ABSTRACT

Title of dissertation: ADAPTIVE SENSING
AND PROCESSING FOR SOME
COMPUTER VISION PROBLEMS

Garrett Warnell, Doctor of Philosophy, 2014

Dissertation directed by: Professor Rama Chellappa
Department of Electrical
and Computer Engineering

This dissertation is concerned with adaptive sensing and processing in computer vision, specifically through the application of computer vision techniques to non-standard sensors.

In the first part, we adapt techniques designed to solve the classical computer vision problem of gradient-based surface reconstruction to the problem of phase unwrapping that presents itself in applications such as interferometric synthetic aperture radar. Specifically, we propose a new formulation of and solution to the classical two-dimensional phase unwrapping problem. As is usually done, we use the wrapped principal phase gradient field as a measurement of the absolute phase gradient field. Since this model rarely holds in practice, we explicitly enforce integrability of the gradient measurements through a sparse error-correction model. Using a novel energy-minimization functional, we formulate the phase unwrapping task as a *generalized lasso* problem. We then jointly estimate the absolute phase and the sparse measurement errors using the alternating direction method of multipliers

(ADMM) algorithm. Using an interferometric synthetic aperture radar noise model, we evaluate our technique for several synthetic surfaces and compare the results to recently-proposed phase unwrapping techniques. Our method applies new ideas from convex optimization and sparse regularization to this well-studied problem.

In the second part, we consider the problem of controlling and processing measurements from a non-traditional, compressive sensing (CS) camera in real time. We focus on how to control the number of measurements it acquires such that this number remains proportional to the amount of foreground information currently present in the scene under observations. To this end, we provide two novel adaptive-rate CS strategies for sparse, time-varying signals using side information. The first method utilizes extra *cross-validation* measurements, and the second exploits extra *low-resolution* measurements. Unlike the majority of current CS techniques, we do not assume that we know an upper bound on the number of significant coefficients pertaining to the images that comprise the video sequence. Instead, we use the side information to predict this quantity for each upcoming image. Our techniques specify a fixed number of spatially-multiplexed CS measurements to acquire, and they adjust this quantity from image to image. Our strategies are developed in the specific context of background subtraction for surveillance video, and we experimentally validate the proposed methods on real video sequences.

Finally, we consider a problem motivated by the application of active pan-tilt-zoom (PTZ) camera control in response to *visual saliency*. We extend the classical notion of this concept to multi-image data collected using a stationary PTZ camera by requiring *consistency*: the property that each saliency map in the set of those

that are generated should assign the *same* saliency value to distinct regions of the environment that appear in more than one image. We show that processing each image independently will often fail to provide a consistent measure of saliency, and that using an image mosaic to quantify saliency suffers from several drawbacks. We then propose *ray saliency*: a mosaic-free method for calculating a consistent measure of bottom-up saliency. Experimental results demonstrating the effectiveness of the proposed approach are presented.

ADAPTIVE SENSING AND PROCESSING FOR SOME COMPUTER VISION PROBLEMS

by

Garrett Warnell

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2014

Advisory Committee:
Professor Rama Chellappa, Chair/Advisor
Professor Larry Davis
Professor Behtash Babadi
Professor Amitabh Varshney
Professor John Benedetto

Dedication

To my parents.

Acknowledgments

The work presented in this dissertation would not have been possible without the support of several individuals to whom I am truly indebted.

First, I would like to thank my advisor, Professor Rama Chellappa, for always providing me with calm, steady guidance and a sense of humor. I am in awe of his generosity, and I am truly honored to be among those who call him their mentor.

Second, I would like to thank Dr. Vishal Patel. He has been a second mentor, a collaborator, and also a friend. I cannot imagine having completed this dissertation without his involvement.

Next, I'd like to acknowledge my committee members, Professors Larry Davis, Behtash Babadi, Amitabh Varshney, and John Benedetto. I am grateful for their willingness to take interest in and read, evaluate, and provide feedback on the work presented here.

I would also like to thank several collaborators for their significant involvement in the work presented below, including Dr. Dikpal Reddy, Dr. Philip David, and Professor Sourabh Bhattacharya.

During my studies, I was fortunate enough to be involved in two brilliant research groups: one at the University of Maryland, and another at the U.S. Army Research Laboratory. Both provided me an inspiring environment in which to work, and I am grateful to everyone involved.

Of course, I would be remiss if I did not especially acknowledge the exceptional sounding board I had in Lab 4438: Chris, Tho, Dave, Priyanka, Swami, Heng, and

Pouya. I could not have asked for better office mates and friends.

Several other friends deserve my thanks for helping and encouraging me throughout my studies. For continually reminding me of the outside world, thanks to all the members of the graduate student ultimate frisbee club. And for distractions of all other kinds, I'm grateful to Alex, Ross, Matt, Lauren, Sarah, Alex, Kevin, Mike, Dani, Kara, and especially Kate. I could not have completed my studies without their kindness and support.

Finally, I would like to thank my family for everything they have given me. Most importantly, they have provided me with stability and perspective. The pages of this dissertation would be blank without them.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

Over the course of the last several decades, there has been an explosion in both the availability and quality of traditional imaging systems. Billions of people have the capability to capture high-resolution images and videos using cheap consumer cameras and mobile phones. Organizations and individuals alike have access to high-resolution, multi-camera systems. The common component in all these devices is the traditional camera: a planar pixel array on which two dimensional projections of visible light are measured. These cameras are usually static in nature, controllable only through manual manipulation. Accordingly, much of contemporary computer vision research has focused on data acquired with these types of sensors: datasets that drive research in areas such as object recognition, visual saliency, and tracking are usually comprised of imagery acquired with static, traditional imaging devices [7] [3] [8]. While this type of data may be the most natural to consider for these tasks, there are several scenarios under which using conventional imaging devices may be undesirable or even impossible.

In this dissertation, we will consider three scenarios that employ unconventional imaging devices. While the acquired data is not of the usual variety, it is still visual in nature, and we will still seek to use it to accomplish tasks associated with

classical computer vision such as depth estimation, background subtraction, and visual saliency estimation. Therefore, we will focus on adapting existing techniques for such tasks to process the various types of visual data and, in some cases, to help control the non-traditional sensing process.

The first type of data we consider is that of complex-valued radar measurements. We are specifically concerned with the problem of *interferometric synthetic aperture radar* (InSAR), in which we seek to recover underlying depth images encoded in the absolute phase of the acquired radar observations. This problem is very similar to the classical computer vision problem of gradient-based surface reconstruction, and our work focuses on adapting techniques from that field in order to solve the phase unwrapping problem.

The second data type considered comes from an imaging device designed to help alleviate the modern problem of *data deluge* [9]: it is now so easy to collect massive amounts of data that information systems are being overwhelmed by the amount of data that they they are typically assigned to process. The research community has recently made progress in addressing this problem with the introduction of *compressive sensing* (CS). By leveraging the fact that most information in visual data is often of much lower dimension than that of the ambient signal space in which traditional sensors operate, CS researchers have proposed a solution in the form of new sensors. It is such a sensor we consider here, namely the *single-pixel camera* (SPC) [10]. Compared to measurements acquired with a traditional camera, far fewer SPC measurements are necessary in order to infer the image of the scene under observation. Previous work [11] has adapted traditional background subtraction

techniques to work with SPC measurements. Here, we extend that work in order to help optimally adjust the number of measurements the SPC collects.

Finally, we will also consider a precisely-controllable pan-tilt-zoom (PTZ) camera. Leveraging their flexible field-of-view, PTZ cameras provide a means by which an operator or autonomous system can limit the amount of image data collected while retaining the freedom make observations over a large physical area. The data acquired differs from that of a traditional static imaging system in that the obtained images are related by a known geometric transformation. Here, we will focus on exploiting this relationship in order to adapt classical *visual saliency* techniques such that a similar quantity can be computed using the acquired multi-image datasets.

1.1 Problems

The work presented in this dissertation provides adaptive sensing and processing techniques in the context of the following problems:

1. **Phase Unwrapping**: Interferometric synthetic aperture radar is a problem that requires visual information to be inferred from non-traditional measurements. The task is to estimate a three-dimensional surface from complex-valued radar measurements. The desired depth information can be easily computed once the absolute phase of these measurements is known, but the nature of the imaging system dictates that only the principal phase (absolute phase modulo 2π) is observable. The inference of absolute phase from the principal phase, called *phase unwrapping*, is an ill-posed problem. However,

there is a way in which principal phase measurements can be transformed into measurements of the absolute phase gradient field. Starting from here, we propose a technique that modifies traditional gradient-based surface reconstruction techniques in computer vision such that they can be used for the phase unwrapping problem.

2. **Adaptive-Rate Compressive Sensing**: CS addresses the data deluge problem by using new, non-traditional cameras. Here, we propose an adaptive sensing algorithm to be used in a scenario in which a compressive imaging device is used to sense a sparse, time-varying signal. We are specifically interested in background subtraction for visual surveillance. Classical CS theory assumes prior knowledge of signal sparsity in order to determine the number of sensor measurements needed to ensure adequate signal reconstruction. However, when dealing with dynamic signals such as video, prior information regarding the exact sparsity may be difficult to obtain. Hence, classical CS methods are forced to use wasteful upper bounds that result in the acquisition of an unnecessarily high quantity of data. Assuming the system uses a sensor that is able to adaptively adjust the number of compressive measurements it collects, we propose algorithms based on various forms of side information that quantitatively evaluate the current CS measurement rate and adjust it as needed.

3. **Multi-Image Visual Saliency**: This problem is motivated by the desire to use classical visual saliency to help guide optimal scene observation using

a PTZ camera. Here, the data of interest is a collection of multiple images related by a known geometric transformation. We propose a technique by which the concept of visual saliency can be adapted to these multi-image datasets. Traditional computational saliency methods involve finding certain portions of visual data that exhibit some notion of importance when compared to others. While the problem of identifying such regions has been well-studied for single images, these approaches are not designed for scenarios in which visual data arrives via multiple, geometrically-related observations collected using a manipulable sensor. Our proposed approach accounts for the known geometric relationship between images, and provides an adapted notion of saliency for visual data. Using a ray-based representation of the scene, we are able to efficiently quantify this new notion of saliency.

1.2 Contributions

With respect to each problem described above, this dissertation makes the following contributions:

1. **Phase Unwrapping**

   - We incorporate recent ideas in sparse error correction for performing ℓ_2-based phase unwrapping.
   - We provide a generalized-lasso formulation to the phase unwrapping problem.

- We use an efficient primal-dual algorithm, ADMM, to perform the unwrapping.

2. **Adaptive-Rate Compressive Sensing**

   - We provide an explicit technique for generating rate-adaptive CS measurement matrices.
   - We adapt the compressive background subtraction technique proposed by Cevher *et al.* [11] to a variable-measurement-rate scenario.
   - We provide real-time techniques by which side information of various types can be used in order to minimize the measurement rate of the imaging system.

3. **Multi-Image Visual Saliency**

   - We explicitly develop the notion of visual saliency for multi-image, geometrically-related data.
   - We explore several possible methods for quantifying visual saliency for this data.
   - We describe *ray saliency*, a novel and efficient means of quantifying visual saliency for multi-image data.

1.3 Organization

This dissertation is organized as follows. In Chapter 2, we provide general overviews of the areas of compressive sensing and visual saliency. Chapter 3 de-

scribes our work on the problem of phase unwrapping. Chapter 4 details the adaptive-rate CS techniques we have developed. Chapter 5 lays out our research efforts toward defining and computing multi-image visual saliency. We conclude with a summary and description of future research directions in Chapter 6.

Chapter 2: Background

In this chapter, we shall provide background discussion of two broad areas in which parts of our work reside. In Chapter 4, we will discuss an adaptive sensing algorithm that is designed for a compressive sensing camera. Here, we shall introduce the concept of CS, specifically in the context of the new imaging platforms it has inspired, including the single-pixel camera. In Chapter 5, we detail our work that extends classical visual saliency to multi-image data acquired using a PTZ camera. Here, we shall introduce visual saliency in computer vision and discuss several ways by which it can be computed.

2.1 Compressive Sensing

One of the primary tools with which the problem of data deluge can be addressed is *compressive sensing*: a relatively new theory in sensing which asserts that a certain class of discrete signals can be adequately sensed by capturing far fewer measurements than the dimension of the ambient space in which they reside. By "adequately sensed," it is meant that the signal of interest can be accurately inferred using the measurements acquired by the sensor.

In this dissertation, we use CS in the context of imaging. Consider a grayscale

image $F \in \mathbb{R}^{N \times N}$, vectorized in column-major order as $\mathbf{f} \in \mathbb{R}^{N^2}$. A traditional camera uses an $N \times N$ array of photodetectors in order to produce N^2 measurements of F: each detector records a single value that defines the corresponding component of $\mathbf{f}$. If we are instead able to gather measurements of a fundamentally different type, CS theory suggests that we may be able to determine $\mathbf{f}$ from far fewer than N^2 of them. Specifically, these *compressive measurements* record linear combinations of pixel values, i.e., $\boldsymbol{\xi} = \boldsymbol{\Phi}\mathbf{f}$, where $\boldsymbol{\Phi} \in \mathbb{C}^{M \times N^2}$ is referred to as a *measurement matrix* and $M << N^2$.

CS theory presents three general conditions under which the above claim is valid. First, $\mathbf{f}$ should be *sparse* or *compressible*. In general, a vector is said to be sparse if very few of its components are nonzero. More precisely, vectors having no more than s nonzero components are said to be *s-sparse*. A vector is said to be compressible if it is well-approximated by a sparse signal, i.e., it has a small number of components with a large magnitude and many with much smaller magnitudes.

Second, the measurement matrix (encoder) should exhibit the *restricted isometry property (RIP)* of a certain order and constant. Specifically, $\boldsymbol{\Phi}$ exhibits the RIP of order s with constant δ_s if the following inequality holds for all s-sparse $\mathbf{f}$:

$$(1 - \delta_s) \leq \frac{\|\boldsymbol{\Phi}\mathbf{f}\|_2^2}{\|\mathbf{f}\|_2^2} \leq (1 + \delta_s) \quad . \tag{2.1}$$

While we will discuss proposed construction methods for a $\boldsymbol{\Phi}$ that exhibits the RIP for specified s and δ_s in Section 4.4, they generally involve selecting M such that it exceeds a lower bound that grows with increasing s and decreasing δ_s.

Finally, an appropriate decoding procedure, $\hat{\mathbf{f}} = \Delta(\boldsymbol{\xi}, \boldsymbol{\Phi})$, should be used.

While many successful decoding schemes have been discussed in the literature, we shall focus here on one in particular:

$$\Delta(\boldsymbol{\xi}, \boldsymbol{\Phi}) = \underset{\mathbf{z} \in \mathbb{R}^{N^2}}{\arg\min} \|\mathbf{z}\|_1 \text{ subject to } \boldsymbol{\Phi}\mathbf{z} = \boldsymbol{\xi} \quad , \tag{2.2}$$

where the ℓ_1 norm is given explicitly by $\|\mathbf{z}\|_1 = \sum_i |z(i)|$.

With these three conditions in mind, CS theory provides us with the following result: for an s-sparse $\mathbf{f}$ measured with a $\boldsymbol{\Phi}$ that exhibits the RIP of order $2s$ with $\delta_{2s} \leq \sqrt{2} - 1$, $\Delta(\boldsymbol{\xi}, \boldsymbol{\Phi})$ will exactly recover $\mathbf{f}$ [12]. If $\mathbf{f}$ is compressible, a similar result that bounds the reconstruction error is available. Thus, by modifying the sensor and decoder to implement $\boldsymbol{\Phi}$ and Δ, respectively, $\mathbf{f}$ can be adequately sensed using only $M << N^2$ measurements.

Sensors based on the above theory are still just beginning to emerge [13]. One of the most notable is the single-pixel camera [14], where measurements specified by each row of $\boldsymbol{\Phi}$ are sequentially computed in the optical domain via a digital micromirror device and a single photodiode. Throughout the remainder of this dissertation, we shall assume that such a device is the primary sensor.

2.2 Visual Saliency

Broadly, *visual saliency* is a measure of how important visual data is in context. Of course, in order to actually quantify this quality, one must provide more precise definitions of both what is meant by importance and what comprises context. The way in which a computational visual saliency technique defines these concepts places it into one of two major categories: *bottom-up* or *top-down*.

Bottom-up saliency methods define the context as visual data belonging to a spatially-localized neighborhood about the region of interest. Classically, the input visual data consists of a single image, and regions of interest are pixels or groups of pixels. Some researchers define context using spatially-limited neighborhoods over image pixels [4] [15], while others consider the context to be all features present in the image [16] [5] [17]. More recent approaches consider multiple definitions of context as defined by scale [2] [18]. Regardless of the choice of context, bottom-up methods equate the notion of saliency with that of *anomaly*. That is, greater saliency is assigned to regions associated with data that is more anomalous with respect to the given context. Methods from this category produce results that agree with the "popout" phenomenon that is typically experienced in biological vision [19], i.e., phenomena that are typically identified as salient in the bottom-up sense also tend to draw the attention of human observers. Figure 2.1 depicts an example of a bottom-up saliency map generated by the technique proposed in [2]. Here, a saliency map is a spatial representation that reflects the importance assigned to image regions: brightness values corresponding to each image region quantify its importance with respect to others.

In contrast, top-down methods [20–22] use training data or other prior information as the context, and the saliency of a datum is quantified according to how *similar* it is to this context. For example, the well-studied task of object detection can be interpreted as a top-down saliency method that uses a very specific prior, namely a visual description of the object(s) of interest. While top-down methods are of great interest to the vision community, the notion of saliency considered in

(a) (b)

Figure 2.1: Sample image (a) from the MSRA Salient Object Database [3] and a bottom-up saliency map (b) generated by [2].

this dissertation belongs to the *bottom-up* family. We will therefore limit further discussion to this category.

Classically, bottom-up visual saliency algorithms are evaluated with respect to how well they agree with what humans think [23]. A given database of images is usually presented to human subjects, and they are asked to view these images. In some cases, researchers use an eye-tracking system and record the subjects' eye fixation data for each image. In other cases, the subjects are asked to manually indicate which regions of the image they feel are salient. Either way, this human-generated information is taken to be the ground truth for bottom-up saliency algorithms. Performance can be measured using classical classification metrics. For example, if the human-generated ground-truth information is in the form of labeled salient regions, an algorithm-generated saliency map can be used to generate a precision-recall curve. This can be done by varying a threshold value that defines a binary saliency decision at each pixel. One can then determine the number of true positives, false positives, etc. for this threshold. Points on the precision-recall curve are generated by repeating the process for all possible threshold values. This technique

will be used as one way to evaluate the method we present in Chapter 5.

Chapter 3: Phase Unwrapping

In this chapter, we focus on the problem of adapting classical gradient-based surface reconstruction algorithms to the unique measurement and inference problem of phase unwrapping. Techniques for solving this problem must operate on unique *wrapped* (modulo-2π) measurement data, which presents a unique set of challenges. Here, we present our own phase unwrapping technique, *phase unwrapping using the generalized lasso* (PUGL), which is based on contemporary ideas from sparsity-based regularization and convex optimization.

3.1 Introduction

Phase unwrapping is a problem that arises in many applications, including magnetic resonance imaging [24] [25], optical interferometry [26] [27], and interferometric synthetic aperture radar (InSAR) [28] [29]. The problem is to infer the real-valued *absolute phase* from measurements of the *principal phase*. It is usually the case that the absolute phase carries the information of interest, but the principal phase is the only observable quantity. The two quantities are related as follows: if $\phi \in \mathbb{R}$ represents the absolute phase, then the corresponding principal phase value

is given by $\psi = \mathcal{W}(\phi)$, where the wrapping operator $\mathcal{W}$ is defined as

$$\mathcal{W} : \mathbb{R} \to [-\pi, \pi)$$

$$\mathcal{W}(\phi) = [(\phi + \pi) \bmod 2\pi] - \pi, \tag{3.1}$$

and applied componentwise in the case of vector-valued $\boldsymbol{\phi}$.

In this chapter, we are primarily interested in the InSAR application. Specifically, the InSAR problem is to recover a three-dimensional surface (i.e., depth) from multiple radar measurements of that surface. Considering two distinct vantage points (e.g., airplane positions), the desired depth information can be easily computed from two physical quantities: *(1)* the (known) distance between vantage points and *(2)* the difference between the two point-to-ground path lengths. Since the distance to the surface is encoded in the absolute phase of a radar measurement, the latter quantity can theoretically be computed from the difference between the absolute phase measurements obtained at each vantage point. Unfortunately, practical InSAR measurement systems can only observe the principal phase, from which only the wrapped phase difference can be computed. Thus, in order to recover the depth information, we must solve the unwrapping problem, i.e., inferring $\boldsymbol{\phi}$ from $\boldsymbol{\psi}$.

Due to the many-to-one nature of $\mathcal{W}$, this problem is ill-posed. Therefore, in order to find a unique solution, additional constraints must be imposed on $\boldsymbol{\phi}$. One such constraint that is applied almost universally in the literature is derived from the *Itoh condition* [30]. Assuming a two-dimensional $\boldsymbol{\phi}$ that is comprised of phase samples obtained on a uniformly-spaced discrete grid, the Itoh condition is said to be satisfied if neighboring phase values do not differ by more than π. Whether or

not this condition is satisfied depends on both the spatial sampling rate and the smoothness of the underlying physical quantity. However, if the Itoh condition is satisfied, then it can be shown that

$$\boldsymbol{\nabla}\boldsymbol{\phi} = \mathcal{W}\left(\boldsymbol{\nabla}\boldsymbol{\psi}\right), \tag{3.2}$$

where $\boldsymbol{\nabla}$ computes differences between four-connected neighbors. Unfortunately, even in the case of sufficiently smooth $\boldsymbol{\phi}$ and sufficient spatial sampling, (3.2) may fail to hold due to noise in the system that acquires $\boldsymbol{\psi}$. Nevertheless, this gradient constraint is used in most unwrapping procedures, and the distinguishing trait among these procedures is the way in which this issue is addressed.

In this chapter, we propose a novel phase unwrapping technique that explicitly models the error in (3.2) as a *sparse* quantity, i.e., that significant inequality occurs in a relatively small number of locations. We formulate the unwrapping problem as one of jointly estimating both the absolute phase and the sparse errors, and cast it as a *generalized lasso* [31] [32] problem. We then propose the use of the *alternating direction method of multipliers* (ADMM) algorithm [31] to compute the estimates.

3.1.1 Organization

This chapter is organized as follows. In Section 3.2, we review related work in the field of phase unwrapping. In Section 3.3 we develop our formulation of sparse-error-corrected phase unwrapping, which culminates in an interpretation of the problem in the generalized lasso framework. In Section 3.4, we detail the ADMM algorithm we use to efficiently perform the absolute phase estimation. Finally, we

present the results of our technique in Section 3.5.

3.2 Related Work

Phase unwrapping is a problem that has received a great deal of attention from the research community. While early efforts focused on estimating the absolute phase directly from wrapped observations, more recent work has also dealt with the more limited task of denoising the wrapped observations (see, e.g., [1,33]). While these denoising methods do not explicitly perform the unwrapping, they often produce very good results when used to preprocess the input before applying techniques that actually estimate the absolute phase.

The method we present here is one that computes an estimate of the absolute phase directly from wrapped, possibly noisy, observations. Virtually all such techniques rely on (3.2), which allows us to use $\boldsymbol{\psi}$ to generate measurements $\mathcal{W}(\boldsymbol{psi})$ of the horizontal and vertical absolute phase differences. The unwrapping problem then becomes one of estimating a two-dimensional image, $\boldsymbol{\phi}$ from measurements of its gradient field. This more general problem is one with a wide variety of applications beyond that of phase unwrapping. For a thorough treatment, see [34].

Early attempts at solving the phase unwrapping problem sought to generate a solution through the use of *path-following* [28, 29, 35] techniques. Starting from a point with known phase, these techniques generate a solution by sequentially summing the phase difference measurements over a path that covers the spatial domain. Ideally, any such path will generate the same solution. However, when the

measured differences have errors (due to, e.g., noise), the solution becomes path-dependent due to *residues*: points around which a closed integration path does not yield a value of zero. Path-following techniques attempt to first identify residue locations and then select integration paths that avoid them in order to mitigate their effects.

Instead of point-by-point unwrapping, a more popular class of phase unwrapping techniques formulates the absolute phase estimation problem as one of energy minimization. Mathematically, energy minimization techniques take the following form:

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathcal{J}(\boldsymbol{\phi}) \tag{3.3}$$

where the distinguishing trait among these methods lies in how $\mathcal{J}$ is defined. The technique proposed by Hunt [36] selects

$$\mathcal{J}(\boldsymbol{\phi}) = \|\boldsymbol{\nabla}\boldsymbol{\phi} - \mathcal{W}(\boldsymbol{\nabla}\boldsymbol{\psi})\|_2^2, \tag{3.4}$$

where $\boldsymbol{\nabla}\boldsymbol{\phi} = \left[\boldsymbol{\nabla}_x\boldsymbol{\phi}^T \ \boldsymbol{\nabla}_y\boldsymbol{\phi}^T\right]^T$, and $\boldsymbol{\nabla}_x\boldsymbol{\phi}$ and $\boldsymbol{\nabla}_y\boldsymbol{\phi}$ denote the vectorized horizontal and vertical components, respectively, of the forward-difference approximation of the spatial gradient. Substituting (3.4) in (3.3), the optimal $\boldsymbol{\phi}$ can be computed by solving the standard Poisson equation.

Several improvements to the above have been made by modifying $\mathcal{J}$. For example, several researches have proposed the addition of regularizing terms. Marroquin *et al.* [37] add regularizing terms $\|\boldsymbol{\nabla}\boldsymbol{\phi}\|_2^2$ and $\|\boldsymbol{P}_2\boldsymbol{\phi}\|_2^2$, where components of $\boldsymbol{P}_2\boldsymbol{\phi}$ represent second-order differences of $\boldsymbol{\phi}$. Guerriero *et al.* [38] also include a regularization term based on second-order differences of $\boldsymbol{\phi}$ and replace the term in

(3.4) with one that explicitly enforces the integrability of $\mathcal{W}(\boldsymbol{\nabla\psi})$. A variety of other regularizing terms have also been proposed, and we refer the reader to the work of Nico *et al.* [39] for a comprehensive review.

Another way in which $\mathcal{J}$ has been modified from its original formulation in (3.4) is with respect to the penalty function. Ghiglia and Romero [40] propose using the more general ℓ_p norm:

$$\begin{aligned}\mathcal{J}(\boldsymbol{\phi}) &= \|\boldsymbol{\nabla\phi} - \mathcal{W}(\boldsymbol{\nabla\psi})\|_p^p \\ &= \sum_i |\nabla\phi_i - \mathcal{W}(\nabla\psi_i)|^p, \end{aligned} \tag{3.5}$$

where the subscript i denotes the i^{th} vector component. When $p = 2$, (3.5) reduces to (3.4), and can be efficiently solved using techniques such as the one proposed by Ghiglia and Romero [41]. However, other values of p can affect the convexity of (3.3), and require more creative computational techniques, such as approaches from network programming [42] [43] in order to compute a solution.

The method we propose here is an energy-minimization approach to phase unwrapping. We select a $\mathcal{J}$ that enforces the gradient constraint as in (3.4) but is able to robustly handle outliers in $\mathcal{W}(\boldsymbol{\nabla\psi})$ by simultaneously enforcing sparsity in an error term that ensures integrability. This formulation allows us to use an iterative primal-dual algorithm in order to efficiently compute a solution.

3.3 Problem Formulation

In more concrete terms, we consider the problem of estimating a two-dimensional absolute phase image from a wrapped observation. Let $\boldsymbol{\phi} \in \mathbb{R}^{mn}$ represent the un-

known, vectorized, $m \times n$ absolute phase image. Similarly, let $\boldsymbol{\psi} \in [-\pi, \pi)^{mn}$ represent the corresponding wrapped observation. We shall assume that $\boldsymbol{\psi} = \mathcal{W}(\boldsymbol{\phi})$, where $\mathcal{W}$ is defined as in (3.1).

It will be useful to later discussion if we first define two sparse matrices, $\mathbf{G}$ and $\mathbf{C}$. The first of these is used to compute the forward-difference approximation of the spatial gradient. Let $\mathbf{G}_x \in \{-1, 0, 1\}^{mn \times mn}$ compute the vectorized forward-difference approximation to the horizontal component of the spatial gradient for an input image vectorized in column-major order. Each row of $\mathbf{G}_x$ corresponds to a pixel location (x, y) in the input image. For pixel locations where $x = n$, the forward-difference approximation to the horizontal gradient cannot be computed, and so we insert an all-zero row for this location in order to impose a gradient value of zero. For illustration, the first two rows of $\mathbf{G}_x$ are given by

$$\begin{bmatrix} -1 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 1 & 0 & \cdots & 0 \end{bmatrix}.$$

Let $\mathbf{G}_y \in \{-1, 0, 1\}^{mn \times mn}$ compute the vectorized forward-difference approximation to the vertical component of the spatial gradient in a similar fashion. $\mathbf{G}_y$ will contain all-zero rows for pixel locations where $y = m$. Note that $\mathbf{G}_x$ and $\mathbf{G}_y$ are sparse matrices: each row is of length mn with at most two nonzero entries. From these two matrices, we form $\mathbf{G} = \left[\mathbf{G}_x^T \ \mathbf{G}_y^T\right]^T$, a $2mn \times mn$ sparse matrix that we can use to computed the stacked, vectorized spatial gradient components, $\left[\boldsymbol{\nabla}_x \boldsymbol{\phi}^T \ \boldsymbol{\nabla}_y \boldsymbol{\phi}^T\right]^T$.

The second matrix we shall define is one that enables the computation of the curl of a gradient field. This can be done by considering two-by-two loop integrals over the underlying spatial domain. Let $\mathbf{p}, \mathbf{q} \in \mathbb{R}^{mn}$ define the horizontal and

vertical components, respectively, of an $m \times n$ gradient field. Then the curl for a single two-by-two loop at spatial location (x, y) is defined as:

$$\begin{aligned} \texttt{curl}(y, x) =& p(y+1, x) - p(y, x) \\ &+ q(y, x) - q(y, x+1), \end{aligned} \tag{3.6}$$

where x and y denote the vertical and horizontal pixel coordinates, respectively.

If we stack the gradient components to form a single vector, then we can compute all mn curl values using the matrix-vector equation $\mathbf{C}\left[\mathbf{p}^T \mathbf{q}^T\right]^T$, where we define $\mathbf{C} \in \{-1, 0, 1\}^{mn \times 2mn}$ as follows. Each row of $\mathbf{C}$ corresponds to a pixel location in the input image, with values of ± 1 place in locations such that (3.6) is computed. For pixel locations at which we cannot compute (3.6), we insert an all-zero row in order to define a curl value of zero. For illustration, the first row of $\mathbf{C}$ is given by

$$\left[\begin{array}{cccccc|cccccc} -1 & 1 & 0 & \cdots & 0 & & 1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 \end{array}\right],$$

where the vertical divider shown above separates coefficients corresponding to the horizontal and vertical gradient components. Note that, like $\mathbf{G}$, $\mathbf{C}$ is also sparse: each row is of length $2mn$ with at most four nonzero entries.

With these matrices defined, we now focus on our formulation of the phase unwrapping problem. We start by considering the gradient constraint (3.2). As we discussed in Section 3.1, this equation can be violated in several locations due to noise and other factors. We explicitly model this error using the modified equation

$$\mathbf{G}\boldsymbol{\phi} = \mathcal{W}\left(\mathbf{G}\boldsymbol{\psi}\right) - \mathbf{e}, \tag{3.7}$$

where $\mathbf{e} \in \mathbb{R}^{2mn}$ represents the error and we have made the gradient computation explicit by replacing $\boldsymbol{\nabla}$ with $\mathbf{G}$. Even in the case of known $\mathbf{e}$, (3.7) does not specify a unique solution. This is due to the fact that the constraints are only in the gradient domain, which allows for a single degree of freedom that corresponds to the unknown constant of integration. Therefore, it is possible that an infinite number of $\boldsymbol{\phi}$ will satisfy (3.7) exactly. To resolve this ambiguity, we impose the constraint that the k^{th} pixel, $k \in \{1, \ldots, mn\}$, of $\boldsymbol{\phi}$ has value zero, thereby explicitly specifying the constant of integration. Therefore, we use the following energy function for unwrapping:

$$\mathcal{J}_u(\boldsymbol{\phi}) = \|\boldsymbol{\nabla}\boldsymbol{\phi} - (\mathcal{W}(\boldsymbol{\nabla}\boldsymbol{\psi}) - \mathbf{e})\|_2^2 + |\mathbf{a}_k^T\boldsymbol{\phi}|^2, \tag{3.8}$$

where $\mathbf{a}_k$ is an mn-dimensional column vector with a value of one in the k^{th} component and values of zero elsewhere.

While (3.8) enables the computation of an optimal $\boldsymbol{\phi}$, we have not yet addressed how to find $\mathbf{e}$. To this end, we examine the integrability of the measured gradient field. For noiseless gradient measurements, such as $\mathbf{G}\boldsymbol{\phi}$, (3.6) yields a value of zero for each loop. That is, the gradient field is *integrable* (also known as *irrotational* in the phase unwrapping literature [29]), or $\mathbf{CG}\boldsymbol{\phi} = \mathbf{0}$. Therefore, if we left-multiply both sides of (3.7) by $\mathbf{C}$,

$$\mathbf{CG}\boldsymbol{\phi} = \mathbf{C}\mathcal{W}(\mathbf{G}\boldsymbol{\psi}) - \mathbf{Ce}$$

$$\mathbf{0} = \mathbf{C}\mathcal{W}(\mathbf{G}\boldsymbol{\psi}) - \mathbf{Ce},$$

we arrive at the following set of constraints for $\mathbf{e}$:

$$\mathbf{Ce} = \mathbf{C}\mathcal{W}(\mathbf{G}\boldsymbol{\psi}). \tag{3.9}$$

Especially in noise-free conditions, it is often the case that (3.2) is violated over a relatively small set of components, i.e., $\mathbf{e}$ is sparse. In a similar fashion to Reddy et al. [44], we use the ℓ_1-norm as a proxy for sparsity and use the sparsity-promoting term $\|\mathbf{e}\|_1$ as a regularizer when seeking the optimal $\mathbf{e}$. We can also cast this problem as one of energy minimization with the functional

$$\mathcal{J}_e = \lambda_c \|\mathbf{C}\mathbf{e} - \mathbf{C}\mathcal{W}(\mathbf{G}\boldsymbol{\psi})\|_2^2 + \lambda_s \|\mathbf{e}\|_1, \tag{3.10}$$

where the values chosen for λ_c and λ_s specify the relative importance of satisfying each criterion.

Combining (3.8) and (3.10) above, we propose to jointly estimate $\boldsymbol{\phi}$ and $\mathbf{e}$ using the following program:

$$\begin{aligned}(\boldsymbol{\phi}^*, \mathbf{e}) &= \arg\min_{\boldsymbol{\phi},\mathbf{e}} \mathcal{J}_u(\boldsymbol{\phi}) + \mathcal{J}_e(\mathbf{e}) \\ &= \arg\min_{\boldsymbol{\phi},\mathbf{e}} \|\mathbf{G}\boldsymbol{\phi} - (\mathcal{W}(\mathbf{G}\boldsymbol{\psi}) - \mathbf{e})\|_2^2 \\ &\quad + \lambda_c \|\mathbf{C}\mathbf{e} - \mathbf{C}\mathcal{W}(\mathbf{G}\boldsymbol{\psi})\|_2^2 \\ &\quad + \lambda_s \|\mathbf{e}\|_1 + |\mathbf{a}_k^T \boldsymbol{\phi}|^2. \end{aligned} \tag{3.11}$$

3.3.1 Generalized Lasso Formulation

Optimization problem (3.11) can be rewritten as

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{F}\mathbf{x}\|_1, \tag{3.12}$$

where $\lambda = \lambda_s$ controls the trade-off between satisfying the ℓ_2 and ℓ_1 constraints, and

$$\begin{aligned}
\mathbf{x} &= \begin{bmatrix} \boldsymbol{\phi} \\ \mathbf{e} \end{bmatrix}, \\
\mathbf{A} &= \begin{bmatrix} \mathbf{G} & \mathbf{I} \\ \mathbf{a}_k^T & \mathbf{0}^T \\ \mathbf{0} & \lambda_c \mathbf{C} \end{bmatrix}, \\
\mathbf{F} &= \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix}, \\
\mathbf{b} &= \begin{bmatrix} \mathcal{W}(\mathbf{G}\boldsymbol{\psi}) \\ 0 \\ \lambda_c \mathbf{C} \mathcal{W}(\mathbf{G}\boldsymbol{\psi}) \end{bmatrix}.
\end{aligned} \tag{3.13}$$

The optimization problem (3.12) can be viewed as the *generalized lasso* problem [31] [32] which can be efficiently solved via the alternating direction method of multipliers (ADMM) algorithm [31].

3.4 Optimization

In a more general form, (3.12) can be seen as an instance of the following optimization problem

$$\min f(\mathbf{x}) + g(\mathbf{z}) \;\text{ such that }\; \mathbf{D}\mathbf{x} + \mathbf{H}\mathbf{z} = \mathbf{c}, \tag{3.14}$$

where f and g are convex functions. The augmented Lagrangian for (3.14) is

$$\mathcal{F}_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{u}^T(\mathbf{D}\mathbf{x} + \mathbf{H}\mathbf{z} - \mathbf{c}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{x} + \mathbf{H}\mathbf{z} - \mathbf{c}\|_2^2, \tag{3.15}$$

where $\rho > 0$ and $\mathbf{u}$ is the Lagrange multiplier corresponding to the linear constraint. The ADMM method consists of the following iterations

$$\mathbf{x}_{k+1} = \arg\min_{\mathbf{x}} \mathcal{F}_\rho(\mathbf{x}, \mathbf{z}_k, \mathbf{u}_k) \tag{3.16}$$

$$\mathbf{z}_{k+1} = \arg\min_{\mathbf{z}} \mathcal{F}_\rho(\mathbf{x}_k, \mathbf{z}, \mathbf{u}_k) \tag{3.17}$$

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \rho(\mathbf{D}\mathbf{x}_{k+1} + \mathbf{H}\mathbf{z}_{k+1} - \mathbf{c}). \tag{3.18}$$

We now apply the ADMM method to solve the optimization problem in (3.12). In ADMM form, (3.12) can be written as

$$\min \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{z}\|_1 \text{ such that } \mathbf{F}\mathbf{x} - \mathbf{z} = \mathbf{0}, \tag{3.19}$$

where $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$, $g(\mathbf{z}) = \lambda\|\mathbf{z}\|_1$, $\mathbf{D} = \mathbf{F}$, $\mathbf{H} = -\mathbf{I}$ and $\mathbf{c} = \mathbf{0}$. The augmented Lagrangian for (3.19) is

$$\mathcal{F}_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{z}\|_1 + \mathbf{u}^T(\mathbf{F}\mathbf{x} - \mathbf{z}) + \frac{\rho}{2}\|\mathbf{F}\mathbf{x} - \mathbf{z}\|_2^2. \tag{3.20}$$

In the ADMM method, variables are optimized one at a time while keeping the other variables fixed. In what follows, we describe each of the sub-optimization problems in detail.

3.4.1 Update Step for $\mathbf{x}$

With fixed $\mathbf{z}$ and $\mathbf{u}$, $\mathbf{x}_{k+1}$ is obtained by minimizing $\mathcal{F}_\rho$ with respect to $\mathbf{x}$. Taking derivative of (3.20) with respect to $\mathbf{x}$ and setting it to zero, we obtain the following update for $\mathbf{x}$

$$\mathbf{x}_{k+1} = (\mathbf{A}^T\mathbf{A} + \rho\mathbf{F}^T\mathbf{F})^{-1}(\mathbf{A}^T\mathbf{b} + \rho\mathbf{F}^T(\mathbf{z}_k - \mathbf{u}_k)). \tag{3.21}$$

In other words, $\mathbf{x}_{k+1}$ is obtained by solving an $N \times N$ system of linear equations. In our case, with $\mathbf{A}$ defined as in (3.13), $N = 3mn + 1$. For large N, conjugate gradient methods can be employed to solve for $\mathbf{x}_{k+1}$.

3.4.2 Update Step for $\mathbf{z}$

To find $\mathbf{z}_{k+1}$, we fix $\mathbf{x}_k$, $\mathbf{u}_k$ and minimize $\mathcal{F}_\rho$ with respect to $\mathbf{z}$. This results in a soft-shrinkage problem whose solution is also of closed form

$$\mathbf{z}_{k+1} = \mathcal{S}_{\frac{\lambda}{\rho}}\left(\mathbf{F}\mathbf{x}_{k+1} + \frac{\mathbf{u}_k}{\rho}\right), \tag{3.22}$$

where the soft-shrinkage operator is defined as

$$\mathcal{S}_\alpha(\mathbf{x}) = \left(1 - \frac{\alpha}{\|\mathbf{x}\|_2}\right)_+ \mathbf{x} \tag{3.23}$$

and $(\cdot)_+$ returns its argument if it is non-negative and zero otherwise.

3.4.3 Update Step for $\mathbf{u}$

Finally, having $\mathbf{x}_{k+1}$ and $\mathbf{z}_{k+1}$ fixed, a gradient ascent update with the step size ρ is performed on the Lagrange multiplier as

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \rho(\mathbf{F}\mathbf{x}_{k+1} - \mathbf{z}_{k+1}). \tag{3.24}$$

These three steps are repeated until convergence is achieved or the number of iterations exceeds some maximum amount.

3.4.4 Stopping Criteria

After each round of updates (3.21), (3.22), (3.24), we check to see if the current objective value is sufficiently optimal. To do so, we define the primal and dual residuals as

$$\mathbf{r}_{k+1} = \mathbf{F}\mathbf{x}_{k+1} - \mathbf{z}_{k+1}, \tag{3.25}$$

$$\mathbf{s}_{k+1} = -\rho\mathbf{F}^T\left(\mathbf{z}_{k+1} - \mathbf{z}_k\right), \tag{3.26}$$

respectively. Boyd *et al.* [31] suggest that the following are reasonable stopping criteria:

$$\|\mathbf{r}_{k+1}\|_2 \leq \epsilon^{\text{pri}}, \tag{3.27}$$

$$\|\mathbf{s}_{k+1}\|_2 \leq \epsilon^{\text{dual}}, \tag{3.28}$$

where

$$\epsilon^{\text{pri}} = \sqrt{P}\epsilon^{\text{abs}} + \epsilon^{\text{rel}} \max\left\{\|\mathbf{F}\mathbf{x}_{k+1}\|_2, \|\mathbf{z}_k\|_2\right\}, \tag{3.29}$$

$$\epsilon^{\text{dual}} = \sqrt{N}\epsilon^{\text{abs}} + \epsilon^{\text{rel}}\|\mathbf{F}^T\mathbf{z}_{k+1}\|_2, \tag{3.30}$$

and $\mathbf{F}$ is a $P \times N$ matrix (in our problem, $P = 2mn$). Above, ϵ^{abs} is chosen with respect to the scale of the problem and ϵ^{rel} is chosen small, e.g., 10^{-3}. Practically, the algorithm can also be terminated if a maximum number of iterations has been executed without residuals that satisfy (3.27) or (3.28).

The ADMM algorithm for solving (3.12) is summarized in Algorithm 1. When we apply it to phase unwrapping using the formulation specified by (3.13), we refer to the procedure as *phase unwrapping using the generalized lasso*, or PUGL.

3.5 Experiments

In order to validate the proposed unwrapping scheme, we performed several experiments using both real and synthetic data. In this section, we shall describe this data, outline our experimental setup, and discuss the results.

3.5.1 Noisy Observation Model

The data we use is noise-free. In order to evaluate algorithm performance in the presence of noise, we adopt a synthetic observation observation model that is commonly used in the InSAR phase unwrapping literature [45]. Let x_1 and x_2 denote the noisy, complex-valued radar measurements of the same two-dimensional location as observed from two separate viewpoints, i.e.,

$$\begin{aligned} x_1 &= z_1 e^{j\phi_1} \\ x_2 &= z_2 e^{j\phi_2}. \end{aligned} \tag{3.31}$$

Algorithm 1: ADMM algorithm for solving (3.12)

Input: λ,$\mathbf{b}$,$\mathbf{A} \in \mathbb{R}^{M\times N}$,$\mathbf{F} \in \mathbb{R}^{P\times N}$,$\rho$,$\epsilon^{\text{abs}}$,$\epsilon^{\text{rel}}$,maxIter

Initialization:

- Set Terminate $\leftarrow$ False.

- Set $\mathbf{z}_0 = \mathbf{0}, \mathbf{x}_0 = \mathbf{0}, \mathbf{u}_0 = \mathbf{0}$.

while (Terminate == False) **do**

- Calculate $\mathbf{x}_{k+1}$ by solving the following system of equations

$$(\mathbf{A}^T\mathbf{A} + \rho\mathbf{F}^T\mathbf{F})\mathbf{x}_{k+1} = (\mathbf{A}^T\mathbf{b} + \rho\mathbf{F}^T(\mathbf{z}_k - \mathbf{u}_k))$$

- Calculate $\mathbf{z}_{k+1}$ according to

$$\mathbf{z}_{k+1} = \mathcal{S}_{\frac{\lambda}{\rho}}\left(\mathbf{F}\mathbf{x}_{k+1} + \frac{\mathbf{u}_k}{\rho}\right)$$

- Calculate $\mathbf{u}_{k+1}$ according to

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \rho(\mathbf{F}\mathbf{x}_{k+1} - \mathbf{z}_{k+1})$$

- Calculate $\mathbf{r}_{k+1}$ and $\mathbf{s}_{k+1}$ according to

$$\mathbf{r}_{k+1} = \mathbf{F}\mathbf{x}_{k+1} - \mathbf{z}_{k+1}$$

$$\mathbf{s}_{k+1} = -\rho\mathbf{F}^T(\mathbf{z}_{k+1} - \mathbf{z}_k)$$

- Calculate ϵ^{pri} and ϵ^{dual} according to

$$\epsilon^{\text{pri}} = \sqrt{P}\epsilon^{\text{abs}} + \epsilon^{\text{rel}}\max\{\|\mathbf{F}\mathbf{x}_{k+1}\|_2, \|\mathbf{z}_k\|_2\}$$

$$\epsilon^{\text{dual}} = \sqrt{N}\epsilon^{\text{abs}} + \epsilon^{\text{rel}}\|\mathbf{F}^T\mathbf{z}_{k+1}\|_2$$

- $k \leftarrow k + 1$

- **if** $\left(\|\mathbf{r}_{k+1}\|_2 \leq \epsilon^{\text{pri}}$ and $\|\mathbf{s}_{k+1}\|_2 \leq \epsilon^{\text{dual}}\right)$ or $(k \geq \text{maxIter})$

then

Terminate $\leftarrow$ True

end if

end while

Output: $\hat{\mathbf{x}} = \mathbf{x}_k$.

Above, we assume that the complex amplitudes, z_i, are circularly symmetric and Gaussian. Ideally, z_1 and z_2 are identical, but due to factors such as scatterer displacement, we only assume that $\mathbb{E}\left[|z_1|^2\right] = \mathbb{E}\left[|z_2|^2\right] = \theta^2$ and $\mathbb{E}\left[z_1 z_2^*\right] = \alpha\theta^2$, where $\alpha \in [0, 1]$ is referred to as the *coherence*. Given a known absolute phase ϕ, we generate synthetic measurements as follows. We first select ϕ_1 and ϕ_2 such that $\phi = \phi_1 - \phi_2$. We then generate z_1 and z_2 such that the above assumptions are satisfied. Finally, we calculate the wrapped phase difference value by extracting the phase angle from the conjugate product of x_1 and x_2, i.e.,

$$\psi = \arg\left(x_1 x_2^*\right). \tag{3.32}$$

The level of noise in this observation model is determined by the value of α: $\alpha = 1$ indicates that there is no noise in ψ, while $\alpha = 0$ corresponds to a ψ that is comprised entirely of of noise. In order to evaluate unwrapping performance in the presence of noise, we performed experiments and generated results for every value of α in the set $\{0.7, 0.75, 0.8, 0.85, 0.9, 0.95, 0.99\}$.

3.5.2 Phase Data

We tested our algorithm using four surfaces. Three of these were constructed synthetically:

- *Gaussian Surface*: A 128×128 image, centered at $(0, 0)$, of a two-dimensional Gaussian with peak height 14π and $\sigma_x = 10$ and $\sigma_y = 15$.
- *Truncated Gaussian Surface*: the above-mentioned Gaussian Surface with the upper-left-hand quadrant set to zero.

- *Shear Surface*: A 100×100 image where one half of the plane contains a linear ramp with a maximum height of 79.

Additionally, we also used the *Longs Peak* surface distributed with [29]: a real elevation map corresponding to a geographic area located in Colorado, USA. These surfaces are displayed in the leftmost columns of Figures 3.1 and 3.2.

For each surface, we generated seven different noisy wrapped observations according to the procedure outlined in Section 3.5.1, one for each value of α specified above. We also generated the noise-free wrapped image. Further, we computed denoised versions of each noisy wrapped observation using the recently-proposed NL-InSAR [1] technique with ten iterations using the implementation made available by the authors. To each unwrapping technique under evaluation, we supplied both the noisy and denoised versions of the wrapped phase observations and recorded the estimated phase.

3.5.3 Evaluation

The primary metric of evaluation we use is the mean-squared error between the true surface, $\boldsymbol{\phi}$, and the estimate $\hat{\boldsymbol{\phi}}$. In order to account for the unresolved degree of freedom that results from using only gradient measurements, we first ensure that each estimated surface has zero mean, i.e., we calculate the mean-squared error according to

$$\mathrm{MSE}\left(\boldsymbol{\phi}, \hat{\boldsymbol{\phi}}\right) = \frac{1}{mn} \sum_i \left[\left(\boldsymbol{\phi}_i - \bar{\boldsymbol{\phi}}_i\right) - \left(\hat{\boldsymbol{\phi}}_i - \bar{\hat{\boldsymbol{\phi}}}_i\right) \right]^2, \tag{3.33}$$

where $\bar{\boldsymbol{\phi}}$ and $\bar{\hat{\boldsymbol{\phi}}}$ denote the across-pixel mean values for $\boldsymbol{\phi}$ and $\hat{\boldsymbol{\phi}}$, respectively.

Table 3.1: Surface Reconstruction MSE for Noisy Wrapped Phase Observations

Gaussian Surface	$\alpha = 0.70$	$\alpha = 0.75$	$\alpha = 0.80$	$\alpha = 0.85$	$\alpha = 0.90$	$\alpha = 0.95$	$\alpha = 0.99$	$\alpha = 1$
PhaseLa	72.18	24.62	29.69	16.84	2.00	**0.02**	**0.01**	1.30
PUMA	**5.09**	**2.35**	**0.86**	0.68	0.48	0.26	0.07	**0.00**
PUGL	6.58	3.26	1.79	**0.62**	**0.44**	0.25	0.07	0.00
Truncated Gaussian Surface	$\alpha = 0.70$	$\alpha = 0.75$	$\alpha = 0.80$	$\alpha = 0.85$	$\alpha = 0.90$	$\alpha = 0.95$	$\alpha = 0.99$	$\alpha = 1$
PhaseLa	64.27	40.90	51.81	63.89	46.34	40.07	41.08	42.40
PUMA	17.83	12.48	10.54	13.56	15.82	9.77	9.92	11.71
PUGL	**17.55**	**10.58**	**9.27**	**8.09**	**6.42**	**9.19**	**9.86**	**9.82**
Shear Surface	$\alpha = 0.70$	$\alpha = 0.75$	$\alpha = 0.80$	$\alpha = 0.85$	$\alpha = 0.90$	$\alpha = 0.95$	$\alpha = 0.99$	$\alpha = 1$
PhaseLa	524.62	431.29	461.34	507.24	546.87	555.08	560.36	529.53
PUMA	428.37	393.72	397.77	390.12	384.83	377.64	307.12	309.83
PUGL	**311.61**	**388.03**	**274.58**	**311.40**	**249.41**	**233.83**	**211.36**	**210.34**
Longs Peak	$\alpha = 0.70$	$\alpha = 0.75$	$\alpha = 0.80$	$\alpha = 0.85$	$\alpha = 0.90$	$\alpha = 0.95$	$\alpha = 0.99$	$\alpha = 1$
PhaseLa	476.00	482.93	303.04	408.01	268.67	406.25	318.19	334.50
PUMA	151.03	137.20	138.33	117.95	103.84	99.98	100.10	100.23
PUGL	**150.48**	**117.39**	**119.46**	**107.70**	**101.75**	**99.13**	**82.05**	**82.08**

We compared our technique with two recent phase-unwrapping algorithms: PhaseLa [46] and PUMA [43]. To generate results, we used the implementations made available by the authors. For the PhaseLa algorithm, we used the ICI-adaptive approach with $H = [1, 2, 3, 4]$ and $\Gamma = 2.0$ (see [46] for definitions). For the PUMA algorithm, we used the convex clique potential induced by selecting $p = 2$. We believe this to be a fair comparison over other values of p since the current formulation of our problem uses the ℓ_2 norm for the terms corresponding to $\boldsymbol{\phi}$.

Table 3.1 shows the mean-squared errors that result from each unwrapping procedure when using the noisy wrapped observations. Table 3.2 is similar, but

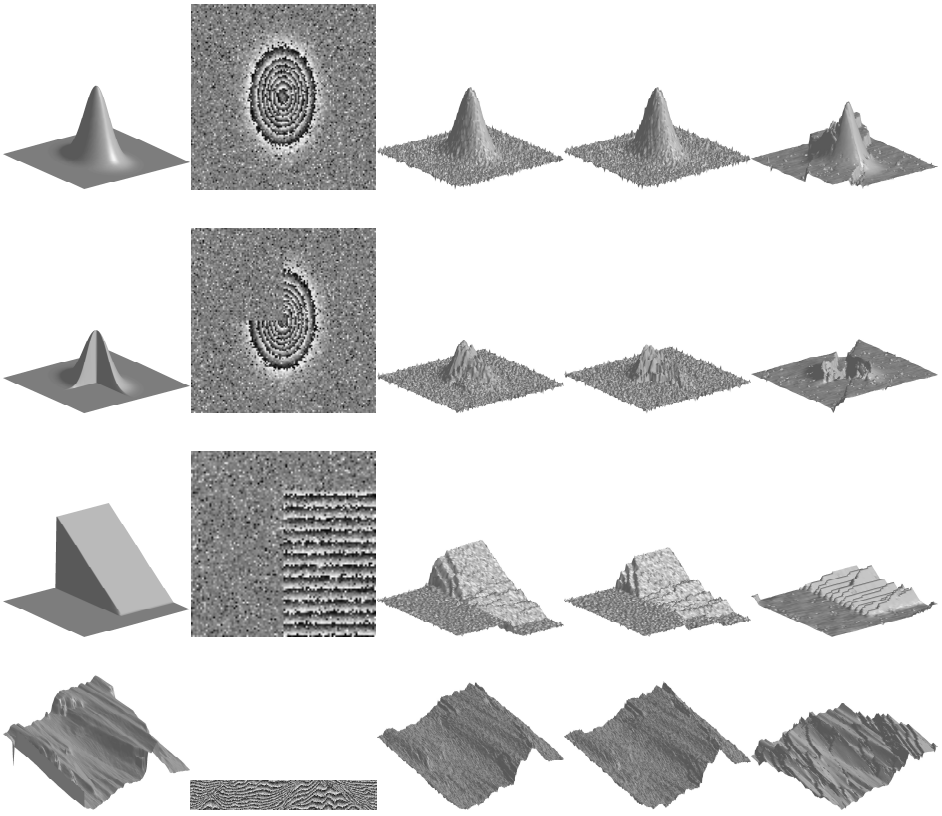


Figure 3.1: Absolute phase reconstructions. Left to right: actual phase, noisy wrapped phase generated using $\alpha = 0.85$, PUGL estimate, PUMA estimate, PhaseLa estimate. Top to bottom: Gaussian Surface, Truncated Gaussian Surface, Shear Surface, and Longs Peak.

Table 3.2: Surface Reconstruction MSE for NL-InSAR-Denoised [1] Wrapped Phase Observations

Gaussian Surface	$\alpha = 0.70$	$\alpha = 0.75$	$\alpha = 0.80$	$\alpha = 0.85$	$\alpha = 0.90$	$\alpha = 0.95$	$\alpha = 0.99$
PhaseLa	23.36	9.15	52.00	1.44	0.21	**0.02**	**0.01**
PUMA	9.91	4.48	4.17	0.25	0.02	0.02	0.02
PUGL	**4.43**	**4.44**	**3.04**	**0.24**	**0.02**	0.02	0.02
Truncated Gaussian Surface	$\alpha = 0.70$	$\alpha = 0.75$	$\alpha = 0.80$	$\alpha = 0.85$	$\alpha = 0.90$	$\alpha = 0.95$	$\alpha = 0.99$
PhaseLa	41.83	38.70	45.84	42.38	39.12	41.92	45.08
PUMA	18.88	14.11	11.48	10.44	9.76	11.43	9.10
PUGL	**9.65**	**7.86**	**6.49**	**8.72**	**8.29**	**9.12**	**8.47**
Shear Surface	$\alpha = 0.70$	$\alpha = 0.75$	$\alpha = 0.80$	$\alpha = 0.85$	$\alpha = 0.90$	$\alpha = 0.95$	$\alpha = 0.99$
PhaseLa	561.07	567.81	569.38	565.92	567.18	574.41	571.13
PUMA	379.35	367.58	378.82	369.77	308.80	308.94	307.43
PUGL	**214.44**	**212.97**	**221.90**	**217.28**	**212.78**	**213.36**	**211.73**
Longs Peak	$\alpha = 0.70$	$\alpha = 0.75$	$\alpha = 0.80$	$\alpha = 0.85$	$\alpha = 0.90$	$\alpha = 0.95$	$\alpha = 0.99$
PhaseLa	395.22	340.24	301.88	357.23	282.63	260.16	300.30
PUMA	206.24	151.70	112.23	105.65	100.45	103.51	108.70
PUGL	**175.99**	**110.49**	**77.07**	**78.41**	**70.82**	**100.58**	**91.65**

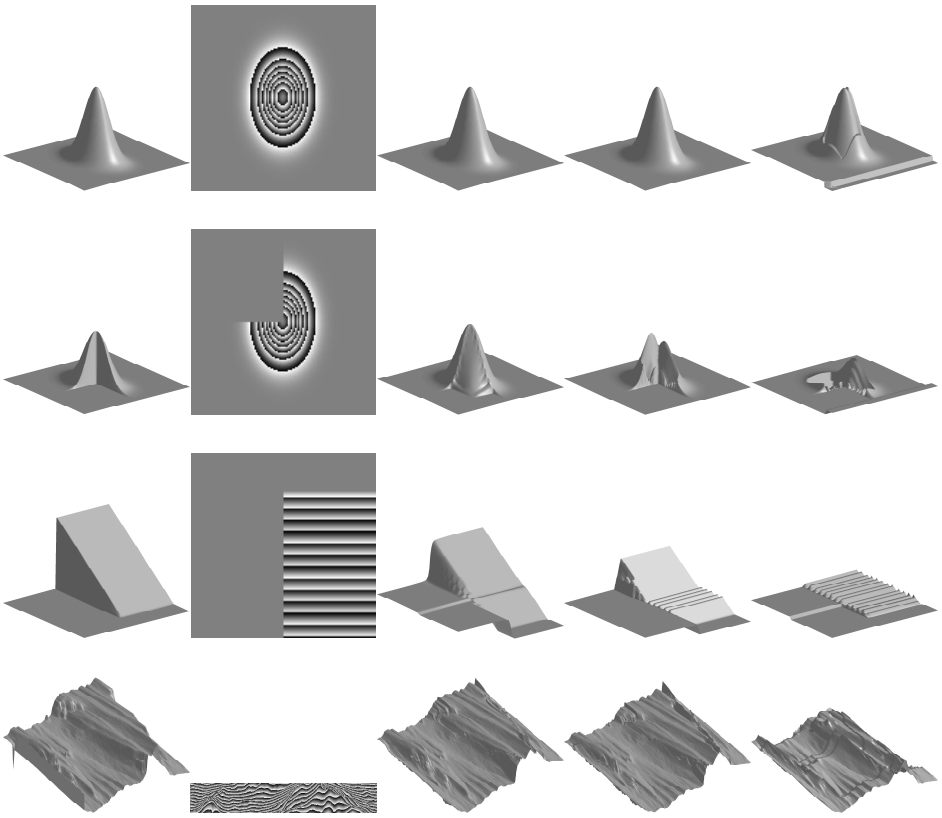


Figure 3.2: Absolute phase reconstructions. Left to right: actual phase, wrapped phase generated using $\alpha = 1$ (noise-free), PUGL estimate, PUMA estimate, PhaseLa estimate. Top to bottom: Gaussian Surface, Truncated Gaussian Surface, Shear Surface, and Longs Peak.

corresponds to the case when the noisy wrapped observations are first denoised by the NL-InSAR algorithm [1]. It can be seen that PUGL performs significantly better than the other algorithms presented, especially for very difficult surfaces like the Shear Surface and Longs Peak. To visualize the unwrapping results, we show absolute phase estimates for all four surfaces under two conditions. The first, depicted in Figure 3.1, shows the phase estimates when $\alpha = 0.85$. Figure 3.2 depicts the estimates for noise-free wrapped observations. In all cases, the PUGL reconstruction appears to be closer, or at least as close, to the true surface as the reconstructions provided by the other algorithms.

3.5.4 Parameters and Error Correction

To generate the PUGL estimates, we tried several parameter values and ultimately selected $\lambda_c = 200$ and $\lambda_s = 1$. We found that these values ensure that the integrability criterion is enforced strongly while still encouraging $\mathbf{e}$ to be reasonably sparse. Doing so is helpful because nonzero curl values corresponding to $\mathcal{W}(\boldsymbol{\nabla}\boldsymbol{\psi})$ are often indicators as to where the gradient measurements are incorrect due to wrapping artifacts. To demonstrate that this is the case, we examine the PUGL output when unwrapping the noise-free wrapped phase for the Truncated Gaussian Surface. Even though there is no noise, the surface itself violates the Itoh condition near the sharp discontinuity induced by masking the upper-left-hand quadrant: for pixels on the border of this region, there are phase differences of magnitude larger than π. Therefore, $\boldsymbol{\nabla}\boldsymbol{\phi} \neq \mathcal{W}(\boldsymbol{\nabla}\psi)$ for these pixels due to the wrapping operation.

Table 3.3: Unwrapping run time (sec.) for the Gaussian surface

	$\alpha = 0.70$	$\alpha = 0.75$	$\alpha = 0.80$	$\alpha = 0.85$	$\alpha = 0.90$	$\alpha = 0.95$
PhaseLa	95.28	88.15	80.16	74.22	67.25	61.12
PUMA	2.93	2.79	2.99	2.75	2.71	3.03
PUGL	48.64	48.34	48.82	48.65	48.96	48.72

It is exactly these types of errors that we wish to correct with $\mathbf{e}$. In Figure 3.3, we can see that the significant components of $\mathbf{e}$ do, in fact, cluster around this region and that the PUGL output is a slightly-smoothed version of the true surface.

3.5.5 Computational Complexity

Table 3.3 displays the running time of each unwrapping algorithm for the Gaussian Surface for various noise levels. It is important to note that the implementations of PhaseLa and PUGL are both in MATLAB, while the PUMA algorithm is implemented in C. Therefore, it is difficult to compare the running times of PhaseLa and PUGL with that of PUMA. However, it can be noted that PUGL is faster than PhaseLa, and that its running time remains constant regardless of the noise level. This is, in part, due to the fact that we set the maximum number of ADMM iterations to 50. However, as Figure 3.4 shows, the algorithm seems to have reasonably converged after about 20 iterations for each noise level.

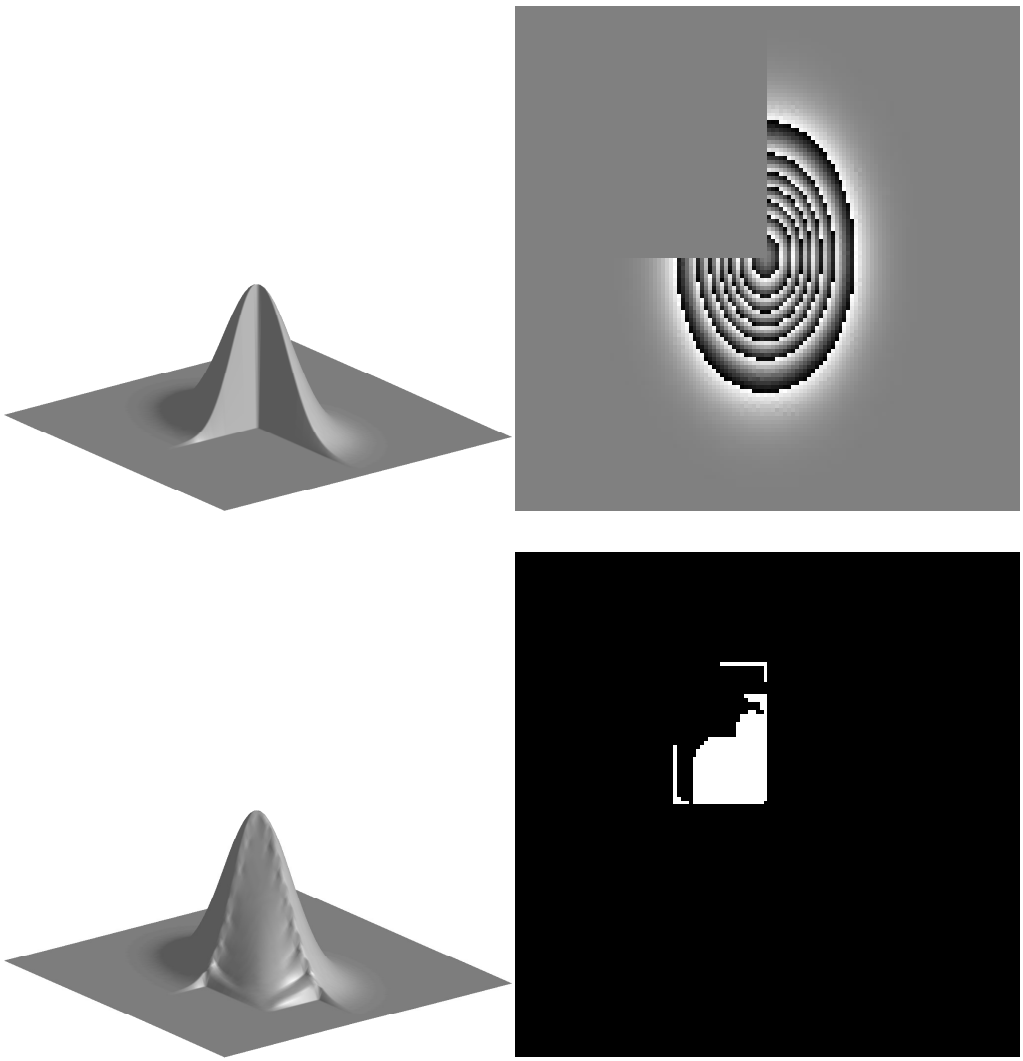


Figure 3.3: Upper-left: Truncated Gaussian Surface. Upper-right: noise-free wrapped phase. Lower-left: PUGL estimate. Lower-right: locations of the significant components of the optimal **e**.

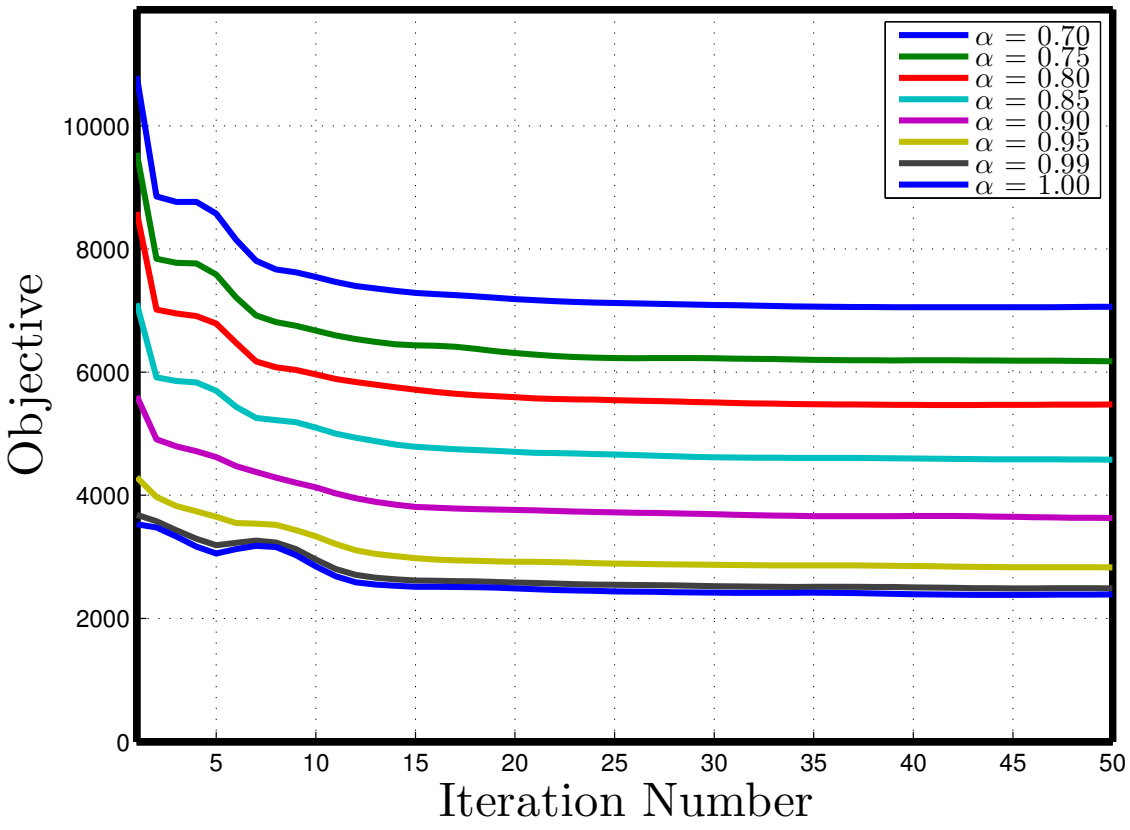


Figure 3.4: PUGL objective value per ADMM iteration for the Shear Surface.

3.6 Summary

In this chapter, we proposed a novel formulation of the phase unwrapping problem, and provided a practical scheme by which to make the corresponding absolute phase estimate. We posed the problem as one of sparse error correction by explicitly modeling the error in gradient field measurements obtained from the wrapped phase. We then estimated the error term as one that induced an integrable gradient field while remaining as sparse as possible. We combined the above with a classical ℓ_2-based unwrapping scheme in such a way that the joint absolute phase and error estimation could be cast in the generalized lasso framework. We used the ADMM algorithm to efficiently compute the optimal values. We termed the overall algorithm *phase unwrapping using the generalized lasso*, or PUGL, and examined its performance for a variety of surfaces and noise levels.

Chapter 4: Adaptive-Rate Compressive Sensing

Motivated by the modern problem of data deluge [9], researches in the field of compressive sensing have devised unconventional imaging devices that collect far fewer measurements than their traditional counterparts. Since most real-world signal classes exhibit some type of sparsity (e.g., image wavelet coefficients), CS provides a data-efficient manner by which they can be sensed. In fact, the sparser the signals are (i.e., the fewer significant coefficients they contain), the fewer measurements CS theory requires in order to guarantee that they are recoverable. In this chapter, we are concerned with using CS imaging devices to observe time-varying signals. Specifically, we focus on sequences of foreground images that can be obtained, e.g., via performing background subtraction on surveillance video. We shall discuss our work that enables online, adaptive adjustment of the compressive imaging device in response to the dynamic properties of the signal under observation.

4.1 Introduction

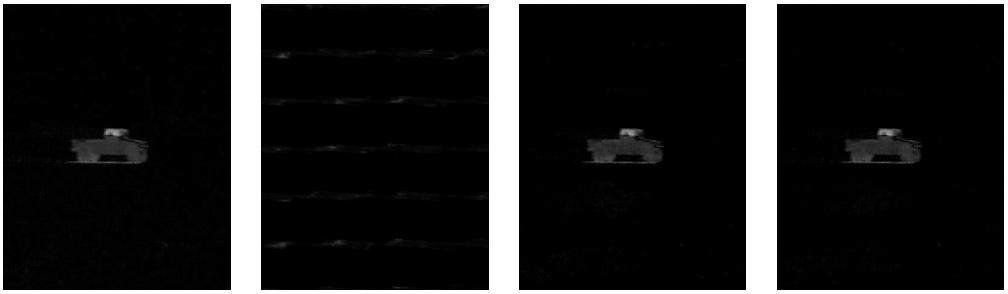
Visual surveillance is a task that often involves collecting a large amount of data in search of information contained in relatively small segments of video. For example, a surveillance system tasked with intruder detection will often spend most

of its time collecting observations of a scene in which no intruders are present. Without any such *foreground objects*, the corresponding surveillance video is useless: it is only the portions of video that depict these unexpected objects in the environment that are relevant to the surveillance task. However, because it is unknown when such objects will appear, many systems gather the same amount of data regardless of scene content. This static approach to sensing is wasteful in that resources are spent collecting unimportant data. However, it is not immediately clear how to efficiently acquire useful data since the periods of scene activity are unknown in advance. If this information were available *a priori*, a better scheme would be to collect data only during times when foreground objects are present.

In any attempt to do so, the system must make some sort of real-time decision regarding scene activity. However, such a decision can be made only if real-time data to that effect is available. We shall refer to such data as *side information*. Broadly, this information can come from two sources: a secondary modality and/or the primary video sensor itself. In this chapter, we develop two adaptive sensing schemes that exploit side information that comes from an example of each. Our first strategy employs a single video sensor to continuously make observations that are simultaneously used to infer both the foreground and the scene activity. The second adaptive method we present determines scene activity using observations that come from a secondary visual sensor. Both methods utilize a *compressive sensing* (CS) [47] [48] [49] [50] [51] camera as the primary modality. While many such sensors are beginning to emerge [13], our methods are specifically developed for a fast variant of a spatially multiplexing camera such as the single-pixel camera [14] [52].

We consider the following basic scenario: a CS camera is tasked with observing a region for the purpose of obtaining the foreground video. Since the foreground often occupies only a relatively small number of pixels, Cevher *et al.* [11] have shown that a small number of compressive measurements provided by this camera are sufficient to ensure that the foreground can be accurately inferred. However, the solution provided in that work implicitly relies on an assumption that is pervasive in the CS literature: that an upper bound on the *sparsity* (number of significant components) of the signal(s) under observation is known. Such an assumption enables the use of a static measurement process for each image in the video sequence. However, foreground video is a dynamic entity: changes in the number and appearance of foreground objects can cause large changes in sparsity with respect to time. Underestimating this quantity will lead to the use of a CS system that will provide too few measurements to obtain an accurate reconstruction. Overestimating signal sparsity, on the other hand, will lead to the collection of more measurements than necessary to achieve a good foreground estimate. For example, consider Figure 4.1. The true foreground's (Figure 4.1(a)) reconstruction is poor when too few compressive measurements are collected (Figure 4.1(b)), but looks virtually the same whether or not an optimal or greater-than-optimal number of measurements are acquired (Figures 4.1(c) and 4.1(d), respectively). Therefore, dependent on the number of measurements acquired at each time instant, the static CS approach is insufficient at worst and wasteful at best.

We provide adaptive-rate CS strategies that seek to address this problem. The approaches we present utilize two different forms of side information: *cross-*



(a) (b) (c) (d)

Figure 4.1: Foreground reconstruction with varying measurement rates. (a) is the true foreground, (b) is the foreground reconstruction when too few measurements are used, (c) is the reconstruction when an optimal number of measurements are used, and (d) is the reconstruction when more than the optimal number of measurements are used.

validation measurements and *low-resolution measurements*. In each case, we use the extra information in order to predict the number of foreground pixels (sparsity) in the next frame.

4.1.1 Related Work

Adapting the standard CS framework to a dynamic, time-varying signal is something that has been studied from various perspectives by several researchers.

Wakin *et al.* [53], Park and Wakin [54], Sankaranarayanan *et al.* [55], and Reddy *et al.* [56] have each proposed video-specific versions of CS that leverage video-specific signal dynamics such as temporal correlation and optical flow. For measurement models that provide streaming CS measurements, Sankaranarayan *et al.* [57], Asif and Romberg [58], and Angelosante *et al.* [59] have proposed adaptive

CS decoding procedures that are faster and more accurate than those that do not explicitly model the video dynamics.

Vaswani *et al.* [60] [61] [62], Cossalter *et al.* [63], and Stankovic *et al.* [64] [65] have proposed modifications to the CS decoding step that leverage extra signal support information in order to provide more accurate reconstructions from a fixed number of measurements. More generally, Scarlett *et al.* [66] provide generic information-theoretic bounds for any support-adaptive decoding procedure. Malioutov *et al.* [67] and Boufonous *et al.* [68] propose decoders with adaptive stopping criteria: sequential signal estimates are made until either a consistency or cross-validation criterion is met.

Several researchers have also considered adaptive encoding techniques. These techniques primarily focus on finding and using the "best" compressive measurement vectors at each instant of time. Ashok *et al.* [69] propose an offline procedure in order to design entire measurement matrices optimized for a specific task. Similarly, Duarte-Carvajalino *et al.* [70] compute class-specific optimal measurements offline, but decide which class to use using an online procedure with a fixed number of measurements. Purely-online procedures include those developed by Averbuch *et al.* [71], Ji *et al.* [72], Chou *et al.* [73], and Haupt *et al.* [74]: the next-best measurement vectors are computed by optimizing criterion functions that penalize quantites such as posterior entropy and expected reconstruction error. A few of these methods use a fixed measurement rate, while others propose stopping criterion similar to several of the adaptive decoding procedures.

Some of the above methods exhibit an adaptive measurement rate in that they

stop collecting measurements when certain criteria are met. However, due to the dynamic nature of video signals, it may not be possible to evaluate these criteria (as they often involve CS decoding) and collect a new measurement before the signal has significantly changed. Recent adaptive-rate work by Yuan *et al.* [75] and Schaeffer *et al.* [76] sidesteps this problem by using a static spatial measurement rate and considering how to adaptively select the *temporal* compression rate through batch analysis. In contrast, we propose here techniques that specify a fixed number of *spatially*-multiplexed measurements to acquire before sensing the signal at a given time instant and modify this quantity between each acquisition without assuming that the signal remains static between acquisitions. That is, we consider a system in which the decoding procedure is fixed and we are able to change the encoding procedure. This is fundamentally different from the previously-discussed work on adaptive decoding procedures (e.g., that of Vaswani *et al.* [60] [61] [62]).

4.1.2 Organization

This chapter is organized as follows. Sections 4.2 and 4.3 provide a precise formulation of and context for our rate-adaptive CS algorithms. Our measurement acquisition technique is described in Section 4.4. The proposed adaptive-rate CS techniques are discussed in Sections 4.5 and 4.6, and they are experimentally validated in Section 4.7. We summarize our work in Section 4.8.

4.2 Problem Statement

We assume that we possess a CS camera that is capable of acquiring a variable number of compressive measurements at discrete instants of time. We denote the sensor's measurement matrix at time t by $\mathbf{\Phi}_t \in \mathbb{R}^{M_t \times N^2}$, and we construct it via a process that depends only on our choice for M_t (see Section 4.4). Prior to time t, we will determine the value of M_t using an adaptive sensing strategy. We will assume that the images under observation are of size $N \times N$, where $X_t \in \mathbb{R}^{N \times N}$ will denote the specific image at time t. Vectorizing X_t using column-major order as $\mathbf{x}_t \in \mathbb{R}^{N^2}$ allows us to write the compressive measurement process at time t as $\mathbf{y}_t = \mathbf{\Phi}_t \mathbf{x}_t$.

We will present two adaptive sensing strategies that will each exploit a different type of side information. The first strategy uses a small set of *cross-validation measurements*, $\boldsymbol{\chi}_t \in \mathbb{R}^r$, obtained from a static linear measurement operator $\mathbf{\Psi} \in \mathbb{C}^{r \times N^2}$, i.e., $\boldsymbol{\chi}_t = \mathbf{\Psi} \mathbf{x}_t$. We will refer to $\mathbf{\Psi}$ as a *cross-validation matrix*. The second strategy we present relies on a set of *low-resolution measurements*, $Z_t \in \mathbf{R}^{L \times L}$, that we obtain via a secondary sensor that collects lower-resolution measurements of X_t. Such multi-camera systems are not uncommon in the surveillance literature (see, e.g., [77] [78]).

Having established the above notation, the problem we address here is one of how to use the observations, $\mathbf{y}_t$, and either of the sources of side information, $\boldsymbol{\chi}_t$ or Z_t, to select a minimal value for M_{t+1} that will ensure $\mathbf{\Phi}_{t+1}$ gathers enough information to ensure accurate reconstruction of the foreground (dynamic) component of the high-resolution X_t.

4.3 Compressive Sensing for Background Subtraction

We present our work in the context of the problem of background subtraction for video sequences. Broadly, background subtraction is the process of decomposing an image into foreground and background components, where the foreground usually represents the objects of interest in the environment under observation. For our purposes, we shall adopt the following model for images $\mathbf{x}_t$:

$$\mathbf{x}_t = \mathbf{f}_t + \mathbf{b} \quad , \tag{4.1}$$

where $\mathbf{b}$ is an unknown but deterministic static component of each image in the video sequence and $\mathbf{f}_t$ is a random variable. At time t, we estimate the locations of foreground pixels by computing the set of indices $\mathcal{F}_t = \{i : |f_t(i)| \geq \tau\}$, for some pre-defined threshold τ. We further assume that the components of $\mathbf{f}_t$ that correspond to $\mathcal{F}_t$ are bounded in magnitude, i.e., $|f_t(i)| \leq 1$ for all $i \in \mathcal{F}_t$.

Throughout this work, we shall assume that the components of $\mathbf{f}_t$ are distributed as follows:

$$f_t(i) \sim \begin{cases} \mathbf{U}\left\{[-1,-\tau] \cup [\tau,1]\right\} & , i \in \mathcal{F}_t \\ \mathcal{N}(0,\sigma_b^2) & , i \notin \mathcal{F}_t \end{cases} , \tag{4.2}$$

where each component is assumed to be independent of the others, $\mathbf{U}$ denotes the uniform distribution, and $\mathcal{N}$ denotes the Gaussian distribution. We have approximated the intensity distribution of those pixels not in $\mathcal{F}_t$ as a zero-mean Gaussian under the assumption that σ_b^2 is much smaller than τ.

Following the work of Cevher *et al.* [11], we seek to perform background sub-

traction in the compressive domain. Often, it is the case that the foreground occupies only a very small portion of the image plane, i.e., $|\mathcal{F}_t| << N^2$. Given the foreground model (4.2), this implies that $\mathbf{f}_t$ is compressible in the spatial domain. Therefore, if $\mathbf{b}$ is known, we can use it, the image model (4.1), and compressive image measurements $\mathbf{y}_t = \mathbf{\Phi}_t \mathbf{x}_t$ to generate the following estimate of $\mathbf{f}_t$:

$$\hat{\mathbf{f}}_t = \Delta(\boldsymbol{\xi}_t, \mathbf{\Phi}_t) \quad , \tag{4.3}$$

where $\boldsymbol{\xi}_t = \mathbf{y}_t - \boldsymbol{\beta}_t$, $\boldsymbol{\beta}_t = \mathbf{\Phi}_t \mathbf{b}$, and Δ is defined as in (2.2).

As we will discuss in Section 4.4, we construct $\mathbf{\Phi}_t$ by taking a subset of rows from a fixed $N^2 \times N^2$ matrix, $\mathbf{\Phi}$, and rescaling the result. We can therefore calculate $\boldsymbol{\beta}_t$ from $\boldsymbol{\beta} = \mathbf{\Phi}\mathbf{b}$ by similarly dropping components and rescaling. Noting (4.2), a maximum-likelihood estimate of $\boldsymbol{\beta}$ can be found by computing the mean of compressive measurements of a background-only video sequence, i.e.,

$$\boldsymbol{\beta} = \frac{1}{J} \sum_{j=1}^{J} \mathbf{y}_j \quad , \tag{4.4}$$

where $\mathbf{y}_j = \mathbf{\Phi}\mathbf{x}_j$ and $|\mathcal{F}_j| = 0$ for all j in the summation. As was proposed by Cevher *et al.* [11], these measurements can be obtained in advance by using the full sensing matrix, $\mathbf{\Phi}$, to observe the scene when it is known that there is no foreground component.

4.4 Sensing Matrix Design

In this section, we will discuss our method for constructing adaptive rate measurement matrices for the purpose of recovering sparse signals from a minimal amount of measurements.

4.4.1 Theoretical Guarantees

In Section 2.1, we presented a theoretical result from CS literature that states that Δ will exactly recover an s-sparse $\mathbf{f}$ from $\boldsymbol{\xi}$ if $\boldsymbol{\Phi}$ exhibits the RIP of order $2s$ with $\delta_{2s} \leq \sqrt{2} - 1$. One of the most prevalent methods discussed in the literature for constructing such matrices involves drawing each matrix entry from a Gaussian distribution with parameters that depend on the number of rows that the matrix possesses. For $\boldsymbol{\Phi} \in \mathbb{R}^{M \times N^2}$, this technique defines entries ϕ_{ij} as independent realizations of a Gaussian random variable with zero mean and variance $1/M$, i.e.,

$$\phi_{ij} \sim \mathcal{N}(0, 1/M) \quad . \tag{4.5}$$

Baraniuk *et al.* [79] provide the following theoretical result for this construction technique: for a given $\delta \in (0, 1)$ and positive integers M and s, $\boldsymbol{\Phi} \in \mathbb{R}^{M \times N^2}$ constructed according to (4.5) exhibits the RIP of order s with $\delta_s = \delta$ with probability exceeding

$$1 - 2e^{-c_0(\delta/2)M + s\left(\log(eN^2/s) + \log(12/\delta)\right)} \quad , \tag{4.6}$$

where $c_0(x) = x^2/4 - x^3/6$.

The scenarios discussed here require us to find the minimum M that will ensure the constructed matrix can successfully recover s-sparse signals. Therefore, we now consider the case where δ, s, and N^2 are fixed. If we impose a lower bound, τ_g, on the probability of success given by (4.6), rearranging terms reveals that the theory

requires

$$M \geq \frac{s[1 + \log(\frac{N^2}{s}) + \log(\frac{12}{\delta})] + \log(\frac{2}{1-\tau_g})}{\frac{\delta^2}{16}(1 - \frac{\delta}{3})} \quad . \tag{4.7}$$

For practical measurement matrices, we are only interested in the case where $N^2 \geq M$ (i.e., matrices for which compression actually occurs). Combining this requirement with (4.7) yields the following lower bound for N^2/s:

$$\frac{N^2}{s} \geq \frac{\log(\frac{N^2}{s}) + \frac{1}{s}\log(\frac{2}{1-\tau_g})}{\frac{\delta^2}{16}(1 - \frac{\delta}{3})} + \frac{1 + \log(\frac{12}{\delta})}{\frac{\delta^2}{16}(1 - \frac{\delta}{3})}. \tag{4.8}$$

For s-sparse signals, the reconstruction guarantee that accompanies Δ requires that $\mathbf{\Phi}$ exhibits the RIP of order $2s$ with $\delta_{2s} \leq \sqrt{2} - 1$. Using only the second term of the lower bound in (4.8) and noting that the first term is always positive, we see that requiring such a δ_{2s} means that s/N^2 can be no greater than ~ 0.0011.

In our system, s/N^2 represents the percentage of foreground pixels in the image, and it is unreasonable to expect that this quantity will never exceed 0.11%. Therefore, if we wish to use CS for compression (i.e., with a measurement matrix that has fewer rows than columns), we must design and use matrices without the guarantee provided by the above result. However, that result is merely sufficient: in the next part, we will experimentally show that similarly-constructed matrices with far fewer rows are indeed still able to provide measurements that enable accurate sparse signal reconstruction.

4.4.2 Practical Sensing Matrix Design Based on Phase Diagrams

Given a candidate sensing matrix construction technique, Donoho and Tanner [80] discuss an associated *phase diagram*: a numerical representation of how

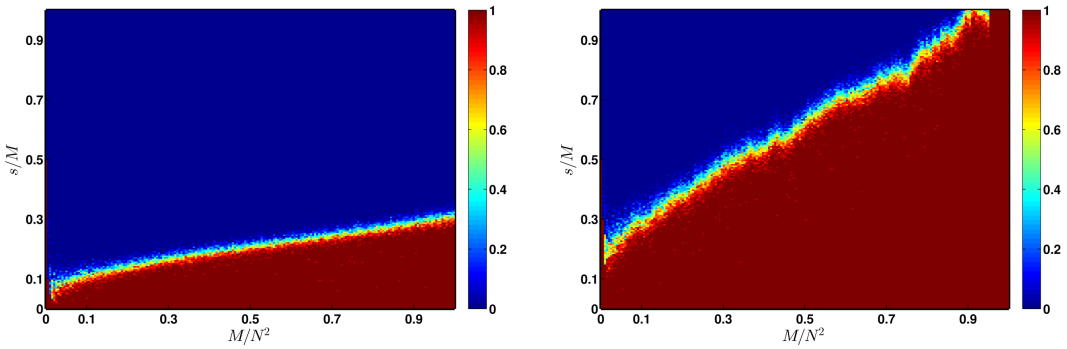
useful the generated matrices are for CS. Specifically, the ratios M/N^2 (signal undersampling) and s/M (signal sparsity) are considered. A phase diagram is a function defined over the *phase space* $(M/N^2, s/M) \in [0,1]^2$. We discretize this space and perform multiple sense-and-reconstruct experiments at each grid point in order to approximate the phase diagram there: the value of M/N^2 provides the information necessary for matrix construction, and s/M provides the information necessary to generate random sparse signals. We make the approximation using the percentage of trials that result in successful signal recovery, which we define as a normalized ℓ_2 reconstruction error of 10^{-3} or less.

Even though we cannot use the theoretical guarantee discussed earlier in this section, the first matrix construction technique we use is based on randomly-generated matrices that rely on independent realizations of a Gaussian random variable. Specifically, we use the following construction technique: we generate $\mathbf{\Phi} \in \mathbb{R}^{N^2 \times N^2}$ by drawing each entry according to (4.5). Then, for a given value of M_t, we form the corresponding $M_t \times N^2$ matrix $\mathbf{\Phi}_t$ via

$$\mathbf{\Phi}_t = \sqrt{\frac{N^2}{M_t}}\, \mathbf{\Phi}_{1:M_t} \quad , \tag{4.9}$$

where $\mathbf{\Phi}_{1:M_t}$ denotes the submatrix of $\mathbf{\Phi}$ corresponding to the first M_t rows. The scaling factor ensures that the relationship between the variance and the number of rows defined in (4.5) is preserved.

We also analyze a second matrix construction technique based on the discrete Fourier transform (DFT). Specifically, we generate $\mathbf{\Phi} \in \mathbb{C}^{N^2 \times N^2}$ by randomly permuting the rows of the DFT matrix and form $\mathbf{\Phi}_t$ according to (4.9).



(a) Gaussian (b) Fourier

Figure 4.2: Phase diagrams for Gaussian and Fourier measurement ensembles. Color corresponds to probability of successful reconstruction (here, normalized ℓ_2 error below 10^{-3}).

In this work, we will make predictions regarding the sparsity of the signals we are about to observe. Given a prediction s_t, we will seek the minimum M_t such that (4.9) generates a sensing matrix capable of providing enough measurements to ensure accurate reconstruction of s_t-sparse signals. In order to determine the mapping from s_t to M_t, we use the associated phase diagram. We construct this diagram (see Figure 4.2) during a one-time, offline analysis. Then, given s_t and a minimum probability of reconstruction success $\tau_d \in (0, 1)$, we use the phase diagram as a lookup table to find the smallest value of M_t that yields at least a τ_d success rate for s_t-sparse signals.

4.5 Method I: Cross Validation

In this section, we describe a rate-adaptive CS method that utilizes a set of linear cross-validation measurements $\boldsymbol{\chi}_t = \boldsymbol{\Psi}\mathbf{x}_t$. An earlier version of this work was presented by Warnell *et al.* [81].

4.5.1 Compressive Sensing with Cross Validation

Let $\boldsymbol{\xi}_t \in \mathbb{C}^{M_t}$ be a set of compressive measurements of a sparse signal $\mathbf{f}_t \in \mathbb{R}^{N^2}$ obtained using $\boldsymbol{\Phi}_t$, i.e., $\boldsymbol{\xi}_t = \boldsymbol{\Phi}_t\mathbf{f}_t$. In this section, we will use $\hat{\mathbf{f}}_t^{(s)}$ to denote the s-sparse point estimate of this signal obtained using $\Delta(\boldsymbol{\xi}_t, \boldsymbol{\Phi}_t)^{(s)}$, where Δ is defined as in (2.2) and $\cdot^{(s)}$ denotes a truncation operation that sets all but the s largest-magnitude components of the vector-valued argument to zero.

Ward [82] bounds the error of the above estimate using a cross-validation technique that is based on the Johnson-Lindenstrauss lemma [83]. At the same time $\boldsymbol{\xi}_t$ is collected, we use a static *cross-validation matrix* $\boldsymbol{\Psi} \in \mathbb{C}^{r\times N^2}$ to collect *cross-validation measurements* $\boldsymbol{\gamma}_t = \boldsymbol{\Psi}\mathbf{f}_t$. We construct $\boldsymbol{\Psi}$ by drawing each of its entries from an i.i.d. Bernoulli distribution with zero mean and variance $1/r$. Such a construction leads to the following statement: for given accuracy and confidence parameters ϵ and ρ (respectively), $r \geq 8\epsilon^{-2}\log\frac{1}{2\rho}$ rows suffice to ensure that

$$(1-\epsilon)^2 \leq \frac{\|\mathbf{f}_t - \hat{\mathbf{f}}_t^{(s)}\|_2^2}{\|\boldsymbol{\gamma}_t - \boldsymbol{\Psi}\hat{\mathbf{f}}_t^{(s)}\|_2^2} \leq (1+\epsilon)^2 \tag{4.10}$$

with probability exceeding $1-\rho$.

Let $e_s(\mathbf{f}_t)_p$ denote the optimal s-sparse approximation error measured with

respect to the ℓ_p norm, i.e.,

$$e_s(\mathbf{f}_t)_p = \underset{\|\mathbf{z}\|_0 \leq s}{\arg\min} \|\mathbf{f}_t - \mathbf{z}\|_p \quad , \tag{4.11}$$

where the ℓ_p-norm is given by $\|\mathbf{x}\|_p = (\sum_i |x(i)|^p)^{1/p}$. Using the fact that $\hat{\mathbf{f}}_t^{(s)}$ is s-sparse, the upper bound in (4.10) can be extended to $e_s(\mathbf{f}_t)_2^2$ as follows:

$$e_s(\mathbf{f}_t)_2^2 \leq \|\mathbf{f}_t - \hat{\mathbf{f}}_t^{(s)}\|_2^2 \leq (1+\epsilon)^2 \|\boldsymbol{\gamma}_t - \boldsymbol{\Psi}\hat{\mathbf{f}}_t^{(s)}\|_2^2 \quad . \tag{4.12}$$

That is, the observable CV error can be used to upper bound the unobservable optimal s-sparse approximation error.

4.5.2 Adaptive-Rate Compressive Sensing via Cross Validation

Let s_t denote the true value of the foreground sparsity at time t, i.e., $s_t = |\mathcal{F}_t|$. The method we present here relies on an estimate of this quantity, which we denote as $\hat{s}_t$. Before sensing begins at time t, we assume $\mathbf{f}_t$ to be $\hat{s}_t$-sparse, and select the corresponding minimal M_t (and thus $\boldsymbol{\Phi}_t$) according to the phase diagram technique described in Section 4.4. We then use $\boldsymbol{\Phi}_t$ and $\boldsymbol{\Psi}$ to collect $\mathbf{y}_t$ and $\boldsymbol{\chi}_t$. Using the technique described in Section 4.3, we can find $\boldsymbol{\xi}_t$ and form the foreground estimate $\hat{\mathbf{f}}_t^{(\hat{s}_t)}$. In a similar fashion, we can also find $\boldsymbol{\gamma}_t$ by subtracting a precalculated set of cross-validation measurements of the static signal component, $\boldsymbol{\zeta} = \boldsymbol{\Psi}\mathbf{b}$, from $\boldsymbol{\chi}_t$. Finally, we select $\hat{s}_{t+1}$ based on the result of a multiple hypothesis test that uses $\boldsymbol{\gamma}_t$ and $\hat{\mathbf{f}}_t^{(\hat{s}_t)}$.

We formulate the multiple hypothesis test by first assuming that we are able to observe $e_{\hat{s}_t}(\mathbf{f}_t)_2^2$. Of course, this is not possible. Nevertheless, we define the

null hypothesis, $\mathbf{H}_0$, as the scenario under which $\hat{s}_t$ exceeds s_t. If this hypothesis is true, then $\mathbf{f}_t^{(\hat{s}_t)}$ (i.e., the optimal $\hat{s}_t$-sparse approximation to $\mathbf{f}_t$) captures all s_t foreground components and $(\hat{s}_t - s_t)$ background components while neglecting the remaining $(N^2 - \hat{s}_t)$ background components. Using (4.2) and ignoring the effect of component ordering for the relatively narrow background component distribution, it can be shown that $e_{\hat{s}_t}(\mathbf{f}_t)_2^2$ is a random variable with mean, μ_0, and variance, σ_0^2, given by

$$\begin{aligned} \mu_0 &= (N^2 - \hat{s}_t)\sigma_b^2 \\ \sigma_0^2 &= 2(N^2 - \hat{s}_t)\sigma_b^4 \quad . \end{aligned} \tag{4.13}$$

We also define a set of hypotheses that are possible when $\mathbf{H}_0$ is not true. Let $\mathbf{H}_k$, $k \in \{\hat{s}_t + 1, \ldots, N^2\}$, describe the scenario under which $s_t = k$. Under $\mathbf{H}_k$, $\mathbf{f}_t^{(\hat{s}_t)}$ does not capture all k foreground components: it neglects the smallest $(k - \hat{s}_t)$ of them and the $(N^2 - k)$ background components. Let $\{Z_1, \ldots, Z_{k-\hat{s}_t}\}$ denote the unordered set of random variables corresponding to the neglected foreground components. Further, let $W_{(\hat{s}_t)}$ denote the $\hat{s}_t^{\text{th}}$ order statistic in a set of k i.i.d. uniform random variables distributed over the interval $[\tau, 1]$ (i.e., the set of random variables corresponding to the foreground components of $\mathbf{f}_t$ under $\mathbf{H}_k$. Then $\left\{Z_1|W_{(\hat{s}_t)}, Z_2|W_{(\hat{s}_t)}, \ldots, Z_{k-\hat{s}_t}|W_{(\hat{s}_t)}\right\}$ are i.i.d. uniform over the interval $\left[\tau, W_{(\hat{s}_t)}\right]$.

Therefore, given $W_{(\hat{s}_t)} = w$, the mean and variance of $e_k(\mathbf{f}_t)_2^2$ under $\mathbf{H}_k$ are given by

$$\begin{aligned}
\mu_k|_{W_{(\hat{s}_t)}=w} &= \frac{1}{2}(k-\hat{s}_t)(\tau+w\tau+w^2) \\
&\quad + (N^2-k)\sigma_b^2 \\
\sigma_k^2|_{W_{(\hat{s}_t)}=w} &= \frac{1}{9}\left[(k-\hat{s}_t)^2-(k-\hat{s}_t)\right](\tau^2+w\tau+w^2)^2 \\
&\quad + \frac{1}{5}(k-\hat{s}_t)(\tau^4+w\tau^3+w^2\tau^2+w^3\tau+w^4) \\
&\quad + \left[(N^2-k)^2+2(N^2-k)\right]\sigma_b^4 \\
&\quad + (k-\hat{s}_t)(N^2-k)(\tau^2+w\tau+w^2)\sigma_b^2 - \mu_k^2|_{W_{(\hat{s}_t)}=w}
\end{aligned} \tag{4.14}$$

To find μ_k and σ_k^2, the mean and variance for the marginal distribution for $e_{\hat{s}_t}(\mathbf{f}_t)_2^2$ under $\mathbf{H}_k$, respectively, one can integrate over the distribution for $W_{(\hat{s}_t)}$, i.e.,

$$\begin{aligned}
\mu_k &= \int_\tau^1 \mu_k|_{W_{(\hat{s}_t)}=w}\; p_{W_{(\hat{s}_t)}}(w)dw \\
\sigma_k^2 &= \int_\tau^1 \sigma_k^2|_{W_{(\hat{s}_t)}=w}\; p_{W_{(\hat{s}_t)}}(w)dw,
\end{aligned} \tag{4.15}$$

where the pdf for $W_{(\hat{s}_t)}$ is given explicitly by

$$p_{W_{(\hat{s}_t)}}(w) = \frac{1}{1-\tau}\text{Beta}\left(\frac{w-\tau}{1-\tau}; \hat{s}_t, k-\hat{s}_t+1\right), \tag{4.16}$$

and $\text{Beta}(w;\alpha,\beta)$ is the pdf for a random variable distributed as $\text{Beta}(\alpha,\beta)$ evaluated at w.

Unfortunately, the integrals specified in (4.15) are very difficult to compute. Therefore, we set

$$\begin{aligned}
\mu_k &= \mu_k|_{W_{(\hat{s}_t)}=1} \\
\sigma_k^2 &= \sigma_k^2|_{W_{(\hat{s}_t)}=1},
\end{aligned} \tag{4.17}$$

and this approximation is validated by the results presented in Section 4.7.

The multi-hypothesis test described above can be succintly written as

$$\begin{aligned} \mathbf{H}_0 &: s_t < \hat{s}_t \\ \mathbf{H}_k &: s_t = k \end{aligned} \tag{4.18}$$

for $k \in \{\hat{s}_t + 1, \ldots, N\}$. Let q_k denote the probability density function for $e_{\hat{s}_t}(\mathbf{f}_t)_2^2$ under the assumption that $\mathbf{H}_k$ is true for $k \in \{0, \hat{s}_t + 1, \ldots, N\}$. We will evaluate explicit assumptions regarding the form of q_k in Section 4.7. The optimal decision rule for (4.18) under the minimum probability of error criterion with an equal prior for each hypothesis is given by

$$k^* = \underset{k \in \{0, \hat{s}_t+1, \ldots, N\}}{\arg\max} \; q_k \left(e_{\hat{s}_t}(\mathbf{f}_t)_2^2 \right) \quad . \tag{4.19}$$

Assuming that the sparsity of $\mathbf{f}_t$ is a slowly-varying quantity, we choose to set $\hat{s}_{t+1}$ equal to what we believe s_t to be. Note here that there is no assumption on how the *location* of the support of $\mathbf{f}_t$ changes, only on how its cardinality varies. If $k^* = 0$, it is our belief that $\hat{s}_t > s_t$, and we expect that the error in $\hat{\mathbf{f}}_t^{(\hat{s}_t)}$ to be very small. Therefore, we find the set of foreground entries for this signal, $\hat{\mathcal{F}}_t = \{i \;\; : \;\; |\hat{f}_t^{(\hat{s}_t)}(i)| \geq \tau\}$, and set $\hat{s}_{t+1} = |\hat{\mathcal{F}}_t|$. For any other value of k^*, we set $\hat{s}_{t+1} = k^*$.

Unfortunately, it is impossible to directly observe $e_{\hat{s}_t}(\mathbf{f}_t)_2^2$. However, we can upper bound this quantity using the cross-validation measurements as specified in (4.12). Therefore, we propose the following modification to (4.19):

$$k^* = \underset{k \in \{0, \hat{s}_t+1, \ldots, N\}}{\arg\max} \; q_k \left((1+\epsilon)^2 \|\boldsymbol{\gamma}_t - \boldsymbol{\Psi} \hat{\mathbf{f}}_t^{(\hat{s}_t)}\|_2^2 \right) \quad . \tag{4.20}$$

Note that using the upper bound in (4.20) will potentially yield a different k^* than that which would have been selected by (4.19). However, as we will show in Section 4.7, the impact on system performance is minimal: using the cross-validation bound still allows us to successfully achieve a minimal measurement rate while maintaining reconstruction fidelity.

We term the strategy we have outlined above *adaptive-rate compressive sensing via cross validation (ARCS-CV)* and summarize the procedure in Algorithm 2.

Algorithm 2: ARCS-CV for Background Subtraction

Require: $\mathbf{\Phi}, \mathbf{\Psi}, \hat{s}_t, \boldsymbol{\beta}, \boldsymbol{\zeta}, \sigma_b^2, \tau$

Select M_t using $\hat{s}_t$ and the phase diagram lookup table

Form $\mathbf{\Phi_t}$ and $\boldsymbol{\beta}_t$

Obtain image measurements $\mathbf{y_t}$, $\boldsymbol{\chi}_t$

Compute foreground-only measurements $\boldsymbol{\xi}_t$, $\boldsymbol{\gamma}_t$

Estimate foreground: $\hat{\mathbf{f}}_t^{(\hat{s}_t)} = \Delta(\boldsymbol{\xi}_t, \mathbf{\Phi}_t)^{(\hat{s}_t)}$

Compute k^* using (4.20)

if $k^* = 0$ **then**

$\hat{s}_{t+1} = |\hat{\mathcal{F}}_t|$

else

$\hat{s}_{t+1} = k^*$

end if

4.6 Method II: Low-Resolution Tracking

In this section, we propose an adaptive method that utilizes a much richer form of side information than the random projections of the previous section: low-resolution images, Z_t, that have been captured using a traditional (i.e., non-compressive) camera.

4.6.1 Low-Resolution Measurements

We assume that the low- and high-resolution images, $Z_t \in \mathbb{R}^{L\times L}$ and $X_t \in \mathbb{R}^{N\times N}(L < N)$, repectively, are related by a simple downsampling operation. Let $\mathbf{t}_Z = \begin{bmatrix} t_Z^x & t_Z^y \end{bmatrix}^T$ denote the coordinates of a pixel in the image plane of the low-resolution camera. If we use $\mathbf{t}_X = \begin{bmatrix} t_X^x & t_X^y \end{bmatrix}^T$ to denote the corresponding coordinate in the image plane of the compressive camera, the effect of the downsampling operation on coordinates is given by

$$\mathbf{t}_X = \begin{bmatrix} D & 0 & -\frac{D-1}{2} \\ 0 & D & -\frac{D-1}{2} \end{bmatrix} \begin{bmatrix} \mathbf{t}_Z \\ 1 \end{bmatrix} \quad , \tag{4.21}$$

where we assume the dowsampling factor, $D = N/L$, to be an integer. Using (4.21), each pixel in Z_t maps to the center of a unique $D \times D$ block of pixels in X_t. The effect of the downsampling operation on image intensity is given by averaging the intensities within this block, i.e.,

$$Z_t(\mathbf{t}_Z) = \frac{1}{D^2} \sum_{\mathbf{t}_X \in \mathcal{B}(\mathbf{t}_Z)} X_t(\mathbf{t}_X) \quad ,$$

where the coordinates of the pixels in the block are given explicitly as

$$\mathcal{B}(\mathbf{t}_Z) = \{(\mathbf{t}_Z^x - 1)D + 1, \ldots, \mathbf{t}_Z^x D\} \times$$

$$\{(\mathbf{t}_Z^y - 1)D + 1, \ldots, \mathbf{t}_Z^y D\}\,.$$

4.6.2 Object Tracking and Foreground Sparsity

We assume that we are able to track the foreground objects in the low-resolution video. Specifically, we assume that at each time index, we are able to estimate a zero-skew affine warp parameter $\mathbf{p}_t = \begin{bmatrix} p_t(1) & \cdots & p_t(4) \end{bmatrix}^T$ that maps coordinates in an object template image, T, to their corresponding location in Z_t. Using $\mathbf{t}_T$ to denote a pixel coordinate in T, $\mathbf{p}_t$ specifies the corresponding coordinate in Z_t via

$$\mathbf{t}_Z = \begin{bmatrix} p_t(1) & 0 & p_t(3) \\ 0 & p_t(2) & p_t(4) \end{bmatrix} \begin{bmatrix} \mathbf{t}_T \\ 1 \end{bmatrix} \quad . \tag{4.22}$$

We further assume that the time-evolution of $\mathbf{p}_t$ is governed by a known Markov dynamical system, i.e.,

$$\mathbf{p}_t = \mathbf{u}_t\left(\mathbf{p}_{t-1}, \boldsymbol{\eta}_t\right) \quad , \tag{4.23}$$

for known $\mathbf{u}_t$ and i.i.d. system noise $\boldsymbol{\eta}_t$.

Let $\{\mathbf{t}_i \,:\, i \in \mathbb{Z}/4\mathbb{Z}\}$ be the set of corner coordinates of T in any order that traces its outline. Then, given $\mathbf{p}_t$, we can calculate the position of the tracked object's bounding box in the high-resolution F_t using (4.22) and (4.21). We shall assume that the area of this bounding box specifies the number of foreground components in $\mathbf{f}_t$, i.e., s_t. If this area is not integer-valued, we simply round up. Using

the well-known formula for the area of a polygon from its corner coordinates, s_t can be written as $s_t = h(\mathbf{p}_t)$, where

$$h(\mathbf{p}_t) = \left\lceil \left| \frac{D^2[p_t(1)p_t(4) - p_t(2)p_t(3)]}{2} \sum_{i \in \mathbb{Z}/4\mathbb{Z}} \mathcal{T}(i) \right| \right\rceil , \tag{4.24}$$

and $\mathcal{T}(i) = t_i^x t_{i+1}^y - t_i^y t_{i+1}^x$. Above, $\lceil \cdot \rceil$ represents the ceiling function.

From (4.24), it is clear that the distribution of the random variable s_t is a function of the distribution of $\mathbf{p}_t$. For the remainder of this section, we will use $q_t(s_t) = p(s_t|\mathbf{p}_t)$ to denote the corresponding probability mass function.

Figure 4.3 illustrates the relationship between a typical high- and low-resolution image pair and shows an example bounding box found by a tracker operating on the low-resolution image.

4.6.3 Sparsity Estimation

We now turn our attention to selecting a value to use for s_t, $\hat{s}_t$, on the basis of the previous image's track, $\mathbf{p}_{t-1}$. Once a value has been selected, we use the method presented in Section 4.4 to select a minimal M_t and the corresponding $\mathbf{\Phi}_t$. We then use $\mathbf{\Phi}_t$ to collect compressive measurements of X_t and calculate $\boldsymbol{\xi}_t$. Using this procedure, the Δ-generated estimate $\hat{\mathbf{f}}_t$ will obey

$$\|\mathbf{f}_t - \hat{\mathbf{f}}_t\|_2 \leq \frac{C_0 e_{\hat{s}_t}(\mathbf{f}_t)_1}{\sqrt{\hat{s}_t}} \quad , \tag{4.25}$$

where $e_{\hat{s}_t}(\cdot)_1$ represents the optimal $\hat{s}_t$-sparse ℓ_1 estimation error [12]. The value of the constant in (4.25) is given explicitly by

$$C_0 = \frac{2 - (2 - \sqrt{2})\delta_{2\hat{s}_t}}{1 - (1 - \sqrt{2})\delta_{2\hat{s}_t}} \quad .$$

(a) (b)

Figure 4.3: Illustration of the downsampling and low-resolution tracking process utilized by ARCS-LRT for a sample image from the PETS_2009 dataset. (a) corresponds to the high-resolution image for which we seek to perform compressive foreground reconstruction. (b) corresponds to the low-resolution obtained by the secondary, non-compressive camera. The bounding box around the subject corresponds to the output of a tracking algorithm.

One criterion we will consider when selecting $\hat{s}_t$ is the expected value of the ℓ_2 reconstruction error, i.e., we would like $\hat{s}_t$ to minimize $\mathbb{E}\left\{\|\mathbf{f}_t - \hat{\mathbf{f}}_t\|_2\right\}$. However, since the nonlinearity of Δ makes determining the statistics of that quantity very difficult, we instead look to minimize the right-hand side of (4.25). It is easy to see that this quantity can be minimized by selecting $\hat{s}_t$ as high as possible, but such a selection would provide no compression. Therefore, inspired by results from the model-order selection literature [84] [85] [86], we penalize larger values of $\hat{s}_t$ and instead propose to select $\hat{s}_t$ by solving

$$\hat{s}_t = \arg\min_{\hat{s}} \ \mathbb{E}\left\{\frac{C_0 e_{\hat{s}}(\mathbf{f}_t)_1}{\sqrt{\hat{s}}}\right\} + \lambda\hat{s} \quad , \tag{4.26}$$

where λ is an importance factor that specifies the tradeoff between low reconstruction error and a small sparsity estimate.

Using the law of total expectation, the foreground model (4.2), and techniques and approximations similar to those used in Section 4.5, we can rewrite (4.26) as

$$\hat{s}_t = \arg\min_{\hat{s}} \frac{C_0}{\sqrt{\hat{s}}}\left[\mathcal{J}_0(\hat{s}) + \mathcal{J}_1(\hat{s})\right] + \lambda\hat{s} \quad , \tag{4.27}$$

where

$$\mathcal{J}_0 = \sum_{k=1}^{\hat{s}} \sqrt{2/\pi}(N-\hat{s})\sigma_b q_t(k) \tag{4.28}$$

$$\mathcal{J}_1 = \sum_{k=\hat{s}+1}^{N} \left[(k-\hat{s})(1+\tau)/2 + \sqrt{2/\pi}(N-k)\sigma_b\right] q_t(k). \tag{4.29}$$

We term the strategy outlined above as *adaptive-rate compressive sensing via low-resolution tracking (ARCS-LRT)* and summarize the procedure in Algorithm 3.

Algorithm 3: ARCS-LRT for Background Subtraction

Require: $\boldsymbol{\Phi}, \hat{s}_t, \boldsymbol{\beta}, \sigma_b^2, \tau, \lambda$

Select M_t using $\hat{s}_t$ and the phase diagram lookup table

Form $\boldsymbol{\Phi}_\mathbf{t}$ and $\boldsymbol{\beta}_t$

Obtain image measurements $\mathbf{y_t}$, $\mathbf{z}_t$

Compute foreground-only measurements $\boldsymbol{\xi}_t$

Estimate foreground: $\hat{\mathbf{f}}_t = \Delta(\boldsymbol{\xi}_t, \boldsymbol{\Phi}_t)$

Compute low-resolution object track $\mathbf{p}_t$

Compute q_{t+1} via (4.23) and (4.24)

Compute $\hat{s}_{t+1}$ by solving (4.27)

4.7 Experiments

We tested the proposed algorithms on real video sequences captured using traditional cameras. The compressive, cross-validation, and low-resolution measurements were simulated in software. The SPGL1 [87] [88] software package was used to implement the decoding procedure (2.2). Three video sequences were used: `convoy2`, `marker_cam`, and `PETS2009_S2L1`. `convoy2` is a video of vehicles driving past a stationary camera. The vehicles comprise the foreground, and the foreground sparsity varies as a result of these vehicles sequentially entering and exiting the camera's field of view. `marker_cam` is a video sequence we captured using a surveillance camera mounted to the side of our building at the University of Maryland, College Park. The sequence begins with a single pedestrian walking in a parking lot, and

Table 4.1: Parameter values used in experiments

	σ_b^2	τ	$\mathbf{\Sigma}$	λ
convoy2	$\frac{4}{255}^2$	0.1	diag([1.0 1.0 3.0 3.0])	0.045
marker_cam	$\frac{4}{255}^2$	0.1	diag([1.0 1.0 3.0 3.0])	1.5
PETS2009_S2L1	$\frac{4}{255}^2$	0.1	diag([1.0 1.0 3.0 3.0])	0.15

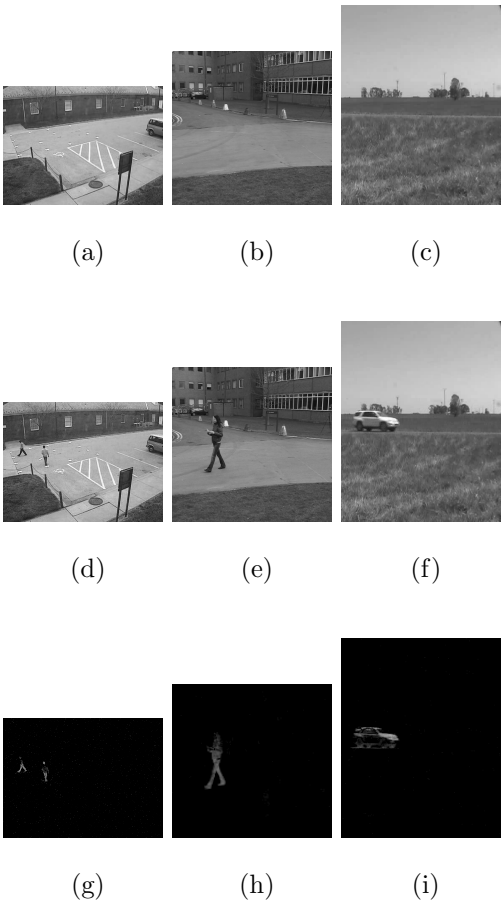
a second pedestrian joins halfway through the sequence. The two pedestrians comprise the foreground, and the foreground sparsity varies due to appearance shifts and the entrance of the second pedestrian. The PETS2009_S2L1 video sequence is a segment taken from the PETS 2009 benchmark data [8]. This sequence consists of four pedestrians entering and exiting the camera's field of view. The foreground sparsity changes as a function of the number and appearance of pedestrians. Example images from each dataset are provided in Figure 4.4.

4.7.1 Practical Considerations

Implementation of the ARCS methods presented in Sections 4.5 and 4.6 requires certain practical choices. In this part, we describe the choices we made that generated the results presented later in this section. Specific choices for parameter values for each video sequence are given in Table 4.1.

4.7.1.1 Foreground Model

The foreground model specified in (4.2) is parameterized by σ_b^2 and τ. The value that should be used for σ_b^2 will depend on the quality of the background



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 4.4: Example images from the `marker_cam`, `PETS2009_S2L1`, and `convoy2` (columns one, two, and three, respectively), video sequences. The first row contains the background images, the second row contains an image with both foreground and background components, and the third image contains the corresponding foreground component.

estimate, $\boldsymbol{\beta}$: the better (4.1) describes images in the video sequence, the smaller σ_b^2 can be. Since τ represents the foreground-background intensity threshold, its value depends on that of σ_b^2: τ should be set high enough to ensure that $\mathcal{N}(\tau; 0, \sigma_b^2)$ is sufficiently small, but low enough to ensure that it does not neglect intensities belonging to the foreground.

4.7.1.2 ARCS-CV

The ARCS-CV algorithm developed in Section 4.5 relies on the hypothesis test specified in (4.18). While we are able to calculate the first- and second-order moments of s_t under the various hypotheses, the maximum-likelihood decision rule (4.20) requires the entire probability density functions, q_k, for each. In our implementation, we approximate these densities by a normal distribution with mean and covariance specified by (4.13) and (4.14) under $\mathbf{H}_0$ and $\mathbf{H}_k$, respectively. That is, we make the approximation $q_k \approx \mathcal{N}(\mu_k, \sigma_k^2)$. As a consequence of this approximation, we observed that (4.20) sometimes yielded a nonzero k^* for sufficiently small cross-validation error upper bounds. However, when this upper bound is low, it is clear that we should select $\mathbf{H}_0$. Therefore, we explicitly impose a selection of $\mathbf{H}_0$ for cross-validation error upper bounds that are less than μ_0 by using

$$k^{**} = \begin{cases} 0, & (1+\epsilon)^2 \|\boldsymbol{\gamma}_t - \boldsymbol{\Psi}\hat{\mathbf{f}}_t^{(\hat{s}_t)}\|_2^2 < \mu_0 \\ k^*, & (1+\epsilon)^2 \|\boldsymbol{\gamma}_t - \boldsymbol{\Psi}\hat{\mathbf{f}}_t^{(\hat{s}_t)}\|_2^2 \geq \mu_0 \end{cases} \tag{4.30}$$

in place of (4.20) in Algorithm 2, where k^* represents the value obtained from (4.20).

4.7.1.3 ARCS-LRT

The ARCS-LRT method of Section 4.5 requires low-resolution object tracks in order to reason about the sparsity of the high-resolution foreground. In order to focus solely on the performance of the adaptive algorithm, we first determined these tracks manually, i.e., by hand-marking bounding boxes around each low-resolution foreground image. We only did this for images in which the object was fully visible. We shall also consider automatically-obtained tracks later in this section.

We used $\mathbf{u}_t(\mathbf{p}_{t-1}, \boldsymbol{\eta_t}) = \mathbf{p}_{t-1} + \boldsymbol{\eta_t}$ to define the system dynamics in (4.23) with $\boldsymbol{\eta_t} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ i.i.d. for each t, where the value of Σ should vary with the expected type of object motion.

Given this selection for $\mathbf{u}_t$, $p(\mathbf{p}_t|\mathbf{p}_{t-1}) = \mathcal{N}(\mathbf{p}_t; \mathbf{p}_{t-1}, \Sigma)$ represents our belief about the next track given the current one. Due to the complexity of h in (4.24), it is difficult to obtain an exact form for $p(s_t|\mathbf{p}_{t-1})$. Therefore, we used the unscented transformation [89] to obtain the first- and second-order moments, μ_{t+1} and σ^2_{t+1}, respectively. We then approximated $p(s_t|\mathbf{p}_{t-1})$ using the pdf for a discrete approximation to the normal distribution with the computed mean and covariance.

The sparsity estimator (4.27) requires values for both C_0 and λ. Since our phase diagram lookup table returns an M_t for which Δ recovers $\hat{s}_t$-sparse signals, we selected $\delta = 1/4 < \sqrt{2} - 1$. We set λ for each video sequence by trying many values and selecting one that resulted in low reconstruction error while maintaining a sparsity estimate that was close to the actual value.

Finally, we must compute a solution to (4.27). To do so, we used MATLAB's

Table 4.2: Experimental comparison of adaptive compressive sensing measurement strategies (oracle, ARCS-CV, ARCS-LRT)

	Average # of Measurements ($\bar{M}/N^2$)			Average Reconstruction Error (ℓ_2)		
	Oracle	ARCS-CV	ARCS-LRT	Oracle	ARCS-CV	ARCS-LRT
`marker_cam`	0.0598	0.0939	0.3356	1.4388	1.7802	1.8229
`PETS2009_S2L1`	0.1209	0.1530	0.4238	1.2811	1.6181	1.4911
`convoy2`	0.0997	0.1251	0.3627	1.6573	2.0296	2.6137

`fminbmd` function, which is based on golden selection search and parabolic interpolation [90].

4.7.2 Comparitive Results

In order to provide some context in which to interpret the results from our ARCS methods, we present them alongside those from the best-case sensing strategy: *oracle CS*. Oracle CS uses the true value of s_t as its sparsity estimate, which is impossible to obtain in practice. We compare the average measurement rates and foreground reconstruction errors for the three methods (oracle, ARCS-CV, and ARCS-LRT) in Table 4.2, and show the more detailed dynamic behavior in Figure 4.5. Note that the measurement values reported for the ARCS algorithms include the necessary overhead for the side information (i.e., the cross-validation and low-resolution measurements).

We first observe that the ARCS-LRT algorithm uses a significantly larger measurement rate than any of the others. This is due to the necessary overhead for

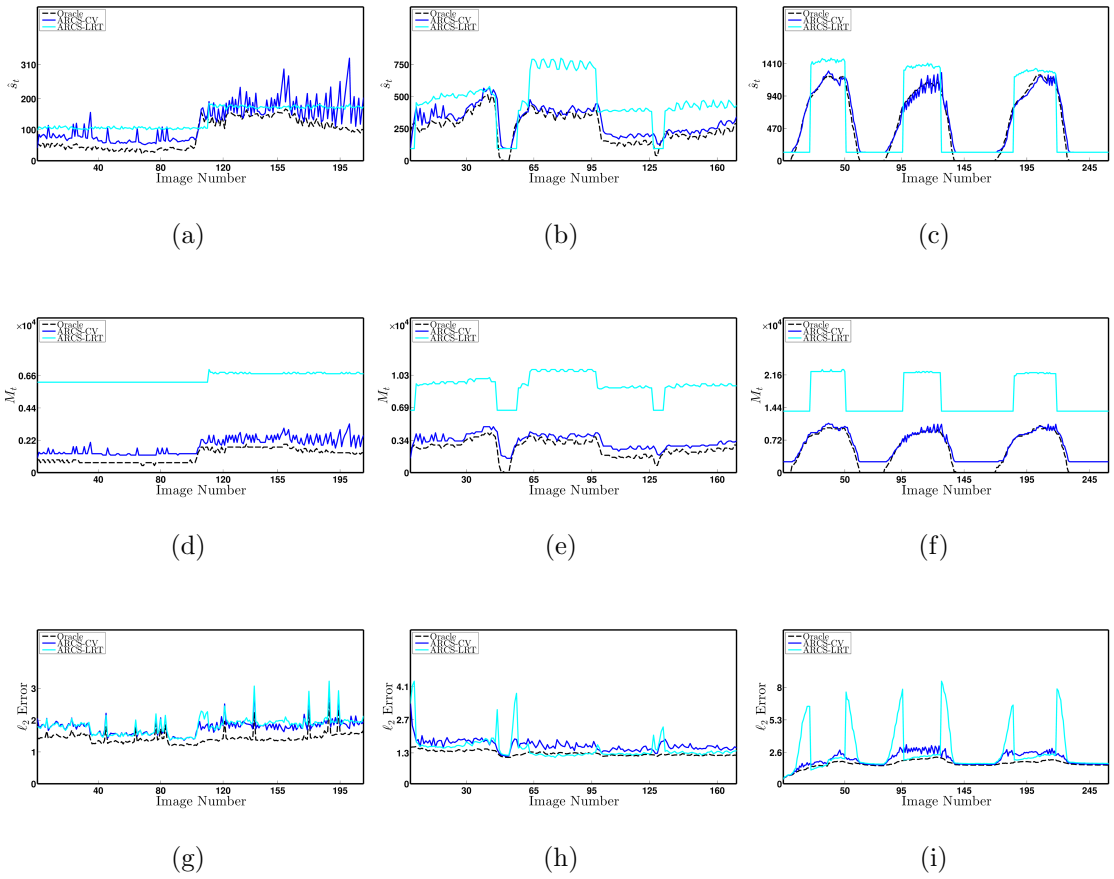


Figure 4.5: Performance of adaptive CS strategies for the `marker_cam` (column one), `PETS2009_S2L1` (column two), and `convoy2` (column three) video sequences. In the first row, $\hat{s}_t$ is used to denote the sparsity estimate used by each strategy. In row two, M_t is used to denote the total number of measurements that must be acquired. The ℓ_2 reconstruction error is plotted in row three.

the low-resolution side information. In our experiments, we used $L = N/2$, i.e. M_t is at least 25% of N^2. A smaller L could be selected at the risk of poorer low-resolution tracking. The ARCS-CV algorithm performs much better in terms of measurement rate since the side-information overhead is relatively small (for all datasets, r is less than 2% of N^2).

It can also be seen that the ARCS-LRT sparsity estimate lags behind the true foreground sparsity for those images in which an object is entering or exiting the camera's field-of-view but not fully visible. The phenomenon is especially visible in the third column (`convoy2`) of Figure 4.5. It is due to the fact that we have manually imposed the condition that the object cannot be tracked unless it is fully visible. This leads to the large spikes in foreground reconstruction error. However, when the object becomes fully visible, the low-resolution tracks provide the algorithm with enough information to monitor the high-resolution signal sparsity and the effect subsides.

4.7.3 Steady-State Behavior

We analyzed the behavior of our ARCS methods when the signal under observation is static (i.e., $\mathbf{f}_t = \mathbf{f}$ for all t). To do so, we created a synthetic data sequence by repeating a single image in the `convoy2` data set for which $s = 1233$. Figure 4.6 shows the behavior of each algorithm when the initial sparsity estimate, $\hat{s}_1$, is wrong. For each method, we ran two experiments. For the first one, we initialized the sparsity estimate using a value that was too low ($\hat{s}_1 = 0$). For the second one,

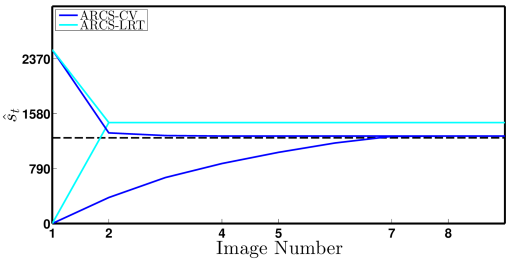
we initialized with a value that was too high ($\hat{s}_1 = 2500$). Note that both methods are able to successfully adapt to the true value of s, and the ARCS-LRT method adapts very quickly (requiring only a single image) due to the immediate availability of the low-resolution track.

4.7.4 ARCS-LRT and Automatic Tracking

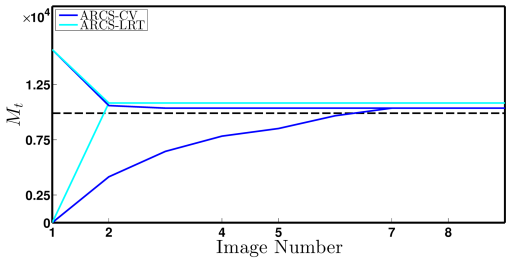
We also investigated the effect of using low-resolution tracks obtained via an automatic method. To do so, we implemented a simple blob tracker in MATLAB for the `convoy2` sequence and used the resulting tracks in the ARCS-LRT framework. A comparison of algorithm performance between using automatic tracks and our manually-marked tracks is shown in Figure 4.7. Given the negligible effect of the blob tracker on the behavior of ARCS-LRT, we would not expect more sophisticated automatic tracking techniques to negatively affect performance.

4.8 Summary

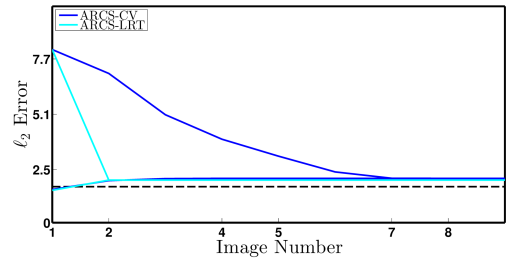
In this chapter, we described two techniques for using side information to adjust the measurement rate of a dynamic compressive sensing system. These techniques were developed in the specific context of using this system for video background subtraction. The first technique involves collecting side information in the form of a small number of extra cross-validation measurements and using an error bound to infer underlying signal sparsity. The second method uses side information from a secondary, low-resolution, traditional camera in order to infer the sparsity of



(a)

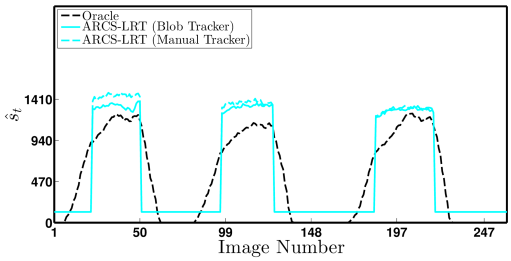


(b)

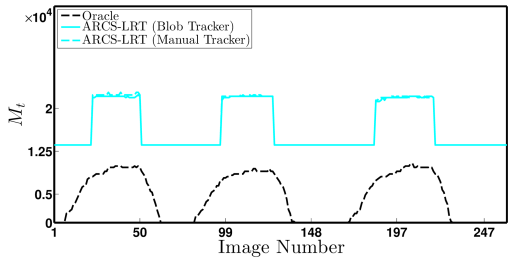


(c)

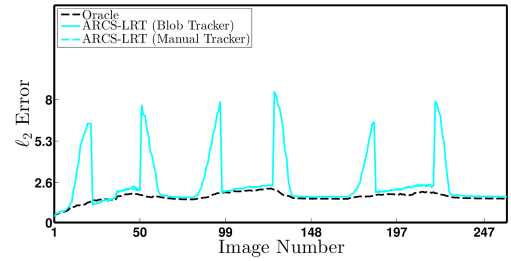
Figure 4.6: Steady-state behavior for both ARCS algorithms using a video sequence constructed by repeating a single image selected from the `convoy2` dataset. For each algorithm, two experimental paths are shown: one generated by initializing the sparsity estimate such that it is too small ($s_1 << s$), and the other generated by initializing the sparsity estimate such that it is too large ($s_1 >> s$).



(a)



(b)



(c)

Figure 4.7: Effect of manual vs. automatic blob tracking on the behavior of the ARCS-LRT method for the `convoy2` dataset.

the high-resolution images. In either case, we used a pre-computed phase diagram as a lookup table to map sparsity estimates to minimal compressive measurement rates. We validated these techniques on real video sequences using practical approximations for theoretical quantities.

Chapter 5: Multi-Image Visual Saliency

While closely related to traditional camera systems, pan-tilt-zoom cameras differ in that they are able to provide multi-image datasets that have a precisely-defined geometric relationship. The work we present in this chapter is motivated by the goal of creating an efficient camera-control technique for monitoring an environment with a PTZ camera. Biological vision seems to have achieved a similar goal: these systems operate in data-rich visual domains with limited sensing and processing resources, yet they often find remarkable success in accomplishing complex tasks such as visual search and object recognition. One of the primary mechanisms that seems to enable this achievement is the ability to discern *visual saliency*: the perceptual quality exhibited by certain portions of visual data that are, in some sense, more important than others. Once such data has been identified, biological systems employ adaptive sensing strategies: they focus their sensing and processing resources on these salient portions of information. This is accomplished via mechanisms such as gaze control, which provides high-quality information pertaining only to a very narrow field of view. If the gaze is directed appropriately, the spatially-limited amount of information it provides is often sufficient for the accomplishment of a given task.

Since this strategy is so effective for biological systems, it is natural to wonder if similar ideas can be used to aid artificial ones. In computational vision, it is often the case that only a small portion of the available data is actually relevant, or salient, to the end-goal (e.g., object recognition, tracking, etc.). In this chapter, we present our work toward enabling the use of visual saliency in adaptive sensing algorithms when the sensor of interest is a PTZ camera. We discuss existing computer vision techniques designed to quantify visual saliency and identify a new constraint, *consistency*, that should be enforced when the imagery of interest is collected using a PTZ camera. Further, we develop a new computational method that is able to efficiently quantify our modified notion of saliency.

5.1 Introduction

Visual saliency is a broad term used to describe some notion of importance in visual data. In the vision community, this concept has been studied for a variety of purposes, including attentional modeling in biological vision and the development of efficient systems for visual search. For example, the limited extent of the high-resolution fovea portion of the retina suggests that the human visual system may exhibit intelligent gaze control [91] in order to efficiently obtain meaningful information from its surroundings. This behavior has been characterized using *saliency maps* over the field of view [19], which are spatial representations that reflect the importance of image regions: brightness values corresponding to each image region quantify its importance with respect to others (see, e.g., Figure 2.1). When char-

acterizing the behavior of the human visual system, eye tracking can be used to generate saliency maps. However, the idea of visual saliency is also of great interest in the computer vision community, where the focus is on the development of computational algorithms that can automatically quantify visual saliency (see, e.g., [92]). The computed saliency maps are useful for a variety of applications such as automatic visual search, where they can help reduce the size of the search space by identifying regions of the visual field in which an object of interest is likely to be found [93]. Other applications include computer-assisted navigation of large panorama images [94], and robotic navigation [95].

In this chapter, we are concerned with one piece of a much larger goal. The larger goal is to use visual saliency to increase the efficiency of tasks in active vision, i.e., scenarios in which a system can control the geometric parameters of the camera (e.g., translation, rotation, zoom, etc.) [96]. Tasks in this field include automatic navigation, surveillance, and exploration. Of particular interest to us is understanding the role visual saliency might play in automatic scene exploration algorithms that control a stationary pan-tilt-zoom (PTZ) camera. That is, we would like to use saliency to determine appropriate camera manipulations such that informative images of the environment can be quickly collected.

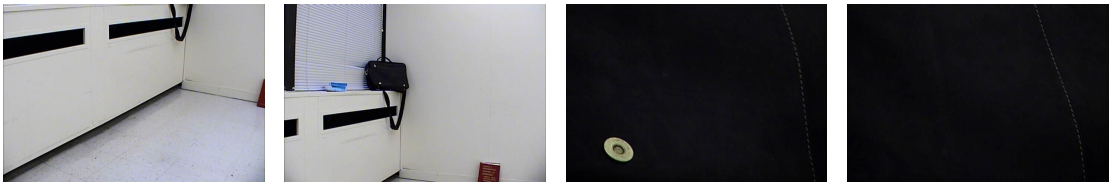
While the above discussion is concerned with the broad goal of using saliency in active vision, the focus of this chapter is on a narrower, more fundamental issue that arises in these scenarios. Active vision systems invariably collect multiple images of their surroundings. If we wish to use such data to quantify the saliency of the observed portions of the environment, we should do so in a manner that utilizes

the entire context that these images provide, and we must do so in an unambiguous fashion.

Toward this end, we suggest a notion of visual saliency that can be used when the data of interest consists of multiple images that are related by a known imaging process. Computing saliency for this type of data is a different problem than that considered by most existing works on computational visual saliency, which seek to transform *single* images into saliency maps that best agree with human annotation. We are instead concerned with extending the existing definition of visual saliency such that it is appropriate for the multi-image data described above. Once we have clearly defined what saliency means in this context, we will discuss several approaches that might be used to quantify it, including the one that we have developed called *ray saliency*.

5.1.1 Organization

This chapter is organized as follows. In Section 5.2, we formulate the problem of calculating visual saliency using multi-image datasets and define our extended notion of saliency. In Sections 5.3 and 5.4, we present the independent-processing- and mosaic-based approaches, respectively, to calculating visual saliency for PTZ imagery. In Section 5.5, we present our own technique for doing so. In Section 5.6, we present a more practical version of the method developed in Section 5.5, and we discuss experimental results for each technique described above. We summarize the chapter in Section 5.7.



(a) (b) (c) (d)

Figure 5.1: Overlapping imagery collected with a stationary PTZ camera: the ventilation ducts and handle of the dark messenger bag are visible in both (a) and (b). (c) and (d) were generated by zooming in on the main body of the bag.

5.2 Problem Formulation

In this chapter, we assume that we possess imagery collected using a stationary PTZ camera. Assuming we have a total of J images, we denote this data by $\mathcal{I} = \{(I_j, \mathbf{c}_j)\}_{j=1}^{J}$, where I_j denotes the j^{th} $m \times n$ image that was acquired using PTZ setting $\mathbf{c}_j \in \mathbb{R}^3$. An example of this type of imagery is given in Figure 5.1.

The task we set out to accomplish is that of using this imagery to quantify the visual saliency of the observed regions of the environment. Since each observed image pixel corresponds to a region of the environment, a solution to this problem can be represented in the form of J saliency maps, $\mathcal{S} = \{S_j\}_{j=1}^{J}$, where $S_j \in \mathbb{R}^{m \times n}$ quantifies the saliency of the regions visible in I_j. When $J = 1$, the problem is exactly that which is considered by the existing computational visual saliency literature: S_1 can be obtained using one of several existing techniques, and the saliency of an observed region is specified by the value assigned to the corresponding pixel.

However, when $J > 1$, there is an additional property that $\mathcal{S}$ should possess:

consistency. For example, suppose Figures 5.1b and 5.1c comprise an image set $\{I_1, I_2\}$. Due to overlap, both images contain pixels that correspond to the same observed region of the environment, namely, part of the dark messenger bag that has been placed on the windowsill. Therefore, the saliency values in S_1 and S_2 that correspond to this region should match: if they do not, it becomes ambiguous as to how to quantify the saliency of the bag. In order to avoid this ambiguity, we want to ensure that the individual maps are computed in such a way that these quantities agree, i.e., that they are consistent. More generally, for a given dataset $\mathcal{I}$, assume $\mathbf{x}_{i_1}$ and $\mathbf{x}_{i_2}$ denote the pixel locations in images j_1 and j_2, respectively, of a single region in the environment. Then the requirement for consistency can be explicitly stated as

$$S_{j_1}(\mathbf{x}_{i_1}) = S_{j_2}(\mathbf{x}_{i_2}) \ , \tag{5.1}$$

i.e., there should be no ambiguity in saliency due to the region appearing in more than one image. This condition does not mean that saliency values cannot change if more imagery becomes available (e.g., images containing previously-unobserved, salient details), just that the saliency values computed for a fixed set of images should exhibit consistency. Further, we understand that this is a theoretical and stringent condition that may not always be achievable using real images and practical techniques. However, we will see that even if (5.1) is met only in an approximate sense, the result is still more desirable than if it were not taken into consideration.

In this chapter, we shall discuss several techniques that might be used to generate a set of consistent saliency maps for multi-image data. We first examine

attempts at solving this problem that involve processing each image independently, and show why these techniques violate the consistency criterion. We then consider methods that apply existing techniques to pre-computed mosaics of the input images. While such methods produce consistent results, the necessity of mosaicing has several drawbacks. Finally, we present our approach, ray saliency, which is capable of producing a consistent set of saliency maps while mitigating the issues caused by mosaicing.

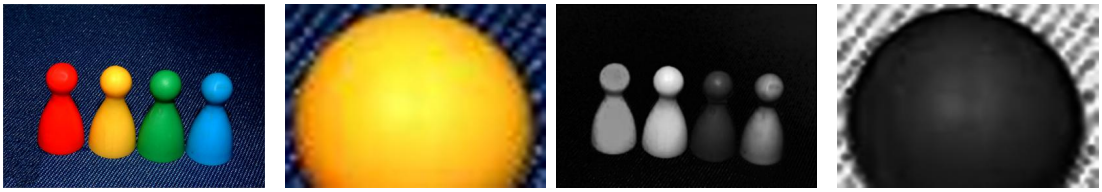
5.3 Independent Processing

In this section, we discuss methods that generate $\mathcal{S}$ by independently processing each image in $\mathcal{I}$. That is, if g denotes a mapping from images to saliency maps, then $S_{j_0} = g(I_{j_0})$ is independent of I_j for $j \neq j_0$. We will see that these approaches do not necessarily provide a solution that satisfies (5.1). Intuitively, this is because the saliency map computation for each image occurs without any knowledge of the contextual information provided by the others.

Several researchers in the computer vision community have proposed independently processing images with known geometric relationships in order to evaluate the saliency of regions of the surrounding environment. This work has mostly focused on using imagery collected by a stationary, robot-mounted, pan-tilt (fixed zoom) camera to assign saliency values to points on a sphere surrounding the camera, which they refer to as the *sensory ego-sphere*. Fleming *et al.* [97] collect these images using a set of pan-tilt settings that uniformly sample the sphere. The settings generate

overlapping images from which a set of independently-computed saliency maps is computed. Regions of the environment are represented using the vertices of a tessellated sphere, and the overlap in imagery causes single vertices to be associated with with sections of many different images. Therefore, each vertex is also associated with portions of many different saliency maps. The authors explicitly note that the different saliency maps can provide different values for the same vertex, i.e., that $\mathcal{S}$ is inconsistent. In order to resolve this ambiguity, they choose to compute the saliency value for each vertex using the *sum* of values given by each corresponding map. Ruesch et al. [98] extended this work to operate on a more faithful representation of the sphere, but also chose to resolve the visual saliency ambiguity via summation.

While the authors of these works generally report good results using this method, it is also the case that their camera systems do not collect images using widely-varying zoom parameters (i.e., focal lengths) or non-uniformly sampled points on the sphere. If such data is present, it is not clear how to handle the possible inconsistencies present in $\mathcal{S}$. Consider, for example, the synthetically-generated image set shown in Figure 5.2. If we define saliency using a context of large-enough spatial extent, the blue and black striped background should not be given a large saliency value. However, with the limited context provided in Figure 5.2b, this region is assigned a large saliency value in the corresponding, independently-generated map. Using the larger context provided by Figure 5.2a, the blue background is correctly assigned a low saliency value. Clearly, summing the two values will not result in the correct (low) saliency value for the blue region.



(a) (b) (c) (d)

Figure 5.2: Sample image (a) from the MSRA Salient Object Database [3] and a synthetically-generated, zoomed-in image of the same scene (b). The corresponding, independently-generated saliency maps are given by (c) and (d), respectively. The saliency maps were generated by the technique proposed in [2].

This issue is closely related to the concept of scale in bottom-up saliency, i.e., the spatial extent of the desired context. When considering only single images, one must define scale in units of pixels. However, this definition of context fundamentally depends on the imaging process since a single pixel represents more or less of the environment depending on the focal length and position of the camera at the time of acquisition. Thus, if multiple images of the same environment are available, defining the context using units of pixels can result in using larger or smaller regions of the environment depending on the specific image under consideration. That is, such a definition of scale introduces ambiguity regarding what the context is with respect to the surrounding environment. Therefore, we advocate that scale should not be defined in terms of pixels in these scenarios.

In our case, where multiple images are generated using a stationary PTZ camera, regions of the environment can be unambiguously associated with points on a sphere surrounding the camera, or *rays* from the camera focal point to the

projections of scene points on this sphere. Our method for calculating saliency for the observed regions of the environment uses the known geometric relationship between images to enable a concept of scale with respect to rays rather than pixels. Further, we simultaneously process the visual information provided by all images and are thus able to generate a consistent set of saliency maps without the need for heuristics (e.g., summation) to resolve ambiguity.

5.4 Mosaicing

Before describing our own approach for doing so, we shall first explore mosaic-based approaches to consistent saliency map generation. Such approaches must first generate an image mosaic, i.e., a single-image representation of all observed image data. We denote this representation as M, which can be computed using any one of a number of mosaicing methods, m, i.e., $M = m(\mathcal{I})$ (see, e.g., the work of [99] and [100]). Ideally, m acts in such a way that M appropriately reflects the known geometric relationship between images, and therefore each observed region of the environment corresponds to a single region within the mosaic. Once M has been formed, mosaicing approaches to saliency computation apply traditional techniques to generate a saliency map S_M, i.e., $S_M = g(M)$. S_M provides a single saliency value for each region of the environment, and a set of consistent saliency maps can be formed by interpolating over S_M, i.e., $\mathcal{S} = m^{-1}(S_M)$. This general approach is depicted in Figure 5.3. While mosaicing methods are able to generate a consistent set of saliency maps, they also suffer from several drawbacks.

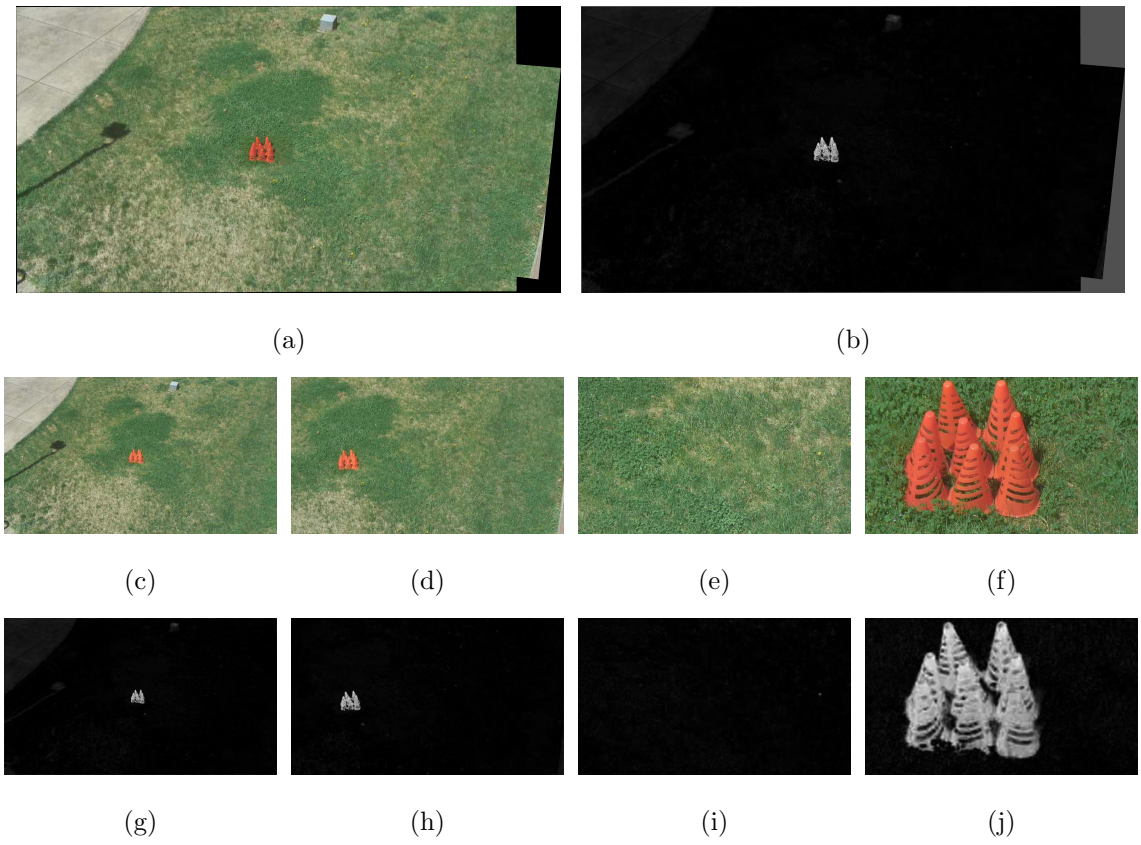


Figure 5.3: Graphical depiction of a mosaicing approach. The mosaic (a) was formed using acquired images (c), (d), (e), and (f). The saliency map (b) was computed using (a) and the approach developed by [2]. Finally, saliency maps (g), (h), (i), and (j) were generated by interpolating over (b) in the regions corresponding to each acquired image. In order to retain the detail provided by images (e) and (f), both (a) and (b) require approximately 26 times more pixels than observed.

In addition to their independent-processing approach, Fleming *et al.* [97] also explored using an image mosaic to calculate visual saliency over their sensory ego-sphere. It is suggested that such methods are computationally expensive since the entire mosaic must be recomputed each time a new image arrives. We further note that these approaches can require the creation of a large amount of extra data in order to represent each mosaic, especially when the images are acquired using widely-varying focal lengths. Consider, for example, the mosaic shown in Figure 5.3a: in order to represent all observed visual data without sacrificing the resolution of Figure 5.3f, it requires 26 times more pixels than were observed.

Bogdanova *et al.* [101, 102], who are concerned with computing visual saliency for images acquired using an omnidirectional camera, also note that traditional mosaicing approaches can cause distortion since it is impossible to accurately represent a sphere using a planar image. This will again lead to a spatially-ambiguous concept of scale: a fixed pixel neighborhood will correspond to more or less of the environment depending on how close it is to the optical center of the mosaic. To address this issue, Bogdanove *et al.* [101] project the observed imagery onto the surface of a sphere and use the spherical Fourier transform to compute visual saliency. While this is an appropriate space for representing the visual data, their method can suffer from the same interpolation issues as described above since it requires a regular angular sampling of the sphere.

5.5 Ray Saliency

We will now present our method for generating a set of consistent saliency maps for multi-image data collected by a stationary PTZ camera. We overcome the problems inherent to independent processing and mosaicing distortion by associating the pixel observations from each planar image with a coordinate in the spherical ray space. We use all such observations together to simultaneously quantify an adapted notion of bottom-up visual saliency. This approach allows us to resolve ambiguity problem described in Section 5.3. Further, our method does not require the generation of a mosaic, and therefore does not suffer from the representation shortcomings discussed in Section 5.4.

5.5.1 Pan-Tilt-Zoom Imaging Geometry

Understanding the way in which three-dimensional points are imaged by a PTZ camera is essential to our method. This projection from three to two dimensions can be succinctly described via a coordinate mapping that is usually specified by a *camera projection matrix*, $\mathbf{C}$ [103]. For a finite projective, stationary, PTZ camera with center at the origin of a specific fixed world coordinate system, this matrix may be expressed as $\mathbf{C} = \mathbf{C}(\theta, \phi, z) \in \mathbb{R}^{3\times3}$. $\mathbf{C}(\theta, \phi, z)$ is dependent upon the values of the camera's pan and tilt angles (θ and ϕ, respectively) and its zoom (z) [104]. The camera matrix can be decomposed as $\mathbf{C}(\theta, \phi, z) = \mathbf{K}(z)\mathbf{R}_X(\phi)\mathbf{R}_Y(\theta)$, where $\mathbf{R}_X(\phi)$ and $\mathbf{R}_Y(\theta)$ are three-dimensional rotation matrices about the X and Y axes (respectively) of the world coordinate system and $\mathbf{K}(z)$ is the camera calibration

(intrinsic) matrix for zoom z, i.e.,

$$\mathbf{K}(z) = \begin{bmatrix} f_x(z) & s(z) & p_x(z) \\ 0 & f_y(z) & p_y(z) \\ 0 & 0 & 1 \end{bmatrix} . \tag{5.2}$$

In (5.2), f_x and f_y represent the camera's focal length in units of the corresponding pixel dimensions, s represents the skew, and $[p_x \; p_y]^T$ specifies the pixel coordinate at which the optical axis intersects the image plane. While these parameters are invariant to extrinsic manipulations of the camera (e.g., panning and tilting), they do vary with respect to zoom. Given a set of collected images with sufficient overlap, auto-calibration can be used to infer the parameter values for each view under varying zoom settings (see Section 5.6.1 for details).

The camera images a three-dimensional point in the world coordinate system, $\tilde{\mathbf{X}} = [\tilde{X} \; \tilde{Y} \; \tilde{Z}]^T$, at a two-dimensional image plane coordinate $\mathbf{x} = [x \; y]^T$ given by

$$\mathbf{x} = \frac{\left[\left[\mathbf{C}(\theta, \phi, z)\tilde{\mathbf{X}}\right]_1 \left[\mathbf{C}(\theta, \phi, z)\tilde{\mathbf{X}}\right]_2\right]^T}{\left[\mathbf{C}(\theta, \phi, z)\tilde{\mathbf{X}}\right]_3} , \tag{5.3}$$

where $[\cdot]_i$ denotes the i^{th} component of the vector-valued argument. A close examination of (5.3) reveals that all three-dimensional points that lie on the same origin-inclusive ray project to the same point on the image plane. This represents the loss of depth information that accompanies the imaging process. In this work, we represent points belonging to the same (origin-inclusive) ray via a single *ray coordinate*, $\mathbf{X}$, which we define to be the point of intersection of the ray with the unit sphere (see Figure 5.4). The ray coordinate of any three-dimensional point $\tilde{\mathbf{X}}$ is given by $\tilde{\mathbf{X}}/\|\tilde{\mathbf{X}}\|_2$, and the ray coordinate corresponding to imaged location $\mathbf{x}$ can

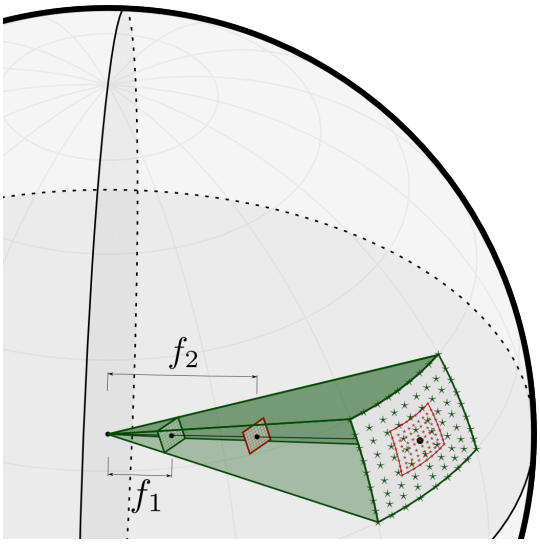


Figure 5.4: Geometry of the pan-tilt-zoom imaging process. Stars represent ray coordinates. The green and red rectangular grids represent the camera's pixel array for focal lengths f_1 and f_2, respectively. The projection of each image pixel, $\mathbf{x}$, on the surface of the sphere yields the corresponding ray coordinate, $\mathbf{X}$. With respect to the discussion in Section 5.5.1, $\tilde{\mathbf{X}}$ lies somewhere on the line that connects the camera center, image pixel, and corresponding ray coordinate. For the case depicted, the ray coordinates for the same pixel sensor zoomed to a larger focal length ($f_2 > f_1$) are much more tightly packed. If both images are collected, the overall set of observed rays is highly nonuniform over the sphere.

be calculated according to

$$\mathbf{X}(\mathbf{x}) = \frac{\mathbf{C}(\theta, \phi, z)^{-1}[\mathbf{x}^T \; 1]^T}{\|\mathbf{C}(\theta, \phi, z)^{-1}[\mathbf{x}^T \; 1]^T\|_2} \; . \tag{5.4}$$

5.5.2 Graph-Based Visual Saliency

The bottom-up saliency method we propose is most similar to the graph-based technique of [5]. Designed to operate on single images, their method quantifies the saliency of a specific pixel using its probability of occurrence under the stationary distribution of an image-dependent, discrete Markov chain. This Markov chain is related to a fully connected graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, that is formed over pixels. The edge weight between pixels i and i' is defined as

$$e_{i,i'} = s(\mathbf{x}_i, \mathbf{x}_{i'}) d(\mathbf{f}_i, \mathbf{f}_{i'}) \; , \tag{5.5}$$

where $\mathbf{x}_i$ and $\mathbf{f}_i$ correspond to the pixel coordinates and feature value associated with the i^{th} pixel, respectively, s produces a large value for arguments that are close in the spatial sense, and d yields a large value when its arguments are dissimilar. Using this construction, the saliency quantity assigned to each pixel (node) is taken to be its probability of occurrence under the stationary distribution, $\boldsymbol{\mu} \in \mathbb{R}^{|\mathcal{V}|}$, of the corresponding Markov chain.

While Harel *et al.* [5] propose to calculate this quantity by repeatedly applying the probability transition matrix to some initial distribution, we observe here that $\boldsymbol{\mu}$ can also be determined by simply summing the incident edge weight at each node [105]. That is, the probability of being at node i under the stationary distribution

is proportional to

$$[\boldsymbol{\mu}]_i \propto \sum_{i' \in \mathcal{V}} e_{i,i'} \ . \tag{5.6}$$

Instead of defining a graph over the image pixels and using the geometry of the image plane, we will form a graph over all observed rays and use modified versions of (5.5) and (5.6) to calculate saliency for observations in this new space.

5.5.3 Ray Space, Distance, and Scale

In order to appropriately incorporate the relative geometry of pixel observations that have been made using different PTZ settings, we quantify visual saliency by first projecting each observation into the ray space. We do so using (5.4), i.e., the k^{th} ray coordinate is given by

$$\mathbf{X}_k = \frac{\mathbf{C}_{j(k)}^{-1} \begin{bmatrix} \mathbf{x}_{i(k)}^T & 1 \end{bmatrix}^T}{\| \mathbf{C}_{j(k)}^{-1} \begin{bmatrix} \mathbf{x}_{i(k)}^T & 1 \end{bmatrix}^T \|_2} \ , \tag{5.7}$$

where $j(k)$ and $i(k)$ give the image and pixel indices (respectively) associated with the k^{th} ray, $\mathbf{C}_j$ denotes the camera matrix used to generate the j^{th} image, and $\mathbf{x}_i$ denotes the pixel coordinates for the i^{th} image pixel.

Since spatial similarity plays a key role in determining the context over which bottom-up saliency is defined, we require an appropriate way to define distance in this space. Observing that all ray coordinates lie on the surface of the unit sphere, this definition is given by the sphere geodesic (i.e., the great-circle distance). That is, the distance between ray coordinates $\mathbf{X}_k$ and $\mathbf{X}_{k'}$ is given by

$$q\left(\mathbf{X}_k, \mathbf{X}_{k'}\right) = \cos^{-1}(\mathbf{X}_k^T \mathbf{X}_{k'}) \ . \tag{5.8}$$

We can now explicitly define a notion of scale to be used in our saliency calculation. We propose to do so by specifying an angle, σ, that controls how strongly a ray is connected to its neighbors. More precisely, we evaluate the spatial similarity of ray coordinates with respect to scale σ using

$$s_\sigma\left(\mathbf{X}_k, \mathbf{X}_{k'}\right) = e^{-\frac{1}{2}\left(\frac{q\left(\mathbf{X}_k, \mathbf{X}_{k'}\right)}{\sigma}\right)^2} \quad . \tag{5.9}$$

Note that larger values of σ will induce less drastic changes in similarity as the angle between $\mathbf{X}_k$ and $\mathbf{X}_{k'}$ grows, which will ultimately yield saliency values that are calculated with respect to a larger context.

5.5.4 Computation of Ray Saliency

Equipped with an appropriate geometry, we can now turn our attention to assigning saliency values to pixel observations made over multiple images. As a space in which all observed data can be jointly considered, let $\mathcal{X} = \{(\mathbf{X}_k, \mathbf{f}_k)\}_{k=1}^K$ denote the set of ray coordinates and features associated with all observed pixels (i.e., if J N-pixel images have been observed, then $K = JN$). Inspired by the graph-based technique discussed in Section 5.5.2, we first form a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each node in $\mathcal{V}$ corresponds to a single observed ray. We define the edge weight between nodes k and k' as

$$e_{k,k'} = s_\sigma\left(\mathbf{X}_k, \mathbf{X}_{k'}\right) d\left(\mathbf{f}_k, \mathbf{f}_{k'}\right) \quad , \tag{5.10}$$

where σ is given in advance, s_σ is defined as in (5.9), and d is a non-negative function that yields larger values for features that are more dissimilar. Here, we concentrate

on CIELAB color features and therefore choose the simple Euclidean distance for d, i.e.,

$$d\left(\mathbf{f}_k, \mathbf{f}_{k'}\right) = \left\|\mathbf{f}_k - \mathbf{f}_{k'}\right\|_2 . \tag{5.11}$$

Depending on the specific PTZ settings used to acquire the set of images, the observed rays are not guaranteed to be uniformly distributed over the surface of the sphere (see Figure 5.4). Therefore, using the right-hand side of (5.6) alone to quantify saliency for nodes in this graph will artificially increase values assigned to nodes that appear in more densely-sampled regions (i.e., they have more neighbors). To compensate for this effect, we add a node-specific normalization constant that depends on the spatial structure of the graph, i.e., we use

$$[\boldsymbol{\mu}]_k = \frac{\sum_{k' \in \mathcal{V}} e_{k,k'}}{\sum_{k'' \in \mathcal{V}} s_\sigma(\mathbf{X}_k, \mathbf{X}_{k''})} \tag{5.12}$$

to define the saliency at node k. This is similar to the normalization that is done in bilateral filtering [106].

Given the graph specified above, we propose to quantify the saliency of each ray according to (5.12). We call this quantity *ray saliency*, and summarize in Algorithm 4 the procedure by which it can be used to generate a set of saliency maps.

5.5.5 Consistency

For this work, we assume that the scene under observation is static and that if the same ray is observed in two separate images, the associated features will be identical. Under these assumptions, we will show that the saliency maps generated by the ray saliency algorithm satisfy the consistency criterion.

Algorithm 4: Ray Saliency

Require: Dataset $\mathcal{I}$, scale parameter σ

1: Determine camera matrices $\{\mathbf{C}_j\}$ using the PTZ settings specified by $\mathcal{I}$

2: Determine $\mathcal{X}$ using (5.7)

3: Form $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ using (5.10)

4: Calculate $\boldsymbol{\mu}$ using (5.12)

5: Form $\mathcal{S}$ according to $S_{j(k)}(\mathbf{x}_{i(k)}) = [\boldsymbol{\mu}]_k$

Assume that the same point in the environment was observed in images j_1 and j_2 at pixel coordinates $\mathbf{x}_{i_1}$ and $\mathbf{x}_{i_2}$, respectively. Since both observations are of the same point in the environment, (5.7) will yield identical ray coordinates for both, i.e., $\mathbf{X}_{k_1} = \mathbf{X}_{k_2}$, where k_1 and k_2 (respectively) are the associated ray indices. From the assumptions stated above, this implies that the observed features will also be identical, i.e., $\mathbf{f}_{k_1} = \mathbf{f}_{k_2}$. Therefore, irrespective of the rest of the graph, (5.12) dictates that $[\boldsymbol{\mu}]_{k_1} = [\boldsymbol{\mu}]_{k_2}$. Thus, when $\mathcal{S}$ is generated, we will have $S_{j_1}(\mathbf{x}_{j_1}) = S_{j_2}(\mathbf{x}_{j_2})$, i.e., the consistency criterion is satisfied.

5.6 Experiments

In this section, we discuss our implementation of ray saliency, including how we obtain the camera matrices and what practical approximations we make. In order to show that ray saliency quantifies bottom-up saliency in the usual sense, we compare it to classical algorithms on a standard, single-image dataset. Finally, we give comparitive results among independent processing, mosaicing, and ray saliency

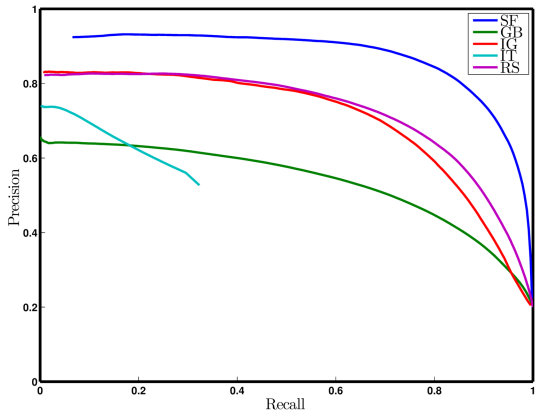


Figure 5.5: Ray saliency produces competitive results when compared to methods explicitly designed for single images. Depicted here is the precision-recall curve for saliency maps generated using the MSRA Salient Object Database [3] and the ground-truth data provided by [2]. `IT` refers to the method of [4], `BG` to [5], `IG` to [2], `SF` to [6], and `RS` to our method. See Section 5.6.3 for further discussion.

on real, multi-image datasets that we collected.

5.6.1 Automatic Camera Calibration

Our method relies on the availability of an accurate camera matrix for each image. We generate $\mathbf{C}_j$, the camera matrix for the j^{th} image, using the corresponding camera-reported PTZ setting, $[\theta_j \ \phi_j \ z_j]^T$, according to

$$\mathbf{C}_j = \mathbf{K}(z_j)\mathbf{R}_X(\phi_j)\mathbf{R}_Y(\theta_j) \quad , \tag{5.13}$$

where $\mathbf{R}_X(\phi_j)$ and $\mathbf{R}_Y(\theta_j)$ are rotation matrices about the optical X and Y axes by angles ϕ_j and θ_j, respectively, and $\mathbf{K}(z_j)$ is given by (5.2). Since the rotations are applied sequentially, the order of applying $\mathbf{R}_Y$ (pan) before $\mathbf{R}_X$ (tilt) is especially

important for our camera systems: the pan angles are reported with respect to a fixed Y axis that corresponds to its optical counterpart only when the camera uses a tilt value of zero, i.e., the axis normal to the physical base of the camera.

In order to generate $\mathbf{K}(z_j)$, the mappings between the zoom value and each intrinsic parameter, $f_x(z_j)$, $f_y(z_j)$, $s(z_j)$, $p_x(z_j)$, and $p_y(z_j)$, must be specified. We learn these functions using an automatic camera calibration method, i.e., one that does not require a calibration object such as a checkered board. Specifically, we collect a set of images using PTZ settings that adequately sample the set of allowable zoom values and ensure that each image overlaps with several others.

Broadly, we then use the technique described by [107] to determine a set of intrinsic parameters that corresponded to each used zoom value. To implement this method, we first automatically match SIFT features [108] and use the criterion from [100] to determine the image correspondence structure. Next, we compute homographies for pairs of corresponding images using the direct linear transform and RANSAC method described by [103] (we used [109]'s implementation) followed by a nonlinear refinement. We then compute linear estimates of the calibration matrices using the image-of-the-absolute-conic constraint described by [107]. From these estimates, we initialize a set of intrinsic parameters corresponding to each used zoom value, and then use nonlinear bundle adjustment [110] to refine them. This involves the joint optimization of the intrinsic values, the pan and tilt angles for every image, and the coordinates of the rays corresponding to feature matches. In the end, we obtain a coarse sampling of each zoom-to-intrinsic-parameter mapping. Using this calibration information, we linearly interpolate to determine intrinsic

parameter values for arbitrary zoom values.

5.6.2 Practical Implementation

Unfortunately, for even a small number of reasonably-sized images, Algorithm 4 is computationally intractable: once $\mathcal{X}$ has been determined, it exhibits $O(K^2)$ computational complexity, where K is the total number of pixels in all images. Therefore, we make two approximations in order to alleviate this complexity. First, we reduce the number of nodes in the graph by grouping rays through a superpixel clustering technique. Second, we enforce a locality constraint on the graph to reduce the number of edge weights that must be computed.

5.6.2.1 Superpixel Clustering

To reduce the number of graph nodes, we first extract intra-image superpixels (see, e.g., [111–114]) and use them to cluster rays from the same image. We then assign a single representative ray coordinate and feature to each superpixel and calculate a per-superpixel saliency value using the ray saliency method.

We specifically employ the SLIC superpixel algorithm developed by [115], which exhibits linear computational complexity. In order to ensure that superpixels extracted from images acquired using different focal lengths occupy roughly the same area on the surface of the sphere, we select the SLIC initial width parameter, w, on a per-image basis. Specifically, we calculate this parameter according to

$$w_j = 2 f_j \tan\left(\sigma_{\text{sp}}\right) \ , \tag{5.14}$$

where f_j is the camera focal length used to acquire the j^{th} image and σ_{sp} controls the approximate angular radius of each superpixel. σ_{sp} should be chosen with respect to the angular scale, σ, discussed in Section 5.5. For our experiments we used $\sigma_{\text{sp}} = \sigma/15$.

Let $\mathcal{P} = \{\mathcal{P}_l\}_{l=1}^{L}$ denote the set of all superpixels extracted from the available images, where $\mathcal{P}_l$ denotes the set of image pixel indices associated with the l^{th} superpixel. We assign a single, representative ray coordinate and feature pair to each superpixel using the mean in each space to form $\mathcal{X}_{\text{sp}} = \{(\mathbf{X}_l, \mathbf{f}_l)\}_{l=1}^{L}$, where

$$\begin{aligned} \mathbf{X}_l &= \mathbf{X}_{j_{\text{sp}}(l)} \left(\frac{1}{|\mathcal{P}_l|} \sum_{i \in \mathcal{P}_l} \mathbf{x}_i \right) , \\ \mathbf{f}_l &= \frac{1}{|\mathcal{P}_l|} \sum_{i \in \mathcal{P}_l} \mathbf{f}_i , \end{aligned} \tag{5.15}$$

and $j_{\text{sp}}(l)$ gives the image index associated with the l^{th} superpixel, $\mathbf{X}_j(\cdot)$ maps pixel coordinates in the j^{th} image to ray coordinates according to (5.4), and $\mathbf{x}_i$ and $\mathbf{f}_i$ are the pixel coordinate and feature (respectively) of the i^{th} image pixel.

We will use $\mathcal{X}_{\text{sp}}$ to calculate saliency values for each superpixel using the technique described below.

5.6.2.2 Locality Approximation

Even for small numbers of reasonably-sized images, it is computationally infeasible to compute the edge weights (5.10) for each pair of rays. To alleviate this issue, we approximate the edge weight function by enforcing a locality constraint that decreases the number edge connections per ray.

Noting that the s specified by (5.9) dictates that edge weights become more

insignificant the further apart two rays are with respect to σ, we approximate it by removing connections between rays that are sufficiently far apart, i.e.,

$$\hat{s}_\sigma(\mathbf{X}_l, \mathbf{X}_{l'}) = \begin{cases} e^{-\frac{1}{2}\left(\frac{q\left(\mathbf{x}_l, \mathbf{x}_{l'}\right)}{\sigma}\right)^2} & , \; l' \in \mathcal{N}_\sigma\left(\mathbf{X}_l\right) \\ 0 & , \text{ otherwise} \end{cases} . \tag{5.16}$$

Above, $\mathcal{N}_\sigma\left(\mathbf{X}_l\right)$ denotes the set of all rays in a certain local neighborhood of $\mathbf{X}_l$. Since determining $\mathcal{N}_\sigma$ for each ray can be computationally intensive, we use a k-d tree [116] which exhibits a worst-case region search complexity of $O(3L^{\frac{2}{3}})$ [117] for three-dimensional rays. In order to use this efficient data structure, we approximate the sphere geodesic with the Euclidean distance when calculating the local neighborhood, i.e., we define $\mathcal{N}_\sigma\left(\mathbf{X}_l\right)$ as

$$\mathcal{N}_\sigma\left(\mathbf{X}_l\right) = \left\{ l' \;\middle|\; \|\mathbf{X}_l - \mathbf{X}_{l'}\|_2 \leq 2\sin\left(\frac{c\sigma}{2}\right) \right\} , \tag{5.17}$$

where the right-hand side of the inequality is chosen to ensure that every l' for which $q\left(\mathbf{X}_l, \mathbf{X}_{l'}\right) \leq c\sigma$ is included in $\mathcal{N}_\sigma(\mathbf{X}_l)$. For our experiments, we used $c = 3$.

Using (5.16) and (5.17), we define the approximate edge weighting function as

$$\hat{e}_{l,l'} = \hat{s}_\sigma\left(\mathbf{X}_l, \mathbf{X}_{l'}\right) \; d\left(\mathbf{f}_l, \mathbf{f}_{l'}\right) . \tag{5.18}$$

5.6.2.3 Approximate Ray Saliency

As in Section 5.5.4, we will use the approximate edge weight (5.18) to compute saliency values for each node. However, we must modify the node-specific normalization constant in (5.12) to account for the locality approximation introduced in the previous section. That is, the approximate saliency for node l is calculated

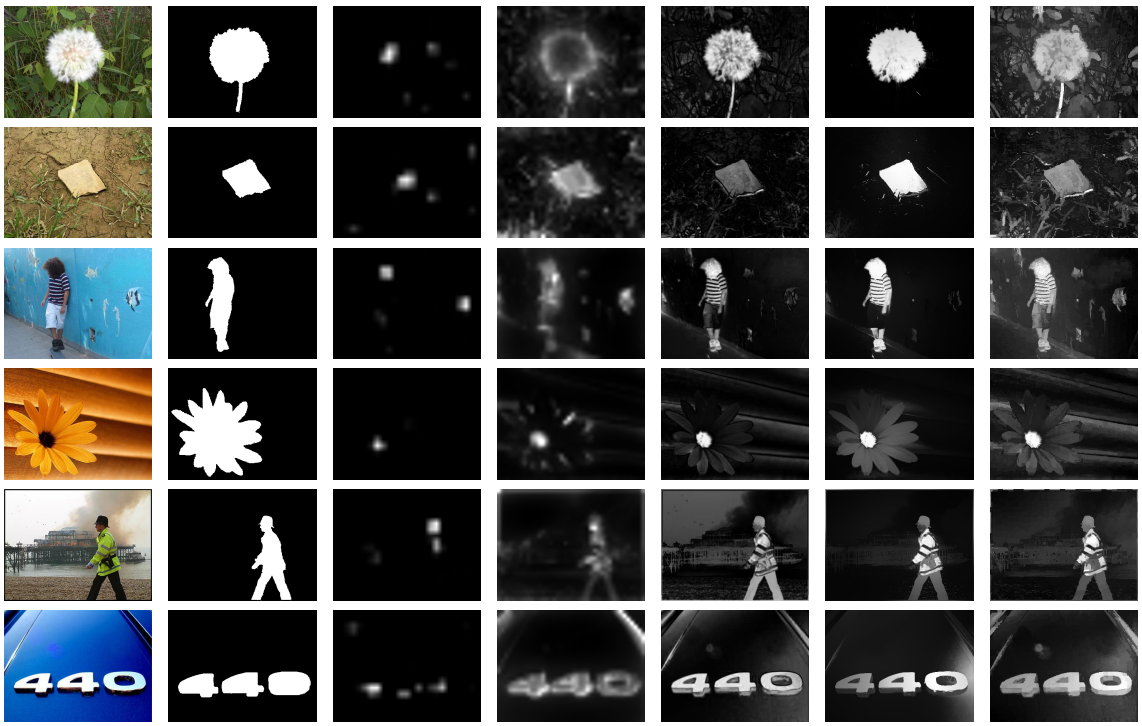


Figure 5.6: Ray saliency produces maps that appear similar to those generated by methods explicitly designed for single images. Depicted here are single-image saliency maps for a selected subset of the MSRA Salient Object Database [3]. The leftmost column shows the original images, and the second column shows the ground-truth saliency masks provided by [2]. The third through seventh columns show the saliency map results from [4], [5], [2], [6], and our method, respectively. See Section 5.6.3 for further discussion.

according to

$$[\hat{\boldsymbol{\mu}}]_l = \frac{\sum_{l' \in \mathcal{N}_\sigma(l)} \hat{e}_{l,l'}}{\sum_{l'' \in \mathcal{N}_\sigma(l)} \hat{s}_\sigma(\mathbf{X}_l, \mathbf{X}_{l''})}. \tag{5.19}$$

We summarize the modified procedure in Algorithm 5, where $S(\mathcal{P}_l) = [\hat{\boldsymbol{\mu}}]_l$ refers to assigning the saliency value given by $[\hat{\boldsymbol{\mu}}]_l$ to all pixels that are members of $\mathcal{P}_l$.

Algorithm 5: Approximate Ray Saliency

Require: Dataset $\mathcal{I}$, scale parameter σ, approximation parameters σ_{sp} and c

1: Determine camera matrices $\{\mathbf{C}_j\}$ using the PTZ settings specified by $\mathcal{I}$

2: Determine superpixels $\mathcal{P}$ across all images using the SLIC algorithm with image-specific width parameters w_j specified by (5.14)

3: Determine $\mathcal{X}_{\text{sp}}$ from (5.15)

4: Construct a k-d tree over the superpixels using the ray coordinates specified by $\mathcal{X}_{\text{sp}}$

5: Form $\hat{\mathcal{G}} = \left(\mathcal{V}, \hat{\mathcal{E}}\right)$ over superpixels using (5.18)

6: Calculate $\hat{\boldsymbol{\mu}}$ for superpixels using (5.19)

7: Form $\mathcal{S}$ according to $S_{j_{\text{sp}}(l)}(\mathcal{P}_l) = [\hat{\boldsymbol{\mu}}]_l$

5.6.2.4 Approximation Efficiency

Our method of approximate ray saliency leverages the two approximation steps to reduce the computational complexity of ray saliency. The superpixel grouping step reduces the number of nodes in the graph from K to $L \ll K$, and the locality approximation further reduces the amount of computation that must be done by

decreasing the number of edge weights that must be calculated. Assuming $|\mathcal{N}_\sigma| \ll L$, using the two together results in an approximate but practical algorithm that requires just $O(L)$ space and $O(K + L^{\frac{5}{3}})$ computation in the worst case, where the addition of K in the complexity term comes from the SLIC algorithm.

5.6.2.5 Approximation Consistency

Because of the superpixel grouping step, we note that the saliency maps generated by ray saliency will only be *approximately* consistent. This is because corresponding pixels may be grouped into superpixels that cover a slightly different regions of the environment. We will see that the results are not significantly altered by this effect.

5.6.3 Single-Image Data

In order to demonstrate that the quantities computed by ray saliency agree with classical notions of bottom-up visual saliency, we used our method to generate saliency maps for single images. These maps allow us to compare our technique with the current techniques that are not designed for multi-image data. Specifically, we ran our algorithm on the MSRA Salient Object Database [3] with ground truth data from [2] using $\phi = \theta = 0$, $f_x = f_y = 1e3$, $p_x = n/2$, and $p_y = m/2$ for the (unknown) camera parameters, and a value for σ that measured 35% of the smallest image dimension in pixels.

Our results for a selected subset of images are shown in Figure 5.6. We also

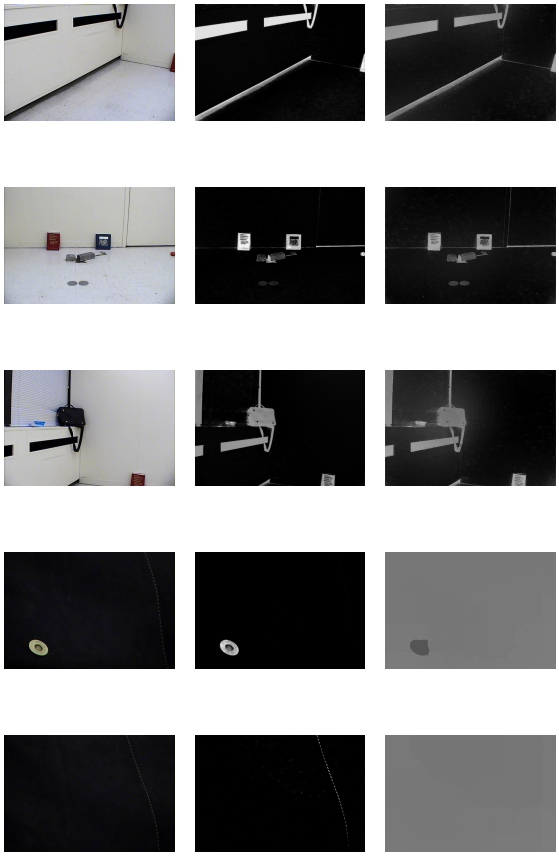


Figure 5.7: Ray saliency produces a consistent set of saliency maps where independent processing fails and mosaicing is not practical. Depicted here are the results of multi-image saliency processing for the `office` dataset. The leftmost column shows the acquired images, where the two bottom images are zoomed-in shots of the dark messenger bag on the windowsill. The second and third columns show the corresponding saliency maps generated using independent processing (using [2]'s method), and our method, respectively. Because of the wide variation in PTZ settings used to acquire these images, the mosaicing method failed due to lack of memory. See Section 5.6.4 for further discussion.

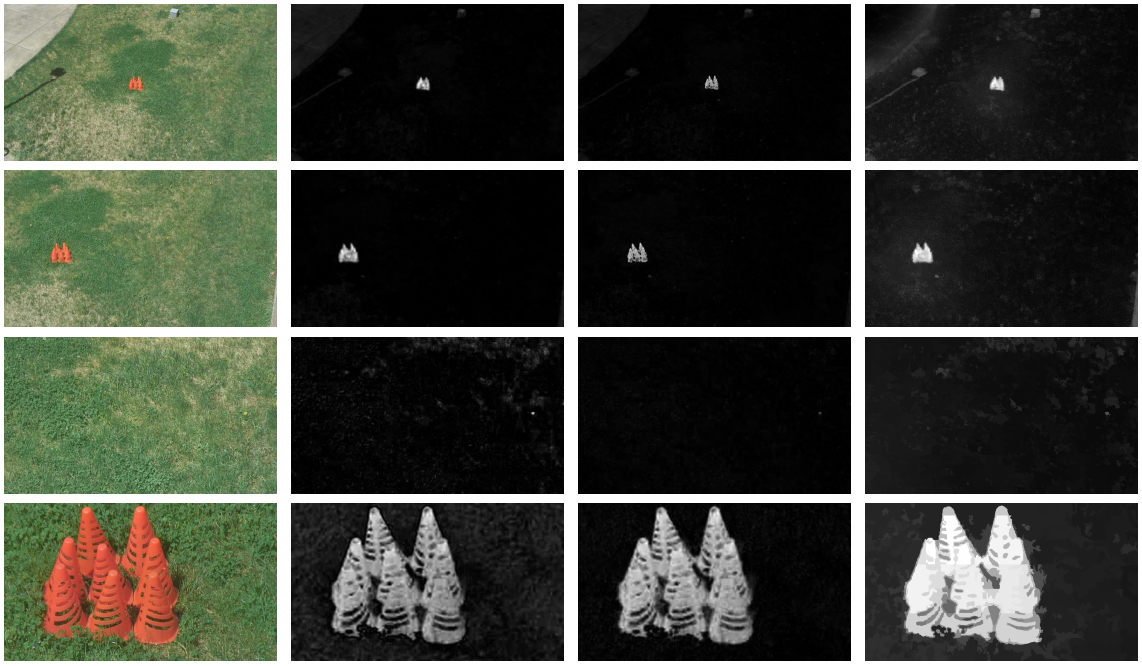


Figure 5.8: For certain datasets, all three methods (independent processing, mosaicing, and ray saliency) are able to produce approximately consistent results. Depicted here are the results of multi-image saliency processing for the `orangecones` dataset. The leftmost column shows the acquired images. The second through fourth columns show the corresponding saliency maps generated using independent processing (using [2]'s method), mosaicing (using [2]'s method), and our method, respectively. See Section 5.6.4 for further discussion.

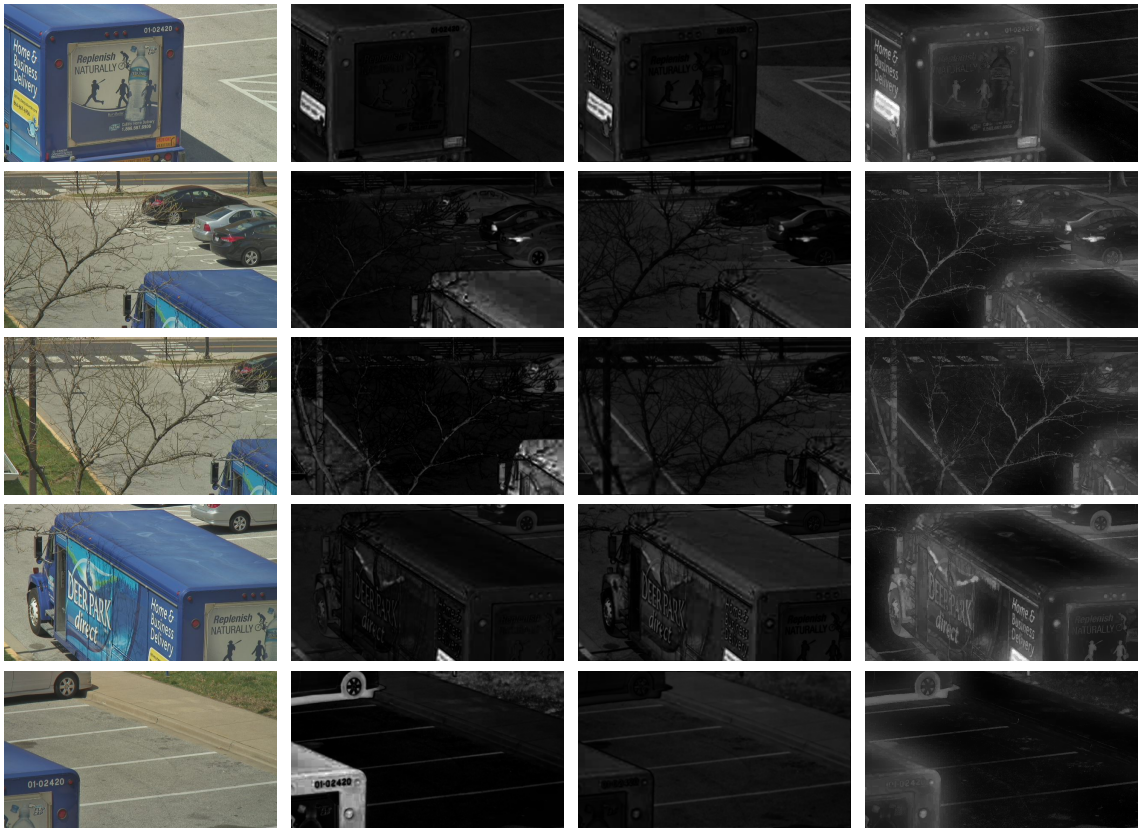


Figure 5.9: Ray saliency and mosaicing are able to produce a consistent set of saliency maps where independent processing fails. Depicted here are the results of multi-image saliency processing for the `watertruck` dataset. The leftmost column shows the acquired images. The second through fourth columns show the corresponding saliency maps generated using independent processing (using [2]'s method), mosaicing (using [2]'s method), and our method, respectively. See Section 5.6.4 for further discussion.

show the results of several popular saliency algorithms: the methods of [4], [5], [2], and [6]. We used the saliency maps generated by each method to produce the precision-recall curve shown in Figure 5.5. While ray saliency performs well here, it is important to note that to be such was not the goal of this work: we set out to design a method which is able to consistently process multi-image data, of which none of the other methods in this comparison is capable. Nevertheless, for single-image data, we see that ray saliency does produce results that agree with these other widely-accepted, bottom-up methods.

5.6.4 Multi-Image Data

We collected three multi-image datasets using two different PTZ cameras. Two of these datasets, `orangecones` and `watertruck`, were acquired using an outdoor-mounted Sony SNC-RH164 camera attached to our building on the campus of the University of Maryland, College Park. The third, `office`, was collected using an Axis 214 camera in a generic, empty office in which we placed unique objects. All datasets are composed of a set of images that result from changing the pan, tilt, and zoom settings of the sensor while viewing a static environment.

We processed images from each dataset using the three multi-image methods discussed in this chapter: independent processing, mosaicing, and (approximate) ray saliency. The results are shown in Figures 5.7, 5.8, and 5.9. For the independent processing and mosaicing approaches, we must also specify a single-image visual saliency method to use. Due to its combination of good performance and ease of

implementation, we chose to use the method of [2].

The saliency maps generated by the mosaicing and ray saliency methods are consistent, while those generated using independent processing are not necessarily so. Consider, for example, the last two rows in Figure 5.7, which represent zoomed-in views of the dark messenger bag on the windowsill. The saliency maps generated by the independent processing method are not consistent with the saliency map corresponding to the zoomed-out view of bag. However, the saliency maps generated by our method do agree with one another: the region of the environment corresponding to the bag is assigned a high saliency value in all saliency maps in which it appears. This phenomenon is also apparent when comparing the independently-processed saliency maps for the `watertruck` sequence: the blue body of the truck is given widely-varying saliency values depending on which independently-generated saliency map is used.

The absence of mosaicing results in Figure 5.7 hints at the primary shortcoming of using a mosaicing approach: the mosaic required to represent these images without loss of resolution required an amount of storage that exceeded the amount available in our experimental system with eight gigabytes of memory. As discussed in Section 5.4, mosaicing methods can require large amounts of space depending on the the specific pan-tilt-zoom settings used to acquire the images in the dataset. In fact, the storage requirements depend more on these settings than the total number of pixels observed, and can easily become prohibitively large. This effect was also apparent when using mosaicing to process the dataset shown in Figure 5.8: the mosaic required approximately 26 times more pixels than were observed. Approximate

ray saliency, on the other hand, does not require the generation of a mosaic and exhibits space complexity that is linear in the number of superpixels, L. Therefore, we were able to efficiently generate a set of consistent saliency maps for both of these datasets.

5.6.5 Comparison of Algorithm Complexity

In Table 5.1, we compare the worst-case storage and computational complexity of the three approaches discussed in this work: independent processing (independent), mosaicing, and ray saliency (approximate RS). We use R to denote the number of pixels required for the mosaic. Since R is a complicated function of K and the pan-tilt-zoom settings, we simply state that $R \gg K$ in the worst case, i.e., that it is prohibitively large. The complexities reported for the independent-processing and mosaicing approaches were determined using [2]'s method; using other approaches may change these values.

While independent processing requires just constant storage, there is also no guarantee that it will provide a set of consistent (or even approximately consistent) saliency maps. Further, the necessity of mosaic generation in approaches that use one can cause the storage requirements to become so large that they are no longer practical. Ray saliency, on the other hand, is able to provide a consistent set of saliency maps with practical storage and computational complexity.

Table 5.1: Worst-case algorithmic comparison for multi-image saliency approaches. The listed complexities for the independent-processing and mosaicing approaches were calculated assuming [2]'s method is used. K is the total number of observed pixels, R is the number of pixels necessary to represent the mosaic, and L is the total number of extracted superpixels.

Approach	**Storage**	**Time**
Independent	$O(1)$	$O(K)$
Mosaicing	$O(R \gg K)$	$O(R)$
Approximate RS	$O(L \ll K)$	$O(K + L^{5/3})$

5.7 Summary

In this chapter, we considered the problem of quantifying the bottom-up visual saliency of regions of an environment using multiple images acquired using a stationary PTZ camera. To resolve the ambiguities that this type of data can cause, we introduced the concept of consistency: that a given region of the environment should be assigned the same saliency value by all maps in which it appears. We then considered two existing approaches that could be used to generate saliency maps from multi-image data: independent processing and mosaicing. We showed that the independent-processing approaches can produce maps that violate the consistency criterion. Our investigation of mosaicing methods concluded that they are able to produce a consistent set of maps, but at the cost of geometric distortion and large space requirements.

As an alternative, we presented our solution to this problem: ray saliency. By associating pixel observations with points on the surface of a sphere and computing bottom-up saliency using a graph defined over those points, we are able to generate a set of consistent saliency maps in a way that does not require unnecessary interpolation while retaining the ability to accurately reflect the geometry of the imaging system. We discussed a practical implementation of this algorithm that leverages superpixel preprocessing and a graph approximation in order to achieve efficiency in both computation time and space. By using this multi-image algorithm on classical single-image datasets used in the visual saliency community, we showed that ray saliency measures a quantity that agrees with current definitions of bottom-up visual saliency. We then used multi-image datasets that we collected in order to present comparative results for our algorithm and the other multi-image processing techniques that were discussed.

Chapter 6: Summary and Directions for Future Research

While much of computer vision research is focused on processing data acquired using traditional cameras, there are several scenarios under which the sensor of interest may not be such a device. When this is the case, it is still often of interest to use ideas and techniques from classical computer vision to process the obtained data. When applicable, the extracted information can also be used to help control the sensor itself. This dissertation has focused on several problems for which it is impossible or undesirable to use a traditional camera. The work we have presented here has demonstrated modified computer vision processing techniques that are useful for several examples of unconventional visual data.

In Chapter 3, we adapted gradient-based reconstruction techniques for the specific problem of InSAR phase unwrapping, where the data arrives in the form of complex-valued radar measurements. We developed a sparse error-correction method in order to better accomplish this task. Specifically, we did so using a sparsity-regularized energy minimization technique that takes the form of a generalized lasso problem. Using this formulation, we were able to utilize the efficient ADMM algorithm to compute a solution.

In Chapter 4, we developed adaptive sensing strategies for a compressive sens-

ing camera. Since we are only concerned with the spatially-sparse foreground of the video, the compressive imaging device is sufficient for observation. Assuming that the measurement rate of the device is controllable, we developed an adaptive technique that adjusts the number of measurements collected in response to various forms of side information. Ultimately, our technique provides us the ability to, in an online fashion, adapt the amount of data collected in response to the complexity of the foreground signal under observation.

In Chapter 5, we extended the classical notion of visual saliency to multi-image data obtained using a stationary PTZ camera. We discovered that current saliency methods are not well-suited for this task, and we proposed a modified definition of visual saliency for this type of data. Using this definition, we developed an efficient technique that can be used to quantify visual saliency for multi-image data.

6.1 Directions for Future Research

We believe that each part of the above work has the potential for future avenues of research:

6.1.1 Phase Unwrapping

Currently, the phase-unwrapping technique presented in Chapter 3 contains no regularizing term for $\boldsymbol{\phi}$. However, several researchers in this field have reported improved results when including such a term in their formulation [37] [39]. Therefore, one interesting direction for future research would involve investigating the effect of

adding a regularization term, e.g. one based on the total-variation norm [118], to (3.11).

Another possible extension of this work involves trying to incorporate more effective techniques in gradient-based reconstruction, such as those proposed by Agrawal *et al.* [34]. We specifically think that the diffusion-tensor-based technique might prove effective. One way in which it might be incorporated is by modifying the ℓ_2-penalty term in (3.11), $\|\mathbf{G}\boldsymbol{\phi} - \mathcal{W}(\mathbf{G}\boldsymbol{\psi}) + \mathbf{e}\|_2^2$, such that an operator based on an edge-preserving diffusion tensor replaces $\mathbf{G}$ in $\mathbf{G}\boldsymbol{\phi}$. Of course, this sort of modification might affect the applicability of the ADMM algorithm, and therefore it remains to be seen if the idea is feasible.

6.1.2 Adaptive-Rate Compressive Sensing

The adaptive-sensing algorithm presented in Chapter 4 could serve as a starting point for several future extensions.

First, it may be possible to achieve even lower measurement rates. One way that this might be accomplished is by using a modified decoding procedure. In our current formulation, the decoder remains fixed even though the side sensors provide extra information that might be used to improve it. For example, the low-resolution sensor is able to provide both sparsity and support information, something that Vaswani *et al.* [62] have shown to lead to better estimates using fewer measurements. Another way in which the measurement rate might be lowered is by dynamically adjusting the measurement matrix beyond simply selecting the number of rows.

A strategy that generates the measurement vectors in an online fashion would be theoretically similar to the work of Duarte-Carvajalino [70] *et al.* and others [71] [72] [73] [74], but would need to incorporate our constraint of a measurement budget that changes between acquisitions.

Further, it may be possible to lift some of the assumptions regarding the ARCS-LRT sensor architecture. In particular, we believe that it might prove interesting to investigate allowing the position of the low-resolution traditional camera to vary from that of the high-resolution compressive one. This might involve using a more complicated mapping function instead of (4.24), i.e., one that also incorporates knowledge of the geometric relationship between the sensors.

6.1.3 Multi-Image Visual Saliency

The multi-image visual saliency method we presented in Chapter 5 also provides several interesting future research avenues.

One of these involves extending the method to handle dynamic scenes, i.e., compute the saliency values in an online fashion as the images are acquired. Such an approach would involve including the time of observation in the saliency calculation. It may also provide a more dynamic approach to scale selection. For example, if certain zoom levels are not used for an extended period of time, then it may be deemed unreasonable to compute the saliency for several corresponding scales.

Further, as was mentioned in the motivation for this problem, the investigation of camera-control strategies that respond to the computed saliency values is of great

interest. Specifically, we are interested in automatically selecting PTZ values such that we can quickly acquire high-resolution imagery of regions deemed to be salient.

Another direction for future research is that of modifying our technique such that camera translation is allowed. The sphere-based representation in our current formulation is only valid for a stationary sensor. In order to allow for translational motion, we imagine that we would need to obtain the three-dimensional structure of the scene with which we could associate the visual features. In such a scenario, we believe that the same graph-based formulation that drives our technique could be used to compute the visual saliency.

One might also investigate more efficient ways in which the saliency values might be computed, especially techniques that would allow us to avoid the super-pixel pre-processing step we use in our implementation. Moreover, recently-modified notions of saliency developed for single images (e.g., the work of [6]) might also be incorporated in our formulation. This might involve modifying our edge weight function or combining saliency values computed using multiple scales.

Finally, it may also be possible to extend our general formulation in order to develop new top-down saliency techniques. Specifically, one might investigate the effect that changing the zoom setting has on object detector confidence. This quantity could be used to define a new notion of saliency that corresponds to detector uncertainty, which might also be used to design camera-control strategies.

Bibliography

[1] C. Deledalle, L. Denis, and F. Tupin. NL-InSAR : Nonlocal Interferogram Estimation. *IEEE Transactions on Geoscience and Remote Sensing*, 49(4):1441–1452, 2011.

[2] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk. Frequency-tuned Salient Region Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[3] T. Liu, J. Sun, N. Zheng, X. Tang, and H. Shum. Learning to detect a salient object. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[4] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998.

[5] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In *Neural Information Processing Systems*, 2006.

[6] F. Perazzi, P. Krahenbuhl, Y. Pritch, and A. Hornung. Saliency Filters: Contrast Based Filtering for Salient Region Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[7] M. Everingham, S. Ali Eslami, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge - a Retrospective. *International Journal of Computer Vision*, 2014.

[8] Pets 2009 benchmark data. `http://www.cvg.rdg.ac.uk/PETS2009/a.html`.

[9] R. Baraniuk. More is less: signal processing and the data deluge. *Science*, 331(6018):717–719, February 2011.

[10] M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly, and R. Baraniuk. An Architecture for Compressive Imaging. In *Proceedings of the IEEE International Conference on Image Processing*, 2006.

[11] V. Cevher, A. Sankaranarayanan, M. Duarte, D. Reddy, R. Baraniuk, and R. Chellappa. Compressive sensing for background subtraction. In *European Conference on Computer Vision*, 2008.

[12] R. Baraniuk, M. Davenport, M. Duarte, and C. Hegde. *An introduction to compressive sensing*. Connexions, 2011.

[13] R. Willett, R. Marcia, and J. Nichols. Compressed sensing for practical optical imaging systems: a tutorial. *Optical Engineering*, 50(7), 2011.

[14] M. Duarte, M. Davenport, D. Takhar, J. Laska, K. Kelly, and R. Baraniuk. Single-Pixel Imaging via Compressive Sampling. *IEEE Signal Processing Magazine*, 25(2):83–91, March 2008.

[15] D. Gao, V. Mahadevan, and N. Vasconcelos. The discriminant center-surround hypothesis for bottom-up saliency. In *Neural Information Processing Systems*, 2007.

[16] N. Bruce and J. Tsotsos. Saliency based on information maximization. In *Neural Information Processing Systems*, 2005.

[17] X. Hou and L. Zhang. Saliency detection: A spectral residual approach. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2007.

[18] J. Li, M. Levine, X. An, X. Xu, and H. He. Visual Saliency Based on Scale-Space Analysis in the Frequency Domain. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(4):996–1010, 2013.

[19] C. Koch and S. Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. *Human Neurobiology*, 4:219–227, 1985.

[20] D. Gao, S. Han, and N. Vasconcelos. Discriminant saliency, the detection of suspicious coincidences, and applications to visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(6):989–1005, June 2009.

[21] A. Oliva, A. Torralba, M. Castelhano, and J. Henderson. Top-Down Control of Visual Attention in Object Detection. In *Proceedings of the IEEE International Conference on Image Processing*, 2003.

[22] V. Mahadevan and N. Vasconcelos. Saliency-based Discriminant Tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[23] A. Borji, D. Sihite, and L. Itti. Quantitative Analysis of Human-Model Agreement in Visual Saliency Modeling: A Comparative Study. *IEEE Transactions on Image Processing*, 22(1):55–69, 2013.

[24] S. Chavez, Q. Xiang, and L. An. Understanding Phase Maps in MRI: A New Cutline Phase Unwrapping Method. *IEEE Transactions on Medical Imaging*, 21(8):966–977, 2002.

[25] M. Jenkinson. Fast, Automated, N-dimensional Phase-Unwrapping Algorithm. *Magnetic Resonance in Medicine*, 49(1):193–197, January 2003.

[26] S. Pandit, N. Jordache, and G. Joshi. Data-dependent systems methodology for noise-insensitive phase unwrapping in laser interferometric surface characterization. *Journal of the Optical Society of America A*, 11(10):2584–2592, October 1994.

[27] J. Huntley and H. Saldner. Temporal phase-unwrapping algorithm for automated interferogram analysis. *Applied Optics*, 32(17):3047–3052, June 1993.

[28] R. Goldstein, H. Zebker, and C. Werner. Satellite radar interferometry: Two-dimensional phase unwrapping. *Radio Science*, 23(4):713–720, July 1988.

[29] D. Ghiglia and M. Pritt. *Two-Dimensional Phase Unwrapping*. John Wiley & Sons, New York, 1998.

[30] K Itoh. Analysis of the phase unwrapping algorithm. *Applied Optics*, 21(14):2470, July 1982.

[31] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

[32] R. Tibshirani. *The Solution Path of the Generalized Lasso*. PhD thesis, 2011.

[33] H. Hongxing, J. Bioucas-Dias, V. Katkovnik, and L. Wu. Interferometric Phase Image Estimation via Sparse Coding in the Complex Domain. 2013.

[34] A. Agrawal, R. Raskar, and R. Chellappa. What Is the Range of Surface Reconstructions from a Gradient Field? In *Proceedings of the European Conference on Computer Vision*, pages 578–591, 2006.

[35] S. Madsen, H. Zebker, and J. Martin. Topographic Mapping Using Radar Interferometry: Processing Techniques. *IEEE Transactions on Geoscience and Remote Sensing*, 31(1):246–256, 1993.

[36] B. Hunt. Matrix formulation of the reconstruction of phase values from phase differences. *Journal of the Optical Society of America*, 69(3):393–399, March 1979.

[37] J. Marroquin and M. Rivera. Quadratic regularization functionals for phase unwrapping. *Journal of the Optical Society of America A*, 12(11):2393, November 1995.

[38] L. Guerriero, G. Nico, G. Pasquariello, and S. Stramaglia. New regularization scheme for phase unwrapping. *Applied Optics*, 37(14):3053–8, May 1998.

[39] G. Nico, G. Palubinskas, and M. Datcu. Bayesian Approaches to Phase Unwrapping: Theoretical Study. *IEEE Transactions on Signal Processing*, 48(9):2545–2556, 2000.

[40] D. Ghiglia and L. Romero. Minimum Lp-norm two-dimensional phase unwrapping. *Journal of the Optical Society of America*, 13(10):1999–2013, 1999.

[41] D. Ghiglia and L. Romero. Robust two-dimensional weighted and unweighted phase unwrapping that uses fast transforms and iterative methods. *Journal of the Optical Society of America A*, 11(1):107, January 1994.

[42] J. Dias and J. Leitao. The ZpiM Algorithm: A Method for Interferometric Image Reconstruction in SAR/SAS. *IEEE Transactions on Image Processing*, 11(4):408–22, January 2002.

[43] J. Bioucas-Dias and G. Valadão. Phase Unwrapping via Graph Cuts. *IEEE Transactions on Image Processing*, 16(3):698–709, March 2007.

[44] D. Reddy, A. Agrawal, and R. Chellappa. Enforcing Integrability by Error Correction using l1-minimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[45] J. Dias, T. Silva, and J. Leitao. Absolute Phase Estimation with Discontinuities: A Stochastic Nonlinear Filtering Approach. In *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium*, 1998.

[46] V. Katkovnik, J. Astola, and K. Egiazarian. Phase Local Approximation (PhaseLa) Technique for Phase Unwrap From Noisy Data. *IEEE Transactions on Image Processing*, 17(6):833–846, 2008.

[47] E. Candès. Compressive sampling. In *International Congress of Mathematicians*, 2006.

[48] D. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, April 2006.

[49] E. Candes and T. Tao. Near-optimal signal recovery from random projections: universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, December 2006.

[50] E.J. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, February 2006.

[51] Richard G. Baraniuk. Compressive Sensing [Lecture Notes]. *IEEE Signal Processing Magazine*, 24(4):118 –121, 2007.

[52] J. Romberg. Imaging via Compressive Sampling. *IEEE Signal Processing Magazine*, 25(2):14–20, March 2008.

[53] M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly, and R. Baraniuk. Compressive imaging for video representation and coding. In *Proceedings of the Picture Coding Symposium*, 2006.

[54] J. Park and M. Wakin. A Multiscale Framework for Compressive Sensing of Video. In *Picture Coding Symposium*, 2009.

[55] A. Sankaranarayanan, C. Studer, and R. Baraniuk. CS-MUVI: Video compressive sensing for spatial-multiplexing cameras. In *Proceedings of the International Conference on Computational Photography*, 2012.

[56] D. Reddy, A. Veeraraghavan, and R. Chellappa. P2C2: Programmable Pixel Compressive Camera for High Speed Imaging. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[57] A. Sankaranarayanan, P. Turaga, R. Chellappa, and R. Baraniuk. Compressive acquisition of linear dynamical systems. *SIAM Journal on Imaging Sciences*, 6(4):2109–2133, 2013.

[58] M. Asif and J. Romberg. Sparse recovery of streaming signals using l1 homotopy. *arXiv*, 2013.

[59] D. Angelosante, J. Bazerque, and G. Giannakis. Online adaptive estimation of sparse signals: where RLS meets the l1 norm. *IEEE Transactions on Signal Processing*, 58(7):3436–3447, 2010.

[60] N. Vaswani. Kalman filtered compressed sensing. In *IEEE International Conference on Image Processing*, 2008.

[61] N. Vaswani and W. Lu. Modified-CS: Modifying Compressive Sensing for Problems With Partially Known Support. *IEEE Transactions on Signal Processing*, 58(9):4595–4607, September 2010.

[62] Namrata Vaswani. LS-CS-residual (LS-CS): compressive sensing on least squares residual. *IEEE Transactions on Signal Processing*, 58(8):4108–4120, August 2010.

[63] M. Cossalter, G. Valenzise, M. Tagliasacchi, and S. Tubaro. Joint Compressive Video Coding and Analysis. *IEEE Transactions on Multimedia*, 12(3):168–183, April 2010.

[64] V. Stankovic, L. Stankovic, and S. Cheng. Compressive image sampling with side information. In *Proceedings of the IEEE International Conference on Image Processing*, 2009.

[65] V. Stankovic, L. Stankovic, and S. Cheng. Sparse signal recovery with side information. In *Proceedings of the European Signal Processing Conference*, 2009.

[66] J. Scarlett, J. Evans, and S. Dey. Compressed sensing with prior information: information-theoretic limits and practical decoders. *IEEE Transactions on Signal Processing*, 61(2):427–439, 2013.

[67] D. Malioutov, S. Sanghavi, and A. Willsky. Sequential Compressed Sensing. *IEEE Selected Topics in Signal Processing*, 4(2):435–444, April 2010.

[68] P. Boufounos, M. Duarte, and R. Baraniuk. Sparse signal reconstruction from noisy compressive measurements using cross validation. In *IEEE Workshop on Statistical Signal Processing*, 2007.

[69] A. Ashok, P. Baheti, and M. Neifeld. Compressive imaging system design using task-specific information. *Applied Optics*, 47(25):4457–71, September 2008.

[70] J. Duarte-Carvajalino, G. Yu, L. Carin, and G. Sapiro. Task-driven adaptive statistical compressive sensing of gaussian mixture models. *IEEE Transactions on Signal Processing*, 61(3):585–600, 2012.

[71] A. Averbuch, S. Dekel, and S. Deutsch. Adaptive compressed image sensing using dictionaries. *SIAM Journal on Imaging Sciences*, 5(1):57–89, January 2012.

[72] Shihao Ji, Ya Xue, and Lawrence Carin. Bayesian Compressive Sensing. *IEEE Transactions on Signal Processing*, 56(6):2346–2356, January 2008.

[73] C. Chou, R. Rana, and W. Hu. Energy efficient information collection in wireless sensor networks using adaptive compressive sensing. In *Proceedings of the IEEE Conference on Local Computer Networks*, 2009.

[74] J. Haupt, R. Baraniuk, R. Castro, and R. Nowak. Sequentially designed compressed sensing. In *Proceedings of the IEEE Statistical Signal Processing Workshop*, 2012.

[75] X. Yuan, J. Yang, P. Llull, X. Liao, G. Sapiro, D. Brady, and L. Carin. Adaptive temporal compressive sensing for video. *arXiv*, 2013.

[76] H. Schaeffer, Y. Yang, and S. Osher. Real-time adaptive video compressive sensing. Technical report, UCLA CAM, 2013.

[77] X. Clady, F. Collange, F. Jurie, and P. Martinet. Object tracking with a pan-tilt-zoom camera: application to car driving assistance. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2001.

[78] A. Senior, A. Hampapur, and M. Lu. Acquiring multi-scale images by pan-tilt-zoom control and automatic multi-camera calibration. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, 2005.

[79] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, January 2008.

[80] D. Donoho and J. Tanner. Precise undersampling theorems. *Proceedings of the IEEE*, 98(6):913–924, June 2010.

[81] G. Warnell, D. Reddy, and R. Chellappa. Adaptive Rate Compressive Sensing for Background Subtraction. In *IEEE International Conference on Audio, Speech, and Signal Processing*, 2012.

[82] Rachel Ward. Compressed Sensing With Cross Validation. *IEEE Transactions on Information Theory*, 55(12):5773–5782, 2009.

[83] W. Johnson and J. Lindenstrauss. Extensions of Lipschitz maps into a Hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

[84] R. Kashyap. A Bayesian comparison of different classes of dynamic models using empirical data. *IEEE Transactions on Automatic Control*, 22(5):715–727, 1977.

[85] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[86] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

[87] E. van den Berg and M. P. Friedlander. Probing the Pareto frontier for basis pursuit solutions. *SIAM Journal on Scientific Computing*, 31(2):890–912, 2008.

[88] E. van den Berg and M. P. Friedlander. SPGL1: A solver for large-scale sparse reconstruction, June 2007. http://www.cs.ubc.ca/labs/scl/spgl1.

[89] S. Julier, J. Uhlmann, and H. Durrant-Whyte. A new approach for filtering nonlinear systems. In *Proceedings of the American Control Conference*, 1985.

[90] MATLAB. *version R2013a*. The MathWorks Inc., Natick, Massachusetts, 2013.

[91] D. Ballard. Animate vision. *Artificial Intelligence*, 48(1):57–86, February 1991.

[92] A. Borji, D. Sihite, and L. Itti. Salient Object Detection: A Benchmark. In *Proceedings of the European Conference on Computer Vision*, 2012.

[93] N. Bruce and J. Tsotsos. Saliency, attention, and visual search: An information theoretic approach. *Journal of Vision*, 9(3):1–24, 2009.

[94] C. Ip and A. Varshney. Saliency-Assisted Navigation of Very Large Landscape Images. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1737–46, December 2011.

[95] Christian Siagian and Laurent Itti. Biologically-Inspired Robotics Vision Monte-Carlo Localization in the Outdoor Environment. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[96] J. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active Vision. *International Journal of Computer Vision*, pages 333–356, 1988.

[97] K. Fleming, R. Peters, and R. Bodenheimer. Image mapping and visual attention on a sensory ego-sphere. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[98] J. Ruesch, M. Lopes, A. Bernardino, J. Hornstein, J. Santos-Victor, and R. Pfeifer. Multimodal saliency-based bottom-up attention a framework for the humanoid robot iCub. In *IEEE International Conference on Robotics and Automation*, 2008.

[99] R. Szeliski. Image Alignment and Stitching: A Tutorial. Technical Report MSR-TR-2004-92, Microsoft, 2006.

[100] M. Brown and D. Lowe. Automatic Panoramic Image Stitching using Invariant Features. *International Journal of Computer Vision*, 74(1):59–73, December 2007.

[101] I. Bogdanova, A. Bur, and H. Hugli. Visual Attention on the Sphere. *IEEE Transactions on Image Processing*, 17(11), November 2008.

[102] I. Bogdanova, A. Bur, H. Hügli, and P. Farine. Dynamic visual attention on the sphere. *Computer Vision and Image Understanding*, 114(1):100–110, January 2010.

[103] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2nd edition, 2003.

[104] R. Hartley. Self-calibration of stationary cameras. *International Journal of Computer Vision*, 22(1):5–23, 1997.

[105] T. Cover and J. Thomas. *Elements of Information Theory.* John Wiley & Sons, 2nd edition, 2006.

[106] C. Tomasi and R. Manduchi. Bilateral Filtering for Gray and Color Images. In *Proceedings of the IEEE International Conference on Computer Vision*, 1998.

[107] L. Agapito, E. Hayman, and I. Reid. Self-calibration of rotating and zooming cameras. *International Journal of Computer Vision*, 45(2):107–127, 2001.

[108] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004.

[109] P. Kovesi. MATLAB and Octave functions for computer vision and image processing, 2000. Available from: <http://www.csse.uwa.edu.au/∼pk/research/matlabfns/>.

[110] S. Agarwal and K. Mierle. *Ceres Solver: Tutorial & Reference*. Google Inc., 2014.

[111] X. Ren and J. Malik. Learning a Classification Model for Segmentation. In *Proceedings IEEE International Conference on Computer Vision*, 2003.

[112] P. Felzenszwalb and D. Huttenlocher. Efficient Graph-Based Image Segmentation. *International Journal of Computer Vision*, 59(2):167–181, September 2004.

[113] A. Levinshtein, A. Stere, K. Kutulakos, D. Fleet, S. Dickinson, and K. Siddiqi. TurboPixels: Fast Superpixels Using Geometric Flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2290–2297, December 2009.

[114] M. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa. Entropy-Rate Clustering: Cluster Analysis via Maximizing a Submodular Function Subject to a Matroid Constraint. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(1):99–112, January 2014.

[115] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. SLIC Superpixels. Technical Report 149300, EPFL, June 2010.

[116] J. Bentley. Multidimensional Binary Search Trees Use for Associative Searching. *Communications of the ACM*, 18(9):509–517, September 1975.

[117] D. Lee and C. Wong. Worst-Case Analysis for Region and Partial Region Searches in Multidimensional Binary Search Trees and Balanced Quad Trees. *Acta Informatica*, 9(1):23–29, 1977.

[118] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60:259–268, 1992.