Shape-Tailored Invariant Descriptors for Segmentation

Dissertation by

Naeemullah Khan

In Partial Fulfillment of the Requirements

For the Degree of

Doctor of Philosophy

King Abdullah University of Science and Technology

Thuwal, Kingdom of Saudi Arabia

September, 2018

EXAMINATION COMMITTEE PAGE

The Dissertation of Naeemullah Khan is approved by the examination committee

Committee Chairperson: Professor Ganesh Sundaramoorthi
Committee Co-Chairperson: Professor Peter Wonka
Committee Members: Professor Wolfgang Heidrich, and Professor Marc Niethammer

ABSTRACT

Shape-Tailored Invariant Descriptors for Segmentation

Naeemullah Khan

Segmentation is one of the first steps in human visual system which helps us see the world around us. Humans pre-attentively segment scenes into regions of unique textures in around 10-20 ms. In this thesis, we address the problem of segmentation by grouping dense pixel-wise descriptors. Our work is based on the fact that human vision has a feed forward and a feed backward loop, where low level feature are used to refine high level features in forward feed, and higher level feature information is used to refine the low level features in backward feed. Most vision algorithms are based on a feed-forward loop, where low-level features are used to construct and refine high level features, but they don't have the feed back loop. We have introduced "Shape-Tailored Local Descriptors", where we use the high level feature information (region approximation) to update low level features i.e. the descriptor, and the low level feature information of the descriptor is used to update the segmentation regions. Our "Shape-Tailored Local Descriptor" are dense local descriptors which are tailored to an arbitrarily shaped region, aggregating data only within the region of interest. Since the segmentation, i.e., the regions, are not known a-priori, we propose a joint problem for Shape-Tailored Local Descriptors and Segmentation (regions).

Furthermore, since natural scenes consist of multiple objects, which may have different visual textures at different scales, we propose to use a multi-scale approach to segmentation. We have used a set of discrete scales, and a continuum of scales in our experiments, both resulted in state-of-the-art performance.

Lastly we have looked into the nature of the features selected, we tried hand-crafted color and gradient channels and we have also introduced an algorithm to incorporate learning optimal descriptors in segmentation approaches. In the final part of this thesis we have introduced techniques for unsupervised learning of descriptors for segmentation. This eliminates the problem of deep learning methods where we need huge amounts of training data to train the networks. The optimum descriptors are learned, without any training data, on the go during segmentation.

ACKNOWLEDGEMENTS

بسم الله الرحمن الرحيم

الحمد لله رب العالمين والصلاة والسلام على رسول الله صلى الله عليه وسلم

I would like to thank Prof. Ganesh Sundaramoorthi for his support and appreciation throughout my stay at Kaust. I would also like to thank Prof. Marc Niethammer, Prof Peter Wonka, and Prof Wolfgang Heidrich for being a part of the thesis committee and for their useful feedback about this thesis.

I would like to thank my parents, who have spent an enormous amount of time and energy pushing me to be the best that I can be. Their sacrifices is the reason of all my success. I would like to thank and acknowledge my siblings who have always been there to help me through any situation that I faced over the years. I would also like to thank my wife, who had to bear my long working hours and the tough schedule of my graduate life. I would also like to thank my niece Anum Khan and my daughter Fatima Khan who has been a ray of sunshine and a source of constant joy for us.

Last but not the least, I would like to thank my friends for their continuous encouragement and support over the years. Particularly, Mr. Sultan Albarakati, who made me feel at home in a foreign country and was always willing to extend the famous Arab hospitality whenever I needed it.

This work is dedicated to the memory of the children of Army Public School Peshawar.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Computer Vision is the field of designing algorithms for machines to extract useful information from images and videos. Human visual tract is a complex neurological system, in which the intensity input from environment is interpreted as useful information. Humans have a remarkable ability to recognize objects and localize them in natural scenes, hence object recognition is also a central problem in computer vision [3]. In order to correctly localize and recognize objects, we need to first know where the different objects are in images and videos. This is achieved through segmentation, where we divide image into region of unique visual properties (surface properties and textures). More complex vision application like object detection, object tracking and recognition can then be built on top of the segmentation results.

One of the most important property of objects for segmentation is surface texture. Texture are important in our perception of the world around us. It is one of the first cues in human vision which allows human to pre-attentively segment objects in a scene. Textures therefore play a key role in segmentation algorithms. A good segmentation algorithm should be able to discriminate between different textures while being invariant to complex nuisances of textures.

In this work we will try to tackle the problem of segmentation, i.e. breaking an image down in to regions of similar looking textures and regions, which is important in computer vision (e.g., for defining a mid-level representation) [4, 5, 6], computer graphics (e.g., as a base step for texture synthesis) [7, 8], and in understanding human visual perception [9].

Segmentation has applications in medical imaging e.g. Pujol and Radeva [10] used segmentation in intravascular ultrasound images to segment the lumen (blood) and the plaque (tissue). Segmentation also has application in aerial image analysis, document processing and surface quality inspection. In [11] Dubuisson-Jolly and Gupta applied color and texture fusion to aerial image segmentation. Their algorithm uses maximum likelihood classification combined with a certainty based fusion criterion. A detailed discussion of applications of segmentation can be found in [12].

One of the major approaches in vision for segmentation is local invariant descriptors. Local invariant descriptors (e.g., [5, 13, 14, 15, 16]) are image statistics at each pixel that describe neighborhoods in a way that is invariant to geometric and photometric nuisances. They are typically computed by aggregating smoothed oriented gradients within a neighborhood of the pixel. These descriptors play an important role in characterizing local textural properties. This is because a texture consists of small tokens, called textons [9], which may vary by small geometric and photometric nuisances but are otherwise stationary. Careful construction of these descriptors is crucial since they play a key role in low-level segmentation, which in turn plays a role in higher level tasks such as object detection and segmentation.

Existing local invariant descriptors aggregate oriented gradients in predefined pixel neighborhoods that could contain image data from different textured regions, especially near the boundary of the texture. This leads to ambiguity in grouping descriptors, especially for descriptors near the boundary. This could lead to segmentation errors if descriptors are grouped to form a segmentation. The problem is exacerbated when the textons in the textures are large. In this case, the neighborhood of the descriptor needs to be chosen large to fully capture texton data. See Fig. 1.1. Ideally, one would need to construct local descriptors that aggregate oriented gradients only from within textured regions. However, the segmentation is not known a-priori. Thus, it is necessary to solve for the local descriptors and the region of the segmentation in

a joint problem.

To construct descriptors on the region of interest of textured regions, we propose the use of PDE based descriptors where information is aggregated on the region of interest. We solve a Poisson equation with Neumann boundary conditions on the region of interest to get the descriptors. This allows us to construct shape-tailored descriptors. A joint energy is constructed on these descriptors where descriptors and segmentation is jointly evaluated to get the segmentation.

Next, we will consider the question of scale selection in segmentation. An image consists of many different structures at different *scales*, and thus the notion of *scale space* [17], which consists of blurs of the image at all degrees, has been central to computer vision. The need for incorporating scale space in segmentation is well-recognized [18]. Further, there is evidence from human visual studies (e.g., [19, 20]) that the coarse scale, i.e., from high levels of blurring, is predominantly processed before the fine scale. This *coarse-to-fine* principle has led to many efficient algorithms that are able to capture the coarse structure of the solution, which is often most important in computer vision. Therefore, it is natural for segmentation algorithms to use scale space and operate in a coarse-to-fine fashion.

Existing methods for segmentation that incorporate scale have either one of the following limitations. First, most segmentation methods (e.g., [21, 22, 23]) based on scale spaces consider *global* scale spaces that are computed on the whole image, which does not capture the fact that there exist multiple *regions* of the segmentation at different scales, and this could lead to the removal and/or displacement of important structures in the image, for instance, when large structures are blurred across small ones, leading to an inaccurate segmentation. Second, algorithms that use a coarse-to-fine principle (e.g., [24, 25]) do so *sequentially* (see Figure 1.2) so that the algorithm operates at the coarser scale and then uses the result to initialize computation at a finer scale. While this warm start may influence the finer scale result, there is

no guarantee that the coarse structure of the segmentation is preserved in the final solution.

In this thesis, we also develop an algorithm that simultaneously addresses these two issues. Specifically, we formulate a novel multi-region energy for segmentation, which integrates a *continuum* of scales from *Shape-Tailored Scale Spaces*. These scale spaces are defined within regions of the segmentation, and thus they prevent removal or displacement of important structures. By integrating over a *continuum* of scales of the scale space determined by the heat equation, we show that this energy has preference to coarse structure of the data without ignoring the fine structure. We show that it operates in a *parallel* coarse-to-fine fashion (see Figure 1.2). That is, it is initially dominated by the coarse structure of the data, then segments finer structure of the data, while preserving the structure from the coarse-scale of the data. We provide analytic solutions for the optimization of the energy, which leads to a computationally more efficient method than similar energies integrating discrete scales. We apply our algorithm to the problem of texture segmentation, and show that our method outperforms discrete scale spaces and existing state of the art. We also apply our method to motion segmentation, the results show the advantage of the shape-tailored continuum scale space, and show out-performance against existing state of the art.

Next, in this thesis, we address the problem of constructing descriptors *invariant* to complex nuisances yet discriminative of textures, while aggregating image statistics only within regions of interest. To do this, we use the *base* shape-tailored descriptors of [26], which are color channels and oriented gradients at various scales defined through PDEs, and learn a function of such base descriptors that is invariant to more complex nuisances than the limited invariances to small contrast and small geometric distortions that the base descriptors possess. By learning a function of base shape-tailored descriptors, we automatically inherit the shape-tailored property, i.e., that the

learned descriptors aggregate image statistics only within regions of interest. Thus, they are naturally suited for a joint problem in the segmentation and the descriptors.

The problem we wish to address does not fit into a labeling problem, where one labels each pixel in the image according to pre-defined labels, representing certain categories in a training set. In particular, we do not wish to segment classes of textures (e.g., different types of tree barks or different types of sea shells should not be labeled the same). Since the set of textures in the natural world is enormous, perhaps not even enumerable, it is infeasible to construct a training set with samples of each texture labeled. Further, we wish to segment textures that are not even in the training set. Instead of associating a class label to each texture, we aim to learn *generic* descriptors, beyond just textures or classes of textures in the training set, by learning a *metric* to discriminate textures by their base shape-tailored descriptors. By learning a metric to discriminate textures, the training set only needs to consist of ground truth segmentations of textures and not class labels. The aim is, by choice of appropriate regularization in the learning method, the metric and hence the descriptor generalizes beyond the training set and learns generic properties of all textures. The learned descriptor is the output of a fully connected neural network whose input is a base shape-tailored descriptor. The metric is formed from a Siamese network [27] composed of the aforementioned neural network, and a weighted $\mathbb{L}^2$ norm between the output of each component of the Siamese network.

1.1 Objectives and Contributions

In this work, we answer the following questions: 1. Assuming that the segmentation region is known, how does one compute local invariant descriptors (aggregations of oriented gradients at various scales/neighborhoods) so that data is aggregated only within the region (shape)? 2. Since the segmentation region is unknown, how does one determine the region? 3. What are right scales to use to get the correct

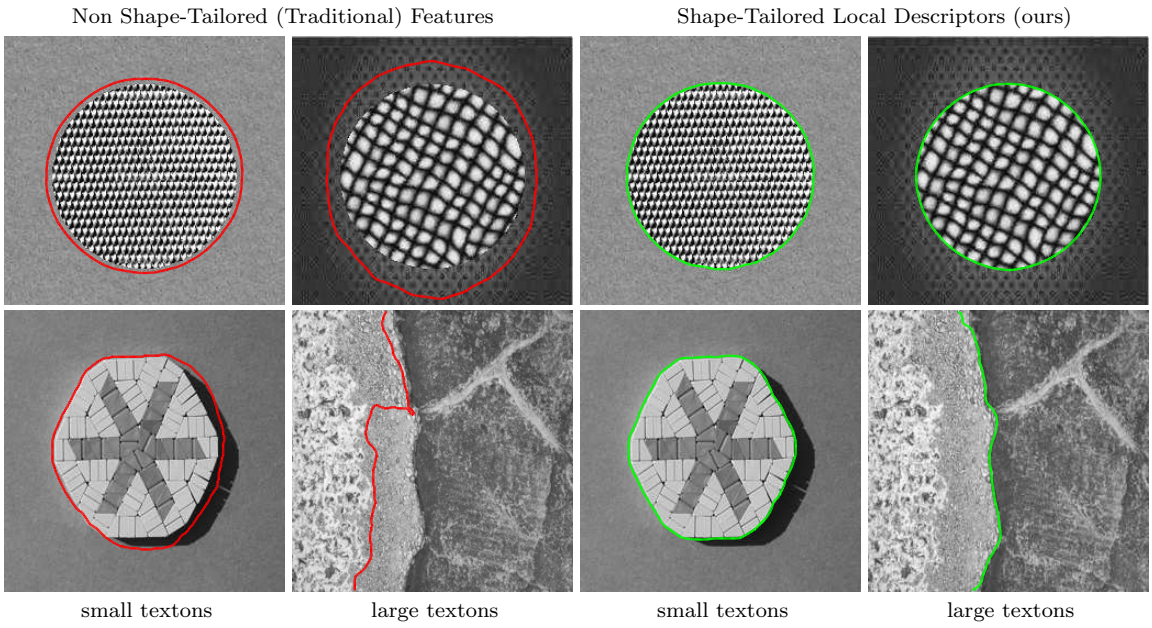

Figure 1.1: **Descriptors Not Tailored to Shape Lead to an Inaccurate Segmentation.** Descriptors that aggregate image data that have no knowledge of the boundaries of different texture regions lead to erroneous segmentation, and the problem is exacerbated as the texton size increases. [Left]: Segmentation using Descriptors that are not aware of boundaries. [Right]: Segmentation by Shape-Tailored Local Descriptors solved as joint problem.

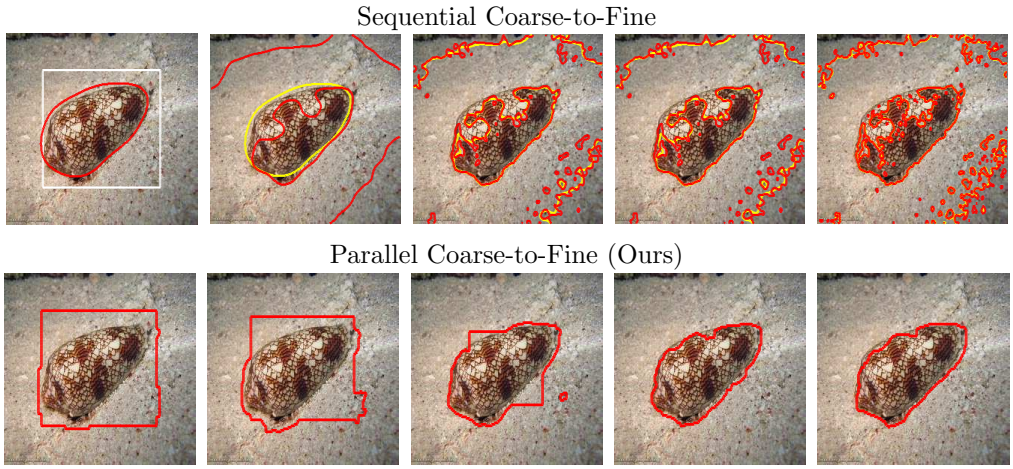

Figure 1.2: **Sequential and Parallel Coarse to Fine Segmentation Comparison.** [Top]: Sequential coarse-to-fine methods use the result of segmentation (red) from the coarse scale to initialize (yellow) the finer scales, and may lose coarse structure of the coarse segmentation solution without additional heuristics. Note that the result of segmentation of the coarse scale is the left image in red (the blurred image is not shown), and towards the right segmentation is done at finer scales. [Bottom]: Our parallel coarse-to-fine approach considers a continuum of scales all at once and has a coarse-to-fine property. The evolution is shown from left to right.

coarse-to-fine segmentation? 4. How to get optimum learned descriptors from hand-crafted channels? 5. How to setup unsupervised learning setup for learning invariant descriptors?

We introduce *Shape-Tailored Local Descriptors* to answer the first question. Shape-Tailored Local Descriptors are solutions of PDEs defined on arbitrarily shaped regions. They are constructed using PDEs since PDEs generalize scale-space to arbitrarily shaped regions, and PDEs provide a convenient framework, whereby the shape-tailored neighborhoods in which to aggregate data are implicitly defined, and never needed to be computed explicitly. Question two is answered with a joint problem for "Shape-Tailored Local Descriptors" and the region. Defining descriptors with PDEs allows one to derive an evolution equation for the estimated region and descriptors (implicitly evolving the neighborhoods) see Fig. 1.3. For question three, we show the continuum scale space of Heat equations manifest a natural coarse-to-fine behavior. Furthermore we formulate an energy based on the continuum scale space of Heat

equation where the scale space does not need to be computed explicitly and only a Poisson equation at native scale is used to evolve the region and get the segmentation. For question number four, we provide a learning strategy using which an optimum non linear combination of hand-crafted descriptors is learned for segmentation task using Siamese twin network where each component of the Siamese twin is composed to fully connected layers to learn the optimum non-linear combination of the input shape-tailored descriptors. For question 5, we construct an energy where the invariant descriptor learning and the segmentation evolution are performed jointly.

The contributions of this thesis folds in the following streams:

• We introduce shape-tailored local descriptors that are local descriptors tailored to an arbitrary shaped region. Eliminating the problems generated by aggregating data across the texture boundaries irrespective of the size of the neighborhood chosen for the local descriptor.

• We compute the shape variation for shape-tailored local descriptors. The change in the value of the local descriptor with a slight perturbation of the boundary of the region. Notice that the Shape-Tailored local descriptors are the function of the boundary of the region because data is aggregated only on the region. A perturbation of the boundary of the region will results in change in the descriptor.

• We construct and optimize a joint energy for segmentation and shape-tailored local descriptors. Shape-tailored local descriptors solve the problem (of traditional local descriptors) created because of aggregating data across the texture boundaries. But the problem of finding the right region to aggregate data from, still remains. To address this we solve the problem of region and descriptors in a joint manner.

• We look at the continuum shape-tailored scale space from heat equation in segmentation and show that it has a natural coarse-to-fine property.

• We show how to construct *learned* shape-tailored descriptors, descriptors that aggregate image statistics only within arbitrary shaped regions of interest and are

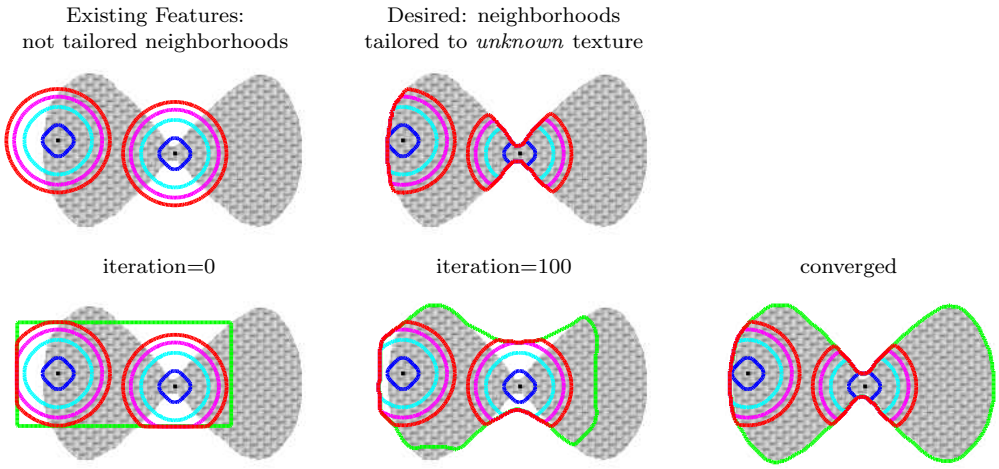

Figure 1.3: **How Does One Compute Descriptor for an Unknown Textured Region?** [Top Block, left image]: Existing local descriptor aggregate data from neighborhoods with no knowledge of texture boundaries. [Top Block, right image]: Desired descriptors should aggregate data from neighborhoods not crossing texture boundaries. [Bottom Block]: our approach starts with an initial estimate of textured region (green), computes descriptors by aggregating data from neighborhoods tailored to the region, updates the region estimate, recomputes the descriptors, until the descriptors are only computed within the textured region.

invariant to complex photometric and geometric nuisances yet discriminative for segmentation. The shape-tailored property is necessary so that segmentation by grouping descriptors can be accomplished, and the invariance is needed to segment textures plagued by nuisances. Invariance is accomplished by learning it from training data.

- We formulate grouping of learned shape-tailored descriptors as a joint optimization problem for the segmentation and descriptors, and derive the optimization algorithm.

- We formulate an energy where the invariant shape-tailored descriptors are learned and the segmentation is evolved jointly. We don't need any training data and the learning is completely unsupervised.

1.2 Related Work

Many approaches [28, 29, 30, 31, 32, 33, 34] to segmentation partition the image into regions that have global intensity distributions that are maximally separated by a distance on distributions. A drawback of global intensity distributions is that spatial relations are lost. This is important in characterizing textures. Spatial correlations between neighboring pixels are considered in [35] by creating a vector of the four neighboring pixel values for each pixel. Grouping these vectors improves segmentation. A recent approach [36] uses frequencies of neighboring pixel pairs within the image to determine texture boundaries. In [37], small neighborhoods are obtained from a super-pixelization and used in segmentation. Super-pixels may cross texture boundaries, aggregating data across boundaries.

Larger neighborhoods are considered in [38]. Gabor filters at various scales and orientations have been used widely in texture analysis (e.g., [5]), and the response of these filters (or others [39, 40]) have been used as a descriptor in texture segmentation (e.g., [41, 42]), and as an edge-detector [43]. These approaches depend on the size of the neighborhood chosen. The optimal size is determined by peaks in the entropy profile of intensity distributions of increasingly sized neighborhoods in [44, 45, 46]. An aspect that remains an issue in all these methods is that neighborhoods may cross texture boundaries, which our method addresses. In [47], these boundary effects are mitigated by a top-down correction step, however, the method only deals with neighborhoods that are a few pixels in length.

We use variational methods to optimize the Mumford-Shah energy incorporating our shape-tailored local descriptors. Many active contours based methods[48] are driven to group pixel intensities based on intensity statistics. For example, global intensity means in the regions are used in [49, 50], and global histograms are used in [30, 32]. Since images are not always described by global intensity statistics, local intensity statistics have been used to group pixels (e.g., [51, 52, 53, 54]). Since these

methods aim to group pixels, they do not capture texture in many cases. These energies are optimized using gradient descent, but more recently methods of convex relaxations have improved results in many cases [55, 56, 57].

Our Shape-Tailored descriptors are the solutions of PDE defined within regions. Thus, the energies we optimize involve integrals over the regions of functions of PDE that are dependent on the regions. While we use direct methods of calculus of variations to optimize these energies, one can also use shape gradients [58] (see also, [59, 60]). Our contribution lies in introducing new descriptors for segmentation, and not in the method of optimization.

Scale space theory [17, 61, 62, 63] has a long and rich history as a theory for analyzing images, and we only provide brief highlights. The idea is that an image consists of structures at various different scales (e.g., a leaf of a tree exists at a different scale than a forest), and thus to analyze an image without a-priori knowledge, it is necessary to consider the image at *all* scales. This is accomplished by blurring the image at a continuum of kernel sizes. The most common kernel is a Gaussian, which is known to be the only scale space satisfying certain axioms such as not introducing any new features as the image is blurred [64]. Scale space has been used to analyze structures in images (e.g., [65, 66, 64, 67]). This has had wide ranging applications in stereo and optical flow [68], reconstruction [69, 70], key-point detection in wide-baseline matching [13], design of descriptors for matching [71], shape matching [72], and curve evolution [73], among others.

Gaussian scale spaces have also been used in image segmentation, most notably in texture segmentation [47, 29, 21, 22, 74], which occur frequently in natural images [23]. While these methods capture important scale information, they use a *global* scale space defined on the entire image, which does not capture the characteristic scale of features within regions and blurs across segmentation boundaries. Anisotropic scale spaces [18, 75] have been applied to reduce blurring across boundaries, but this could

blur across regions where edges are not salient. Recently, [26] have addressed this issue by computing scales locally within the evolving regions of the segmentation. However, only a discrete number of scales are used and thus the method does not exhibit coarse-to-fine behavior. Such methods for segmentation have been numerically implemented with various optimization methods, including level sets [76], convex methods [57, 77], and others [78]. The energy we consider is not convex, and thus we rely on gradient descent on curves. The energy we consider involves optimization with partial differential equation (PDE) constraints, and thus we build on optimization methods from [59, 58].

Coarse-to-fine methods, where coarse representations of the image or objective function are processed and then finer aspects of the data are successively revealed, have a long history in computer vision [24]. In these methods, data or the objective function is smoothed, and the smoothed problem is solved. The result is used to initialize the problem with less smoothing, where finer details of the data are revealed. The hope is that this finer result retains aspects of coarse solution, while gradually finding finer detail. However, without additional heuristics such as restricting the finer solution to be around the solution of the coarse problem, there is no guarantee that coarse structure is preserved when solving the finer problem. Recently, [25] provided analysis and derived closed form solutions for the smoothing of the objective in problems of point cloud matching. Our method uses a single energy integrating over a continuum of scales in *parallel*, rather than a sequential approach where multiple energies from coarse to fine are solved. This guarantees that the coarse and fine scale aspects of the desired solution are obtained.

Since we also apply our method to the problem of segmenting moving objects in video based on motion, we highlight some aspects of that literature most relevant to this work. Methods for motion segmentation are based on optical flow (e.g., [79]). Piecewise parametric models for motion of regions in segmentation are used in e.g.,

[80, 81]. Non-parametric warps are used for motion models (e.g.,[82, 83, 84]). Our goal here is *not* to estimate motion, but rather we use existing techniques for motion estimation, and improve the segmentation of regions by merely replacing a single scale formulation with our novel continuum scale space approach.

Existing approaches for segmentation can be roughly divided into learning based approaches and "hand-crafted" approaches. Further, hand-crafted approaches can be divided into edge-based and region-based approaches. Some region-based approaches attempt to partition the image into regions that have global intensity distributions that are maximally separated [28, 30, 34]. Since spatial relations are lost, other approaches have tried to incorporate spatial relations by considering distributions of pairs or neighborhoods of pixels (e.g., [36]). Larger neighborhoods are described, by for instance the output of Gabor filters at various scales and orientations [5], and grouped in other approaches for texture segmentation [40, 42, 44]. However, such approaches are affected by the problem that describing neighborhoods without knowing or having an estimation of the segmentation is prone to errors as neighborhoods that aggregate statistics across segmentation boundaries are difficult to group. This problem was addressed by [26], who formulated the estimation of descriptors and segmentation as a joint problem. However, the descriptors constructed in [26] were hand-crafted, and do not exhibit invariances to complex nuisances.

Edge-based approaches (e.g., [43]) attempt to locate edges as a response to a filter bank. [43] use a filter bank of Gabor filters among other hand-crafted filters. Such responses are then post-processed to fill gaps, and generate a segmentation. Learning-based approaches to edge detection have been shown to achieve better results [1, 85, 2]. Such approaches have used deep learning to derive a probability that a pixel belongs to boundaries between segments. While these approaches achieve impressive results, still a difficulty remains in generating the segmentation from edges, which still rely on hand-crafted approaches [43] and the problem remains not fully solved. Alternatively,

region-based approaches, like our method, solve directly for the regions, and avoid this problem. However, they have the problem of selecting the correct number of regions, which cannot be fully addressed without a hierarchy of segmentations. Our approach addresses one of the problems in region-based approaches, that of learning descriptors to group. Our approach is the first to address this problem to the best of our knowledge.

There has been recent interest in methods for semantic segmentation using deep learning [86, 87, 88]. These approaches aim to label each pixel in the image as semantically distinct objects from a pre-defined set of objects. Such approaches achieve impressive results. However, they are limited to object classes in the training set, and it would be difficult to apply this approach to our problem of texture segmentation, as the set of textures that we wish to segment is probably not even enumerable.

Recently there has been a huge body of work applying deep learning to different vision problems [89, 1, 90, 85, 91, 92, 86, 87, 88, 93, 94, 95, 2], Segmentation being one of them. Semantic pixel-wise segmentation is an active research topic in vision [96]. Traditional methods for segmentation related tasks uses hand-engineered features for classifying pixels [97]. Patch descriptors can then be input to the classifier e.g. Boosting or Random forest to predict the class probability of the center pixel.

Recently deep convolutional Neural Network have shown great success in classification related tasks and this has encouraged researchers to exploit the optimum feature learning properties of the CNN for structured prediction problems like segmentation. The other main deep learning approach is Recurrent Neural Network where several low resolution predictions are merged to create image size prediction map. These methods show improved performance over hand engineered features but show poor performance in localizing exact boundaries because of pooling operations.

There has also been several attempts at training deep networks to find edges in images[2, 1, 89, 90]. These deep networks for edges have shown state-of-the-art result

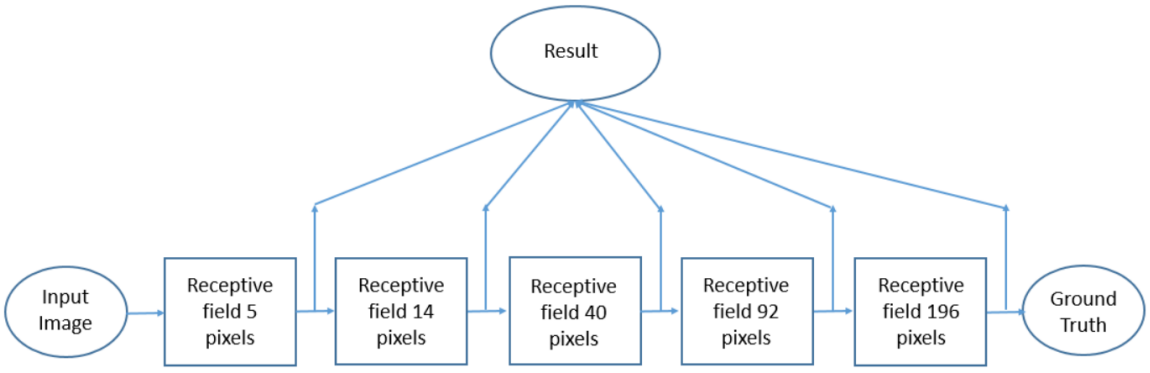

Figure 1.4: **Illustration of Holistically-Nested Edge Detection Method. Image based on network from [1]**

on BSD dataset for boundary matric [43]. [2] also showed state of the art results on BSD dataset and PASCAL VOC 2012 [98] on boundary detection task. The numbers obtained by these methods are comparable to human perception. A key point to notice here is that the boundary metric on BSD is determined for general boundaries of the dataset and not the boundaries of unique objects. The performance of human observers rises significantly when asked to identify the boundaries of objects as shown by [99]. This issue typical gives the illusion for edge-based methods that they are comparable to human perception on boundary datasets[98, 43], this information has to be taken with a grain of salt as the ground-truth for these dataset is designed for all edges and not object boundaries hence the slightly lower number for human perception.

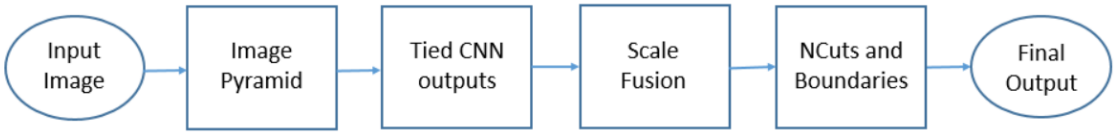

Figure 1.5: **Pushing the Boundaries of Boundary Detection Using Deep Learning. Image based on network architecture of [2]**

1.3 Structure of the thesis

In **Chapter 2**, we introduce shape-tailored local descriptors that are used in this thesis for segmentation and tracking related tasks. In **Chapter 3**, we formulate a joint energy for region and descriptor that uses our shape-tailored local descriptors in segmentation. In **Chapter 4**, we present continuum Gaussian scale space which gives a coarse-to-fine property in segmentation. In **Chapter 5**, we present a learning approach to learn invariant shape-tailored local descriptors for segmentation. In **Chapter 6** we extend the learned shape-tailored invariant descriptors to unsupervised case where we don't need any training data to learn the descriptors. In **Chapter 7**, we detail our experiment for the synthetic texture segmentation, real-world texture segmentation, and motion segmentation datasets. This include performance and accuracy analysis. In **Chapter 8**, we present the conclusion of this work, we identify the key contribution, challenges, drawback and the future direction of this work.

Chapter 2

Shape-Tailored Descriptors Formulation

In this chapter[1], we define Shape-Tailored Descriptors. We compute their gradient with respect to shape perturbations, and then the gradient of a region-based functional involving the descriptors. These results will be needed to optimize the energy for segmentation

2.1 Defining Shape-Tailored Descriptors

Let $\Omega \subset \mathbb{R}^2$ be the domain of an image $I : \Omega \to \mathbb{R}^k$ ($k \geq 1$). Let $R \subset \Omega$ be an arbitrarily shaped region with non-zero area and smooth boundary ∂R. We compute local descriptors for each $x \in R$. The descriptor describes I in a neighborhood of x inside R. The descriptors at $x \in R$ will be aggregations of image data I and oriented gradients within multiple neighborhoods of x in R. This can be accomplished conveniently using scale-spaces [101] defined by PDE. This motivates the definition below.

Definition 1 (Shape-Tailored Local Descriptors)**.** *Let $R \subset \Omega$ be a bounded region with non-zero area and smooth boundary ∂R. Let $I : \Omega \to \mathbb{R}^k$. A* **Shape-Tailored Descriptor***, $\mathbf{u} : R \to \mathbb{R}^M$ (where $M = n \times m$, $n, m \geq 1$) consists of components $u_{ij} : R \to \mathbb{R}$ so that $\mathbf{u} = (u_{11}, \ldots, u_{1m}, \ldots, u_{n1}, \ldots, u_{nm})^T$. The components are*

[1]Work presented in this chapter is based on [26, 100]

defined as:

$$\begin{cases} u_{ij}(x) - \alpha_i \Delta u_{ij}(x) = J_j(x) & x \in R \\ \nabla u_{ij}(x) \cdot N = 0 & x \in \partial R \end{cases}, \tag{2.1}$$

where $1 \leq i \leq n$, $1 \leq j \leq m$, Δ *denotes the Laplacian,* ∇ *denotes the gradient,* N *is the unit outward normal to* R, $\alpha_i > 0$ *are* scales, *and* $J_j : R \to \mathbb{R}$ *are point-wise functions of the image* I. *In vector form, this is equivalent to*

$$\begin{cases} \mathbf{u}(x) - A\Delta\mathbf{u}(x) = \mathbf{J}(x) & x \in R \\ D\mathbf{u}(x)N = \mathbf{0} & x \in \partial R \end{cases}, \tag{2.2}$$

where $A = diag(\alpha_1 1_{1\times m}, \ldots, \alpha_n 1_{1\times m})$ *(an* $M \times M$ *diagonal matrix),* $1_{1\times m}$ *is a* $1 \times m$ *matrix of ones,* D *denotes the spatial derivative operator, and* $\mathbf{J} = (J_1, \ldots, J_m, \ldots, J_1, \ldots, J_m, \ldots)^T$.

Remark 2.1.1. *Possible choices for* $\mathbf{J}$ *can include oriented gradients of the gray-scale value of* I, *color channels of* I, *and the grayscale image* I_g. *Note oriented gradients of the grayscale image* I_g, *for an angles* θ_i *are defined as* $I_{\theta_i}(x) := \int_{\theta_i}^{\theta_i+\Delta\theta} |\nabla I_g(x) \cdot e_{\theta'}| \, \mathrm{d}\theta'$ *where* e_θ *indicates a unit direction vector in the direction of* θ, $|\cdot|$ *is absolute value, and* $\Delta\theta > 0$ *is the angle bin size. Unless otherwise specified, we choose* J_j*'s to be the color channels and oriented gradients at angles* $\theta = \{0, \pi/8, 2\pi/8, \ldots, 7\pi/8\}$.

Remark 2.1.2. *The PDE* (2.1), *for each* θ, *form a scale space with scale parameter* α_i. *The PDE is the minimizer of*

$$E(u) = \int_R (J_j(x) - u(x))^2 \, \mathrm{d}x + \alpha_i \int_R |\nabla u(x)|^2 \, \mathrm{d}x.$$

Thus, u_{ij} *is a smoothing of* J_j *and* α_i *controls the amount of smoothing. Using the Green's function* K_{α_i}, *to be introduced in Section 2.2,* $u_{ij}(x) = \int_R K_{\alpha_i}(x, y) J_j(y) \, \mathrm{d}y$, *where* $K_\alpha(x, .)$ *is a weight function. It has weight concentrated near* x, *and therefore defines an effective neighborhood around* x *in which to aggregate data. An advantage*

of solving the PDE (2.1) *is that* K_{α_i}*, i.e., the neighborhood, does not need to be computed explicitly, and the PDE can be solved in faster computational time than integrating the kernel (Green's function) directly.*

Remark 2.1.3. *The key property in defining Shape-Tailored Local Descriptors is the scale space defined within a region of arbitrary shape. Any other PDE besides the Poisson-like PDE* (2.1) *could also be a valid choice.*

Remark 2.1.4. *The descriptor* $\mathbf{u}$ *is motivated by its covariance / robustness properties. Indeed, the descriptor is covariant to planar rotations and translations. This follows from the covariance of the Laplacian. Further, the descriptor is robust to small deformations of the set* R*. This can be seen since locally any deformation is a translation, and the solution of the PDE can be approximated by taking local averages, which is robust to small translations. This robustness is useful for textures since textons (especially in textures in nature) within regions vary by small deformations.*

Figure 2.1 visualizes Shape-Tailored Descriptors for a region R and the region $R^c = \Omega \backslash R$ outside R. Notice that Shape-Tailored Descriptors aggregate data only within regions and thus do not cause ambiguity near the boundary, whereas the non shape-tailored descriptors aggregate the data on the entire domain, thus across the boundary ∂R, resulting in ambiguity as seen by the blur near the boundary.

2.2 Shape-Tailored Descriptor Gradient

We now compute the variation of the descriptor $\mathbf{u}_R$ as the boundary ∂R is perturbed. The gradient with respect to the boundary can then be computed. Since the computations (proofs of Lemmas and Propositions) are involved, they are left to **Appendix B**.

Since $\mathbf{u}$ has components u_{ij}, we compute the variation of u_{ij}. For simplicity of notation, we suppress ij and write u. We denote by h, a vector field defined on ∂R.

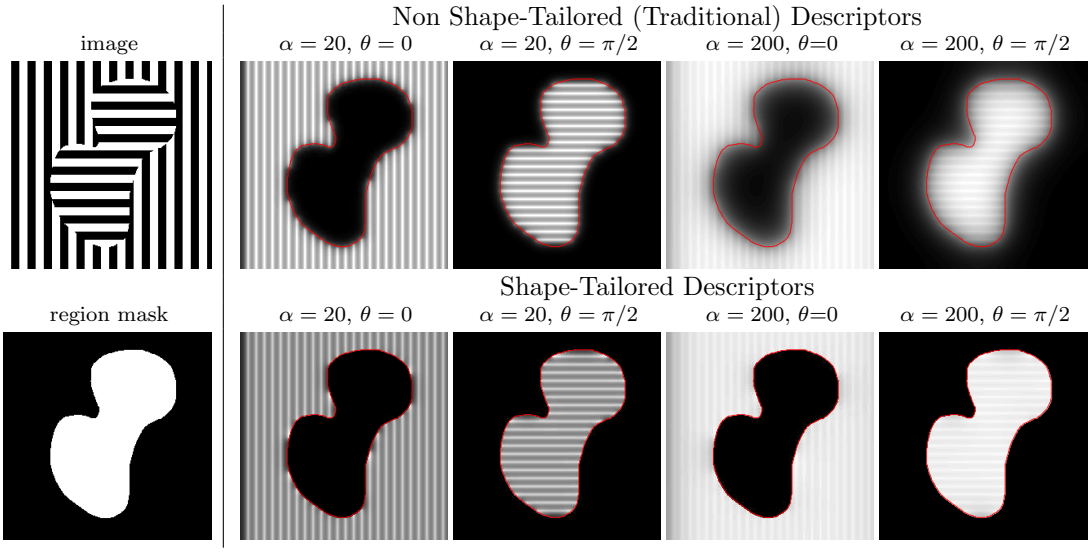

Figure 2.1: **Shape-Tailored versus non Shape-Tailored Descriptors**. [Top Left]: Image, [Bottom Left]: region R (white), and outside R^c (black). [Top Right Block]: non Shape-Tailored Descriptors have no knowledge of the shape, R, and thus aggregate data across the boundary ∂R (red), causing ambiguity that is more pronounced as the scale increases. [Bottom Right Block]: Shape-Tailored Descriptors (of R and R^c) aggregate data only within their respective regions, eliminating ambiguity near ∂R. The Shape-Tailored Feature of R, $u_{\alpha,\theta} : R \to \mathbb{R}$, and of R^c, $v_{\alpha,\theta} : R^c \to \mathbb{R}$, are displayed in a single image.

This is a perturbation of ∂R. Thus, $h : \mathbb{S}^1 \to \mathbb{R}^2$ where $\mathbb{S}^1$ is the unit interval. We denote by $u_h(x) := \mathrm{d}u(x) \cdot h$ the variation of u at x with respect to perturbation of the boundary by h.

We first show that u_h satisfies a PDE that is the same as the descriptor PDE (2.1) but with a different boundary condition and forcing term:

Lemma 2.2.1 (PDE for Descriptor Variation)**.** *Let u satisfy the PDE* (2.1)*, h be a perturbation of ∂R, and u_h denote the variation of u with respect to the perturbation h. Then*

$$\begin{cases} u_h(x) - \alpha_i \Delta u_h(x) = 0 & x \in R \\ \nabla u_h(x) \cdot N = u_s(x)(h_s \cdot N) - N^T H u(x) \cdot h & x \in \partial R \end{cases} \tag{2.3}$$

where s is the arc-length parameter of ∂R, h_s denotes the derivative with respect to arc-length, and $Hu(x)$ denotes the Hessian matrix.

One can now use the previous result to compute the gradient of u, $\nabla_c u$, with respect to $c = \partial R$. To do this, we express the solution of (2.3) using the Green's function [102], i.e., the fundamental solution, defined on R. The Green's function for (2.3) depends only on the structure of the PDE, i.e., left hand sides of (2.3), and not the particular forcing function or the right hand side of the boundary condition. Hence the Green's function for (2.3) is the same as the Green's function for (2.1). The Green's function is defined as follows:

Definition 2 (Green's Function for (2.3))**.** *The Green's function, $K_{\alpha_i} : R \times R \to \mathbb{R}$, for the problem* (2.3) *(and* (2.1)*) satisfies*

$$\begin{cases} K_{\alpha_i}(x,y) - \alpha_i \Delta_x K_{\alpha_i}(x,y) = \delta(x-y) \quad x, y \in R \\ \nabla_x K_{\alpha_i}(x,y) \cdot N = 0 \qquad\qquad\qquad\qquad x \in \partial R, y \in R \end{cases} \tag{2.4}$$

where Δ_x (∇_x) is the Laplacian (gradient) with respect to x, and δ is the Delta function.

The gradient $\nabla_c \mathbf{u}(x)$ can now be computed:

Proposition 2.2.2 (Descriptor Gradient)**.** *The gradient with respect to $c = \partial R$ of $u_{ij}(x)$ (one component of $\mathbf{u}(x)$), which satisfies the PDE* (2.1)*, is $\nabla_c u_{ij}(x) =$*

$$\left[\nabla u_{ij} \cdot \nabla_y K_{\alpha_i}(x, \cdot) + \frac{1}{\alpha_i} K_{\alpha_i}(x, \cdot)(u_{ij} - J_j) \right] N \tag{2.5}$$

where N is the outward normal, ∇_y denotes the gradient wrt the second argument of K_{α_i}, and Du indicates the spatial derivative of u. We define $\nabla_c \mathbf{u}(x)$ to be the $2 \times M$ matrix with columns as the components $\nabla_c u_{ij}(x)$.

Remark 2.2.3. *Note that $\nabla_c \mathbf{u}(x)$ is defined at each point of c for each x, and all the terms in expression* (2.5) *are evaluated at a point of the curve $c(s)$, which is suppressed for simplicity of notation.*

The Green's function is not expressible in analytic form for arbitrary shapes R. We will see that we will need to only compute region integrals of the gradient multiplied by a function. This, fortunately, may be expressed as a solution to a PDE, and thus does not require the Green's function. The integrals of descriptor gradients can be computed as:

Proposition 2.2.4 (Integrals of Descriptor Gradient)**.** *Let* $\mathbf{f}, \mathbf{g} : R \to \mathbb{R}^M$ *and* $\mathbf{u}$ *be the Shape-Tailored Descriptor in* R *(as in* (2.2)*). Define* $\mathbb{I}_d[R, \mathbf{u}, \mathbf{f}, \mathbf{g}]$ *as the quantity*

$$-\int_{\partial R} \nabla_c \mathbf{u}(x)\mathbf{g}(x)\,\mathrm{d}s(x) + \int_R \nabla_c \mathbf{u}(x)\mathbf{f}(x)\,\mathrm{d}x.$$

where $\mathrm{d}x$ *and* $\mathrm{d}s$ *are the area and arclength measure. Then*

$$\mathbb{I}_d[R, \mathbf{u}, \mathbf{f}, \mathbf{g}] = \left(tr[(D\mathbf{u})^T D\hat{\mathbf{u}}] + (\mathbf{u} - \mathbf{J})^T A^{-1}\hat{\mathbf{u}}\right) N \tag{2.6}$$

where N *is the outward normal to the boundary of* R*, tr denotes matrix trace, and*

$$\begin{cases} \hat{\mathbf{u}}(x) - A\Delta\hat{\mathbf{u}}(x) = \mathbf{f}(x) & x \in R \\ D\hat{\mathbf{u}}(x)N = \mathbf{g}(x) & x \in \partial R \end{cases}. \tag{2.7}$$

We now compute the gradient of a weighted area functional involving Shape-Tailored Descriptors. This result will be useful for computing gradients of energies designed for segmentation in Section 3.

Proposition 2.2.5 (Weighted Area Gradient)**.** *Let* $F : \mathbb{R}^M \to \mathbb{R}$ *and* $\mathbf{u} : R \to \mathbb{R}^M$ *be the Shape-Tailored Descriptor on* R*. Define the weighted area functionals as* $A_F = \int_R F(\mathbf{u}(x))\,\mathrm{d}x$*. Then*

$$\nabla_c A_F = (F \circ \mathbf{u})N + \mathbb{I}_d[R, \mathbf{u}, (\nabla F) \circ \mathbf{u}, \mathbf{0}] \tag{2.8}$$

where $\mathbb{I}_d$ is defined as in Proposition 2.2.4.

The dependence of the descriptor on the region induces the terms involving $\mathbb{I}_d$ in the above gradient. Those terms depend on $\hat{\mathbf{u}}$ defined in (2.7), which is the solution to another PDE defined on R. Thus, when performing a gradient descent of A_F, $\mathbf{u}$ and $\hat{\mathbf{u}}$ must be updated as the region evolves.

Chapter 3

Segmentation of Shape-Tailored Descriptors

To illustrate the use of Shape-Tailored Descriptors in segmentation, we incorporate the descriptors into the Mumford-Shah energy [51], and then use the results of the previous section to compute its gradient.

Let $I : \Omega \to \mathbb{R}^k$ be the image, and $\mathbf{J} : \Omega \to \mathbb{R}^M$ be the vector of channels computed from I. We assume that the region R that we wish to segment and the background $R^c = \Omega \backslash R$ each consist of Shape-Tailored Descriptors that are mostly constant within neighborhoods of R and R^c following the Mumford-Shah model. We denote by $\mathbf{u} : R \to \mathbb{R}^M$ (resp., $\mathbf{v} : R^c \to \mathbb{R}^M$) the Shape-Tailored Descriptor in region R (resp., R^c) computed from $\mathbf{J}$. Note that $\mathbf{u}$ and $\mathbf{v}$ are both computed from $\mathbf{J}$ at the same scales α_i. The piecewise smooth Mumford-Shah [51, 103, 104] applied to $\mathbf{u}$ and $\mathbf{v}$ is

$$E(\mathbf{a}_i, \mathbf{a}_o, R) = \int_R (|\mathbf{u}(x) - \mathbf{a}_i(x)|^2 + \beta |D\mathbf{a}_i(x)|^2)\, \mathrm{d}x + \int_{R^c} (|\mathbf{v}(x) - \mathbf{a}_o(x)|^2 + \beta |D\mathbf{a}_o(x)|^2)\, \mathrm{d}x + \gamma L, \quad (3.1)$$

where $\mathbf{a}_i : R \to \mathbb{R}^M$ and $\mathbf{a}_o : R^c \to \mathbb{R}^M$ are functions that vary smoothly within their respective regions. In other words, they are roughly constant within local neighborhoods of their respective regions. Note that D indicates the Jacobian, and the two terms involving D enforce a smoothness penalty on $\mathbf{a}_i$ and $\mathbf{a}_o$. $\beta > 0$ controls the size of the neighborhoods for which the descriptors are assumed constant. $\beta \to \infty$ implies the whole region is assumed to have a constant descriptor (as in the simplified

piecewise constant Mumford-Shah or Chan-Vese model [49]). Smaller β assumes that descriptors are constant within smaller neighborhoods. The functions $\mathbf{a}_i, \mathbf{a}_o$ are also solved as part of the optimization problem. Regularity of the region boundary is induced by the penalty on the length L of ∂R, where $\gamma > 0$.

We use alternating minimization in R and $\mathbf{a}_i, \mathbf{a}_o$. One can optimize for $\mathbf{a}_i$ and $\mathbf{a}_o$ given $\mathbf{u}$, $\mathbf{v}$ and R to find

$$\begin{cases} \mathbf{a}_i(x) - \beta\Delta\mathbf{a}_i(x) = \mathbf{u}(x) & x \in R \\ \mathbf{a}_o(x) - \beta\Delta\mathbf{a}_o(x) = \mathbf{v}(x) & x \in R^c \end{cases}. \tag{3.2}$$

Optimization in the region is performed using gradient descent, and the gradient can be computed using results of the previous section:

$$\nabla E = (|\mathbf{u} - \mathbf{a}_i|^2 - |\mathbf{v} - \mathbf{a}_o|^2 + \beta|D\mathbf{a}_i|^2 - \beta|D\mathbf{a}_o|^2)N + 2(\mathbb{I}_d[R, \mathbf{u}, \mathbf{u} - \mathbf{a}_i, \mathbf{0}] + \mathbb{I}_d[R^c, \mathbf{v}, \mathbf{v} - \mathbf{a}_o, \mathbf{0}]). \tag{3.3}$$

Figure 3.2 shows the gradient descent of E to segment a sample texture for the case that $\mathbf{a}_i, \mathbf{a}_o$ are assumed constant, i.e., the Chan-Vese model. To illustrate the motivation for segmentation with Shape-Tailored Descriptors, we show comparison to non-shape tailored descriptors (choosing the full image domain Ω to compute descriptors by solving (2.1) once on Ω, and using the standard Chan-Vese algorithm to segment these descriptors).

3.1 Numerical Implementation

We use level set methods [76] to implement the gradient descent of E. Discretization follows the standard schemes of level sets. Let Ψ be the level set function, F be the normal component of the gradient of energy ∇E, $\Delta t > 0$ be the step size, and t the

iteration number. Steps 2-5 below are iterated until convergence of Ψ:

1. Initialize Ψ_0, $R_0 = \{\Psi_0 < 0\}$, $R_0^c = \Omega \backslash R_0$.

2. Solve for the Shape-Tailored Descriptors $\mathbf{u}_t : R_t \to \mathbb{R}^M$, $\mathbf{v}_t : R_t^c \to \mathbb{R}^M$ by solving (2.2) using an iterative scheme initialized with the Shape-Tailored Descriptors from the previous iteration $(\mathbf{u}_{t-1}, \mathbf{v}_{t-1}) : \Omega \to \mathbb{R}$ (zero for $t = 0$).

3. Solve for $\mathbf{a}_{i,t} : R_t \to \mathbb{R}^M$, $\mathbf{a}_{o,t} : R_t^c \to \mathbb{R}^M$ by solving (3.2) using an iterative scheme with initialization $\mathbf{a}_{i,t-1}, \mathbf{a}_{o,t-1}$. For the piecewise constant model, $\mathbf{a}_{i,t}$ and $\mathbf{a}_{o,t}$ are the averages of $\mathbf{u}_t$ and $\mathbf{v}_t$, respectively.

4. Solve for the "hat" descriptors $\hat{\mathbf{u}}_t : R_t \to \mathbb{R}^M$, $\hat{\mathbf{v}}_t : R_t^c \to \mathbb{R}^M$ by solving (2.7) (with the corresponding forcing and boundary functions determined by the arguments of $\mathbb{I}_d$ in (3.3)) using an iterative scheme with initialization $(\hat{\mathbf{u}}_{t-1}, \hat{\mathbf{v}}_{t-1})$.

5. Solve for F using (3.3). Then $\Psi_t = \Psi_{t-1} - \Delta t F |\nabla \Psi_{t-1}|$, and $R_t = \{\Psi_t < 0\}$, $R_t^c = \Omega \backslash R_t$.

The multigrid algorithm is used to solve for $\mathbf{u}_t$, $\mathbf{v}_t$, $\mathbf{a}_{i,t}$, $\mathbf{a}_{o,t}$, $\hat{\mathbf{u}}_t$, and $\hat{\mathbf{v}}_t$. After the first iteration, the update of these descriptors is fast since the solution changes only slightly between $t-1$ and t. Details of the numerical scheme is left to **Appendix B**.

Updates for each of the components of $\mathbf{u}_t, \mathbf{v}_t$ can be done in parallel as the components are independent. Similarly for $\mathbf{a}_{i,t}, \mathbf{a}_{o,t}$ and $\hat{\mathbf{u}}_t, \hat{\mathbf{v}}_t$. Using a 12 core processor, our implementation to minimize E on a 1024×1024 image roughly takes 18 seconds for the piecewise constant model. This is with a box tessellation initialization, and the number of descriptor components is $M = 55$.

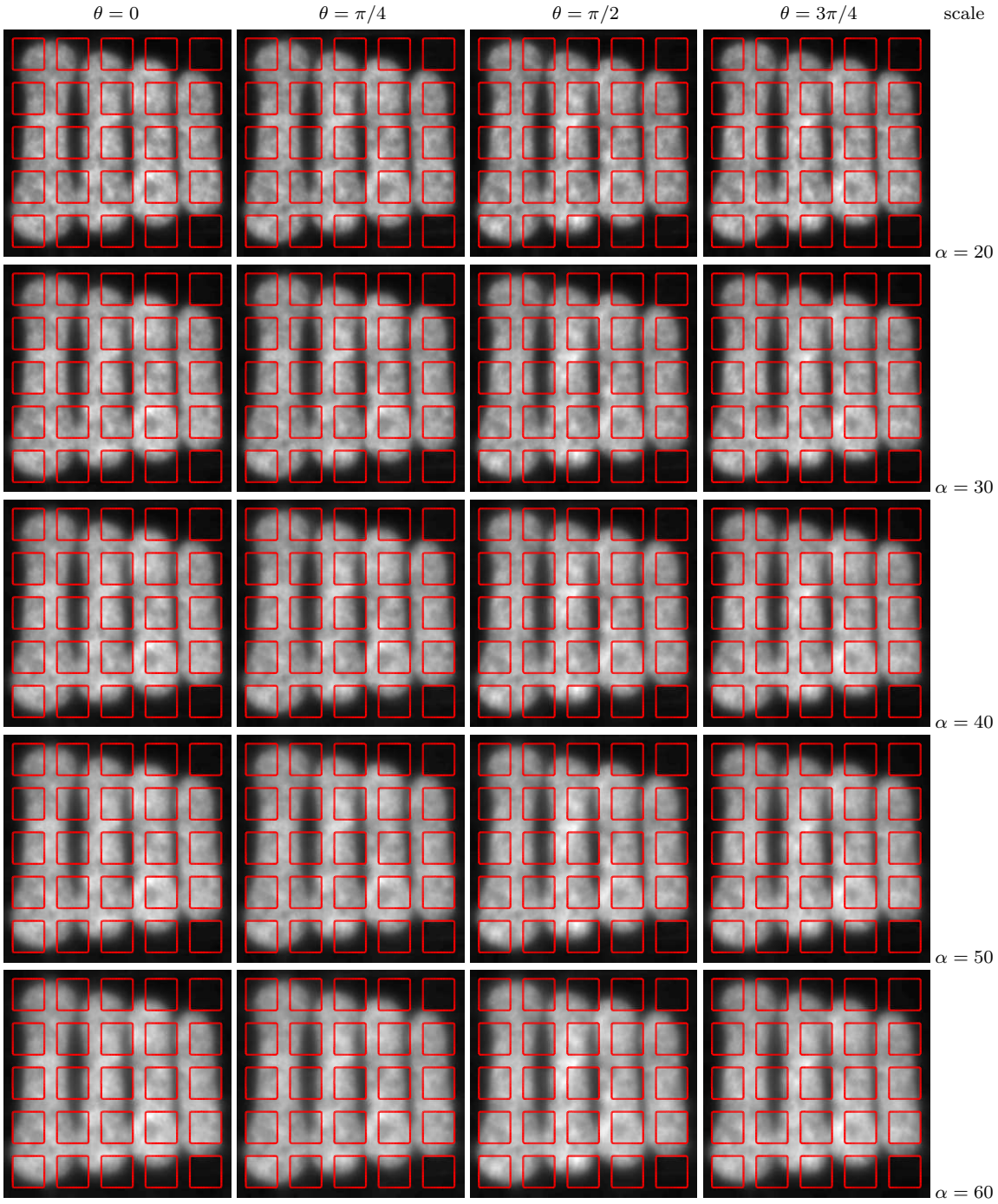

Figure 3.1: **Shape-Tailored Descriptors**. Shape-Tailored Descriptors for R and R^c are shown for different scales and orientations, the boundary between two regions is shown in red.

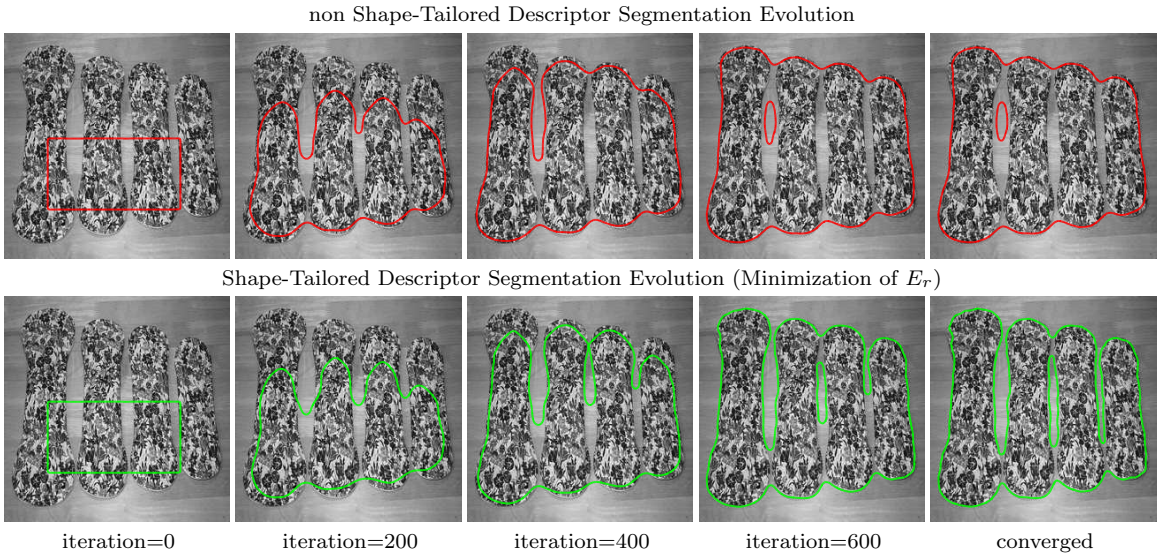

Figure 3.2: **Shape-Tailored vs. Non Shape-Tailored Descriptor Segmentation Evolution.** [Top]: Non Shape-Tailored (traditional) local Descriptors segmented with Chan-Vese. Steps in the evolution are shown. Since traditional Descriptors aggregate image data across textures, Descriptors near texture boundaries are ambiguous, causing errors. [Bottom]: Shape-Tailored Descriptors (same α, θ) are computed jointly with the segmentation, and since data is aggregated within regions, accurate texture boundaries are captured.

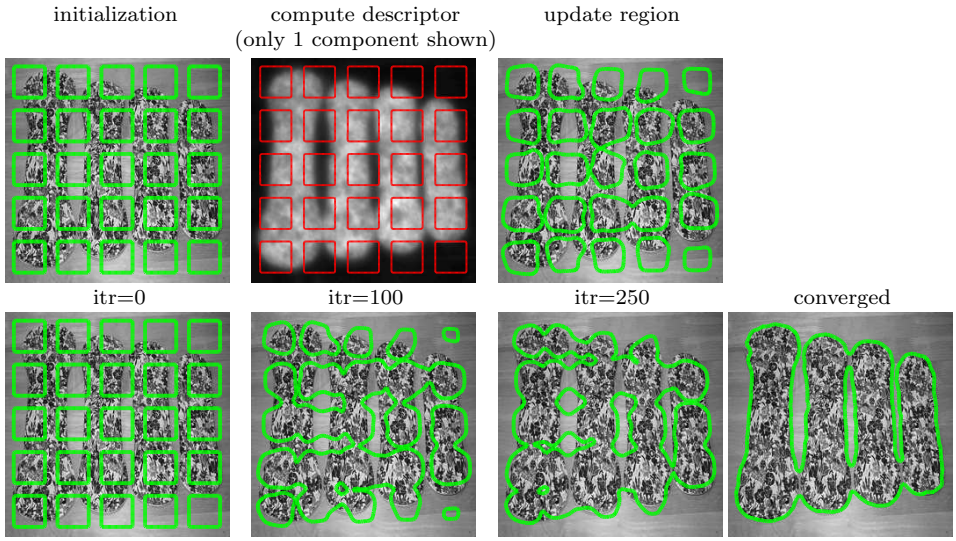

Figure 3.3: **Numerical Implementation**. Row1: Steps of numerical implementation, step 1: initialization of region with box tessellation, step 2: computation of descriptor on R and R^c both descriptor are shown with boundary in red, step 3: region update, 2 and 3 are repeated until convergence. Row 2: Segmentation evolution

Chapter 4

Continuum Scale Space and Coarse-to-Fine Segmentation

In this section, we construct a coarse-scale preferential energy without blurring across segments. To achieve this, we introduce a Shape-Tailored *Continuum* Scale Space[1]. A Shape-Tailored Scale Space avoids blurring across regions, and a continuum of scales obtains a coarse-to-fine property.

4.0.1 Shape-Tailored Heat Scale Space

The Gaussian Scale Space, constructed by smoothing the image with a Gaussian at a continuum of scales (variances), can be generalized to be defined within regions (subsets of the image) of arbitrary shape by using the heat equation (see Figure 4.1). The solution to the heat equation defaults to Gaussian smoothing when the domain is $\mathbb{R}^2$. The heat equation, defined in a region R, is:

$$\begin{cases} \partial_t u(t,x) = \Delta u(t,x) & x \in R,\, t > 0 \\ \nabla u(t,x) \cdot N = 0 & x \in \partial R,\, t > 0 \\ u(0,x) = I(x) & x \in R \end{cases} \tag{4.1}$$

where $u : [0, +\infty) \times R \to \mathbb{R}^k$ denotes the scale space, $R \subset \Omega \subset \mathbb{R}^2$ is the domain (or subset) of the image Ω, $I : \Omega \to \mathbb{R}^k$ ($k \geq 1$ is the number of channels) is the image, ∂R denotes the boundary of R, N is the unit outward normal vector to R, ∇ denotes the vector of partials, Δ denotes the Laplacian, ∂_t denotes the partial derivative with

[1]Work presented in this chapter is based on [105]

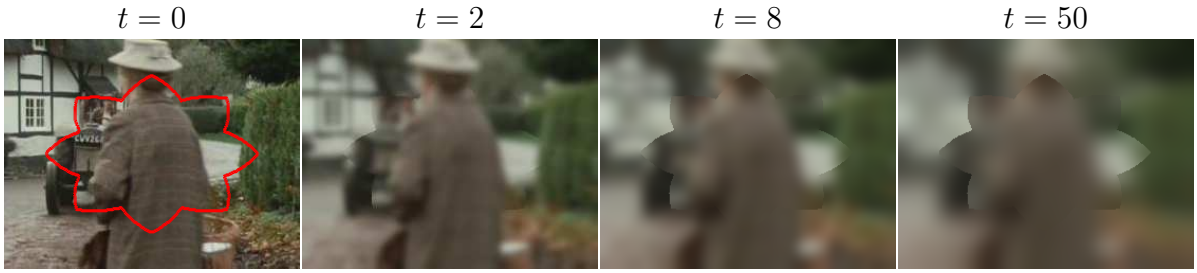

Figure 4.1: Shape-tailored scale space (solution of heat equation within regions with boundary in red) for various times (scales). Notice the quick diffusion of fine scale structures, and the persistence of coarse structure. The persistence of coarse structure is important to our coarse-to-fine segmentation scheme.

respect to t, and t is the scale parameter parameterizing the scale space. Increasing t indicates increasing amount of smoothing.

The construction of scale space using the heat equation is useful for segmentation as it allows us to conveniently compute coarse scales of the data *within* regions of a segmentation. If the regions are chosen to be the correct segmentation, this avoids blurring data across segmentation boundaries. However, one does not know the segmentation a-priori, and thus the regions are simultaneously optimized with the scale spaces in the optimization problem defined next.

4.0.2 Coarse-Scale Preferential Energy

The Gaussian scale space is relevant in defining our coarse-scale preferential energy as the heat equation removes the fine structure of the image in short time, and spends more time removing coarse structure (see Figure 4.1) [106]. Therefore, a data term integrating the scale space over the scale parameter of the heat equation gives preference to segmentations separating the coarse over the fine structure. We thus propose the following energy for segmentation integrating over a continuum of scales:

$$E = \sum_{i=1}^{N} \int_{R_i} \int_0^T |u_i(t,x) - a_i|^2 w(t)\,\mathrm{d}t\,\mathrm{d}x + \mathrm{Reg}(\partial R_i), \tag{4.2}$$

where $T > 0$ is the final time, $\{R_i\}_{i=1}^N$ are a collection of regions forming the segmentation, $a_i \in \mathbb{R}^k$ is the average of $u_i(t, \cdot)$, and $w : \mathbb{R}^+ \to \mathbb{R}$ is a function that weights each scale. It can be shown that a_i is independent of t. This energy is the mean-squared error of the image within the region across all scales. It generalizes common single scale segmentation models, including piecewise constant Mumford-Shah (Chan-Vese [103, 51]). Reg denotes usual curve regularization that will be discussed in the implementation section, Section 4.1.3.

To further demonstrate the coarse preference of our energy, we write the data term of the energy in Fourier domain. For simplicity, we choose $w(t) = 1$; other weights lead to a similar conclusion. Choosing the whole domain as a region, the data term can be written in Fourier domain as:

Lemma 4.0.1. *Suppose $I : \mathbb{R}^2 \to \mathbb{R}$ and $a = \int_{\mathbb{R}^2} I(x)\,\mathrm{d}x = \int_{\mathbb{R}^2} u(t,x)\,\mathrm{d}x$. Then*

$$\int_0^\infty \int_{\mathbb{R}^2} |u(t,x) - a|^2\,\mathrm{d}x\,\mathrm{d}t = \int_{\mathbb{R}^2} |H(\omega)\hat{I}(\omega)|^2\,\mathrm{d}\omega, \tag{4.3}$$

where $H(\omega) = \frac{1}{\sqrt{2}|\omega|}$, $\hat{I}$ denotes the Fourier transform, and ω denotes frequency.

The proof can be found in **Appendix C**. The function H decays the high frequency components of I at a linear rate, thus the energy gives preference to the coarse image structure. Without integrating over the scale space, the energy in Fourier domain would result in $H = 1$, which has equal preference to coarse and fine structure.

4.1 Optimization and Scale Weighting

We now derive the optimization scheme for the energy (4.2), and propose and analyze weight choices.

4.1.1 Constrained Optimization Problem

The energy (4.2) is optimized with respect to the regions. Since the integrand of the energy depends on the regions nonlinearly, as the heat equation has a non-linear dependence on the region, the energy is not convex, and thus we apply gradient descent. In order to compute the gradient, we formulate the energy minimization as a constrained optimization problem. That is, we treat the minimization of the energy (4.2) as defined on both the regions R_i *and* u_i with the constraint that u_i satisfies the heat equation (4.1). This formulation allows us to apply the technique of Lagrange multipliers, which makes computations simpler since the nonlinear dependence of u_i on R_i is decoupled.

Since all data terms of the energy in (4.2) have the same form, we focus on computing the gradient for any one term. For convenience in notation, we avoid the subscript i denoting the index of the region. Using Lagrange multipliers, we formulate the energy as a function of region R, u, and the Lagrange multiplier $\lambda : [0,T] \times R \to \mathbb{R}^k$ with the constraint that u satisfies the heat equation:

$$E(R,u,\lambda) = \int_0^T \int_R f(u)\,\mathrm{d}x\,\mathrm{d}t + \int_0^T \int_R (\nabla\lambda \cdot \nabla u + \lambda \partial_t u)\,\mathrm{d}x\,\mathrm{d}t, \quad (4.4)$$

where $f(t,u) = (u-a)^2 w(t)$. We have excluded the dependencies on x, t for convenience of notation. We have also provided a more general form of the squared error with a general function f of u. The second term comes from the weak form of the heat equation. Integrating by parts to move the gradient from λ to ∇u gives the classical form of the heat equation in (4.1). Therefore, the second term in (4.4) is indeed obtained by Lagrange multipliers.

We may now compute the gradient for E (4.4) by deriving the optimizing conditions in u and λ. Details are found in **Appendix C**. Optimizing in λ simply results

in the original heat equation constraint, so we compute the optimizing condition for u by computing the derivative (variation) of E with respect to u. This results in a solution for λ as given below:

Lemma 4.1.1 (PDE for Lagrange Multiplier λ)**.** *The Lagrange multiplier λ satisfies the following heat equation with forcing term, evolving backwards in time:*

$$\begin{cases} \partial_t \lambda(t,x) + \Delta\lambda(t,x) = f_u(t,u(t,x)) & x \in R \times [0,T] \\ \nabla\lambda(t,x) \cdot N = 0 & x \in \partial R \times [0,T] \\ \lambda(T,x) = 0 & x \in R \end{cases}, \tag{4.5}$$

where f_u denotes the partial with respect to the second argument.

Duhamel's Principle [107] leads to the following solution:

Lemma 4.1.2 (Lagrange Multiplier λ)**.** *The solution of* (4.5) *can be written as*

$$\lambda(t,x) = -\int_t^T F(s-t,x;s)\,\mathrm{d}s. \tag{4.6}$$

where $F(\cdot,\cdot;s) : [0,T] \times R \to \mathbb{R}$ is the solution of the forward heat equation (4.1) *with zero forcing and initial condition $f_u(u)$ evaluated at time s, i.e.,*

$$\begin{cases} \partial_t F(t,x;s) - \Delta F(t,x;s) = 0 & x \in R \times [0,T] \\ \nabla F(t,x;s) \cdot N = 0 & x \in \partial R \times [0,T] \cdot \\ F(0,x;s) = f_u(s,u(s,x)) & x \in R \end{cases} \tag{4.7}$$

In the case that $f(t,u) = (u-a)^2 w(t)$, λ can be expressed as

$$\lambda(t,x) = -2\int_t^T (u(2s-t,x) - a)w(s)\,\mathrm{d}s. \tag{4.8}$$

The formula for λ in (4.8) is convenient for particular choices of the weight w

as taking the limit as T gets large leads to the energy gradient being computable without explicitly computing the scale space u, as shown in the next section.

With the optimizing conditions for u and λ of E, we can now compute the gradient of the energy E with respect to R in terms of λ and u:

Proposition 4.1.3. *The gradient of E with respect to the boundary ∂R can be expressed as*

$$\nabla_{\partial R} E = \int_0^T \left[f(u) + \nabla\lambda \cdot \nabla u + \lambda \partial_t u \right] \, \mathrm{d}t \cdot N, \tag{4.9}$$

where N is the normal vector to ∂R.

4.1.2 Weighting Functions

We now explore possible choices of weights, w. Some choices of weights may have convenient solutions for the gradient that does not require computation of the scale-space u, which makes the computational cost much less expensive than the generic formula (4.9). As observed in the experiments, all have a coarse-to-fine behavior, but each differs in the extent of this property. Calculations are provided in **Appendix C**.

Exponential With Positive Exponent (ExpPos): We consider the weight $w(t) = e^{1/\alpha[(t/T)^2 - 1]} \mathbf{1}_{[0,T]}(t)$, where $\alpha > 0$ and $\mathbf{1}$ denotes the indicator function. Here, the weight increases with scale so that the largest scales between 0 and T are weighted the most. We truncate at a finite T. This is because for large scales, the image is blurred too much to be used in segmentation, and very large scales should have either low or zero weight. This weighting exhibits the most coarse-to-fine behavior of any weightings we consider. Although this is the ideal weighting, to the best of our knowledge, the gradient (4.9) cannot be written in a form that does not require computation of the scale space. Thus, it is computationally more costly than other weightings we consider. However, typically T is chosen small (e.g., $T = 10$ for a

256×256 image) in comparison to other weightings, which offers cost savings.

Truncated Uniform Weight (Uniform): We consider the weight function $w(t) = \mathbf{1}_{[0,T]}(t)$. This uses a uniform weight on all scales between 0 and T. Since we want to avoid very large scales ($T \to \infty$), we choose a finite T. The gradient when T is large (but still finite) is approximated as

$$\nabla_{\partial R} E \cdot N \approx (u_0 + a)(aT - U_T) + \frac{1}{2}|\nabla U_T|^2, \tag{4.10}$$

where u_0 is initial condition to the heat equation (original data), and

$$\begin{cases} U_T(x) - T\Delta U_T(x) = Tu_0(x) & x \in R \\ \nabla U_T \cdot N = 0 & x \in \partial R \end{cases}. \tag{4.11}$$

U_T is the integral of the scale space from 0 to T and this can be approximated as the solution of (4.11) (**Appendix C**). The advantage of (4.10) is that it does not require explicit computation of the scale space, and (4.11) can be solved efficiently iteratively. Indeed, in gradient descent of R, the solution for the previous iteration can be used as a warm start for the next iteration. Analysis of the approximation is in **Appendix C**.

Exponential With Negative Exponent (ExpNeg): We consider the weight $w(t) = e^{-(1/\alpha)t}$ for all $t \in [0, \infty)$, where $\alpha > 0$. A small value of α implies that only the small scales are relevant. A large value of α includes larger scales, which is desired. The intuition for using this weighting is that it includes moderately large scales with non-negligible weight as desired, it disregards very large scales as desired by having exponentially decaying weight, and it has an exact solution for the gradient that does not require the computation of scale space. One can show that the gradient is

$$\nabla_{\partial R} E \cdot N = a\alpha(a + 2u_0) - u_0 U_{2\alpha} + \frac{1}{4\alpha} U_{2\alpha}^2 - \frac{1}{2}|\nabla U_{2\alpha}|^2, \tag{4.12}$$

where $U_{2\alpha}$ solves (4.11) with T replaced by 2α. Like the uniform weighting, the gradient yields a form that does not require the computation of the scale space. An advantage over the uniform case is that the solution is exact.

4.1.3 Multi-Region Segmentation

We now present the numerical implementation of the gradient descent for energy (4.2), when there are multiple regions. The term involving regularization is discussed later. Let $G_i N_i$ be the gradient of the i^{th} summand of E in (4.2), where N_i is the outward normal to R_i. For instance, $G_i N_i$ can be any one of the expressions (4.9), (4.10), (4.12). As shown in [28], the gradient of the full energy evaluated at a point x is just the sum of $G_i N_i$ for all i such that $x \in \partial R_i$. For a point $x \in \partial R_i \cap \partial R_j$, this yields that the gradient is $(G_i - G_j)N_i$.

To achieve sub-pixel accuracy, we use relaxed indicator functions $\phi_i : \Omega \to [0, 1]$ for $i = 1, \ldots, N$ to represent the regions, similar to level set methods [76]. By a slight abuse of notation, denote by G_i the quantity multiplying the normal vector for region R_i in either of (4.9), (4.10), (4.12), which is defined in the entire region R_i. We extend it from R_i to $D(R_i)$, a small dilation of R_i, by solving for G_i in $D(R_i)$. The extension beyond the region is done so that the evolution of ϕ_i can be defined around the curve, as in level set methods. Following [76] to convert a curve to a level set evolution, the update scheme for ϕ_i inducing the gradient descent of the regions is Algorithm 1.

The update of the ϕ_i in Line 7 of Algorithm 1 involves the term $\Delta\phi_i^\tau$, which provides smoothness of the curve. More sophisticated regularizers (such as length regularization) may be used, but we have found this simple regularization sufficient. We choose $\varepsilon = 0.005$ in experiments, and this does not need to be tuned, as it is mainly for inducing regularity for computation of derivatives of ϕ. Further, considering the scale space naturally induces regularity.

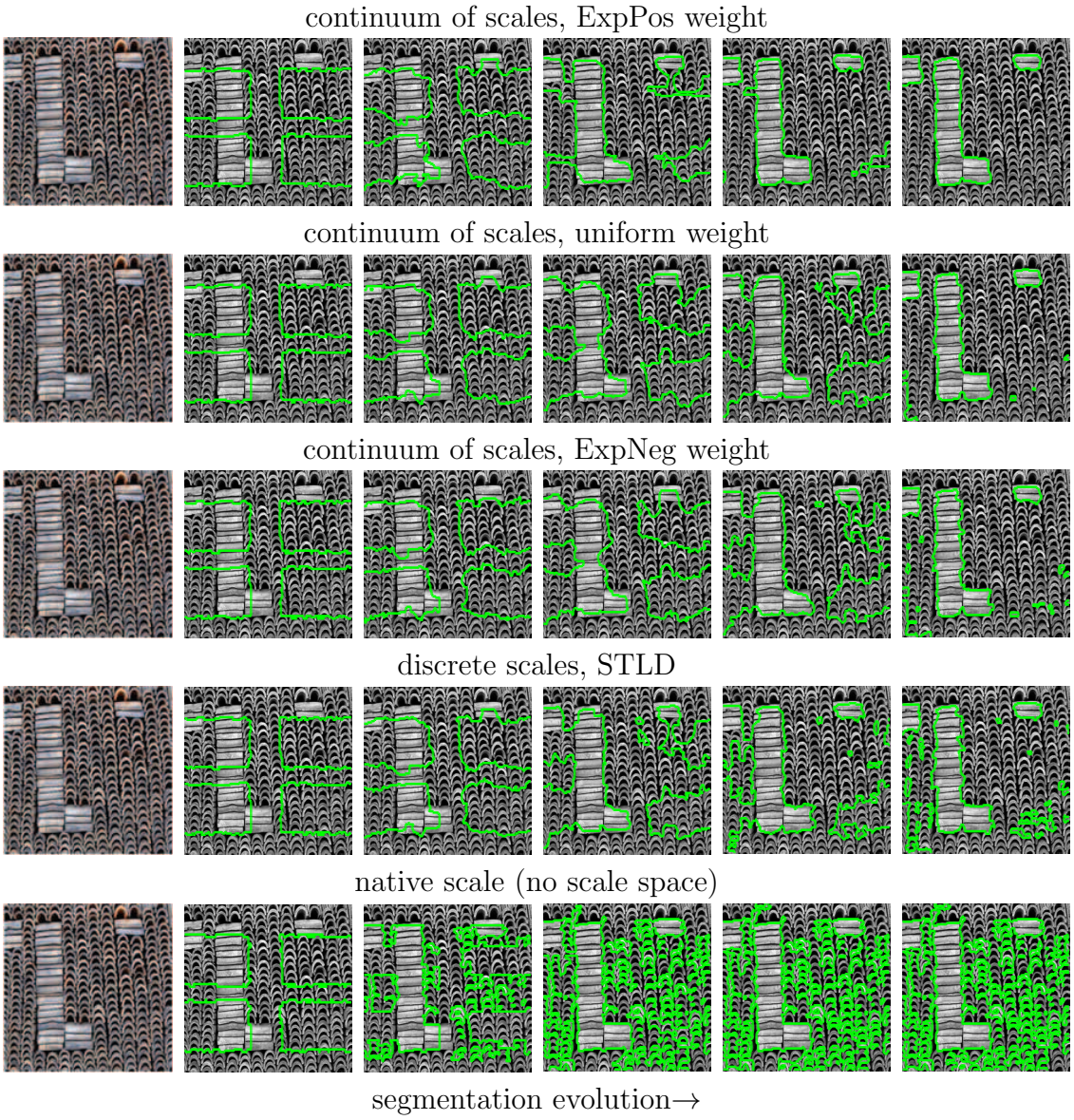

Figure 4.2: We compare usual segmentation of the native image scale, a discrete shape-tailored scale space (STLD), ExpPos, Uniform, and ExpNeg weightings for the continuum scale space. No coarse-to-fine behavior is exhibited for the native image scale and STLD. The continuum scale spaces give coarse-to-fine behavior, with ExpPos more so than other weightings.

Algorithm 1 Multi-Region Gradient Descent

1: Input: An initialization of ϕ_i
2: **repeat**
3: Set regions: $R_i = \{x \in \Omega \, : \, i = \text{argmax}_j \phi_j(x)\}$
4: Compute dilations, $D(R_i)$, of R_i
5: Compute band pixels $B_i = D(R_i) \cap D(\Omega \backslash R_i)$
6: Compute G_i for pixels in B_i
7: Update pixels $x \in D(R_i) \cap D(R_j)$ as follows:

$$\begin{aligned}\phi_i^{\tau+\Delta\tau}(x) = \phi_i^{\tau}(x) - \Delta\tau(G_i(x) - G_j(x))|\nabla\phi_i^{\tau}(x)| \\ + \Delta\tau \cdot \varepsilon\Delta\phi_i^{\tau}(x).\end{aligned}$$

8: Update all other pixels as

$$\phi_i^{\tau+\Delta\tau}(x) = \phi_i^{\tau}(x) + \Delta\tau \cdot \varepsilon\Delta\phi_i^{\tau}(x).$$

9: Clip between 0 and 1: $\phi_i = \max\{0, \min\{1, \phi_i\}\}$.
10: **until** regions have converged

4.2 Application to Motion Segmentation

In this section, we show how the results of the previous section can be applied to motion segmentation. Motion segmentation is the problem of segmenting objects and/or regions with similar motions computed using multiple images of the object(s). One of the challenges of motion segmentation is that motion is inferred through a sparse set of measurements (e.g., along image edges or corners), and thus the motion signal is typically only reliable for segmentation in sparse locations. By using a scale space formulation of an energy for motion segmentation, coarse representations of the motion signal are integrated and more significantly impact the segmentation. This property increases the reliability of motion segmentation (Figure 4.3), and the coarse-to-fine approach captures the coarse-structure without being impacted by fine-scale distractions at the outset.

With this motivation, we reformulate the motion segmentation problem with scale space. Let $I_0, I_1 : \Omega \to \mathbb{R}^k$ be two images of a sequence where Ω is the domain of the image. For a given region R_i, we define a mapping $w_i : R_i \to \Omega \subset \mathbb{R}^2$, which we

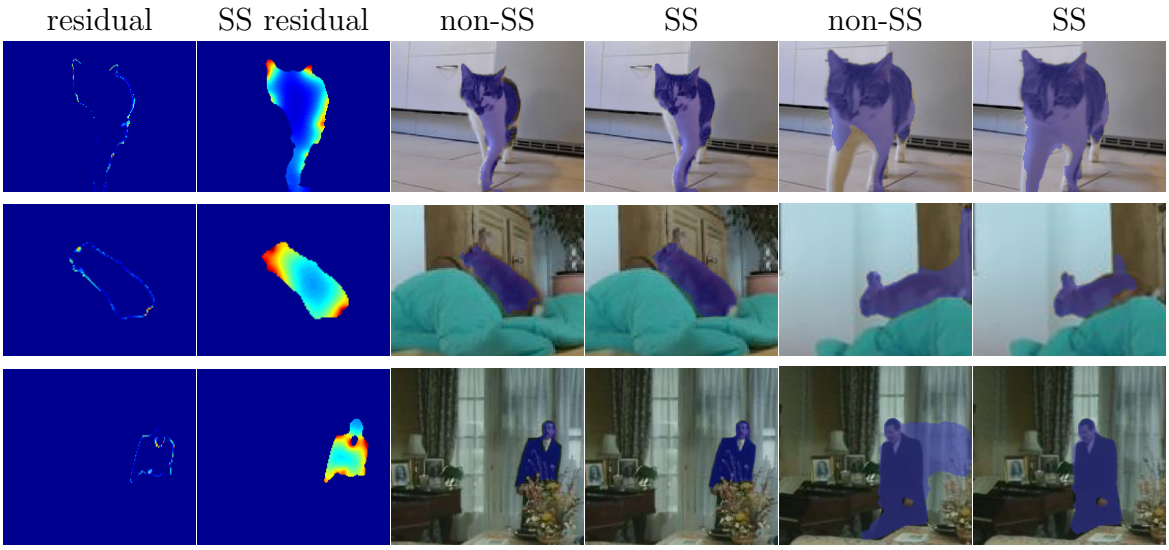

Figure 4.3: Motion residuals at a single scale are sparse (left column), leading to difficulties in using these cues in segmentation (non-SS). Motion cues at a continuum of scales (SS) provide a richer signal (2nd column), which improves segmentation. Segmentations (in purple) are shown for a frame (middle two) and a few frames ahead (right two). Although errors in the non-SS approach are subtle between frames, they quickly propagate across frames, compared to our approach.

call a warp or deformation that back warps I_1 to I_0. We assume that I_0 and I_1 are related through w_i by the Brightness Constancy Assumption, except for occlusions, as in typical works in the optical flow [79]. Define the energy

$$E_{mseg} = \sum_{i=1}^{N} \int_{R_i} \int_0^T [1 - m(x)]|u_i(t,x)|^2 w(t)\,\mathrm{d}t\,\mathrm{d}x - \int_{R_i} m(x) \log p_{R_i}(I_0(x))\,\mathrm{d}x + \mathrm{Reg}(\partial R_i), \quad (4.13)$$

where u_i is the scale space of the difference of I_0 and the back-warping of I_1 in the un-occluded region $R_i \backslash O_i$:

$$u_{0,i} = \begin{cases} I_1(w_i(x)) - I_0(x) & x \in R_i \backslash O_i \\ 0 & x \in O_i \end{cases}, \quad (4.14)$$

and $m : \Omega \to [0,1]$ is the motion ambiguity function. Note that the energy in the case

$m = 0$ is equivalent to integrating over all scales the difference of the scale spaces of I_0 and of $\hat{I}_1$ (defined as $I_1 \circ w_i$ inside $R_i \backslash O_i$ and I_0 in O_i). Note that $\hat{I}_1$ is used rather than $I_1 \circ w_i$ as the latter does not correspond to I_0 in the occlusion. This energy requires that the regions are chosen so that *all* scales of the images between 0 and T match. The motion ambiguity function m indicates whether the motion at a pixel is reliable for segmentation (1 in a textureless or occluded region and 0 otherwise). In case the motion is ambiguous, local color histograms p_{R_i} within regions are used for grouping. As is typical in optical flow [79], we set the occlusion to be a threshold of the residual: $O_i = \{x \in R_i \,:\, |I_1(w_i(x)) - I_0(x)|^2 > \beta\}$.

The optimization involves iterative alternating updates of the warps and the regions. To update warps, we use the method of warp estimation in [114]. To update the regions, we use the results of the previous section and use the exponential weight with negative exponent, for computational efficiency. This yields the gradient of the i^{th} data terms in (4.13) approximately as

$$\left[(1-m)(\frac{\alpha}{4}U^2 - u_0 U - \frac{1}{2}|\nabla U|^2) - m \log p_{R_i}(I_0)\right] N_i, \tag{4.15}$$

where U is the solution of (4.11) using $T = 2\alpha$ and right hand side $u_{0,i}$. The gradient descent of E_{mseg} is then given by Algorithm 1, choosing G_i to be the component of (4.15) multiplying N_i. We apply our method frame-by-frame. Then we propagate the result to the next frame via the computed warp to warm-start the segmentation in the next frame.

Chapter 5

Learned Shape-Tailored Descriptors for Segmentation

In this section, we describe our approach to learning descriptors that are descriptive of neighborhoods around a pixel within a specific region of interest, while having invariance to complex photometric and geometric nuisances[1]. We first review Shape-Tailored Descriptors [26], which are descriptors invariant to minor photometric and geometric nuisances, and are computed only from image information within a region of interest. We then describe how to use these "base" descriptors to learn such shape-tailored descriptors that are invariant to more complex nuisances, such as illumination change, shading, etc.

5.0.1 Base Shape-Tailored Descriptors

Let Ω be the domain of the image, $J_j : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^+$ for $j = 1, \ldots, N_c$ (N_c is the number of channels) be channels of the image, these could be for instance, color and oriented gradients. Let $R \subset \Omega$ be a region of interest, which can have arbitrary shape. Shape-Tailored Descriptors are defined as the solution of Poisson-type partial differential equations (PDE):

$$\begin{cases} u_{ij}(x) - \alpha_i \Delta u_{ij}(x) = J_j(x) & x \in R \\ \nabla u_{ij}(x) \cdot N = 0 & x \in \partial R \end{cases}, \tag{5.1}$$

[1]Work presented in this chapter is based on [108]

where ∇ is the gradient, Δ is the Laplacian, ∂R is the boundary of R, N is the unit outward normal to ∂R, $i = 1, \ldots, N_s$ and N_s is the number of scales, and $\alpha_i \in \mathbb{R}^+$ are the scales. The solution of the PDE can be shown to be the minimizer of the energy $E = \int_R (J_j(x) - u_{ij}(x))^2 \, \mathrm{d}x + \alpha_i \int_R |\nabla u_{ij}(x)|^2 \, \mathrm{d}x$. The solution is thus a balance between fidelity to the image and smoothness, with α_i larger implying more smoothness. We set $\mathbf{u} : R \to \mathbb{R}^{N_s N_c}$ as the vector of all components of scales and channels:

$$\mathbf{u}(x) = (u_{11}(x), \ldots, u_{1N_c}(x), \ldots, u_{N_s 1}(x), \ldots, u_{N_s N_c}(x))^T.$$

The u_{ij} are smoothed channels of the image and since the PDE is defined in a specific region R, no image information outside R is used to determine u_{ij}. This is important in region-based approaches to segmentation, as aggregating image information across segmentation boundaries mixes unrelated statistics and then such descriptors are difficult to group. Due to the smoothing, the descriptors exhibit invariance to small geometric transformations. However, they are not in general invariant to more complex geometric transformations or complex photometric transformations, such as illumination change. Therefore, in the next section, we use the descriptors above and learn more invariant descriptors. Since these learned descriptors are built from the descriptors above, they inherit the shape-tailored property.

5.0.2 Metric and Descriptor Learning

In this section, we learn a function, $f : \mathbb{R}^n \to \mathbb{R}^m$ where $n = N_s \times N_c$ and $m > 0$, from the space of base Shape-Tailored Descriptors to another vector space, with better invariance properties. In other words, f takes in $\mathbf{u}(x) \in \mathbb{R}^n$ at a particular pixel and returns a descriptor with m components. We choose f to be the output of a fully-connected neural network. Since we will eventually use the descriptor to discriminate between descriptors of different regions, we learn f by learning a Siamese neural network [27] designed to discriminate descriptors of different segmentation regions.

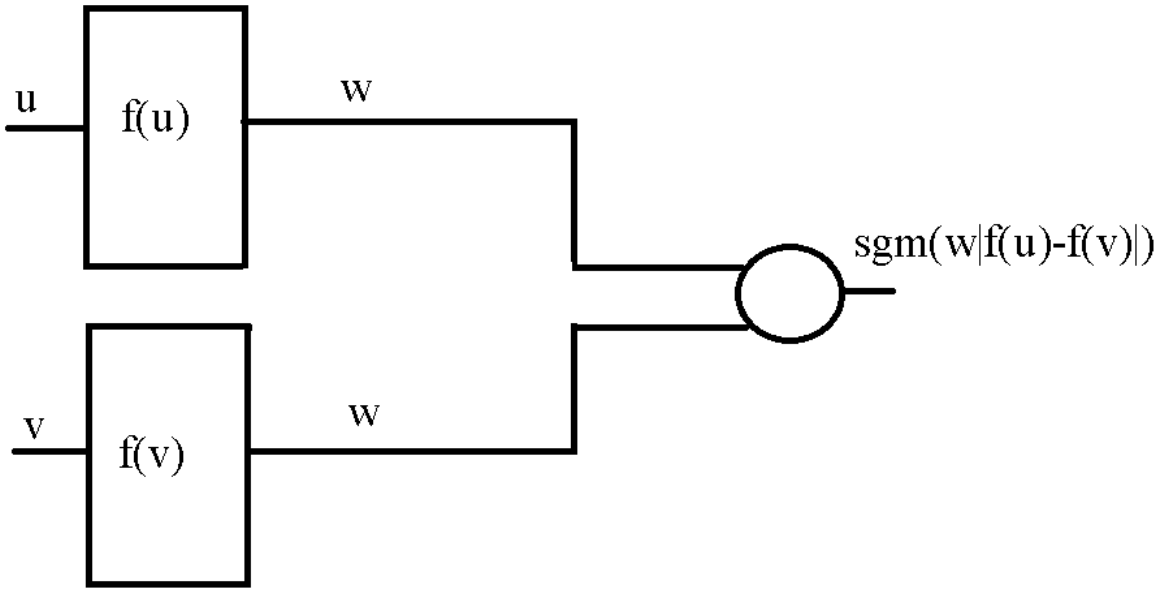

Figure 5.1: **Siamese Network for Metric Learning.**

Thus, the overall system is a symmetric function, $D : \mathbb{R}^n \times \mathbb{R}^n \to [0, 1]$, with a value of 1 indicating the descriptors are from different segmentation regions, and 0 indicating the descriptors are from the same region. The architecture is shown in Figure 5.1. The network takes in two different base descriptors $\mathbf{u}(x)$ and $\mathbf{v}(x)$, each are input to f and the output are two descriptors with m-components, then a weighted $\mathbb{L}^2$ norm of the difference of the descriptors are computed, followed by a sigmoid function. Note that for descriptors at pixels x and y, the metric from the Siamese network is defined as

$$D(\mathbf{u}(x), \mathbf{v}(y))^2 := \|f(\mathbf{u}(x)) - f(\mathbf{v}(y))\|_w^2 = \sum_{i=1}^{m} w_i |f(\mathbf{u}(x))_i - f(\mathbf{v}(y))_i|^2, \quad w_i \geq 0 \tag{5.2}$$

where w_i, $i = 1, \ldots, m$ are weights, and $f(\mathbf{u}(x))_i$ is the i^{th} component of $f(\mathbf{u}(x))$.

5.0.3 Training Data

The training data to train the network is generated from ground truth segmentations of images in the training set of images. Given a training image, we compute base Shape-Tailored Descriptors from the ground truth segmentation. For any pair of pixels x and y in adjacent ground truth regions or the same region in the same image, we form the training data as

$$D(\mathbf{u}_l(x), \mathbf{u}_k(y)) = \begin{cases} 0 & x, y \in R_l, l = k \\ 1 & x \in R_l, y \in R_k, l \neq k \end{cases},$$

where $\mathbf{u}(x)$ is the base Shape-Tailored Descriptor computed within R_l at x and $\mathbf{v}(x)$ is the base shape-tailored descriptor computed within R_k at y. Note we only choose adjacent regions since during segmentation, only discriminating between adjacent regions will be needed.

During segmentation at test time, we will solve a joint problem for the base descriptors $\mathbf{u}$ and the regions of the segmentation. The method iteratively updates the regions and the base descriptors. Thus, the metric D also needs to discriminate shape-tailored descriptors, when the descriptors are not computed on the ground truth segmentation. To this end, we perturb the ground truth segmentations by dilations and erosions to form regions $\tilde{R}_l$, and compute base shape-tailored descriptors $\tilde{\mathbf{u}}$ within the perturbed regions. This simulates possible base-descriptors anticipated during test time. We augment the training data with these descriptors as follows:

$$D(\tilde{\mathbf{u}}_l(x), \tilde{\mathbf{u}}_k(y)) = \begin{cases} 0 & x, y \in R_l, l = k \\ 1 & x \in R_l, y \in R_k, l \neq k \end{cases},$$

where x and y are in the same or adjacent ground truth regions R_l and R_k. Note that the descriptors are computed in the perturbed regions, whereas the distance above is

defined according to the ground-truth regions where pixels belong.

5.1 Segmentation

In this section, we describe our method for segmentation by using the invariant descriptors and the metric learned in the previous section.

5.1.1 Optimization Problem

We assume that the image consists of N_r regions with a constant in location learned shape-tailored descriptor in each region. We design an optimization problem for segmentation to be optimal when the regions are placed so that the learned shape-tailored descriptors are nearly constant within the regions. Let $\mathbf{u}^i(x) \in \mathbb{R}^n$ denote the base shape-tailored descriptor within region R_i, and let $\mathbf{a}^i \in \mathbb{R}^m$ be the constant learned shape-tailored descriptor representing the region, which is unknown. The energy for segmentation is as follows:

$$E(\{R_i\}_{i=1}^{N_r}) = \sum_{i=1}^{N_r} \int_{R_i} \|f(\mathbf{u}^i(x)) - \mathbf{a}^i\|_w^2 \, \mathrm{d}x + \beta \int_{\partial R_i} \mathrm{d}s, \tag{5.3}$$

where there are N_r regions, $\beta > 0$, and the second term above is to induce spatial regularity of the segmentation and consists of penalizing boundary length ($\mathrm{d}s$ is the arc-length element). The first term measures how similar the learned shape-tailored descriptor at each pixel within a region is to a constant vector $\mathbf{a}^i$. Thus, the optimal regions will be such that the regions have nearly constant learned descriptors within regions. This energy can be seen as a generalization of the energies considered by [51, 49].

5.1.2 Optimization Algorithm

If we minimize in $\mathbf{a}^i$, we see that the optimizer is $\mathbf{a}^i = 1/|R_i| \cdot \int_{R_i} f(\mathbf{u}^i(x))\,\mathrm{d}x$ where $|R_i|$ denotes the area of R_i, i.e., the average value of the learned descriptor within the region. Since the energy above is non-convex in the regions, as the descriptor $\mathbf{u}^i$ depends on R_i non-linearly and f non-convex, we use a gradient descent to optimize the energy. The gradient with respect to the boundary of R_i of the i^{th} term, using techniques from [26], is

$$(\|f(\mathbf{u}^i) - \mathbf{a}^i\|_w^2 + \kappa_i)N_i + (\mathrm{tr}[(D\mathbf{u}^i)^T D\hat{\mathbf{u}}^i] + (\mathbf{u}^i - \mathbf{J})^T A^{-1}\hat{\mathbf{u}}^i)N_i \tag{5.4}$$

where κ_i is the signed curvature of ∂R_i, N_i is the inward normal to ∂R_i, tr is the trace, D is the derivative, A is a diagonal matrix of size n with diagonal entries $(\alpha_1, \ldots, \alpha_1, \ldots, \alpha_{N_s}, \ldots, \alpha_{N_s})$, $\mathbf{J} = (J_1, \ldots, J_{N_c}, \ldots, J_1, \ldots, J_{N_c})^T$ is a vector of size n, and $\hat{\mathbf{u}}^i$ satisfies the PDE

$$\begin{cases} \hat{\mathbf{u}}^i(x) - A\Delta\hat{\mathbf{u}}^i(x) = 2\nabla f(\mathbf{u}^i(x))[f(\mathbf{u}^i(x)) - \mathbf{a}^i] & x \in R_i \\ \nabla\hat{\mathbf{u}}^i(x) \cdot N_i = 0 & x \in \partial R_i \end{cases}.$$

Note that the first term in (5.4) arises from the variation of the integrals as the boundary is deformed, and the second term arises from the variation of the descriptor as the boundary is changed. The gradient ∇f, which involves the neural network, can be approximated numerically. However, for simplicity of implementation, we neglect the variation of the descriptor since the numerical algorithm will involve only small changes of the boundary at each iteration and the descriptors $\mathbf{u}^i$ do not change much, and so the term is negligible.

To implement the gradient descent numerically, we represent the regions with relaxed indicator or "level-set" functions $\phi_i : \Omega \to [0, 1]$, $i = 1, \ldots, N_r$. R_j is the region where ϕ_j achieves the maximum over all $i = 1, \ldots, N_r$. We can then convert

the boundary evolution into an evolution of ϕ_i analogous to level set methods [76]. In order to extend the evolution beyond just the boundary, we extend the terms in the gradient to a band around the boundary. Computing the full gradient of the energy and neglecting variation of the descriptor terms, our algorithm to minimize the energy is given in Algorithm 2.

Algorithm 2 Gradient Descent of Learned Shape-Tailored Energy

Input: An initialization of ϕ_i
repeat
  Set regions: $R_i = \{x \in \Omega : i = \text{argmax}_j \phi_j(x)\}$
  Compute dilations, $D(R_i)$, of R_i
  Compute $\mathbf{u}^i$ in $D(R_i)$, compute $\mathbf{a}^i = 1/|R_i| \cdot \int_{R_i} \mathbf{u}^i(x)\,\mathrm{d}x$.
  Compute band pixels $B_i = D(R_i) \cap D(\Omega \backslash R_i)$
  Compute $G_i = \|f(\mathbf{u}^i(x)) - \mathbf{a}^i\|_w^2$ for $x \in B_i$. f is evaluated from the neural network.
  Update pixels $x \in D(R_i) \cap D(R_j)$ as follows:

$$\phi_i^{\tau+\Delta\tau}(x) = \phi_i^{\tau}(x) - \Delta\tau(G_i(x) - G_j(x))|\nabla\phi_i^{\tau}(x)| + \Delta\tau \cdot \beta\kappa_i|\nabla\phi_i^{\tau}(x)|.$$

  Update all other pixels as

$$\phi_i^{\tau+\Delta\tau}(x) = \phi_i^{\tau}(x) + \Delta\tau \cdot \beta\kappa_i|\nabla\phi_i^{\tau}(x)|.$$

  Clip between 0 and 1: $\phi_i = \max\{0, \min\{1, \phi_i\}\}$.
until regions have converged

Chapter 6

Unsupervised Learning

Next, we tackle the problem of training in an unsupervised fashion. The idea is to have an annotation independent learning. We want to learn key properties of images on the run without the need for information about the ground truth in the target images unlike supervised learning framework shown in Figure 6.1. Our energy 6.1 depends on region and weights of the Neural Network, both of which are learned. This makes the algorithm independent of any training data. The idea is be to learn "weights" of the neural network and segmentation which will minimize an "energy" on the image. The energy is defined in a way which matches our understanding of the natural texture images.

The energy for unsupervised learning is defined below 6.1. We want to minimize this energy for learning without supervision. The key idea is to construct a CNN like NN where receptive fields of the filters are tailored to region of interest. This is accomplished by projecting the filters onto the space of shape-tailored scale space and its gradients, this mimics the approximation of functions with Hermite basis. The detailed structure of the a single layer of the network can be seen in figure 6.2.

$$E(I, R, W) = \sum_i \int_{R_i} |F_W[I](x) - a(x)|dx \tag{6.1}$$

where $F_W[I]$ is the output of the Neural Network. Let r denote a rectified linear unit, L_W are linear transformation defined as $L_W(x) = W_1 x + W_0$ and T is defined as a smoothing layer of the form given below in equation (6.2).

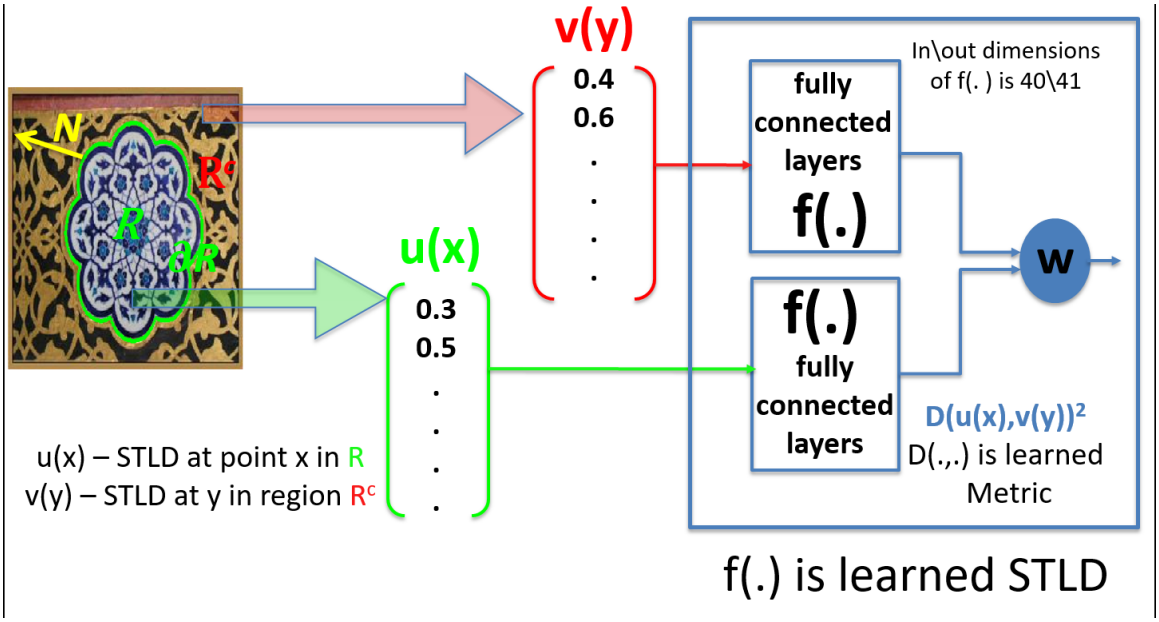

Figure 6.1: A schematic of the supervised learning scheme using Siamese twin network. Shape-tailored descriptors are extracted from the image and fed to a Siamese twin network. The network learns invariant features which are then used in applications likes segmentation.

$$T[I](x) = \int_R K(x,y)dy \tag{6.2}$$

Notice that in our example we use $T[I]$ as solution of Poisson equation (see (6.3)), and $K(.,.)$ in equation 6.2 is the Green's function of the Poisson equation (6.3).

$$\begin{cases} T[I](x) - \alpha\Delta T[I](x) = I(x) & x \in R \\ \nabla T[I](x) \cdot N = 0 & x \in \partial R \end{cases}, \tag{6.3}$$

One layer of the NN is represented as $f = r \circ L_w \circ T$. And $F = f_0 \circ f_1 \circ f_2 \circ ....f_m$ and $f_0, ...f_m$ represent layers 0 to layers m of the network. We would like to jointly optimize E w.r.t $W_0, W_1, W_2....W_m$ which are the weights of the linear transformation layers of the NN. Below we give the variations of different quantities of the network.

First, we calculate the variation of the smoothing layer of the Neural Network. The variation of the smoothing layer w.r.t a perturbation turns out to be a smoothing

of the perturbation.

$$\delta T(I).\delta I = T[\delta I] \tag{6.4}$$

Next, we calculate the variation of a complete layer of the neural network.

$$\delta f(I).\delta I = (r \circ L_W)'(T(I))\delta T(I).\delta I = (r \circ L_W)'(T(I))T[\delta I] \tag{6.5}$$

Finally, we calculate the variation of the complete Neural Network.

$$\delta F(I).\delta I = \delta[f_0 \circ f_1 \circ ....f_m].\delta I = \delta f_0(f^{m-1}(I)) \circ \delta f_1(f^{m-2}(I))... \circ \delta f_m(I).\delta I \tag{6.6}$$

The variation of the Energy of unsupervised learning is given below.

$$\delta E(I).\delta I = 2 < (F[I] - a), \delta F(I).\delta I >_{L^2} \tag{6.7}$$

Next, we compute the derivative w.r.t. the weights W_i will denote the weights for the i^{th} layer. We note that $\partial_W f(I) = r'(L_W \circ T(I))\partial_W L_W \circ T(I)$. Also,

$$\partial_{W_i} F[I] = (\delta f)^{m-i}(f^{i+1}(I))\partial_W f(f^{i-1}(I)) \tag{6.8}$$

Finally, We calculate the gradient of the unsupervised segmentation energy w.r.t. the weights of liner transformation layers 6.3.

$$\partial_{W_i} E(I) = \sum_i \int_{R_i} 2(F[I](x) - a(x))(\delta f)^{m-i}(f^{i+1}(I))\partial_{W_i} f(f^{i-1}(I))dx \tag{6.9}$$

A key point to notice here is that the formulation of $E(R, W)$ is for general unsupervised segmentation but this energy is also valid for supervised segmentation if R is known before hand. With this formulation we can extend the supervised segmentation of previous chapter to multiple layers where each layer has it's own

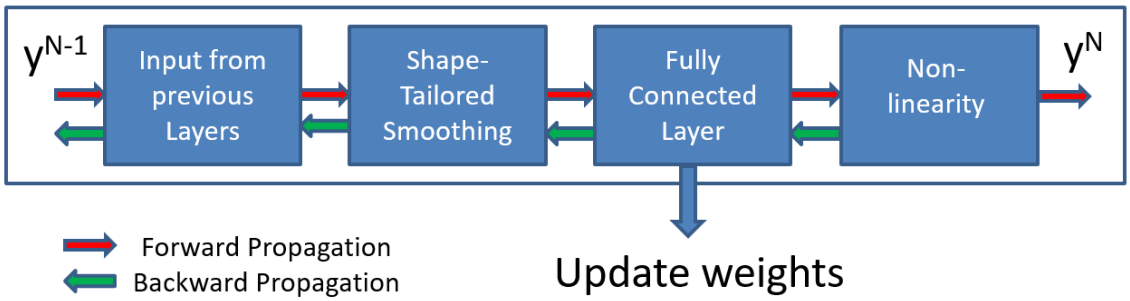

Figure 6.2: A schematic of the unsupervised learning scheme where we don't need any training data. The descriptors are learned on the fly during the segmentation. We can learn descriptors for any size dataset or even an individual image. The network mimics CNN, which are ubiquitous in vision application, with the exception that the receptive fields of the descriptors are tailored to the region of interest only. This is achieved by calculating filters as projections on shape-tailored scale space and its gradients. This is somewhat similar to approximating of a function with Hermite basis.

smoothing component.

6.1 Energy Functions for Unsupervised Training

The tools we have provided are of general nature and can be used with different energy functions. We have used a variety of energy functions each with it's unique advantages and disadvantages. In this section we provide the summary of different energy functions tried in this work.

6.1.1 Energy based on Invariance Term

The first energy term we tried is based on invariance of descriptors in textured region. Invariance is a key property of any texture descriptor, i.e. a good texture descriptors should be invariant to the intrinsic and extrinsic variation of texton in a texture region. The energy is provided in equation (6.10)

$$E(R,W) = \sum_i \int_{R_i} |(F_W(u(x)) - a(x))|^2 dx \tag{6.10}$$

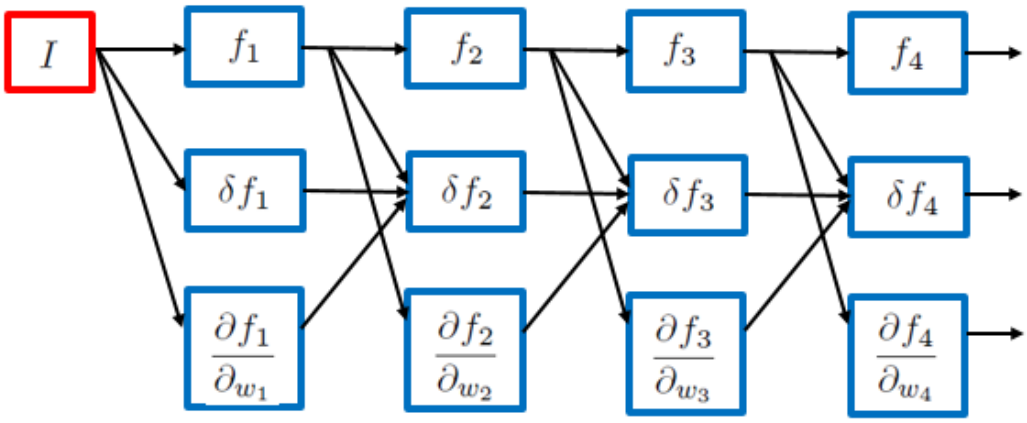

Figure 6.3: A schematic of the forward pass and the pass for gradient calculation of the network. Notice that due to the smoothing layer gradient of the network is not calculated with backward pass rather it is calculated with a modified forward pass.

Where $F_W(u)$ is the learned invariant descriptor and $a(x)$ is the texture model. If $a(x)$ the texture model is jointly calculated with the $F_W(u(x))$ the invariant descriptor, the solution obtained has zero energy, where descriptor is zero everywhere. In this case we lose the discriminability, which is required to differentiate between descriptors from different textures. To tackle this issue we fix the texture descriptors as foreground and background mask respectively. As a result the learned descriptor can discriminate between foreground and background. A drawback with this approach is that the learned descriptor can not be generalized to any general texture. Since the texture model used is based on foreground/ background respectively rather than texture description for general textures. Such approach will be helpful in applications like saliency detection, where we are more interested in discriminating "salient" features of an image rather than describing these "salient" features.

6.1.2 Energy based on Invariance and Discrimination Terms

The second approach we try is to use discrimination term along with invariance term in our energy. This eliminate the problem faced with only the invariance term where

the learned descriptor goes to zero and we lose the discrimination between descriptor for different textons. The energy function is provided in (6.11)

$$E(R,W) = \sum_i \int_{R_i} [|(F(u(x)) - a(x))|^2 dx - |(F(u(x)) - a_o(x))|^2 dx] \tag{6.11}$$

A careful study of the energy shows that first term is similar to the energy of the previous section. However the addition of the second term where we are maximization the difference of the descriptor on a region from the texture description of the complimentary region prevents the case of zero learned descriptors. Furthermore, now the texture model can be jointly calculated with the descriptors. One issue faced with this energy is that it is not bounded from below hence we might get texture models which are very large in value. In order to prevent this case we can fix the non-linearity of the last layer of the network as a sigmoid function. Now, the energy function is bounded from above and below and the texture models $a_i \in [0, 1]$. Now, this energy function can provide good learned descriptors which are invariant to intrinsic and extrinsic nuisance of textons and discriminative of different textures.

6.1.3 Energy based on Discrimination of Texture Model

The final energy function we have tried with our approach is based on maximizing the discrimination of texture model of different region.

$$E(R,W) = -\sum_i \sum_{j \neq i} |(a_i - a_j)|^2 \tag{6.12}$$

where a_i is defined as

$$a_i = \frac{1}{|R_i|} \int_{R_i} F(u(x)) dx \tag{6.13}$$

This energy gives the most promising results of all cases. In order to prevent the case where both a_i and a_j are either decreasing or increasing, which will result in very large values of the texture model while there difference would be small, we use the modified gradient flow presented in [109]. In this gradient scheme instead of going in the gradient direction we go in the component of the gradient direction where the quantities evolve in the direction we want them to evolve. This makes sure that the difference between the a_i and a_j is always increasing. A comparison of different energy function in unsupervised segmentation is given in Figure 6.1.3

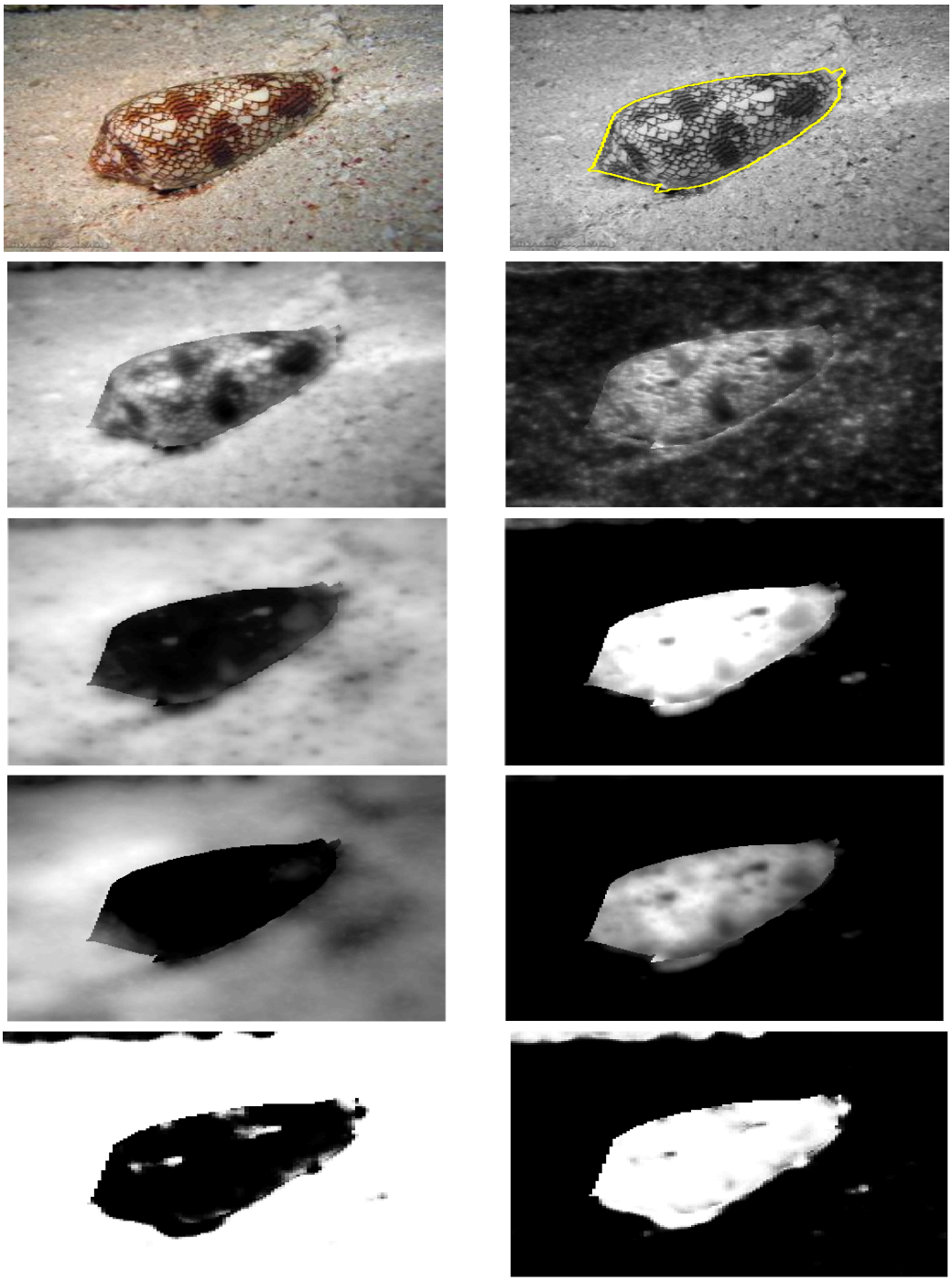

Figure 6.4: **Figure shows the comparison of different energies.** The learned weights are applied to the base descriptors calculated on ground truth. The better the energy the more invariance of descriptor on region of interest and discrimination between foreground and background will be seen. [Top Row] Image and ground truth, [Row Two] base shape-tailored descriptors, [Row Three] results of energy based on invariance, [Row Four] results of energy based on invariance and discrimination, [Row Five] results of energy based on discrimination of texture models.

Chapter 7

Experiments

In this section, we test our methods on image and motion segmentation problem on different challenging datasets.

7.1 Shape-Tailored Local Descriptors Experiments

The first set of experiments tests the ability of Shape-Tailored Descriptors to discriminate a variety of real-world textures. To this end, we compare Shape-Tailored Descriptors to a variety of descriptors for segmenting textured images based on the piecewise constant model. We compare on both a standard synthetic dataset and then on a dataset of real world images. The second set of experiments shows sample application of Shape-Tailored Descriptors to the problem of disocclusions in object tracking where objects consist of multiple textured regions. We thus use the piecewise smooth model. This shows that a state-of-the-art method in object tracking can be improved using Shape-Tailored Descriptors.

7.1.1 Performance of STLD in Segmentation

We test the performance of our new STLD by testing its ability to discriminate textures on two datasets, and then compare to other descriptors. Code and datasets will be available [1].

Datasets: The first dataset is a synthetic data set. It consists of images constructed from the textured images in the Brodatz dataset. Each is composed of two

[1] `https://sites.google.com/site/shapetailoreddescriptors/`

different textures. One texture is used as background and the other texture is masked with a shape from the MPEG 7 shape dataset and used as the foreground. The dataset consists of 50 images (5 different masks times 10 different foreground/background pairs). The second dataset consists of images obtained from Flickr that have two dominant textures. A variety of real textures (man-made and natural) have been chosen with common nuisances (e.g., small deformations of the domain, some illumination variation). The size of the dataset is 256 images with 20 training images. We have hand segmented these images to facilitate quantitative comparison.

Methods Compared: We compare STLD to various other recent descriptors that are used for texture segmentation. Descriptors include simple global means used in Chan-Vese [49], global histograms (*Global Hist* [32]), local means (LAC [52]), more advanced descriptors based on local histograms in predefined neighborhood sizes (*Hist* [38]), SIFT descriptors (*SIFT*), the entropy profile (*Entropy* [44]), and non-STLD. For methods that can be formulated with convex relaxations, we use the segmentation based on global convex methods [56], which are more robust than gradient descent. This does not include our method, which uses gradient descent. We also compare to the hierarchical segmentation approach (*gPb* [43]). Note that gPb is not a descriptor, but uses several descriptors (e.g., Gabor filtering, and local histograms) to build a segmentation after edge detection. It is also for more general image segmentation, which is not the goal of our work, but we compare to it since it uses several descriptors. We also compare to [36] (CB), a recent texture segmentation method build on gPb, but using different edge detection.

Parameters: For all the methods, the training images were used to obtain the best regularity parameter γ, and that same parameter was used for the rest of the images. For STLD, the scales $\alpha = (5 + (10, 20, 30, 40, 50)) \times (s/256)^2$ where s is the size of the image, and $\theta = 0, \pi/8, \ldots, 7\pi/8$ are kept fixed on the whole datasets. All methods that require initialization are initialized with a box tessellation pattern that

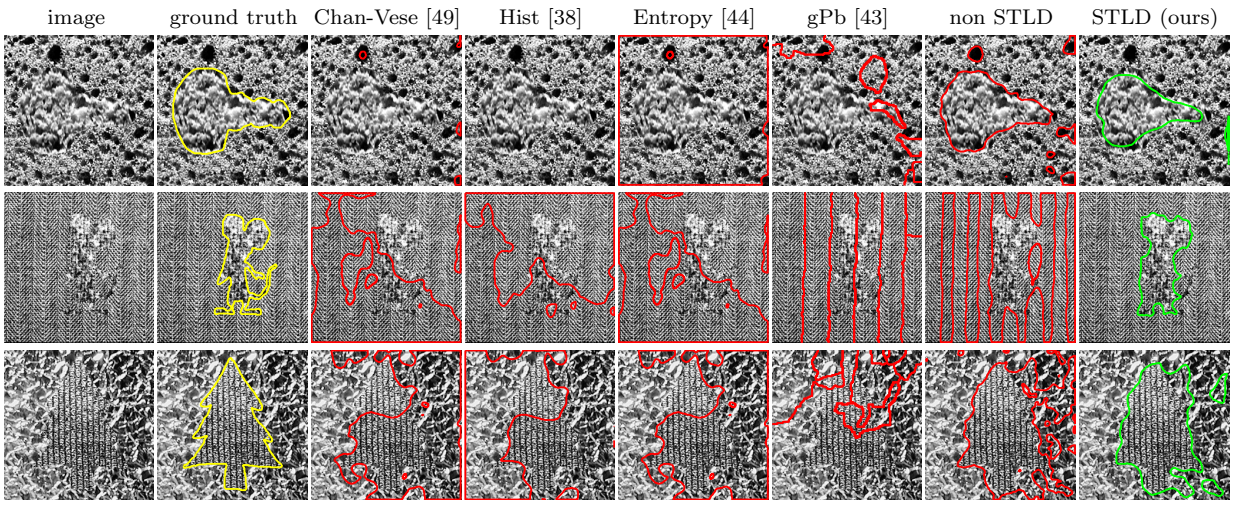

Figure 7.1: **Sample Results on the Synthetic (Brodatz) Texture Dataset.**

is standard in these types of methods.

Discussion of Qualitative Results: Figure 7.1 shows sample visualizations of results on the Brodatz dataset. Figure 7.2 to Figure 7.8 shows sample results on our Real Texture dataset. Results are shown only for the top performing methods tested, and ground truth is displayed. STLD consistently performs well, clearly performing better than or at least as good as other methods. One can see that the boundaries are more accurate for STLD than non-STLD, and in many cases, the smoothing of data across textured regions also leads to more severe errors beyond overshooting the boundaries. The other region-based methods many times cannot capture the intrinsic texture differences on the datasets. The edge-based segmentation approach of gPb and CB works well detecting brightness edges, but in many cases does not detect texture boundaries. This may because sometimes texture boundaries are faint edges, and many times gPb and CB detect edges inside textons.

Discussion of Quantitative Results: Table 7.1 shows quantitative evaluation. We evaluate the algorithms using the evaluation protocol developed in [43]. The algorithms are evaluated both in terms of boundary and region accuracy by comparing to ground truth. For all metrics (except variation of information), a higher value

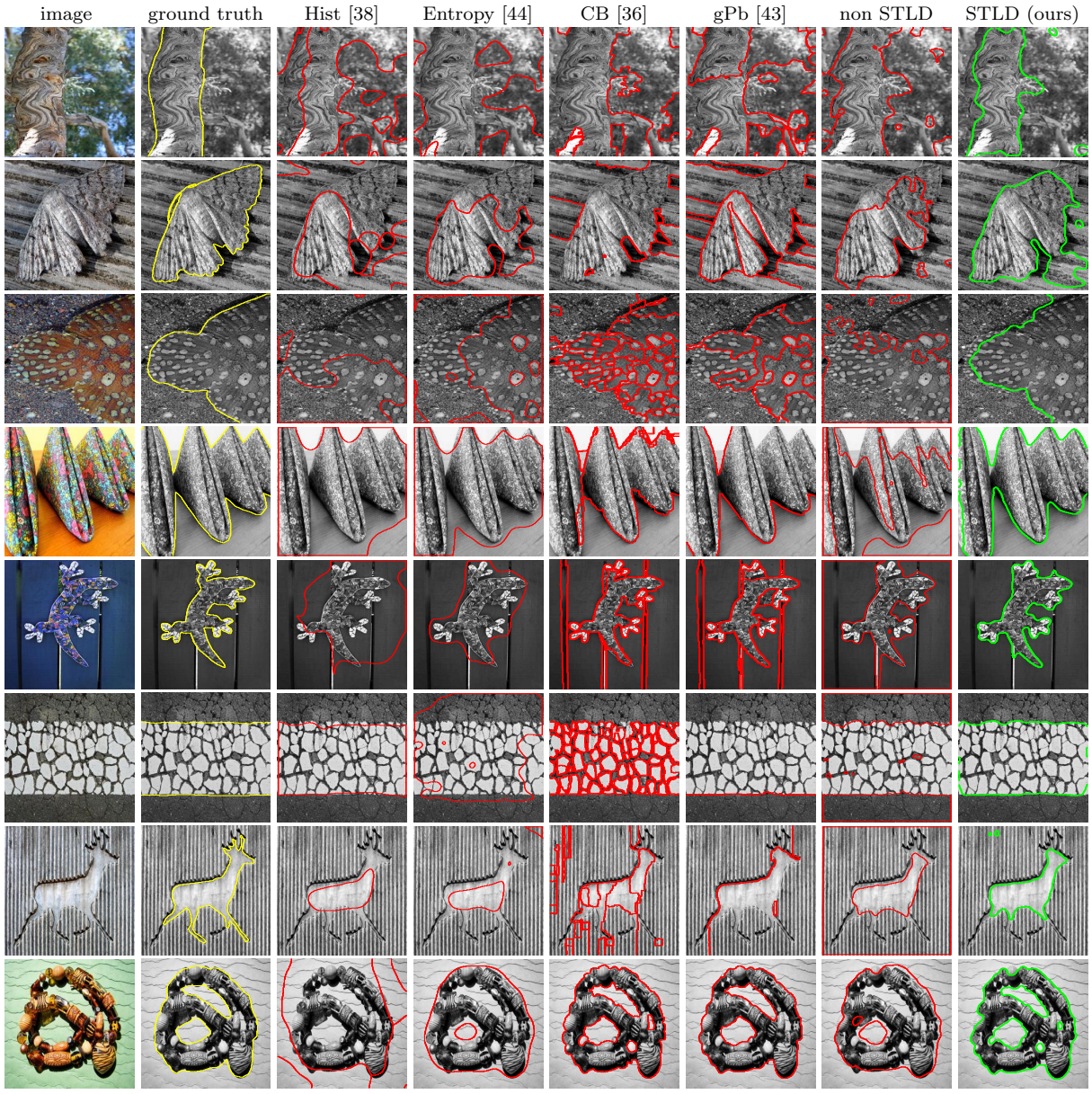

Figure 7.2: **Sample Results on the Real Texture Dataset. Segmentation boundaries are displayed for various methods.**

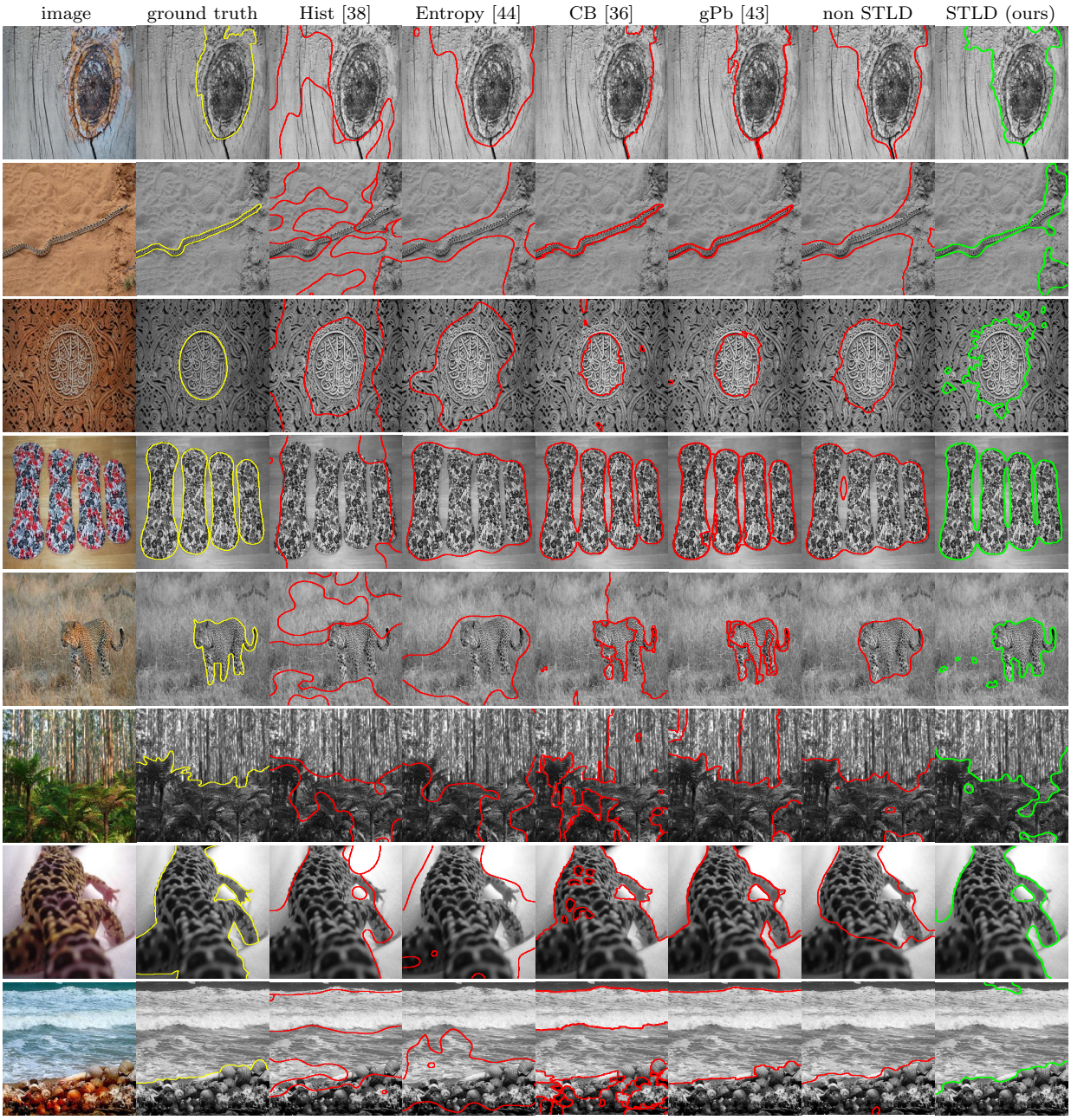

Figure 7.3: **Sample Results on the Real Texture Dataset. Segmentation boundaries are displayed for various methods.**

Figure 7.4: **Sample Results on the Real Texture Dataset. Segmentation boundaries are displayed for various methods.**

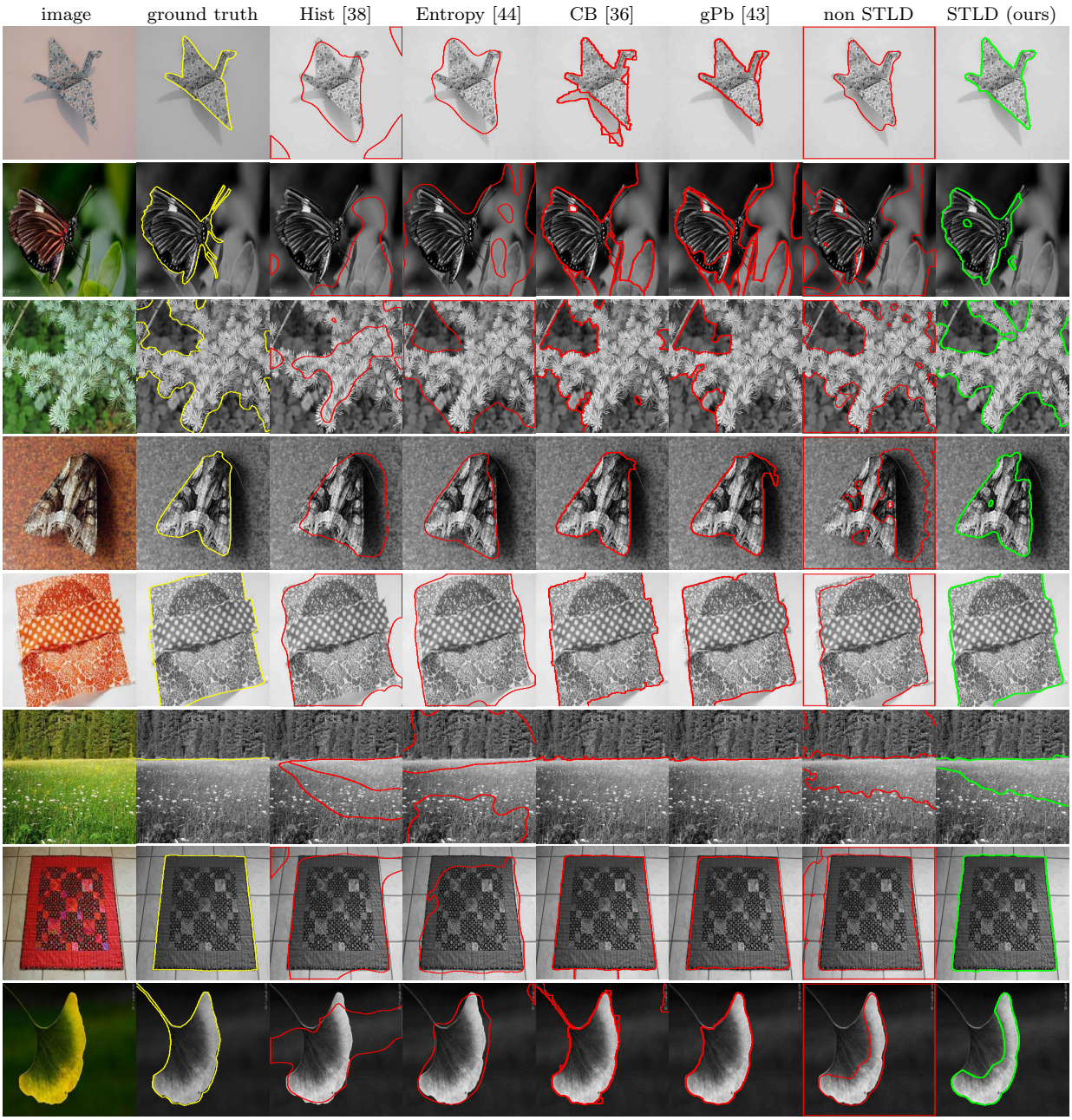

Figure 7.5: **Sample Results on the Real Texture Dataset. Segmentation boundaries are displayed for various methods.**

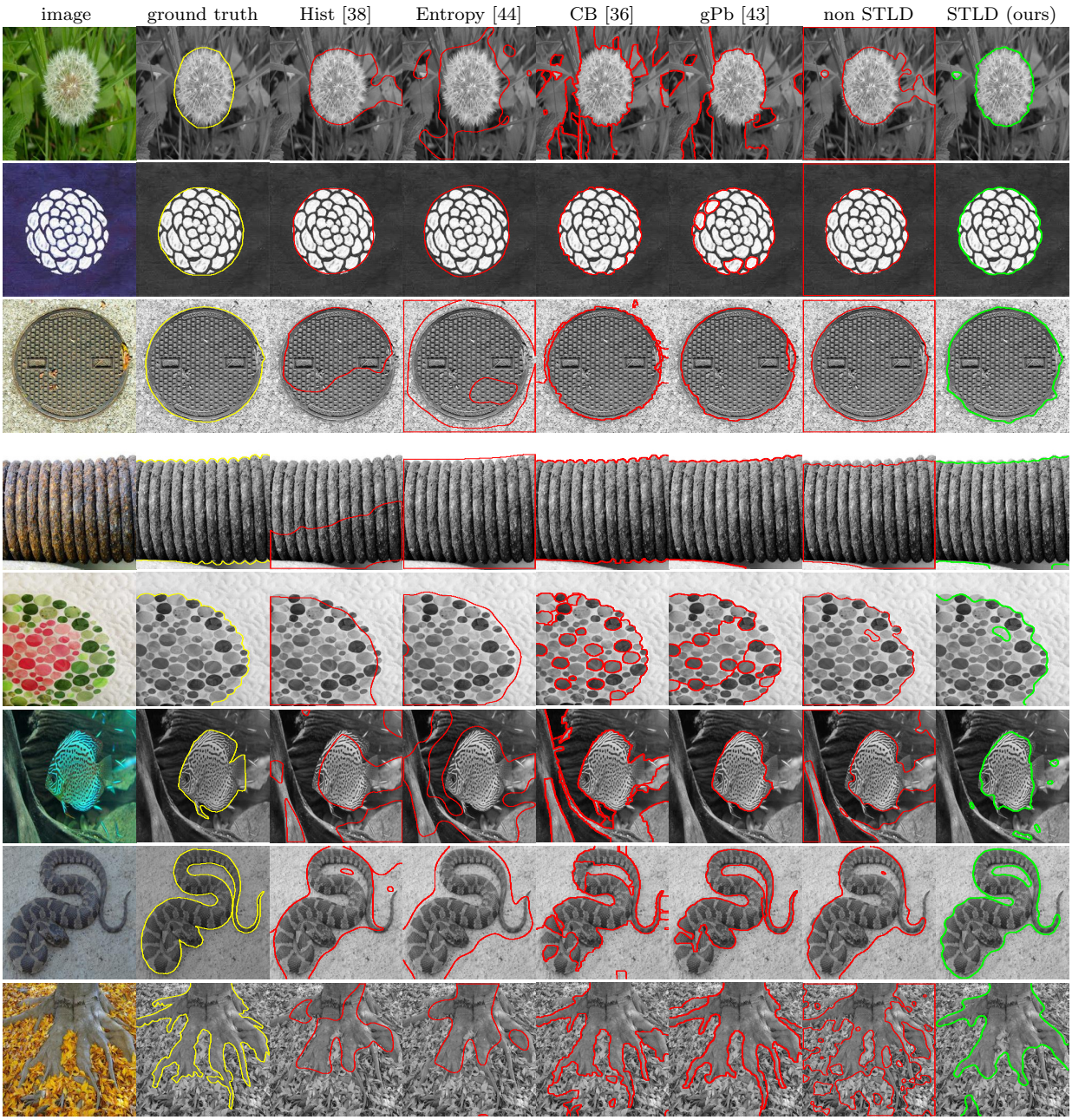

Figure 7.6: **Sample Results on the Real Texture Dataset. Segmentation boundaries are displayed for various methods.**

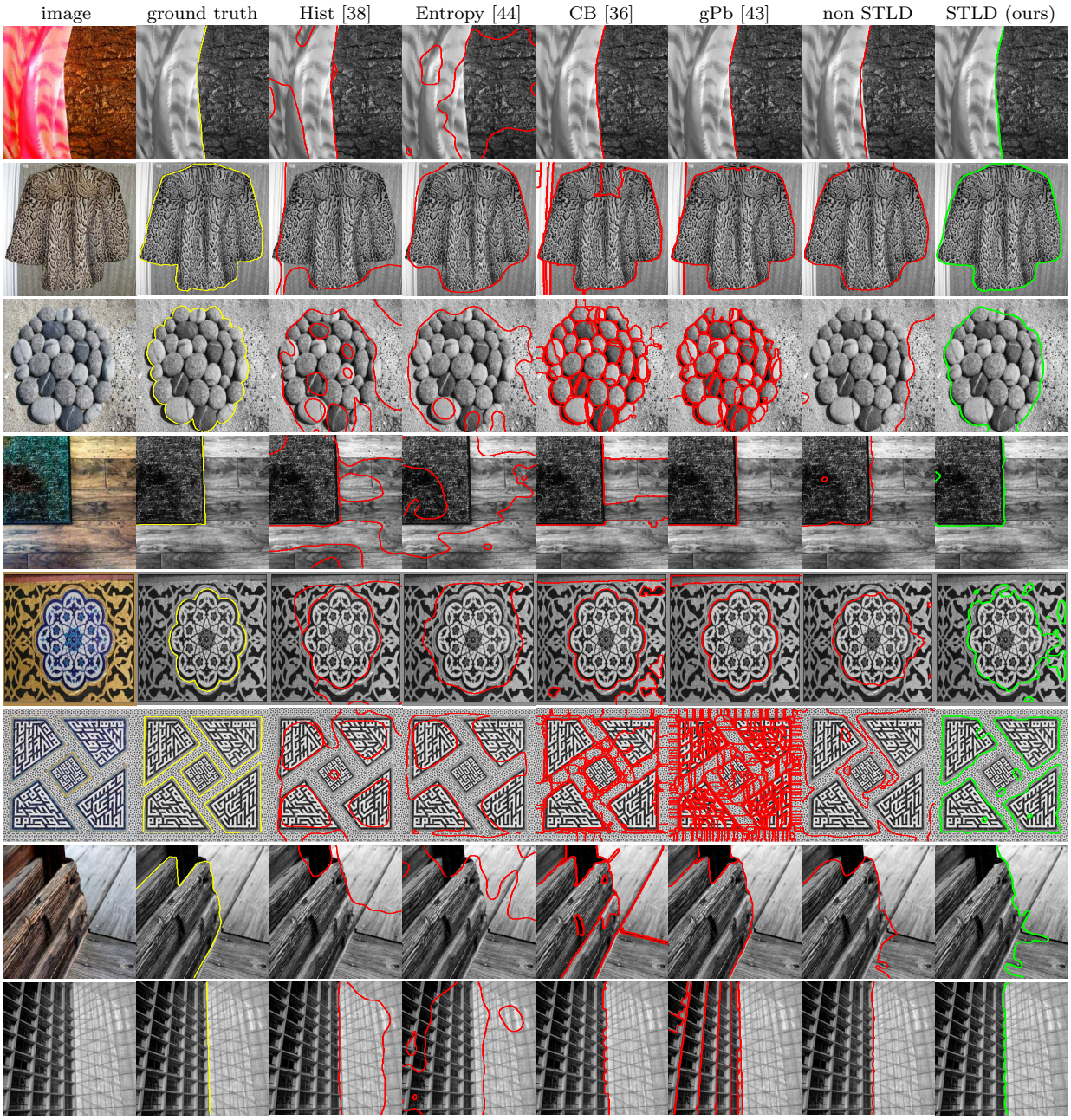

Figure 7.7: **Sample Results on the Real Texture Dataset. Segmentation boundaries are displayed for various methods.**

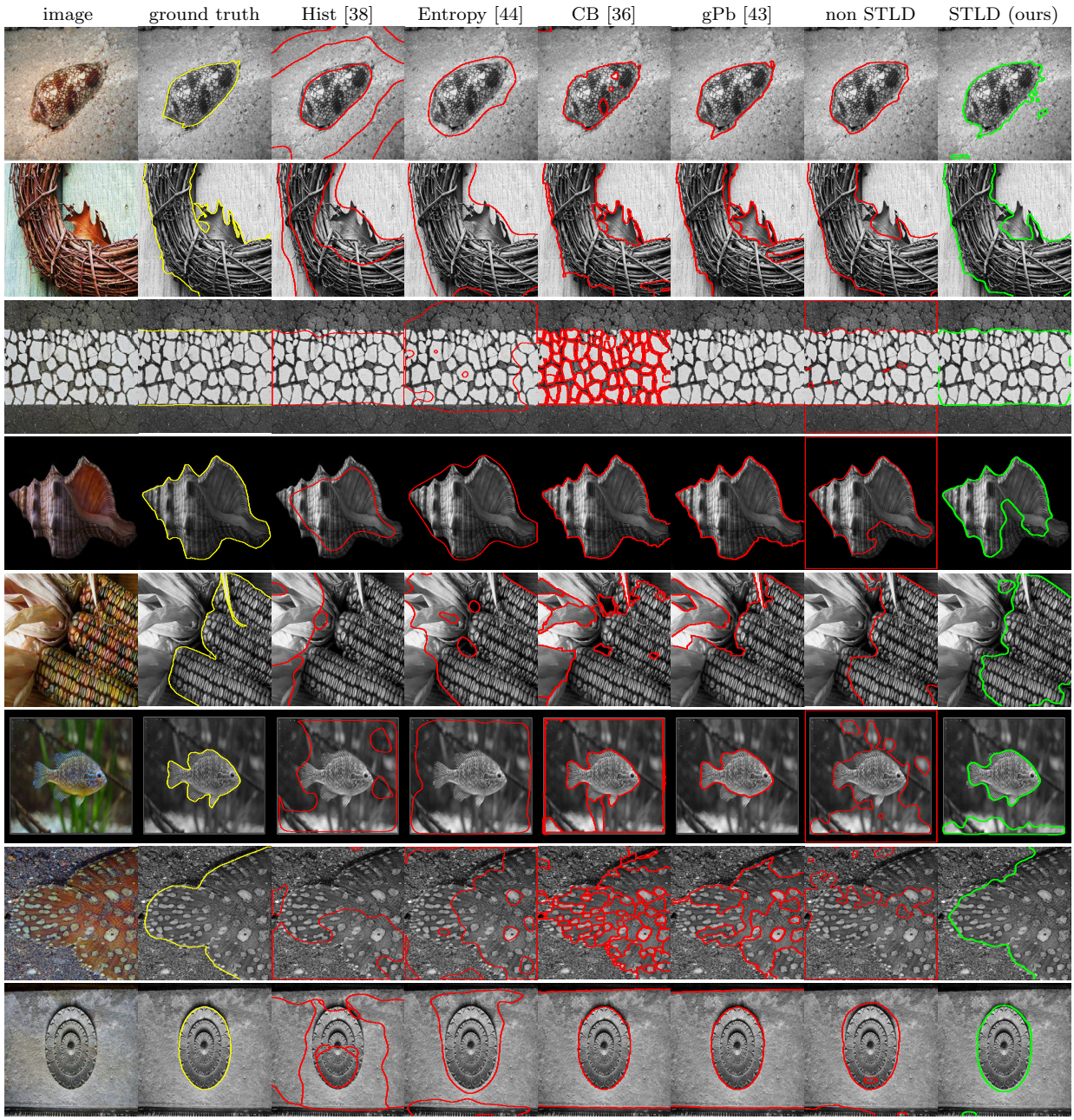

Figure 7.8: **Sample Results on the Real Texture Dataset. Segmentation boundaries are displayed for various methods.**

Brodatz Synthetic Dataset

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
STLD	**0.30**	**0.30**	**0.81**	**0.81**	**0.81**	**0.81**	**0.88**	**0.88**
non STLD	**0.28**	**0.28**	**0.78**	**0.78**	**0.77**	**0.77**	**0.98**	**0.98**
gPb [43]	0.20	0.20	0.56	0.56	0.57	0.57	1.17	1.17
SIFT	0.10	0.11	0.66	0.66	0.66	0.66	1.20	1.20
Entropy [44]	0.09	0.09	0.61	0.61	0.61	0.61	1.17	1.17
Hist-5 [38]	0.12	0.12	0.56	0.56	0.63	0.63	1.09	1.09
Hist-10 [38]	0.11	0.11	0.60	0.60	0.64	0.64	1.01	1.01
Chan-Vese [49]	0.09	0.09	0.61	0.61	0.61	0.61	1.17	1.17
LAC [52]	0.07	0.07	0.66	0.66	0.68	0.68	1.16	1.16
Global Hist [32]	0.10	0.10	0.38	0.38	0.52	0.52	2.41	2.41

Real Texture Dataset

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
STLD	**0.58**	**0.58**	**0.87**	**0.87**	**0.87**	**0.87**	**0.59**	**0.59**
non-STLD	0.17	0.17	**0.81**	0.81	**0.82**	0.82	**0.77**	0.77
gPb [43]	**0.50**	**0.54**	0.74	**0.84**	0.78	**0.86**	0.80	**0.65**
CB [36]	0.48	0.52	0.64	0.70	0.66	0.75	0.89	0.78
SIFT	0.10	0.10	0.55	0.55	0.59	0.59	1.44	1.44
Entropy [44]	0.08	0.08	0.74	0.74	0.75	0.75	0.95	0.95
Hist-5 [38]	0.14	0.14	0.66	0.66	0.70	0.70	1.18	1.18
Hist-10 [38]	0.13	0.13	0.66	0.66	0.70	0.70	1.19	1.19
Chan-Vese [49]	0.14	0.14	0.71	0.71	0.73	0.73	1.04	1.04
LAC [52]	0.09	0.09	0.55	0.55	0.58	0.58	1.41	1.41
Global Hist [32]	0.12	0.12	0.65	0.65	0.67	0.67	1.12	1.12

Table 7.1: **Summary of Results on Texture Segmentation Datasets**. Algorithms are evaluated using contour and region metrics (see text for details). Higher F-measure for the contour metric, ground truth covering (GT-cov), and rand index indicate better fit to the ground truth, and lower variation of information (Var. Info) indicates a better fit to ground truth. Bold red indicate best results and bold black indicates second-best results.

indicates better fit to ground truth. ODS and OIS are the best values of results of the algorithm tuned with respect to a threshold on the entire dataset (ODS) and each image individually (OIS), and the difference applies only to gPb. Our method out-performs all methods on all metrics.

7.1.2 Application of STLD to Disocclusions

We now show application of STLD to the problem of disocclusion detection in object tracking. One can track objects in a video by propagating an initial segmentation across frames, but two difficulties are self-occlusions and disocclusions of the object. Recently, [110] addressed the problem of self-occlusions and removed them from the

	Cheetah	CowFish	Turtle	WG Fish
Occlusion Tracker [110]	0.222	0.658	0.493	0.705
STLD	**0.937**	**0.929**	**0.958**	**0.909**

Table 7.2: **Quantitative Evaluation of Object Tracking Results.** Ground-Truth covering is used to evaluate results (higher means better fit to ground truth).

segmentation propagation. This propagation and self-occlusion removal step does not obtain the full object segmentation since there may be parts of the object that become disoccluded. [110] detects disocclusions by comparing pixel intensities outside the propagated segmentation to local color histograms of the propagated segmentation. Pixels that match the local distributions are classified as disocclusion and included as part of the object segmentation. Our descriptors are more descriptive than local color histograms and are thus able to deal with more challenging object appearances, especially textured objects. Thus, we now use STLD to perform the disocclusion detection by segmentation of STLD based on the piecewise smooth Mumford-Shah. This is initialized with the propagation of the segmentation from the previous frame based on [110]. This detects as disocclusions those pixels that have similar STLDs locally to the segmentation propagation. Note that a piecewise constant STLD model of two regions is not adequate since the object and background consist of multiple textures.

Results on four challenging videos are shown in Figure 7.9 and compared against [110]. Table 7.2 gives quantitative analysis. The videos contain objects with multiple textures, and the backgrounds also consist of multiple textures. In all sequences, $\beta = 10$, the scales α_i are chosen the same as in the previous section. Shape-Tailored Descriptors capture the textured object of interest accurately. [110] fails to capture disoccluded regions that are textured. These errors, slight at first as only small parts are disoccluded between frames, are then propagated forward and the method fails to segment the object accurately. Only 4 out of 50 frames are shown.

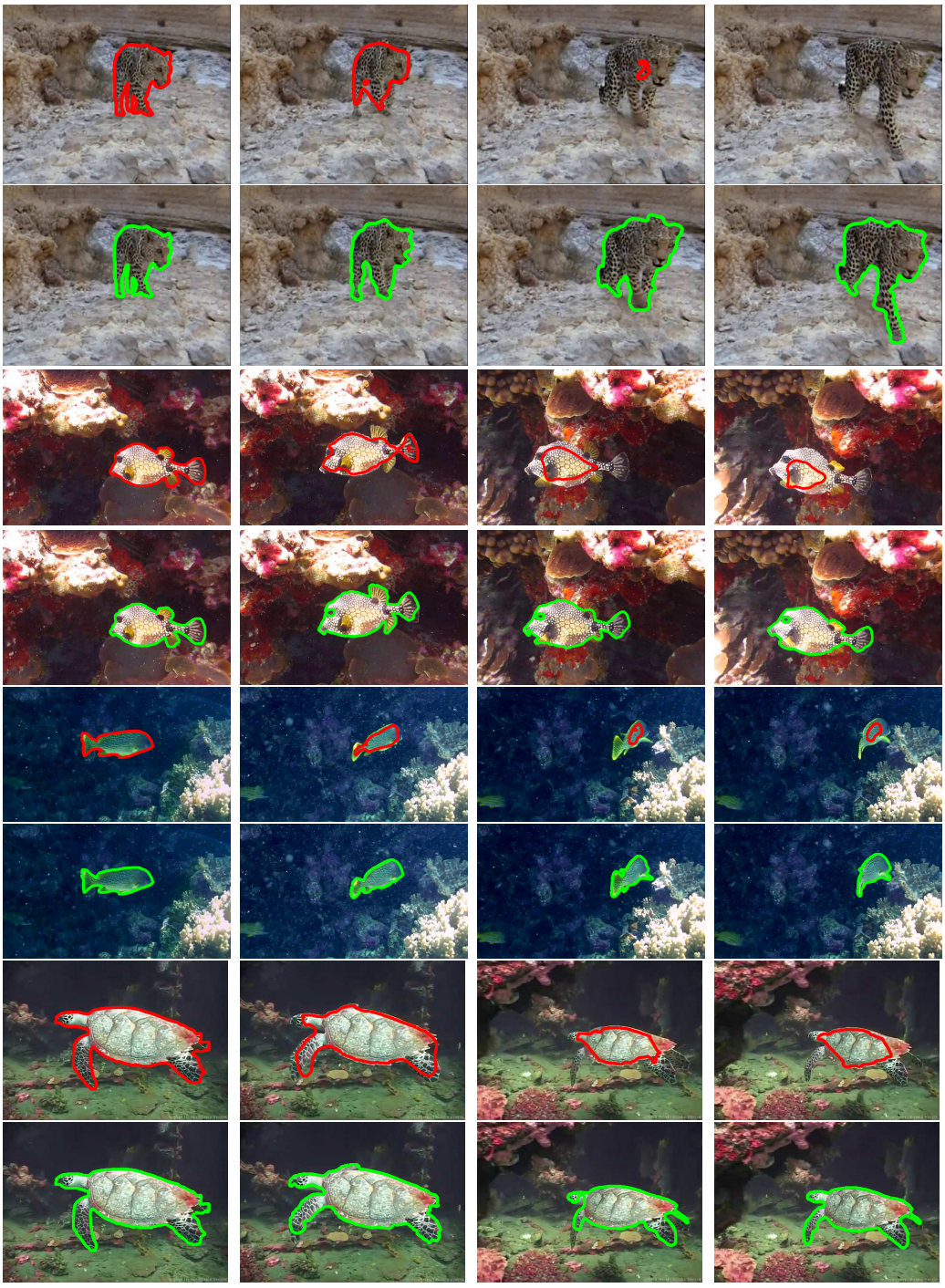

Figure 7.9: **Results on Textured Object Tracking**. [Top]: Results of a state-of-the-art method [110] (red). The method fails early since the disocclusion detection is based on local color histogram descriptors, which fail to capture textures. [Bottom]: Results of [110] by replacing local color histograms in disocclusion detection with STLD based on piecewise smooth Mumford-Shah. **(See video on website)**.

7.2 Continuum Scale Space Experiments

7.2.1 Texture Segmentation

Datasets and Methods Compared: We first test our method on texture segmentation, a task where multiscale information is important. We test on two datasets used in [26]. The Brodatz Synthetic Dataset has 198 images generated from textures in Brodatz and random shapes from MPEG dataset. The second is the Real-World Texture Dataset, which consists of 256 textured images obtained from photographs of real-world scenes. We use RGB color channels and binned oriented gradients at four angles, as the features for segmentation. Since the contribution in this paper is the use of shape-tailored scale spaces at a continuum of scales, we compare to [26] (STLD), which uses scale space but only considers a discrete number of scales. For reference, we include other segmentation methods. We use the abbreviations ExpPos, Uniform, and ExpNeg for the positive exponent exponential, uniform, and negative exponent exponential weights in our method. The methods are all initialized with a standard box tessellation.

Results on Brodatz: First, we compare on Brodatz with different weighting schemes introduced in Section 4.1.2 for continuum scale spaces against STLD. To compare weightings and not the quality of various approximations, we use (4.9) to compute the gradient. Images are 128×128 and we choose $\alpha = T = 10$ (corresponding to the max scale used in STLD) for all weightings. Results are displayed in Table 7.3. All weightings give similar results, and all are significantly more accurate than STLD. This indicates that using continuum scale space leads to increased performance.

Results on Real-World Texture Images: Since all results for different weightings are similar, we now use ExpNeg for comparison on the Real-World Texture Dataset because of its speed. Results, in Table 7.3, for $\alpha = 20$, show that the accuracy of the continuum scale space is greater than discrete scales (STLD). Sample

Brodatz Synthetic Dataset

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
ExpPos (ours)	**0.41**	**0.41**	**0.80**	**0.80**	**0.79**	**0.79**	**0.68**	**0.68**
ExpNeg (ours)	0.39	0.39	0.78	0.78	0.77	0.77	0.68	0.68
Uniform (ours)	0.40	0.40	0.79	0.79	0.78	0.78	0.68	0.68
STLD	0.33	0.33	0.71	0.71	0.70	0.70	0.74	0.74

Real-World Texture Dataset

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
ExpNeg (ours)	**0.60**	**0.60**	**0.91**	**0.91**	**0.91**	**0.91**	**0.45**	**0.45**
STLD	0.58	0.58	0.87	0.87	0.87	0.87	0.59	0.59
non-STLD	0.17	0.17	0.81	0.81	0.82	0.82	0.77	0.77
mcg [23]	0.51	0.54	0.74	0.82	0.77	0.85	0.80	0.66
gPb [43]	0.50	0.54	0.74	c0.84	0.78	0.86	0.80	0.65
CB [36]	0.48	0.52	0.64	0.70	0.66	0.75	0.89	0.78
SIFT	0.10	0.10	0.55	0.55	0.59	0.59	1.44	1.44
Entropy [44]	0.08	0.08	0.74	0.74	0.75	0.75	0.95	0.95
Hist-5 [38]	0.14	0.14	0.66	0.66	0.70	0.70	1.18	1.18
Hist-10 [38]	0.13	0.13	0.66	0.66	0.70	0.70	1.19	1.19
Chan-Vese [49]	0.14	0.14	0.71	0.71	0.73	0.73	1.04	1.04
LAC [52]	0.09	0.09	0.55	0.55	0.58	0.58	1.41	1.41
Global Hist [32]	0.12	0.12	0.65	0.65	0.67	0.67	1.12	1.12

Table 7.3: **Results on Texture Segmentation Datasets**. Algorithms are evaluated using contour and region metrics. Higher F-measure for the contour metric, ground truth covering (GT-cov), and rand index indicate better fit to the ground truth, and lower variation of information (Var. Info) indicates a better fit to ground truth.

representative visual results are shown in Figure 7.10.

Next, we test our approach with different choices of α using the ExpNeg weighting. We also compare against STLD in terms of speed and accuracy. Results are shown in Table 7.4. Results of STLD show that more than one scale is necessary, and faster speed by using fewer scales leads to worse segmentation. Second, results of ExpNeg show that the results are stable across different parameter choices for α. Finally, a speed comparison is performed between ExpNeg and STLD. Note that each scale that is used in STLD requires the solution of a PDE, whereas our approach of ExpNeg requires only a single PDE. This makes our continuum scale space approach computationally less expensive, as confirmed in Table 7.4. Our approach also requires only a single parameter in contrast to STLD that requires choosing a list of scales.

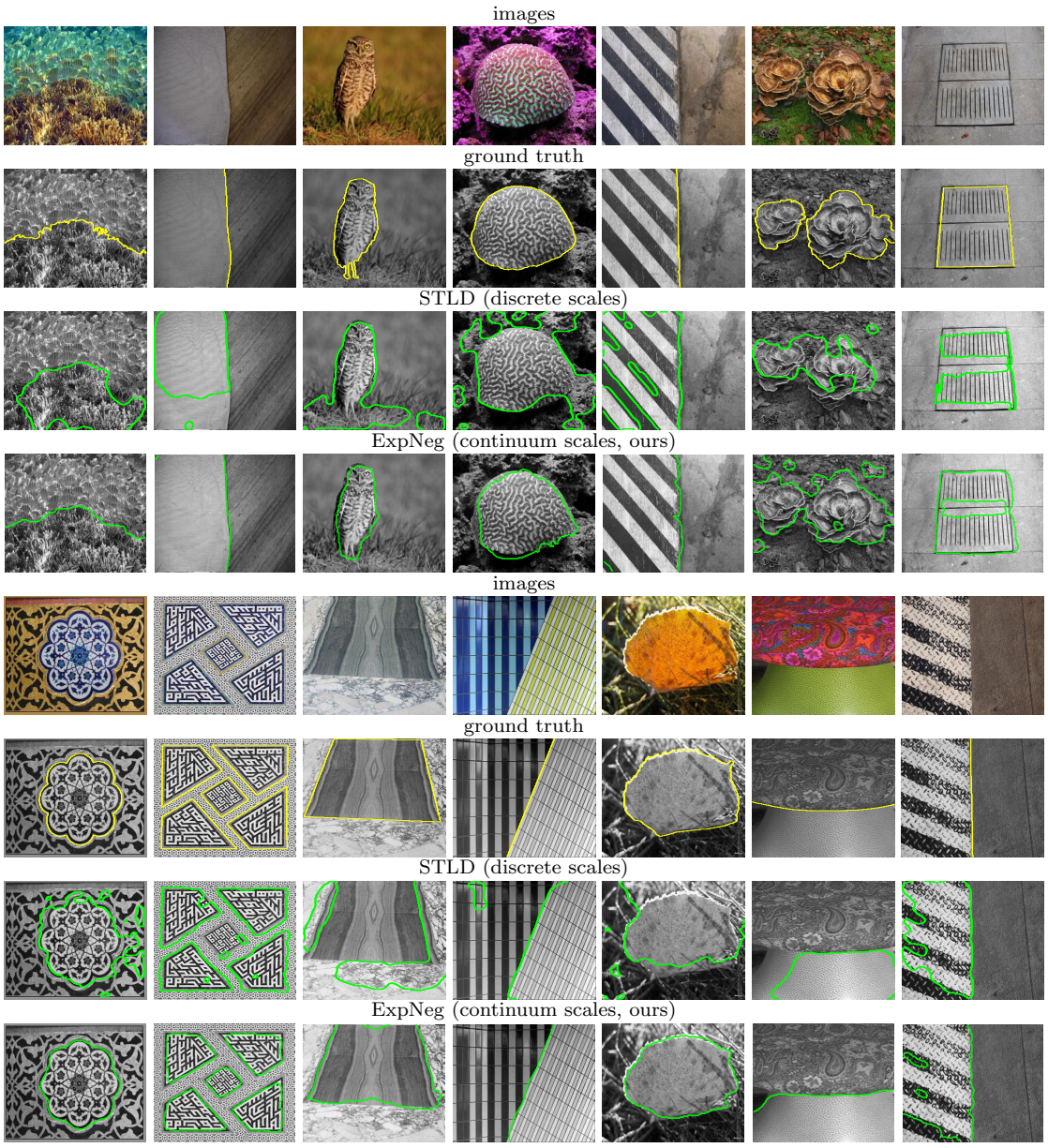

Figure 7.10: **Sample representative results on Real-World Texture Dataset.** We compare the best two methods (ours ctf) and STLD (using discrete scale spaces).

STLD Scale Comparison

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
STLD scales	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
4	0.56	0.56	0.85	0.85	0.85	0.85	0.63	0.63
20	0.55	0.55	0.84	0.84	0.84	0.84	0.64	0.64
4,8,12,16,20	**0.58**	**0.58**	**0.87**	**0.87**	**0.87**	**0.87**	**0.59**	**0.59**

ExpNeg Parameter α Comparison

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
$\alpha = 20$	0.60	0.60	0.91	0.91	0.91	0.91	0.45	0.45
$\alpha = 30$	0.60	0.60	0.90	0.90	0.90	0.90	0.46	0.46
$\alpha = 50$	0.60	0.60	0.90	0.90	0.90	0.90	0.46	0.46

Speed Comparison

method	average iterations	average time
ExpNeg ($\alpha = 20$)	12.9 ± 4.4	10.3 sec
STLD (scale 4,8,12,16,20)	16 ±4.1	83.7 sec

Table 7.4: **Analysis of Scale Parameters and Speed**. [Top]: Comparison of different scale choices for discrete scale spaces (STLD). [Middle]: Results for different α in continuum scale space with ExpNeg weight. [Bottom]: Speed comparison on a single processor for ExpNeg continuum scale space and STLD.

7.2.2 Motion Segmentation

Datasets: We test our method on the Freiburg-Berkeley Motion Segmentation (FBMS-59) [82] dataset. FBMS-59 consists of two sets - training, 29 sequences, and test, 30 sequences. Videos range between 19 and 800 frames, and have multiple objects.

Comparison: To demonstrate the advantage of our continuum space energy over a corresponding single scale energy, we compare to [114]. Our approach replaces the single scale motion term there with the energy (4.13). Further, additional regular-

	Training set (29 sequences)				Test set (30 sequences)			
	P	R	F	$N/65$	P	R	F	$N/69$
[111]	79.17	47.55	59.42	4	77.11	42.99	55.20	5
[82]	81.50	63.23	71.21	16	74.91	60.14	66.72	20
[112]	83.00	70.10	76.01	23	77.94	59.14	67.25	15
[113]	86.91	71.33	78.35	25	87.57	**70.19**	**77.92**	25
[114]	89.53	70.74	79.03	26	91.47	64.75	75.82	27
ExpNeg (ours)	**93.04**	**72.68**	**81.61**	**29**	**95.94**	65.54	77.87	**28**

Table 7.5: .

FBMS-59 results Average precision (P), recall (R), F-measure (F), and number of objects detected (N) over all sequences in training and test datasets. Higher values indicate superior performance. All methods are fully automatic.

Figure 7.11: **FBMS-59 Results.** Sample visual results on representative sequences for the FBMS-59 dataset (segmented objects in purple and red). The change of energy to integrate over all scales (our approach) is generally less sensitive to clutter than using an energy that contains only one scale (non-SS).

ization used in [114] is not used, as the scale-space provides inherent regularization. Since we test on benchmarks, we also compare to other state-of-the-art approaches, although our main purpose is to show the improvements that occur by merely using our continuum scale space energy.

Initialization: We initialize each with a segmentation of optical flow from [79] between frame 1 and 20.

Parameters: Our method with ExpNeg weighting requires one parameter α in (4.12). We choose it to be $\alpha = 20$ by selecting it based on a few sequences from the training set. Other parameters e.g., histogram sizes are chosen based on [114].

Results on FBMS-59: Figure 7.11 shows some representative visual results of our method and the single scale approach. Table 7.5 shows quantitative results of the two approaches, as well as other state-of-the-art methods. Visual results show our approach generally avoids distracting clutter and thus prevents leakages in comparison to the single scale approach. In many cases, it also captures more of the object. Quantitative results show that we improve the F-measure of [114] by about 2% on both training and test sets, and that we increase the number of objects detected. We also have highest F-measure of all competing methods.

Computational cost: The additional processing cost required for our scale space is small compared with the overall cost of [114]. Our approach adds about 5 secs per frame (one core) to the total time on average of about 30 secs per frame by [114] on a 12-core processor.

7.3 Learned Invariant Descriptors Experiments

Datasets: We use four different datasets to test our method. We use the Real World Texture Dataset and Brodatz Synthetic Dataset introduced in [26]. The first consists of 256 total real-world textured images collected from the internet with two dominant textures. 128 images are used for training and 128 for testing. The Brodatz

Synthetic Dataset consists of 200 images of two textured regions of various shapes. We also use the Graz Segmentation dataset [115], which consists of 243 images of real-world textured objects with multiple objects per image. Finally, we use the Berkeley Segmentation Dataset, which consists of 200 training and test images, and 100 validation images, and various numbers of textured objects in each image. Each of the datasets exhibit complex nuisances, such as illumination, shading, perspective effects, etc.

Architecture Details: We use a Siamese twin network, where each component has two fully connected layers. We test the sensitivity number of hidden layers and hidden units later. Our input base shape-tailored descriptor is a 40 dimensional descriptor (RGB channels, gray scale and four oriented gradients at 5 scales, $\alpha = (10, 20, 30, 40, 50)$). The output descriptor f of the Siamese network is same size as the number of hidden units used. The sigmoid of the (learned) weighted difference of the two twins is used to compute the metric D of a pair of descriptors.

Results on Real-World Texture Dataset: We use 128 images in the training set to train our network and test on the 128 images in the test set. This gives us 9153732 training pairs of descriptors. We initialize our method by a 5×5 standard block tessellation, with random labels (out of 1 or 2) chosen for each block. Quantitative results are in Table 7.6. We compare to hand-crafted Shape-Tailored Descriptors (STLD) [26], to non-STLD (the descriptors in (5.1) when $R = \Omega$ the whole image), learned non-STLD (non-STLD base descriptors used to learn invariant descriptors through the Siamese network), and other methods. non-STLD handcrafted performs the worst, followed by learned non-STLD, then hand-crafted STLD performs better, and the learned STLD (our approach) performs the best. This shows that both properties of shape-tailored and learned are necessary to achieve the best results. Figure 7.12 shows some visual comparisons of our approach to handcrafted STLD.

Robustness to Initialization, Training Data, and Architecture: First we

test sensitivity to initialization. We vary the box tessellation from 3×3 to 5×5. Results are in Table 7.7. They show that the method is robust to initialization. Now we test the sensitivity to the architecture in our approach and the learned non-STLD approach. To this end, we vary the architecture of our network by changing the number of hidden units. Results are shown in Table 7.8. With two layers, the performance is mostly stable as the number of hidden layers are changed. We also show results with 3 and 4 layers with 41 hidden units. Performance degrades somewhat, and we believe this to be an overfit. We now test sensitivity to the number of training images, results are in Table 7.9. The results do not deteriorate much as we vary the number of images.

Results on Synthetic Texture Dataset: We test our method on synthetic Brodatz with the previous network. Table 7.10 shows results, and our method performs the best by a wide margin.

Results on Graz and BSDS 500 Dataset: We now test our method against hand-crafted STLD on Graz and BSD500. These experiments provide more verification that the network is learning a generic property for segmentation over STLD. We initialize methods with a Voronoi partition of the seed points provided in Graz. For BSD500, we initialize the segmentation by randomly perturbing the ground truth segmentation by 30 pixels. Table 7.11 shows the results. It shows that the learned STLD better capture properties of textures than the hand-crafted STLD.

Real-World Texture Dataset

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
Learned [108]	**0.62**	**0.62**	**0.91**	**0.91**	**0.91**	**0.91**	**0.44**	**0.44**
Learned(non-STLD)	0.53	0.53	0.89	0.89	0.89	0.89	0.47	0.47
STLD	0.58	0.58	0.86	0.86	0.88	0.88	0.63	0.63
non-STLD	0.20	0.20	0.83	0.83	0.84	0.84	0.79	0.79
mcg [23]	0.51	0.54	0.74	0.82	0.77	0.85	0.80	0.66
gPb [43]	0.53	0.57	0.81	0.84	0.82	0.85	0.82	0.78
CB [36]	0.54	0.56	0.75	0.80	0.79	0.84	0.81	0.76
SIFT	0.13	0.13	0.54	0.54	0.58	0.58	1.50	1.50
Entropy [44]	0.19	0.19	0.74	0.74	0.76	0.76	1.00	1.00
Hist-5 [38]	0.17	0.17	0.67	0.67	0.72	0.72	1.25	1.25
Hist-10 [38]	0.16	0.16	0.67	0.67	0.72	0.72	1.26	1.26
Chan-Vese [49]	0.19	0.19	0.73	0.73	0.76	0.76	1.07	1.07
LAC [52]	0.14	0.14	0.54	0.54	0.58	0.58	1.51	1.51
Global Hist [32]	0.14	0.14	0.66	0.66	0.68	0.68	1.16	1.16

Table 7.6: **Results on Texture Segmentation Datasets.** Algorithms are evaluated using contour and region metrics. Higher F-measure for the contour metric, ground truth covering (GT-cov), and rand index indicate better fit to the ground truth, and lower variation of information (Var. Info) indicates a better fit to ground truth.

Initialization

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
5by5	0.62	0.62	0.91	0.91	0.91	0.91	0.44	0.44
4by4	0.61	0.61	0.91	0.91	0.91	0.91	0.44	0.44
3by3	0.61	0.61	0.91	0.91	0.91	0.91	0.44	0.44

Table 7.7: **Insensitivity to Initialization.** The results remain similar as we vary the box tessellation initialization for segmentation.

Network Architecture

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
61 units	0.60	0.60	0.91	0.91	0.90	0.90	0.45	0.45
51 units	0.61	0.61	0.91	0.91	0.91	0.91	0.45	0.45
41 units	0.62	0.62	0.91	0.91	0.91	0.91	0.44	0.44
31 units	0.60	0.60	0.91	0.91	0.91	0.91	0.45	0.45
21 units	0.57	0.57	0.90	0.90	0.90	0.90	0.48	0.48
3 Layers	0.55	0.55	0.89	0.89	0.89	0.89	0.48	0.48
4 Layers	0.54	0.54	0.88	0.88	0.88	0.88	0.52	0.52
61 units	0.52	0.52	0.89	0.89	0.89	0.89	0.48	0.48
51 units	0.51	0.51	0.88	0.88	0.88	0.88	0.48	0.48
41 units	0.53	0.53	0.89	0.89	0.89	0.89	0.47	0.47
31 units	0.49	0.49	0.88	0.88	0.88	0.88	0.49	0.49
21 units	0.46	0.46	0.87	0.87	0.87	0.87	0.51	0.51
3 Layers	0.54	0.54	0.87	0.87	0.87	0.87	0.55	0.55
4 Layers	0.53	0.53	0.87	0.87	0.87	0.87	0.58	0.58

Table 7.8: **Performance vs. Architecture.** The top half of the table shows the performance for learned STLD descriptor (ours) and the bottom part show the results for learned non-STLD descriptor. We have varied the number of hidden units in the two-layer network, and the number of layers from 3-4 with 41 units.

Training Images

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
128 images	0.62	0.62	0.91	0.91	0.91	0.91	0.44	0.44
100 images	0.59	0.59	0.90	0.90	0.90	0.90	0.47	0.47
75 images	0.58	0.58	0.90	0.90	0.90	0.90	0.49	0.49
50 images	0.54	0.54	0.89	0.89	0.89	0.89	0.52	0.52

Table 7.9: **Varying Number of Images and data in Training.** We vary the number of training images and report the results. We vary the number of dilation of the ground truth and report the effect on performance, higher number of dilations means more data per image.

Synthetic Dataset

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
Learned (ours)	**0.45**	**0.45**	**0.90**	**0.90**	**0.89**	**0.89**	**0.46**	**0.46**
STLD	0.41	0.41	0.87	0.87	0.86	0.86	0.53	0.53
non-STLD	0.18	0.18	0.84	0.84	0.84	0.84	0.65	0.65
gPb [43]	0.40	0.38	0.79	0.81	0.79	0.82	0.75	0.73
CB [36]	0.30	0.29	0.75	0.77	0.76	0.79	1.09	1.08
SIFT	0.11	0.11	0.70	0.70	0.70	0.70	1.07	1.07
Entropy [44]	0.13	0.13	0.75	0.75	0.75	0.75	0.91	0.91
Hist-5 [38]	0.32	0.32	0.67	0.67	0.68	0.68	1.10	1.10
Hist-10 [38]	0.32	0.32	0.65	0.65	0.67	0.67	1.15	1.15
Chan-Vese [49]	0.19	0.19	0.72	0.72	0.72	0.72	0.95	0.95
LAC [52]	0.14	0.14	0.72	0.72	0.70	0.70	1.14	1.14
Global Hist [32]	0.28	0.28	0.75	0.75	0.75	0.75	0.79	0.79

Table 7.10: **Results on Synthetic Texture Segmentation Dataset.** See Table 3 caption for details on the measures (higher is better except for Var. of Info.)

Graz Dataset

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
Learned STLD	0.42	0.42	0.76	0.76	0.82	0.82	1.02	1.02
STLD	0.34	0.34	0.70	0.70	0.77	0.77	1.21	1.21

BSD Dataset

	Contour		Region metrics					
	F-meas.		GT-cov.		Rand. Index		Var. Info.	
	ODS	OIS	ODS	OIS	ODS	OIS	ODS	OIS
Learned STLD	0.66	0.66	0.62	0.62	0.82	0.82	1.75	1.75
STLD	0.56	0.56	0.57	0.57	0.79	0.79	1.99	1.99

Table 7.11: **Graz and BSDS 500 Dataset Results.** See Table 3 caption for details on the measures (higher is better except for Var. of Info.

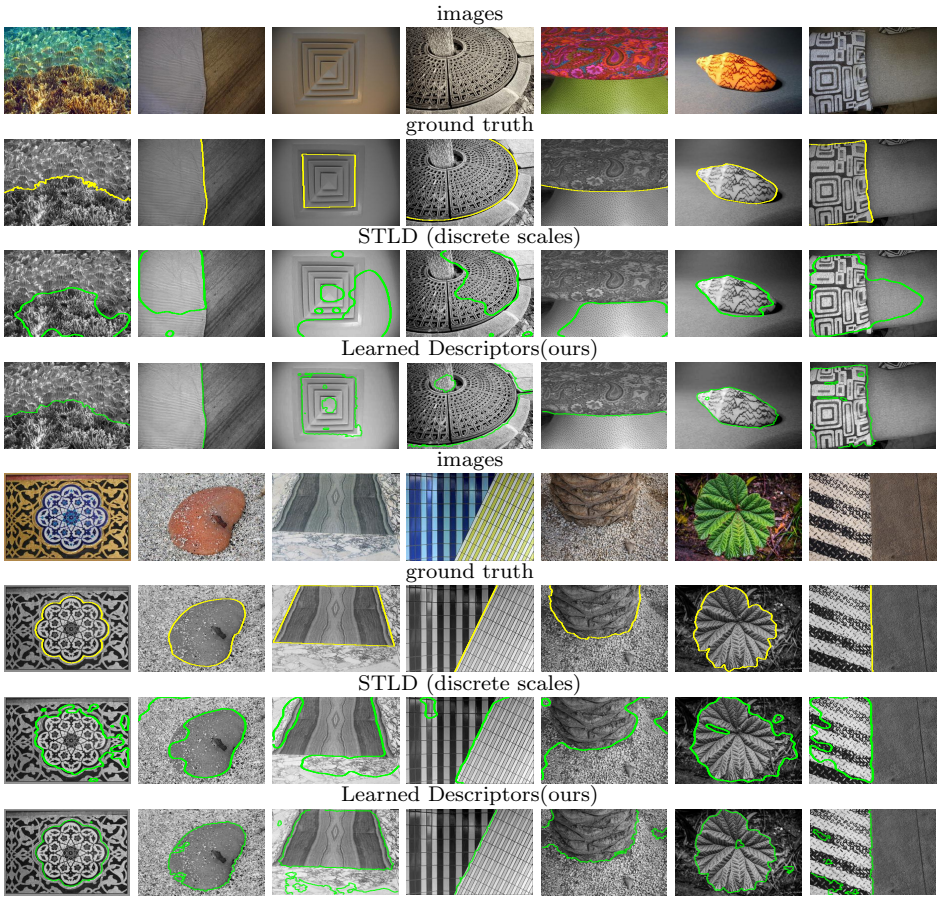

Figure 7.12: **Sample representative results on Real-World Texture Dataset.**We compare the best two methods (ours) and STLD (using discrete scale spaces).

Chapter 8

Conclusion

This work presents the fundamentals concepts of constructing descriptors for vision applications, particularly segmentation. We start with constructing descriptors on region of interest, as most descriptors in vision are built on some receptive field around a pixel. Traditional descriptors assume these receptive fields to be of fixed shape and size on image domain. There is a fundamental flaw in this assumption as aggregating of statistics should be tailored to region of interest only, and should not aggregate information from background. To tackle this problem we have introduced Shape-Tailored Local Descriptors, where we aggregated information only on a region of interest.

Next, we provide tools for finding the correct segmentation based on shape-tailored local descriptors. Notice, that with shape-tailored descriptors we have tools to aggregate information from region of interest but the region of interest in segmentation is not known a priori. To tackle this issue we provide tools for jointly calculating the shape-tailored descriptor and region of interest. We start with an initialization for segmentation, based on this initialization we construct shape-tailored descriptors. Once shape-tailored descriptors are calculated the segmentation is updated based on the descriptors. This update of descriptors and region is repeated until convergence of region and descriptors.

Once the region of interest is known the next important question to tackle is that of how the descriptors should be constructed? How can we aggregate information on region of interest in a way that the descriptors capture useful information on the region

of interest. We started with aggregating oriented gradients on these region of interest, this performs well in most cases but is not completely invariant to complex nuisances of textured regions. Ideally we would want the descriptors to be completely invariant to intrinsic and extrinsic nuisances of textured regions, i.e. illumination changes, scale changes etc. To tackle the problem more carefully we have employed Neural Networks. We use Siamese twin network to learn invariant shape-tailored descriptors and a difference metric which can discriminate between different textures, while being invariant to nuisances of textured regions.

We have also tackled the problem of training of Neural Network with minimal or no training data, as one of the key bottleneck in learning in computer vision is the prohibitive size of data required. We have introduced unsupervised learning techniques where we are learning dense descriptors without any training data. Significant features in the data are learned on the go as we look for segmentation into unique regions.

Lastly, there are some side issues we have also tackled in this work, particularly coarse-to-fine segmentation which is of particular interest in multi-scale segmentation algorithms. We show that a continuum Gaussian scale space has a natural coarse-to-fine property in segmentation. We also provide an energy based on Gaussian scale space that incorporates the coarse to fine segmentation without the explicit need for calculating all scales. We only need to solve a Poisson equation at native scale to calculate the gradient flow of the energy.

One of the key issues of segmentation not tackled in this thesis is that of hierarchy of segmentation based on region descriptors. The number of regions in an image is not known a priori, and a good segmentation algorithms must be capable of handling different regions in different images. The most promising approach to tackle multiple regions is that of constructing a hierarchy of segmentation, where different layers of hierarchy represent segmentation at different scales. There has been good work in

hierarchy based segmentation on edge-based methods, where the strength of the edge corresponds to the level of hierarchy. The drawback of these edge-based approaches is that a stronger edge does not always correspond to more salient region, particularly in the case of texture segmentation. To the best my knowledge, there has not been any promising attempt at constructing segmentation hierarchy based on region statistics/ descriptors. This is one of the open questions that needs to be addressed in the future.

REFERENCES

[1] S. Xie and Z. Tu, "Holistically-Nested Edge Detection," *ArXiv e-prints*, apr 2015.

[2] I. Kokkinos, "Surpassing humans in boundary detection using deep learning," *CoRR*, vol. abs/1511.07386, 2015. [Online]. Available: http://arxiv.org/abs/1511.07386

[3] P. H. Pinheiro, "Large-scale image segmentation with convolutional networks," 2017.

[4] D. Marr, "Vision: A computational investigation into the human representation and processing of visual information, henry holt and co," *Inc., New York, NY*, p. 2, 1982.

[5] J. Malik and P. Perona, "Preattentive texture discrimination with early vision mechanisms," *JOSA A*, vol. 7, no. 5, pp. 923–932, 1990.

[6] S. C. Zhu, Y. Wu, and D. Mumford, "Filters, random fields and maximum entropy (frame): Towards a unified theory for texture modeling," *International Journal of Computer Vision*, vol. 27, no. 2, pp. 107–126, 1998.

[7] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 2. IEEE, 1999, pp. 1033–1038.

[8] L.-Y. Wei, S. Lefebvre, V. Kwatra, G. Turk *et al.*, "State of the art in example-based texture synthesis," in *Eurographics 2009, State of the Art Report, EG-STAR*, 2009, pp. 93–117.

[9] B. Julesz, "Textons, the elements of texture perception, and their interactions," *Nature*, vol. 290, no. 5802, pp. 91–97, 1981.

[10] O. Pujol and P. Radeva, "Texture segmentation by statistical deformable models," *International Journal of Image and Graphics*, vol. 04, no. 03, pp. 433–452, 2004.

[11] M.-P. Dubuisson-Jolly and A. Gupta, "Color and texture fusion: application to aerial image segmentation and gis updating," *Image and Vision Computing*, vol. 18, no. 10, pp. 823 – 832, 2000.

[12] M. Tuceryan and A. K. Jain, *Handbook of Pattern Recognition and Computer Vision, Texture Analysis*, ch. 10, pp. 235–276.

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[14] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[15] E. Tola, V. Lepetit, and P. Fua, "Daisy: An efficient dense descriptor applied to wide-baseline stereo," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 5, pp. 815–830, 2010.

[16] L. Sifre and S. Mallat, "Rotation, scaling and deformation invariant scattering for texture discrimination," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 1233–1240.

[17] J. J. Koenderink, "The structure of images," *Biological cybernetics*, vol. 50, no. 5, pp. 363–370, 1984.

[18] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 12, no. 7, pp. 629–639, 1990.

[19] J. Hegdé, "Time course of visual perception: coarse-to-fine processing and beyond," *Progress in neurobiology*, vol. 84, no. 4, pp. 405–439, 2008.

[20] P. Neri, "Coarse to fine dynamics of monocular and binocular processing in human pattern vision," *Proceedings of the National Academy of Sciences*, vol. 108, no. 26, pp. 10 726–10 731, 2011.

[21] X. Bresson, P. Vandergheynst, and J.-P. Thiran, "Multiscale active contours," *International Journal of Computer Vision*, vol. 70, no. 3, pp. 197–211, 2006.

[22] I. Kokkinos, G. Evangelopoulos, and P. Maragos, "Texture analysis and segmentation using modulation features, generative models, and weighted curve evolution," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 1, pp. 142–157, 2009.

[23] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 328–335.

[24] A. Blake and A. Zisserman, *Visual reconstruction*. MIT press Cambridge, 1987, vol. 2.

[25] H. Mobahi and J. W. Fisher III, "Coarse-to-fine minimization of some common nonconvexities," in *Energy Minimization Methods in Computer Vision and Pattern Recognition*, 2015, pp. 71–84.

[26] N. Khan, M. Algarni, A. Yezzi, and G. Sundaramoorthi, "Shape-tailored local descriptors and their application to segmentation and tracking," in *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 2015, pp. 3890–3899.

[27] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, June 2005, pp. 539–546 vol. 1.

[28] S. C. 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*, vol. 18, no. 9, pp. 884–900, 1996.

[29] N. Paragios and R. Deriche, "Geodesic active regions and level set methods for supervised texture segmentation," *International Journal of Computer Vision*, vol. 46, no. 3, pp. 223–247, 2002.

[30] J. Kim, J. W. Fisher, A. Yezzi, M. Çetin, and A. S. Willsky, "A nonparametric statistical method for image segmentation using information theory and curve evolution," *Image Processing, IEEE Transactions on*, vol. 14, no. 10, pp. 1486–1502, 2005.

[31] D. Cremers, M. Rousson, and R. Deriche, "A review of statistical approaches to level set segmentation: integrating color, texture, motion and shape," *International journal of computer vision*, vol. 72, no. 2, pp. 195–215, 2007.

[32] O. Michailovich, Y. Rathi, and A. Tannenbaum, "Image segmentation using active contours driven by the bhattacharyya gradient flow," *Image Processing, IEEE Transactions on*, vol. 16, no. 11, pp. 2787–2801, 2007.

[33] N. Houhou, J. 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.* IEEE, 2008, pp. 1–8.

[34] G. Peyré, J. Fadili, and J. Rabin, "Wasserstein active contours," in *Image Processing (ICIP), 2012 19th IEEE International Conference on.* IEEE, 2012, pp. 2541–2544.

[35] S. P. Awate, T. Tasdizen, and R. T. Whitaker, "Unsupervised texture segmentation with nonparametric neighborhood statistics," in *Computer Vision–ECCV 2006.* Springer, 2006, pp. 494–507.

[36] P. Isola, D. Zoran, D. Krishnan, and E. H. Adelson, "Crisp boundary detection using pointwise mutual information," in *Computer Vision–ECCV 2014.* Springer, 2014, pp. 799–814.

[37] S. Todorovic and N. Ahuja, "Texel-based texture segmentation," in *Computer Vision, 2009 IEEE 12th International Conference on.* IEEE, 2009, pp. 841–848.

[38] K. Ni, X. Bresson, T. Chan, and S. Esedoglu, "Local histogram based segmentation using the wasserstein distance," *International Journal of Computer Vision*, vol. 84, no. 1, pp. 97–111, 2009.

[39] M. Rousson, T. Brox, and R. Deriche, "Active unsupervised texture segmentation on a diffusion based feature space," in *Computer vision and pattern recognition, 2003. Proceedings. 2003 IEEE computer society conference on*, vol. 2. IEEE, 2003, pp. II–699.

[40] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry, "Unsupervised segmentation of natural images via lossy data compression," *Computer Vision and Image Understanding*, vol. 110, no. 2, pp. 212–225, 2008.

[41] T. S. Lee, D. Mumford, and A. Yuille, "Texture segmentation by minimizing vector-valued energy functionals: The coupled-membrane model," in *Computer Vision—ECCV'92.* Springer, 1992, pp. 165–173.

[42] C. Sagiv, N. A. Sochen, and Y. Y. Zeevi, "Integrated active contours for texture segmentation," *Image Processing, IEEE Transactions on*, vol. 15, no. 6, pp. 1633–1646, 2006.

[43] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 5, pp. 898–916, 2011.

[44] B.-W. Hong, S. Soatto, K. Ni, and T. Chan, "The scale of a texture and its application to segmentation," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[45] S. Boltz, F. Nielsen, and S. Soatto, "Texture regimes for entropy-based multiscale image analysis," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 692–705.

[46] B.-W. Hong, K. Ni, and S. Soatto, "Entropy-scale profiles for texture segmentation," in *Scale Space and Variational Methods in Computer Vision*. Springer, 2012, pp. 243–254.

[47] M. Galun, E. Sharon, R. Basri, and A. Brandt, "Texture segmentation by multiscale aggregation of filter responses and shape elements," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. IEEE, 2003, pp. 716–723.

[48] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International journal of computer vision*, vol. 1, no. 4, pp. 321–331, 1988.

[49] T. F. Chan and L. A. Vese, "Active contours without edges," *Image processing, IEEE transactions on*, vol. 10, no. 2, pp. 266–277, 2001.

[50] A. Yezzi Jr, A. Tsai, and A. Willsky, "A statistical approach to snakes for bimodal and trimodal imagery," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 2. IEEE, 1999, pp. 898–903.

[51] D. Mumford and J. Shah, "Optimal approximations by piecewise smooth functions and associated variational problems," *Communications on pure and applied mathematics*, vol. 42, no. 5, pp. 577–685, 1989.

[52] S. Lankton and A. Tannenbaum, "Localizing region-based active contours," *Image Processing, IEEE Transactions on*, vol. 17, no. 11, pp. 2029–2039, 2008.

[53] C. Darolti, A. Mertins, C. Bodensteiner, and U. G. Hofmann, "Local region descriptors for active contours evolution," *Image Processing, IEEE Transactions on*, vol. 17, no. 12, pp. 2275–2288, 2008.

[54] T. Brox and D. Cremers, "On local region models and a statistical interpretation of the piecewise smooth mumford-shah functional," *International journal of computer vision*, vol. 84, no. 2, pp. 184–193, 2009.

[55] T. F. Chan, S. Esedoglu, and M. Nikolova, "Algorithms for finding global minimizers of image segmentation and denoising models," *SIAM journal on applied mathematics*, vol. 66, no. 5, pp. 1632–1648, 2006.

[56] X. Bresson, S. Esedoḡlu, P. Vandergheynst, J.-P. Thiran, and S. Osher, "Fast global minimization of the active contour/snake model," *Journal of Mathematical Imaging and vision*, vol. 28, no. 2, pp. 151–167, 2007.

[57] T. Pock, D. Cremers, H. Bischof, and A. Chambolle, "An algorithm for minimizing the mumford-shah functional," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 1133–1140.

[58] M. C. Delfour and J.-P. Zolésio, *Shapes and geometries: metrics, analysis, differential calculus, and optimization*. Siam, 2011, vol. 22.

[59] G. Aubert, M. Barlaud, O. Faugeras, and S. Jehan-Besson, "Image segmentation using active contours: Calculus of variations or shape gradients?" *SIAM Journal on Applied Mathematics*, vol. 63, no. 6, pp. 2128–2154, 2003.

[60] A. Herbulot, S. Jehan-Besson, S. Duffner, M. Barlaud, and G. Aubert, "Segmentation of vectorial image features using shape gradients and information measures," *Journal of Mathematical Imaging and Vision*, vol. 25, no. 3, pp. 365–386, 2006.

[61] A. P. Witkin, "Scale-space filtering: A new approach to multi-scale description," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'84.*, vol. 9. IEEE, 1984, pp. 150–153.

[62] J.-M. Geusebroek, R. Van Den Boomgaard, A. W. Smeulders, and A. Dev, "Color and scale: The spatial structure of color images," in *Computer Vision-ECCV 2000*. Springer, 2000, pp. 331–341.

[63] G. Koutaki and K. Uchimura, "Scale-space processing using polynomial representations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2744–2751.

[64] T. Lindeberg, "Scale-space for discrete signals," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 12, no. 3, pp. 234–254, 1990.

[65] L. Florack and A. Kuijper, "The topological structure of scale-space images," *Journal of Mathematical Imaging and Vision*, vol. 12, no. 1, pp. 65–79, 2000.

[66] R. Van Den Boomgaard and A. Smeulders, "The morphological structure of images: The differential equations of morphological scale-space," *Pattern Analysis*

and Machine Intelligence, IEEE Transactions on, vol. 16, no. 11, pp. 1101–1113, 1994.

[67] A. Sironi, V. Lepetit, and P. Fua, "Multiscale centerline detection by learning a scale-space distance transform," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014, pp. 2697–2704.

[68] B. D. Lucas, T. Kanade *et al.*, "An iterative image registration technique with an application to stereo vision." in *IJCAI*, vol. 81, 1981, pp. 674–679.

[69] R. Hummel and R. Moniot, "Reconstructions from zero crossings in scale space," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 37, no. 12, pp. 2111–2130, 1989.

[70] B. Ummenhofer and T. Brox, "Global, dense multiscale reconstruction for a billion points," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1341–1349.

[71] T. Hassner, V. Mayzels, and L. Zelnik-Manor, "On sifts and their scales," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1522–1528.

[72] M. M. Bronstein and I. Kokkinos, "Scale-invariant heat kernel signatures for non-rigid shape recognition," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 1704–1711.

[73] G. Sapiro and A. Tannenbaum, "Affine invariant scale-space," *International journal of computer vision*, vol. 11, no. 1, pp. 25–44, 1993.

[74] J. C. Rubio, J. Serrat, A. López, and N. Paragios, "Unsupervised co-segmentation through region matching," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 749–756.

[75] J.-F. Aujol, G. Gilboa, T. Chan, and S. Osher, "Structure-texture image decomposition—modeling, algorithms, and parameter selection," *International Journal of Computer Vision*, vol. 67, no. 1, pp. 111–136, 2006.

[76] S. Osher and J. A. Sethian, "Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations," *Journal of computational physics*, vol. 79, no. 1, pp. 12–49, 1988.

[77] N. Komodakis, M. P. Kumar, and N. Paragios, "(hyper)-graphs inference through convex relaxations and move making algorithms: Contributions and applications in artificial vision," *Foundations and Trends® in Computer Graphics and Vision*, vol. 10, no. 1, pp. 1–102, 2016.

[78] G. Sundaramoorthi and B.-W. Hong, "Fast label: Easy and efficient solution of joint multi-label and estimation problems," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3126–3133.

[79] D. Sun, S. Roth, and M. J. Black, "Secrets of optical flow estimation and their principles," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2432–2439.

[80] J. Y. Wang and E. H. Adelson, "Representing moving images with layers," *Image Processing, IEEE Transactions on*, vol. 3, no. 5, pp. 625–638, 1994.

[81] D. Cremers and S. Soatto, "Motion competition: A variational approach to piecewise parametric motion segmentation," *International Journal of Computer Vision*, vol. 62, no. 3, pp. 249–265, 2005.

[82] P. Ochs, J. Malik, and T. Brox, "Segmentation of moving objects by long term video analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 6, pp. 1187–1200, 2014.

[83] D. Sun, J. Wulff, E. Sudderth, H. Pfister, and M. Black, "A fully-connected layered model of foreground and background flow," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2451–2458.

[84] Y. Yang and G. Sundaramoorthi, "Shape tracking with occlusions via coarse-to-fine region-based sobolev descent," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 5, pp. 1053–1066, 2015.

[85] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang, "Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[86] A. Arnab, S. Jayasumana, S. Zheng, and P. H. S. Torr, "Higher order potentials in end-to-end trainable conditional random fields," *CoRR*, vol. abs/1511.08119, 2015. [Online]. Available: http://arxiv.org/abs/1511.08119

[87] V. Badrinarayanan, A. Handa, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling," *CoRR*, vol. abs/1505.07293, 2015. [Online]. Available: http://arxiv.org/abs/1505.07293

[88] F. Liu, G. Lin, and C. Shen, "CRF learning with CNN features for image segmentation," *CoRR*, vol. abs/1503.08263, 2015. [Online]. Available: http://arxiv.org/abs/1503.08263

[89] J. Yang, B. Price, S. Cohen, H. Lee, and M. Yang, "Object Contour Detection with a Fully Convolutional Encoder-Decoder Network," *ArXiv e-prints*, mar 2016.

[90] G. Bertasius, J. Shi, and L. Torresani, "Deepedge: A multi-scale bifurcated deep network for top-down contour detection," *CoRR*, vol. abs/1412.1123, 2014. [Online]. Available: http://arxiv.org/abs/1412.1123

[91] M. Lai, "Deep learning for medical image segmentation," *CoRR*, vol. abs/1505.02000, 2015. [Online]. Available: http://arxiv.org/abs/1505.02000

[92] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr, "Conditional random fields as recurrent neural networks," *CoRR*, vol. abs/1502.03240, 2015. [Online]. Available: http://arxiv.org/abs/1502.03240

[93] A. Kendall, V. Badrinarayanan, and R. Cipolla, "Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," *CoRR*, vol. abs/1511.02680, 2015. [Online]. Available: http://arxiv.org/abs/1511.02680

[94] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *CoRR*, vol. abs/1511.00561, 2015. [Online]. Available: http://arxiv.org/abs/1511.00561

[95] P. H. O. Pinheiro and R. Collobert, "Recurrent convolutional neural networks for scene parsing," *CoRR*, vol. abs/1306.2795, 2013. [Online]. Available: http://arxiv.org/abs/1306.2795

[96] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1915–1929, 2013.

[97] N. Khan, M. Algarni, A. Yezzi, and G. Sundaramoorthi, "Shape-tailored local descriptors and their application to segmentation and tracking," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 3890–3899.

[98] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[99] Y. Zhu, Y. Tian, D. Mexatas, and P. Dollár, "Semantic amodal segmentation," *CoRR*, vol. abs/1509.01329, 2015. [Online]. Available: http://arxiv.org/abs/1509.01329

[100] N. Khan, "Shape-tailored features and their application to texture segmentation," 2014. [Online]. Available: http://hdl.handle.net/10754/317258

[101] T. Lindeberg, *Scale-space theory in computer vision*. Springer, 1993.

[102] L. C. Evans, "Partial differential equations," 1998.

[103] L. A. Vese and T. F. Chan, "A multiphase level set framework for image segmentation using the mumford and shah model," *International journal of computer vision*, vol. 50, no. 3, pp. 271–293, 2002.

[104] A. Tsai, A. Yezzi, and A. S. Willsky, "Curve evolution implementation of the mumford-shah functional for image segmentation, denoising, interpolation, and magnification," *IEEE transactions on Image Processing*, vol. 10, no. 8, pp. 1169–1186, 2001.

[105] N. Khan, B.-W. Hong, A. Yezzi, and G. Sundaramoorthi, "Coarse-to-fine segmentation with shape-tailored continuum scale spaces," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[106] C. Chen and H. Edelsbrunner, "Diffusion runs low on persistence fast," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 423–430.

[107] L. C. Evans, "Partial differential equations," 2010.

[108] N. Khan and G. Sundaramoorthi, "Learned shape-tailored descriptors for segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[109] A. Yezzi, A. Tsai, and A. Willsky, "A fully global approach to image segmentation via coupled curve evolution equations," *Journal of Visual Communication and Image Representation*, vol. 13, no. 1, pp. 195 – 216, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1047320301905000

[110] Y. Yang and G. Sundaramoorthi, "Modeling self-occlusions in dynamic shape and appearance tracking," in *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 2013, pp. 201–208.

[111] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Efficient hierarchical graph-based video segmentation," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2141–2148.

[112] B. Taylor, V. Karasev, and S. Soatto, "Causal video object segmentation from persistence of occlusions," in *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 2015, pp. 4268–4276.

[113] M. Keuper, B. Andres, and T. Brox, "Motion trajectory segmentation via minimum cost multicuts," *IEEE International Conference on Computer Vision (ICCV)*, pp. 3271–3279.

[114] Y. Yang, G. Sundaramoorthi, and S. Soatto, "Self-occlusions and disocclusions in causal video object segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4408–4416.

[115] J. Santner, T. Pock, and H. Bischof, "Interactive multi-label segmentation," in *Proceedings 10th Asian Conference on Computer Vision (ACCV), Queenstown, New Zealand*, November 2010.

A Papers Submitted and Under Preparation

• Naeemullah Khan, Marei Algarni, Anthony Yezzi, Ganesh Sundaramoorthi, "Shape-tailored local descriptors and their application to segmentation and tracking", *IEEE Conference on Computer Vision and Pattern Recognition*, 2015 (Accepted)

• Naeemullah Khan, Byung-Woo Hong, Anthony Yezzi, Ganesh Sundaramoorthi, "Coarse-to-Fine Segmentation With Shape-Tailored Continuum Scale Spaces", *IEEE Conference on Computer Vision and Pattern Recognition*, 2017 (Accepted)

• Naeemullah Khan, Ganesh Sundaramoorthi, "Learned Shape-Tailored Descriptors for Segmentation", *IEEE Conference on Computer Vision and Pattern Recognition* , 2018 (Accepted)

• Naeemullah Khan, Ganesh Sundaramoorthi, "Unsupervised Learned Shape-Tailored Descriptors for Segmentation" (Under Preparation)

B Shape-Tailored Descriptors based Energy and it's Gradient

We detail the calculations that were left out in Section 2.2 of the manuscript. We repeat some definitions for the convenience of the reader. Note the Lemma and Proposition numbers correspond to the numbers in the manuscript, however, equation numbers do not correspond.

Definition 3 (Shape-Tailored Local Descriptors)**.** *Let $R \subset \mathbb{R}^2$ be a bounded region with non-zero area and smooth boundary ∂R. Let $I : R \to \mathbb{R}^k$. A* **Shape-Tailored Descriptor***, $\mathbf{u} : R \to \mathbb{R}^M$ (where $M = n \times m$, $n, m \geq 1$) consists of components $u_{ij} : R \to \mathbb{R}$ so that $\mathbf{u} = (u_{11}, \ldots, u_{1m}, \ldots, u_{n1}, \ldots, u_{nm})^T$. The components are defined as:*

$$\begin{cases} u_{ij}(x) - \alpha_i \Delta u_{ij}(x) = J_j(x) & x \in R \\ \nabla u_{ij}(x) \cdot N = 0 & x \in \partial R \end{cases}, \tag{B.1}$$

where $1 \leq i \leq n$, $1 \leq j \leq m$, Δ denotes the Laplacian, ∇ denotes the gradient, N is the unit outward normal to R, $\alpha_i > 0$ are scales*, and $J_j : R \to \mathbb{R}$ are point-wise functions of the image I. In vector form, this is equivalent to*

$$\begin{cases} \mathbf{u}(x) - A\Delta\mathbf{u}(x) = \mathbf{J}(x) & x \in R \\ D\mathbf{u}(x)N = \mathbf{0} & x \in \partial R \end{cases}, \tag{B.2}$$

where $A = diag(\alpha_1 1_{1\times m}, \ldots, \alpha_n 1_{1\times m})$ (an $M \times M$ diagonal matrix), $1_{1\times m}$ is a $1 \times m$ matrix of ones, D denotes the spatial derivative operator, and $\mathbf{J} = (J_1, \ldots, J_m, \ldots, J_1, \ldots, J_m, \ldots)^T$.

Lemma 2.0.1 (PDE for Descriptor Variation)**.** *Let u satisfy the PDE* (B.1)*, h be a*

perturbation of ∂R, and u_h denote the variation of u with respect to the perturbation h. Then

$$\begin{cases} u_h(x) - \alpha_i \Delta u_h(x) = 0 & x \in R \\ \nabla u_h(x) \cdot N = u_s(x)(h_s \cdot N) - N^T H u(x) \cdot h & x \in \partial R \end{cases} \tag{B.3}$$

where s is the arc-length parameter of ∂R, h_s denotes the derivative with respect to arc-length, and $Hu(x)$ denotes the Hessian matrix.

Proof. The PDE for u_h is obtained by computing the variation with respect to h of both conditions of the PDE (B.1). The variation of the first equation in (B.1) leads to the first equation in (B.3). This is because the variation and spatial derivatives commute by equality of mixed partials as the variation and spatial derivative operators are independent. Next we compute the variation of the boundary condition using the Chain Rule:

$$\mathrm{d}[\nabla u(c(p)) \cdot N] \cdot h = N^T H u(c(p)) \cdot h + \nabla u(c(p)) \cdot N_h = 0, \tag{B.4}$$

where again we have switched the order of spatial derivatives and the variation by equality of mixed partials. Let c be a parameterization of ∂R with parameter p, and c_p indicate the derivative w.r.t the parameter. The variation of $N = JT = Jc_p/|c_p|$ (J is a 90° rotation matrix) is

$$N_h = J \frac{h_p |c_p| - \frac{c_p \cdot h_p}{|c_p|} c_p}{|c_p|^2} = J(h_s - (h_s \cdot T)T) = -(h_s \cdot N)T. \tag{B.5}$$

Substituting (B.5) into (B.4) leads to the boundary condition in (B.3). □

Definition 4 (Green's Function for (B.3))**.** *The Green's function, $K_{\alpha_i} : R \times R \to \mathbb{R}$,*

for the problem (B.3) *(and* (B.1)*) satisfies*

$$\begin{cases} K_{\alpha_i}(x,y) - \alpha_i \Delta_x K_{\alpha_i}(x,y) = \delta(x-y) & x,y \in R \\ \nabla_x K_{\alpha_i}(x,y) \cdot N = 0 & x \in \partial R, y \in R \end{cases} \tag{B.6}$$

where Δ_x *(*∇_x*) is the Laplacian (gradient) with respect to* x*, and* δ *is the Delta function.*

Lemma 2.0.2 (Region and Boundary Integrals of K)**.** *If* $f : R \to \mathbb{R}$*,* $g : \partial R \to R$ *and*

$$\hat{u}(x) = \int_R K_{\alpha_i}(x,y) f(y)\,\mathrm{d}y - \int_{\partial R} K_{\alpha_i}(x,y) g(y)\,\mathrm{d}s(y) \tag{B.7}$$

where K *is the Green's function that satisfies* (B.6)*, then* $\hat{u}$ *satisfies*

$$\begin{cases} \hat{u}(x) - \alpha_i \Delta \hat{u}(x) = f(x) & x \in R \\ \nabla \hat{u}(x) \cdot N = g(x) & x \in \partial R \end{cases}. \tag{B.8}$$

Proof. This is a standard result in PDE (see, for example [102], for a proof). □

Proposition 2.0.3 (Descriptor Gradient)**.** *The gradient with respect to* $c = \partial R$ *of* $u(x)$ *(one component of* $\mathbf{u}(x)$*), which satisfies the PDE* (B.1)*, is*

$$\nabla_c u(x) = \left[(Du)^T D_y K_{\alpha_i}(x,\cdot) + \frac{1}{\alpha_i} K_{\alpha_i}(x,\cdot)(u - J_j) \right] N \tag{B.9}$$

where N *is the outward normal,* D_y *denotes the derivative wrt the second argument of* K_{α_i}*, and* Du *indicates the spatial derivative of* u*.*

Proof. By the property (B.7), we may express the solution of (B.3) as

$$\begin{aligned} u_h(x) &= -\int_{\partial R} K_{\alpha_i}(x,y) \left[u_s(y) h_s \cdot N - N^T H u(x) \cdot h \right] \mathrm{d}s(y) \\ &= \int_{\partial R} \left[\partial_s (K_{\alpha_i} u_s N) + K_{\alpha_i} N^T H u \right] \cdot h \,\mathrm{d}s, \end{aligned}$$

where ∂_s denotes derivative with respect to arc-length. Therefore, $\nabla_c u(x)$ is the bracketed expression above, which we now simplify. We note that $N_s = \kappa T$, and by differentiating the boundary condition $\nabla u(c(s)) \cdot N = 0$ in s, we find that $N^T Hu(x) \cdot T = -u_s \kappa$ where κ is the signed curvature of c. Using the former two properties, we have

$$\begin{aligned}\nabla_c u(x) &= \partial_s(K_{\alpha_i} u_s)N + K_{\alpha_i} u_s \kappa T + K_{\alpha_i}(N^T Hu \cdot N)N + K_{\alpha_i}(N^T Hu \cdot T)T \\ &= \left[\partial_s(K_{\alpha_i} u_s) + K_{\alpha_i}(N^T Hu \cdot N)\right] N. \end{aligned} \tag{B.10}$$

Now note that $u_{ss} = \partial_s(\nabla u \cdot T) = T^T Hu \cdot T + \nabla u \cdot \kappa N = T^T Hu \cdot T$ using that $\nabla u \cdot N = 0$. This implies that $\Delta u = u_{ss} + N^T Hu \cdot N$. Using (B.10), we have that

$$\nabla_c u(x) = \left[K_{\alpha_i, s} u_s + K_{\alpha_i} \Delta u\right] N. \tag{B.11}$$

Using $\nabla u \cdot N = 0$ on ∂R and $u - \alpha_i \Delta u = J_j$ into (B.11) gives

$$\nabla_c u(x) = \left[\nabla u \cdot \nabla_y K_{\alpha_i}(x, \cdot) + \frac{1}{\alpha_i}(u - J_j) K_{\alpha_i}(x, \cdot)\right] N. \tag{B.12}$$

□

Proposition 2.0.4 (Integrals of Descriptor Gradient)**.** *Let* $\mathbf{f}, \mathbf{g} : R \to \mathbb{R}^M$ *and* $\mathbf{u}$ *be the Shape-Tailored Descriptor in* R *(as in* (B.2)*). Then*

$$\mathbb{I}_d[R, \mathbf{u}, \mathbf{f}, \mathbf{g}] := -\int_{\partial R} \nabla_c \mathbf{u}(x) \mathbf{g}(x) \, \mathrm{d}s(x) + \int_R \nabla_c \mathbf{u}(x) \mathbf{f}(x) \, \mathrm{d}x = \left(tr[(D\mathbf{u})^T D\hat{\mathbf{u}}] + (\mathbf{u} - \mathbf{J})^T A^{-1} \hat{\mathbf{u}}\right) N \tag{B.13}$$

where $\mathrm{d}x$ *is the area measure,* $\mathrm{d}s$ *is the arclength measure,* N *is the outward normal*

to the boundary of R, tr denotes matrix trace, and

$$\begin{cases} \hat{\mathbf{u}}(x) - A\Delta\hat{\mathbf{u}}(x) = \mathbf{f}(x) & x \in R \\ D\hat{\mathbf{u}}(x)N = \mathbf{g}(x) & x \in \partial R \end{cases}. \tag{B.14}$$

Proof. Let u, f, g denote components of $\mathbf{u}$, $\mathbf{f}$, $\mathbf{g}$. Then

$$\int_{\partial R} \nabla_c u(x) g(x)\,\mathrm{d}s(x) = \int_{\partial R} N\left[(Du)^T D_y K_{\alpha_i}(x,\cdot) + \frac{1}{\alpha_i} K_{\alpha_i}(x,\cdot)(u - J_j)\right] g(x)\,\mathrm{d}s(x) \tag{B.15}$$

$$= N\left[\nabla_y \cdot \int_{\partial R} K_{\alpha_i}(x,\cdot)(Du)^T g(x)\,\mathrm{d}s(x) + \frac{1}{\alpha_i}\int_{\partial R} K_{\alpha_i}(x,\cdot)(u - J_j) g(x)\,\mathrm{d}s(x)\right] \tag{B.16}$$

$$= N\left[\nabla_y \cdot (Du)^T \int_{\partial R} K_{\alpha_i}(x,\cdot) g(x)\,\mathrm{d}s(x) + \frac{1}{\alpha_i}(u - J_j)\int_{\partial R} K_{\alpha_i}(x,\cdot) g(x)\,\mathrm{d}s(x)\right] \tag{B.17}$$

Note that $\nabla_y \cdot$ indicates divergence with respect to the second argument of the kernel K_{α_i}, and also that the arguments of u, Du, J_j depend on the point of the curve that has been suppressed for ease of notation. We may follow a similar computation to arrive at

$$\int_R \nabla_c u(x) f(x)\,\mathrm{d}x = N\left[\nabla_y \cdot (Du)^T \int_R K_{\alpha_i}(x,\cdot) f(x)\,\mathrm{d}x + \frac{1}{\alpha_i}(u - J_j)\int_R K_{\alpha_i}(x,\cdot) f(x)\,\mathrm{d}x\right]. \tag{B.18}$$

Then by summing expressions, we arrive at

$$-\int_{\partial R} \nabla_c u(x) g(x)\,\mathrm{d}s(x) + \int_R \nabla_c u(x) f(x)\,\mathrm{d}x = N\left[(Du)^T D\hat{u} + \frac{1}{\alpha_i}(u - J_j)\hat{u}\right] \tag{B.19}$$

where

$$\hat{u}(y) = -\int_{\partial R} K_{\alpha_i}(x,y)g(x)\,\mathrm{d}s(x) + \int_R K_{\alpha_i}(x,y)f(x)\,\mathrm{d}x, \tag{B.20}$$

by symmetry of the Green's function and Lemma 2.0.2. Writing (B.19) and (B.20) in vector form gives the result of this proposition. □

Proposition 2.0.5 (Weighted Area Gradient)**.** *Let $F : \mathbb{R}^M \to \mathbb{R}$ and $\mathbf{u} : R \to \mathbb{R}^M$ be the Shape-Tailored Descriptor on R. Define the weighted area functionals as $A_F = \int_R F(\mathbf{u}(x))\,\mathrm{d}x$. Then*

$$\nabla_c A_F = (F \circ \mathbf{u})N + \mathbb{I}_d[R, \mathbf{u}, (\nabla F) \circ \mathbf{u}, \mathbf{0}] \tag{B.21}$$

where $\mathbb{I}_d$ is defined as in Proposition 2.0.4.

Proof. The gradient above can be derived by using the Chain-Rule. The gradient of the functional, assuming that the descriptor does not vary with the curve, is added to the gradient of the functional with respect to the descriptor. The former is obtained using classical results (e.g., [28]), and is the first term in (B.21). The latter is $\int_R \nabla_c \mathbf{u}(x) \nabla F(\mathbf{u}(x))\,\mathrm{d}x$, which by Proposition 2.0.4 results in the second term of (B.21). □

2.1 Numerical Discretization

We show the discretization scheme for the PDE

$$\begin{cases} u(x) - \alpha \Delta u(x) = f(x) & x \in R \\ \nabla u(x) \cdot N = g(x) & x \in \partial R \end{cases}, \tag{B.22}$$

where N is the outward normal. Note that this is the type of equations that are satisfied by the descriptors $\mathbf{u}, \mathbf{v}$ and the functions $\hat{\mathbf{u}}, \hat{\mathbf{v}}$. We assume that $R = \{x \in \Omega : \Psi(x) \leq 0\}$ where $\Psi : \Omega \to \mathbb{R}$ is the level set function. We use central differences

to discretize the Laplacian:

$$\Delta u(x) = \sum_{y \in N_x} (u(y) - u(x)) = \sum_{y \in N_x \cap R} (u(y) - u(x)) + \sum_{y \in N_x \cap R^c} (u(y) - u(x)) \quad \text{(B.23)}$$

where N_x is a 4-neighbor of x. Note that $u(y)$ for $y \in R^c$ is not defined, but using a discretization of the boundary condition, we have that $u(y) - u(x) = g(x)$ for $y \in N_x \cap R^c$ and $x \in R$. Thus, we have

$$\Delta u(x) = \sum_{y \in N_x \cap R} (u(y) - u(x)) + \sum_{y \in N_x \cap R^c} g(x). \quad \text{(B.24)}$$

Therefore, the final discretization of the PDE is

$$(1 + \alpha |N_x \cap R|) u(x) - \sum_{y \in N_x \cap R} u(y) = f(x) + |N_x \cap R^c| g(x),\ x \in R \quad \text{(B.25)}$$

where $|N_x \cap R|$ is the number of pixels in $N_x \cap R$. This is now in the form where standard linear solvers (e.g., conjugate gradient, multigrid) may be applied.

Note that in narrowband level set methods, the speed function must be extended into the narrowband (1-pixel dilation of R), and this requires that $\mathbf{u}, \hat{\mathbf{u}}$ be extended into the narrowband. Therefore, we show how u defined in (B.22) can be extended to the narrowband. This can be accomplished by discretizing the boundary condition, which yields

$$u(y) = u(x) + g(x),\ y \in N_x \cap R^c \quad \text{(B.26)}$$

and $x \in N_y \cap R$ is such that x is the point with closest distance to a zero crossing of the level set function Ψ.

C Continuum Gaussian Scale Space based Energy and it's Gradient

3.1 Proofs of Lemmas and Propositions

Lemma 3.1.1. *Suppose* $I : \mathbb{R}^2 \to \mathbb{R}$ *and* $a = avg(I) = avg(u(t,\cdot))$. *Then*

$$E = \int_0^\infty \int_{\mathbb{R}^2} |u(t,x) - a|^2 \,\mathrm{d}x \,\mathrm{d}t = \int_{\mathbb{R}^2} |H(\omega)\hat{I}(\omega)|^2 \,\mathrm{d}\omega, \text{ where } H(\omega) = \frac{1}{\sqrt{2}|\omega|}, \quad \text{(C.1)}$$

where $\hat{I}$ *denotes the Fourier transform, and* ω *denotes frequency.*

Proof. Taking the Fourier transform of the Heat Equation:

$$\begin{cases} \partial_t u(t,x) = \Delta u(t,x) & x \in \mathbb{R}^2 \\ u(0,x) = I(x) & t = 0 \end{cases}$$

yields:

$$\partial_t \hat{u}(t,\omega) = (i\omega)\cdot(i\omega)\hat{u}(t,\omega) = -|\omega|^2 \hat{u}(t,\omega),$$

where $\hat{u}(t,\omega)$ is the Fourier transform of u. Solving this differential equation yields

$$\hat{u}(t,\omega) = e^{-|\omega|^2 t}\hat{I}(\omega).$$

We note that $a = 0$ when $I \in \mathbb{L}^2$ since $\hat{I}(0) = \int_{\mathbb{R}^2} I(x)\,\mathrm{d}x$ is finite. Then by Parseval's

Theorem,

$$E = \int_0^\infty \int_{\mathbb{R}^2} |u(t,x) - a|^2 \,\mathrm{d}x \,\mathrm{d}t = \int_0^\infty \int_{\mathbb{R}^2} |\hat{u}(t,\omega)|^2 \,\mathrm{d}\omega \,\mathrm{d}t \tag{C.2}$$

$$= \int_{\mathbb{R}^2} \int_0^\infty e^{-2|\omega|^2 t} \,\mathrm{d}t \cdot |\hat{I}(\omega)|^2 \,\mathrm{d}\omega \tag{C.3}$$

$$= \int_{\mathbb{R}^2} -\frac{1}{2|\omega|^2} e^{-2|\omega|^2 t} \Big|_{t=0}^{t=+\infty} |\hat{I}(\omega)|^2 \,\mathrm{d}\omega = \int_{\mathbb{R}^2} \frac{1}{2|\omega|^2} |\hat{I}(\omega)|^2 \,\mathrm{d}\omega \tag{C.4}$$

$$= \int_{\mathbb{R}^2} |A(\omega)\hat{I}(\omega)|^2 \,\mathrm{d}\omega \tag{C.5}$$

where $A(\omega) = 1/(\sqrt{2}|\omega|)$. □

Lemma 3.1.2. *The Lagrange multiplier λ satisfies the following Heat Equation with forcing term, evolving backwards in time:*

$$\begin{cases} \partial_t \lambda(t,x) + \Delta\lambda(t,x) = f'(u(t,x)) & x \in R \times [0,T] \\ \nabla\lambda(t,x) \cdot N = 0 & x \in \partial R \times [0,T] \cdot \\ \lambda(T,x) = 0 & x \in R \end{cases} \tag{C.6}$$

The solution of this equation can be expressed with Duhamel's Principle [107] as

$$\lambda(t,x) = -\int_t^T F(s-t,x;s) \,\mathrm{d}s. \tag{C.7}$$

where $F(\cdot,\cdot;s) : [0,T] \times R \to \mathbb{R}$ is the solution of the forward Heat equation Eqn. (1) (in the paper) with zero forcing and initial condition $f'(u)$ evaluated at time s, i.e.,

$$\begin{cases} \partial_t F(t,x;s) - \Delta F(t,x;s) = 0 & (t,x) \in [0,T] \times R \\ \nabla F(t,x;s) \cdot N = 0 & x \in [0,T] \times \partial R \quad \cdot \\ F(0,x;s) = f(u(s,x)) & x \in R \end{cases} \tag{C.8}$$

In the case that $f(u) = (u-a)^2$, λ can be expressed as

$$\lambda(t,x) = -2\int_t^T (u(2s-t,x)-a)\,\mathrm{d}s. \tag{C.9}$$

Proof. We define

$$E(R,u,\lambda) = \int_R\int_0^T f(u)\,\mathrm{d}x\,\mathrm{d}t + \int_R\int_0^T (\nabla\lambda\cdot\nabla u + \lambda\cdot u_t)\,\mathrm{d}x\,\mathrm{d}t. \tag{C.10}$$

Integrating by parts, we have that

$$E = \int_{R\times[0,T]} [f(u) - (\partial_t\lambda + \Delta\lambda)u]\,\mathrm{d}x\,\mathrm{d}t \tag{C.11}$$

$$+ \int_R \lambda u|_{t=0}^{t=T}\,\mathrm{d}x + \int_{\partial R\times[0,T]} (\nabla\lambda\cdot N)u\,\mathrm{d}s(x)\,\mathrm{d}t, \tag{C.12}$$

where $\mathrm{d}s$ denotes the arc-length measure of ∂R, and N is the unit outward normal of ∂R. Differentiating E in the direction (perturbation) $\tilde{u}$ of u evaluated at u yields

$$\mathrm{d}E(u)\cdot\tilde{u} = \int_{R\times[0,T]} [f'(u) - (\partial_t\lambda + \Delta\lambda)]\,\tilde{u}\,\mathrm{d}x\,\mathrm{d}t \tag{C.13}$$

$$+ \int_R \lambda\tilde{u}|_{t=0}^{t=T}\,\mathrm{d}x + \int_{\partial R\times[0,T]} (\nabla\lambda\cdot N)\tilde{u}\,\mathrm{d}s(x)\,\mathrm{d}t. \tag{C.14}$$

Note that $\tilde{u}(0) = 0$ since $u(0) = I$ is fixed and thus may not be perturbed. We may choose $\nabla\lambda\cdot N = 0$ on ∂R and $\lambda(T) = 0$. We are interested in u such that $\mathrm{d}E(u)\cdot\tilde{u} = 0$ for all $\tilde{u}$. This yields the condition that

$$\begin{cases} \partial_t\lambda(t,x) + \Delta\lambda(t,x) = f'(u(t,x)) & x\in R\times[0,T] \\ \nabla\lambda(t,x)\cdot N = 0 & x\in\partial R\times[0,T]\cdot \\ \lambda(T,x) = 0 & x\in R \end{cases} \tag{C.15}$$

To express the solution to the above equation in a more convenient form, we may use

Duhamel's Principle. The latter states that a linear PDE with forcing term $\delta(t-s)$ is equivalent to the same PDE with zero forcing and initial condition at s of 1. We may express the forcing term as $\int_{[0,T]} f(u(x,s))\delta(s-t)\,\mathrm{d}s$, and thus combining linearity of the PDE with Duhamel's Principle yields that

$$\lambda(t,x) = -\int_t^T F(s-t,x;s)\,\mathrm{d}s, \tag{C.16}$$

i.e., it is the sum of solutions of the PDE with zero forcing and initial condition $f(u(x,s))$ at time s, specifically,

$$\begin{cases} \partial_t F(t,x;s) - \Delta F(t,x;s) = 0 & (t,x) \in [0,T] \times R \\ \nabla F(t,x;s) \cdot N = 0 & (t,x) \in [0,T] \times \partial R \cdot \\ F(0,x;s) = f(u(s,x)) & x \in R \end{cases} \tag{C.17}$$

In the case that $f(u) = (u-a)^2$ then $f'(u) = 2(u-a)$, the PDE for F becomes

$$\begin{cases} \partial_t F(t,x;s) = \Delta F(t,x;s) & x \in R \times [0,T] \\ \nabla F(t,x;s) \cdot N = 0 & x \in \partial R \times [0,T]\,, \\ F(0,x;s) = 2(u(s,x)-a) & x \in R \end{cases} \tag{C.18}$$

which is the forward Heat Equation with initial condition being the solution of the same Heat Equation evaluated at time s. By the semi-group property of the Heat Equation, we have that

$$F(t,x;s) = 2(u(s+t,x)-a), \tag{C.19}$$

and therefore using (C.16),

$$\lambda(t,x) = -2\int_t^T (u(s+s-t,x)-a)\,\mathrm{d}s = -2\int_t^T (u(2s-t,x)-a)\,\mathrm{d}s. \tag{C.20}$$

□

Proposition 3.1.3. *The gradient of E with respect to the boundary ∂R can be expressed as*

$$\nabla_{\partial R} E = \int_0^T \left[f(u) + \nabla\lambda\cdot\nabla u + \lambda\partial_t u\right]\,\mathrm{d}t\cdot N, \tag{C.21}$$

where N is the normal vector to ∂R. In the case that $f(u) = (u-a)^2$ and as T gets large, the gradient approaches

$$\nabla_{\partial R} E = \left(-\frac{1}{2}|\nabla\lambda(0)|^2 - \lambda(0)[u(0)-a]\right) N, \quad \lambda(0,x) = -\int_0^{2T} (u(s,x)-a)\,\mathrm{d}s, \tag{C.22}$$

where $\lambda(0)$ and $u(0)$ denote the functions λ and u at time zero.

Proof. To compute the gradient of E, we compute the gradient of E in (C.10) with respect to ∂R treating λ and u independent of R as in the theory of Lagrange multipliers. In this case, this is just a classical result in the calculus of variations (e.g., [28]), in particular the integrand (with respect to R) is multiplied by the outward normal along ∂R to obtain the gradient:

$$\nabla_{\partial R} E = \int_0^T \left[f(u) + \nabla\lambda\cdot\nabla u + \lambda\partial_t u\right]\,\mathrm{d}t\cdot N. \tag{C.23}$$

Note that using a change of variables $\tau = 2s - t$, we may write λ in (C.9) as

$$\lambda(t,x) = -\int_t^{2T-t} (u(\tau,x)-a)\,\mathrm{d}\tau, \quad t\in[0,T], \tag{C.24}$$

as in (C.22).

If $f(u) = (u-a)^2$, then we may write $\nabla E = FN$ where

$$F = \int_0^T (u-a)^2 + \nabla\lambda \cdot \nabla u + \lambda \partial_t u \, \mathrm{d}t. \tag{C.25}$$

Integrating by parts in t yields

$$F = \int_0^T (u-a)^2 + \nabla\lambda \cdot \nabla u - \partial_t \lambda u \, \mathrm{d}t - \lambda(0)u(0). \tag{C.26}$$

If we let $T \to \infty$, then

$$\lambda(t,x) = -\int_t^\infty (u(\tau,x) - a) \, \mathrm{d}\tau, \tag{C.27}$$

and so

$$F = \int_0^\infty (u-a)^2 + \nabla\lambda \cdot \nabla u - u(u-a_i) \, \mathrm{d}t - \lambda(0)u(0) \tag{C.28}$$

$$= \int_0^\infty -a(u-a) + \nabla\lambda \cdot \nabla u \, \mathrm{d}t - \lambda(0)u(0), \tag{C.29}$$

where we used integration by parts and noted that $\lambda(\infty) = 0$. Therefore,

$$F = \int_0^\infty \nabla\lambda \cdot \nabla u \, \mathrm{d}t - \lambda(0)(u(0) - a), \tag{C.30}$$

by noting the first term is $a\lambda(0)$. We may now simplify the integral above:

$$\int_0^\infty \nabla\lambda(t,x) \cdot \nabla u(t,x) \, \mathrm{d}t = -\int_0^\infty \int_t^\infty \nabla u(\tau,x) \cdot \nabla u(t,x) \, \mathrm{d}\tau \, \mathrm{d}t \tag{C.31}$$

$$= -\frac{1}{2} \int_0^\infty \int_0^\infty \nabla u(\tau,x) \cdot \nabla u(t,x) \, \mathrm{d}\tau \, \mathrm{d}t \tag{C.32}$$

$$= -\frac{1}{2} \int_0^\infty \nabla u(\tau,x) \, \mathrm{d}\tau \cdot \int_0^\infty \nabla u(t,x) \, \mathrm{d}t \tag{C.33}$$

where we have used symmetry of the integrand in the second line above. Therefore,

$$\int_0^\infty \nabla\lambda(t,x)\cdot\nabla u(t,x)\,\mathrm{d}t = -\frac{1}{2}\left|\int_0^\infty \nabla u(t,x)\,\mathrm{d}t\right|^2 = -\frac{1}{2}|\nabla\lambda(0)|^2, \tag{C.34}$$

and thus substituting into (C.30), we find that

$$F = -\frac{1}{2}|\nabla\lambda(0)|^2 - \lambda(0)(u(0)-a). \tag{C.35}$$

□