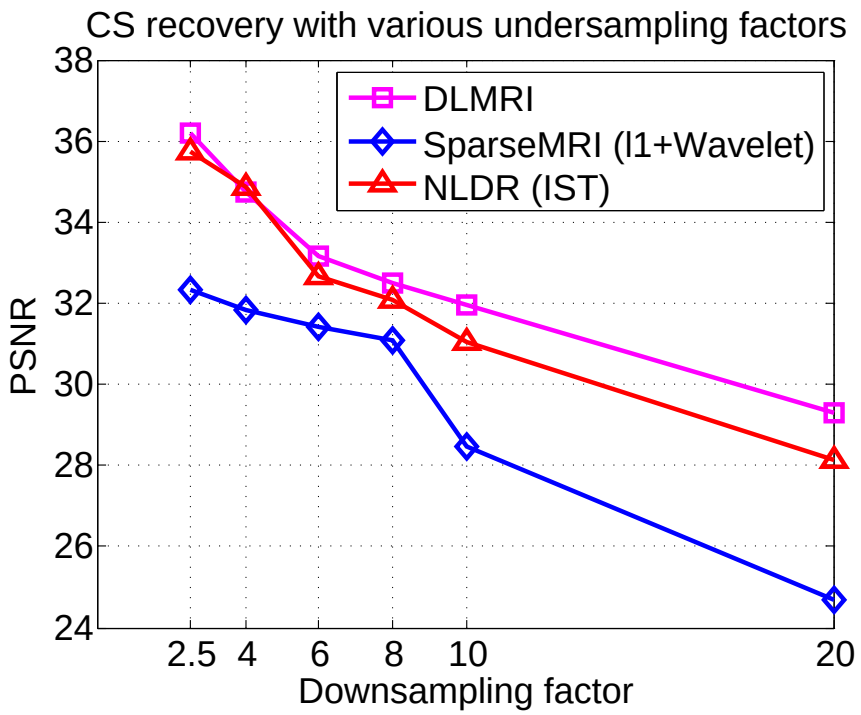


Figure 5.7: CS recovery results comparison with various downsampling factors.

the DLMRI method, while the SparseMRI generates much lower recovery PSNRs. When the downsampling factor reaches 20, the reconstructed image PSNR drops drastically for SparseMRI, and the NLDR is 1.15dB less than DLMRI PSNR. The reason that DLMRI performs better than NLDR is that, DLMRI uses dictionary learning to find the best sparse representation basis for each single test image. NLDR naturally utilizes a general DCT basis to represent the original test image. As a universal basis, it is not chosen to be optimal for one image. The DLMRI also has its disadvantages-the recovery time usually lasts for hours for a large image as the dictionary learning takes a lot of computations. The computation time needed for NLDR is at the same level as those of TVAL3 and BCS-SPL. For all our test images of size 512×512, NLDR takes, on average, about 10 minutes to finish on a Laptop PC.

5.5 Summary

This chapter presented a CS image recovery algorithm based on Douglas-Rachford Splitting with nonlocal estimation. The proposed **NLDR** algorithm first used the iterative thesholding algorithm to obtain the intermediate image reconstruction result. Then a nonlocal estimation step was applied to the reconstructed image to improve the recovery performance. In the nonlocal estimation step, we reformulated the patches estimation as patch denoising problem using low-rank matrix approximation. We proposed a Douglas-Rachford splitting method to solve the CS recovery problem with the nonlocal estimation. Experimental results validated the performance of the proposed NLDR algorithm in both PSNR and visual perception on standard test images with both noiseless and noisy settings. NLDR outperformed the state-of-the-art CS recovery algorithms and showed it can be applied on top of existing recovery algorithms to further improve the recovery performance. Experiments on MRI data also demonstrated it is practical for real applications with competing results.

Chapter 6

Recursive Low-rank and Sparse Recovery of Surveillance Video using Compressed Sensing

This chapter focuses on surveillance video processing using Compressed Sensing (CS). The CS measurements are used for recovery of the video frame into a low-rank background component and sparse component that corresponds to the moving object. The spatial and temporal low-rank features of the video frame, e.g., the nonlocal similar patches within the single video frame and the low-rank background component residing in multiple frames, are successfully exploited. We propose rLSDR that consists of three major components. First we develop an efficient single frame CS recovery algorithm, called NLDR, that operates on the nonlocal similarity patches within each frame to solve the low-rank optimization problem with the CS measurements constraint using Douglas-Rachford splitting method. Second, after obtaining a few NLDR recovered frames as training, a fast bilateral random projections (BRP) scheme is adopted for quick low-rank background initialization. Third, rLSDR then incorporates real-time single video frame to recursively recover the sparse component and update the background, where the proposed NLDR algorithm can also be used here for sparse component estimation. Experimental results

on standard surveillance videos demonstrate that NLDR performs the best for single frame CS recovery compared with the state-of-the-art and rLSDR could successfully recover the background and sparse object with less resource consumption.

6.1 Introduction

Smart Camera Networks (SCNs) have been traditionally used in surveillance and security applications Rinner and Wolf (2008), where a plural of cameras are deployed and networked with each other through wireless connections. The cameras transmit surveillance videos to a processing center where the videos are processed and analyzed. Of particular interest in surveillance video processing is the ability to detect anomalies and moving objects in a scene automatically and quickly Jiang et al. (2012).

Detection of moving objects is a well-established problem that has received a great deal of attention from the research community Tian et al. (2005); Yilmaz et al. (2006). Classical techniques often involve performing background subtraction, object segmentation, and sequential estimation for the objects of interest Stauffer and Grimson (1999). Another approach is based on low-rank and sparse modeling Candès et al. (2011), where the background is modeled by a low rank matrix, and the moving objects are identified by a sparse component (e.g., Zhou and Tao (2011); Bouwmans and Zahzah (2014)). These methods require all pixels of surveillance video to be captured, transmitted and analyzed.

However, due to the growing availability of cheap, high-quality cameras, the amount of data generated by the video surveillance system has grown drastically. The challenge arises on how to process, store or transmit such enormous amount of data under real-time and bandwidth constraints Wu et al. (2008). At the same time, most of the data is uninteresting due to inactivity (e.g., background). There is a high risk of the network being overwhelmed by the mostly uninteresting data that prevents timely detection of anomalies and moving objects Jiang et al. (2012). Thus, it is imperative for SCNs to transmit a small amount of data with enough information for reliable detection and tracking of moving objects or anomalies. The theory of Compressed Sensing (CS) Donoho (2006); Candès et al. (2006)

allows us to address this problem. Under certain conditions related to sparse representations of video frames, CS can effectively reduce the amount of data collected by the system while retaining the ability to faithfully reconstruct the information of interest.

When applying CS on the surveillance video acquisition, the CS measurements are transmitted to the processing center. The original pixel values of the video frame are unknown, and therefore, the traditional background subtraction Stauffer and Grimson (1999), low-rank and sparse modeling Zhou and Tao (2011); Bouwmans and Zahzah (2014) cannot be applied directly. A direct approach is to recover the video frame first and then apply the traditional techniques. The drawbacks are two-fold: First, the CS recovery algorithm does not take advantage of special characteristics of surveillance video in which a well defined, relatively static background exists Jiang et al. (2012). Second, in many applications, one would like to quickly obtain the background and object estimates on-the-fly, rather than in a batch fashion, it is also desirable to incorporate real-time sample-by-sample (i.e., streaming) frame to update the recovery result.

In this chapter, we propose a method named **rLSDR** (recursive Low-rank and Sparse estimation through Douglas-Rachford splitting) for segmentation of background by recursively estimating low-rank and sparse components in the reconstructed surveillance video frames from CS measurements. As in Candès et al. (2011), the low-rank component is the background, and the sparse component identifies moving objects. In this method, First, the proposed**NLDR** (NonLocal Douglas-Rachford splitting) in the previous chapter is adopted to solve the single frame CS recovery problem. NLDR takes advantage of self-similarities within the single frame and models it as a low-rank matrix. Second, after obtaining a few NLDR recovered frames as training, a fast bilateral random projections (BRP) scheme is adopted for quick low-rank background initialization. Third, we propose a scheme to recursively estimate the low-rank background part and sparse object part in a "frame-by-frame" fashion, where the proposed NLDR algorithm can also be used for sparse component estimation.

The rest of the chapter is organized as follows. Section 6.2 presents some related work on background subtraction, low-rank and sparse modeling. Section 6.3 discusses

the problem formulation. Section 6.4 introduces the proposed rLSDR algorithm. The performance evaluation on three videos is given in Section 6.5. Finally, we conclude in Section 6.6.

6.2 Related Work

In Oliver et al. (2000), the authors first proposed to use Principal Component Analysis (PCA) to model the background. Object detection is then achieved by thresholding the difference between generated background image and the current image. PCA provides a robust model of the probability distribution function of the background, but not the moving objects Bouwmans and Zahzah (2014). The work in De la Torre and Black (2001); De La Torre and Black (2003) improved classical PCA with respect to outlier and noise, yielding the field of robust PCA. Later on, this was advanced by very recent works based on the idea that the data matrix X can be decomposed into two components such that $X = L + S$, where L is a low-rank matrix and S is a matrix that is sparse. This decomposition can be obtained by robust Principal Component Analysis (rPCA) solved via Principal Component Pursuit (PCP) Wright et al. (2009); Candès et al. (2011). While PCP is an elegant solution, it suffers some practical limitations. First, it requires the number of nonzero pixels in the moving objects to be small (i.e., the object should be exact sparse), this may not hold if there are large size or multiple moving objects. Second, PCP is a batch method and computationally expensive, it would be more useful to quickly obtain the low-rank matrix and the sparse matrix in an incremental way for each new frame and gradually improve the estimates.

The bandwidth challenge in the network of surveillance cameras was addressed by CS. The author in Jiang et al. (2012) proposed to recovery the CS measurements into low-rank and sparse components and adopted the alternative direction method (ADM) for solving the optimization problem in a batch fashion. ReProCS Qiu and Vaswani (2011), an algorithm that addressed the limitation in PCP by recursively projecting the CS recovered frame to the subspace perpendicular to the subspace spanned by the PC component to nullify

the background. It then recovers the sparse component by solving a noisy CS problem. Although robust and can be implemented on-the-fly, it needed to acquire the high accurate estimation of background PC component (e.g., through PCA) to successfully nullify the low-rank part in the data matrix. The performance could easily be affected by the training process to obtain the PC component.

6.3 Problem Formulation

We consider a video sequence consisting of a number of frames (i.e., images). Let $x_t \in \mathbb{R}^{m \times n}$ be a vector formed from pixels of frame t of the video sequence, for $t = 1, \cdots, T$, where T is the total number of frames, m and n are the dimensions of each frame. The current frame x_t, is an overlay of foreground image, F_t, over the background image, B_t. The goal is to recover both F_t and B_t at each time frame t in real-time. Many foreground pixels are zero and hence F_t is a sparse matrix. We let T_t denote the foreground support set, i.e., $T_t := \{i : (F_t)_i \neq 0\}$. Thus,

$$(x_t)_i := \begin{cases} (F_t)_i & \text{if } i \in T_t \\ (B_t)_i & \text{otherwise} \end{cases} \tag{6.1}$$

where i is the entry indices corresponding to the raster scan order in the data matrix.

Let Φ_t be an $M \times N$ measurement matrix, where $M < N$. The measurement matrix Φ_t may be chosen as a random Gaussian or Fourier Scrambled matrices Candès et al. (2006). We choose the Φ_t as a sparse binary measurement matrix based on the expander graph Li and Qi (2013a,b) which serves the same purpose as the traditional CS measurement matrices but further reduces the computation (e.g., only addition operations on cameras) and the amount of measurements transmitted.

Assume, each frame can be re-arranged as an $N \times 1$ vector (i.e., $N = m \times n$). The single frame CS measurements from the video are defined as

$$y_t = \Phi_t x_t \tag{6.2}$$

where y_t is a vector of length M. To recover x_t from y_t, first y_t is sparsely coded with respect to the basis $\Psi \in \mathbb{R}^{N \times N}$ by solving the following minimization problem

$$\hat{\alpha} = \arg\min_{\alpha} \{\|y_t - \Phi_t \Psi \alpha\|_2^2 + \lambda_\alpha \|\alpha\|_1\} \tag{6.3}$$

and then x_t is reconstructed by $\hat{x}_t = \Psi \hat{\alpha}$.

Following the same notation in Qiu and Vaswani (2011). Let μ_t denote the mean background image and let $L_t := B_t - \mu_t$, and $M_t := \hat{x}_t$ be the frame t reconstructed from CS recovery algorithm (e.g., Daubechies et al. (2004)) with mean subtracted. By defining

$$(S_t)_i := \begin{cases} (F_t - B_t)_i = (F_t - \mu_t - L_t)_i & \text{if } i \in T_t \\ 0 \quad \text{otherwise} \end{cases} \tag{6.4}$$

we can formulate as a problem of recovering L_t and S_t from

$$M_t := S_t + L_t \tag{6.5}$$

Here, S_t is a sparse vector with support set, T_t, and L_t are dense matrices lie in a slowly changing low dimensional subspace.

6.4 The Proposed Algorithm

The proposed rLSDR algorithm consists of three major components: (1) single frame recovery from CS measurement, (2) low-rank component initialization, and (3) recursive recovery of sparse component and update of the low-rank component.

6.4.1 Single Frame Recovery

For the single frame recovery, we adopt the aforementioned NLDR algorithm in Chapter 5 where each video frame is treated as an image for the CS recovery task.

6.4.2 Low-rank Component Initialization

After denoising the CS recovered frame using NLDR, the second component of the proposed rLSDR algorithm is to estimate the low-rank background image based on the first few video frames (e.g., around 50). In order to estimate the background, a common approach would be applying SVD on the recovered video frames to obtain its low-rank approximation. However, performing SVD operation is usually very time-consuming, especially for large resolution video frames which hinders the "on-the-fly" estimation. The other drawback is that, often we just need a rough estimation of the low-rank component which can later be refined upon receiving new video frames.

In this work, we adopt the bilateral random projections (BRP) based low-rank approximation with closed-form solution. Given r bilateral random projections of a $p \times q$ dense matrix X (w.l.o.g, $p \geq q$), i.e., $U = XA_1$ and $V = X^T A_2$, where $A_1 \in \mathbb{R}^{q \times r}$ and $A_2 \in \mathbb{R}^{p \times r}$ are random matrices,

$$L = U(A_2^T U)^{-1} V^T \tag{6.6}$$

is a fast rank-r approximation of X. The L in Eq. (6.6) has been proposed in Fazel et al. (2008) as a recovery of a rank-r matrix X from U and V, where A_1 and A_2 are independent Gaussian or subsampled Fourier random matrices. It was later advanced by Zhou et al. in Zhou and Tao (2011) showed that L is a tight rank-r approximation to a full rank matrix X, when A_1 and A_2 are correlated random matrices updated from V and U, respectively.

The computation of L includes an inverse of an $r \times r$ matrix and three matrix multiplications. Thus, for a dense X, $2pqr$ floating-point operations (flops) are required

to obtain BRP, $r^2(2q+r)+pqr$ flops are required to obtain L. The computational cost is much less than SVD based approximation.

6.4.3 Recursive Sparse Recovery and Low-rank Updates

After the low-rank background component L_t has been estimated, we now proceed to the third component of rLSDR, where we recursively update the sparse component and background estimation upon receiving the CS measurements y_{t+1} of new frame x_{t+1}. The CS recovered new frame $\hat{x}_{t+1}$ is obtained using the proposed NLDR algorithm. The sparse recovery problem to find S_{t+1} can be formulated as follows

$$\begin{aligned} \min_{S_{t+1}} \frac{1}{2}\|\hat{x}_{t+1} - L_t - S_{t+1}\|_2^2 + \lambda_s \|S_{t+1}\|_1 \\ \text{s.t.} \quad \|y_{t+1} - \Phi_{t+1}(L_t + S_{t+1})\|_2^2 \leq \epsilon \end{aligned} \tag{6.7}$$

where L_t is estimated background at the frame t. The only unknown in Eq. (6.7) is S_{t+1}. Again it can be solved using NLDR algorithm to estimate $\hat{S}_{t+1}$.

After the sparse component is obtained using Eq. (6.7), the corresponding low-rank background component at $t+1$ frame can be calculated as $L_{t+1} = \hat{x}_{t+1} - \hat{S}_{t+1}$. This single frame background will be incorporated into Eq. (6.6) to update L, which is the initial trained background matrix. The final low-rank background at frame $t+1$ is then estimated as $\hat{L}_{t+1} = L(t+1)$ from the output of Eq. (6.6).

We summarize the proposed rLSDR in Algorithm 3.

6.5 Experimental Results

We apply **rLSDR** to two surveillance videos [*], i.e., *Restaurant* and *Curtain*. *Curtain* consists of 304 frames each of dimension 64 $\times$ 80. *Restaurant* contains 200 frames with dimension 144 $\times$ 176. We first experiment on the single frame recovery result by

[*] http://perception.i2r.a-star.edu.sg/bk_model/bk_index.html

Algorithm 3: rLSDR Algorithm

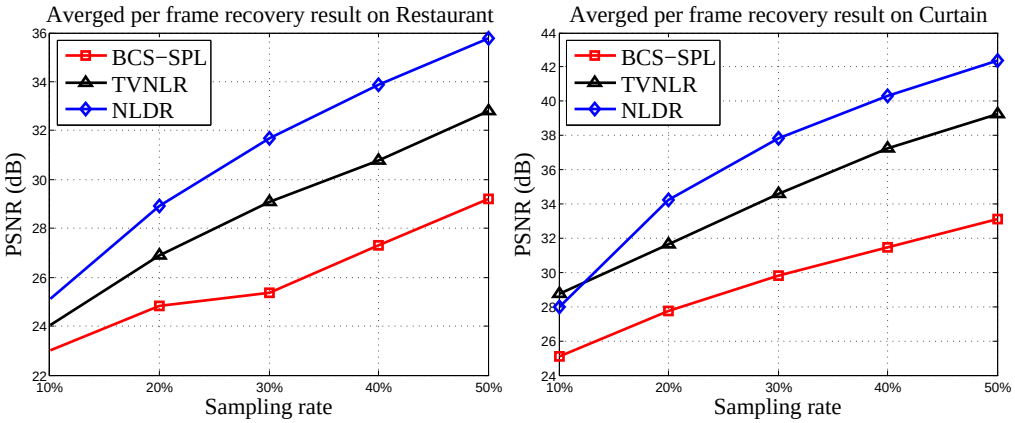
Input:

- ▶ CS Measurement matrix $\Phi_t \in \mathbb{R}^{M\times N}$
- ▶ Measurements data matrix $y_t \in \mathbb{R}^{M\times p}$
- ▶ Initialize random matrices A_1, A_2
- ▶ Number of training frames trn.

Output:

- ▶ CS recovered frames $\hat{x} \in \mathbb{R}^{N\times p}$,
- ▶ Background and object estimate $\hat{L}, \hat{S}$.

Step 1: Initial frame recovery
for $i = 1, \cdots,$ trn **do**
  $X(1 : \text{trn}) \leftarrow$ **NLDR**(y_i)
end for
Step 2: Background initialization
Estimate L using Eq. (6.6)
Step 3: Recursive update L and S
for $t =$ trn, $\cdots,$ p **do**
  Frame recovery: $\hat{x}_{t+1} \leftarrow$ **NLDR**(y_{t+1})
  Sparse est.: Solve Eq. (6.7) for $\hat{S}_{t+1}$ using **NLDR**
  Calculate L_{t+1}: $L_{t+1} = \hat{x}_{t+1} - \hat{S}_{t+1}$, update Eq. (6.6)
  Background est.: $\hat{L}_{t+1} = L(t+1)$
end for
return $\hat{x}, \hat{L}, \hat{S}$



(a) (b)

Figure 6.1: Averaged per frame recover result comparison on (a) **Restaurant** (b) **Curtain**.

Figure 6.2: First column: original *Restaurant* video frames at t = 70, 116, 140. Second column: frame recovered by **NLDR** with 30% measurements. Next 2 columns: background and object estimated by **rLSDR**.

comparing **NLDR** with two popular CS image recovery algorithms, BCS-SPL Mun and Fowler (2009) and TVNLR Zhang et al. (2013).

The block-based image patch is of size 6×6. We set the number of similar patches q in the nonlocal estimation step as 45. We use the scrambled Fourier matrix as the CS measurement matrix Φ and DCT matrix as the basis Ψ to represent the original image in the initial IST recovery. The parameter is selected as $\mu = 1$ for DR iteration and $\lambda_f = \frac{c_i}{\max(s_i)}$ for each iteration where $c_i = C_0 * \epsilon, 0 < \epsilon < 1$ and C_0 is a constant.

Fig. 6.1 shows the averaged per frame recover result of NLDR compared with BCS-SPL and TVNLR using the PSNR metric. Generally, NLDR outperforms the state-of-the-art CS image recovery algorithm in the two video frames.

We then conduct experiments to compare the rLSDR on background and object estimation. For each video sequence, a number of frames, 150 for *Curtain* and 50 for *Restaurant*, are selected as the training frames to initialize the background.

(a) Original (b) Recovered (c) **rLSDR** (d) **rLSDR** (e) **PCP** (f) **PCP** (g) **ReProCS** (h) **ReProCS**

Figure 6.3: First column: original *Curtain* video frames at t = 65, 103, 140. Second column: frame recovered by **NLDR** with 30% measurements. Next 6 columns: background and object estimated by **rLSDR**, **PCP** and **ReProCS**.

Fig. 6.2 shows the CS recovered frame on *Restaurant* with background and object extracted. We also compare the result with **PCP** Candès et al. (2011) and **ReProCS** Qiu and Vaswani (2011) in Fig. 6.3 where the NLDR recovery video frames are used as the batch input. rLSDR could successful recover the background and the object and performs better than PCP, while having similar result as ReProCS. Compared with ReProCS, rLSDR requires much less initial training frames and thus less resource consumptions.

6.6 Summary

In this chapter, we presented **rLSDR**, a CS-based surveillance video processing algorithm to recursively estimate the low-rank background and sparse object. The spatial and temporal low-rank features of the video frame were successfully exploited. Capitalized on the self-similarities within each spatial frame, we proposed **NLDR** for the single frame CS recovery that had high recovery PSNR under various sampling rates compared with the-state-of-art recovery algorithm. We proposed rLSDR that recursively estimates the background through efficient bilateral random projection (BPR). Experimental results on three standard surveillance videos showed that NLDR performs best for CS frame

recovery and rLSDR could successfully recovery the background and sparse object with less resource consumption.

Chapter 7

Conclusions and Future Work

In this dissertation, the issues of applying the compressed sensing technique in resource-constrained environments was studied. We focused on the two major aspects of compressed sensing (i.e., sensing mechanism design and recovery algorithm) in mobile sensing platform and visual sensor network platform. For the mobile sensing platform, we proposed the Compressed Phone Sensing (CPS) framework. The proposed CPS framework consisted of two components, the sensing scheduling and the sample scheduling. In the sensing scheduling component, the pattern-based activity was first defined and modeled based on the "pattern matrix", we then proposed two different sensing scheduling approaches: model-based and CS-based sensing scheduling, where CS was applied in scheduling to turn on/off smartphone sensors. In the sample scheduling component, when the sensors were turned on, CS was then adopted to schedule on how to sense samples. In order to achieve this, we first designed the sparse binary measurement matrix. Then a light-weight randomized CS scheme was proposed for activity sensing that results in only addition operations for the resource-limited smartphones. With CS being its core and applied in two layer of activity sensing with different resolution, CPS framework had hence cohesively achieves energy efficiency without sacrificing sensing accuracy. A case study on driving activity sensing showed that CPS framework can have, on average, the sensing scheduling accuracy about 83.92% but with 62.86% less overall energy consumption as compared

to the continuous sensing. For the visual sensor network platform, we first proposed a single frame recovery algorithm, NLDR, that utilized the nonlocal similarity patches within each frame to solve the low-rank optimization problem with the CS measurements constraint using Douglas-Rachford splitting method. We further extended NLDR algorithm for object detected in surveillance video processing and proposed rLSDR (recursive Low-rank and Sparse estimation through Douglas-Rachford splitting) . rLSDR then incorporated real-time single video frame to recursively recover the sparse component and update the background. Experimental results on standard surveillance videos demonstrated that NLDR performed the best for single frame CS recovery compared with the state-of-the-art and rLSDR could successfully recover the background and sparse object with less resource consumption.

7.1 Future Work

There are one major project on the future work for this dissertation research. The project is corresponding to the first component of the dissertation research on sensing mechanism design. We detail here the abstract, background and problem formulation here.

7.1.1 1-bit CS with the *stable random projection* measurement

A. 1-bit CS

Quantization is an indispensable part of digital signal processing and digital communications systems. To incorporate CS methods in these systems, it is thus necessary to analyze and evaluate them considering the effect of measurement quantization. In the recent years there is a growing interest in quantized CS in the literature Zymnis et al. (2010); Laska et al. (2011), particularly the extreme case of quantization to a single bit, known as 1-bit Compressive Sensing Boufounos and Baraniuk (2008), where only the sign of the linear measurements are recorded. The quantization process is useful in the bandwidth-limited environment.

Measurement Model

Each measurement is the sign of the inner product of the sparse signal with a measurement vector ϕ_i:

$$y_i = \text{sign}(\langle \phi_i, x \rangle) \tag{7.1}$$

It follows that the product of each quantized measurement with the measurement is always non-negative:

$$y_i \text{sign}(\langle \phi_i, x \rangle) \geq 0. \tag{7.2}$$

The measurements are compactly expressed using:

$$y = \text{sign}(\Phi x), \tag{7.3}$$

where y is the vector of measurements, Φ is a matrix representing the measurement system and the 1-bit quantization function sign$(\cdot)$ is applied element-wise to the vector Φx. Several algorithms have been proposed to solve the 1-bit CS problem, including BIHT Jacques et al. (2011); Yan et al. (2012)

B. maximally-skewed $\alpha-$stable random projection

Li, et. al Li et al. (2013) proposed to design a CS measurement matrix based on $\alpha-$stable random projection, called compressed counting (CC). Specifically, their results showed that by using maximally-skewed $\alpha-$stable random projection, the CS recovery procedure is computationally very efficient in that it requires only one linear scan of the coordinates.

Maximally-Skewed Stable Distributions and Recovery

The design matrix s_{ij} is sampled from an α-stable maximally-skewed distribution, denoted by $S(\alpha, 1, 1)$, where the first "1" denotes maximal skewness and the second "1" denotes unit scale. If a random variable $Z \sim S(\alpha, 1, 1)$, then its characteristic function is

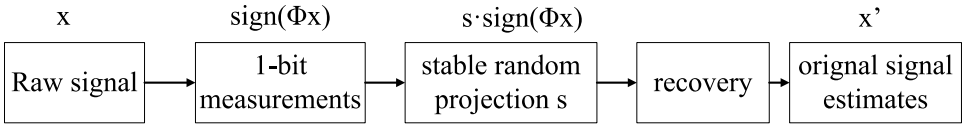


Figure 7.1: 1-bit CS combined with stable random projection measurements.

$$\mathscr{F}_Z(\lambda) = \mathrm{E}\exp\left(\sqrt{-1}Z\lambda\right) = \exp\left(-|\lambda|^\alpha\left(1 - \mathrm{sign}(\lambda)\sqrt{-1}\tan\left(\frac{\pi\alpha}{2}\right)\right)\right), \quad \alpha \neq 1$$

Suppose $s_1, s_2 \sim S(\alpha, 1, 1)$ i.i.d. For any constants $c_1 \geq 0, c_2 \geq 0$, we have $c_1 s_1 + c_2 s_2 \sim S(\alpha, 1, c_1^\alpha + c_2^\alpha)$. More generally, $\sum_{i=1}^N x_i s_i \sim S\left(\alpha, 1, \sum_{i=1}^N x_i^\alpha\right)$ if $s_i \sim S(\alpha, 1, 1)$ i.i.d.

For recovering a nonnegative signal $x_i \geq 0$, $i = 1$ to N, we collect linear measurements $y_j = \sum_{i=1}^N x_i s_{ij}$, $j = 1$ to M, where $s_{ij} \sim S(\alpha, 1, 1)$ i.i.d. At the decoding stage, we estimate the signal coordinate-wise:

$$\hat{x}_{i,min} = \min_{1 \leq j \leq M} y_j / s_{ij} \tag{7.4}$$

The number of measurements M is chosen so that $\sum_{i=1}^N \mathbf{Pr}\left(\hat{x}_{i,min} - x_i \geq \epsilon\right) \leq \delta$ (e.g., $\delta = 0.05$).

Main Result: When $\alpha \in (0,\ 0.5]$, it suffices to use $M = (C_\alpha + o(1))\epsilon^{-\alpha}\left(\sum_{i=1}^N x_i^\alpha\right)\log N/\delta$ measurements so that, with probability $1 - \delta$, all coordinates will be recovered within ϵ additive precision, in one scan of the coordinates. The constant $C_\alpha = 1$ when $\alpha \to 0$ and $C_\alpha = \pi/2$ when $\alpha = 0.5$. In particular, when $\alpha \to 0$, the required number of measurements is essentially $M = K \log N/\delta$, where $K = \sum_{i=1}^N 1\{x_i \neq 0\}$ is the number of nonzero coordinates of the signal.

C. Proposed Ideas

We proposed to combine the 1-bit CS measurement with the *stable random projection* measurement, the diagram is shown in Fig 7.1, where the raw signal of interests is first projected and quantized by the 1-bit CS sensing scheme. Then the 1-bit measurement is further quantized by the *stable random projection* matrix, at the decoding side, the one bit measurement will first be reliably recovered, then the original signal will be restored.

Bibliography

Akimura, D., Kawahara, Y., and Asami, T. (2011). Reducing power consumption of human activity sensing using compressed sensing. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, pages 355–356. ACM. 21, 23

Akimura, D., Kawahara, Y., and Asami, T. (2012). Compressed sensing method for human activity sensing using mobile phone accelerometers. In *Networked Sensing Systems (INSS), 2012 Ninth International Conference on*, pages 1–4. IEEE. 12

ARS (2009). American radiology services. http://www3.americanradiology.com/pls/web1/wwimggal.vmg. [online]. 72

Barakat, W., Saliba, R., and Evans, B. L. (2008). Compressive sensing for multimedia communications in wireless sensor networks. *MDDSP Project Final Report*. xii, 13

Baraniuk, R. (2007). Compressive sensing. *IEEE signal processing magazine*, 24(4). 2, 23

Baraniuk, R. G. (2008). Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 25(2):83–91. 14

behavio (2012). Behav.io. http://www.behav.io/. [Webpage; Accessed on 08/23/2012]. 20

Berinde, R., Gilbert, A., Indyk, P., Karloff, H., and Strauss, M. (2008). Combining geometry and combinatorics: A unified approach to sparse signal recovery. In *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, pages 798–805. IEEE. 23, 33

Boufounos, P. T. and Baraniuk, R. G. (2008). 1-bit compressive sensing. In *Information Sciences and Systems, 2008. CISS 2008. 42nd Annual Conference on*, pages 16–21. IEEE. 88

Bouwmans, T. and Zahzah, E. H. (2014). Robust PCA via principal component pursuit: A review for a comparative evaluation in video surveillance. *Computer Vision and Image Understanding*, 122:22–34. 76, 77, 78

Buades, A., Coll, B., and Morel, J.-M. (2005). A review of image denoising algorithms, with a new one. *Multiscale Modeling & Simulation*, 4(2):490–530. 57

Burke, J., Estrin, D., Hansen, M., Parker, A., Ramanathan, N., Reddy, S., and Srivastava, M. B. (2006). Participatory sensing. In *In: Workshop on World-Sensor-Web (WSW06): Mobile Device Centric Sensor Networks and Applications*, pages 117–134. 2

Campbell, A., Eisenman, S., Lane, N., Miluzzo, E., and Peterson, R. (2006). People-centric urban sensing. In *Proceedings of the 2nd annual international workshop on Wireless internet*, page 18. ACM. 3

Candes, E. and Romberg, J. (2005). ℓ_1-magic: Recovery of sparse signals via convex programming. 24, 29, 39, 48

Candès, E., Romberg, J., and Tao, T. (2006). Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *Information Theory, IEEE Transactions on*, 52(2):489–509. 2, 8, 17, 54, 76, 79

Candes, E. and Tao, T. (2005). Decoding by linear programming. *Information Theory, IEEE Transactions on*, 51(12):4203–4215. 9

Candès, E. and Wakin, M. (2008). An introduction to compressive sampling. *Signal Processing Magazine, IEEE*, 25(2):21–30. 9

Candès, E. J., Li, X., Ma, Y., and Wright, J. (2011). Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11. 76, 77, 78, 85

Candes, E. J., Romberg, J. K., and Tao, T. (2006). Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223. 2

Candès, E. J. and Tao, T. (2006). Near-optimal signal recovery from random projections: Universal encoding strategies? *Information Theory, IEEE Transactions on*, 52(12):5406–5425. 8

Candès, E. J. and Tao, T. (2010). The power of convex relaxation: Near-optimal matrix completion. *Information Theory, IEEE Transactions on*, 56(5):2053–2080. 59, 62

Chen, P., Ahammad, P., Boyer, C., Huang, S.-I., Lin, L., Lobaton, E., Meingast, M., Oh, S., Wang, S., Yan, P., et al. (2008). Citric: A low-bandwidth wireless camera network platform. In *Distributed smart cameras, 2008. ICDSC 2008. Second ACM/IEEE international conference on*, pages 1–10. IEEE. 14

Chen, S., Donoho, D., and Saunders, M. (1998). Atomic decomposition by basis pursuit. *SIAM journal on scientific computing*, 20(1):33–61. 24

Chen, S. S., Donoho, D. L., and Saunders, M. A. (2001). Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159. 9

Chierchia, G., Pustelnik, N., Pesquet-Popescu, B., and Pesquet, J.-C. (2014). A nonlocal structure tensor-based approach for multicomponent image recovery problems. *Image Processing, IEEE Transactions on*, 23(12):5531–5544. 57, 58

Chong, C.-Y. and Kumar, S. P. (2003). Sensor networks: evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256. 4

Chou, C. T., Rana, R., and Hu, W. (2009). Energy efficient information collection in wireless sensor networks using adaptive compressive sensing. In *Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on*, pages 443–450. IEEE. 12

Cioffi-Revilla, C. (2010). Computational social science. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(3):259–271. 2

Combettes, P. L. and Pesquet, J.-C. (2011). Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer. 63

Combettes, P. L. and Wajs, V. R. (2005). Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200. 63

Consolvo, S., McDonald, D., Toscos, T., Chen, M., Froehlich, J., Harrison, B., Klasnja, P., LaMarca, A., LeGrand, L., Libby, R., et al. (2008). Activity sensing in the wild: a field trial of ubifit garden. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1797–1806. ACM. 2, 16

Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. (2007). Image denoising by sparse 3-d transform-domain collaborative filtering. *Image Processing, IEEE Transactions on*, 16(8):2080–2095. 66

Dahl, J., Hansen, P. C., Jensen, S. H., and Jensen, T. L. (2010). Algorithms and software for total variation image reconstruction via first-order methods. *Numerical Algorithms*, 53(1):67–92. 55

Dai, W. and Milenkovic, O. (2009). Subspace pursuit for compressive sensing signal reconstruction. *Information Theory, IEEE Transactions on*, 55(5):2230–2249. 11

Das, T., Mohan, P., Padmanabhan, V. N., Ramjee, R., and Sharma, A. (2010). Prism: platform for remote sensing using smartphones. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 63–76. ACM. 3

Daubechies, I., Defrise, M., and De Mol, C. (2004). An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on pure and applied mathematics*, 57(11):1413–1457. 57, 80

Daubechies, I., Fornasier, M., and Loris, I. (2008). Accelerated projected gradient method for linear inverse problems with sparsity constraints. *Journal of Fourier Analysis and Applications*, 14(5-6):764–792. 57

Davenport, M. A., Duarte, M. F., Eldar, Y. C., and Kutyniok, G. (2011). Introduction to compressed sensing. *Preprint*, 93. 1, 2

Davis, G., Mallat, S., and Zhang, Z. (1994). Adaptive time-frequency decompositions with matching pursuits. *Optical Engineering*, 33. 10

De la Torre, F. and Black, M. J. (2001). Robust principal component analysis for computer vision. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 362–369. IEEE. 78

De La Torre, F. and Black, M. J. (2003). A framework for robust subspace learning. *International Journal of Computer Vision*, 54(1-3):117–142. 78

Dempster, A. P., Laird, N. M., Rubin, D. B., et al. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal statistical Society*, 39(1):1–38. 31

Do, T., Gan, L., Nguyen, N., and Tran, T. (2008). Sparsity adaptive matching pursuit algorithm for practical compressed sensing. In *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*, pages 581–587. IEEE. 11

Dong, W., Shi, G., and Li, X. (2013). Nonlocal image restoration with bilateral variance estimation: a low-rank approach. *Image Processing, IEEE Transactions on*, 22(2):700–711. 59

Dong, W., Shi, G., Li, X., Ma, Y., and Huang, F. (2014a). Compressive sensing via nonlocal low-rank regularization. *Image Processing, IEEE Transactions on*, 23(8):3618–3632. xiii, xiv, 57, 58, 66, 67, 68

Dong, W., Shi, G., Li, X., Ma, Y., and Huang, F. (2014b). Compressive sensing via nonlocal low-rank regularization. *Image Processing, IEEE Transactions on*, 23(8):3618–3632. 57

Donoho, D. (2006). Compressed sensing. *Information Theory, IEEE Transactions on*, 52(4):1289–1306. 2, 17, 54, 76

Donoho, D., Tsaig, Y., Drori, I., and Starck, J. (2012). Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. *Information Theory, IEEE Transactions on*, 58(2):1094–1121. 10

Douglas, J. and Rachford, H. (1956). On the numerical solution of heat conduction problems in two and three space variables. *Transactions of the American mathematical Society*, 82(2):421–439. 62

Eagle, N. and Pentland, A. (2006). Reality mining: sensing complex social systems. *Personal and ubiquitous computing*, 10(4):255–268. 2

Economist, T. (2010). The data deluge. http://www.economist.com/node/15579717. [Webpage; Accessed on 04/09/2014]. 1

Elad, M. and Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. *Image Processing, IEEE Transactions on*, 15(12):3736–3745. 56, 57

Evans, J. and Krishnamurthy, V. (2001). Optimal sensor scheduling for hidden markov model state estimation. *International Journal of Control*, 74(18):1737–1742. 30

Fazel, M., Candes, E., Recht, B., and Parrilo, P. (2008). Compressed sensing and robust recovery of low rank matrices. In *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*, pages 1043–1047. IEEE. 81

Gan, L. (2007). Block compressed sensing of natural images. In *Digital Signal Processing, 2007 15th International Conference on*, pages 403–406. IEEE. 54

Gan, L., Do, T. T., and Tran, T. D. (2008). Fast compressive imaging using scrambled block hadamard ensemble. *European Signal Processing Conference 2008*. 14

Gilbert, A. and Indyk, P. (2010). Sparse recovery using sparse matrices. *Proceedings of the IEEE*, 98(6):937–947. 23

Gonzalez, M., Hidalgo, C., and Barabasi, A. (2008). Understanding individual human mobility patterns. *Nature*, 453(7196):779–782. 20

Han, B., Wu, F., and Wu, D. (2010). Image representation by compressive sensing for visual sensor networks. *Journal of Visual Communication and Image Representation*, 21(4):325–333. 13

Huynh, T., Fritz, M., and Schiele, B. (2008). Discovery of activity patterns using topic models. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 10–19. ACM. 20

Jacques, L., Laska, J. N., Boufounos, P. T., and Baraniuk, R. G. (2011). Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors. *CoRR*, abs/1104.3160. 89

Ji, H., Liu, C., Shen, Z., and Xu, Y. (2010). Robust video denoising using low rank matrix completion. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1791–1798. 59

Jiang, H., Deng, W., and Shen, Z. (2012). Surveillance video processing using compressive sensing. *Inverse Problems & Imaging*, 6(2). 76, 77, 78

Jiang, Y., Li, D., Yang, G., Lv, Q., and Liu, Z. (2011). Deliberation for intuition: a framework for energy-efficient trip detection on cellular phones. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 315–324. ACM. 2, 16

Jiang, Y., Qiu, H., McCartney, M., Halfond, W. G., Bai, F., Grimm, D., and Govindan, R. (2014). Carlog: a platform for flexible and efficient automotive sensing. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, pages 221–235. ACM. 21

Kaelbling, L. P., Littman, M. L., and Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285. 32

Kim, E., Helal, S., and Cook, D. (2010). Human activity recognition and pattern discovery. *Pervasive Computing, IEEE*, 9(1):48–53. 17

Kotelnikov, V. A. (1933). On the carrying capacity of the ether and wire in telecommunications. In *Material for the First All-Union Conference on Questions of Communication, Izd. Red. Upr. Svyazi RKKA, Moscow*. 1

l1magic (2006). Software ℓ_1-magic. http://users.ece.gatech.edu/~justin/l1magic/. [Webpage; Accessed on 08/23/2014]. 65

Lane, N., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., and Campbell, A. (2010). A survey of mobile phone sensing. *Communications Magazine, IEEE*, 48(9):140–150. 2, 3, 16

Lane, N., Xu, Y., Lu, H., Campbell, A., Choudhury, T., and Eisenman, S. (2011). Exploiting social networks for large-scale human behavior modeling. *Pervasive Computing, IEEE*, 10(4):45–53. 17

Laska, J. N., Boufounos, P. T., Davenport, M. A., and Baraniuk, R. G. (2011). Democracy in action: Quantization, saturation, and compressive sensing. *Applied and Computational Harmonic Analysis*, 31(3):429–443. 88

Li, C., Yin, W., and Zhang, Y. (2009a). TVAL3: TV minimization by augmented lagrangian and alternating direction algorithms. http://www.caam.rice.edu/~optimization/L1/TVAL3/. [Online]. xiii, 54, 65, 67

Li, P., Zhang, C.-H., and Zhang, T. (2013). Compressed counting meets compressed sensing. *arXiv preprint arXiv:1310.1076*. 89

Li, S. and Qi, H. (2013a). Distributed data aggregation for sparse recovery in wireless sensor networks. In *Proceedings of the 9th IEEE international conference on Distributed Computing in Sensor Systems*, pages 62–69. 4, 33, 79

Li, S. and Qi, H. (2013b). Pattern-based compressed phone sensing. In *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*, pages 169–172. 79

Li, Y., Krakow, L. W., Chong, E. K., and Groom, K. N. (2009b). Approximate stochastic dynamic programming for sensor scheduling to track multiple targets. *Digital Signal Processing*, 19(6):978–989. 30

Li, Z., Zhu, Y., Zhu, H., and Li, M. (2011). Compressive sensing approach to urban traffic sensing. In *Distributed Computing Systems (ICDCS), 2011 31st International Conference on*, pages 889–898. IEEE. 2, 16

Liu, J., Priyantha, B., Hart, T., Ramos, H. S., Loureiro, A. A., and Wang, Q. (2012). Energy efficient gps sensing with cloud offloading. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, pages 85–98. ACM. 21

Louchet, C. and Moisan, L. (2011). Total variation as a local filter. *SIAM Journal on Imaging Sciences*, 4(2):651–694. 55

Luo, C., Wu, F., Sun, J., and Chen, C. W. (2009). Compressive data gathering for large-scale wireless sensor networks. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, pages 145–156. ACM. 12

Lustig, M., Donoho, D., and Pauly, J. M. (2007). Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic resonance in medicine*, 58(6):1182–1195. 54, 72

Mallat, S. and Zhang, Z. (1993). Matching pursuits with time-frequency dictionaries. *Signal Processing, IEEE Transactions on*, 41(12):3397–3415. 10

Meier, L., Peschon, J., and Dressler, R. M. (1967). Optimal control of measurement subsystems. *Automatic Control, IEEE Transactions on*, 12(5):528–536. 30

Miluzzo, E., Cornelius, C., Ramaswamy, A., Choudhury, T., Liu, Z., and Campbell, A. (2010). Darwin phones: the evolution of sensing and inference on mobile phones. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 5–20. ACM. 20

Miluzzo, E., Lane, N., Fodor, K., Peterson, R., Lu, H., Musolesi, M., Eisenman, S., Zheng, X., and Campbell, A. (2008). Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 337–350. ACM. 2, 16

Misra, A. and Lim, L. (2011). Optimizing sensor data acquisition for energy-efficient smartphone-based continuous event processing. In *Mobile Data Management (MDM), 2011 12th IEEE International Conference on*, volume 1, pages 88–97. IEEE. 51

Mohan, P., Padmanabhan, V., and Ramjee, R. (2008). Nericell: rich monitoring of road and traffic conditions using mobile smartphones. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 323–336. ACM. 2, 16

Mun, M., Reddy, S., Shilton, K., Yau, N., Burke, J., Estrin, D., Hansen, M., Howard, E., West, R., and Boda, P. (2009). Peir, the personal environmental impact report, as a platform for participatory sensing systems research. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*, pages 55–68. ACM. 2, 16

Mun, S. and Fowler, J. E. (2009). Block compressed sensing of images using directional transforms. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 3021–3024. IEEE. xiii, 54, 65, 67, 84

Nardi, B. (1996). *Context and consciousness: activity theory and human-computer interaction*. The MIT Press. 24

Nath, S. (2012). ACE: exploiting correlation for energy-efficient and continuous context sensing. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pages 29–42. ACM. 21

Nawaz, S. and Mascolo, C. (2014). Mining users' significant driving routes with low-power sensors. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, pages 236–250. ACM. 21

Needell, D. and Tropp, J. (2009). Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321. 10, 11

Nyquist, H. (1928). Certain topics in telegraph transmission theory. *American Institute of Electrical Engineers, Transactions of the*, 47(2):617–644. 1

Obraczka, K., Manduchi, R., and Garcia-Luna-Aveces, J. (2002). Managing the information flow in visual sensor networks. In *Wireless Personal Multimedia Communications, 2002. The 5th International Symposium on*, volume 3, pages 1177–1181. IEEE. 4

Oliver, N. M., Rosario, B., and Pentland, A. P. (2000). A bayesian computer vision system for modeling human interactions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):831–843. 78

Osher, S., Burger, M., Goldfarb, D., Xu, J., and Yin, W. (2005). An iterative regularization method for total variation-based image restoration. *Multiscale Modeling & Simulation*, 4(2):460–489. 58

Pati, Y., Rezaiifar, R., and Krishnaprasad, P. (1993). Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on*, pages 40–44. IEEE. 10

Peyré, G., Bougleux, S., and Cohen, L. (2008). Non-local regularization of inverse problems. In *Computer Vision–ECCV 2008*, pages 57–68. Springer. 57, 58

Priyantha, B., Lymberopoulos, D., and Liu, J. (2011a). Littlerock: Enabling energy-efficient continuous sensing on mobile phones. *Pervasive Computing, IEEE*, 10(2):12–15. 3

Priyantha, B., Lymberopoulos, D., and Liu, J. (2011b). LittleRock: Enabling energy-efficient continuous sensing on mobile phones. *IEEE Pervasive Computing*, 10:12–15. 21

Qiu, C. and Vaswani, N. (2011). ReProCS: A missing link between recursive robust PCA and recursive sparse recovery in large but correlated noise. *arXiv preprint arXiv:1106.3286*. 78, 80, 85

Rana, R. K., Chou, C. T., Kanhere, S. S., Bulusu, N., and Hu, W. (2010). Ear-phone: an end-to-end participatory urban noise mapping system. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 105–116. ACM. 2

Ravindranath, L., Newport, C., Balakrishnan, H., and Madden, S. (2011). Improving wireless network performance using sensor hints. In *Proceedings of the 8th USENIX conference on Networked systems design and implementation*, pages 21–21. USENIX Association. 38

Ravishankar, S. and Bresler, Y. (2011). MR image reconstruction from highly undersampled k-space data by dictionary learning. *Medical Imaging, IEEE Transactions on*, 30(5):1028–1041. 72

Reddy, S., Mun, M., Burke, J., Estrin, D., Hansen, M., and Srivastava, M. (2010). Using mobile phones to determine transportation modes. *ACM Transactions on Sensor Networks (TOSN)*, 6(2):13. 38

Rinner, B. and Wolf, W. (2008). An introduction to distributed smart cameras. *Proceedings of the IEEE*, 96(10):1565–1575. 76

Roy, A., Rumble, S. M., Stutsman, R., Levis, P., Mazières, D., and Zeldovich, N. (2011). Energy management in mobile devices with the cinder operating system. In *Proceedings of the sixth conference on Computer systems*, pages 139–152. ACM. 3

Ryder, M. (2012). What is activity theory? http://carbon.ucdenver.edu/~mryder/itc_data/act_dff.html. [Webpage; Accessed on 08/23/2012]. 24

Sankaranarayanan, A. C., Veeraraghavan, A., and Chellappa, R. (2008). Object detection, tracking and recognition for multiple smart cameras. *Proceedings of the IEEE*, 96(10):1606–1624. 4

Shannon, C. E. (1949). Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21. 1

Shu, X., Yang, J., and Ahuja, N. (2014). Non-local compressive sampling recovery. In *Computational Photography (ICCP), 2014 IEEE International Conference on*, pages 1–8. xiii, xiv, 57, 58, 59, 66, 67, 68

Soro, S. and Heinzelman, W. (2009). A survey of visual sensor networks. *Advances in Multimedia*, 2009. 4

Stauffer, C. and Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE. 76, 77

Thiagarajan, A., Ravindranath, L., LaCurts, K., Madden, S., Balakrishnan, H., Toledo, S., and Eriksson, J. (2009). Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, pages 85–98. ACM. 2, 16

Tian, D. and Georganas, N. D. (2003). A node scheduling scheme for energy conservation in large wireless sensor networks. *Wireless Communications and Mobile Computing*, 3(2):271–290. 5

Tian, Y., Rao, J., Wang, W., Chen, C., and Ma, J. (2011). The organization of mobile personal lifelog by activity. *Advances in Multimedia Information Processing-PCM 2010*, pages 31–42. 24

Tian, Y.-L., Lu, M., and Hampapur, A. (2005). Robust and efficient foreground analysis for real-time video surveillance. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 1182–1187. IEEE. 76

Times, N. (2012). So you're a good driver? let's go to the monitor. http://www.nytimes.com/2012/11/25/business/seeking-cheaper-insurance-drivers-accept-monitoring-devices.html. [Online; Accessed on 08/27/2013]. 36

Tropp, J. and Gilbert, A. (2007). Signal recovery from random measurements via orthogonal matching pursuit. *Information Theory, IEEE Transactions on*, 53(12):4655–4666. 10

Wakin, M., Laska, J., Duarte, M., Baron, D., Sarvotham, S., Takhar, D., Kelly, K., and Baraniuk, R. G. (2006). Compressive imaging for video representation and coding. In *Picture Coding Symposium*, volume 1. 13

Wang, Z., Taylor, C., Cao, Q., Qi, H., and Wang, Z. (2011). Friendbook: privacy preserving friend matching based on shared interests. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, pages 397–398. ACM. 2, 16

Whittaker, E. T. (1915). *On the functions which are represented by the expansions of the interpolation-theory*. Edinburgh University. 1

Wikipedia (2014). Moore's law. http://en.wikipedia.org/wiki/Moore's_law. [Webpage; Accessed on 04/09/2014]. 1

Wright, J., Ganesh, A., Rao, S., Peng, Y., and Ma, Y. (2009). Robust principal component analysis: Exact recovery of corrupted low-rank matrices by convex optimization. In *Proc. of Neural Information Processing Systems*, volume 3. 78

Wu, C., Aghajan, H., and Kleihorst, R. (2008). Real-time human posture reconstruction in wireless smart camera networks. In *Proceedings of the 7th international conference on Information processing in sensor networks*, pages 321–331. IEEE Computer Society. 76

Wu, X. and Liu, M. (2012). In-situ soil moisture sensing: measurement scheduling and estimation using compressive sensing. In *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, pages 1–12. ACM. 30

Yan, M., Yang, Y., and Osher, S. (2012). Robust 1-bit compressive sensing using adaptive outlier pursuit. *Signal Processing, IEEE Transactions on*, 60(7):3868–3875. 89

Yang, B. and Hairong, Q. (2010). Feature-based image comparison for semantic neighbor selection in resource-constrained visual sensor networks. *EURASIP Journal on Image and Video Processing*, 2010. 4

Yang, S. and Gerla, M. (2010). Energy-efficient accelerometer data transfer for human body movement studies. In *Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC), 2010 IEEE International Conference on*, pages 304–311. IEEE. 12, 21, 23

Yang, Z. and Jacob, M. (2013). Nonlocal regularization of inverse problems: a unified variational framework. *Image Processing, IEEE Transactions on*, 22(8):3192–3203. 57, 58

Yick, J., Mukherjee, B., and Ghosal, D. (2008). Wireless sensor network survey. *Computer networks*, 52(12):2292–2330. 4

Yilmaz, A., Javed, O., and Shah, M. (2006). Object tracking: A survey. *ACM computing surveys*, 38(4):13. 76

Zhang, J., Liu, S., Xiong, R., Ma, S., and Zhao, D. (2013). Improved total variation based image compressive sensing recovery by nonlocal regularization. In *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on*, pages 2836–2839. IEEE. xiii, xiv, 65, 67, 68, 84

Zhang, J., Liu, S., and Zhao, D. (2012a). Improved total variation based image compressive sensing recovery by nonlocal regularization. *arXiv preprint arXiv:1208.3716*. 57

Zhang, J., Zhao, D., Zhao, C., Xiong, R., Ma, S., and Gao, W. (2012b). Compressed sensing recovery via collaborative sparsity. In *Data Compression Conference (DCC), 2012*, pages 287–296. IEEE. 54, 55

Zhang, X., Burger, M., Bresson, X., and Osher, S. (2010). Bregmanized nonlocal regularization for deconvolution and sparse reconstruction. *SIAM Journal on Imaging Sciences*, 3(3):253–276. xiii, xiv, 58, 65, 67, 68

Zhou, T. and Tao, D. (2011). Godec: Randomized low-rank & sparse matrix decomposition in noisy case. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 33–40. 76, 77, 81

Zhuang, Z., Kim, K., and Singh, J. (2010). Improving energy efficiency of location sensing on smartphones. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 315–330. ACM. 2, 16, 21

Zymnis, A., Boyd, S., and Candes, E. (2010). Compressed sensing with quantized measurements. *Signal Processing Letters, IEEE*, 17(2):149–152. 88

Appendix

Publications

1. **Shuangjiang, Li** and Wei, Wang and Hairong, Qi and Bulent, Ayhan and Chiman, Kwan and Steven, Vance, "Low-rank tensor decomposition based anomaly detection for hyperspectral imagery," *IEEE International Conference on Image Processing (ICIP)*, In Press, Quebec City, Canada, September 27-30, 2015.

2. **Shuangjiang, Li** and Hairong, Qi, "A Douglas-Rachford splitting approach to Compressed Sensing image recovery using low-rank regularization", *IEEE Transactions on Image Processing*, undergoing Marjor Revision.

3. **Shuangjiang, Li** and Hairong, Qi, "Compressed Phone Sensing", *IEEE Transactions on Mobile Computing*, under review.

4. **Shuangjiang, Li** and Hairong, Qi, "Compressed dictionary learning for detecting activations in fMRI using double sparsity", *2nd IEEE Global conference on Signal and Information Processing*, Dec. 3-5 2014, pages 434 - 437.

5. **Shuangjiang, Li** and Hairong, Qi, "Recursive Low-rank and Sparse Recovery of Surveillance Video using Compressed Sensing", *IEEE International Conference on Distributed Smart Cameras (ICDSC)*, Venice, Italy, Nov. 4-7, 2014.

6. **Shuangjiang, Li** and Rui, Guo and Li, He and Wei, Gao and Hairong, Qi and Gina, Owens, "Demo: MoodMagician: A Pervasive and unobtrusive emotion sensing system using mobile phones for improving human mental health", *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Nov. 3-5, 2014, Memphis, TN, pages 310-311.

7. **Shuangjiang, Li** and Hairong, Qi, "Pattern-based compressed phone sensing", *1st IEEE Global conference on Signal and Information Processing*, 2013, pages 169–172.

8. **Shuangjiang, Li** and Hairong, Qi, "Distributed data aggregation for sparse recovery in wireless sensor networks", *9th IEEE international conference on Distributed Computing in Sensor Systems (DCoSS)*, 2013, pages 62–69.

9. **Shuangjiang, Li** and Hairong, Qi, "Sparse representation based band selection for hyperspectral images", *18th IEEE International Conference on Image Processing (ICIP)*, 2011, pages 2693–2696.

10. Wei, Wang and **Shuangjiang, Li** and Hairong, Qi and Bulent, Ayhan and Chiman, Kwan and Steven, Vance, "Identify anomaly component by sparsity and low rank", *IEEE Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensor (WHISPERS)*, In Press, Tokyo, Japan, June 2-5, 2015.

11. Sisi, Xiong and Yanjun, Yao and **Shuangjiang, Li** and Qing, Cao and Tian, He and Hairong, Qi and Leon, Tolbert and Yilu, Liu, "kBF: Towards approximate and bloom filter based Key-Value storage for cloud computing systems", *IEEE Transactions on Cloud Computing*, In Press.

12. Wei, Wang and **Shuangjiang, Li** and Hairong, Qi, and Bulent, Ayhan and Chiman, Kwan and Steven, Vance, "Revisiting the preprocessing procedures for elemental concentration estimation based on chemcam LIBS on mars rover", *6th IEEE GRSS Workshop on Hyperspectral Image and Signal Processing*, Switzerland, June 25-27, 2014.

13. Rui, Guo and **Shuangjiang, Li** and Li, He and Wei, Gao and Hairong, Qi and Gina, Owens, "Pervasive and unobtrusive emotion sensing for human mental health", *7th International Conference on Pervasive Computing Technologies for Healthcare*, 2013, pages 436–439.

14. Jenson, Yin and Bulent, Ayhan and Chiman, Kwan and Wei, Wang and **Shuangjiang, Li** and Hairong, Qi and Steven, Vance, "Enhancement of JMARS", *44th Lunar and Planetary Science Conference*, 2013.

Vita

Shuangjiang Li (李双江) was born in Taoxi town, Shucheng county, Lu'an city, Anhui province, P.R. China in 1988. He is the son of Liming Li (李立明) and Ju Cheng (程菊) and grows up with his elder sister Jinguo Li (李巾帼). After graduating in 2005 from Shucheng High School, he attended University of Science and Technology Beijing, where he received a Bachelor of Science degree in 2009 from the department of Electrical and Information Engineering. In Fall 2009, Shuangjiang enrolled into the doctoral program at the University of Tennessee at Knoxville in the department of Electrical Engineering and Computer Science. At the same time, he joined the Advanced Imaging and Collaborative Information Processing (AICIP) group as a graduate teaching assistant and research assistant where he completed his Master of Science degree in the winter of 2011. His major research areas are compressed sensing, mobile phone sensing, wireless sensor networks, pattern recognition, and image processing.