Extracting Orientation and Scale from Smoothly Varying Textures with Application to Segmentation

by

Jason Chang

B.S., Electrical Engineering
University of Illinois at Urbana-Champaign, 2007

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2009

Author . . . .
Department of Electrical Engineering and Computer Science
September 4, 2009

Certified by . . . .
John W. Fisher III
Principal Research Scientist
Thesis Supervisor

Accepted by . . . .
Terry P. Orlando
Chairman, Department Committee on Graduate Students
Department of Electrical Engineering and Computer Science

Extracting Orientation and Scale from Smoothly Varying Textures with Application to Segmentation

by

Jason Chang

Submitted to the Department of Electrical Engineering and Computer Science
on September 4, 2009, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

The work in this thesis focuses on two main computer vision research topic: image segmentation and texture modeling. Information theoretic measures have been applied to image segmentation algorithms for the past decade. In previous work, common measures such as mutual information or J divergence have been used. Algorithms typically differ by the measure they use and the features they use to segment an image. When both the information measure and the features change, it is difficult to compare which algorithm actually performs better and for what reason. Though we do not provide a solution to this problem, we do compare and contrast three distances under two different measures.

This thesis considers two forms of information theoretic based image segmentation algorithms that have previously been considered. We denote them here as the *label method* and the *conditional method*. Gradient ascent velocities are derived for a general Ali-Silvey distance for both methods, and a unique bijective mapping is shown to exist between the two methods when the Ali-Silvey distance takes on a specific form. While the conditional method is more commonly considered, it is implicitly limited by a two-region segmentation by construction. Using the derived mapping, one can easily extend a binary segmentation algorithm based on the conditional method to a multi-region segmentation algorithm based on the label method. The importance of initializations and local extrema is also considered, and a method of multiple random initializations is shown to produce better results. Additionally, segmentation results and methods for comparing the utility of the different measures are presented.

This thesis also considers a novel texture model for representing textured regions with smooth variations in orientation and scale. By utilizing the steerable pyramid of Simoncelli and Freeman, the textured regions of natural images are decomposed into explicit local attributes of contrast, bias, scale, and orientation. Once found, smoothness in these attributes are imposed via estimation of Markov random fields. This combination allows for demonstrable improvements in common scene analysis applications including segmentation, reflectance and shading estimation, and estimation of the radiometric response function from a single grayscale image.

Thesis Supervisor: John W. Fisher III
Title: Principal Research Scientist

Acknowledgments

I would like to start by thanking John Fisher for his limitless ideas, unrelenting patience, and general enthusiasm in my research. It seems like every conversation I have had with John sparks new and interesting research directions. His ability to analyze problems and develop even more interesting questions to those problems is truly inspiring. I would also like to thank Alan Willsky for providing new insights and observations to my approach on addressing topics. His vast knowledge about all the research topics in SSG is astonishing. This thesis would not have been possible without John and Alan's guidance and support.

I would also like to thank my fellow graduate students and friends. In the computer vision group and SSG, there was always someone to turn to when I had a question. In particular, my office mates Andrew Mastin, Archana Venkataraman, and Michal Depa have always been there to answer questions about things I have long forgotten in subjects such as grammar, probability, and inference (even when everyone has their headphones on). Also, John Sun, who has taken almost every graduate class with me at MIT, has helped me develop a deeper understanding of class material and career paths. It is nice to have a friend that shares similar research interest, hobbies, and even money-saving techniques (thanks and repped!). To the rest of my friends at MIT, thank you for your support and the happy memories we have shared.

Most of all, I would like to thank my mother, father, sister Cindy, and girlfriend Julia for their love and support. I could not have made it through the past two years without you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The focus of this thesis is to provide methods for image segmentation using level set methods and to present a novel texture model that can be used in many common computer vision problems. Additionally, we combine these two to represent and segment textured images. In this chapter, we motivate the problem of image segmentation and texture modeling. We then identify why these problems are difficult, and how our work attempts to solve them.

1.1 Motivation

Both image segmentation and texture modeling are some of the most fundamental problems in computer vision. In this section, we will briefly motivate why these problems are important in the field of computer vision.

1.1.1 Image Segmentation

An image segmentation algorithm is a method used to partition an image into meaningful and distinct regions. For example, in a binary image segmentation, one is challenged to distinguish pixels in the foreground from background. In Figure 1-1, an example segmentation of a zebra done by an expert is shown. In general, images will contain more than just one object, and we

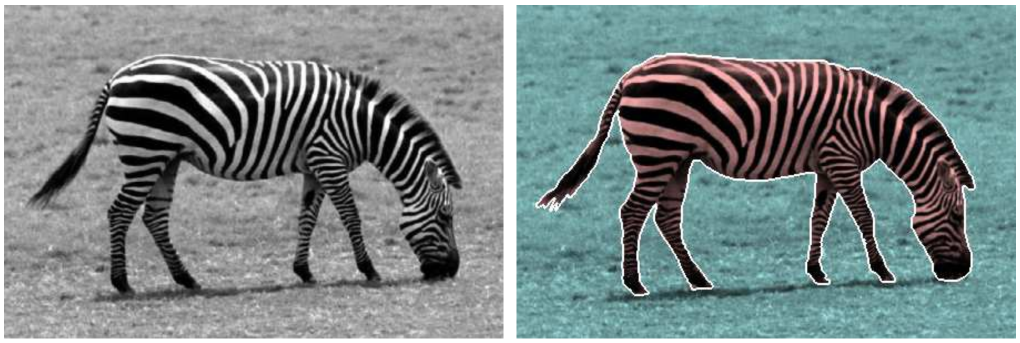

(a) Original Image (b) Segmented Image

Figure 1-1: An example segmentation done by hand

are then challenged with an m-ary segmentation problem where there are m separate regions to identify. Image segmentation is not to be confused with object recognition, where algorithms actually recognize that the scene in Figure 1-1 is a picture of a zebra. Rather, segmentation is the process of determining that two regions exist, one of which is the foreground (which happens to be the zebra) and one of which is the background (which happens to be the grass).

One might ask why image segmentation is such an important task. Segmentation is typically the first process in a computer vision system. In common applications such as object recognition and motion analysis, algorithms perform much better when the individual objects are first distinguished from each other. As a trivial example in object recognition, consider an application of trying to find and recognize a human. It is much easier to ask if the object outlined by a segmentation algorithm is a human rather than ask if there is a human in the entire image. In motion analysis, finding the optical flow field can be much simpler if objects are first found instead of dealing with entire scenes.

1.1.2 Texture Modeling

Prior to segmenting an image one must first choose what type of image model to use. In early works, pixel intensities were chosen as a very simple model. The well known Chan and Vese paper [9] implicitly modeled pixel intensities with a Gaussian distribution for purposes of segmentation. Though this method works well for very simple images, a complicated intensity distribution does not fit this model. Later, Kim et al. [26] modeled the pixel intensities nonparametrically using kernel density estimates. While this allowed for a more flexible appearance model at the pixel level, it still did not represent textures very well.

The underlying reason why these pixel-based approaches fail in textured images is because of an assumption that many of the algorithms make: the observed pixel intensities are statistically independent conditioned on their region label. While this assumption does not generally hold, it also does not greatly impact segmentation of non-textured images where it is common to model spatial correlations only in the region labels. However, such correlations cannot be discounted when segmenting textures. Furthermore, additional phenomenon in natural images (e.g. illumination) exhibit spatial dependencies which may also violate this assumption. This motivates incorporating a texture model in segmentation to aid in segmenting difficult images with strong spatial correlations.

Texture modeling is a widely studied problem in computer vision with many applications. For example, when using a generative model, one can replicate and extend textures in a random but visually appealing fashion (e.g. [38]). In a discriminative model, a robust texture representation can aid in segmentation to distinguish different objects (e.g. [21]). One common approach to texture modeling is to analyze the outputs of a filter bank, such as a set of Gabor filters [17] or the steerable pyramid [16]. The details of how we model textures is discussed in Chapter 4.

1.2 Previous Work

To someone unfamiliar of computer vision, image segmentation may seem like a trivial task. After all, the human visual system is quite skilled at segmenting a wide variety of images. One may ask why a seemingly simple task, one at which the human visual system seems so adept, is so

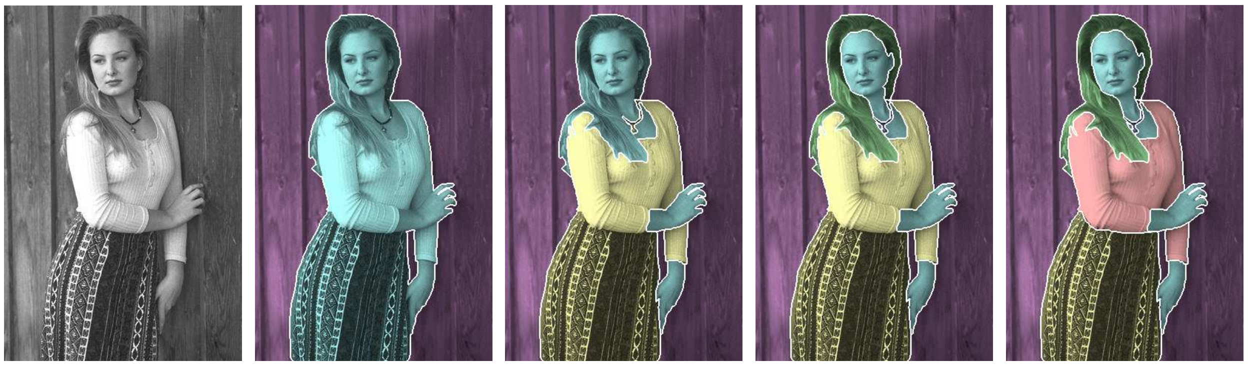

(a) Original Image (b) Two regions (c) Three regions (d) Five Regions (e) Six Regions

Figure 1-2: Example segmentations with different number of regions

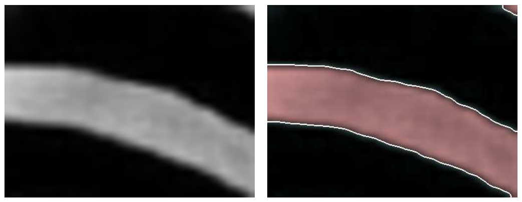

(a) Original Image (b) Original Image

Figure 1-3: An example segmentation of a section of a zebra

difficult to do using computational methods. Firstly, the task of image segmentation is an ill-posed problem [30]. If two people are asked to segment the image of the woman in Figure 1-2a, it is likely that their results will be different. In fact, any of the segmentations in Figure 1-2 can be a plausible solution. More detailed directions on whether or not clothes, hair, or jewelry should be separated are needed to decide which segmentation is better. Consider another example shown in Figure 1-3a. When presented with this image, one will typically segment the image into Figure 1-3b. However, this image is just a window of pixels from the zebra image in 1-1. This problem, sometimes referred to as the aperture problem in computer vision, deals with a limited field of view. Knowing when to segment the black stripes from the white stripes and when to combine them into a single object can be quite difficult. These examples illustrate the ill-posed nature of the image segmentation problem.

Despite this obstacle, image segmentation algorithms have been fairly successful. Algorithms over the past decade have proven to perform very well on a subset of natural images. They typically address the ill-posed nature of segmentation with one of two methods: supervised and unsupervised techniques. Supervised segmentation algorithms rely on training data from images segmented by experts in an attempt to gain some prior knowledge on how to segment images. Unsupervised algorithms are typically designed to model images (or objects within the image) based on what the designer thinks is best. Many recent unsupervised segmentation algorithms have attempted to maximize an information-theoretic distance to segment an image. For example, [21] uses J-Divergence and [26] uses mutual information. Though these types of algorithms perform well for a subset of simple images, we demonstrate in Chapter 5 that such methods do not perform well on a large set of textured images.

When using an unsupervised segmentation method on textured images, one needs to design a model to capture elements of the texture. There is extensive previous work on applying texture models to image segmentation. For example, [34] and [46] represent each texture with a constant measure within a region. In contrast to the method presented here, these methods either do not consider variations in scale and orientation or treat them as nuisance parameters. Other methods have been developed based on analysis of filter-bank responses. For example, [10] has looked at using wavelets and [12] utilizes Gabor filters. The work in [46] presents an approach that uses a set of Gabor filters, observing that changes in scale and orientation are manifested as shifts in their feature space. Montoya-Zegarra et al. [34] proposed a method using steerable pyramids [16] where they treat each filter output independently. One drawback of these approaches is the coarse discretization of scale and orientation.

1.3 Outline of the Thesis

This thesis is organized as follows. Chapter 2 begins with a brief discussion of background material and previous work essential in understanding the concepts presented in later chapters. Topics in this chapter include using level set methods for segmentation, nonparametric kernel density estimates, steerable pyramids, and Ali-Silvey distances.

Chapter 3 focuses on generalizing recent algorithms in image segmentation that use information-theoretic measures by extending the derivation to the broader class of Ali-Silvey distances. We show here that there exists an intimate relationship between two approaches of using an information theoretic measure, and additionally derive the resultant gradient ascent velocities for evolving a level set to maximize these distances. Using three information measures, we show a set of segmentation results and comment on how to compare algorithms.

Chapter 4 develops the novel texture model. Similar to previous work, we also utilize the steerable pyramid as a precursor to texture analysis. However, we exploit the property that responses at an arbitrary orientation can be efficiently interpolated with bounded error. We suggest an analysis of the pyramid response and demonstrate that it accurately measures the local scale and orientation of textures. We show that our feature set is able to classify thirteen textures from the commonly used Brodatz texture dataset [5], and that the algorithm can segment many synthetic images composed of two of these textures. Furthermore, we empirically show that because the space of possible segmentations contains many local extrema, performance gains are possible by using multiple different initializations for each image. After measuring the contrast, bias, scale, and orientation, we impose smoothness in these four features via Markov random fields to capture the spatial changes of the texture. As with most measures based on filter outputs, boundary effects can greatly affect an observed model. We address this issue and show reasonable performance gains using our methods.

Chapter 5 shows results of using our texture model in a few common computer vision applications. We begin by presenting a set of segmentation results using our method and compare to the methods of [26] and [21]. As a consequence of imposed smoothness in our feature set, we obtain a method for estimating a simple model of the nonlinear intensity response of a camera from a single grayscale image. We compare to the method of [14] showing significantly better performance. Through this analysis, we are able to obtain an accurate estimate of the irradiance image that when combined with the aforementioned smoothness assumptions, enables estimation of shading and reflectance for textured objects. We empirically validate our shading estimates by inferring the shape of the object using the algorithm of [44].

Finally, we conclude the thesis in Chapter 6 with possible changes to our development and future research directions. Appendices A and B provide details for the derivations of gradient ascent velocities and nonparametric Markov random field estimation.

Chapter 2

Background Material

In this chapter, we will briefly discuss some background material for readers that are not familiar with the topics. We cover the basics of using level set methods for image segmentation, nonparametric kernel density estimation, texture representation using the steerable pyramid, and Ali-Silvey distances.

2.1 Level Set Methods

Level set methods provide a way to implicitly represent and evolve an N-dimensional (or less) hyper-surface in an N-dimensional space. The works of Osher and Fedkiw [36] and Sethian [40] provide the original development of level set methods and a wealth of knowledge on this subject. When applied to image segmentation, a scalar function, $\underline{\varphi}$, is defined by values on a two-dimensional Cartesian grid. In practice, this function is stored as an image, and the height of the level set function is defined for each pixel in the image. For this reason, it will be convenient to refer to the height of the level set at a specific point as the value of the level set at a particular pixel, or $\varphi(x, y)$. Oftentimes, it will be more convenient to reference a pixel location by a single variable i instead of (x, y), where i references some pixel location. We will use the notation φ_i where the subscript references the pixel.

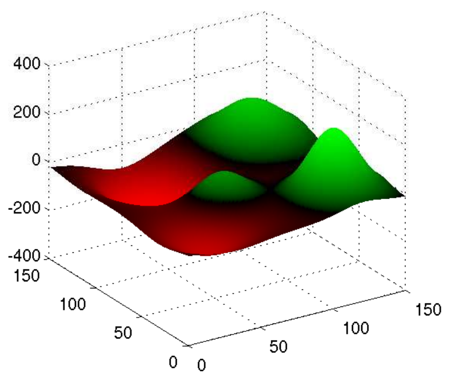
The implicit hyper-surface as it pertains to image segmentation is just a curve that exists in the two-dimensional support of the image. Any level set (the intersection of the surface with a constant height plane) of $\underline{\varphi}$ can be used as the implicit hyper-surface, but the zero level set is typically chosen for the representation. Throughout this thesis, the terms zero level set and curve will be used interchangeably to mean the same thing. The implied curve, $\mathcal{C}$, is defined as the set of all pixels on the three-dimensional level set function that have height zero:

$$\mathcal{C} = \{i \mid \varphi_i = 0\}. \tag{2.1}$$

The zero level set divides the image into two regions, R^+ and R^-, which consist of the positive and negative values of the level set function respectively:

$$R^{\pm} = \{i \mid \varphi_i \gtrless 0\}. \tag{2.2}$$

One can often think of a level set function as a terrain, where the region R^+ contains the land


(a) Plotted in 3 Dimensions

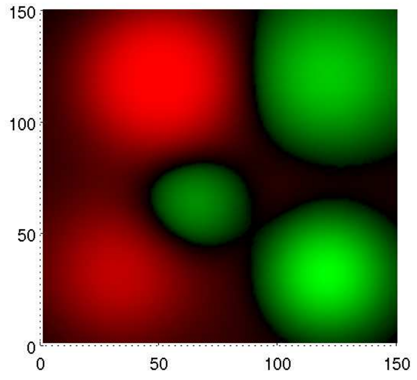

(b) Viewed as an Image

Figure 2-1: Level Set Function Viewed as Terrain

above sea level and R^- contains the land below sea level. An example of a level set function is given in Figure 2-1, where the green pixels belong to R^+, the red pixels belong to R^-, and the black pixels belong to the zero level set.

Level set methods involve representing a hyper-surface in a higher dimension, typically leading to increased memory and computation. However, the utility of representing a curve with level set methods is that the curve is implicitly represented. Creating or removing a new region is a matter of perturbing the underlying surface. If an explicit representation (e.g. snakes [24]) is used, it requires the user to maintain the explicit set of points on each curve. Creating or removing regions with an explicit representation requires bookkeeping and suffers from what is known as reparameterization of the curve (i.e. resampling points on the curve as it changes shape). In fact, the overhead of representing the entire underlying surface with an implicit representation typically outweighs the nuisance of an explicit representation.

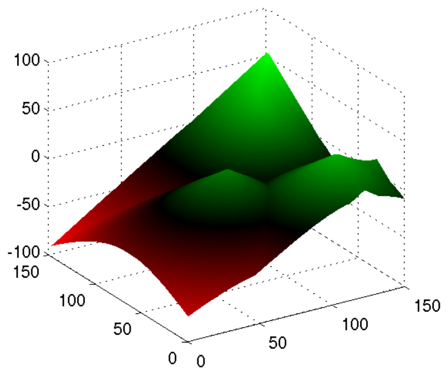
2.1.1 Signed Distance Function

In image segmentation algorithms using level set methods, the user is only concerned with the zero level set because it is the curve that segments the image. Consequently, this restricts pixels on the curve to have zero height, but pixels away from the curve need only have the same sign. An infinite number of parameterizations of level set functions exist that have the same zero level set.

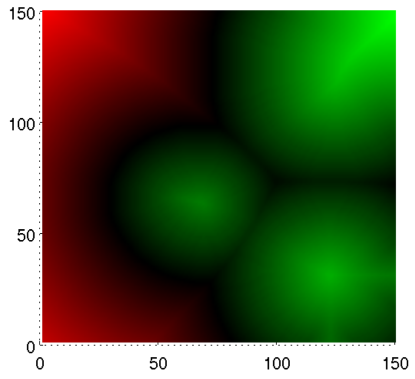
A very common approach is to make the level set function a signed distance function. A signed distance function has the property that, in addition to the value having the correct sign, the absolute value at each pixel is the minimum distance to the zero level set. The signed distance function looks more conical, and thus, a level set function using this property looks more like Figure 2-2 rather than Figure 2-1.

Though a signed distance function is not required for level set methods, it does include some nice properties. It provides for a very confident discrete derivative approximation near the region boundaries which allows for better numerical stability. Signed distanced functions also have the nice property that, for most pixels,

$$|\nabla \underline{\varphi}| = 1. \tag{2.3}$$



(a) Plotted in 3 Dimensions



(b) Viewed as an Image

Figure 2-2: Level Set Function as Signed Distance Function

The only location where this relation fails to hold is for pixels that are equidistant from the zero level set at more than one point. One can potentially use this property to simplify the level set evolution equations (discussed later), but it is generally not advised for stability issues. A more in-depth description of the formulation and benefits of using a signed distance function can be found in [36]. There are a few different methods for computing the signed distance function efficiently. In our implementation, the Fast Marching Method [48] was used.

2.1.2 Evolving the Level Set Function

The following is a typical process for curve evolution:

1. Initialize the zero level set with a random guess
2. Reinitialize the level set function to a signed distance function
3. Calculate a velocity $\underline{\dot{\varphi}}$ at every pixel in the level set
4. Update the level set for a small time interval according to $\underline{\dot{\varphi}}$
5. Repeat from Step 2 until convergence

In level set methods, an energy functional, E, is chosen for the particular application. The problem of calculating the evolving velocity field is then equivalent to maximizing this energy functional. Oftentimes, the energy functional will consist of multiple terms where one of the terms imposes some sort of smoothness constraint on the zero level set, making the ill-posed nature of image segmentation more well-posed.

In the algorithms covered in this thesis, the energy functional will consist of two terms. The first of these, E_{I}, is an energy functional that is dependent on the image statistics. The specifics of this term will be covered in later chapters. The other energy functional term, E_{S}, is a smoothing, regularization term that is only dependent on the actual shape of the curve and not the image. One commonly used regularization term penalizes longer curve lengths. Intuitively, a very jagged curve will have a longer curve length then a smooth curve. Therefore, our energy functional can be rewritten as

$$E(\mathcal{C}) = E_{\text{I}}(\mathcal{C}) - E_{\text{S}}(\mathcal{C}) = E_{\text{I}}(\mathcal{C}) - \alpha \oint_{\mathcal{C}} ds, \tag{2.4}$$

where α is a constant scalar that chooses how much weight to put on the regularization term.

Given a specific energy functional, one can maximize the term using gradient ascent to find the velocity field by which to evolve the level set. It is important to note that although the energy functional may depend on the entire image statistics, the optimal velocity is often only defined for pixels on the zero level set, $\mathcal{C}$. This will be discussed in greater detail in Section 2.1.3, but the consequence is that the energy functional only gives a valid velocity for pixels on the zero level set. Keeping this in mind, the proof in [19] showed that the gradient ascent velocity due to the smoothness constraint is just

$$\overrightarrow{V_{\text{S}}}(\ell) = \alpha\kappa_\ell \overrightarrow{N} \qquad \forall \ell \in \mathcal{C} \tag{2.5}$$

where κ_ℓ is the mean curvature at pixel ℓ given by

$$\kappa_\ell = \frac{\Delta\varphi_\ell}{|\nabla\varphi_\ell|} \tag{2.6}$$

As previously stated, if $\underline{\varphi}$ is a signed distance function, then the denominator of κ_ℓ simplifies to 1. However, for numerical stability, the norm of the gradient of the level set function is typically still calculated.

In the derivation of level set methods in [36], this velocity vector field must be applied to the level set function in the following form

$$\underline{\dot{\varphi}} + \overrightarrow{V} \cdot \nabla\underline{\varphi} = 0 \tag{2.7}$$

where $\underline{\dot{\varphi}}$ is the partial derivative of the level set function w.r.t. time indicating how to evolve the function. Following the steps in [36, p.42], the equation for the velocity update of the level set becomes

$$\underline{\dot{\varphi}} = V_n \left|\nabla\underline{\varphi}\right| \tag{2.8}$$

where $\overrightarrow{V} = V_n\overrightarrow{N} + V_t\overrightarrow{T}$. Note that we differ from the derivation in [36] in one way: our evolution, $\dot{\varphi}$ is equal to $V_n\left|\nabla\varphi\right|$, not $-V_n\left|\nabla\varphi\right|$. This is because we define the interior of our curve to be the positive values of φ, whereas [36] defines it to be the negative values of φ. The tangential velocities can be safely ignored because they reparameterize the level set function, but do not change the implicit definition of the curve. Thus, by combining Equations 2.5 and 2.8, the velocity due to the curve length penalty is

$$\underline{\dot{\varphi}}_{\text{S}} = \alpha\kappa\left|\nabla\underline{\varphi}\right|. \tag{2.9}$$

Given an energy functional, the updating velocity field due to the curve length penalty can now be calculated. We will discuss many different energy functionals and their resulting gradient ascent velocities in Chapter 3.

2.1.3 Velocity Off the Curve

It was noted in Equation 2.4 that gradient ascent on the energy functional only defines a velocity for the actual curve, $\mathcal{C}$. If we were somehow explicitly representing the curve, then this curve velocity would suffice. However, because we are implicitly defining the curve with a level set function, the evolution is not as straightforward. Perturbing an implicitly represented curve can not be achieved by changing only one value; the curve velocity must move a group of pixels in the level set representation to move the implied curve. There are two popular methods to extend the curve velocity obtained from gradient ascent to evolve the actual implicit level set representation: the smooth Heaviside function and velocity extension.

Smooth Heaviside Function

One method to extend the gradient ascent velocities to evolve the level set was developed by Chan and Vese in [9]. They define the level set function using the Heaviside function, which is a binary function that assigns labels to pixel regions.

$$H(\varphi_i) = \begin{cases} 1 & \text{if } \varphi_i \geq 0 \\ 0 & \text{if } \varphi_i < 0 \end{cases} \tag{2.10}$$

The derivative of the Heaviside function w.r.t. its argument is one on the curve, and zero elsewhere.

$$\delta_0(\varphi_i) = \frac{\partial}{\partial \varphi_i} H(\varphi_i) = \begin{cases} 1 & \text{if } \varphi_i = 0 \\ 0 & \text{else} \end{cases} \tag{2.11}$$

In the continuous case, with an explicit representation of the curve, one could use this delta function to describe the velocity on the curve. Chan and Vese use a smooth Heaviside function which results in smearing the curve velocity to a neighborhood around the curve.

We can derive this smearing more precisely. For any gradient ascent curve velocity, $\overrightarrow{V}$, we can express it with only the velocities of points on the curve:

$$\overrightarrow{V}_\ell = -f(\ell)\overrightarrow{N}_\ell \qquad \forall \ell \in \mathcal{C} \tag{2.12}$$

where $f(\cdot)$ is any function. With the previous definitions of the Heaviside function and its derivative, we can rewrite this velocity over all points in the image domain, Ω, instead of only on the curve.

$$\overrightarrow{V}_i = -f(i)\overrightarrow{N}_i \delta_0(\varphi_i) \qquad \forall i \in \Omega. \tag{2.13}$$

In the ideal Heaviside function, Equations 2.12 and 2.13 are exactly the same. However, when a smooth Heaviside function is used, the velocity in Equation 2.13 is smeared across the actual curve. A commonly used smooth Heaviside function [9] is:

$$H(\varphi_i) = \begin{cases} 1 & \text{if } \varphi_i \geq \epsilon \\ 0 & \text{if } \varphi_i < \epsilon \\ \frac{1}{2}\left[1 + \frac{\varphi_i}{\epsilon} + \frac{1}{\pi}\sin\left(\frac{\pi\varphi_i}{\epsilon}\right)\right] & \text{if } |\varphi_i| \leq \epsilon \end{cases} \tag{2.14}$$

with a corresponding derivative:

$$\delta_0(\varphi_i) = \begin{cases} \frac{1}{2\epsilon}\left[1 + \cos\left(\frac{\pi\varphi_i}{\epsilon}\right)\right] & \text{if } |\varphi_i| \leq \epsilon \\ 0 & \text{else} \end{cases} \tag{2.15}$$

where ϵ is some small constant (we chose to use $\epsilon = 1$).

When using a smooth Heaviside function, one can easily extend a binary segmentation into a multi-region segmentation using the method proposed in [6]. This method uses multiple level sets, where each level set represents exactly one region. By construction, the curves are not allowed to intersect or overlap. Thus, a segmentation using M level sets can represent up to M different regions.

Velocity Extension

An alternative method to evolve the level set from the curve velocity is called velocity extension. This method extends the velocity off of the curve such that the velocity at any point in the level set has the same value as the closest point on the curve. The method of velocity extension that was developed by Adalsteinsson and Sethian [1] constructs the extension velocities in such a way that the signed distance property of the level set function is preserved perfectly after each iteration. However, this method also requires that the velocity be defined for the value at the exact zero level set with sub-pixel accuracy (interpolated through pixels bordering the zero level set). This velocity is not always well defined for images if the pixels used to interpolate belong to separate objects.

When using velocity extension, the only currently proposed method to do multi-region segmentation is presented in [8]. This method allows for overlap amongst the level sets, and treats each level set as a bit in a region label. When using two level sets, a region can be inside or outside of both level sets, resulting in the labels: $\{00, 01, 10, 11\}$. This results in 2^M regions when using M level sets. When an image actually contains a power of two number of regions, this method can perform very well. If an image contained three regions, this method should allow for three regions by only using three of the four possible labels. However, in natural images, this method will favor using all possible labels because it can explain the image statistics better. This problem is equivalent to overfitting a model by allowing too many parameters. For this reason, we chose to implement the smooth Heaviside function instead of using velocity extension.

2.2 Nonparametric Kernel Density Estimation

Oftentimes, we will need to model the distribution of a random variable based on many observed occurrences. Density estimation can be broadly grouped into three categories: parametric (a fixed number of parameters), semi-parametric (the number of parameters grows sub-linearly with the number of observations), and nonparamtric (the number of parameters grows linearly with the number of observations). Parametric models are typically used when the underlying distribution is known to have some specific form specified by a set of parameters. When the unknown distribution does in fact come from the parametric family, methods such as maximum likelihood [47] are known to perform well. However, when the distribution is not from a parametric family or the parametric family is unknown, semi-parametric (e.g. mixtures of Gaussians [32]) or nonparametric methods (e.g. kernel density estimators [41]) may yield superior performance.

In images, the statistics of pixel intensities or features vectors are rarely known a priori. If one considers a scene of a zebra on grass, the zebra will have somewhat of a bimodal distribution, whereas the grass may have a unimodal distribution. Learning models and estimating the parameters of those models can be quite difficult given the variety in natural image statistics [21]. For this reason, we choose to model random variables nonparametrically using a Parzen density estimate

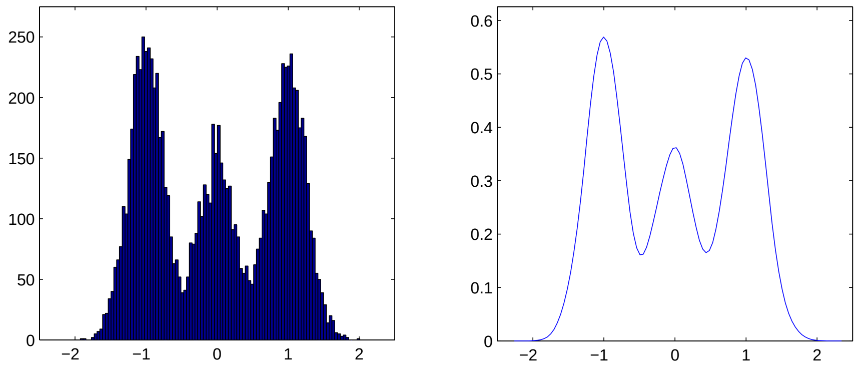

(a) Histogram of Samples (b) KDE of the PDF

Figure 2-3: Level Set Function as Terrain

[37], or what is also known as a kernel density estimate (KDE). The basic concept of the KDE is to smooth the histogram of observations with a specific kernel, $K(x)$, to estimate the value of the probability distribution at any value. An example result of a KDE is shown in Figure 2-3.

We only consider one-dimensional estimates in this section, but the machinery is easily extended to multi-dimensional estimates. The equation for the estimated PDF using a KDE is

$$\hat{p}_X(x) = \frac{1}{Nh}\sum_{i=1}^{N} K\left(\frac{x - x_i}{h}\right), \tag{2.16}$$

where N is the number of source points, x is the point at which the PDF is estimated, x_i is the i^{th} source point, and h is the bandwidth of the kernel used to estimate the PDF. A commonly used kernel and the one we utilize here is a Gaussian kernel:

$$K^G(x) = \frac{1}{\sqrt{\pi}} e^{-x^2}. \tag{2.17}$$

One important detail of using a KDE is selecting an appropriate value of h. If the bandwidth is selected to be too large, the estimated distribution will be too smooth and may not capture multiple modes. However, if the bandwidth is selected to be too small, overfitting may occur with a very peaky distribution. One method is known as the rule of thumb (ROT) bandwidth [41],

$$h_{\text{ROT}} = \left(\frac{4}{3N}\right)^{\frac{1}{5}} \sigma, \tag{2.18}$$

where σ is the standard deviation (known a priori or estimated from the samples). The ROT bandwidth typically oversmooths the density, but we will still use this value for our estimates. The bias of the KDE is the convolution of the kernel with the source density, independent of the number of observations. This would lead one to make the bandwidth zero (i.e. a delta function). However, the variance of the estimate depends on the number of observations, leading to the usual bias/variance tradeoff.

Though the nonparametric estimate has the benefit of being able to capture a much broader class of distributions, it comes with a significant computational cost. If the probability is needed at M points, then the summation in Equation 2.16 needs to be calculated M times. For each summation, N points are added together. Therefore, the total computational complexity of the KDE is $\mathcal{O}(MN)$. Clearly, this computation could be a bottleneck in an algorithm, especially if the estimation needs to be computed multiple times. In the next section, we will discuss a fast approximate algorithm to compute the KDE.

2.2.1 The Fast Gauss Transform

In 1991, Greengard and Strain [20] proposed the algorithm called the Fast Gauss Transform (FGT), which approximately evaluates a sum of Gaussians at multiple points in a fast manner

$$G(x_j) = \sum_{i=1}^{N} q_i e^{\left(\frac{x_j - x_i}{h}\right)^2}, \tag{2.19}$$

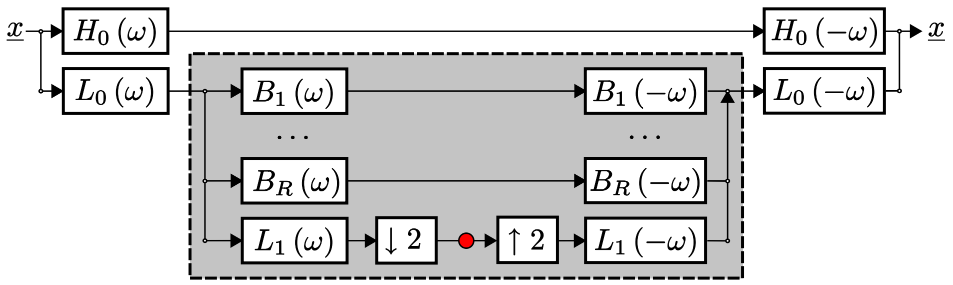
where x_j is the j^{th} point at which the sum is calculated, x_i is the i^{th} source point, and h is the scalar quantity that describes the bandwidth of the kernel.

To achieve a performance gain, the FGT forms a grid by partitioning the sample space into non-overlapping boxes. The algorithm uses these boxes as a clustering of the source and target points. It then uses Hermite and truncated Taylor series expansions to quickly approximate the affect of sources onto the targets. Using this method, the FGT achieves a theoretical computational complexity of $\mathcal{O}(M+N)$, with a constant factor dependent on the accuracy needed and the bandwidth, h.

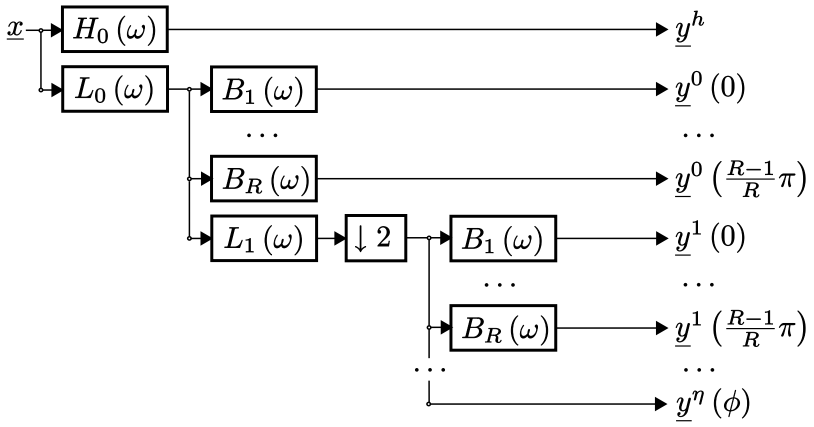
The FGT greatly improves computation times for calculating sums of Gaussians at multiple target points compared to the direct calculation. There has been some more recent work on further improving the FGT. The Improved Fast Gauss Transform (IFGT) was proposed in 2003 by Yang, Duraiswami, Gumerov, and Davis [52]. The improvements on the FGT focus on two major points: a better clustering algorithm and the multivariate Taylor expansion. The IFGT uses the farthest-point clustering algorithm proposed by Gonzalez [18] to more efficiently divide the source and target points. The multivariate Taylor expansion speeds up the Hermite expansion by decreasing the number of terms in the expansion. Though the speed of the FGT and the IFGT are comparable in one dimension, the improvement gained in using the IFGT for a multi-dimensional estimate are much more apparent. We use the provided code of the IFGT algorithm for all nonparametric kernel density estimates.

2.3 Steerable Pyramids

Part of this thesis focuses on developing a new texture model. Our novel representation will be formed upon the basis of the steerable pyramid, which provides a multi-scale, multi-orientation decomposition of an image (depicted in Figure 2-4). The interested reader can consult [15, 43, 42] for a more in-depth description and development of steerable pyramids. The basics will be discussed in this section.



(a) Steerable pyramid recursive structure: replace the red dot with an instance of the gray box, and recursively do this until no more down sampling can be done



(b) Steerable pyramid outputs at orientations (ϕ) and scales (η)

Figure 2-4: Steerable pyramid structure and outputs

A steerable pyramid uses a polar-separable filter to decompose the image into a number of orientation bands, R, at a number of scales. The basic idea of a steerable pyramid is that the output of the polar-separable filter oriented at any angle or scale can be approximated with a small bounded error by interpolating images in the pyramid. Thus, the pyramid provides a complete representation of the image because any oriented output of the polar-separable filter can be found. This property allows us to capture the orientation and scale of a texture fairly straightforwardly.

We choose to implement the steerable pyramid with four orientations. We use the provided implementation (and their filters) to decompose images into the four oriented images, $\underline{y}^{\eta}(0)$, $\underline{y}^{\eta}\left(\frac{\pi}{4}\right)$, $\underline{y}^{\eta}\left(\frac{\pi}{2}\right)$, and $\underline{y}^{\eta}\left(\frac{3\pi}{4}\right)$ at each scale η. The filter output at any orientation, θ, can then be approximated by:

$$y_i^{\eta}(\theta) = \sum_{\phi} \frac{y_i^{\eta}(\phi)\left[\cos(\theta-\phi) + \cos(3(\theta-\phi))\right]}{2}, \tag{2.20}$$

where i is a pixel location and $\phi \in \left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\right\}$.

2.3.1 Previous Texture Models based on Steerable Pyramids

Since the development of steerable pyramids, there has been considerable work on using this filter bank for texture modeling. The most rudimentary model using steerable pyramids is to treat each filter output as statistically independent from each other. Among others, [21] has considered this approach applied to image segmentation. However, this approach excessively simplifies the filter bank output for reasons that will be discussed here.

Many authors (e.g. [11] and [7]) have observed that the outputs of multi-scale models such as the steerable pyramid are not independent. Without actually capturing these dependencies, one can not develop an accurate representation using the steerable pyramid. [11] tries to capture correlations across scale, implicitly introducing correlations across orientation as well. However, their method does not easily extend to trying to capture changes in the appearance of textures.

More recently, [34] (among others) has developed a scale-invariant and orientation-invariant texture measure. However, these approaches typically have two shortcomings. Firstly, they usually do not have a very accurate method to estimate the texture orientation (e.g. t[34] is only accurate up to $\frac{\pi}{4}$ radians). More importantly, they treat scale and orientation as nuisance parameters meaning that once they find them, they remove the scale or rotation and do not consider it anymore. Our approach differs from these approaches in both ways. We are able to accurately and precisely measure the orientation and scale of textures. Additionally, we wish to exploit changes in the orientation and scale of textures under the assumption that these changes provide important information in understanding images that will ultimately aid in segmentation.

2.4 Ali-Silvey Distances

Though stemming from communications, information theory has long been used to analyze other problems such as classification and hypothesis testing. Measures such as the Kullback-Leibler divergence [27] provide an asymptotically provably-optimum hypothesis tester [4]. Additionally, the design of a classifier is aided by the bounds on the probability of error provided by the specific information measure (e.g. the probability of classification error when using mutual information can be lower bounded with Fano's inequality [13]).

Information theoretic measures have also shown to perform well in image segmentation algorithms. For example, [26] considered using mutual information (MI) between the pixel intensities and their associated labels, and [21] and [23] looked at maximizing the J divergence of the densities of the features in the two regions. Both MI and J divergence belong to the broader class of information theoretic measures called Ali-Silvey distances [2]. In the next chapter, we consider segmentation using various Ali-Silvey distances and derive a unique relationship between various measures.

An Ali-Silvey distance is categorized as an increasing function of the expected value of a convex function of a likelihood ratio. It can be written as

$$d\left(p,q\right) = f\left(\mathbb{E}_p\left[C\left(\frac{q\left(\cdot\right)}{p\left(\cdot\right)}\right)\right]\right), \tag{2.21}$$

where $f(\cdot)$ is an increasing function, p and q are two distributions, $C(\cdot)$ is a convex function, and the expectation is taken over the distribution p. We will only consider cases where $f(\cdot) = (\cdot)$, which is true for many commonly used distances. Though these information measures are referred to as "distances", it is important to note that they do not typically satisfy the conditions of a true distance function. For example, KL divergence ($C(\cdot) = -\log(\cdot)$), is not typically symmetric ($D\left(p\|q\right) \neq D\left(q\|p\right)$), and Chernoff distance ($C(\cdot) = (\cdot)^s,\ s \in (0,1)$) does not satisfy the triangle inequality ($d(p,q) \not\leq d(p,g) + d(g,q)$).

Regardless of this fact, Ali-Silvey distances provide a class of information measures that help to distinguish distributions. Their proven usefulness in similar problems such as hypothesis testing allude to possible successes when applied to image segmentation.

Chapter 3

Generic Distance Measures

The image segmentation problem can be formulated as a classification problem where the classes are not known a priori. If we define a label at each pixel, L_i, that indicates which class the pixel belongs to, we can equivalently pose the image segmentation problem as an inference problem of the labels. A common method for parameter estimation (in this case, the labels) is to use a maximum a posteriori (MAP) estimate. If we assume that the pixels are *i.i.d.* conditioned on the labels, it is easily shown (Appendix A.1) that maximizing the posterior probability is equivalent to maximizing the mutual information between a pixel and its label. This should not be surprising, as we know there is an intimate relationship between mutual information and hypothesis testing. This method of maximizing the mutual information between pixel intensities ($p_X\left(x\right)$) and their labels ($p_L\left(L\right)$) for segmentation was considered in [26]. This worked showed promising results on a wide range of grayscale images. Though it fails on complicated texture images, it was still able to segment basic textures from non-textured regions (such as the zebra in Figure 1-1).

They proposed to use nonparametric density estimates for the intensity distributions and found that an approximate gradient ascent velocity for maximizing the mutual information was simply the log likelihood ratio of the pixels on the boundary. Their evolving velocity field is

$$\frac{\partial \varphi_\ell}{\partial t} = \log \frac{p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}, \quad \forall \ell \in \mathcal{C}. \tag{3.1}$$

The reason why this velocity field is only approximating the gradient ascent velocity is because they ignore two terms that contain how the estimated distributions and the probability of pixels within each region change due to the inclusion or exclusion of a pixel on the boundary. By using the assumption that the current segmentation contains a large amount of correct and incorrect pixels, they showed that these other terms do not have a large impact on the evolution. By using the approximations in Equations 3.3 and 3.4 (which will be discussed later), a similar result can be obtained without making the assumption in [26].

The work in [26] was relevant because they considered using a nonparametric density estimate and used mutual information as a means for segmenting images. Mutual information is a quantity that measures the dependence between two random variables. It is equivalent to the KL divergence between the joint distribution and the product of the marginal distributions:

$$I\left(X;L\right) = D\left(p_{XL} \| p_X p_L\right). \tag{3.2}$$

Here the random variable X represents the data at a random pixel which could contain intensity or, as will be used in later chapters, a texture measure. KL divergence belongs to the broader class of information theoretic measures, the Ali-Silvey distances [2], and it seems only plausible to consider any distance measure in the set of Ali-Silvey distances as a means for segmentation.

In the realm of information theory based algorithms for image segmentation, there exists another common approach: maximizing the distance of the distributions conditioned on each label ($p_X^+(x)$ for the pixels inside and $p_X^-(x)$ for the pixels outside). Although this does not directly relate to hypothesis testing, it is still intuitive because this method attempts to separate two regions as much as possible. [21] and [23] both consider segmentation algorithms using this method, proposing to maximize another Ali-Silvey distance, J divergence, of the two distributions: $J(p_X^+, p_X^-)$. Again, one can consider maximizing any Ali-Silvey distance to segment the image.

This chapter focuses on exploring and comparing different distance measures as a criterion for segmentation. We consider the case of maximizing some distance between the joint distribution and the product of the marginals, $d\left(p_{XL}, p_X p_L\right)$, which we call the *label method*, and the case of maximizing the distance between the conditional distributions, $d\left(p_X^+, p_X^-\right)$, which we call the *conditional method*. We derive the gradient ascent curve velocity for a general Ali-Silvey distance in both methods. Additionally, we show that when the distance measure in the conditional method takes on a specific form, which we call the symmetric Ali-Silvey distance, there exists a unique bijective mapping of equivalent distance measures from the label method to the conditional method. Finally, we comment on how to compare the differences of using various measures when segmenting images and provide some results.

3.1 Approximations for the Gradient Ascent Velocities

Throughout this chapter, we will assume that we have enough samples of a distribution such that the law of large numbers (LLN) holds. With this assumption, we can approximate the expected value by averaging over samples drawn from the same distribution and vice versa. This relationship can be expressed mathematically as

$$\mathbb{E}_{p_X}\left[f(\cdot)\right] = \int_{x\in\mathcal{X}} p_X(x)\, f(x)dx \approx \frac{1}{|R|}\int_{i\in R} f(x_i)di, \tag{3.3}$$

where $\mathcal{X}$ is the support of the distribution p_X, each x_i is an observation of a random variable drawn i.i.d. from p_X, and R is a set containing all indices, i, of the observations. This LLN approximation aids in simplifying gradient ascent velocities by using empirical expected values.

One additional approximation is needed to obtain the expressions we will derive in the next section. We claim that for a given smooth function $f(x)$ and a kernel $K(x)$ that is very narrow compared to the smoothness of $f(x)$, the following approximation can be made:

$$\int_{\mathcal{X}} f(x)K(x-a)\,dx \approx f(a). \tag{3.4}$$

This approximation allows for much simpler gradient ascent velocity expressions that are both more compact and efficient to compute.

In KDEs, consistency of the estimate [37] is achieved when (in addition to other constraints) the bandwidth of the kernel satisfies

$$\lim_{N\to\infty} h(N) = 0, \tag{3.5}$$

where N is the number of samples, and $h(N)$ is the chosen bandwidth as a function of the number of pixels. For many typical kernels (such as the Gaussian kernel used here), this constraint implies that the kernel approaches a delta function as the number of samples approaches infinity. Thus, in the limit, the approximation in Equation 3.4 trivially holds.

Regardless of the number of samples, this approximation is always first-order accurate. Because the kernel integrates to one, the entire integral is essentially taking a weighted average of the function $f(x)$ around the point $x = a$. When the kernel bandwidth is much smaller than the changes in $f(x)$, this weighted average is taken over a very small, slowly changing neighborhood. We can approximate $f(x)$ using a first-order Taylor series expansion around the point $x = a$:

$$f(x) = f(a) + f'(a)(x-a) + o\left((x-a)^2\right) \approx f(a) + f'(a)(x-a).$$

Using this Taylor series expansion, the approximation of Equation 3.4 is easily shown to hold:

$$\begin{aligned}
&\int_{\mathcal{X}} f(x)K(x-a)\,dx \\
&\approx \int_{\mathcal{X}} \left[f(a) + f'(a)(x-a)\right] K(x-a)\,dx \\
&= f(a)\int_{\mathcal{X}} K(x-a)\,dx + f'(a)\int_{\mathcal{X}} xK(x-a)\,dx - f'(a)a\int_{\mathcal{X}} K(x-a)\,dx \\
&= f(a) + af'(a) - af'(a) = f(a).
\end{aligned}$$

Though we do not precisely define the notion of smoothness of $f(\cdot)$ or narrowness of $K(\cdot)$, we will argue why there is enough of a distinction for the approximation to hold in our situations. When solving for the gradient ascent velocity of an energy involving nonparametric density estimates, terms similar to those in Equation 3.4 appear where $f(\cdot)$ is typically a function of the densities: $f\left(p_X(\cdot), p_X^+(\cdot), p_X^-(\cdot)\right)$. As we noted in Section 2.2, the bandwidth of the kernel is chosen to be the rule of thumb bandwidth (Equation 2.18) which is inversely related to $N^{1/5}$, where N is the number of samples. The kernel becomes more narrow as more samples are used to estimate the distribution. Additionally, the underlying assumption of choosing a good kernel bandwidth is that the kernel is much narrower than the smoothness of the distribution. If this were not true, than a multi-modal distribution such as a mixture of two Gaussians would be oversmoothed to look like a unimodal Gaussian. Thus, when $f(\cdot)$ takes on the form $f\left(p_X(\cdot), p_X^+(\cdot), p_X^-(\cdot)\right)$, the assumption that the kernel is much narrower than the smoothness of $f(\cdot)$ holds, and the approximation of Equation 3.4 can be used. Principled bandwidth selection is the subject of much research and is considered in more detail in [39] and [41].

In the following sections, we will derive the gradient ascent velocities and express them with a speed $S(x_\ell)$. The speed defines the update of the level set with the following equation:

$$\frac{\partial \varphi_\ell}{\partial t} = S(x_\ell)\,\delta_0(\varphi_\ell), \tag{3.6}$$

where ℓ is a pixel on the level set, x_ℓ is the feature(s) at pixel ℓ, and $\delta_0(\cdot)$ is the derivative to a smooth Heaviside function discussed in Section 2.1.3.

3.2 The Conditional Method

In this section, we derive the gradient ascent velocities for the conditional method. We begin with a general Ali-Silvey distance between the conditional distributions $p_X^+(x)$ and $p_X^-(x)$, weighted by the number of pixels:

$$|\Omega|\, d\left(p_X^+, p_X^-\right) = |\Omega|\, \mathbb{E}_{p_X^+}\left[C\left(\frac{p_X^-(x)}{p_X^+(x)}\right)\right], \tag{3.7}$$

where $C(\cdot)$ is a convex function. It is important to note that this Ali-Silvey distance is a function of the likelihood ratio and not of the priors on the labels, π^+ and π^-. If the priors were indeed known a priori, then they would just be constants that would not affect the velocity. However, we assume that the priors are functions of time and are chosen to be the empirical estimate of observing a label, or $\frac{|R^\pm|}{|\Omega|}$. This does not affect common distance measures such as KL divergence or J divergence because their convex functions, $C(\cdot)$, do not explicitly contain priors. We show in the appendix (Section A.3) that the speed of gradient ascent velocity for this Ali-Silvey distance is

$$\begin{aligned} S_{CM}(x_\ell) =& \frac{1}{\pi^+}\left[C\left(\frac{p_X^-(x_\ell)}{p_X^+(x_\ell)}\right) - d\left(p_X^+, p_X^-\right)\right] + \frac{1}{\pi^+\pi^-}\mathbb{E}_{p_X^-}\left[C'\left(\frac{p_X^-(\cdot)}{p_X^+(\cdot)}\right)\right] \\ & - \left[\frac{1}{\pi^-} + \frac{p_X^-(x_\ell)}{\pi^+ p_X^+(x_\ell)}\right] C'\left(\frac{p_X^-(x_\ell)}{p_X^+(x_\ell)}\right), \end{aligned} \tag{3.8}$$

where $C'(\cdot)$ is the derivative of $C(\cdot)$ with respect to its argument.

3.3 The Label Method

In contrast to the previous method, the label method looks at the distributions of intensities and labels. Specifically, we consider the distance between the joint distribution and the product of marginals. As stated previously, when the KL divergence is used (i.e. $C(\cdot) = -\log(\cdot)$), we have exactly the mutual information between a pixel intensity and its label. We begin this analysis with a general Ali-Silvey distance between the joint and the product of the marginals, also weighted by the number of pixels:

$$\begin{aligned} |\Omega|\, d\left(p_{XL}, p_X p_L\right) &= |\Omega|\, \mathbb{E}_{p_{XL}}\left[C\left(\frac{p_X p_L}{p_{XL}}\right)\right] \\ &= |\Omega| \sum_{l\in\mathcal{L}} \int_{\mathcal{X}} p_{XL}(x,l) C\left(\frac{p_X(x)p_L(l)}{p_{XL}(x,l)}\right) dx \\ &= |\Omega| \sum_{l\in\mathcal{L}} \pi^l \int_{\mathcal{X}} p_X^l(x) C\left(\frac{p_X(x)}{p_X^l(x)}\right) dx, \end{aligned}$$

where $C(\cdot)$ is a convex function and $p_X^l(\cdot)$ is the distribution conditioned on the label taking on a value of l (equivalently $p_{X|L}(\cdot|L=l)$). Here, we consider the two region case, where $L \in \{+,-\}$. We show in the appendix (Section A.4) that the speed of the gradient ascent velocity for this Ali-Silvey distance is

$$\begin{aligned} S_{LM}(x_\ell) = & C\left(\frac{p_X(x_\ell)}{p_X^+(x_\ell)}\right) + \mathbb{E}_{p_X}\left[C'\left(\frac{p_X}{p_X^+}\right)\right] - C'\left(\frac{p_X(x_\ell)}{p_X^+(x_\ell)}\right)\frac{p_X(x_\ell)}{p_X^+(x_\ell)} \\ & -C\left(\frac{p_X(x_\ell)}{p_X^-(x_\ell)}\right) - \mathbb{E}_{p_X}\left[C'\left(\frac{p_X}{p_X^-}\right)\right] + C'\left(\frac{p_X(x_\ell)}{p_X^-(x_\ell)}\right)\frac{p_X(x_\ell)}{p_X^-(x_\ell)}. \end{aligned} \tag{3.9}$$

One advantage of the label method over the conditional method is that the generalization to segmentation with more than two regions is slightly more straightforward. For example, when using J Divergence in the conditional method with more than two regions, what energy functional should be used? It is not as simple as the two region case, where one maximizes $J\left(p_X^+, p_X^-\right)$. In the conditional method, the multiple regions are implicitly represented in the label values. As we will show in the next section, the conditional method and the label method actually have a very intimate relationship; when the convex function of the conditional method takes on a specific form, it is equivalent to using a different convex function in the label method. Through this analysis, one can extend a two-region energy functional in the conditional method to a multi-region energy functional in the label method.

3.4 The Symmetric Ali-Silvey Distance

In the previous two sections, we presented two different versions of Ali-Silvey distances and their resulting gradient ascent velocities. Interestingly, the two methods are related when the convex function takes on a specific form.

We first consider the label method. As shown previously, we can express it as

$$d(p_{XL}, p_X p_L) = \sum_l \pi^l \mathbb{E}_{p_X^l}\left[\tilde{C}\left(\frac{p_X(\cdot)}{p_X^l(\cdot)}\right)\right], \tag{3.10}$$

where $\tilde{C}(\cdot)$ is just a convex function and the tilde is used so that we can distinguish it from another convex function later. For the two region case, we can expand this to be

$$d(p_{XL}, p_X p_L) = \pi^+ \mathbb{E}_{p_X^+}\left[\tilde{C}\left(\frac{p_X(\cdot)}{p_X^+(\cdot)}\right)\right] + \pi^- \mathbb{E}_{p_X^-}\left[\tilde{C}\left(\frac{p_X(\cdot)}{p_X^-(\cdot)}\right)\right]. \tag{3.11}$$

Now, we consider the conditional method. If we use a symmetric distance measure such as J divergence in the conditional method, we can express it as a sum of two Ali-Silvey distances instead of just a single Ali-Silvey distance:

$$\begin{aligned} J\left(p_X^+, p_X^-\right) &= D\left(p_X^+ \| p_X^-\right) + D\left(p_X^- \| p_X^+\right) \\ \Rightarrow d_S(p_X^+, p_X^-) &= d(p_X^+, p_X^-) \quad + d(p_X^-, p_X^+). \end{aligned} \tag{3.12}$$

We use the subscript S to denote that the original Ali-Silvey distance can be written in the symmetric form above. When the original Ali-Silvey distance takes on this form, we can directly use Equation 3.8 to find the gradient ascent velocity of each distance in the symmetric form (with the sign difference taken into account). Thus, the resulting speed of the gradient ascent velocity for an energy function of the symmetric Ali-Silvey distance is just

$$\begin{aligned} S_S\left(x_\ell\right) = & \frac{1}{\pi^+}\left[C\left(\frac{p_X^-\left(x_\ell\right)}{p_X^+\left(x_\ell\right)}\right) - d\left(p_X^+, p_X^-\right)\right] + \frac{1}{\pi^+\pi^-}\mathbb{E}_{p_X^-}\left[C'\left(\frac{p_X^-\left(\cdot\right)}{p_X^+\left(\cdot\right)}\right)\right] \\ & - \left[\frac{1}{\pi^-} + \frac{p_X^-\left(x_\ell\right)}{\pi^+ p_X^+\left(x_\ell\right)}\right] C'\left(\frac{p_X^-\left(x_\ell\right)}{p_X^+\left(x_\ell\right)}\right) \\ & - \frac{1}{\pi^-}\left[C\left(\frac{p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}\right) - d\left(p_X^-, p_X^+\right)\right] - \frac{1}{\pi^+\pi^-}\mathbb{E}_{p_X^+}\left[C'\left(\frac{p_X^+\left(\cdot\right)}{p_X^-\left(\cdot\right)}\right)\right] \\ & + \left[\frac{1}{\pi^+} + \frac{\pi^+ p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}\right] C'\left(\frac{p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}\right). \end{aligned} \tag{3.13}$$

We can expand this symmetrized distance to be

$$d(p_X^+, p_X^-) + d(p_X^-, p_X^+) = \mathbb{E}_{p_X^+}\left[C\left(\frac{p_X^-\left(\cdot\right)}{p_X^+\left(\cdot\right)}\right)\right] + \mathbb{E}_{p_X^-}\left[C\left(\frac{p_X^+\left(\cdot\right)}{p_X^-\left(\cdot\right)}\right)\right]. \tag{3.14}$$

Notice that Equations 3.11 and 3.14 only differ in three ways: there is a prior probability in front each expectation in Equation 3.11, the convex functions are different, and the term within the convex functions are different. The additional prior probability can be included in the convex function, $\tilde{C}(\cdot)$, while still keeping the correct convexity assumptions in the distance measure. The terms within the convex functions are also related, and we show here that these two equations are equivalent when using a specific pair of convex functions, $C(\cdot)$ and $\tilde{C}(\cdot)$.

For Equations 3.11 and 3.14 to be equal, the following condition must hold:

$$\pi^+\tilde{C}\left(\frac{p_X\left(\cdot\right)}{p_X^+\left(\cdot\right)}\right) = C\left(\frac{p_X^-\left(\cdot\right)}{p_X^+\left(\cdot\right)}\right). \tag{3.15}$$

It may seem like this only accounts for one of the two terms to be equal. However, this relationship forces both terms to be equal in the two equations. We rewrite this condition in two equivalent ways:

$$\left\{ \begin{array}{l} \tilde{C}\left(\frac{p_X(\cdot)}{p_X^+(\cdot)}\right) = \frac{1}{\pi^+}C\left(\frac{\frac{1}{\pi^-}\left[p_X(\cdot) - \pi^+ p_X^+(\cdot)\right]}{p_X^+(\cdot)}\right) = \frac{1}{\pi^+}C\left(\frac{1}{\pi^-}\left[\frac{p_X(\cdot)}{p_X^+(\cdot)} - \pi^+\right]\right) \\ C\left(\frac{p_X^-(\cdot)}{p_X^+(\cdot)}\right) = \pi^+\tilde{C}\left(\frac{\pi^+ p_X^+(\cdot) + \pi^- p_X^-(\cdot)}{p_X^+(\cdot)}\right) = \pi^+\tilde{C}\left(\pi^-\frac{p_X^-(\cdot)}{p_X^+(\cdot)} + \pi^+\right) \end{array} \right. . \tag{3.16}$$

Note that these expressions are functions of the prior probabilities, π^+ and π^-. To generalize this relationship, we add a subscript l to the convex functions that corresponds to the label of the expectation in the Ali-Silvey distance.

$$\tilde{C}_l(\cdot) = \frac{1}{\pi^l}C_l\left(\frac{1}{1-\pi^l}\left[(\cdot) - \pi^l\right]\right) \tag{3.17}$$

$$C_l(\cdot) = \pi^l\tilde{C}_l\left(\left(1-\pi^l\right)(\cdot) + \pi^l\right) \tag{3.18}$$

This relationship has a very strong meaning. If one chooses to optimize a symmetric Ali-Silvey distance of the form in Equation 3.14 with a specific $C_l(\cdot)$ based on the conditional method, Equation 3.17 tells us that there exists a convex function $\tilde{C}_l(\cdot)$ that combined with the label method of Equation 3.11 produces exactly the same result. A similar relationship can be said in the other direction using Equation 3.18. This implies that as long as the Ali-Silvey distance in the conditional method is symmetric and can be written in the form of Equation 3.12, one can find an equivalent optimization with a different convex function in the label method. Additionally, this relationship allows one to extend many conditional method segmentation algorithms that are limited to two-regions to a label method algorithm able to segment images into multiple regions.

3.5 The Balanced Symmetric Ali-Silvey Distance

We consider one additional form of Ali-Silvey distances, which we define as balanced and symmetric. Our notion of a symmetric measure is an Ali-Silvey distance that can be written in the form of Equation 3.12. As shown in Equations 3.17 and 3.18, one of the differences between the label method and the conditional method is the additional constant factor of the prior probability. In the label method, we know that the gradient ascent velocity (Equation 3.9) is independent of the priors. However, as shown in Equation 3.8, the conditional method is sensitive to the estimated label priors. As an example, consider KL divergence (where the convex function is just $-\log(\cdot)$). When the likelihood ratio is one, the KL divergence is zero. However, evaluating Equation 3.8 with this convex function when the likelihood ratio is one shows that the velocity is not necessarily zero:

$$S_{CM,KL}\left(x_\ell \middle| \frac{p_X^+(x_\ell)}{p_X^-(x_\ell)} = 1\right) = \frac{1}{\pi^+} D\left(p^+ \| p^-\right). \tag{3.19}$$

A closer examination of the derivation for this velocity shows that a distance that is multiplied by a label prior will not have this term. Thus, we define a balanced symmetric Ali-Silvey distance as a measure that can take on the following form

$$d_{BS}(p_X^+, p_X^-) = \pi^+ d(p_X^+, p_X^-) + \pi^- d(p_X^-, p_X^+). \tag{3.20}$$

Notice that this form of an Ali-Silvey distance measure is even more similar to Equation 3.11. Additionally, the relationships shown in Equations 3.17 and 3.18 for the balanced symmetric form of a distance measure do not have the additional prior factors outside of the convex functions. This hints that a distance of this form is inherently more similar to the distance used in the label method. In Section A.5 of the appendix, we show the gradient ascent velocity for this balanced symmetric distance to be

$$\begin{aligned} S_{BS}(x_\ell) = & C\left(\frac{p_X^-(x_\ell)}{p_X^+(x_\ell)}\right) + \frac{1}{\pi^-}\mathbb{E}_{p_X^-}\left[C'\left(\frac{p_X^-(\cdot)}{p_X^+(\cdot)}\right)\right] - \left[\frac{\pi^+}{\pi^-} + \frac{p_X^-(x_\ell)}{p_X^+(x_\ell)}\right] C'\left(\frac{p_X^-(x_\ell)}{p_X^+(x_\ell)}\right) \\ & - C\left(\frac{p_X^+(x_\ell)}{p_X^-(x_\ell)}\right) - \frac{1}{\pi^+}\mathbb{E}_{p_X^+}\left[C'\left(\frac{p_X^+(\cdot)}{p_X^-(\cdot)}\right)\right] + \left[\frac{\pi^-}{\pi^+} + \frac{p_X^+(x_\ell)}{p_X^-(x_\ell)}\right] C'\left(\frac{p_X^+(x_\ell)}{p_X^-(x_\ell)}\right). \end{aligned} \tag{3.21}$$

Table 3.1: Summary of compared distance measures. Because each measure is just a form of KL divergence, the convex function used in the equations listed in the third column are all $C(\cdot) = -\log(\cdot)$.

Distance Measure	Expression	$\frac{\partial \varphi_\ell}{\partial t}$	$C(\cdot)$
$I(X;L)$	$D\left(p_{XL} \Vert p_X p_L\right)$	Eqn. 3.9	$-\log(\cdot)$
$J\left(p_X^+, p_X^-\right)$	$D\left(p_X^+ \Vert p_X^-\right) + D\left(p_X^- \Vert p_X^+\right)$	Eqn. 3.13	$-\log(\cdot)$
$J_B\left(p_X^+, p_X^-\right)$	$\pi^+ D\left(p_X^+ \Vert p_X^-\right) + \pi^- D\left(p_X^- \Vert p_X^+\right)$	Eqn. 3.21	$-\log(\cdot)$

3.6 Comparison of Measures

As mentioned previously, the derivation of the gradient ascent velocities assumed that the convex function was only a function of one time-dependent term: the likelihood ratio. When true priors are known and are approximated with empirical estimates, one cannot just plug in the convex function to the gradient ascent velocity expressions derived in Equations 3.8 and 3.9. Keeping this in mind, we compare three different measures: the mutual information of intensities and labels $I(X;L)$, the J divergence of the conditional distributions $J\left(p_X^+, p_X^-\right)$, and the balanced J divergence of the conditional distributions $J_B\left(p_X^+, p_X^-\right)$. These measures and their relationships to our derivations of general Ali-Silvey distance measures are shown in Table 3.1.

Using the equations listed, it is straightforward to simplify the expressions for each specific measure. We find the speed of the gradient ascent velocities to be

$$S_I\left(x_\ell\right) = \log\left(\frac{p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}\right), \tag{3.22}$$

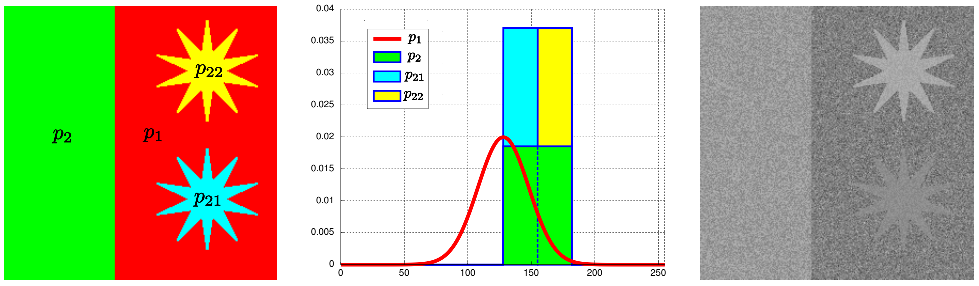
$$\begin{aligned} S_J\left(x_\ell\right) = & \frac{1}{\pi^+}\left[\log\left(\frac{p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}\right) - D\left(p_X^+ \Vert p_X^-\right) - \frac{p_X^-\left(x_\ell\right)}{p_X^+\left(x_\ell\right)} + 1\right] \\ & - \frac{1}{\pi^-}\left[\log\left(\frac{p_X^-\left(x_\ell\right)}{p_X^+\left(x_\ell\right)}\right) - D\left(p_X^- \Vert p_X^+\right) - \frac{p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)} + 1\right], \end{aligned} \tag{3.23}$$

$$S_{J_B}\left(x_\ell\right) = 2\log\left(\frac{p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}\right) + \frac{\pi^+}{\pi^-}\left[\frac{p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)} - 1\right] - \frac{\pi^-}{\pi^+}\left[\frac{p_X^-\left(x_\ell\right)}{p_X^+\left(x_\ell\right)} - 1\right]. \tag{3.24}$$

Each speed is a function of the likelihood ratio, $\frac{p_X^+(x_\ell)}{p_X^-(x_\ell)}$, and the priors π^+ and π^-. In addition, the J divergence term depends on the actual KL divergences, which we will refer to as the *individual divergences*. By varying these parameters, we can get a sense of what each distance measure does differently.

The calculated gradient ascent velocities are only a result of the information theoretic term and have not incorporated the regularization term that penalizes the curve length as shown in Equation 2.4. Even though Equation 2.4 shows a constant scaling factor of α in front of the curve length penalty, we can equivalently think of it as a constant scaling factor of $\frac{1}{\alpha}$ in front of the information theoretic term. This tradeoff between separability and smoothness gives an additional parameter that scales the entire speed function depending on the value of α.

We proceed to compare the three speeds of the gradient ascent velocity by plotting their values as a function of the likelihood ratio. We show these functions for various values of priors, α's, and



(a) Mask (b) Probability Distributions (c) Synthetic Image

Figure 3-1: A synthetic image created such that the area of pixels from p_1 is the same as the area of pixels from p_2, p_{21}, and p_{22}.

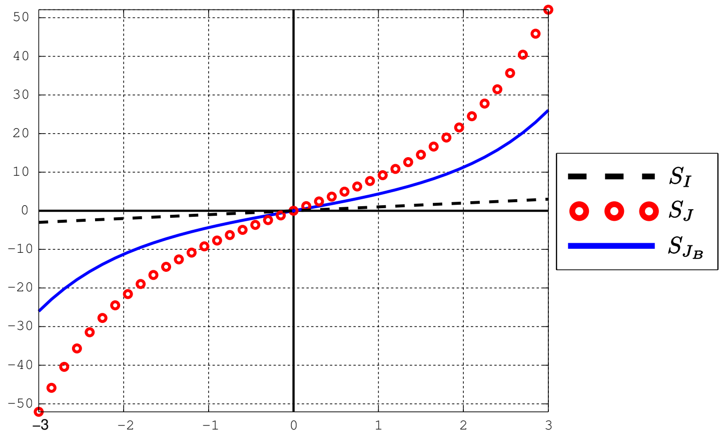


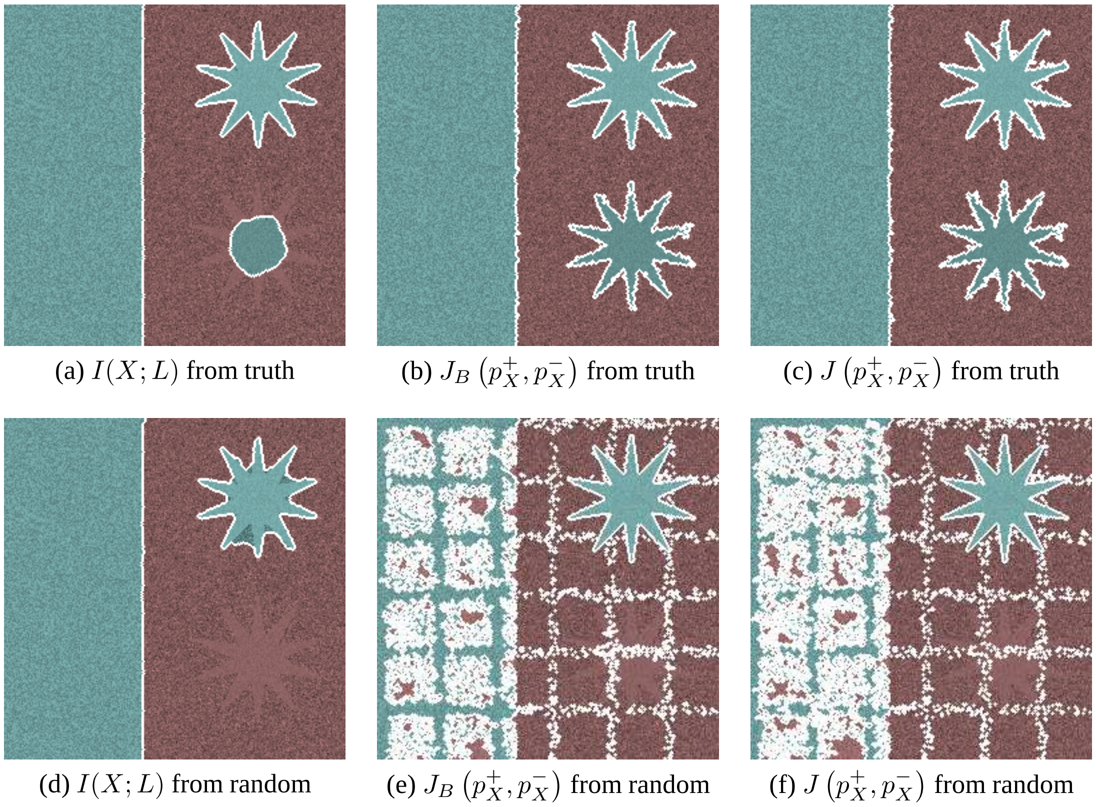
Figure 3-2: A comparison of the speed of three distance measures as a function of $\log \frac{p_X^+(\cdot)}{p_X^-(\cdot)}$. All parameters are equal (i.e. $\pi^+ = \pi^-$, $D\left(p_X^+ \| p_X^-\right) = D\left(p_X^- \| p_X^+\right)$, and $\alpha_I = \alpha_J = \alpha_{JB}$).

individual divergences. In addition, we will show resulting segmentations starting from the ground truth and a random initialization.

We created the synthetic image shown in Figure 3-1 for comparative purposes. The image is constructed such that p_{21} and p_{22} are both subfunctions (properly scaled) of p_2. If a pixel is random drawn from p_{21} or p_{22}, then it has an equivalent distribution that is equal to p_2. Additionally, we have constrained the areas such that the area of pixels from p_1 is the same as the area of pixels from p_2, p_{21}, and p_{22}. The distribution p_{21} has support over the range of values where p_1 and p_2 are comparable in likelihood, while the distribution p_{22} contains values where p_2 is much more likely than p_1. This difference in likelihoods coupled with the very sharp star shape will allow us to evaluate the tradeoffs between the information theoretic term with the regularization term.

3.6.1 Equal Regularization Weights

We first plot, in Figure 3-2, the speeds as a function of the log likelihood ratio when all the parameters are equal. This plot shows that mutual information induces a gradient ascent that is linearly proportional to the log likelihood ratio, and that J divergence and balanced J divergence both induce a gradient ascent that is exponentially proportional. It also implies that each of the distance


(a) $I(X;L)$ from truth (b) $J_B\left(p_X^+, p_X^-\right)$ from truth (c) $J\left(p_X^+, p_X^-\right)$ from truth

(d) $I(X;L)$ from random (e) $J_B\left(p_X^+, p_X^-\right)$ from random (f) $J\left(p_X^+, p_X^-\right)$ from random

Figure 3-3: Segmentation using three different measures and equal regularization weighting ($\alpha_I = \alpha_{J_B} = \alpha_J = 1$). Top row is obtained initializing to the correct segmentation and bottom row is obtained with a random initialization.

measures puts a different weight on the information theoretic terms versus the regularization term. Clearly, J divergence emphasizes the information theoretic terms the most, followed by balanced J divergence and mutual information.

Figure 3-3 shows segmentation results obtained using equal regularization weights. From the ground truth segmentations, the balanced J divergence and J divergence hold the shape of the stars better. The shape will only change when the regularization weight is high enough. However, from the segmentations obtained using random initializations, it is clear that the curve length penalty is not large enough for the balanced J divergence and J divergence cases. There are many single pixels regions with very high likelihood that are not eliminated because of the small regularization. In the mutual information case with random initializations, the segmentation is not able to capture the bottom star, but also does not contain the single pixel regions of the balanced J divergence and J divergence. This result indicates that the regularization weight chosen for the mutual information case is sufficient for eliminating unrealistically small regions.

3.6.2 Equal Slope at Zero

The unbalanced weighting of information theoretic and regularization terms discussed in the previous section leads to another comparison of the three distance terms. By choosing the α's appropriately, we can enforce the three speeds to have the same slope when the log likelihood ratio is zero. Assuming equal priors and divergences, we easily find the relationship to be

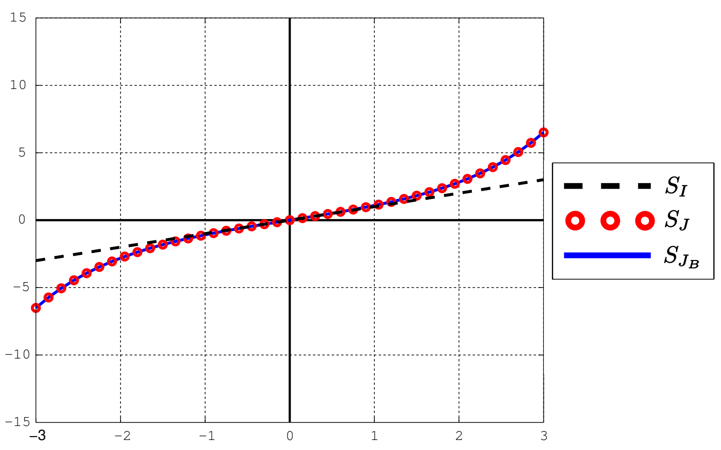
$$\alpha_I = \frac{1}{2}\alpha_{J_B} = \frac{1}{4}\alpha_J = C, \tag{3.25}$$

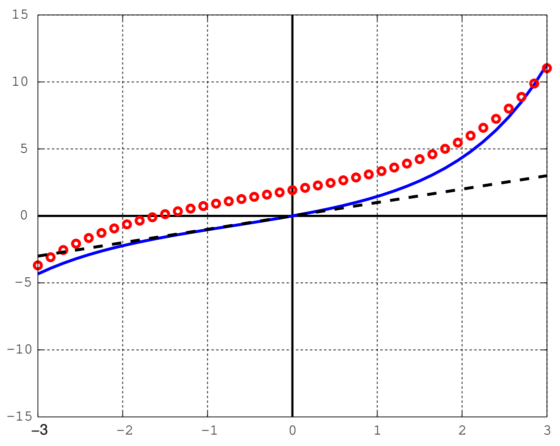
where C is any constant. The plots in Figure 3-4 show the three speeds when this condition is met (with $C = 1$). The first of these plots, Figure 3-4a, shows the speeds when the priors and individual divergences are equal. The gradient ascent velocity of the J divergence is then exactly equal to that of the balanced J divergence. When using this set of α's, the three speeds behave very similarly for small log likelihood ratios, or equivalently when there is not very much evidence that a pixel should belong in one region over another. However, the J divergence and balanced J divergence terms put much more emphasis on pixels that have a very large magnitude of log likelihood ratios. One would expect that segmentations only differ when there is a large amount of evidence indicating that a pixel should be in one region. If that region is very small, then the curve length penalty in the mutual information case may eliminate the region because the information theoretic velocity is not large enough to overcome it. In the J divergence and balanced J divergence cases, the extra emphasis on the information theoretic terms may overcome the competing effect of the curve length penalty and allow the small region to grow.

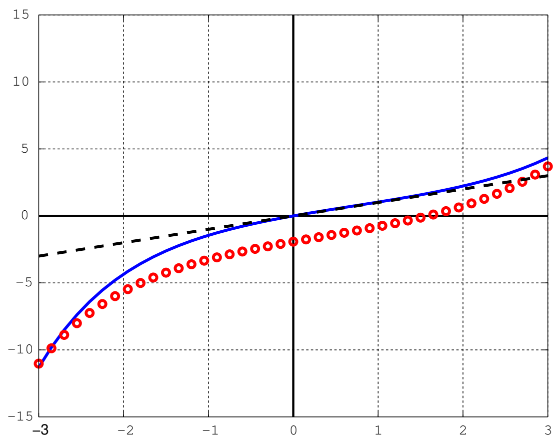
Figure 3-4a considers the case when the priors and individual divergences are equal (i.e. $\pi^+ = \pi^-$ and $D\left(p_X^+ \| p_X^-\right) = D\left(p_X^+ \| p_X^-\right)$). It is more interesting to consider Figures 3-4b and 3-4c where the priors are different. Unlike the gradient ascent velocity for mutual information, both J divergence and balanced J divergence depend on the priors. As stated previously, these priors are not known a priori so they are estimated by the empirical size of each region. When $\pi^+ > \pi^-$, both J divergence and balanced J divergence weight positive log likelihood ratios more than negative values. This is reasonable because the difference in priors implies that the label at a random pixel has a higher probability of belonging to R^+ rather than R^-. In addition, when the priors are not equal, the speed of the J divergence term is nonzero when the log likelihood ratio is zero. This could be a self-fulfilling prophecy in that it will tend to grow larger regions, and as those larger regions grow, the gradient ascent velocity tries to grow it even more.

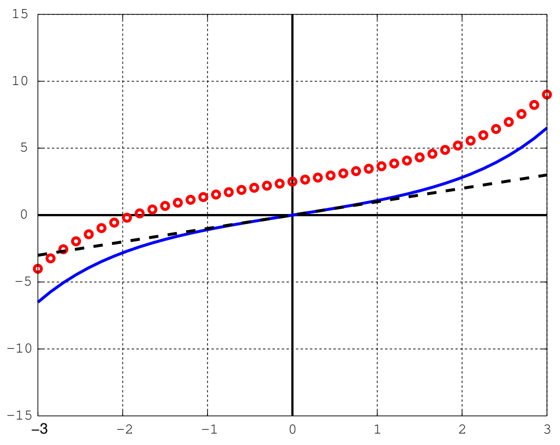
Figures 3-4d and 3-4e consider the case when the individual divergences are different (i.e. $D\left(p_X^+ \| p_X^-\right) \neq D\left(p_X^+ \| p_X^-\right)$). As expected, the mutual information and balanced J divergence gradient ascent velocities are unaffected. However, the difference in divergences affects the zero crossing of the J divergence case. To analyze this difference, we consider a case where both conditional distributions are Gaussian with zero mean and different variances, shown in Figure 3-5. If we consider a pixel that has equal probability of being drawn from either p_X^+ or p_X^- (which takes a value that is the intersection of the blue and red curves in Figure 3-5), there is no evidence that it was drawn from one distribution over the other. However, the plots show that even when the probability of the pixel belonging to either region is the same, and the prior on the labels is also equal, there is still a bias to include the pixel in the R^+ region. Unlike the different prior case, there is no reasonable explanation for this biased zero crossing.

We now consider segmentations of our synthetic image with this equal slope case, shown in Figure 3-6. From the plot shown in Figure 3-4a, we see that even though the slope is equal when the likelihoods are equal, for every other likelihood value, more weight is put on the divergence terms as compared to the mutual information terms. In the ground truth segmentations, the lower star is still captured very well by the divergence terms because it has more weight on the likeli-


(a) $\frac{\pi^+}{\pi^-} = 1$, $D\left(p_X^+ \| p_X^-\right) = D\left(p_X^- \| p_X^+\right) = 10$


(b) $\frac{\pi^+}{\pi^-} = 2$, $D\left(p_X^+ \| p_X^-\right) = D\left(p_X^- \| p_X^+\right) = 10$


(c) $\frac{\pi^+}{\pi^-} = \frac{1}{2}$, $D\left(p_X^+ \| p_X^-\right) = D\left(p_X^- \| p_X^+\right) = 10$


(d) $\frac{\pi^+}{\pi^-} = 1$, $D\left(p_X^+ \| p_X^-\right) = 10$, $D\left(p_X^- \| p_X^+\right) = 20$

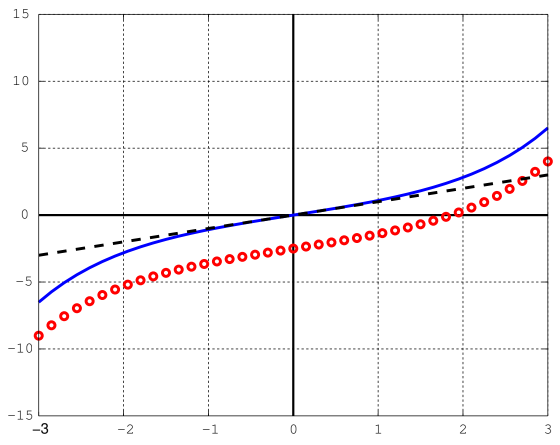

(e) $\frac{\pi^+}{\pi^-} = 1$, $D\left(p_X^+ \| p_X^-\right) = 20$, $D\left(p_X^- \| p_X^+\right) = 10$

Figure 3-4: A comparison of the speed of three distance measures as a function of $\log \frac{p_X^+(\cdot)}{p_X^-(\cdot)}$. α is chosen such that each speed has the same slope when the likelihood ratio is 1. (a) Equal priors and divergences; (b)-(c) Difference in priors by a factor of 2; (d)-(e) Difference in divergences by a factor of 2

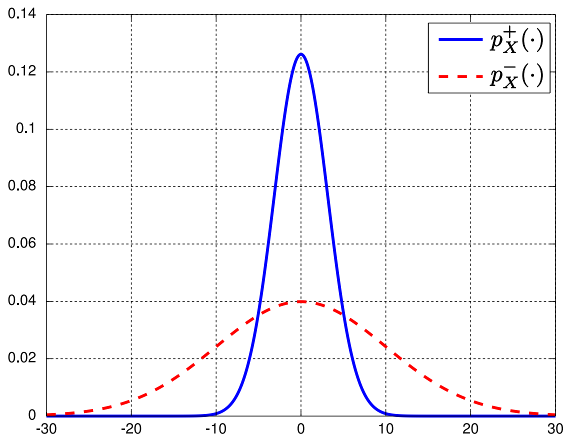


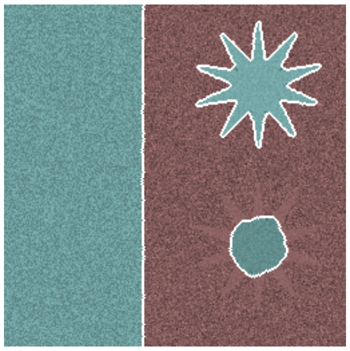
Figure 3-5: Two Gaussian distributions with different variances. $\sigma_+^2 < \sigma_-^2$ leads to $D\left(p_X^+ \| p_X^-\right) < D\left(p_X^- \| p_X^+\right)$.

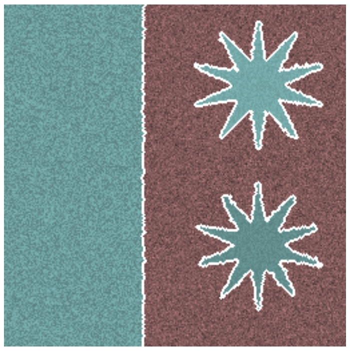
hood. Though there are many less small pixel regions compared to the previous section, we still see some in the divergence terms when we start with a random initial seed. The segmentations obtained using J divergence and balanced J divergence are slightly different, but we have verified that each segmentation is a local extremum for both distance measures. As expected, the upper star is captured more accurately in the divergence cases because the regularization weight does not overcome the likelihood terms.

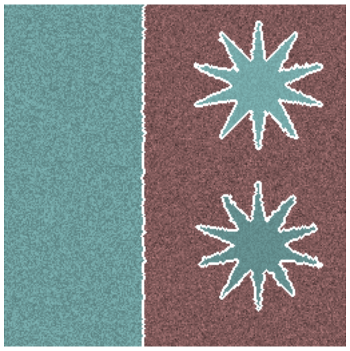
3.6.3 Equal Comparison

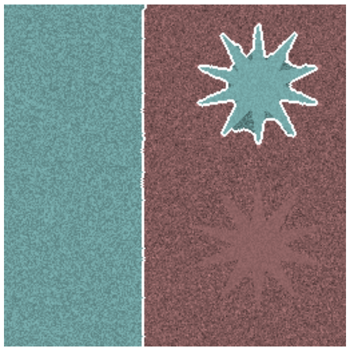
The previous section considered the case where we forced the regularization parameters such that the speed had equal slope for all distance measures when the log likelihood was zero. However, the J divergence and balanced J divergence velocities were still putting more weight on the likelihood terms compared to the mutual information case. With a few tries, we were able to identify a regularization parameter for each distance measure such that their segmentation on our synthetic image was approximately the same. The segmentation results using this relationship ($\alpha_I = \frac{1}{8}\alpha_{J_B} = \frac{1}{16}\alpha_J = 1$) are shown in Figure 3-7. Figure 3-8 shows the gradient ascent velocity speeds as a function of the log likelihood ratio.

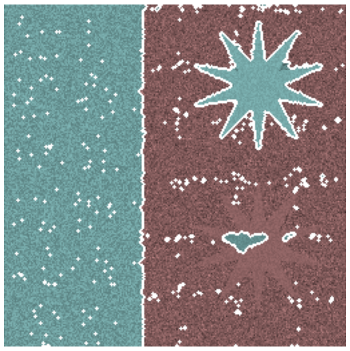
Using this set of curve length penalty weights, we segmented a set of images with each algorithm. The results are shown in Figure 3-9. In general, the three measures produce similar results. The most notable difference is seen in the zebras of the last row of Figure 3-9. The white stripes of the zebra are much more likely to be in the background based on pixel intensity. In the mutual information case, the curve length penalty is able to overcome this likelihood and include some white stripes with the black stripes. This may explain why a pixel intensity based segmentation algorithm (which has no specific consideration of texture analysis) is still able to successfully separate the textured zebra from the background. However, in the divergence cases, the exponential curve of the velocity on the likelihood term completely dominates the curve length penalty. We explore these results further in Figure 3-10. The histograms shown in the middle column of Figure 3-10 count the log likelihood ratios (where we have binned anything less than -5 or greater than 5 into the first or last bin respectively. In the divergence cases, the likelihood ratios have a very high


(a) $I(X;L)$ from truth


(b) $J_B\left(p_X^+, p_X^-\right)$ from truth


(c) $J\left(p_X^+, p_X^-\right)$ from truth


(d) $I(X;L)$ from random


(e) $J_B\left(p_X^+, p_X^-\right)$ from random

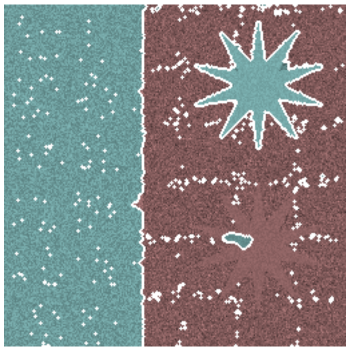
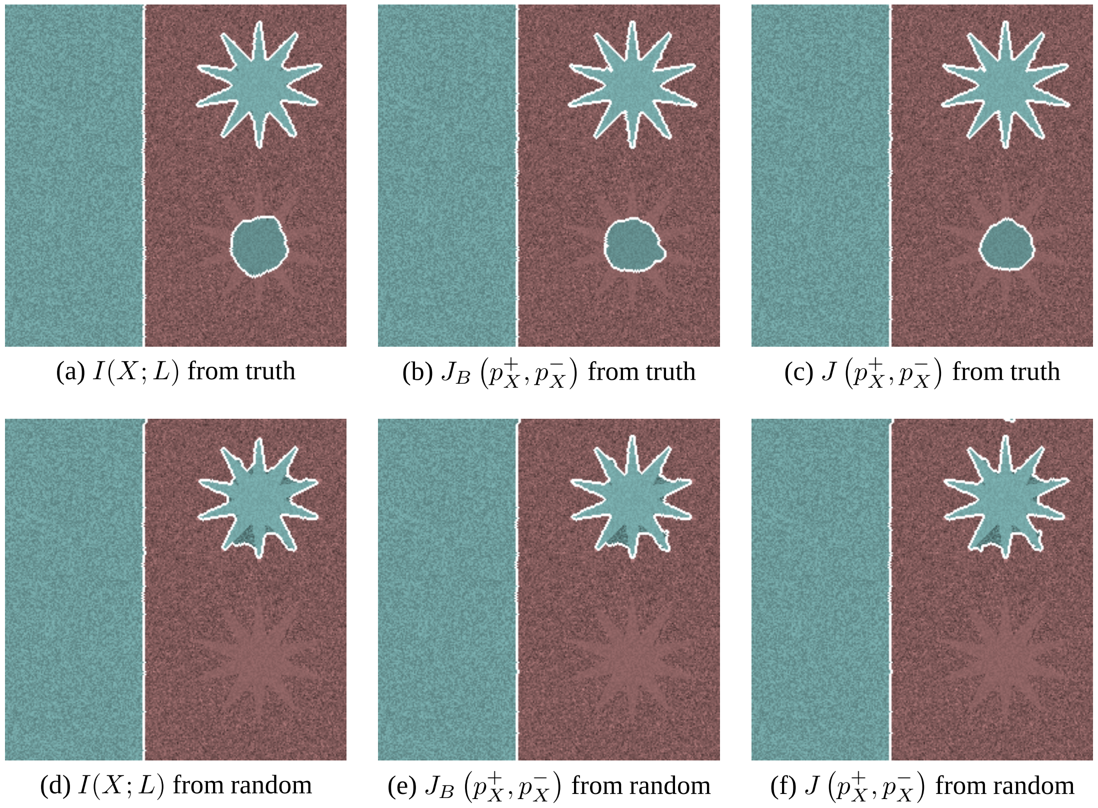

(f) $J\left(p_X^+, p_X^-\right)$ from random

Figure 3-6: Segmentation using three different measures and equal slope in the log likelihood domain when likelihoods are equal ($\alpha_I = \frac{1}{2}\alpha_{J_B} = \frac{1}{4}\alpha_J = 1$). Top row is obtained initializing to the correct segmentation and bottom row is obtained with a random initialization.


(a) $I(X;L)$ from truth (b) $J_B\left(p_X^+, p_X^-\right)$ from truth (c) $J\left(p_X^+, p_X^-\right)$ from truth

(d) $I(X;L)$ from random (e) $J_B\left(p_X^+, p_X^-\right)$ from random (f) $J\left(p_X^+, p_X^-\right)$ from random

Figure 3-7: Segmentation using three different measures and uneven weighting ($\alpha_I = \frac{1}{8}\alpha_{J_B} = \frac{1}{16}\alpha_J = 1$). Top row is obtained initializing to the correct segmentation and bottom row is obtained with a random initialization.

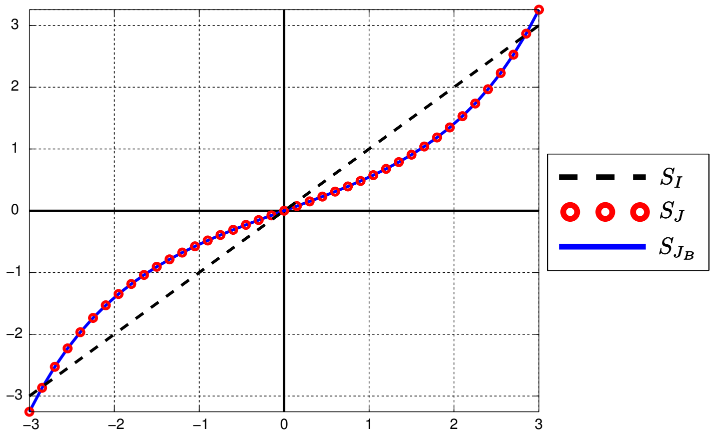


Figure 3-8: A comparison of the speed of three distance measures as a function of $\log \frac{p_X^+(\cdot)}{p_X^-(\cdot)}$. All parameters are equal (i.e. $\pi^+ = \pi^-$, $D\left(p_X^+ \| p_X^-\right) = D\left(p_X^- \| p_X^+\right)$, and $\alpha_I = \frac{1}{8}\alpha_{J_B} = \frac{1}{16}\alpha_J = 1$).

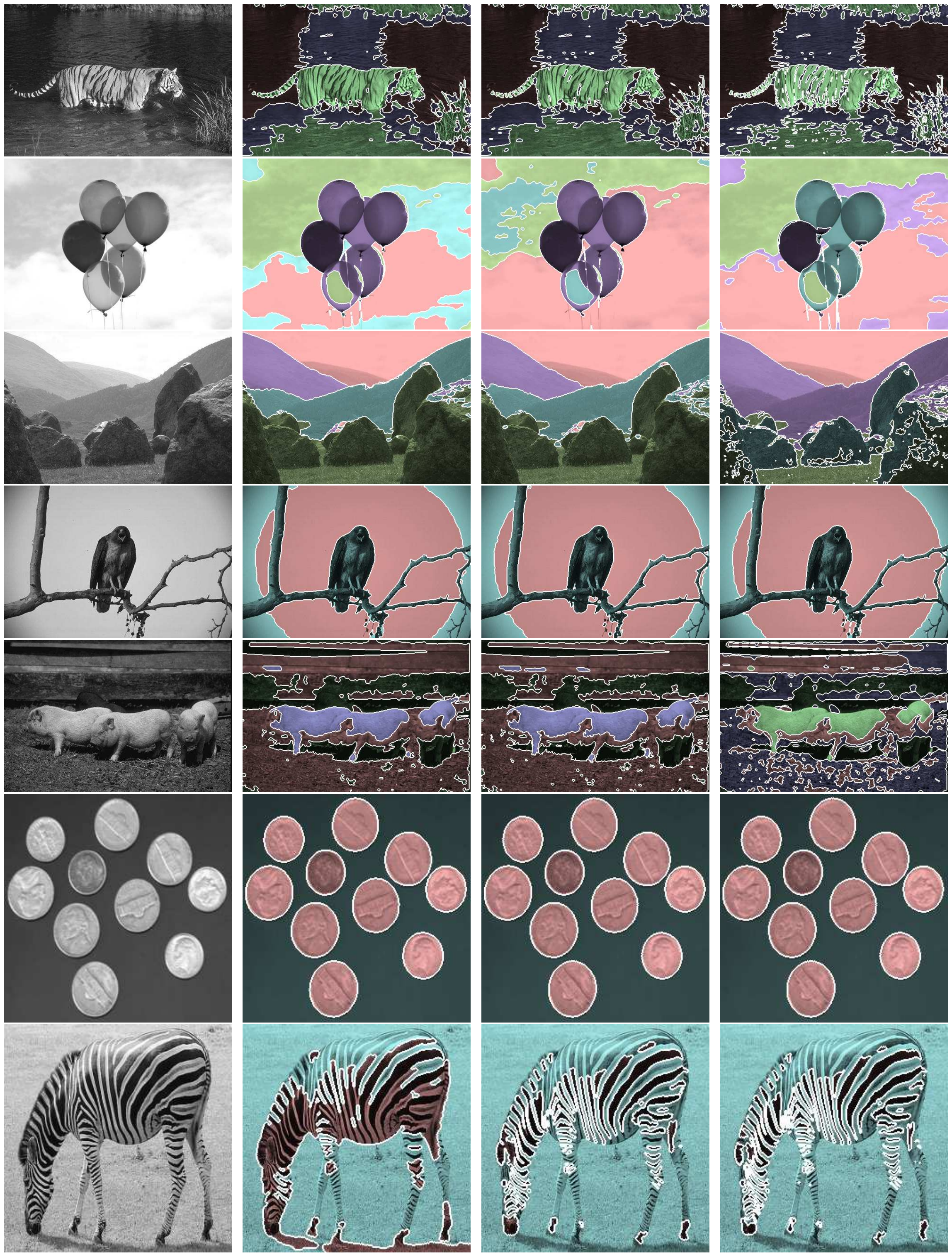


Figure 3-9: Segmentation using three different measures and uneven weighting ($\alpha_I = \frac{1}{8}\alpha_{J_B} = \frac{1}{16}\alpha_J = 1$). First column is the original image, second column is obtained using mutual information, third column is obtained using balanced J divergence, and fourth column is obtained using J divergence.

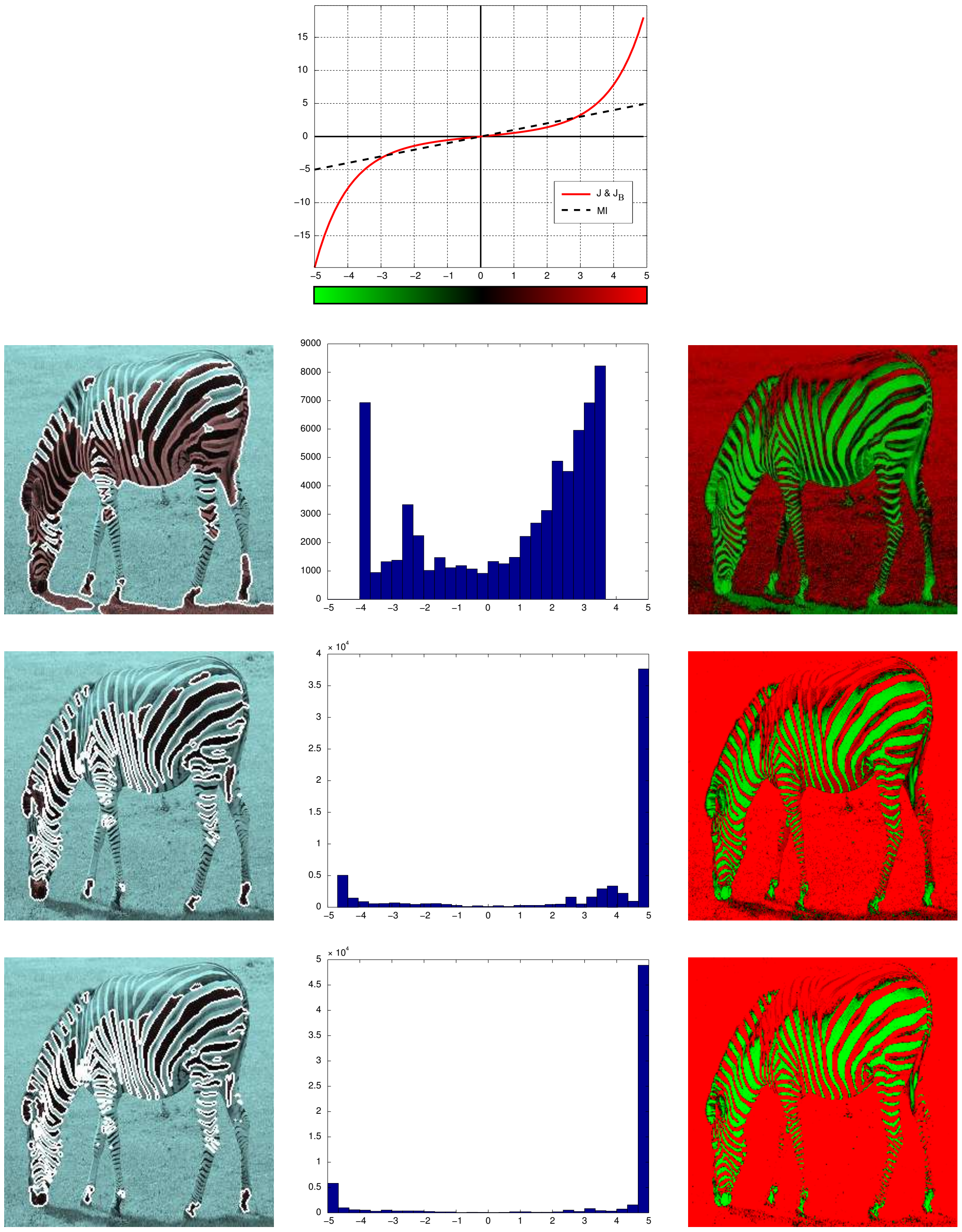


Figure 3-10: Comparing segmentation results using different measures. The top plot shows the speeds plotted as a function of the log likelihood ratio (assuming equal priors and individual divergences). The left column shows the segmentation results, the middle column shows the histograms, and the right column shows the log likelihood ratios mapped to a color (using the colorbar in the speed plot). The first row is mutual information, the second row is balanced J divergence, and the last row is J divergence.

value for much of the image. The third column maps the log likelihood ratio to a color, clearly showing that the white stripes of the zebra have a high log likelihood ratio. Looking at the plot of gradient ascent velocity speeds shows that when the log likelihood ratio has an absolute value of 5 or more, the speed is so large that it most likely will overcome the regularization term.

One can argue about the benefit of using any of the distance measures. In the zebra case, mutual information seems to be a better criterion, but one can create other synthetic images that favor other distance measures. We typically choose to implement our subsequent algorithms with mutual information for simplicity, keeping in mind that the extension to balanced J divergence or J divergence is fairly straightforward to implement.

Chapter 4

Texture Modeling

In the previous chapter, we considered segmentation using level set methods based on a variety of Ali-Silvey distance measures. The results presented were based on algorithms that considered pixel intensities and treated the pixels as statistically independent. Though this assumption is often made, it typically does not hold in natural images. We know that pixels must have some spatial correlation because of the underlying scene that they represent. In this chapter, we attempt to model these correlations by considering a new representation of the image that is not based solely on intensity values.

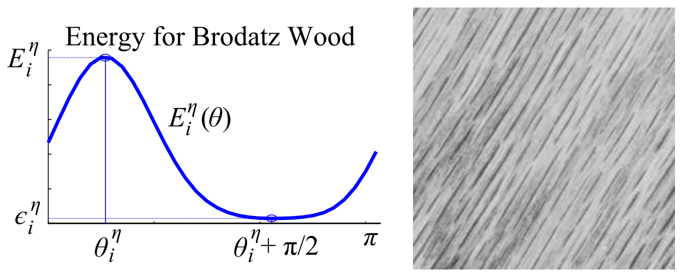
One particular type of image that does not perform well under pixel-based methods are textured images. These images contain very complex repeating patterns which that cannot be captured in pixel-based methods. Instead, we present a method to represent these spatial correlations with a novel texture feature extraction based on the steerable pyramid representation (discussed in Section 2.3). Our features are designed to detect and measure the dominant orientation and scale at each pixel in an image. We show that our feature set allows us to decompose an image into separate contrast, bias, orientation, and scale fields. Because of this decomposition, we are able to capture spatial correlations and model smoothly changing textures in each of our four features. Additionally, our representation allows us to estimate an unknown radiometric camera response function and a shading image that can be used to recover shapes of objects. While the measure implicitly assumes that a dominant orientation and scale exist, we show empirically that it works well on a much broader set of textures.

4.1 Texture Descriptors

In this section we describe the machinery used to extract our features from an image. Once the features describing contrast, bias, orientation, and scale are found, we impose an additional set of smooth Markov random fields (MRFs) to capture the spatial changes of each feature.

4.1.1 Extracting the Dominant Scale and Orientation

Within the steerable pyramid representation, the filter output at any orientation can be well approximated [16] by interpolating the outputs of the filter bank at a discrete set of orientations. This attribute of the steerable pyramid, in addition to its multi-scale representation, allows for accurate



(a) (b)

Figure 4-1: (a) Features at a given scale at the center pixel w.r.t. orientation; (b) A Brodatz image used in the plot

and efficient estimation of texture orientations. Within each scale, denoted by η, we use the response at four orientations: $\underline{y}^\eta(0)$, $\underline{y}^\eta\left(\frac{\pi}{4}\right)$, $\underline{y}^\eta\left(\frac{\pi}{2}\right)$, and $\underline{y}^\eta\left(\frac{3\pi}{4}\right)$. The filter output at an arbitrary orientation, θ, can the be well approximated by:

$$y_i^\eta(\theta) = \sum_\phi \frac{y_i^\eta(\phi)\left[\cos(\theta-\phi)+\cos(3(\theta-\phi))\right]}{2} \tag{4.1}$$

where i is a pixel location and $\phi \in \left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\right\}$. Let

$$E_i^\eta(\theta) \triangleq \frac{1}{|R_i^\eta|}\sum_{j\in R_i^\eta} |y_j(\theta)|^2 \tag{4.2}$$

be the angular energy over a local region R_i^η (typically a 3x3 region) at an angle θ, scale η, and location i. Using golden section search [25], we find the orientation within each scale, θ_i^η, with the maximum angular energy. This is defined to be the orientation of the texture at scale η, that is

$$\theta_i^\eta \triangleq \arg\max_\theta E_i^\eta(\theta). \tag{4.3}$$

As $E_i^\eta(\theta)$ is periodic with period π, we only search the range $[0,\pi)$. As a function of scale η and location i, we extract the contrast energy

$$E_i^\eta \triangleq E_i^\eta(\theta_i^\eta) \tag{4.4}$$

and what we refer to as the residual contrast energy

$$\epsilon_i^\eta \triangleq E_i^\eta\left(\theta_i^\eta+\frac{\pi}{2}\right). \tag{4.5}$$

The term residual energy is used since, for strongly oriented textures, the energy of the response when the filter is orthogonal to the dominant orientation tends to be small. These features are depicted graphically in Figure 4-1.

Within a scale, the last feature that we introduce is the bias of the texture, μ_i^η. This term captures the low-frequency energy of the image that is not directly measured by the steerable pyramid. While nearly any lowpass filter is suitable, we use a circularly symmetric Gaussian blur filter.

Summarizing, in each scale and at each pixel, we calculate the contrast energy (E_i^η), the residual energy (ϵ_i^η), the orientation (θ_i^η), and the bias (μ_i^η). For each location i we then select the set of features corresponding to the scale, η_i, with maximum contrast energy:

$$\eta_i \triangleq \arg\max_{\eta} E_i^\eta. \tag{4.6}$$

These become the feature set at each pixel. We drop the superscript of scale for each feature once we have η_i. This leads us to our final feature set at pixel i: $\{\eta_i, E_i, \epsilon_i, \theta_i, \mu_i\}$.

Except η_i, all features are continuous. We sample scale logarithmically in order to approximate η as a continuous parameter. Specifically, we use scale factors of: unity, $2^{-0.25}$, $2^{-0.5}$, and $2^{-0.75}$. For each scale factor, we create a separate steerable pyramid for feature extraction. Sampling at a finer scale can be accomplished at the cost of memory and computation time. Hereafter, the detected scale feature is treated as a continuous quantity.

4.1.2 Likelihood Model

Because of the decomposition presented in the previous section, we treat each feature as statistically independent. The model was specifically designed to represent a texture with one dominant orientation. Later, it is shown empirically that the model is able to represent other textures as well. However, the extracted angle and scale are only robust when the texture contains one dominant orientation.

If a texture is not strongly oriented or contains multiple dominant orientations, the angular energy changes drastically. Instead of containing one very dominant peak as shown in Figure 4-1, the energy will be much more flat. When noise is present in this flat case, the location of maximum angular energy (i.e. the orientation θ) can easily be corrupted. Subsequently, the measured scale, eta, can also be corrupted by a small amount of noise.

To address this issue, we introduce an auxiliary Bernoulli random variable, T, that indicates whether a texture is strongly oriented. Recall that one of the features, ϵ, is called the residual energy. The ratio, $\frac{E}{\epsilon}$ roughly captures the peakiness of the angular energy curve (i.e. higher ratios correspond to a more pronounced peak and smaller ratios correspond to a flatter curve). We define the PMF of T to be a function of this ratio, $\frac{E}{\epsilon}$. The function is empirically chosen to have a fairly sharp transition when the ratio is approximately 15. The specific function chosen is

$$p_T\left(1|\epsilon, E\right) = \max\left\{1, \frac{1}{3}\left(\tan^{-1}\left(-100\left(\frac{\epsilon}{E} - \frac{1}{15}\right)\right) + \frac{\pi}{2}\right)\right\} \tag{4.7}$$

and is displayed in Figure 4-2, though we note that any similar function would suffice. The residual energy only plays a role in calculating the probability of the texture being strongly oriented. If the texture is not strongly oriented, we model the scale and orientation as being drawn from uniform distributions to represent the uncertainty of the measurements. The likelihood of a given pixel i

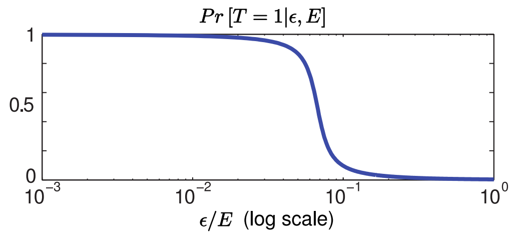


Figure 4-2: Probability of a texture being strongly oriented as a function of $\frac{E}{\epsilon}$.

conditioned on being in region R^l is then

$$\begin{aligned}&p\left(E_i, \mu_i, \eta_i, \theta_i | i \in R^l\right)\\&= p_E^l\left(E_i\right) p_\mu^l\left(\mu_i\right) \sum_{t\in\{0,1\}} p_{\eta|T}^l\left(\eta_i | T_i = t\right) p_T\left(t|\epsilon_i, E_i\right) \sum_{t\in\{0,1\}} p_{\theta|T}^l\left(\theta_i | T_i = t\right) p_T\left(t|\epsilon_i, E_i\right),\end{aligned} \tag{4.8}$$

where the distributions $p_E^l(\cdot)$, $p_\mu^l(\cdot)$, $p_{\eta|T}^l(\cdot|T=1)$, and $p_{\theta|T}^l(\cdot|T=1)$ are estimated using a KDE from the pixels in R^l. All of these nonparametric estimates can use the Fast Gauss Transform [20] to be computed efficiently.

Periodic KDE

The orientation of the texture, represented by θ, is periodic with period π. Consequently, the estimated distribution of θ should also be periodic. A slight modification to the typical KDE is needed to compute a periodic PDF. In the estimate, the periodic PDF should be estimated at M target points (equally spaced in the range $[0,\pi)$) from the set of N source points, S. Each source value in S, denoted by θ_s is confined to the range $[0,\pi)$. If we assume that the kernel bandwidth is small enough such that the contribution of a source point two periods away ($\pm 2\pi$) is negligible, then there are two straightforward methods to estimate a periodic KDE.

The first method is to replicate every source point twice: once with a shift by a positive period $+\pi$, and once with a shift by a negative period $-\pi$. Thus, the new set of source points has cardinality $3N$, and consists of the set of points $\{S, S+\pi, S-\pi\}$. The KDE is still estimated at the same M target points. Because of the shifted sets of source points, an approximately periodic distribution will exist in the range $[0,\pi)$. The periodic KDE in this case is

$$p_\theta(\theta) = \frac{\beta}{3Nh} \sum_{s=1}^{N} \left(K^G\left(\frac{\theta-\theta_s}{h}\right) + K^G\left(\frac{\theta-(\theta_s+\pi)}{h}\right) + K^G\left(\frac{\theta-(\theta_s-\pi)}{h}\right)\right), \tag{4.9}$$

where β is a scale factor to make p_θ a valid distribution.

Similarly, one can replicate every target point twice: once with a shift by a positive period $+\pi$, and once with a shift by a negative period $-\pi$. In this case, the source points do not change, but the $3M$ target points now span the range $[-\pi, 2\pi)$. We refer to this new KDE as $\tilde{p_\theta}$. Once $\tilde{p_\theta}$ is estimated, to obtain the periodic distribution at a specific target point, θ, one combines the new KDE at $\tilde{p_\theta}(\theta)$, $\tilde{p_\theta}(\theta+\pi)$, and $\tilde{p_\theta}(\theta-\pi)$. This also results in an approximately periodic distribution.

The periodic KDE in this case is

$$p_\theta(\theta) = \beta \left[\tilde{p_\theta}(\theta) + \tilde{p_\theta}(\theta + \pi) + \tilde{p_\theta}(\theta - \pi)\right], \quad \theta \in [0, \pi), \tag{4.10}$$

where β is a scale factor to make p_θ a valid distribution and $\tilde{p_\theta}$ is estimated as follows

$$\tilde{p_\theta}(\theta) = \frac{1}{Nh} \sum_{s=1}^{N} \left(K^G \left(\frac{\theta - \theta_s}{h} \right) \right), \quad \theta \in [-\pi, 2\pi). \tag{4.11}$$

Each non-periodic KDE is estimated using the Fast Gauss Transform [20], which has an approximate complexity of $\mathcal{O}(M + N)$, where M is the number of target points and N is the number of source points. The method of replicating the source points has an approximate complexity of $\mathcal{O}(M+3N)$, whereas the method of replicating the target points has an approximate complexity of $\mathcal{O}(3M + 3N)$. Either of these methods can be used; however, because the number of target points is typically much less than the number of source points, the method of replicating target points is used in this thesis.

4.1.3 Verification on Brodatz Images

We apply our feature extraction to the Brodatz textures [5]. Using the first thirteen images from [49], we attempt to classify the images based on our feature set to validate our model. The original images are shown in Figure 4-3. Each Brodatz image is 512-by-512 pixels. For each image, we use the top-left 256-by-256 corner to train the feature distributions, and the top-right 256-by-256 corner as the test data. Classification results were 100% correct for all tests, showing that our features are able to distinguish these different textures. This result does not mean that our representation is able to categorize all natural textures perfectly. However, the results on a small and widely used database of textures does encourage the validity of the model. For each of these cases, we estimate the mutual information between the likelihood of a pixel under our feature set with the label of that pixel. This number gives us a confidence measure, where larger numbers correspond to a more confident classification. Table 4.1 shows the most likely incorrect classification (i.e. the most similar texture with the lowest confidence measure) and the most unlikely incorrect classification (i.e. the most different texture with the highest confidence measure).

In addition to classification of the Brodatz textures, we also segment a few synthetic Brodatz images using our feature set. We overlayed the most similar and most different textures for each of the thirteen textures and show the results of the segmentation for mutual information, balanced J divergence, and J divergence (using $\alpha_I = \frac{1}{8}\alpha_{J_B} = \frac{1}{16}\alpha_J = 1$) in Figure 4-4. It is important to note that we are performing a very crude segmentation where we blindly measure the features at every pixel, and use these as the segmentation features without any further consideration. It will later be shown that a better segmentation algorithm can be achieved. These results support our intuition that any Ali-Silvey distance could be a suitable segmentation criterion. It is interesting to note that the most similar textures portrayed in Table 4.1 are not necessarily the hardest to segment. Segmentation is a much harder problem than classification. In addition to having to learn the statistics during the process of segmentation, boundary effects due to using a local measure will also change the result.

The majority of segmentation results using the three different measures in Figure 4-4 are very

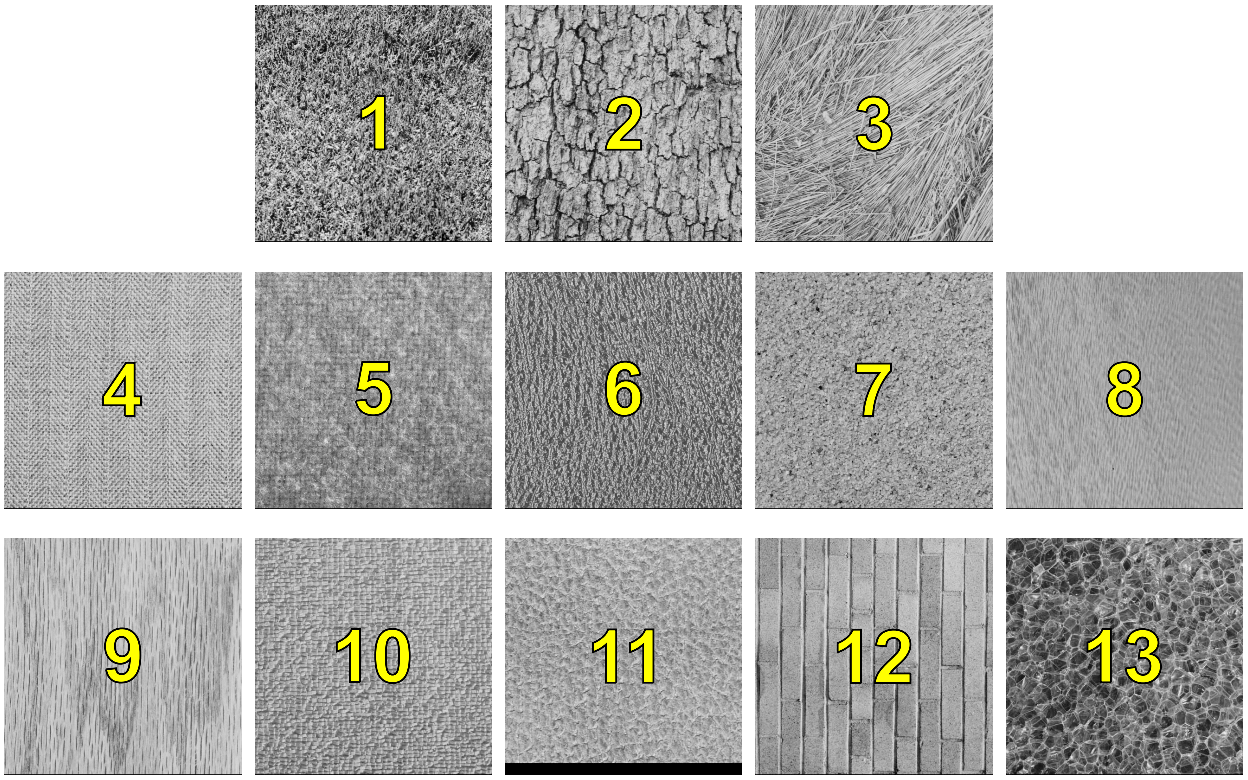


Figure 4-3: Brodatz textures used in classification. The yellow number is the label assigned to it.

Table 4.1: Confidence of Brodatz Classification

Texture $(\mathcal{T}_1)$	Most Similar Texture $(\mathcal{T}_2)$	Most Different Texture $(\mathcal{T}_3)$	$I(X;L)$ $L \in \{\mathcal{T}_1, \mathcal{T}_2\}$	$I(X;L)$ $L \in \{\mathcal{T}_1, \mathcal{T}_3\}$
1	2	8	0.264	0.688
2	7	8	0.237	0.687
3	7	8	0.281	0.675
4	7	8	0.182	0.691
5	10	1	0.269	0.657
6	13	8	0.055	0.686
7	4	8	0.182	0.688
8	9	4	0.465	0.691
9	12	1	0.301	0.676
10	11	8	0.184	0.658
11	10	1	0.184	0.672
12	9	1	0.301	0.599
13	6	8	0.055	0.682

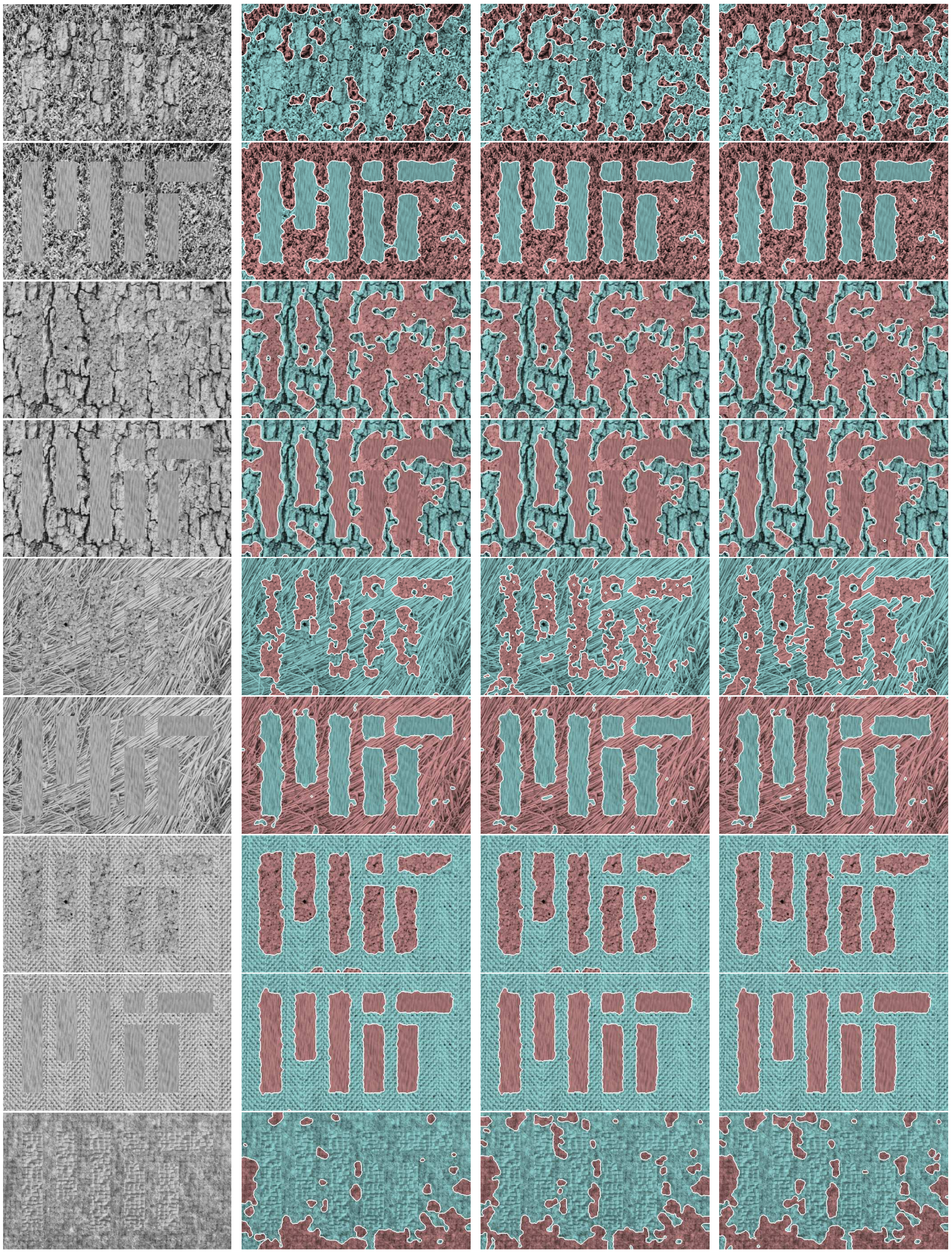


Figure 4-4: Segmentation using three different measures on synthetic Brodatz images. First column is the original image, second column is obtained using MI, third column is obtained using balanced J divergence, and fourth column is obtained using J divergence.

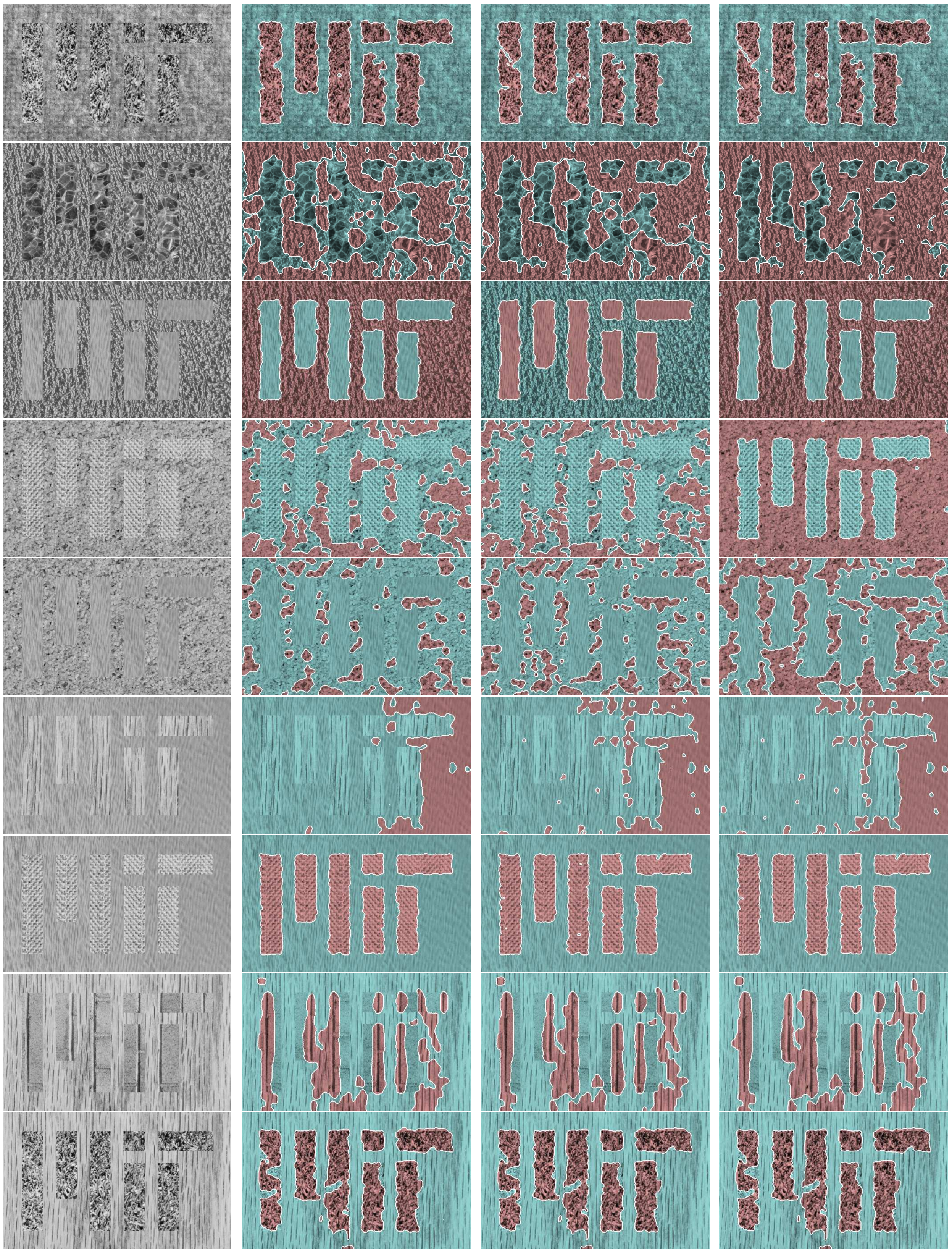


Figure 4-4: Segmentation using three different measures on synthetic Brodatz images. First column is the original image, second column is obtained using MI, third column is obtained using balanced J divergence, and fourth column is obtained using J divergence.

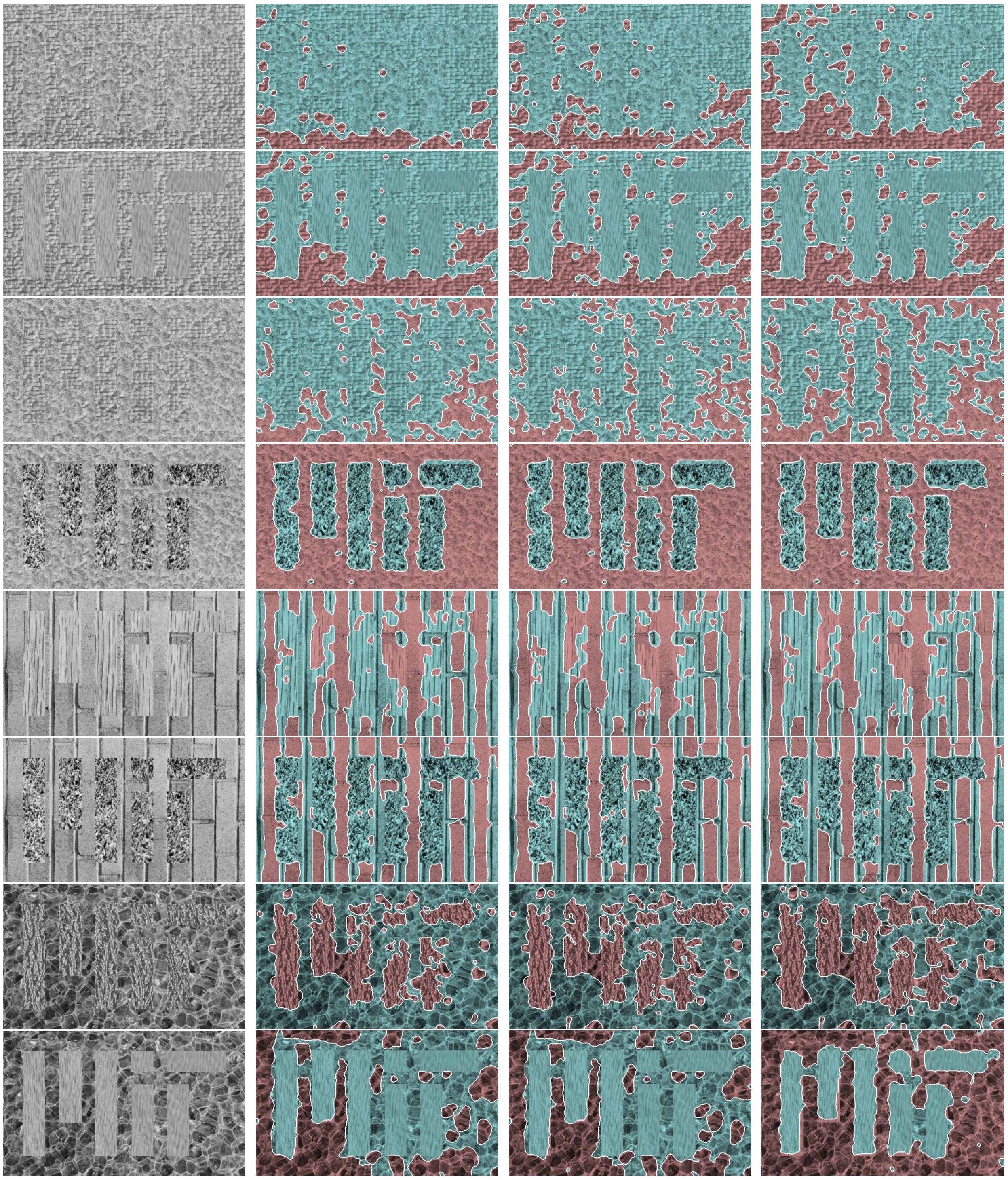


Figure 4-4: Segmentation using three different measures on synthetic Brodatz images. First column is the original image, second column is obtained using MI, third column is obtained using balanced J divergence, and fourth column is obtained using J divergence.

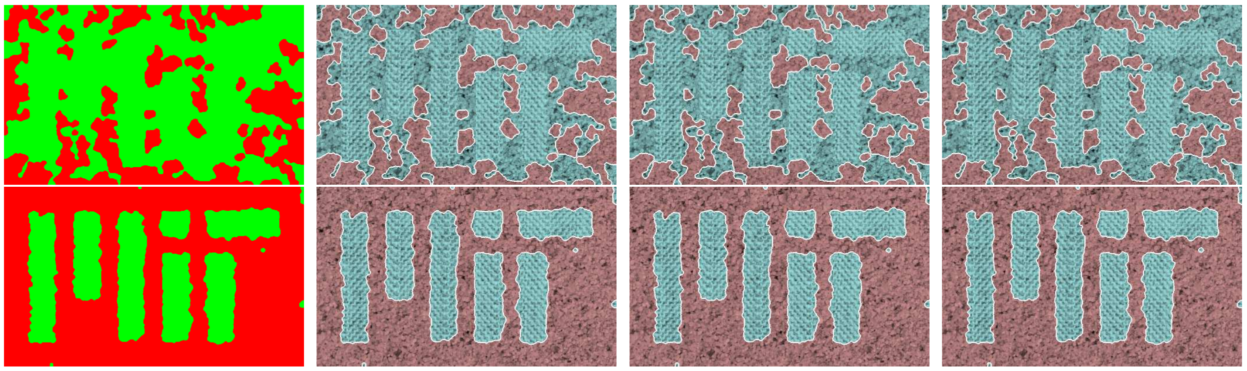


Figure 4-5: Segmentation using three different measures on a synthetic Brodatz images. First column is the the initialization used, second column is obtained using MI, third column is obtained using balanced J divergence, and fourth column is obtained using J divergence. The first row's initialization was the original segmentation result using MI and a gridded initialization. The second row's initialization was the original segmentation result using J divergence and the same gridded initialization.

Table 4.2: Distances of local extrema

Initialization	MI	Balanced J Divergence	J Divergence
MI's segmentation	0.262	1.346	2.446
J Divergence's segmentation	0.423	2.132	4.620

similar. There are a few surprising cases where J divergence seems to perform much better than both mutual information and balanced J divergence. We examine one of these cases (fourth row of the second page of Figure 4-4) in more detail. It is interesting to consider how each measure performs when initialized to the segmentation of another algorithm. Because the segmentations obtained using MI and balanced J divergence are comparable, we initialize each algorithm to the MI segmentation and the J divergence segmentation. Resulting segmentations with these initializations are shown in Figure 4-5. These segmentations show that given either initialization, any of the distance measures will not drastically perturb the curve. In other words, this means that each of the original segmentations is very close to a local extremum under any of the distance measures. We can evaluate which local extremum is a better segmentation under each distance criterion by evaluating the actual distance. Table 4.2 shows these values. The distances obtained when initializing to the J divergence segmentation are greater than those obtained when initializing to the mutual information segmentation under all three distance measures. Though we found that both the original MI and J divergence segmentations are local extremum, these values show that the mutual information segmentation was only a local extremum, whereas the one obtained using J divergence may be the global extremum. We believe our original initialization (a uniformly spaced grid of seeds) may have contributed to the result of reaching a local extremum instead of the global extremum in the MI and balanced J divergence cases.

We proceed to verify this claim by generating a set of random initializations and segmenting based on those seeds. We generate a random initialization by first randomly picking locations of seeds. Each pixel has approximately 0.1% chance of being the center of a seed. We then draw

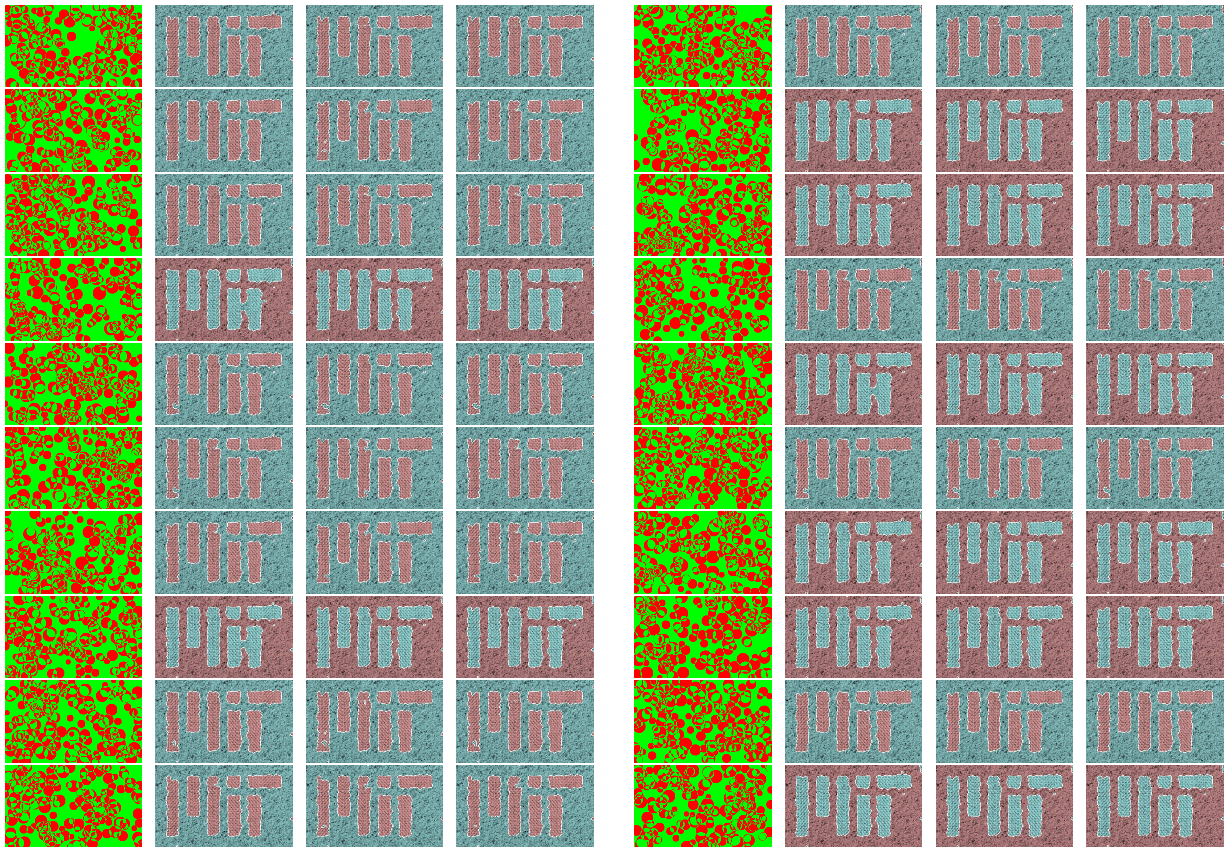


Figure 4-6: Segmentation using three different measures on a synthetic Brodatz images. For each set of four images, the first shows the random initialization used, the second shows the result obtained using MI, the third shows the result obtained using balanced J divergence, and the fourth shows the result obtained using J divergence.

a circle around that seed having a random radius within some fixed range. When overlapping circles occur, we take the exclusive-or of the elements. Figure 4-6 shows the results for twenty random initializations. Clearly, the results are comparable in these random initialization cases. The erroneous result we obtained for this image in Figure 4-4 seems to be an error that occurred only because of the specific initialization we used. To explore even further, we show the best segmentation results obtained using this random initialization scheme in Figure 4-7. The “best” segmentation is chosen by finding the result among twenty different initializations that has the maximum distance (i.e. the ground truth is never used).

In most cases, the non-gridded initialization improves the algorithm regardless of the distance measure. We compare our original gridded initialization results with our non-gridded initialization results by looking at the probability of error statistics as compared to the ground truth. Although these empirical probabilities are computed with access to the ground truth, it is important to keep in mind that none of the actual segmentation results considered knowing the ground truth. These results are shown in Figure 4-8. This figure shows that using the best of eight non-gridded initializations generally performs better than using one gridded initialization. Additionally, a scatter plot of the probability of errors using the two initialization methods is shown in Figure 4-9. Because

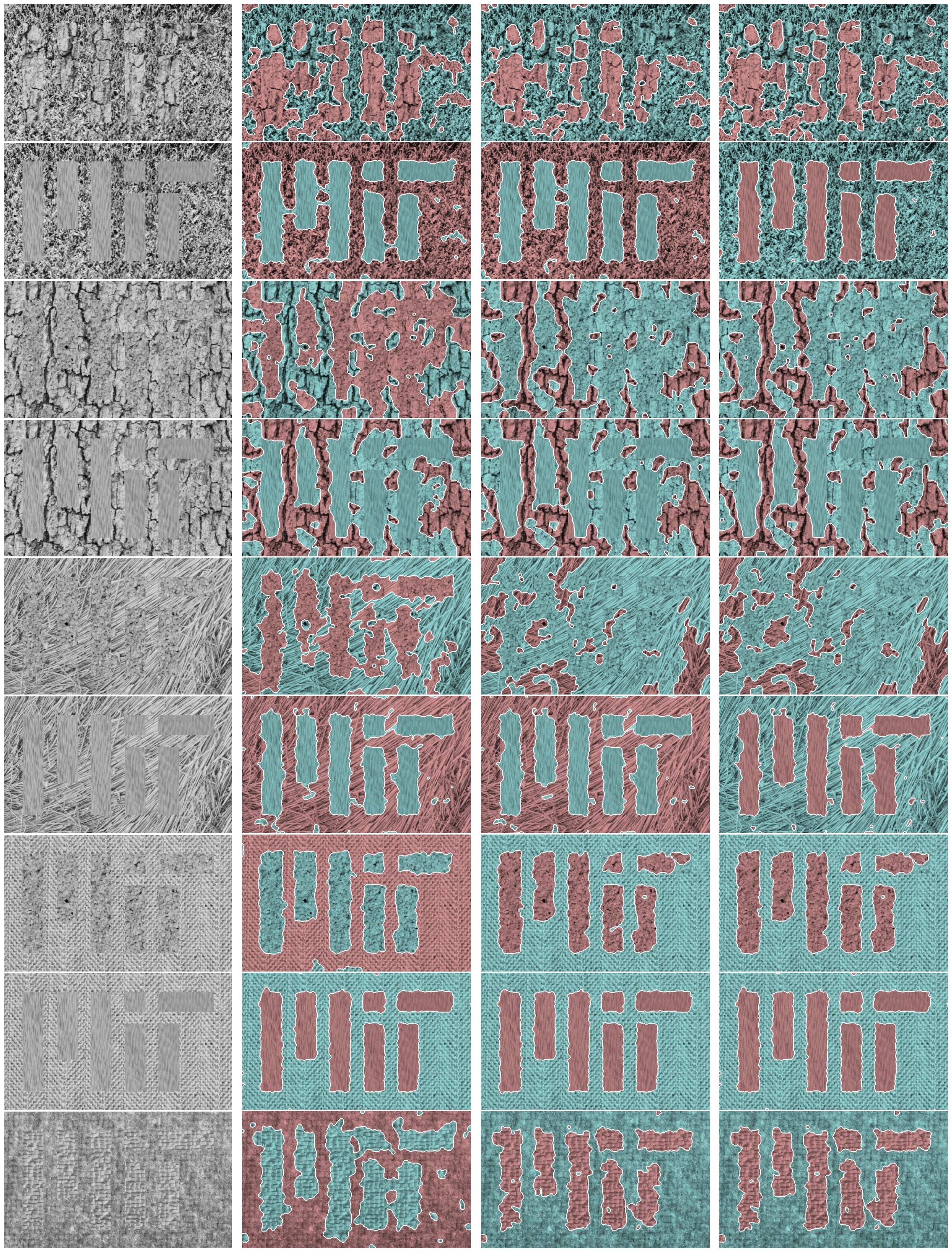


Figure 4-7: Segmentation using three different measures on synthetic Brodatz images. First column is the original image, second column is obtained using MI, third column is obtained using balanced J divergence, and fourth column is obtained using J divergence. Results obtained using a non-gridded random initialization.

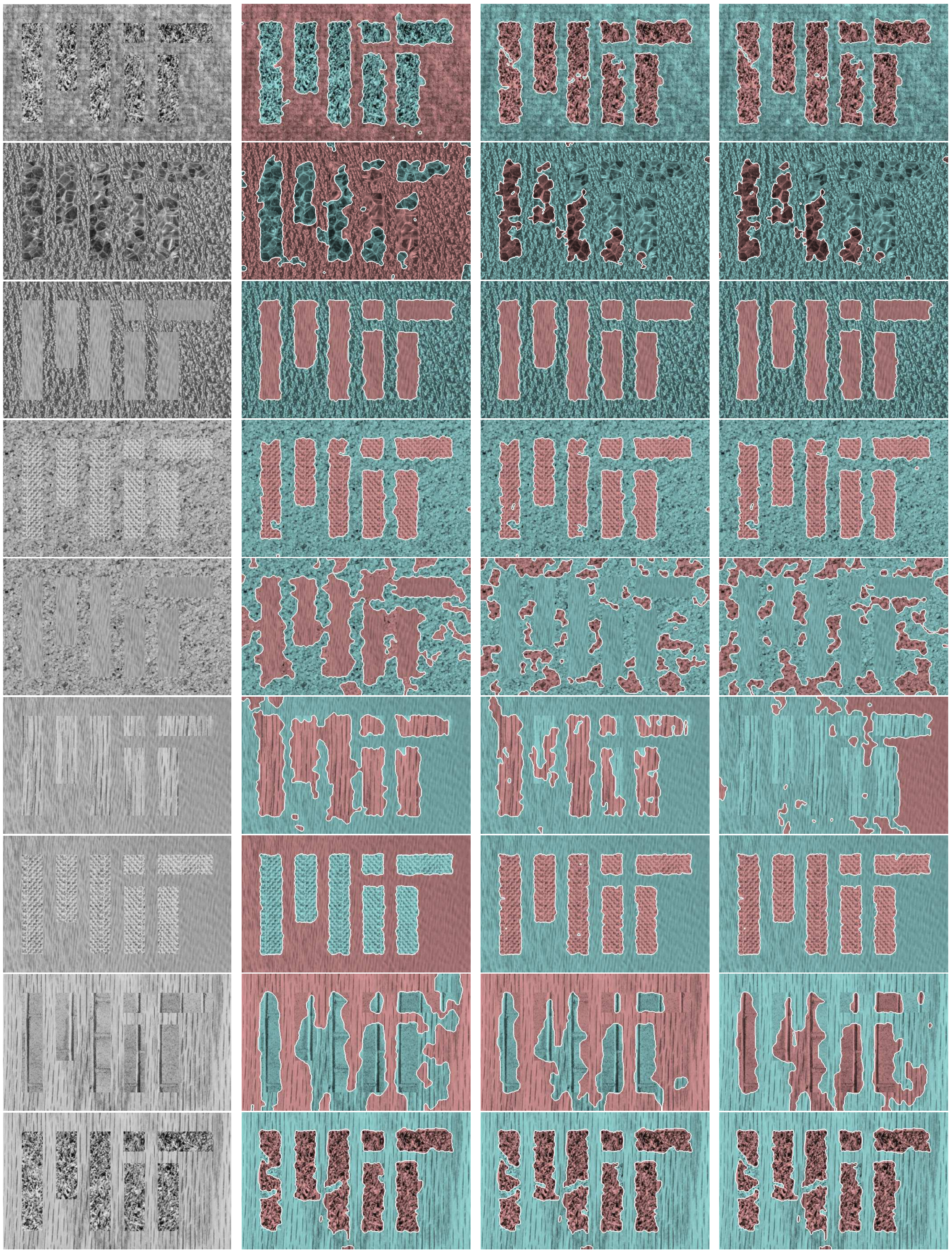


Figure 4-7: Segmentation using three different measures on synthetic Brodatz images. First column is the original image, second column is obtained using MI, third column is obtained using balanced J divergence, and fourth column is obtained using J divergence. Results obtained using a non-gridded random initialization.

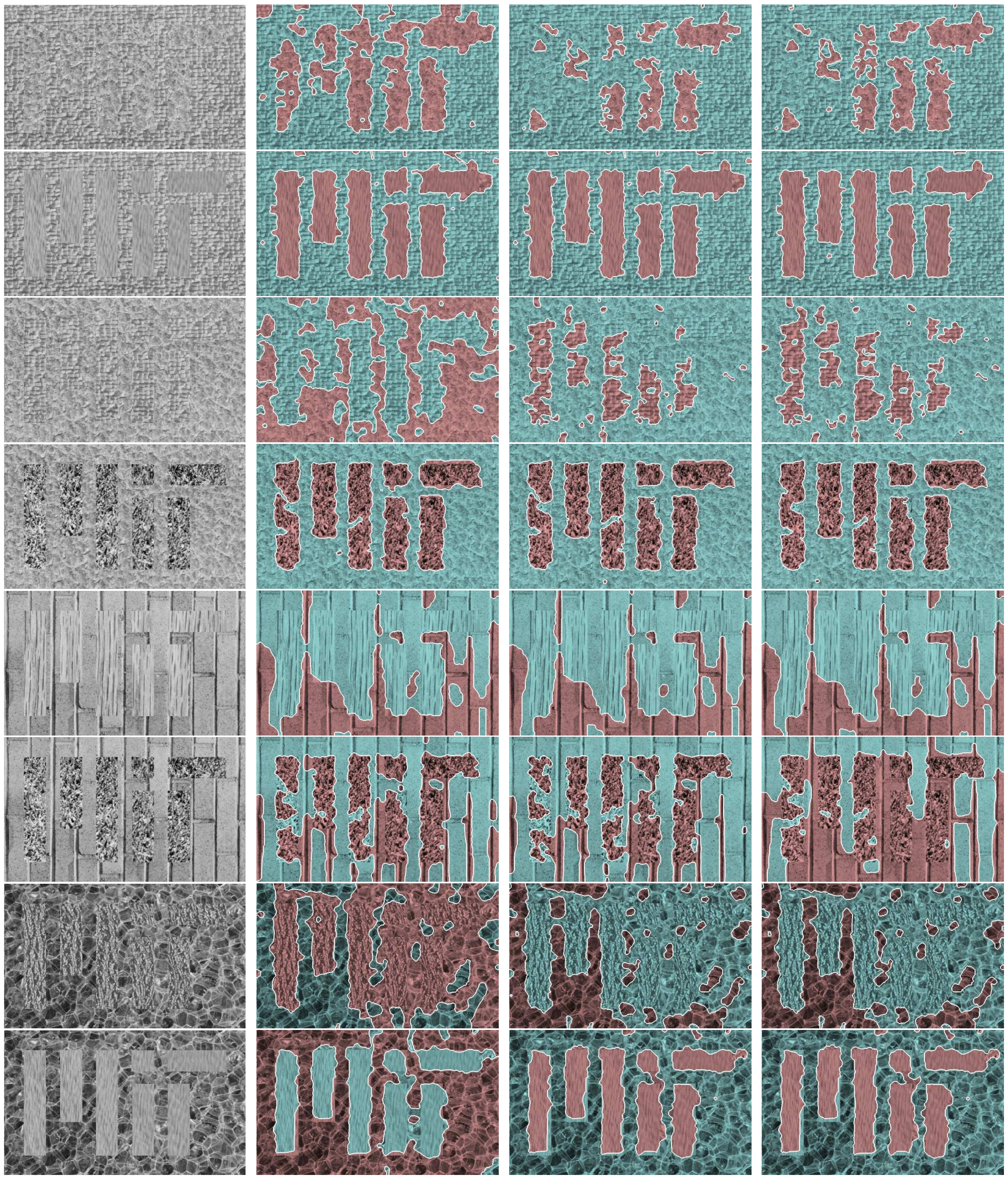


Figure 4-7: Segmentation using three different measures on synthetic Brodatz images. First column is the original image, second column is obtained using MI, third column is obtained using balanced J divergence, and fourth column is obtained using J divergence. Results obtained using a non-gridded random initialization.

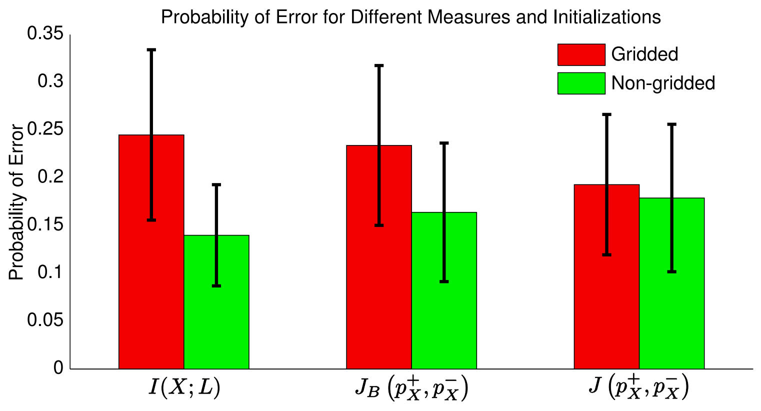


Figure 4-8: Plot of labeling error for three measures using the gridded and non-gridded initializations. The mean of the labeling errors for all synthetic Brodatz images is plotted with the standard deviation of the errors shown.

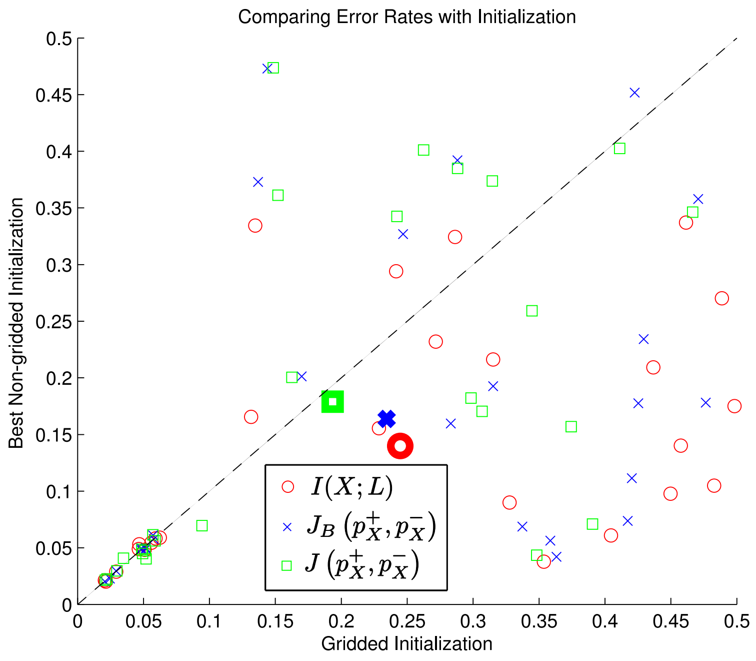


Figure 4-9: Scatter plot of labeling error for three measures using the gridded and non-gridded initializations. The y-axis represents the probability of error for the best of eight non-gridded initializations and the x-axis represents the probability of error for the gridded initialization. The thick markers represent the mean of the associated information measure.

most points lie below the diagonal line, this plot also shows that the non-gridded method generally performs better.

When using gradient ascent to find the maximum, we conjectured that when searching the energy manifold, a particular initialization may converge to a local maximum instead of the global. The idea of using multiple non-gridded initializations was to overcome the problem of local extrema. As expected, Figures 4-8 and 4-9 indicate that the multiple initializations have helped in some cases. The results also indicate that using only eight initializations may not be enough to globally optimize the complex energy manifold, and more initializations may benefit in the final segmentation. Regardless, this section shows that choosing the best energy among multiple segmentations typically leads to improved results.

Similar to Chapter 3, we have shown that in most cases, the three Ali-Silvey distances produce very similar results. At times, the specific initialization can produce a different local extremum, but with the use of many random initializations, we are much more likely to find the global extremum. As stated previously, in most of our applications henceforth, we will choose to use multiple non-gridded initializations with mutual information as the distance measure for its more simple gradient ascent velocity.

4.1.4 Smoothness Priors on Extracted Features

Having defined and verified our feature set, we now discuss how to capture spatial correlations for each feature. By capturing these correlations, we are able to model how the texture changes. We take each measured feature as the output of a smooth Markov Random Field (MRF) plus additive noise conditioned on the segmentation label. The model for each of these random fields is similar modulo differing degrees of assumed smoothness. We derive the inference procedure for the orientation field only; the derivation for scale, contrast, and bias follow accordingly. One notable departure from standard MRF methods is that we model the additive noise term in each field nonparametrically. This is due to the fact that the "noise" terms correspond to quantities of interest. Effectively, smoothness is utilized as a means to separate the contributing factors to the observed feature. We also derive a novel fixed point update utilizing the nonparametric model.

Under a standard Gaussian MRF model of smooth variation one assumes the observation model shown in Figure 4-10a, where $\underline{n}$ is typically Gaussian and $\underline{\hat{\phi}}$ is a smooth MRF. However, in the types of images we consider, the fields within a commonly labeled region may be locally smooth while exhibiting a small number of abrupt changes. One way to account for this is to decompose $\underline{\hat{\phi}}$ into a completely smooth component, $\underline{\tilde{\phi}}$ and a piecewise constant component, $\underline{C}$, as depicted in Figure 4-10b. Under the assumption that the abrupt changes in $\underline{C}$ are large relative to the standard deviation of $\underline{n}$, this is equivalent to an MRF where the additive noise term comprised of both $\underline{C}$ and $\underline{n}$ is equivalent to a mixture of Gaussians or, more generally, a kernel density estimate (KDE).

In total we infer four different smooth fields: an orientation field on $\underline{\theta}$, a scale field on $\underline{\eta}$, a gain field on $\underline{E}$, and a bias field on $\underline{\mu}$. It is important to note that the piecewise constant component of the noise, $\underline{C}$, captures relevant information about the intrinsic properties of the observed image. This is most apparent when looking at the estimated smooth gain and bias fields whose specific details are covered later. These two fields aim to capture a slowly changing contrast and bias caused by lighting effects. The piecewise constant field $\underline{C}$ is an image with a uniformly contrasted and biased texture. Consequently, the observed image after removing the effects of the gain and bias field (i.e. $\underline{C} + \underline{n}$) is essentially a noisy approximation of the intrinsic reflectance image. Likewise, the

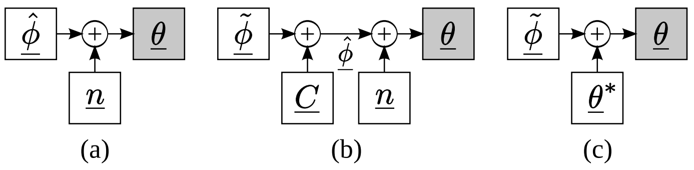


Figure 4-10: Smooth Field Model: (a) $\hat{\underline{\phi}}$ is smooth with abrupt changes; (b) $\tilde{\underline{\phi}}$ is completely smooth; (c) Intrinsic Image, $\underline{\theta}^*$

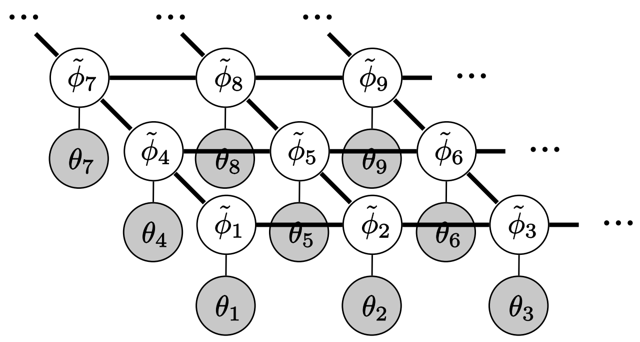


Figure 4-11: Graphical model for the MRF estimation of orientation. Each θ_i is measured directly from the observed image and $\tilde{\phi}_i$ is estimated through the fixed-point iteration.

residual terms associated with the orientation and scale fields provide what we call the intrinsic texture image. Accounting for all of the smooth fields gives us an image with a uniform texture oriented in the same direction and with constant scale.

Ultimately, we iterate between estimation of these fields conditioned on a given segmentation and segmentation based upon the estimated intrinsic texture images. Figure 4-10c depicts the observation model incorporating the intrinsic unoriented texture image, $\underline{\theta}^*$ where the superscript, $*$, denotes the intrinsic feature.

We formulate MAP estimation of the various fields in a similar fashion to [51]. The primary difference is the incorporation of nonparametric noise model. Furthermore, as a consequence of the nonparametric model, we derive a fixed point iteration in the absence of an analytic solution. A key assumption is that each θ_i is *i.i.d.* conditioned on ϕ_i. This relationship is depicted in the graphical model of Figure 4-11. A full derivation is given in the appendix (Section B.1). Though this derivation follows from the smooth field only affecting one parameter, we also show the derivation and the resulting fixed-point update for a field affecting multiple independent parameters in the appendix (Section B.3). We assume that $\tilde{\underline{\phi}} \backsim \mathcal{N}\left(\underline{0}, \Lambda_\phi\right)$ and $\tilde{\underline{\theta}} \backsim \frac{1}{Nh}\sum_{i=1}^{N} K^G\left(\frac{\theta-\theta_I}{h}\right)$.

$$\tilde{\underline{\phi}} = \arg\max_{\underline{\phi}} P\left(\underline{\phi}|\underline{\theta}\right) \tag{4.12}$$

Using Bayes rule and differentiating w.r.t. $\underline{\phi}$, we arrive at the following equality

$$\underline{\theta} - \tilde{\underline{\phi}} - w_p^\theta\left(\underline{\theta} - \tilde{\underline{\phi}}\right) - \frac{h^2}{2}\Lambda_\phi^{-1}\tilde{\underline{\phi}} = 0, \tag{4.13}$$

where h is the bandwidth used in the KDE, and $w_p^\theta(\cdot)$ is the ratio of a weighted KDE to the KDE defined as follows

$$w_p^\theta(\cdot) = \frac{\sum_{s=1}^N \left(\theta_s - \tilde{\phi}_s\right) e^{-\frac{\left((\cdot)-\theta_s+\tilde{\phi}_s\right)^2}{h^2}}}{\sum_{s=1}^N e^{-\frac{\left((\cdot)-\theta_s+\tilde{\phi}_s\right)^2}{h^2}}}. \tag{4.14}$$

Rearranging this equation results in a fixed-point iteration to solve for $\underline{\tilde{\phi}}$

$$\underline{\tilde{\phi}}^{(k+1)} = F^{-1}\left(\underline{\theta} - w_p^\theta\left(\underline{\theta} - \underline{\tilde{\phi}}^{(k)}\right)\right), \tag{4.15}$$

where the matrix, F, is defined as follows

$$F = \left(\frac{2}{h^2}\Lambda_\phi\right)^{-1} + I = F_1^{-1} + I. \tag{4.16}$$

As in [51], by treating Λ_ϕ as the result of filtered i.i.d. noise, the fixed-point update can be computing efficiently using FFT methods. Consequently, we write the matrix, F_1, in Equation 4.16 as follows

$$F_1 = \frac{2}{h^2}\Lambda_\phi = \frac{2}{h^2}H\sigma_\phi^2 I H^T = \frac{2\sigma_\phi^2}{h^2}HH^T, \tag{4.17}$$

where H is the matrix that performs a convolution with a unity DC gain lowpass filter and σ_ϕ^2 is the variance of ϕ_i. This implies that multiplying by the covariance matrix, Λ_ϕ, is equivalent to applying two lowpass filters. In general, the bandwidth used to estimate the PDF, h, is very small because it is chosen to be inversely proportional to $N^{1/5}$. With this assumption, the matrix F^{-1} equivalently performs a lowpass filtering operation with approximately unity DC gain. This will be shown here.

By construction, H (or equivalently H^T) is the matrix that performs a lowpass filtering operation with unity DC gain. Thus HH^T must equivalently perform a lowpass filtering operation (e.g. Figure 4-12a). If we assume that $h << \sigma_\phi^2$, then the matrix F_1 performs a lowpass filtering operation with a very large DC gain (e.g. Figure 4-12b). The inverse matrix, F_1^{-1} must then perform a highpass filter operation with a very small DC gain (e.g. Figure 4-12c). The identity matrix, I does not alter the signal and consequently must be an all pass filter (e.g. Figure 4-12d).

Now we consider the matrix F in Equation 4.16. Summing the two matrices F_1^{-1} and I is equivalent to summing their frequency responses. Thus, the filter F must perform a highpass filter operation with near unity DC gain (e.g. Figure 4-12e). When we invert this operation, the resultant F^{-1} must then perform a lowpass filter operation also with approximately unity DC gain (e.g. Figure 4-12f). This shows that the fixed-point update of 4.15 can be efficiently computed using a unity DC gain lowpass filter.

While each of the separate smooth fields is modeled similarly, there are differences between them which we briefly explain now. For the orientation feature, $\underline{\theta}$, we assume there is an underlying smooth, additive orientation field, $\underline{\phi}$. We cannot use a conventional lowpass filter in (4.15) because the orientation is periodic (with period π). Instead, we double each angle and convert it into the sine and cosine components (i.e. mapping it onto the unit circle). We lowpass these X and Y coordinates, take the inverse tangent, and then halve the angle to find the equivalent lowpassed

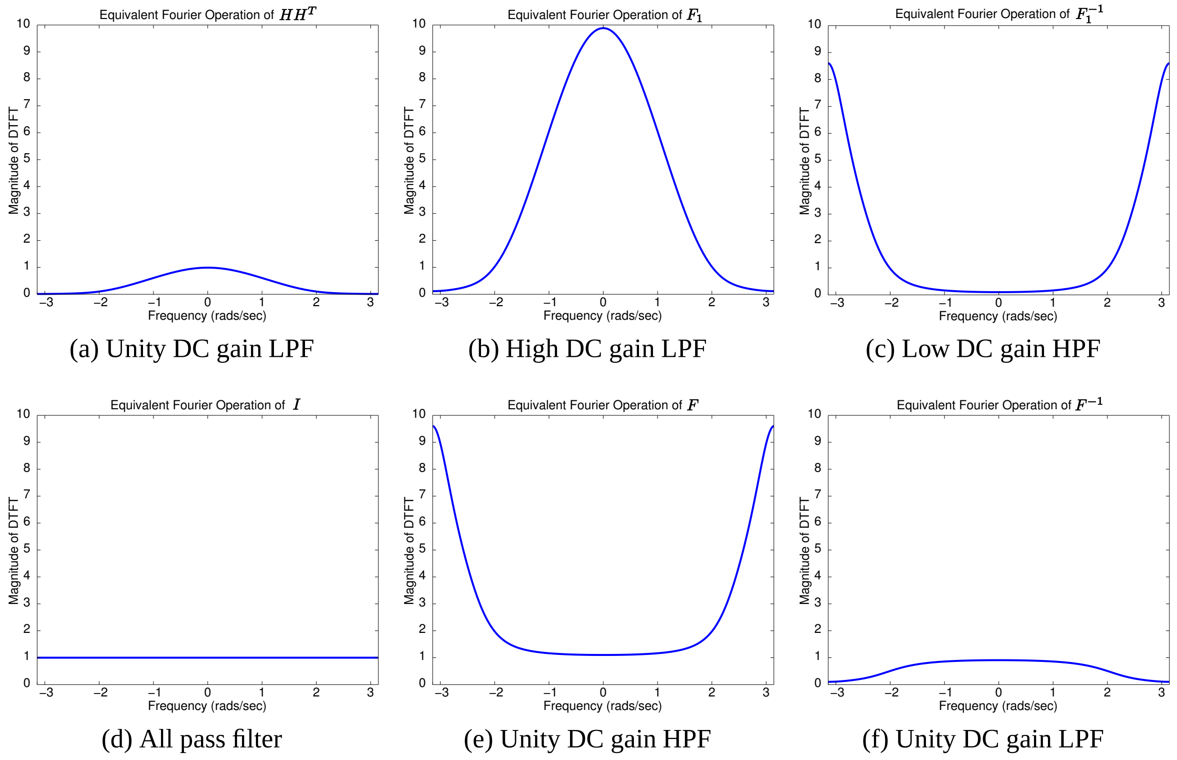

(a) Unity DC gain LPF (b) High DC gain LPF (c) Low DC gain HPF (d) All pass filter (e) Unity DC gain HPF (f) Unity DC gain LPF

Figure 4-12: Equivalent Fourier operations to matrices used in fixed-point update

angle. To find the intrinsic angle, we simply subtract the smooth field from our measurement:

$$\theta_i^* = \theta_i - \phi_i. \tag{4.18}$$

As mentioned, we treat the discrete scale measurement as a continuous quantity. The bandwidth of the KDE is chosen to be large enough to sufficiently smooth over adjacent scales. The additive term is denoted by the symbol $\underline{\nu}$ and the intrinsic scale is computed as

$$\eta_i^* = \eta_i - \nu_i. \tag{4.19}$$

The treatment of the remaining fields is somewhat more complex. The gain field $\underline{g}$, is a multiplicative field accounting for contrast changes. The bias field, $\underline{b}$, is an additive field that corrects the low-frequency component of the texture. We impose smoothness on the log of the gain field allowing us to treat it as an additive field. Having accounted for these fields, we are left with an estimate of the intrinsic reflectance image, $\underline{\mathcal{R}}$. Figure 5-11b depicts this relationship.

In contrast to the orientation and scale fields, the gain and bias fields both operate directly on the pixel intensities. To find the reflectance image, we divide our observed image by the gain field and then subtract the bias field. As the steerable pyramid is comprised of linear filters, dividing the image by a value implies that the filter outputs are divided by the same value. Thus, both the energy and bias features of our texture are affected by the gain field. The contrast energy captures the filter response at some bandpass frequency, whereas the bias captures low frequency components. Consequently, adding a constant value should only change the low frequency bias and not the contrast energy. If we assume that the bias field can correct for any changes that the

Table 4.3: Empirically Chosen Smoothness Constants

Field	h_{min}	h_{scale}
Orientation Field ($\underline{\phi}$)	0.8	2
Scale Field ($\underline{\nu}$)	0.2	2
Gain Field ($\underline{\mathcal{G}}$)	-	2
Bias Field ($\underline{\mathcal{B}}$)	-	4

gain field has made to the bias term, then we can estimate these two fields independently. The gain field is first estimated assuming that it only affects the contrast energy, and then the bias field is estimated only based on the bias term. If we assume that these fields are sufficiently smooth within a small neighborhood, then we can approximate the effect of them on the intrinsic features as follows

$$E_i^* = \frac{1}{|R_i^{\eta_i}|} \sum_{j \in R_i^{\eta_i}} \left| \frac{y_j\left(\theta_i^{\eta_i}\right)}{g_j} \right|^2 \approx \frac{E_i}{g_i^2}, \tag{4.20}$$

$$\mu_i^* = \sum_{j \in R_{f,i}^{\eta_i}} \left(\frac{x_j}{g_j} - b_j \right) \cdot f_j \approx \frac{\mu_i}{g_i} - b_i \sum_{j \in R_{f,i}^{\eta_i}} f_j = \frac{\mu_i}{g_i} - b_i, \tag{4.21}$$

where $R_{f,i}^{\eta_i}$ is the support of the lowpass filter around pixel i and in scale η_i, and f_j is a lowpass filter coefficient. Note that the last equality in (4.21) is by design since our filter has constant DC gain.

The last implementation detail for field estimation is the assumed degree of smoothness. This is determined by the properties of the covariance matrices associated with each field, which is in our case is determined by the structure of the filter used in (4.15). In our experiments, we use a 15x15 averaging filter for each field, noting that the segmentation is not overly sensitive to the window size. Another parameter that we can tune is the bandwidth of our KDE. Larger bandwidths equate to grouping more regions into the same intrinsic texture representation. This is advantageous because it allows us to capture more extreme lighting and geometric effects. However, if the bandwidth is too large, textures that belong to two different intrinsic textures will look the same. We chose a minimum bandwidth for each smooth field and set the scaling factor to the "rule of thumb" bandwidth [41] so that they provided pleasing results, noting that the segmentation is not very sensitive to a fairly large range of these values. Table 4.3 contains the values we used in our final algorithm.

4.2 Boundary Effects

When segmentation of synthetic Brodatz images was done in the previous section, it was noted that the most similar textures were not necessarily the hardest to segment. The main reason for this anomaly is that we have yet to consider boundary effects. In this section, we discuss the different boundary effects that are encountered and our solutions to them. This nuisance surfaces in two different ways: feature extraction and smooth field estimation.

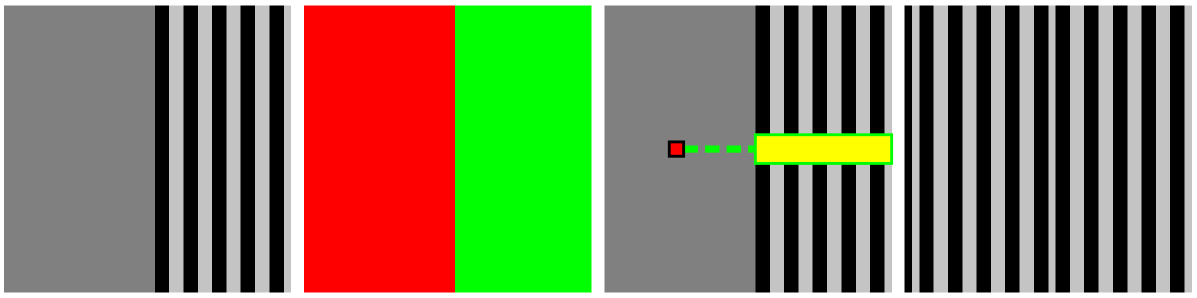

(a) (b) (c) (d)

Figure 4-13: A toy example of extending an object: (a) the original image; (b) the segmentation mask provided for the extension algorithm; (c) the red pixel is where we would like to extend the object to and the yellow region is the texture patch we extend; (d) the resulting extended object.

4.2.1 Boundary Effects with Feature Extraction

Our originally proposed method for feature extraction was to always look at a fixed size local window around each pixel. When segmenting an image, object boundaries indicate places where the texture we measure should abruptly change. When using the original feature extraction method, a local window around a pixel near a boundary will include pixels from two separate objects. This method provides for a smooth change in our features across a boundary that is a mixture of the two different objects. Ideally, we desire to have an abrupt change at an object boundary. Thus, we define a new region, R_i', which is similar to our original region, R_i, but is confined to the segmented region, $R^{\pm}$.

$$R_i' = R_i \cap R^{\pm} \tag{4.22}$$

A more subtle effect from boundaries originates from the actual steerable pyramid output. Each output is due to a 9-by-9 filter. When we convolve the filter with the image, boundaries will again contain mixtures of both objects. To address this problem, we can extend each object past its boundary and calculate the steerable pyramid for the extended object. Note that an extension needs to be performed for each object (e.g. if there are two objects, there will be two resulting extended images). Algorithm 1 gives pseudo-code on how to extend the R^+ region into the R^- region, Figure 4-13 portrays how the extension is performed graphically for one region, and Figure 4-14 shows the extension performed on a natural image. From Figure 4-14 it's clear that this

Algorithm 1 Object Extension

```
1: for each pixel $i$ in $R^-$ that needs to be extended to $R^+$ do
2:     Draw a line from $i$ to the closest point in $R^+$
3:     Repeat the pattern of the line from the object boundary to $i$
4: end for
```

method to reproduce textures seems fairly accurate, at least near boundaries. The center of what used to be the zebra in Figure 4-14b shows some glaring discontinuities. However, since level set velocities for segmentation are typically only concerned with the pixels near the boundary, this error can be tolerated.

The second row in Figure 4-15 shows three of our features for a Brodatz image when the

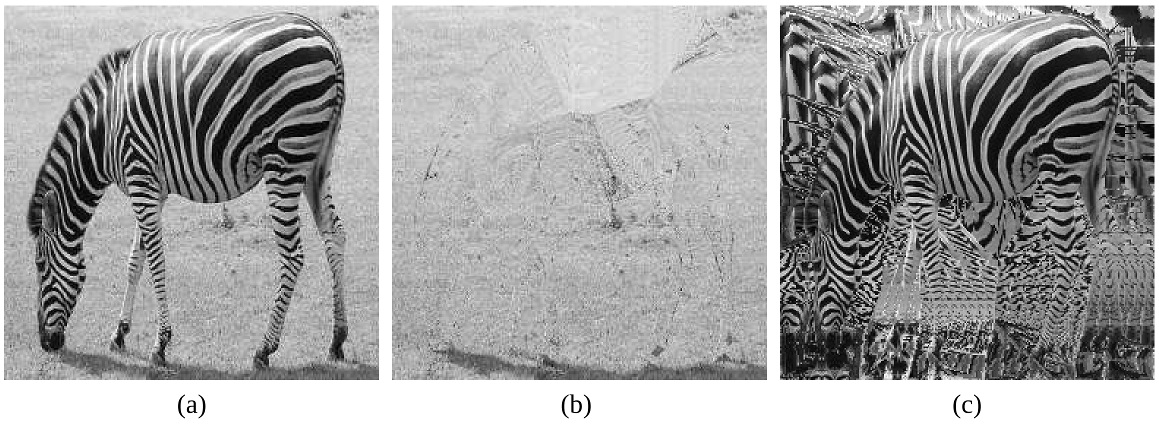


Figure 4-14: An example of extending a natural image (given the correct segmentation): (a) the original image; (b) the first extended image; (c) the second extended image.

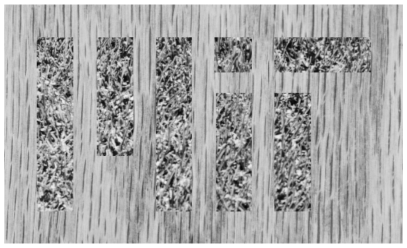
boundary effects are ignored. The third row in Figure 4-15 shows the same three features after our refinement to incorporate boundary effects.

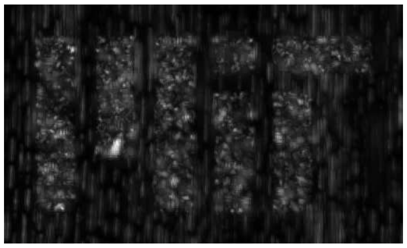
We would like to use the new local region described in Equation 4.22 when extracting our feature set combined with the object extension to eliminate the filter boundary effects. We call the combination of these two solutions to be the refinement of our algorithm. Using these two solutions are problematic for a few reasons. The new region requires the features to be calculated any time the curve changes, or essentially at every iteration in segmentation. In addition to this, the steerable pyramid must also be calculated any time a new extended image is created. These computations can take a considerable amount of time, and is not something that is particularly feasible. Additionally, the refinement creates many more local extremum which is probably attributed to the difficulty of distinguishing an object boundary from an edge within a texture. For these reasons, using this refinement needs to be treated with special care.

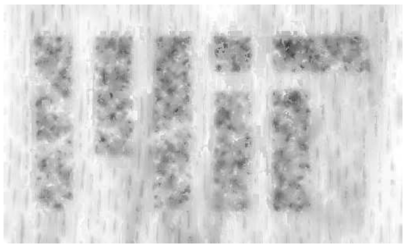
In our segmentation algorithm, our solution to these boundary effects was to first segment based on our original feature extraction (ignoring boundary effects) so that we would reach a local extremum near the global extremum. Then, we refine our segmentation by using the newly defined region in Equation 4.22, extending the object boundaries, and recalculating the steerable pyramid. This prevents the segmentation from falling into a local extremum that is very different than the global, and will decrease the computation time. The result before and after the refinement are shown for a sample image in Figure 4-16.

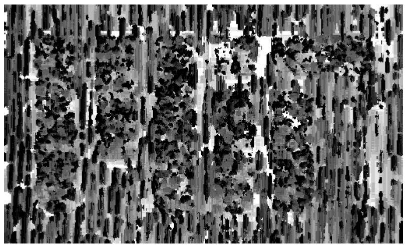
4.2.2 Boundary Effects with Smooth Field Estimation

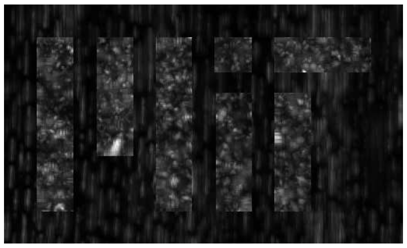
As stated previously, we estimate a smooth field for each feature under each label. The smooth field is only valid for pixels corresponding to the same label, but it is still necessary to define it for other pixels. One reason for this is because when we lowpass the values with the matrix F^{-1} in Equation 4.15, we would like to have values defined everywhere so that this can be done in a faster way. However, the more important reason that the smooth fields need to be defined everywhere is because of the gradient ascent velocity used to update the level sets. We need the smooth field under each label to be well defined for pixels near the boundary so we can evaluate the likelihood of the intrinsic feature. The values far from the border are less important, similar to the boundary

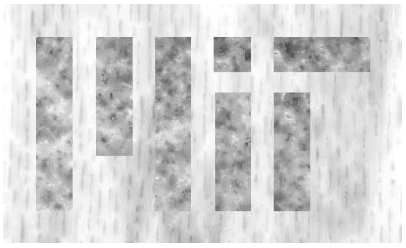

(a) Original image


(b) E ignoring boundary effects


(c) μ ignoring boundary effects


(d) η ignoring boundary effects


(e) E refined


(f) μ refined

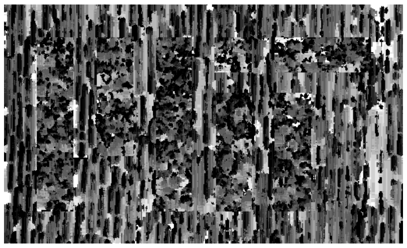

(g) η refined

Figure 4-15: An example of our originally proposed feature extraction with and without considering boundaries. (a) The original image; (b)-(d) features ignoring boundary effects; (e)-(g) features accounting for boundary effects

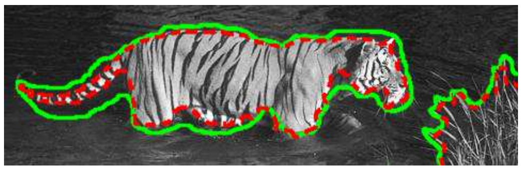


Figure 4-16: Zoomed in portion of segmentation. Green solid line is before refinement, red dotted line is after refinement

effect solution in the feature extraction.

As a result of this issue, we use a similar method as before and extend the smooth field before applying the lowpass filter. The extension is done by simply setting the values for pixels outside of a region to be the same value as the nearest pixel inside the region. Our final algorithm for estimating the smooth orientation field is described with pseudocode in Algorithm 2.

Algorithm 2 Estimating Smooth Orientation Field in R^+

```
1: while Fixed point iteration has not converged do
2:     for each pixel $i \in R^+$ do
3:         Estimate the smooth field as if there was no spatial correlation by
               $\tilde{\phi}_i^{(k+1)} = \underline{\theta} - w_p \left( \underline{\theta} - \underline{\tilde{\phi}}^{(k)} \right)$
4:     end for
5:     for each pixel $i \notin R^+$ do
6:         Extend the smooth field $\tilde{\phi}_i^{(k+1)}$ to be the same value as $\tilde{\phi}_j^{(k+1)}$, where $j$ is the nearest point
           in $R^+$
7:     end for
8:     Finish applying Equation 4.15 by lowpass filtering $\underline{\tilde{\phi}}^{(k+1)}$
9: end while
```

Chapter 5

Texture Modeling Applications and Results

Prior to this chapter, we have discussed various distance measures for segmentation purposes and feature extraction for textures. We have chosen a very explicit model for textures, namely one that tries to capture a strongly oriented texture. Though the development of our model was for a fairly specific type of texture, we show here why our modeling choices have advantages. In addition to being able to segment natural images fairly well, we are also able to extend our texture features to estimate the radiometric response of a camera and estimate intrinsic shading images.

5.1 Segmentation

We begin with image segmentation, which has already been mentioned throughout previous chapters. We use mutual information [26] to segment the image incorporating our estimation of smooth random fields and our solution to the boundary effects. Additionally, we utilize the multi-region segmentation presented in [6]. We assume that the number of regions in the image is known, noting that this quantity could be estimated in various ways. Our final algorithm for image segmentation is summarized in the flowcharts of Figures 5-1 and 5-2.

In this section, we will compare segmentation results of our algorithm with the algorithms of [26] and [21]. The algorithm in [26] is a mutual information based label method on the pixel

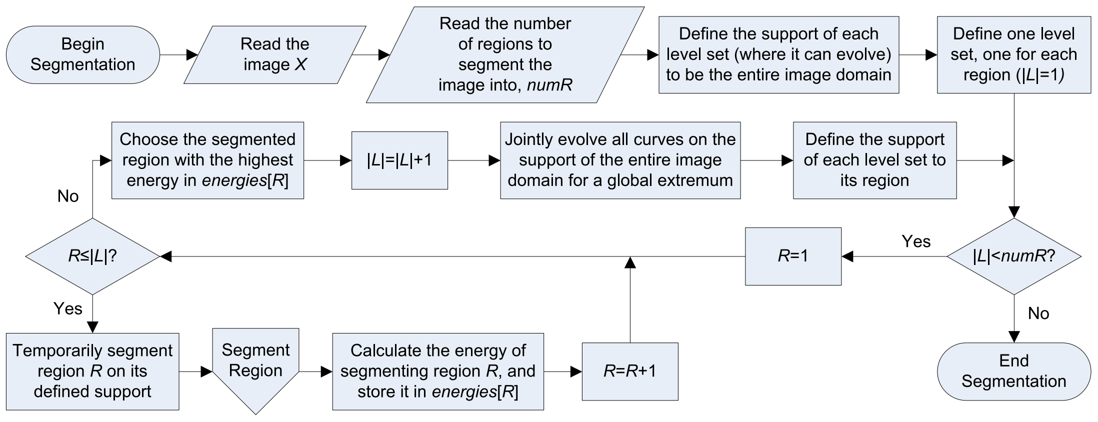

Begin Segmentation
Read the image X
Read the number of regions to segment the image into, numR
Define the support of each level set (where it can evolve) to be the entire image domain
Define one level set, one for each region (|L|=1)
Choose the segmented region with the highest energy in energies[R]
|L|=|L|+1
Jointly evolve all curves on the support of the entire image domain for a global extremum
Define the support of each level set to its region
No
R≤|L|?
R=1
Yes
|L|<numR?
Yes
No
Temporarily segment region R on its defined support
Segment Region
Calculate the energy of segmenting region R, and store it in energies[R]
R=R+1
End Segmentation

Figure 5-1: Flowchart for segmenting an image with multiple regions.

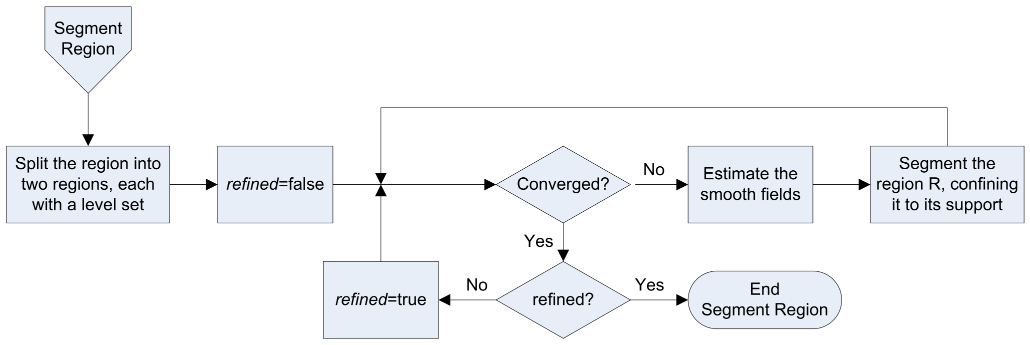

Segment Region
Split the region into two regions, each with a level set
refined=false
Converged?
No
Estimate the smooth fields
Segment the region R, confining it to its support
Yes
refined=true
No
refined?
Yes
End Segment Region

Figure 5-2: Flowchart for segmenting a region within an image.

intensities. However, the combination of a curve length penalty and nonparametric estimates on the intensity distributions enables the segmentation of simple textures without an explicit texture representation. The algorithm in [21] uses steerable pyramids, but differs from the method in that here because they use these filter outputs directly as features. They treat each filter output as statistically independent and segment based on this assumption. The exact algorithms of [26] and [21] produce very poor results in our experiments due to the fact they do not model changes due to illumination and other physical phenomenon. To make a fairer comparison, we incorporate our gain field into the approach of [21] and the gain and bias field into the approach of [26]. The bias field is excluded from [21] because it does not fit their model well and should have little or no effect on the features. This allows us to compare the difference gained in using our texture model by removing lighting effects from all algorithms.

5.1.1 Sensitivity to Parameters

We begin our comparison by looking at the sensitivity of segmentation results to the algorithm parameters. We show the sensitivity to three user-specified values: the local region size $|R_i|$ around pixel i, the initialization for the segmentation, and the curve length penalty weight, α.

Local Region Size

The first parameter we consider is the local region size, $|R_i|$. This term is not used in the other algorithms so the sensitivity is only considered for the method presented here. We note that this region size was used in the calculation of angular energy in Equation 4.2. If $|R_i|$ is too small, the measure may not accurately capture the local angular energy. However, if $|R_i|$ is too large, we oversmooth our feature set which may lead to problems in small regions. We show the results of segmenting an image based on varying $|R_i|$ within the range 3-by-3 to 9-by-9 in two different ways. Figure 5-3a shows the regions for the various segmentations, where the closer a pixel is to white or black, the more often it was labeled into the same region when varying values of $|R_i|$. Pixels that are gray are more sensitive to change. We also show our results in Figure 5-3b, where pixels that changed between segmentations are red. The more red a pixel is, the more sensitive it is to varying $|R_i|$. These figures show that our feature set typically captures the body of the zebra very well. As expected, thin regions (e.g. the legs and tail of the zebra) are most sensitive to this parameter. When a larger region size is used, the contrast energy is oversmoothed for these small

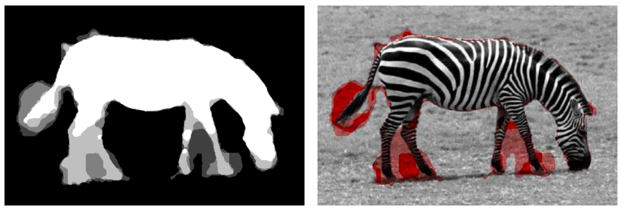


Figure 5-3: Sensitivity to local region size: (a) Regions changed when varying $|R_i|$ (gray pixels are more sensitive); (b) Pixels changed when varying $|R_i|$ (red pixels are more sensitive)

regions. Our algorithm is fairly sensitive to this parameter, but we show empirically that using the small, 3-by-3 region size typically allows for enough smoothing while still being able to capture small regions.

Initialization

The next parameter we consider is the initialization used to start the segmentation. We showed extensively in Section 4.1.3 that segmentation algorithms can converge to a local extremum (instead of a global) if given a poor initialization. Here, we consider multiple gridded initializations, where the size of each seed in the grid changes. If the seeds are too small, the curve length penalty may dominate the evolution and not capture all parts of an object. If the seeds are too large, then smaller regions in the image may not be captured. In general, the random seeds to initialize the segmentation should contain a mixture of the region statistics of the final segmentation. In the case of the zebra, when considering pixel intensities, if the initial seeds only contain pixels in the black stripes of the zebra, then there is no driving force in the evolution to include the white stripes. Thus, especially for regions that have a a multi-modal feature distribution, the initialization can produce very different results. Figure 5-4 shows a comparison of the sensitivity of the three algorithms to this parameter.

The middle image obtained using [21] is clearly the most sensitive to the initialization. Their texture model discriminates textures at different scales and orientation even if the textures look very similar or if they smoothly change into each other. Consequently, the stripes in the midsection of the zebra are represented as a different texture as the stripes in the hindquarters of the zebra. This discrimination creates many more local extrema which, as shown in Section 4.1.3, increases the sensitivity to the initialization.

The left image obtained using [26] performs fairly well for the set of initializations used. The few stripes that are occasionally not captured in this case can be corrected with a stronger curve length penalty weight. Without using any texture analysis, their pixel measure is oblivious to the actual structure of the pixels. In fact, if one were to randomly permute the pixels in the zebra so that stripes no longer existed, this algorithm would probably still segment the image correctly. While this algorithm works well for simple textured images, the empirical results in Section 5.1.2 will show why utilizing an explicit texture representation can aid in segmentation.

The right image obtained using the method presented here shows that it is not very susceptible to changes in the initialization. Unlike the method of [21], this most likely indicates that there are very few local extrema. Additionally, this method always captures all of the zebra stripes and

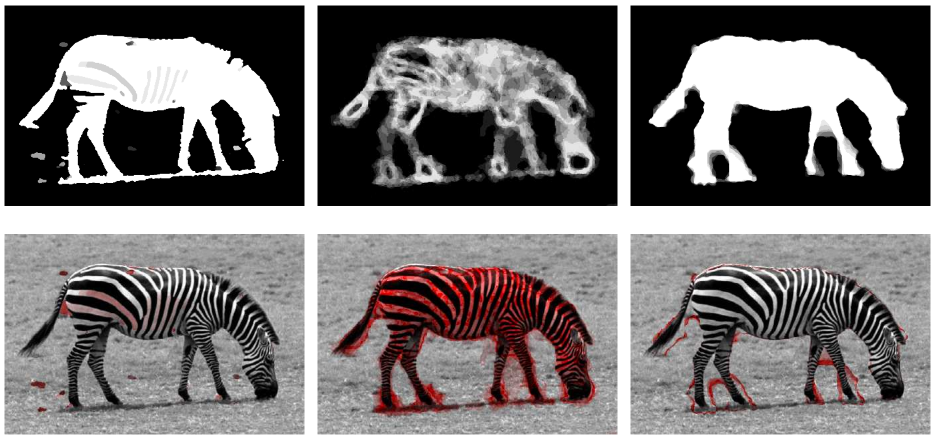


Figure 5-4: Sensitivity to random initialization: 1st column shows algorithm from [26]; 2nd column shows algorithm from [21]; 3rd column shows our algorithm; 1st row shows regions changed when varying initial seed size (gray pixels are more sensitive); 2nd row shows pixels changed when varying initial seed size (red pixels are more sensitive)

excludes the shadow because it models the texture appearance.

Curve Length Penalty Weight

The last parameter we consider is the weight of curve length penalty, α. This has a significant impact on the segmentation when there are small regions or there are sharp corners. The comparison is shown in Figure 5-5. Note that the region between the left legs of the zebra in the method presented here is not captured well when α is too large because this region is already small due to boundary effects. However, it is always able to capture the entire body of the zebra in one region, unlike the other methods.

The method of [21] shown in the middle image is again fairly sensitive to the curve length penalty weight. This is not surprising because as stated previously, this method creates many local extrema. By altering any segmentation parameter, we can expect this specific segmentation to change due to the gradient ascent approach used in level set methods.

As expected, the method of [26] shown in the left image is fairly sensitive to the curve length penalty weight. When this weighting is too small, the cost of assigning neighboring stripes different labels is not large enough to overcome the likelihoods. Thus, the thicker white stripes sometimes are grouped with the background when smaller curve length penalty weights are chosen.

5.1.2 Segmentation Results

In this section, we show empirical results for image segmentation. In each case, we compare the results obtained using the algorithms of [26] with an estimated gain and bias field, [21] with an estimated gain field, and the algorithm presented here. We use the multiple random initialization method described in Section 4.1.3 to provide a more robust result for each of these methods that is less sensitive to local extrema. The best result amongst the multiple random initializations are chosen as our segmentation, where ranking is determined solely by the energy associated with the

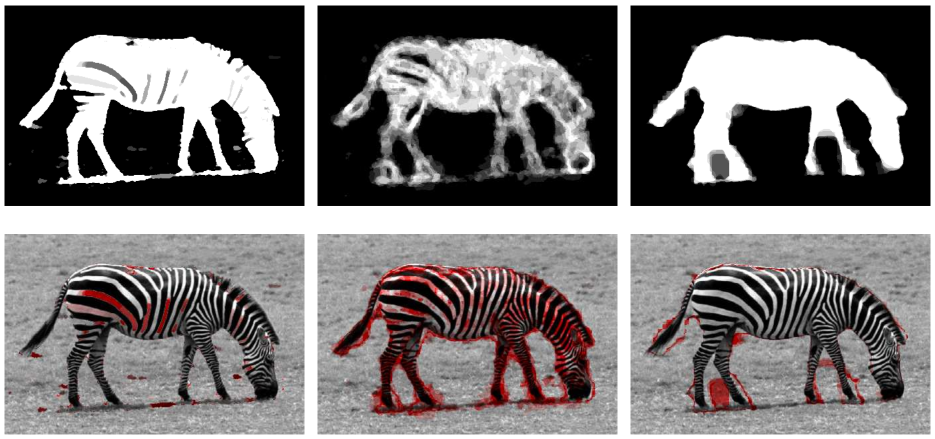


Figure 5-5: Sensitivity to curve length penalty weight: 1^{st} column shows algorithm from [26]; 2^{nd} column shows algorithm from [21]; 3^{rd} column shows our algorithm; 1^{st} row shows regions changed when varying curve length penalty weight (gray pixels are more sensitive); 2^{nd} row shows pixels changed when varying curve length penalty weight (red pixels are more sensitive)

segmentation without having access to the ground truth.

We first consider the segmentation shown in Figure 5-6. This image was segmented into both two and four regions for each of the algorithms. In the two region case, the scalar algorithm of [26] and the algorithm presented here both do very well in distinguishing the shirt from the background. Because the method of [21] does not capture dependencies between orientation, the sleeve of the shirt is categorized as a different texture from the body of the shirt.

In the four region case, the algorithm presented here is able to further distinguish the sleeve of the shirt and some of the folds in the body of the shirt. Notice how the other two algorithms fail in predictable ways. In a sense, the algorithm of [26] is approximately orientation and scale invariant for textures because it ignores those attributes of the texture. Thus, because the sleeve is only a different orientation from the body of the shirt, this method is not able to differentiate the two regions. On the other hand, the method of [21] is not orientation or scale invariant because each pyramid output is considered to be independent. Thus, it fails in grouping the sleeve with the body of the shirt in the two region case. The texture measure presented here is only partially invariant in scale and orientation, in that it is only invariant to smooth changes in these features. Consequently it performs well in both the two region and four region cases.

We now present additional segmentation results on images from the Berkeley Segmentation Dataset (BSD) [31] and others found on the internet. Again, we compare to the same algorithms as before and comment on a the segmentations in 5-7. The baseball field is separated fairly well in both the scalar algorithm and out algorithm whereas the independent steerable pyramid case did not perform very well. The scalar algorithm groups some of the background with the grass, while our algorithm does not. However, our algorithm is also not able to capture the grass between the mound and second base which is most likely due to using a local measure combined with the boundary effects. We have verified that excluding this small incorrect region has a higher energy (and thus a better segmentation), meaning that this segmentation must be a local extremum and not a global one.

In the second image of Figure 5-7 containing pillows, the difference between the algorithms is

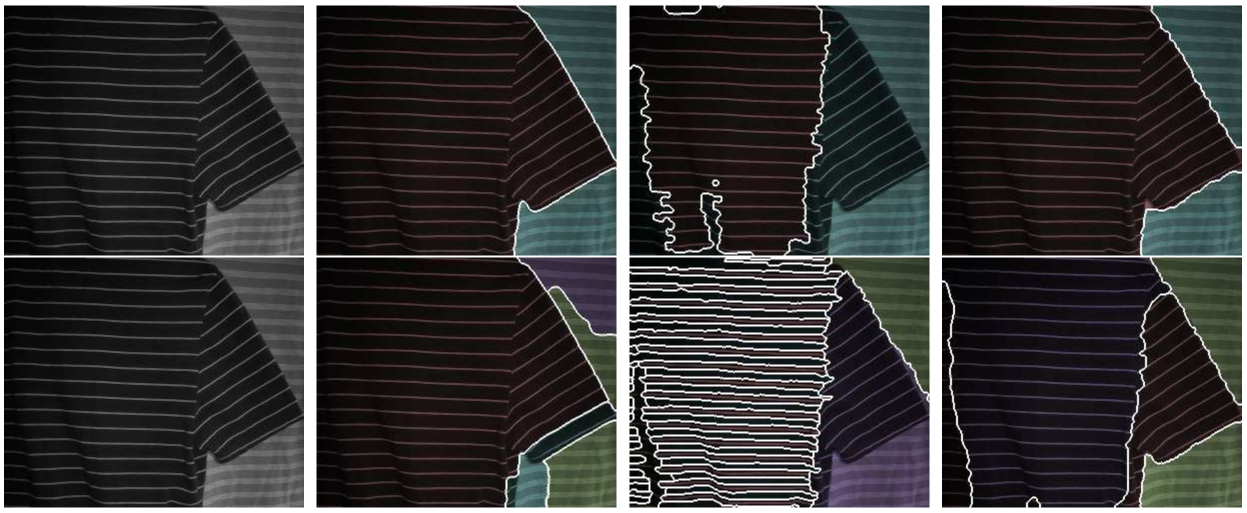


Figure 5-6: Segmentation Results: first column contains the original images, second column contains results from scalar algorithm [26] with estimated gain and bias fields, third column contains results from independent steerable pyramid algorithm [21] with estimated gain field, and fourth column contains results from our algorithm. First row is segmented into two regions, and second row is segmented into four regions. Notice how our algorithm is able to segment the shirt from the background in the two region case, and upon further segmenting into four regions, it can distinguish the sleeve and folds of the shirt.

more pronounced. The scalar algorithm groups only based on pixel intensities. Thus, most of the two pillows have similar intensities and are grouped together. However, some of the background is also grouped with the pillows due to the similar intensities. In the independent steerable pyramid case, only part of the left pillow is separated from the rest of the image. Because the texture is somewhat similar (modulo orientation and scale), we can think of the actual distributions within a scale and orientation to be fairly similar. The part of the pillow that is segmented has a different orientation from the other stripes resulting in the pyramid having high response at this orientation only in this region of the image. Consequently, this algorithm seems to segment the image based highly on orientation. Our algorithm is able to represent the textures of this image well and group the two pillows in one region. There is a slight error in the left pillow which is due to a rapidly changing texture in scale, gain, and bias fields. Our smoothly varying field is not able to capture this abrupt change well, and therefore the image is not properly segmented.

The third image of Figure 5-7 differs from other images because it does not contain a strong texture. One would think that the method of [26] (i.e. using scalar pixel intensities) would be best suited for this type of image. Though it is able to capture much of the image, it misses the corners that have a vignetting effect. Ideally, the gain and bias fields would have solved this issue; however the smoothness assumptions were not able to completely overcome the strong vignetting present in the picture (though it did help). The independent steerable pyramid method of [21] is able to capture a rough outline on the segmentation, but suffers significantly from border effects. Also, note how the body of the bird is not segmented properly. Our method is able to capture most of the image well, though it does have some errors near the image boundaries.

We chose to segment the fourth image in Figure 5-7 of zebras into three regions. We hoped that

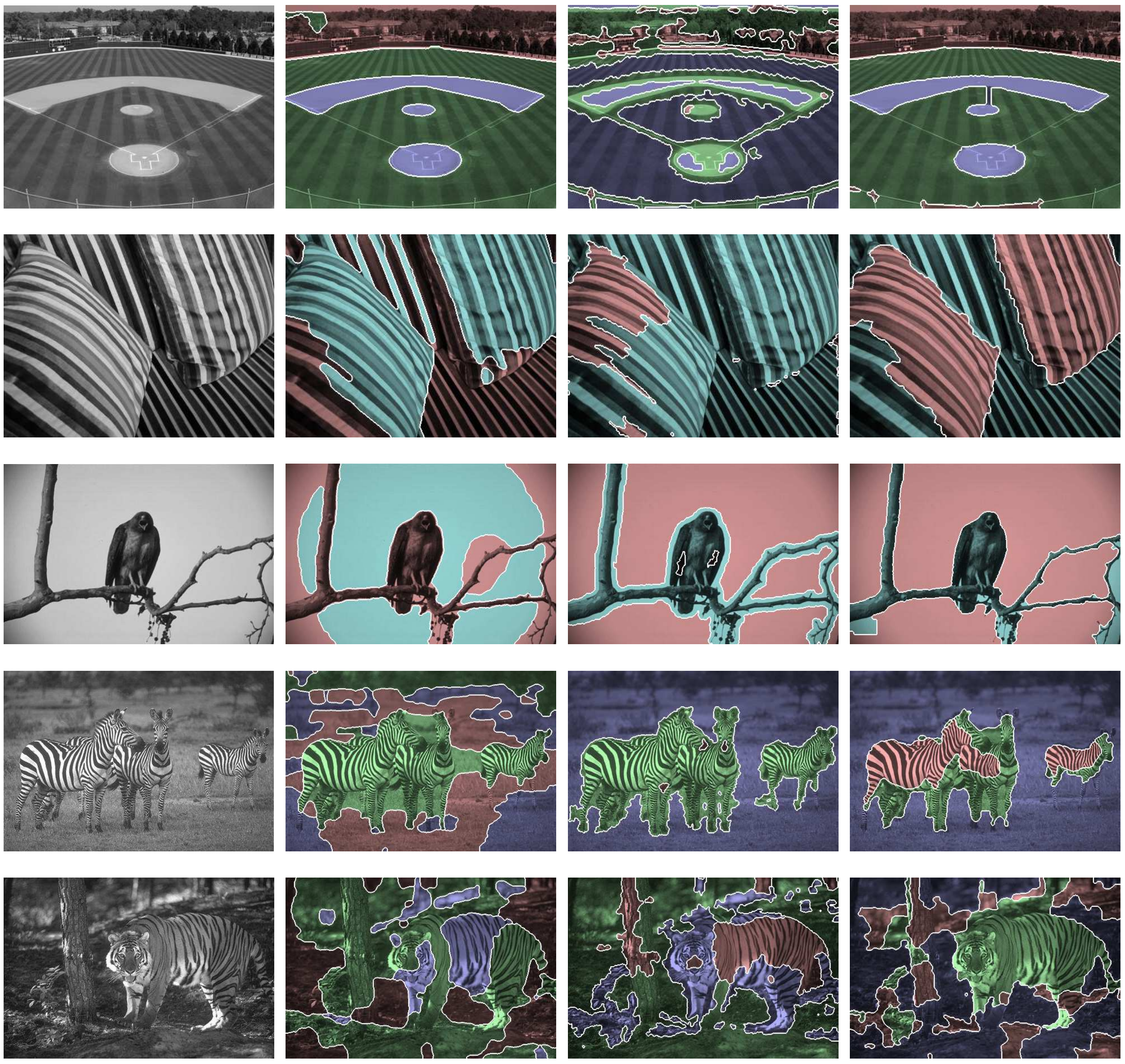


Figure 5-7: Segmentation Results: first column contains the original images, second column contains results from scalar algorithm [26] with estimated gain and bias fields, third column contains results from independent steerable pyramid algorithm [21] with estimated gain field, and fourth column contains results from our algorithm.

the zebras would be well described by a homogeneous texture with some smooth orientation and scale variations. Furthermore, we anticipated that abrupt changes in these features on boundaries between zebras would help distinguish the zebras. In the scalar case, the zebras are mainly grouped into one region, and the background is split into two based on their pixel intensities. The independent steerable pyramid method also does a good job in grouping all of the zebras in one region, but it is not able to differentiate between the zebras. The third region created by this method, which may be hard to see, captures the noses of two of the zebras and a small area between the legs. As shown previously, our algorithm is able to identify object boundaries based on abrupt changes in orientation or scale of a texture. However, these parameters are actually fairly smooth across the boundaries of the zebras. Yet, the legs of the zebras have an abrupt change in orientation, and the necks or chests of the zebras have an abrupt change in scale. Our algorithm therefore predictably segments under this model to distinguish these changes instead of separating the zebras.

The last image of Figure 5-7, similar to others, is taken from the BSD. For the algorithms compared in the dataset, this image is the 7^{th} hardest image out of the 100 images. Additional segmentation results are shown in Figure 5-8.

5.2 Feature Visualization and Intrinsic Texture Image

Once a segmentation is computed, we can use our feature set for various other computer vision problems. A common useful decomposition of an image is to find certain intrinsic images. An intrinsic image is a representation of the image that contains one intrinsic characteristic of the scene [3]. One pair of useful intrinsic images contains the intrinsic reflectance and shading images. An intrinsic reflectance image represents what the objects look like without any lighting effects. Each pixel is represented with the color attributes (or how it reflects light) of the object it belongs to. An intrinsic shading image represents the exact opposite; it contains only the lighting effects present in the image and describes how they interact with the specific geometries of the objects. The problem of finding intrinsic images can be quite difficult, especially with only one observed image. The problem becomes even harder when considering textured images because changes in intensities can be an effect of changes from either the reflectance (i.e. texture appearance) or shading.

The decomposition into shading and reflectance images has been studied by many scientists and remains to be a difficult problem. [50] considered a scenario in which multiple images of the same scene are obtained where the illumination changes between images. Though the problem is still ill-posed, they are able to obtain fairly good results on grayscale images. Recently, [28] developed a fairly robust algorithm to decompose the image into shading and reflectance from one color image. Their method uses the changing colors and a pre-trained gray-scale pattern identifier to classify whether changes are due to a reflectance change or shading change. While their method does perform very well, it is limited to color images and requires training data.

Though we have never explicitly represented these intrinsic images, we show that we can produce visually appealing results. We will discuss how the intrinsic reflectance image and what we call the intrinsic texture image are obtained in this section. The shading image recovery procedure will be discussed in Section 5.4.

The estimated gain and bias fields represent the changes of the intensities due to lighting conditions. Thus, if we remove these fields, then we have an estimate (at least to a scale factor) of the intrinsic reflectance image. If we extend the same reasoning to the estimated scale and orientation

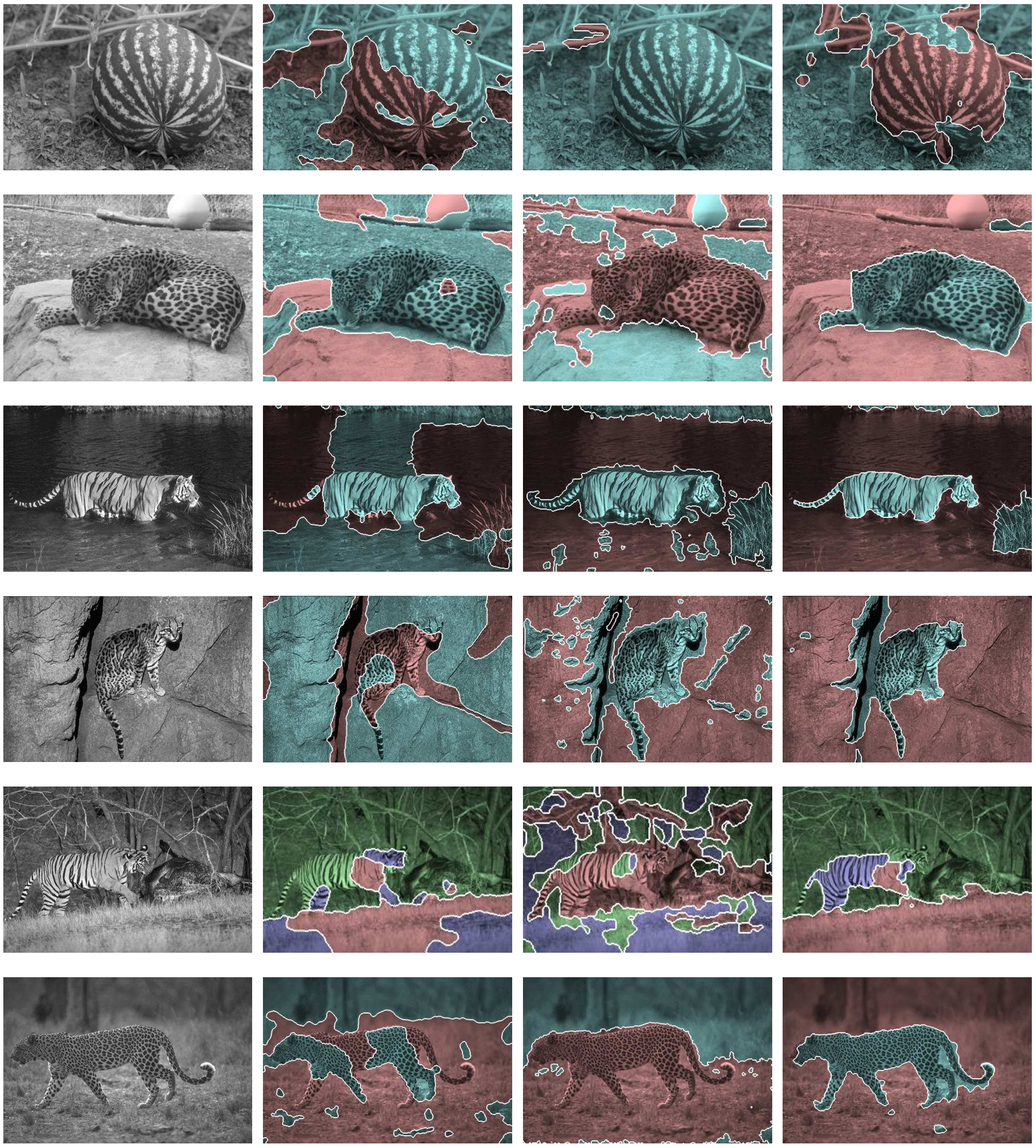


Figure 5-8: Additional Segmentation Results: first column contains the original images, second column contains results from scalar algorithm [26] with estimated gain and bias fields, third column contains results from independent steerable pyramid algorithm [21] with estimated gain field, and fourth column contains results from our algorithm.

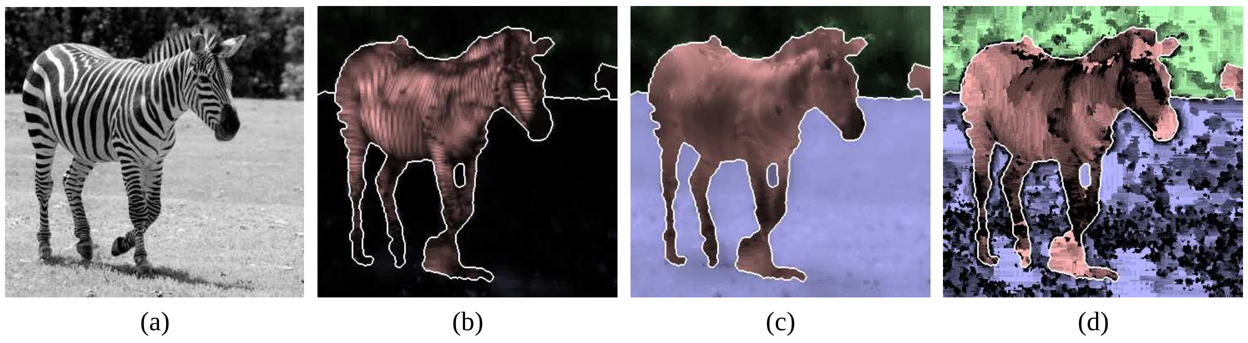
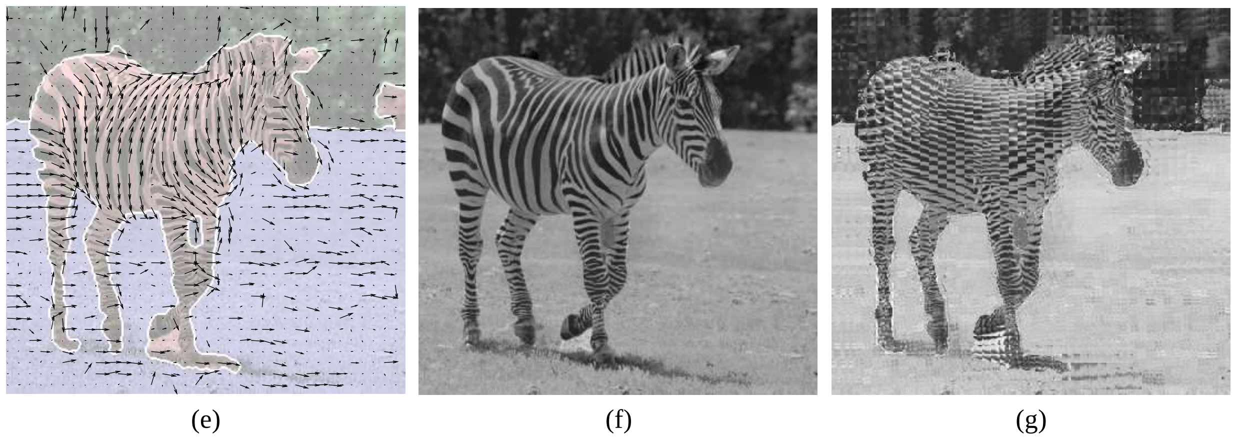



Figure 5-9: Calculated feature set: (a) Original image; (b) Contrast energy; (c) Bias; (d) Scale (brighter is coarser scale); (e) Orientation; (f) Estimated intrinsic reflectance image; (g) Visualization of the estimated intrinsic texture image

fields, removing these effects should provide us with an image that is uniformly illuminated with textures that are at uniform scales and orientations. We call this type of image the intrinsic texture image. In Figure 5-9, we show our measured features, our estimated intrinsic reflectance image, and a visualization of our estimated intrinsic texture image. The blocky artifacts in Figure 5-9g are a result of how we created the image. This visualization is obtained by rotating and scaling small regions of the estimated reflectance image with the estimated scale and orientation fields. This image is for illustrative purposes only; it is never actually used in the segmentation.

5.3 Nonlinear Camera Model Estimation

In the previous section, we estimated the intrinsic reflectance image by removing the effects of the gain and bias field. However, to formalize a more precise estimate of the reflectance image, we need to first consider the process a camera follows to output an image. We use a very simple model of the camera where the raw data from the camera sensor (also known as the irradiance image, $\underline{\mathcal{I}}$) is passed through some nonlinear mapping function (called the radiometric response function, $f(\cdot)$) to get to the observed image ($\underline{x}$). This process is illustrated in Figure 5-10. The radiometric function was first introduced to correct the γ factor, $(\cdot)^{\gamma}$, that is in most computer monitors. Today's cameras use more advanced functions to provide more visually appealing photographs. We sim-

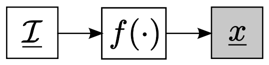


Figure 5-10: Model of the radiometric response function of the camera $f(\cdot)$ acting on the irradiance image $\underline{\mathcal{I}}$ to get to the observed image $\underline{x}$

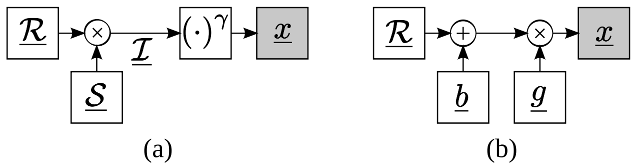


Figure 5-11: Model for the intrinsic reflectance image: (a) A one parameter, γ model for an unknown camera; (b) Our feature model of the image with a gain ($\underline{g}$) and bias ($\underline{b}$) field

plify this by approximating the radiometric response with a simple one-parameter power function:

$$f(\cdot) = (\cdot)^\gamma. \tag{5.1}$$

The intrinsic reflectance ($\underline{\mathcal{R}}$) and shading ($\underline{\mathcal{S}}$) images discussed previously have an intimate relationship with the irradiance image. Under certain common assumptions, the product of the reflectance and shading images is exactly the irradiance image [28]. We show this model with our assumptions in Figure 5-11a juxtaposed to our feature model in Figure 5-11b.

Here, we see that the two models in Figure 5-11 are very similar. As described previously, we first estimate the gain field and then the bias field. If there were no radiometric response function ($\gamma = 1$), then the lighting effects could be completely explained by a multiplicative gain field, and our estimate of the bias field should have very little energy. Thus, in the absence of γ, our gain field is approximately our shading image. We exploit this observation to find the optimal γ estimate in our camera model.

Given a value of γ, we can estimate the gain and bias fields as stated in Section 4.1.4. Assuming a noiseless model, the reflectance image is simply

$$\mathcal{R}_i(\gamma) = \frac{x_i^{1/\gamma}}{g_i(\gamma)} - b_i(\gamma), \quad \forall i \in \Omega. \tag{5.2}$$

Setting the bias field to be zero everywhere, which would only occur if the camera model was correct, we reconstruct the observed image using only the reflectance image and the gain field for a given γ value. The reconstruction error energy under the zero bias field assumption is

$$e(\gamma) = \left\| \left(\underline{\mathcal{R}}(\gamma) \cdot \underline{g}(\gamma) \right)^\gamma - \underline{x} \right\|_2, \tag{5.3}$$

where all operations are done element-wise. We find the optimal γ value by minimizing (5.3) within the range $[0.2, 1]$ using golden section search [25].

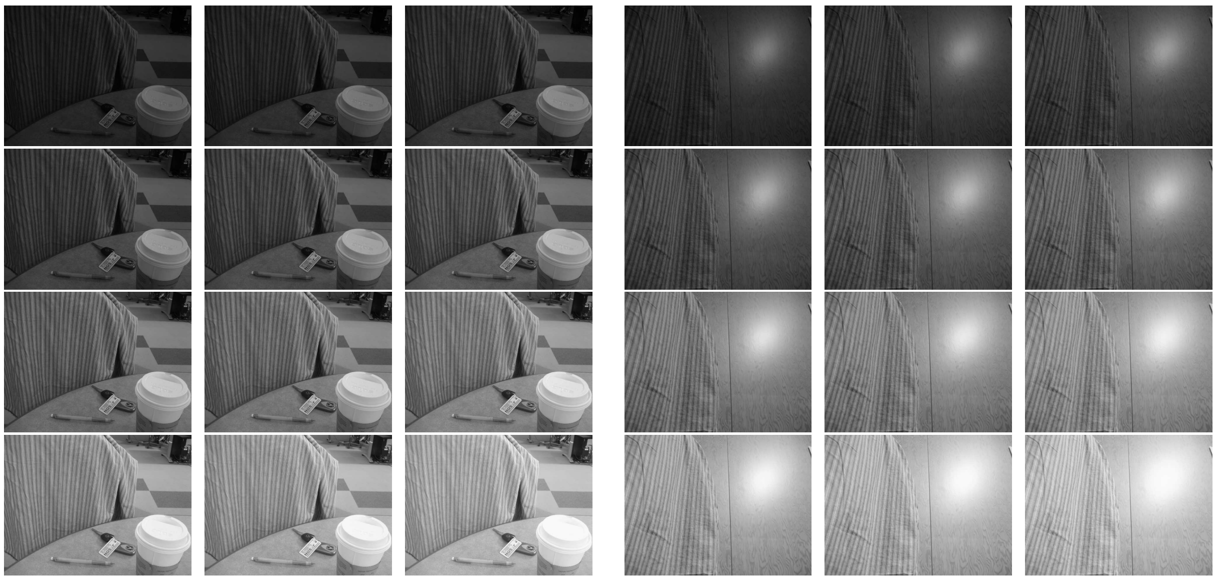


Figure 5-12: Two scenes taken at different exposures used to calibrate the radiometric response function of a camera

5.3.1 Verification with Multiple Exposed Images

We used a Canon A430 to take two different scenes at multiple exposures in order to capture the entire radiometric function. There are many reliable ways to determine the function from this set of multiple photos. The method of [33] has shown to provide reliable results on a wide variety of functions by using a polynomial parameterization. We estimated the radiometric response using this method on two sets of photos (displayed in Figure 5-12) and averaged the response.

Once the calibration was done, we took photographs using the same camera of various scenes. Those images are shown in Figure 5-13. We segmented each image in Figure 5-13 into two regions, and then found the best γ curve for the segmented image using our method and the method described in [14]. Because the method presented here finds a γ curve, we first obtain the optimal γ curve by finding the γ value to produce the closest curve in the L2 sense to the calibrated polynomial curve. We compare the estimated curves with the γ curve closest to the calibrated curve and show the results in Figure 5-14. The results obtained using the texture model presented here are superior to the results obtained using [14] in every test image, and were obtained using a single grayscale image.

5.4 Shading Estimation

Given the camera model (i.e. γ), it is straightforward to determine the shading image. One could use the estimated gain field as the shading image, but we have empirically observed that re-estimating the gain field assuming the bias field is zero produces better results. Here, we take into account that the bias feature, μ, is also affected by the gain field. As we assume that these two values are independent, it is straightforward to make this slight modification. The details of the modification are shown in the Appendix (Section B.2).

Figure 5-13: Various scenes photographed to test the accuracy of the radiometric response estimation procedure

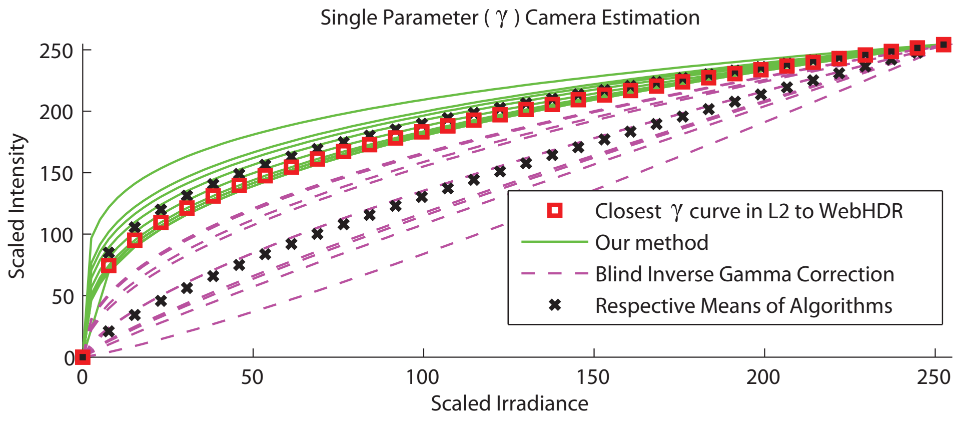


Figure 5-14: Camera model estimation comparing the method presented here with the method of [14]. The method presented here outperforms the method of [14] on every test image.

Once the shading is estimated (given that the camera response is known), the reflectance can be easily estimated by dividing the shading image from the irradiance image. Additionally, the shape of the object in the image can be estimated using common shape from shading techniques [22]. We visually verify our shading results by inspecting the estimated shape using the algorithm presented in [44] with post Gaussian smoothing. The results of the entire segmentation and estimation process is shown in Figure 5-15.

In the segmentation of the zebra, the method presented here is able to segment the three regions better than [26] and [21], even with incorporating the smooth fields. In each subsequent step, all algorithms are given the advantage of having the segmentation and the estimated radiometric response obtained using the algorithm presented here. The second row of Figure 5-15 shows the intrinsic reflectance and shading estimates for each algorithm respectively estimated using the same smoothness assumptions. The intensities of the shading image within each region have been scaled to $[0, 255]$ for illustrative purposes. When using the measure of [26], the reflectance seems to have some of the desired attributes (e.g. very minimal lighting or shading effects). However, the estimated shading image contains many indications of what should be in the reflectance image (e.g. the stripes of the zebra). The estimated reflectance and shading using the measure of [21] performs even worse. In fact, the decomposition makes the brighter patches in the reflectance even brighter and the darker patches even darker. The third column of estimates using the method presented here produces good results for both the reflectance and shading. The reflectance image seems to have minimal lighting or shading effects, and the shading image does not contain an overwhelming evidence of the reflectance image. Note that the shading of the nose of the zebra is incorrectly estimated because the texture of the nose (or lack thereof) is very different from the rest of the zebra.

As expected, the shape estimates reflect what was seen in the shading estimates. The stripes near the hindquarters of the zebra using [26] ripple with the intensity which is evidence of an incorrect shading estimate. When using [21], the shape of the zebra almost seems to be inverted, which indicates a very bad estimate of the shading and reflectance decomposition. The third column seems to provide a fairly good estimate of shape.

There has been extensive work (e.g. [45] and [29]) on estimating shape from texture cues such as changes in orientation and scale. The work presented here realizes that these cues could be due to the texture itself and not the geometric properties of the object. Where typical shape from texture or shape from shading algorithms may fail, the shape from shading from textures presented here does fairly well.

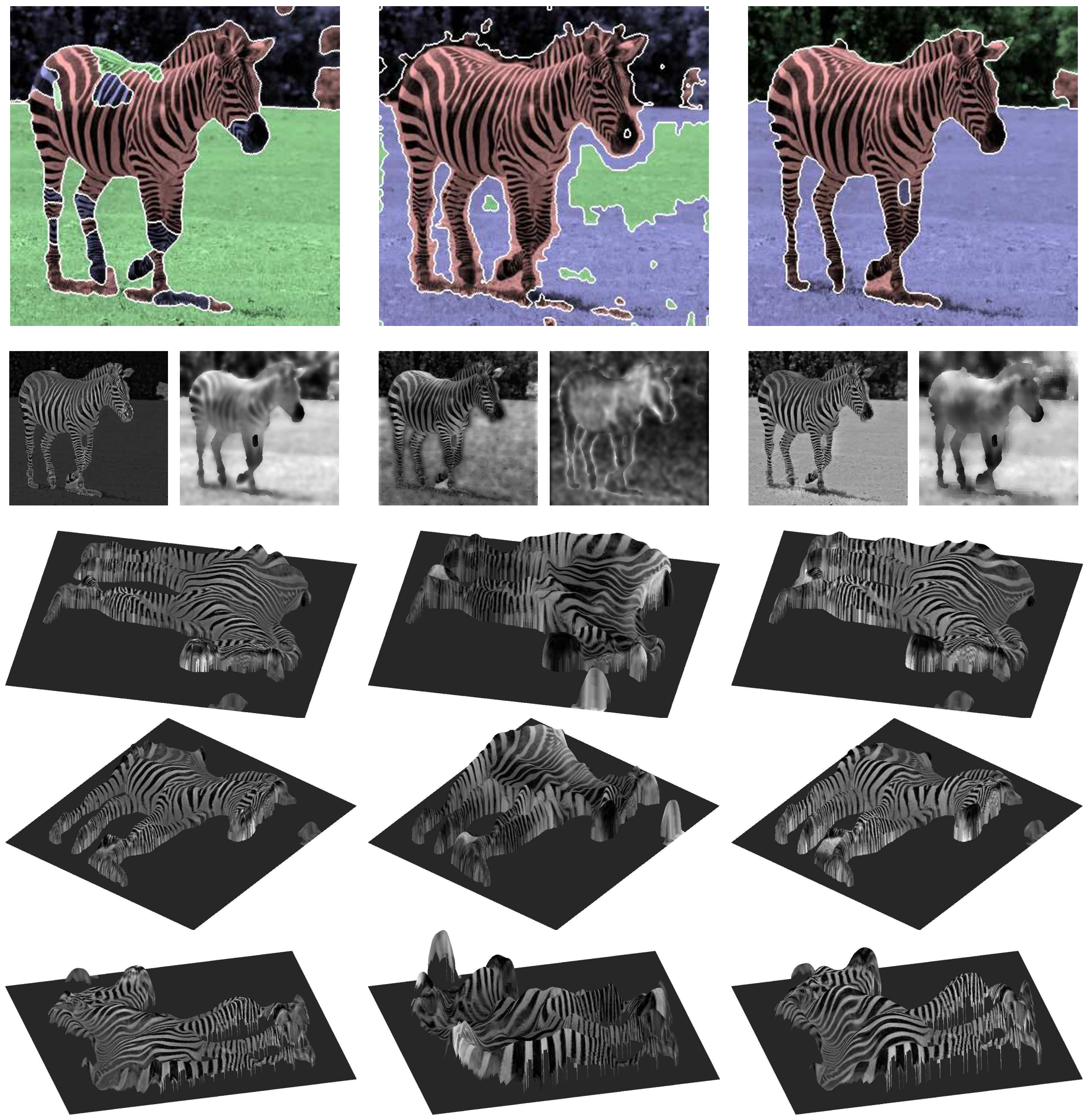


Figure 5-15: Segmentation, intrinsic shading estimation, and shape from shading shown for three texture measures. The first column is the scalar algorithm from [26] with estimated gain and bias fields, the second column is the independent steerable pyramid algorithm from [21] with estimated gain field, and the third column is from the method presented in this thesis. In each column, the first row shows the three-region segmentation, the second row shows the estimated intrinsic reflectance and shading images respectively, and the last three rows show the shape estimated using [44]. In each step of the process, the algorithm presented here outperforms the other two methods. The algorithms of [26] and [21], in addition to incorporating our smooth fields, is also given the benefit of our segmentation and camera response curve for the reflectance, shading, and shape estimation.

Chapter 6

Conclusion

Two main topics were discussed in this thesis: using information measures (specifically Ali-Silvey distances) for an energy criterion in image segmentation and modeling smoothly varying textures. Previous use of information measures for segmentation was categorized into either the label method (i.e. $d\left(p_{XL}, p_X p_L\right)$) or the conditional method (i.e. $d\left(p_X^+, p_X^-\right)$). Gradient ascent velocities were derived for general Ali-Silvey distances for both methods. More importantly, in the binary segmentation case, a simple relationship between the two methods was found that maps a distance taking on a specific form from one method to the other. This mechanism allows one to extend the limited binary segmentation of the conditional method to a multiple region segmentation using the label method.

This thesis has also presented a novel texture measure that decomposes a local image patch into the contrast, bias, scale, and orientation of a local texture patch. We incorporated smoothness, via Markov random fields, which, to our knowledge, has not been considered previously. This combination of texture features and imposed smoothness combine to segment images with various textures more robustly than other measures. Additionally, this representation easily extends to estimate the radiometric response of a camera from a *single* image more accurately than previously proposed methods. Lastly, it also allows for an accurate estimate of the irradiance, reflectance, and shading image, which have been empirically validated by recovering the shape of the object.

6.1 Possible Changes

Some very explicit choices were made in the work of this thesis. Though these decisions aided in the derivations, they are not the only possibilities. Here, we conjecture on aspects of our work that could be changed or improved.

The model (i.e. the features) we designed was chosen to very explicitly represent a texture with a dominant orientation. Empirically, we have validated that the model still seems to hold for textures that are not necessarily dominated by a single orientation (e.g. the spots of the leopard or the non-textured bird in Figure 5-7). Regardless of the success, one can potentially increase performance by considering a model that is not designed explicitly for strongly oriented textures.

One possible way to capture more textures is related to the angular energy defined in Section 4.1. The angular energy is defined for the range of angles in $[0, \pi)$, yet we only chose to use the maximum energy and the energy orthogonal to the maximum. While this lends itself well to the incorporation of the MRF estimation, one could consider other aspects of the distribution of angular energy.

Additionally, while the contrast energy and residual energy attempted to capture correlations amongst the various orientations of the steerable pyramid, our method for extracting the scale does not elegantly capture correlations across scales of the pyramid (a simple maximum operator is used). The steerable pyramid is an invertible representation, meaning that any scale and orientation can be computed from the basis. However, the process of inverting the pyramid and computing a new response is fairly inefficient. In the work presented here, we took advantage of the efficient interpolation that can be achieved in orientation for the steerable pyramid. If a particular transform could be interpolated in scale and orientation, correlations among both of these attributes would be more easily captured. Although [43] showed that a pyramid that is shiftable (i.e. can be interpolated) in scale can also be designed, they also argued that a pyramid that is shiftable in scale and orientation is not possible (without further approximations). One could possible consider using a pair of pyramids, one that is shiftable in scale and another one that is shiftable in orientation, but further efforts are needed to develop the idea.

Another possible change is relevant to the function chosen to describe the probability of a texture being strongly oriented in Section 4.1.2. It was noted that a strong change in the function should occur when the ratio of the contrast energy to the residual energy was approximately 15. This value was chosen because it empirically gave good results. The analysis would benefit from a more thorough experiment where, for example, humans are actually asked at what threshold in the ratio does a texture appear oriented.

6.2 Future Work

There are still many open areas of research and unanswered questions in the topics covered in this thesis. The following describes potential future research related to the work presented.

6.2.1 Information Measures

In Chapter 3, we considered the gradient ascent velocity for a general Ali-Silvey distance. The segmentation comparisons were computed with three specific measures: mutual information in the label method, J divergence in the conditional method, and balanced J divergence in the conditional method. A more comprehensive comparison of information measures would be beneficial to understand when particular measures are more suited for specific images. Additionally, one may consider incorporating the work of [35] which links information measures to surrogate loss functions.

6.2.2 Intelligent Initializations

As shown in Sections 4.1.3 and 5.1.1, image segmentation algorithms (especially those involving more complicated models) tend to have numerous local extrema and are fairly sensitive to the initialization. We have presented two approaches to a random initialization: the gridded and non-gridded method. We have shown that the using the best result of multiple non-gridded initializations typically outperforms the single gridded initialization. However, this method requires segmenting an image multiple times which increases computation time. A more intelligent initialization that analyzes the model measurements could be developed and used to overcome the local extrema issue without increasing the computation time. Additionally, one could consider perturbations to the level set function to avoid falling into a local extremum.

6.2.3 Experimental Validation

Methods for image segmentation and intrinsic reflectance and shading estimation are difficult to evaluate. We have presented a small set of results that is encouraging for our model; however, more extensive experimental validation is needed to determine how effective our model is. In segmentation, we have visually compared our results with two other algorithms. These algorithms were chosen because they fit into our framework fairly straightforwardly. Comparison to more recent algorithms would be beneficial, but is typically hard because of the lack of publicly available source code.

One possibility is to compare to a large segmentation database like the Berkeley Segmentation Database (BSD) [31]. This has not been done here for a few reasons. Firstly, this database typically reports the probability that a pixel is declared to be on a boundary, rather than the results of region-based segmentation methods. The method that the BSD uses to evaluate the success of an algorithm favors a soft boundary detector. Ground truth segmentations are combined from multiple hand segmentations where each typically is segmented into a different number of regions. Because the algorithm presented here considers segmenting an image into a specified number of regions, the goodness criterion of [31] may penalize this algorithm more than boundary detection algorithms. Thus, for a more accurate comparison, an extension using the texture model presented here to a boundary detection algorithm could be developed for a better comparison.

Additionally, large image databases like [31] inevitably contain a wide variety of natural images. Though we have tried to show results for various scenes, it is important to realize that our algorithm is specifically designed for textured images. Thus, a more suitable image database would contain only images that fit our model. To our knowledge, there is no database that is solely composed of natural textured images.

Appendix A

Derivations of Distance Measures

A.1 Equivalence of MAP and Maximizing MI

In this section we will show that finding the MAP estimate of the labeling is equivalent to finding the labeling that maximizes the mutual information of a random label and its likelihood. We denote the region, R^l, as the set of all pixels that have the labeling l.

$$R^l = \{i | L_i = l\} \tag{A.1}$$

We start with the typical MAP estimate on the labels, $\underline{L}$.

$$\begin{aligned}
\underline{L}^* &= \arg\max_{\underline{L}} P\left(\underline{L}|\underline{X}\right) && \text{(A.2)}\\
&= \arg\max_{\underline{L}} \log P\left(\underline{X}|\underline{L}\right) + \log P\left(\underline{L}\right) && \text{(A.3)}\\
&= \arg\max_{\underline{L}} \sum_{i\in\Omega} \log P\left(X_i|L_i\right) + \log P\left(\underline{L}\right) && \text{(A.4)}\\
&= \arg\max_{\underline{L}} \sum_{l} \left[\sum_{i\in R^l} \log P\left(X_i|L_i = l\right)\right] + \log P\left(\underline{L}\right) && \text{(A.5)}
\end{aligned}$$

where we have assumed in Equation A.4 that the pixels x_i are i.i.d. conditioned on the labeling. The inner summation in Equation A.5 is proportional to the empirical expected value of $\log P\left(x_i|L_i\right)$. If we assume that each region is fairly large, we can approximate the empirical expected value with the actual expected value. Thus, we have the following.

$$\begin{aligned}
\underline{L}^* &\approx \arg\max_{\underline{L}} \sum_{l} \left|R^l\right| \mathbf{E}\left[\log P\left(X_i|L_i = l\right)\right] + \log P\left(\underline{L}\right) && \text{(A.6)}\\
&= \arg\max_{\underline{L}} -\sum_{l} \left|R^l\right| H\left(X_i|L_i = l\right) + \log P\left(\underline{L}\right) && \text{(A.7)}\\
&= \arg\max_{\underline{L}} -|\Omega| \sum_{l} \frac{\left|R^l\right|}{|\Omega|} H\left(X_i|L_i = l\right) + \log P\left(\underline{L}\right) && \text{(A.8)}
\end{aligned}$$

Here, we note that the ratio, $\frac{|R^l|}{|\Omega|}$ is the empirical prior on the label L_i taking on value l. We can then relate this expression to mutual information by the following.

$$\begin{aligned}
\underline{L}^* &\approx \arg\max_{\underline{L}} - |\Omega| \sum_l P\left(L_i = l\right) H\left(X_i | L_i = l\right) + \log P\left(\underline{L}\right) && \text{(A.9)} \\
&= \arg\max_{\underline{L}} - |\Omega| H\left(X_i | L_i\right) + \log P\left(\underline{L}\right) && \text{(A.10)} \\
&= \arg\max_{\underline{L}} |\Omega| \left(I\left(X_i; L_i\right) - H\left(X_i\right)\right) + \log P\left(\underline{L}\right) && \text{(A.11)} \\
&= \arg\max_{\underline{L}} |\Omega| I\left(X_i; L_i\right) + \log P\left(\underline{L}\right) && \text{(A.12)}
\end{aligned}$$

This result shows the equivalence of finding the MAP labeling versus finding the labeling that maximizes the MI. The prior term on the labeling, $\log P\left(\underline{L}\right)$ can be thought of as a regularization term to ensure that the labeling tends to group neighboring pixels together. When using level-set methods, this term is related to the curve length penalty. Thus, we have shown that finding the MAP estimate is equivalent to maximizing the mutual information of a pixel and its label under certain commonly made assumptions.

A.2 General Formulas and Derivatives

Two approximations will be used throughout the derivation. They were stated originally in Chapter 3, but will be reproduced here for convenience:

$$\mathbb{E}_{p_X}\left[f(\cdot)\right] = \int_{x \in \mathcal{X}} p_X\left(x\right) f(x) dx \approx \frac{1}{|R|} \int_{i \in R} f(x_i) di \tag{A.13}$$

$$g_1(a) \approx g_2(a) \qquad \forall g_1, g_2 \in \left\{ \begin{array}{c} \frac{1}{|R|} \int_{i \in R} f(x_i) K(x_i - a) di, \\ \mathbb{E}_{p_X}\left[f(x_i) K(x_i - a)\right], \\ \int_{x \in \mathcal{X}} p_X\left(x\right) f(x) K(x - a) dx, \\ p_X\left(a\right) f(a) \end{array} \right\} \tag{A.14}$$

Two formulas will be used throughout the derivation. The first of these appears when we have an equation of the following form:

$$\frac{\partial E}{\partial t} = \oint_{\mathcal{C}} \left\langle \overrightarrow{V}, S\left(x_\ell\right) \overrightarrow{N} \right\rangle d\ell, \tag{A.15}$$

where the vector, $\overrightarrow{N}$, is a unit normal vector pointing outwards from the positive level set values. If we can take the partial derivative of an energy w.r.t. time, and write it in this form, then the gradient ascent velocity to maximize E for the level set at pixel ℓ is the following [53]:

$$\frac{\partial \varphi_\ell}{\partial t} = S\left(x_\ell\right) \delta_0(\varphi_\ell). \tag{A.16}$$

We typically use the smooth Heaviside function in implementation to smear $\delta_0(\varphi_\ell)$ over the curve as discussed in Section 2.1.3.

The other formula we will be using is the multidimensional Leibniz Integral rule (also known as the Reynolds Transport Theorem in two dimensions). This states the following:

$$\frac{\partial}{\partial t}\int_{R(t)} F\left(x_A,t\right)dA = \int_{R(t)} \frac{\partial}{\partial t}F\left(x_A,t\right)dA + \oint_{\mathcal{C}} \left\langle \overrightarrow{V}, F\left(x_\ell,t\right)\overrightarrow{N}\right\rangle d\ell, \tag{A.17}$$

where the vector, $\overrightarrow{N}$ is a unit normal vector pointing outwards from R. Because we are considering the same velocity for one level set, we will indicate a negative velocity for this equation when we integrate over R^-.

In addition to these two formulas, we will also often need the time derivative of a few terms.

$$\begin{aligned}
\frac{\partial}{\partial t}p_X^{\pm}\left(x_i\right) &= \frac{\partial}{\partial t}\left[\frac{1}{|R^{\pm}|}\int_{s\in R^{\pm}} K^{\pm}(x_i - x_s)ds\right] \\
&= \frac{\partial}{\partial t}\left[\frac{1}{|R^{\pm}|}\int_{s\in R^{\pm}} \frac{e^{-\frac{(x_i-x_s)^2}{h^2}}}{\sqrt{\pi h^2}}ds\right] \\
&= \frac{\partial}{\partial t}\left[\frac{1}{|R^{\pm}|}\right]\int_{s\in R^{\pm}} \frac{e^{-\frac{(x_i-x_s)^2}{h^2}}}{\sqrt{\pi h^2}}ds + \frac{1}{|R^{\pm}|}\frac{\partial}{\partial t}\left[\int_{s\in R^{\pm}} \frac{e^{-\frac{(x_i-x_s)^2}{h^2}}}{\sqrt{\pi h^2}}ds\right] \\
&= \oint_{\mathcal{C}} \left\langle \overrightarrow{V}, \mp\frac{p_X^{\pm}\left(x_i\right)}{|R^{\pm}|}\overrightarrow{N}\right\rangle d\ell + \oint_{\mathcal{C}} \left\langle \overrightarrow{V}, \pm\frac{K^{\pm}(x_i-x_\ell)}{|R^{\pm}|}\overrightarrow{N}\right\rangle d\ell
\end{aligned}$$

Thus, the time derivative to our nonparametric conditional distributions are

$$\frac{\partial}{\partial t}p_X^{\pm}\left(x_i\right) = \oint_{\mathcal{C}} \left\langle \overrightarrow{V}, \pm\frac{1}{|R^{\pm}|}\left[K^{\pm}(x_i-x_\ell) - p_X^{\pm}\left(x_i\right)\right]\overrightarrow{N}\right\rangle d\ell. \tag{A.18}$$

We will also need the time derivative of the size of a region:

$$\frac{\partial}{\partial t}\left|R^{\pm}\right| = \frac{\partial}{\partial t}\int_{i\in R^{\pm}} di = \oint_{\mathcal{C}} \left\langle \overrightarrow{V}, \pm\overrightarrow{N}\right\rangle d\ell. \tag{A.19}$$

A.3 Gradient Ascent for Conditional Method

We will now derive the gradient ascent velocity for the conditional method described in Section 3.2. We begin with an energy functional based on a general Ali-Silvey distance between the two distributions conditioned on the labeling:

$$E_C = |\Omega|\, d\left(p_X^+, p_X^-\right) = |\Omega| \int_{\mathcal{X}} p_X^+\left(x\right) C\left(\frac{p_X^-\left(x\right)}{p_X^+\left(x\right)}\right)dx, \tag{A.20}$$

where $|\Omega|$ is the number of pixels in the image and $C(\cdot)$ is a convex function not to be confused with $\mathcal{C}$, the curve. As stated previously, the $|\Omega|$ term is used so that the velocities can be accurately compared with the label method. This term only contributes a constant scaling on the velocity and does not affect the majority of the derivation. We proceed by taking the partial derivative of this energy w.r.t. time.

$$\frac{\partial}{\partial t}E_{CM} = |\Omega| \int_{\mathcal{X}} C\left(\frac{p_X^-(x)}{p_X^+(x)}\right)\frac{\partial p_X^+(x)}{\partial t} + C'\left(\frac{p_X^-(x)}{p_X^+(x)}\right)\left[\frac{\partial p_X^-(x)}{\partial t} - \frac{p_X^-(x)}{p_X^+(x)}\frac{\partial p_X^+(x)}{\partial t}\right]dx$$

where $C'(\cdot)$ is the derivative of $C(\cdot)$ w.r.t. to $(\cdot)$:

$$C'(\cdot) = \frac{\partial C(\cdot)}{\partial (\cdot)} \tag{A.21}$$

We now use Equation A.18 to obtain the following:

$$\begin{aligned}\frac{\partial}{\partial t}E_{CM} = &\oint_{\mathcal{C}}\left\langle \overrightarrow{V}, \frac{1}{\pi^+}\int_{\mathcal{X}}\left[K^+(x_i - x_\ell) - p_X^+(x_i)\right]C\left(\frac{p_X^-(x)}{p_X^+(x)}\right)dx\overrightarrow{N}\right\rangle d\ell\\ &+\oint_{\mathcal{C}}\left\langle \overrightarrow{V}, \frac{1}{\pi^-}\int_{\mathcal{X}}\left[-K^-(x_i - x_\ell) + p_X^-(x_i)\right]C'\left(\frac{p_X^-(x)}{p_X^+(x)}\right)dx\overrightarrow{N}\right\rangle d\ell\\ &-\oint_{\mathcal{C}}\left\langle \overrightarrow{V}, \frac{1}{\pi^+}\int_{\mathcal{X}}\left[K^+(x_i - x_\ell) - p_X^+(x_i)\right]\frac{p_X^-(x)}{p_X^+(x)}C'\left(\frac{p_X^-(x)}{p_X^+(x)}\right)dx\overrightarrow{N}\right\rangle d\ell.\end{aligned}$$

Using the approximation in Equation A.14, we can rewrite this expression as

$$\begin{aligned}\frac{\partial}{\partial t}E_{CM} = &\oint_{\mathcal{C}}\left\langle \overrightarrow{V}, \frac{1}{\pi^+}\left[C\left(\frac{p_X^-(x_\ell)}{p_X^+(x_\ell)}\right) - d\left(p_X^+, p_X^-\right)\right]\overrightarrow{N}\right\rangle d\ell\\ &+\oint_{\mathcal{C}}\left\langle \overrightarrow{V}, \frac{1}{\pi^-}\left[-C'\left(\frac{p_X^-(x_\ell)}{p_X^+(x_\ell)}\right) + \int_{\mathcal{X}} p_X^-(x_i)\, C'\left(\frac{p_X^-(x)}{p_X^+(x)}\right)dx\right]\overrightarrow{N}\right\rangle d\ell\\ &-\oint_{\mathcal{C}}\left\langle \overrightarrow{V}, \frac{1}{\pi^+}\left[\frac{p_X^-(x_\ell)}{p_X^+(x_\ell)}C'\left(\frac{p_X^-(x_\ell)}{p_X^+(x_\ell)}\right) - \int_{\mathcal{X}} p_X^-(x_i)\, C'\left(\frac{p_X^-(x)}{p_X^+(x)}\right)dx\right]\overrightarrow{N}\right\rangle d\ell\end{aligned}$$

Thus, the gradient ascent velocity at a point ℓ is

$$S_{CM}(x_\ell) = \frac{1}{\pi^+}\left[C\left(\frac{p_X^-(x_\ell)}{p_X^+(x_\ell)}\right) - d\left(p_X^+, p_X^-\right)\right] + \frac{1}{\pi^+\pi^-}\mathbb{E}_{p_X^-}\left[C'\left(\frac{p_X^-(\cdot)}{p_X^+(\cdot)}\right)\right] \tag{A.22}$$

$$-\left[\frac{1}{\pi^-} + \frac{p_X^-(x_\ell)}{\pi^+ p_X^+(x_\ell)}\right]C'\left(\frac{p_X^-(x_\ell)}{p_X^+(x_\ell)}\right). \tag{A.23}$$

A.4 Gradient Ascent for Label Method

We will now derive the gradient ascent velocity for the label method. We begin with a general Ali-Silvey distance between the joint distribution and the product of the marginal distributions.

$$E_{LM} = |\Omega|\, d\left(p_{XL}, p_X p_L\right) = |\Omega|\,\mathbb{E}_{p_{XL}}\left[C\left(\frac{p_X p_L}{p_{XL}}\right)\right] = |\Omega|\sum_{l\in\mathcal{L}}\int_{\mathcal{X}} p_{XL}(x,l)\, C\left(\frac{p_X(x)p_L(l)}{p_{XL}(x,l)}\right)dx$$

By conditioning on the labels, l, we can rewrite this expression as

$$E_{LM} = |\Omega| \sum_{l \in \mathcal{L}} \pi^l \int_{\mathcal{X}} p_X^l(x) C\left(\frac{p_X(x)}{p_X^l(x)}\right) dx,$$

where the notation $p_X^l(\cdot)$ is the distribution conditioned on the label taking on a value of l (equivalently $p_{X|L}(\cdot|L=l)$). We proceed by using the LLN approximation in Equation A.13:

$$\begin{aligned} E_{LM} &= |\Omega| \sum_{l \in \mathcal{L}} \frac{|R^l|}{|\Omega|} \frac{1}{|R^l|} \int_{R^l} C\left(\frac{p_X(x_i)}{p_X^l(x_i)}\right) di \\ &= \sum_{l \in \mathcal{L}} \int_{R^l} C\left(\frac{p_X(x_i)}{p_X^l(x_i)}\right) di. \end{aligned}$$

For simplicity, we now consider the two region case:

$$E_{LM} = \int_{R^+} C\left(\frac{p_X(x_i)}{p_X^+(x_i)}\right) di + \int_{R^-} C\left(\frac{p_X(i)}{p_X^-(i)}\right) di.$$

We proceed by taking the partial derivative w.r.t. time of this energy functional:

$$\frac{\partial}{\partial t} E_{LM} = \frac{\partial}{\partial t} \int_{R^+} C\left(\frac{p_X(x_i)}{p_X^+(x_i)}\right) di + \frac{\partial}{\partial t} \int_{R^-} C\left(\frac{p_X(x_i)}{p_X^-(x_i)}\right) di. \tag{A.24}$$

We consider these two terms separately, noting that there is only a sign difference due to the opposite normal directions:

$$\begin{aligned} &\frac{\partial}{\partial t} \int_{R^\pm} C\left(\frac{p_X(x_i))}{p_X^\pm(x_i)}\right) di \\ &= \oint_{\mathcal{C}} \left\langle \overrightarrow{V}, \pm C\left(\frac{p_X(x_\ell)}{p_X^\pm(x_\ell)}\right) \overrightarrow{N} \right\rangle d\ell + \int_{R^\pm} C'\left(\frac{p_X(x_i)}{p_X^\pm(x_i)}\right) \frac{\partial}{\partial t} \frac{p_X(x_i)}{p_X^\pm(x_i)} di. \end{aligned} \tag{A.25}$$

The second term in this equation can be simplified as follows:

$$\begin{aligned} &\int_{R^\pm} C'\left(\frac{p_X(x_i)}{p_X^\pm(x_i)}\right) \frac{\partial}{\partial t} \frac{p_X(x_i)}{p_X^\pm(x_i)} di \\ &= \int_{R^\pm} C'\left(\frac{p_X(x_i)}{p_X^\pm(x_i)}\right) \frac{p_X(x_i)}{-p_X^\pm(x_i)^2} \oint_{\mathcal{C}} \left\langle \overrightarrow{V}, \left[\pm \frac{K^\pm(x_i - x_\ell) - p_X^\pm(x_i)}{|R^\pm|}\right] \overrightarrow{N} \right\rangle d\ell di \\ &= \oint_{\mathcal{C}} \left\langle \overrightarrow{V}, \left[\pm \frac{1}{|R^\pm|} \int_{R^\pm} C'\left(\frac{p_X(x_i)}{p_X^\pm(x_i)}\right) \frac{p_X(x_i)}{p_X^\pm(x_i)} \left(1 - \frac{K^+(x_i - x_\ell)}{p_X^\pm(x_i)}\right) di\right] \overrightarrow{N} \right\rangle d\ell \\ &\approx \oint_{\mathcal{C}} \left\langle \overrightarrow{V}, \left[\pm \int_{\mathcal{X}} C'\left(\frac{p_X(x)}{p_X^\pm(x)}\right) p_X(x) \left(1 - \frac{K^\pm(x - x_\ell)}{p_X^\pm(x)}\right) dx\right] \overrightarrow{N} \right\rangle d\ell \\ &\approx \oint_{\mathcal{C}} \left\langle \overrightarrow{V}, \left[\pm \mathbb{E}_{p_X}\left[C'\left(\frac{p_X}{p_X^\pm}\right)\right] \mp C'\left(\frac{p_X(x_\ell)}{p_X^\pm(x_\ell)}\right) \frac{p_X(x_\ell)}{p_X^\pm(x_\ell)}\right] \overrightarrow{N} \right\rangle d\ell, \end{aligned} \tag{A.26}$$

where we have used the approximation in Equations A.13 and A.14. Combining Equations A.24 - A.26, we find that the gradient ascent velocity for pixel ℓ is

$$\begin{aligned} S_{LM}\left(x_\ell\right) =& C\left(\frac{p_X\left(x_\ell\right)}{p_X^+\left(x_\ell\right)}\right) + \mathbb{E}_{p_X}\left[C'\left(\frac{p_X}{p_X^+}\right)\right] - C'\left(\frac{p_X\left(x_\ell\right)}{p_X^+\left(x_\ell\right)}\right)\frac{p_X\left(x_\ell\right)}{p_X^+\left(x_\ell\right)} \\ & -C\left(\frac{p_X\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}\right) - \mathbb{E}_{p_X}\left[C'\left(\frac{p_X}{p_X^-}\right)\right] + C'\left(\frac{p_X\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}\right)\frac{p_X\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}. \end{aligned} \tag{A.27}$$

A.5 Gradient Ascent for a Balanced Symmetric Ali-Silvey Distance

We begin with an energy functional of the form of a balanced symmetric Ali-Silvey distance measure as described in Section 3.5

$$E_{BS} = |\Omega|\, d_{BS}\left(p_X^+, p_X^-\right) = |\Omega|\left[\pi^+ d\left(p_X^+, p_X^-\right) + \pi^- d\left(p_X^-, p_X^+\right)\right].$$

We now derive the gradient ascent velocity of the left term and extend it to find the overall velocity. Taking the partial derivative w.r.t. time of the left term leads us to

$$\begin{aligned} &\frac{\partial}{\partial t}E_{BS,left} \\ &= \frac{\partial \pi^+}{\partial t}|\Omega|\, d\left(p_X^+, p_X^-\right) + \pi^+\frac{\partial}{\partial t}\left[|\Omega|\, d\left(p_X^+, p_X^-\right)\right] \\ &= \frac{\partial R^+}{\partial t} d\left(p_X^+, p_X^-\right) + \pi^+\frac{\partial}{\partial t}E_{CM} \\ &= \oint_{\mathcal{C}}\left\langle \overrightarrow{V}, \left[C\left(r\left(x_\ell\right)\right) + \frac{1}{\pi^-}\mathbb{E}_{p_X^-}\left[C'\left(r\left(\cdot\right)\right)\right] - \left[\frac{\pi^+}{\pi^-} + r\left(x_\ell\right)\right]C'\left(r\left(x_\ell\right)\right)\right]\overrightarrow{N}\right\rangle d\ell. \end{aligned}$$

Thus, the total gradient ascent velocity for a balanced symmetric distance is

$$\begin{aligned} S_{BS}\left(x_\ell\right) = & \left[C\left(\frac{p_X^-\left(x_\ell\right)}{p_X^+\left(x_\ell\right)}\right) + \frac{1}{\pi^-}\mathbb{E}_{p_X^-}\left[C'\left(\frac{p_X^-\left(\cdot\right)}{p_X^+\left(\cdot\right)}\right)\right] - \left[\frac{\pi^+}{\pi^-} + \frac{p_X^-\left(x_\ell\right)}{p_X^+\left(x_\ell\right)}\right]C'\left(\frac{p_X^-\left(x_\ell\right)}{p_X^+\left(x_\ell\right)}\right)\right. \\ & \left. -C\left(\frac{p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}\right) - \frac{1}{\pi^+}\mathbb{E}_{p_X^+}\left[C'\left(\frac{p_X^+\left(\cdot\right)}{p_X^-\left(\cdot\right)}\right)\right] + \left[\frac{\pi^-}{\pi^+} + \frac{p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}\right]C'\left(\frac{p_X^+\left(x_\ell\right)}{p_X^-\left(x_\ell\right)}\right)\right]. \end{aligned} \tag{A.28}$$

Appendix B

Smooth MRF Estimation

In this chapter of the appendix, derivations for Markov random field (MRF) estimation are shown. First, the fixed-point update for a field affecting a single parameter is shown. Then, we extend this derivation to a field that affects two parameters (specifically, the shading image estimation affecting both contrast energy and bias). Finally, we derive a generalized MRF estimation for a field that affects a set of independent parameters.

B.1 Single Parameter MRF Estimation

We start with a smooth additive Markov random field called $\underline{\tilde{\phi}}$. We use the model shown in Figure 4-10c where our intrinsic parameter, $\underline{\theta}^*$, is estimated using a kernel density estimate. We assume that our MRF is $\underline{\tilde{\phi}} \sim \mathcal{N}\left(\underline{0}, \Lambda_\phi\right)$. Our goal is to find the MAP estimate of $\underline{\tilde{\phi}}$. Starting from Equation 4.12, we have:

$$\begin{aligned}
\underline{\tilde{\phi}} &= \arg\max_{\underline{\phi}} \left[P\left(\underline{\phi}|\underline{\theta}\right)\right] \\
&= \arg\max_{\underline{\phi}} \left[\log P\left(\underline{\theta}|\underline{\phi}\right) + \log P\left(\underline{\phi}\right)\right] \\
&= \arg\max_{\underline{\phi}} \left[\sum_{i=1}^{N} \log P\left(\theta_i|\phi_i\right) + \log P\left(\underline{\phi}\right)\right]
\end{aligned}$$

where we have assumed that the elements of $\underline{\theta}^*$ are *i.i.d.* conditioned on knowing $\underline{\tilde{\phi}}$. We then differentiate the expression and set it to zero to find the maximum.

$$\frac{\partial}{\partial \underline{\phi}} \left[\sum_{i=1}^{N} \log P\left(\theta_i|\phi_i\right) + \log P\left(\underline{\phi}\right)\right]_{\underline{\tilde{\phi}}} = 0 \tag{B.1}$$

We first look at the derivative of the left term. Because each term in the summation is only a function of ϕ_i, we can bring the derivative inside the sum.

$$\frac{\partial}{\partial \phi_i} \log P\left(\theta_i|\phi_i\right) = \frac{\frac{\partial}{\partial \phi_i} P\left(\theta_i|\phi_i\right)}{P\left(\theta_i|\phi_i\right)} \tag{B.2}$$

The numerator in (B.2) can be found by the following:

$$\begin{aligned}
\frac{\partial}{\partial \phi_i}\left[P\left(\theta_i|\phi_i\right)\right] &= \frac{\partial}{\partial \phi_i}\left[p_{\theta_i|\phi_i}\left(\theta_i|\phi_i\right)\right] = \frac{\partial}{\partial \phi_i}\left[p_\theta\left(\theta_i-\phi_i\right)\right] \\
&= \frac{\partial}{\partial \phi_i}\left[\frac{1}{Nh\sqrt{\pi}}\sum_{s=1}^{N} e^{-\frac{\left(\theta_i-\phi_i-\theta_s+\phi_s\right)^2}{h^2}}\right] \\
&= \frac{1}{Nh\sqrt{\pi}}\sum_{s=1}^{N} e^{-\frac{\left(\theta_i-\phi_i-\theta_s+\phi_s\right)^2}{h^2}}\left(\frac{2\left(\theta_i-\phi_i-\theta_s+\phi_s\right)}{h^2}\right) \\
&= \frac{2}{h^2}\left[\left(\theta_i-\phi_i\right)p_\theta\left(\theta_i-\phi_i\right) - w_\theta\left(\theta_i-\phi_i\right)\right]
\end{aligned}$$

where the terms $p_{\theta_i}\left(\theta_i-\phi_i\right)$ and $w_{\theta_i}\left(\theta_i-\phi_i\right)$ are the estimated PDF and weighted PDF defined as follows

$$p_\theta\left(\theta_i-\phi_i\right) = \frac{1}{Nh\sqrt{\pi}}\sum_{s=1}^{N} e^{-\frac{\left(\theta_i-\phi_i-\theta_s+\phi_s\right)^2}{h^2}}$$

$$w_\theta\left(\theta_i-\phi_i\right) = \frac{1}{Nh\sqrt{\pi}}\sum_{s=1}^{N} e^{-\frac{\left(\theta_i-\phi_i-\theta_s+\phi_s\right)^2}{h^2}}\left(\theta_s-\phi_s\right)$$

Defining the ratio, $w_p^\theta(\cdot)$, as the weighted PDF over the PDF, Equation B.2 becomes the following:

$$\frac{\partial}{\partial \phi_i}\log P\left(\theta_i|\phi_i\right) = \frac{2}{h^2}\left[\theta_i-\phi_i-w_p^\theta\left(\theta_i-\phi_i\right)\right] \tag{B.3}$$

We note that the derivative of a zero mean multivariate normal distribution is the following:

$$\frac{\partial}{\partial \underline{\phi}}P\left(\underline{\phi}\right) = \Lambda_\phi^{-1}\underline{\tilde{\phi}}P\left(\underline{\phi}\right) \tag{B.4}$$

Using Equations B.3 and B.4, we can rewrite Equation B.1 as the following:

$$\begin{aligned}
0 &= \frac{2}{h^2}\left[\underline{\theta}-\underline{\tilde{\phi}}-w_p^\theta\left(\underline{\theta}-\underline{\tilde{\phi}}\right)\right] - \Lambda_\phi^{-1}\underline{\tilde{\phi}} \\
&\Rightarrow \underline{\theta}-\underline{\tilde{\phi}}-w_p^\theta\left(\underline{\theta}-\underline{\tilde{\phi}}\right) - \frac{h^2}{2}\Lambda_\phi^{-1}\underline{\tilde{\phi}} = 0
\end{aligned}$$

which is exactly the expression in Equation 4.13 of Section 4.1.4.

B.2 Shading Estimation

When estimating the shading image, we look at its effects on both the contrast energy and the bias term. It is important to note that this is only for estimating the shading image, $\underline{\mathcal{S}}$, and not the gain field, $\underline{g}$, or the bias field $\underline{b}$. This is only a slight modification, but should be mentioned nonetheless.

The derivation for this field follows straightforwardly from the previous derivation. From Equation 4.12, we simply replace the first conditional distribution with the product of our two features

$$\underline{\mathcal{S}} = \arg\max_{\underline{S}} \left[\log\left(P\left(\underline{E}|\underline{\mathcal{S}}\right) P\left(\underline{\mu}|\underline{\mathcal{S}}\right)\right) + \log P\left(\underline{\mathcal{S}}\right)\right].$$

We assume the shading image is smooth in the log domain which allows us to treat it as an additive field. The contrast energy was a sum of squared filter outputs, thus the log of the shading image is additive on the log of the square root of the contrast energy. We define our fields in the log domain as follows

$$\begin{aligned} \underline{\mathcal{S}}_{\log} &= \log\left(\underline{\mathcal{S}}\right) \\ \underline{E}_{\log} &= \log\left(\sqrt{\underline{E}}\right) \\ \underline{\mu}_{\log} &= \log\left(\underline{\mu}\right). \end{aligned}$$

Because of the independence assumption, we can propagate this change through most of the proof and arrive at the following

$$\begin{array}{l} \frac{2}{h^2_{E_{\log}}}\left[\underline{E}_{\log} - \underline{\mathcal{S}}_{\log} - w_p^E\left(\underline{E}_{\log} - \underline{\mathcal{S}}_{\log}\right)\right] \\ +\frac{2}{h^2_{\mu_{\log}}}\left[\underline{\mu}_{\log} - \underline{\mathcal{S}}_{\log} - w_p^\mu\left(\underline{\mu}_{\log} - \underline{\mathcal{S}}_{\log}\right)\right] \end{array} = \Lambda^{-1}_{\mathcal{S}_{\log}}\underline{\mathcal{S}}_{\log}.$$

Now we rearrange the equation as follows

$$\begin{array}{l} \frac{2}{h^2_{E_{\log}}}\left[\underline{E}_{\log} - w_p^E\left(\underline{E}_{\log} - \underline{\mathcal{S}}_{\log}\right)\right] \\ +\frac{2}{h^2_{\mu_{\log}}}\left[\underline{\mu}_{\log} - w_p^\mu\left(\underline{\mu}_{\log} - \underline{\mathcal{S}}_{\log}\right)\right] \end{array} = \left[\Lambda^{-1}_{\mathcal{S}_{\log}} + \frac{2}{h^2_{E_{\log}}}I + \frac{2}{h^2_{\mu_{\log}}}I\right]\underline{\mathcal{S}}_{\log} = M\underline{\mathcal{S}}_{\log},$$

where the matrix M is introduced for notational purposes. Continuing, we have the following

$$\begin{aligned} \underline{\mathcal{S}}_{\log} &= M^{-1}\frac{2}{h^2_{E_{\log}}}\left[\underline{E}_{\log} - w_p^E\left(\underline{E}_{\log} - \underline{\mathcal{S}}_{\log}\right)\right] + M^{-1}\frac{2}{h^2_{\mu_{\log}}}\left[\underline{\mu}_{\log} - w_p^\mu\left(\underline{\mu}_{\log} - \underline{\mathcal{S}}_{\log}\right)\right] \\ \underline{\mathcal{S}}_{\log} &= \underline{\mathcal{S}}_{\log,1} + \underline{\mathcal{S}}_{\log,2}. \end{aligned}$$

Now we consider only one of these terms, as it is straightforward to find the other one.

$$\begin{aligned} \underline{\mathcal{S}}_{\log,1} &= \left[\Lambda^{-1}_{\mathcal{S}_{\log}} + \frac{2}{h^2_{E_{\log}}}I + \frac{2}{h^2_{\mu_{\log}}}I\right]^{-1}\frac{2}{h^2_{E_{\log}}}\left(\underline{E}_{\log} - w_p^E\left(\underline{E}_{\log} - \underline{\mathcal{S}}_{\log}\right)\right) \\ &= \left[\frac{h^2_{E_{\log}}}{2}\Lambda^{-1}_{\mathcal{S}_{\log}} + I + \frac{h^2_{E_{\log}}}{h^2_{\mu_{\log}}}I\right]^{-1}\left(\underline{E}_{\log} - w_p^E\left(\underline{E}_{\log} - \underline{\mathcal{S}}_{\log}\right)\right) \\ &= \left[\frac{h^2_{E_{\log}}}{2}\Lambda^{-1}_{\mathcal{S}_{\log}} + \frac{h^2_{\mu_{\log}} + h^2_{E_{\log}}}{h^2_{\mu_{\log}}}I\right]^{-1}\left(\underline{E}_{\log} - w_p^E\left(\underline{E}_{\log} - \underline{\mathcal{S}}_{\log}\right)\right) \\ &= F_E^{-1}\left[\underline{E}_{\log} - w_p^E\left(\underline{E}_{\log} - \underline{\mathcal{S}}_{\log}\right)\right] \end{aligned}$$

where F_E^{-1} is a lowpass filter. However, instead of having unity DC gain, the same reasoning as in the paper reveals that F_E^{-1} has a DC gain of $\frac{h^2_{\mu_{\log}}}{h^2_{\mu_{\log}}+h^2_{E_{\log}}}$. A similar filter, F_μ^{-1} can be found for $\underline{\mathcal{S}}_{\log,2}$ to have a DC gain of $\frac{h^2_{E_{\log}}}{h^2_{\mu_{\log}}+h^2_{E_{\log}}}$. Thus, our fixed point iteration becomes the following

$$\underline{\mathcal{S}}_{\log}^{(k+1)} = F_E^{-1}\left[\underline{E}_{\log} - w_p^E\left(\underline{E}_{\log} - \underline{\mathcal{S}}_{\log}^{(k)}\right)\right] + F_\mu^{-1}\left[\underline{\mu}_{\log} - w_p^\mu\left(\underline{\mu}_{\log} - \underline{\mathcal{S}}_{\log}^{(k)}\right)\right]. \tag{B.5}$$

B.3 Generalized Multiple Parameter MRF Estimation

We can easily generalize the formulation in the previous section to estimate the MRF based on it affecting a set of parameters, Θ. Defining the smooth field, $\underline{f}$, to be additive on the $|\Theta|$ different independent parameters, we can write a set of relationships of 4.12 as follows:

$$\underline{f} = \arg\max_{\underline{f}}\left[\log\prod_{\theta\in\Theta} P\left(\underline{\theta}|\underline{f}\right) + \log P\left(\underline{f}\right)\right],$$

noting that if the field were multiplicative, the log domain could be used. Again, this propagates through most of the proof to the following

$$\sum_{\theta\in\Theta}\frac{2}{h_\theta^2}\left[\underline{\theta} - \underline{f} - w_p^\theta\left(\underline{\theta} - \underline{f}\right)\right] = \Lambda_f^{-1}\underline{f}.$$

We extract the field term from the left side of the equation and simplify as follows:

$$\sum_{\theta\in\Theta}\frac{2}{h_\theta^2}\left[\underline{\theta} - w_p^\theta\left(\underline{\theta} - \underline{f}\right)\right] = \Lambda_f^{-1}\underline{f} + \sum_{\theta\in\Theta}\frac{2}{h_\theta^2}\underline{f} = \left[\Lambda_f^{-1} + I\sum_{\theta\in\Theta}\frac{2}{h_\theta^2}\right]\underline{f},$$

where I is the identity matrix. Solving for $\underline{f}$, we have

$$\begin{aligned}\underline{f} &= \left[\Lambda_f^{-1} + I\sum_{\theta_1\in\Theta}\frac{2}{h_{\theta_1}^2}\right]^{-1}\sum_{\theta\in\Theta}\frac{2}{h_\theta^2}\left[\underline{\theta} - w_p^\theta\left(\underline{\theta} - \underline{f}\right)\right]\\ &= \sum_{\theta\in\Theta}\left(\left[\Lambda_f^{-1} + I\sum_{\theta_1\in\Theta}\frac{2}{h_{\theta_1}^2}\right]^{-1}\frac{2}{h_\theta^2}\left[\underline{\theta} - w_p^\theta\left(\underline{\theta} - \underline{f}\right)\right]\right)\\ &= \sum_{\theta\in\Theta}\left(\left[\frac{h_\theta^2}{2}\Lambda_f^{-1} + I + Ih_\theta^2\sum_{\theta_1\neq\theta}\frac{1}{h_{\theta_1}^2}\right]^{-1}\left[\underline{\theta} - w_p^\theta\left(\underline{\theta} - \underline{f}\right)\right]\right).\end{aligned}$$

We then expand the inner summation as follows

$$
\begin{aligned}
\underline{f} &= \sum_{\theta\in\Theta} \left(\left[\frac{h_\theta^2}{2}\Lambda_f^{-1} + I \left(1 + \frac{h_\theta^2 \sum_{\theta_1\neq\theta} \prod_{\theta_2\neq\theta,\theta_1} h_{\theta_2}^2}{\prod_{\theta_1\neq\theta} h_{\theta_1}^2} \right) \right]^{-1} \left[\underline{\theta} - w_p^\theta \left(\underline{\theta} - \underline{f} \right) \right] \right) \\
&= \sum_{\theta\in\Theta} \left(\left[\frac{h_\theta^2}{2}\Lambda_f^{-1} + I \left(\frac{\prod_{\theta_1\neq\theta} h_{\theta_1}^2 + h_\theta^2 \sum_{\theta_1\neq\theta} \prod_{\theta_2\neq\theta,\theta_1} h_{\theta_2}^2}{\prod_{\theta_1\neq\theta} h_{\theta_1}^2} \right) \right]^{-1} \left[\underline{\theta} - w_p^\theta \left(\underline{\theta} - \underline{f} \right) \right] \right) \\
&= \sum_{\theta\in\Theta} \left(\left[\frac{h_\theta^2}{2}\Lambda_f^{-1} + I \left(\frac{\sum_{\theta_1\in\Theta} \prod_{\theta_2\neq\theta_1} h_{\theta_2}^2}{\prod_{\theta_1\neq\theta} h_{\theta_1}^2} \right) \right]^{-1} \left[\underline{\theta} - w_p^\theta \left(\underline{\theta} - \underline{f} \right) \right] \right)
\end{aligned}
$$

Thus, for a field that affects the parameters in the set Θ, the fixed point update is

$$
\underline{f}^{(k+1)} = \sum_{\theta\in\Theta} F_\theta^{-1} \left[\underline{\theta} - w_p^\theta \left[\underline{\theta} - \underline{f} \right) \right], \tag{B.6}
$$

where F_θ^{-1} is a lowpass filter with DC gain

$$
\text{DC Gain}\left(F_\theta^{-1}\right) = \frac{\prod_{\theta_1\neq\theta} h_{\theta_1}^2}{\sum_{\theta_1\in\Theta} \prod_{\theta_2\neq\theta_1} h_{\theta_2}^2}. \tag{B.7}
$$

We note that all of the DC filter gains sum to 1:

$$
\frac{\sum_{\theta\in\Theta} \prod_{\theta_1\neq\theta} h_{\theta_1}^2}{\sum_{\theta_1\in\Theta} \prod_{\theta_2\neq\theta_1} h_{\theta_2}^2} = 1.
$$

Bibliography

[1] D. Adalsteinsson and J. Sethian. The fast construction of extension velocities in level set methods. *Journal of Computational Physics*, 148:2–22, 1998.

[2] S. M. Ali and S. D. Silvey. A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society*, pages 131–142, 1966.

[3] H. G. Barrow and J. M. Tenenbaum. Recovering intrinsic scene characteristics from images. *Computer Vision Systems*, pages 3–26, 1978.

[4] R. Blahut. Hypothesis testing and information theory. *Information Theory, IEEE Transactions on*, 20(4):405–417, Jul 1974.

[5] P. Brodatz. *Textures: A Photographic Album for Artists and Designers*. Dover Publications Inc., 1966.

[6] T. Brox and J. Weickert. Level set segmentation with multiple regions. *Image Processing, IEEE Transactions on*, 15(10):3213–3218, Oct. 2006.

[7] R. Buccigrossi and E. Simoncelli. Progressive wavelet image coding based on a conditional probability model. *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, 4:2957, 1997.

[8] T.F. Chan and L.A. Vese. An efficient variational multiphase motion for the mumford-shah segmentation model. *Signals, Systems and Computers, 2000. Conference Record of the Thirty-Fourth Asilomar Conference on*, 1:490–494 vol.1, 2000.

[9] T.F. Chan and L.A. Vese. Active contours without edges. *Image Processing, IEEE Transactions on*, 10(2):266–277, Feb 2001.

[10] T. Chang and C.-C.J. Kuo. Texture analysis and classification with tree-structured wavelet transform. *Image Processing, IEEE Transactions on*, 2(4):429–441, Oct 1993.

[11] Jeremy S. De Bonet and Paul Viola. A non-parametric multi-scale statistical model for natural images. In *NIPS '97: Proceedings of the 1997 conference on Advances in neural information processing systems 10*, pages 773–779, Cambridge, MA, USA, 1998. MIT Press.

[12] J. M. H. Du Buf and P. Heitkämper. Texture features based on gabor phase. *Signal Process.*, 23(3):227–244, 1991.

[13] R. M. Fano. Class notes for transmission of information, couse 6.574. 1959.

[14] H. Farid. Blind inverse gamma correction. *Image Processing, IEEE Transactions on*, 10(10):1428–1433, Oct 2001.

[15] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 13(9):891–906, 1991.

[16] W.T. Freeman and E.H. Adelson. The design and use of steerable filters. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 13(9):891–906, Sep 1991.

[17] D. Gabor. Theory of communication. *JIEE*, 93(3):429–459, 1946.

[18] Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

[19] M. Grayson. The heat equation shrinks embedded plane curves to round points. *Journal of Differential Geometry*, 26(2):285–314, 1987.

[20] Leslie Greengard and John Strain. The fast gauss transform. *SIAM Journal on Scientific and Statistical Computing*, 12(1):79–94, 1991.

[21] M. Heiler and C. Schnorr. Natural image statistics for natural image segmentation. *Computer Vision, 2003. Proceedings Ninth IEEE International Conference on*, pages 1259–1266 vol.2, Oct. 2003.

[22] Berthold K. P. Horn. Obtaining shape from shading information. pages 123–171, 1989.

[23] N. Houhou, J.-P. Thiran, and X. Bresson. Fast texture segmentation model based on the shape operator and active contour. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008.

[24] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, V1(4):321–331, January 1988.

[25] J. Kiefer. Sequential minimax search for a maximum. *Proceedings of the American Mathematical Society 4*, pages 502–506, 1953.

[26] Junmo Kim, III Fisher, J.W., A. Yezzi, M. Cetin, and A.S. Willsky. A nonparametric statistical method for image segmentation using information theory and curve evolution. *Image Processing, IEEE Transactions on*, 14(10):1486–1502, Oct. 2005.

[27] S. Kullback. *Informaiton Theory and Statistics*. John Wiley & Sons, New York, 1959.

[28] W. T. Freeman M. F. Tappen and E. H. Adelson. Recovering intrinsic images from a single image. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(9):1459–1472, 2005.

[29] Jitendra Malik and Ruth Rosenholtz. Computing local surface orientation and shape from texture for curved surfaces. *International Journal of Computer Vision*, 23(2):149–168, 1997.

[30] J. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill-posed problems in computational vision. *Journal of the American Statistical Association*, 82(397):76–89, March 1987.

[31] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings 8th International Conference on Computer Vision*, volume 2, pages 416–423, July 2001.

[32] Geoffrey J. Mclachlan and Thriyambakam Krishnan. *The EM Algorithm and Extensions*. Wiley-Interscience, October 2007.

[33] T. Mitsunaga and S. K. Nayar. Radiometric self calibration. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:1374, 1999.

[34] Javier A. Montoya-Zegarra, Neucimar J. Leite, and Ricardo da S. Torres. Rotation-invariant and scale-invariant steerable pyramid decomposition for texture image retrieval. In *SIBGRAPI '07: Proceedings of the XX Brazilian Symposium on Computer Graphics and Image Processing*, pages 121–128, Washington, DC, USA, 2007. IEEE Computer Society.

[35] Xuanlong Nguyen, Martin J. Waonwright, and Michael I. Jordan. On surrogate loss functions and f-divergences. *Annals of Statistics*, 37(2):876–904, 2009.

[36] Stanley Osher and Ronald Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 31 October 2002.

[37] E. Parzen. On the estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

[38] Javier Portilla and Eero P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40(1):49–70, 2000.

[39] D. W. Scott. *Multivariate Density Estimation: Theory, Practice, and Visualization*. John Wiley & Sons, 1992.

[40] J.A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge University Press, second edition, 13 June 1999.

[41] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC, April 1986.

[42] Eero P. Simoncelli and William T. Freeman. The steerable pyramid: A flexible architecture for multi-scale derivative computation. In *International Conference on Image Processing*, volume 3, pages 444–447, 23-26 Oct. 1995, Washington, DC, USA, 1995.

[43] Eero P. Simoncelli, William T. Freeman, Edward H. Adelson, and David J. Heeger. Shiftable multi-scale transforms. *Information Theory, IEEE transactions on*, 38(2), 1992.

[44] Ping sing Tsai and Mubarak Shah. Shape from shading using linear approximation. *Image and Vision Computing*, 12:487–498, 1994.

[45] B.J. Super and A.C. Bovik. Shape from texture using local spectral moments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(4):333–343, Apr 1995.

[46] A. Teuner, O. Pichler, J.E. Santos Conde, and B.J. Hosticka. Orientation- and scale-invariant recognition of textures in multi-object scenes. *Image Processing, 1997. Proceedings., International Conference on*, 3:174–177 vol.3, Oct 1997.

[47] Harry L. Van Trees. *Detection, Estimation, and Modulation Theory, Part 1*. Wiley, New York, 1968.

[48] J. Tsitsiklis. Efficient algorithms for globally optimal trajectories. *Automatic Control, IEEE Transactions on*, 40(9):1528–1538, Sept 1995.

[49] A.G. Weber. The USC-SIPI image data base: Version 5. *USC-SIPI Report No. 315*, 1997.

[50] Y. Weiss. Deriving intrinsic images from image sequences. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 68–75 vol.2, 2001.

[51] III Wells, W.M., W.E.L. Grimson, R. Kikinis, and F.A. Jolesz. Adaptive segmentation of mri data. *Medical Imaging, IEEE Transactions on*, 15(4):429–442, Aug 1996.

[52] C. Yang, R. Duraiswami, N. Gumerov, and L. Davis. Improved fast gauss transform and efficient kernel density estimation. *Computer Vision, IEEE International Conference on*, pages 464–471, 2003.

[53] Song Chun Zhu and A. Yuille. Region competition: unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(9):884–900, Sep 1996.