A Fixation Based Segmentation Framework

Ajay Kumar Mishra
(B.Tech, Indian Institute of Technology, Kanpur(2003))

A thesis submitted
for the degree of doctor of philosophy
department of electrical and computer engineering
National University of Singapore

2010

Acknowledgments

I would first like to thank my supervisors, Dr. Cheong Loong Fah and Dr. Ashraf Kassim, for all the support and guidance they provided over the five years of my PhD. I would also like to thank Dr. Stefan Winkler for his support as my supervisor for the period of 2006-2007. I have had tremendous opportunity to learn from all of my supervisors. I feel truly enriched both academically and professionally.

During my PhD, I was lucky to get excellent opportunities to interact and work with many prominent researchers. While those were all exciting and stimulating people to work with, I am particularly grateful to Prof. Yiannis Aloimonos for sharing his knowledge and wisdom with me. He has been a true inspiration to me as a researcher and thinker.

Finally, I express my deepest gratitude to my family, my parents, and my siblings for all of their emotional support during the long and strenuous journey of PhD. It was their constant and unwavering support that helped me plow ahead even when the odds were stacked against me. And for that, I say to them: Thank you!

Contents

Summary

This thesis proposes a novel fixation based segmentation algorithm that takes a point ("fixation") in the image (or video) as input and outputs the region containing that fixation point. Such formulation of segmentation with fixation is a well-defined problem in contrast with the traditional segmentation formulation that tries to break an image (or a scene) into mutually exclusive regions. The is shown qualitatively in the thesis that the traditional definition of segmentation is not well defined and that segmentation can be defined optimally only if the object of interest is identified prior to segmentation.

The proposed algorithm carries out segmentation as a two-step process. In the first step, all visual cues, static monocular, stereo and(or) motion, are used to generate a probabilistic boundary edge map that contains the probability of an edge pixel being at a depth boundary. In the second step, the probabilistic boundary edge map is transformed from the Cartesian space to the polar space with the fixation point chosen as the pole for this transformation. In the polar space, the segmentation of the fixated region is carried out as a binary labeling problem that finds the optimal cut through the polar edge map, which becomes the closed contour around the fixation point as it gets transformed back to the Cartesian space.

Motivated by the experiments in the psychophysics that suggest humans do not just make a single fixation but a series of them, a subset of which are related to each other, we propose a multi-fixation strategy that starts with a given fixation and makes a series of dependent fixations to segment the object of interest completely even when the shape of the object is complex. The multiple fixation strategy is also used to segment complex shaped objects especially the ones with a thin elongated part.

Finally, an attempt to use the sparse motion information instead of the dense optic flow map to segment moving objects has also been made. The motivation for this lies in the difference between motion and color cues. Motion cues are inherently sparse since motion can be detected unambiguously only at some salient locations in the scene whereas color cues are known at every location with high accuracy. We propose an algorithm to segment moving objects in a video without having to calculate the dense optic flow map of the scene.

List of Figures

[1]Thanks to Anthony Santella
[2]Thanks to Anthony Santella

List of Tables

Chapter 1

Introduction

To see an object in the scene, we look at it. The human visual system makes a series of fixations at various salient locations while it observes the scene and try to makes sense of it [110]. See Fig. 1.1 for the fixations made by an observer while looking at a still image, as recorded by an eye tracker. The eyes can not be static. They have to keep moving to a new location either voluntarily or involuntarily. There are four main types of eye movements:

Saccades are the ballistic eye movements between consecutive fixations in the scene (see the dotted lines in Fig. 1.1(b)). The amplitude of a saccade varies greatly depending upon the task. It ranges from a small movement while reading a text to a large shift while observing a big room. Saccades can be made voluntarily but are usually made unconsciously while the eyes are open. At the end of every saccade, the eyes stay fixated at the target location for at least 200-300ms before starting a new saccade.

Smooth pursuit movements , as the name suggests, are made to keep a moving stimulus on the fovea of both eyes. In the absence of a moving stimulus, it is hard to generate this type of eye movement voluntarily. So, this eye movement is made unconsciously while we look at a moving stimulus in the scene. However, an observer can, in the presence of a moving stimulus, choose to either track it or not.

Vergence movements align a close target object on the fovea of both eyes to perceive its distance from the observer. It is useful only for the objects in the close vicinity of the observer.

Vestibulo-ocular movements are made to compensate for the shift in the visual image on the retina caused by the movement of the head. It is a reflexive eye movement.

1.1 Motivation

Out of the four types of eye movements mentioned above, saccades are the most general and non-specific type of eye movement. The eyes move continuously to fixate at “salient” locations in the scene. Even during a fixation, the eyes are not completely static but move slightly in a small neighborhood around the fixation point [55]. The fact that the eyes keep moving raises an important question: why does the human visual system make all these eye movements after all?

The eye movements have long been studied in the visual attention research with the focus on examining the characteristics (e.g color, intensity, texture cues) of the locations in the scene that draw attention of the human visual system [67, 80, 92, 43, 99, 22]. These studies are generally based on the data collected using an eye tracker as it records the eye movements of observers looking at still images and(or) videos on a monitor. On the basis of this information, a number of computational models have been proposed that estimate the likelihood of a pixel in the image to be fixated [43, 68, 22]. An image of these likelihood values for all pixels is also called

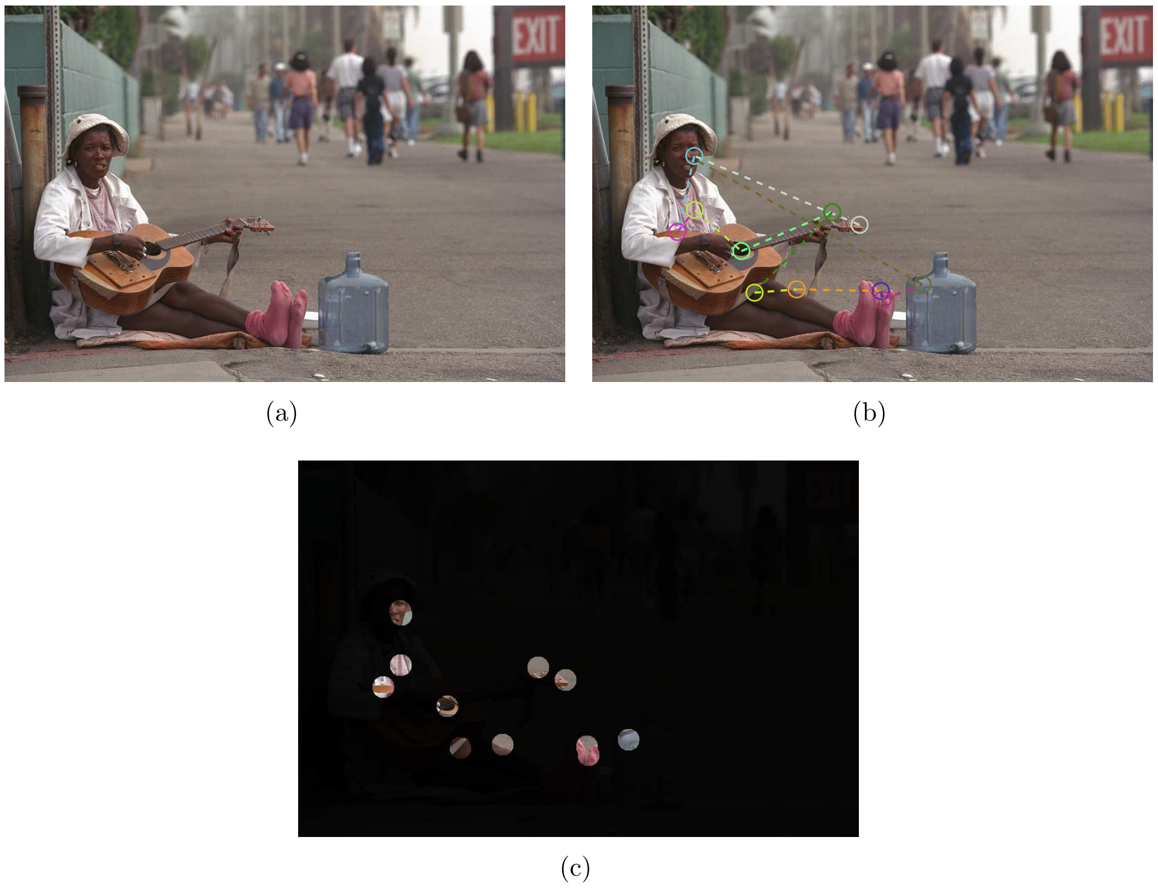


Figure 1.1: (a) A street image[3] (b) The eye movement of a human subject as recorded by an eye tracker[4]. The fixation points are indicated by circles and a saccade between fixations is shown by a dotted line. (c) The small circular regions centered at the fixation points.

saliency map. While these approaches have been able to predict the possible locations to fixate, they do not explain what happens while the human visual system is fixated at a particular location in the scene.

The most prevalent view on why the human visual system fixates at any location is that the fixations are made to capture high resolution visual information from the salient locations in the scene. The basis of this viewpoint lies in the structure of the human retina which has a high concentration of cones (with fine resolution) in the central fovea [85, 107]. So, as the eyes fixate at different locations, the representation of a scene is built in terms of the highly accurate details captured from the fixated locations (see Fig. 1.1(c)). While this might be a reason to fixate at different locations in the scene, it suggests a limited role of fixation in visual perception. There are psychophysical experiments on change blindness that suggests a more critical role of fixation in our visual perception.

The phenomenon of change blindness is, in itself, a counter example to the traditional view that the visual system generates a complete representation of the external world it sees. It has been reported that, during change blindness experiments, observers are unable to notice a significant change in the scene during a saccade [56, 35, 18] unless the change occurs close to the target location of the saccade [25] or it alters the gist of the scene. Recently, [40, 38] further establish a strong connection between the relative distance between the fixation point and the changing stimulus, and the accuracy of change detection. The detection accuracy is significantly higher if the changing stimulus is directly fixated upon. This clearly suggests a fundamental role of fixation in generating visual perception of a scene (or image).

While we do not claim to know the exact purpose of these eye movements, we certainly draw our inspiration from the need of the human visual system to fixate at different locations to consciously perceive those parts of the scene. We believe that fixation should be an essential component of any visual system. Motivated by the need to make fixation a part and parcel of visual processing, we hypothesize that, during a fixation, the human visual system segments the region it is currently fixating at in the scene. Besides incorporating fixation in the segmentation framework, this new segmentation formulation is also well-defined compared to the traditional approach to segmentation, as we shall see in the next subsection.

1.2 Fixation based segmentation: a well-posed problem

In computer vision literature, segmentation essentially means breaking a scene into non overlapping and compact regions made up of connected pixels with similar properties. Over the years, many different algorithms [96, 78, 28] have been proposed to segment an image into regions, but the definition of what is a correct or "desired" segmentation of an image (or scene) has largely been elusive to the computer vision community. In fact, in our view, this traditional problem definition is not well-posed.

To illustrate this point further, let us take an example of a scene (or image) shown in Fig. 1.2. In this scene, consider two of the prominent objects: the tiny horse and the pair of trees. Fig. 1.2(b) and Fig. 1.2(c) are the segmentation of the image using the normalized cut algorithm [78] for different input parameters (these outputs would also be typical of many other segmentation algorithms). Now, if we ask the question: which one of the two is the correct segmentation of the image? The answer to this question depends entirely on another question: which is the object of interest in the scene? In fact, there cannot be a single correct segmentation of an image unless it has only one object in prominence in which case the correct segmentation of the image is essentially the correct segmentation of that object.

With respect to a particular object of interest, the correct/desired segmentation of the scene is the one wherein the object of interest is represented by a single or just a couple of regions. So, if the tiny horse is of interest, the segmentation shown in Fig. 1.2(c) is correct whereas the segmentation shown in Fig. 1.2(b) is correct if the trees are of interest. In fact, in Fig. 1.2(b), the horse does not even appear in the segmentation. So, the goal of segmenting a scene is intricately linked with the object of interest in the scene and can be well defined only if the object of interest is identified and known to the segmentation algorithm beforehand.

But, knowing the object of interest even before segmenting the scene seems to turn the problem as one of many chicken and egg problems in computer vision, as we usually need to segment the scene in order to recognize objects in it. So, how can we identify an object even before segmenting it? What if the identification of the object of interest is just a weak identification such as a point on that object? Now, obtaining these points without doing any segmentation is not a difficult problem as it can be done using low-level cues in the visual attention systems.

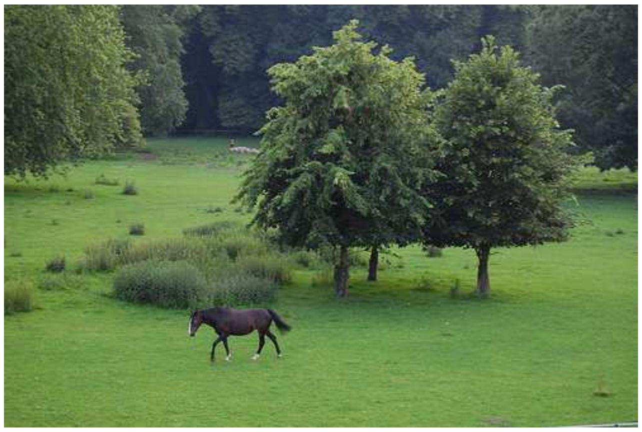
A visual attention system can predict the location in the scene that attracts attention [67, 92, 99, 22]. The human visual system has two types of attention: overt attention (eye movements) and covert attention (without eye movement). In this work, we mean overt attention whenever we use the term attention. The attention causes eye to move and fixate at this new location in the scene. Each fixation will lie on an object, identifying that object (which can be a region in the background too) for the segmentation step. Now, segmenting that fixated region is defined as finding the "optimal" enclosing contour – a connected set of boundary edge fragments – around the fixation. This new formulation of segmenting fixated regions is a well-defined problem.

Note that we are addressing an easier problem than the general problem of segmentation where one attempts to find all region segments at once. In the general segmentation formulation, the exact number of regions is not known and thus several ad-hoc techniques have been proposed to estimate this number automatically. In fact, for a scene with prominent objects appearing at significantly different scales, having a single global parameter for segmenting the scene is not even meaningful, as explained above.

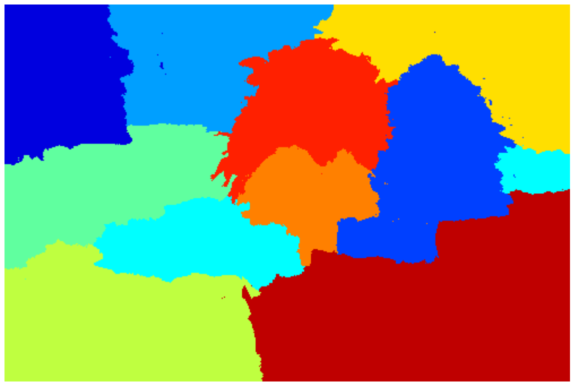
1.3 Overview

We propose a segmentation framework that takes as its input a fixation (a point location) in the scene and outputs the region containing that fixation. The fixated region is segmented in terms of the area enclosed by the "optimal" closed boundary around the fixation using the probabilistic boundary edge map of the scene (or image). The probabilistic boundary edge map, which is generated by using all available visual cues, contains the probability of an edge pixel being at an object (or depth) boundary. The separation of the cue handling from the actual segmentation step is an important contribution of our work, because it makes segmentation of a region independent of types of visual cues that are used to generate the probabilistic boundary edge map.

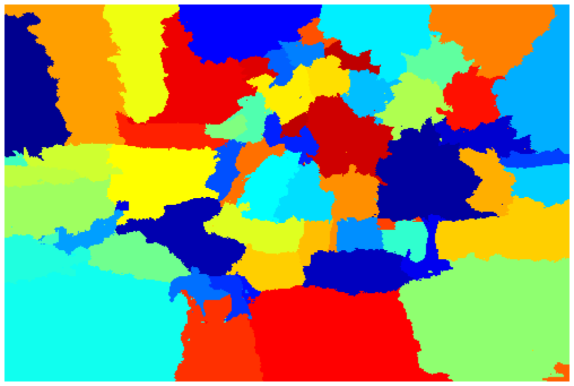
The proposed segmentation framework is a two step process: first, the probabilistic boundary edge map of the image is generated using all available low level cues (section 3.2); second, the probabilistic edge map is transformed into the polar space with the fixation as the pole (section 3.3), and the path through this polar probabilistic edge map (the blue line in Fig.3.4g) that "optimally" splits the map into two parts (Fig.3.4f) is found. This path maps back to a closed contour around the fixation point. The pixels on the left side of the path in the polar space correspond to the inside of the region enclosed by the contour in the Cartesian space, and those on the



(a)



(b)



(c)

Figure 1.2: Segmentation of a natural scene in (a) using the Normalized Cut algorithm [78] for two different values of its input parameter (the expected number of regions) 10 and 60 are shown in (b) and (c) respectively.

right side correspond to the outside of that region. So, finding the optimal path in the polar probabilistic edge map is a binary labeling problem and graph cut is used to find this globally optimal solution to this binary problem (section 3.4).

The main contributions of the work described in this thesis are as follows;

- We propose an automatic method to segment an object (or region) given a fixation on that object (or region) in the scene/image. Segmenting the region containing a given fixation point is a well-posed binary labeling problem in the polar space, generated by transforming the probabilistic boundary edge map from the Cartesian space to the polar space with the fixation point as pole. In the transformed polar edge map, lengths of the possible closed contours around the fixation point are normalized (section 3.1) and thus the segmentation results are not affected by the scale of the fixated region. The proposed framework does not depend upon any user input to output the optimal segmentation of the fixated region.

- Since we carry out segmentation in two separate steps, it provides an easy way to incorporate feedback from the current segmentation output to influence the segmentation result for the next fixation by just changing the probabilities of the edge pixels in the probabilistic boundary edge map (see chapter 7 for how it is used to generate a multi-fixation framework). Also, using noisy motion and stereo cues to only modify the boundary probabilities of the static monocular edges provides better localization of the region boundaries while tracing actual depth boundaries around any fixation point in the scene.

Chapter 2

Literature Review

Segmentation is one of the extensively researched topics of computer vision. It is beyond the scope of this thesis to provide an extensive review of all segmentation methods proposed so far. Instead, a short discussion on the important aspects of the segmentation philosophy shared among different algorithms is presented. The segmentation algorithms are analyzed from four different viewpoints: 1) Objective, 2) Grouping criteria, 3) User parameters, 4) Visual cues.

2.1 Objective

Segmentation algorithms differ greatly in their ultimate objective. Segmentation is generally carried out as an intermediate process to some high level visual tasks such as object recognition and localization [36, 72, 51, 88], 3D reconstruction [74, 39]. In these circumstances, segmentation is seen as a tool to break a scene into regions that are fewer in number and more distinctive than pixels. Here, segmentation is just a means to an end.

On the other hand, there are segmentation approaches that intend to segment the object of interest from its background, usually termed as foreground/background segmentation [16, 111, 7, 71]. Clearly, for these methods, the exact segmentation of the foreground object is their end result. Another example can be medical image segmentation where, once again, the objective is not to recognize but to accurately delineate the contours of organs such as lung, heart, liver in, say X-ray images. For these algorithms, segmentation is not just a means to an end but the end itself.

2.1.1 Segmentation is a means to an end

Segmentation as an intermediate process is relatively less constrained and has to output just a suitable decomposition containing regions that are not too big or too small. A decomposition containing large regions is called under-segmentation. It is usually undesirable because the regions contain pixels from surfaces at different depths which are usually unrelated. It is the other kind of decomposition, called over-segmentation, that is used as an intermediate processing step. Even here, an overly refined segmentation is useless as each region has just too few pixels to be discriminative, but a boundary preserving over-segmentation [28] is often used to obtain the regions for the next level of processing.

[51] considers the segmented region as a complete object and uses it to compare against the stored exemplars of the objects in the database. Obviously, such an assumption will lead to error when there is no single region that contains the object in the scene completely. The performance of this method critically depends upon getting the right segmentation of the scene. In [88], they employ a better strategy and represent objects as region trees. It helps avoid, to some extent, the problems due to variation in the segmentation of the object.

Recently, regions are being used to determine 3D orientation of the scene from just a single images [74, 39]. In these works, regions are treated as basic processing units, also called superpixels. They learn the association between the local appearance (such as color, texture) of these superpixels with their geometrical properties, its 3D orientation. Once learned, the mapping can be used to assign a particular orientation to a region in the scene without having to use any real depth information such as from stereo vision or from a laser sensor.

2.1.2 Segmentation is the end

While any of the several possible over-segmentations is a reasonable outcome for several high level processing, it is not so when the objective is to segment the object (or foreground) in the scene. The segmentation in this case is unique – the region containing the entire object and no background pixels. One of the research topics where it is required to accurately cut out the foreground is in interactive segmentation literature.

The interactive segmentation algorithms [16, 111, 7, 71] always segment the entire image into only two regions: foreground and background. [16] poses the problem of foreground/background segmentation as a binary labeling problem which is solved exactly using the maxflow algorithm[15]. It, however, requires users to label some pixels as foreground or background to build their color models. [13] improved upon [16] by using a Gaussian mixture Markov random field to better learn the foreground and background models. [71] requires users to specify a bounding box containing the foreground object. [4] requires a seed point for every region in the image. For foreground/background segmentation, at least two seed points are needed. Although these approaches give impressive results, they can not be used as an automatic segmentation algorithm as they critically depend upon the user inputs. [111] tries to automatically select the seed points by using spatial attention based methods and then use these seed points to introduce extra constraints into their normalized cut based formulation.

[97, 7] need only a single seed point from the user. [97] imposes a constraint on the shape of the object to be a star, meaning the algorithm prefers to segment convex objects. Also, the user input for this algorithm is critical as it requires the user to specify the center of the star shape exactly in the image. [7] needs only one seed point to be specified on the region of interest and segments the foreground region using a compositional framework. However the algorithm is computationally intensive. It runs multiple iterations to arrive at the final segmentation.

Finally, there are several active contour models [45, 108] that take as an input a parametric contour and evolve it into a locally optimal contour in the image. The pioneering work on this topic was done by Kass et al. [45] using the proposed Snakes. Since then, several variants of Snakes (e.g. finite elements snakes [23],B-Snakes [12]) have been proposed over the years. Another variant of this approach where contours are implicitly represented is based on level sets proposed by Osher and Sethian [66]. In the end, all of these algorithms output a segmented region close to the initialized contour.

2.2 Grouping criteria

The boundary and the inside of a region are complementary information. The boundary is represented by the edge pixels in the scene. So, the problem of finding a boundary can be seen as that of grouping edge fragments, also called edgels, into a group on the basis of their perceptual similarity. Such methods are termed as contour based methods in this work. Traditionally, segmentation algorithms put together similar pixels to form a homogeneous region. Sometimes, the grouping criteria is determined by the discontinuity (the edges) in the scene rather than comparing the pixels itself.

These methods are termed as pixel-based methods.

2.2.1 Contour based

Contour based segmentation methods start with finding edge fragments in the image first, and then joining the edge fragments to form closed contours. The regions are enclosed by each of these closed contours. Due to the presence of textures and low contrast regions in the image, detecting edge fragments is a hard problem. The second step of joining the edge fragments are done in probabilistic fashion using image statistics. In [105], first order Markov model is used for contour shape and the contours were completed using random walk. In [70], multiple scales are used to join the contours using orientation and texture cues. [70, 105, 83] are edge based segmentation methods.

Similar to the global region based segmentation methods, edge based segmentation algorithms suffer from ambiguity of choosing the appropriate closed loops which are actually the boundaries of the regions in the image. To avoid that confusion, the contour based interactive segmentation algorithms[59, 8] need the user to specify the seed points along the contour to be traced. [45, 108] need the user to initialize a closed contour which then evolves to adjust the actual boundary in the image.

Contour fragments can also be grouped on the basis of Gestalt cues such as parallelism, continuity, proximity and closure. In fact, a lot of research has been done on the topic of perceptual grouping where low-level image elements are grouped together generally in a hierarchical fashion to form a more semantically significant entities [47].

2.2.2 Pixel based

A simple way to form regions is to start at any seed pixel and combine it with all the pixels in its neighborhood if the difference between the intensity of the seed pixel and the pixel in the neighborhood is below a threshold. Such an approach is also called pixel aggregation [34]. Growing regions in such a way leads to unstable segmentation. So, instead of using a threshold, a statistical approach is used to group the pixels together. Two regions can merged if the pixels in both regions come from the same distribution[81].

2.3 User input

Without user input(s), it is impossible to define the optimal segmentation. Every segmentation method needs an user input to stop the process of segmentation and output the optimal decomposition of the scene as its segmentation. Interactive techniques need an extensive user input whereas global methods only require a few parameters.

2.3.1 Global parameter

Generally, an image or a scene has only one object in prominence and correctness of the segmentation is decided depending upon how well that object is segmented in the current decomposition. Such cases, however, needs only one parameter connected with the scale of that object. As it is difficult to estimate the scale of the object automatically, the segmentation algorithms depend on the users to specify a parameter(s) to output the optimal decomposition of the scene. For instance, in [50], the expected number of regions is required as an input and [28] takes the threshold to stop their clustering process.

Segmentation is essentially a clustering process wherein pixels with similar properties form a compact set. An image is generally considered to be a graph with each pixel (or node) connected to either 4 or 8 of its neighboring pixels. The edge connecting two neighboring pixels i and j is weighted according depending upon the similarity or dissimilarity between them, which is decided by examining the visual cues like color, texture with these pixels.

In [50], the edge weight is computed based on the texture cues and the intervening contour between the pixels. The graph is divided into clusters using eigenvectors of the similarity matrix formed by collecting all the edge weights. In [28], the dissimilarity of the color information of the pixels are used to assign the weights to the edges and clusters are formed in an hierarchical clustering fashion. The criterion to group the nodes as it moves up the hierarchy is adapted to the degree of variability in the neighboring regions. Both [28, 32] and all other region based segmentation algorithms need a user input to stop the process of grouping the pixels. [50] needs the expected number of regions as input whereas [28] takes the threshold to stop the clustering process. In fact, without any user input, it is impossible to define the optimal segmentation. There are many other segmentation algorithms [95, 112] which are based on global user parameters like the number of regions or threshold.

As it is difficult to estimate the parameter that can result in the optimal decomposition, the resulting segmentation is usually either under-segmented or over-segmented. Over segmentation is usually undesirable because once the pixels from different objects are merged into one, post processing cannot correct that step. By the same logic, the high level algorithms which work with regions prefer to work with over-segmented regions as they can then be put together to segment the entire object or just roughly estimate the location of the object in the scene (e.g. bounding box around the detected object).

2.3.2 Interactive segmentation

The snakes can be regarded as an example of an interactive segmentation algorithm. These algorithms depend upon user input to constraint the problem of segmentation. Also, the ultimate objective of these algorithms is generally the segmentation of a particular object in the scene.

The user inputs to these algorithms are specified either in terms of points with specified labels (e.g foreground/background) [16, 71, 83] or a closed curve close to the actual contour of the object of interest [45, 108]. Sometimes the user input is not the points inside the regions or the closed curve approximating the contour, but the points along the contour of the object [59].

The interactive segmentation generally intends to segment the image into two regions: foreground and background. But, how they use the user inputs to generate the knowledge of the foreground and background differs between these algorithms. In [71], the rectangle containing the object of interest is specified by the user and the distribution of the color of the foreground and background pixels is learned from using the pixels inside and outside that rectangle. Once done, these distribution are used to assign probabilities to all pixels to be foreground and background; a graph cut formulation is then used to find the optimal labels for all the pixels in the image. Such an approach works well if the foreground and background have predominantly different colors. Also, the foreground should encompass most of the rectangle such that the background pixels present in the rectangle do not contaminate the foreground distribution. In [83], a few pixels are assigned hard labels to be either foreground or background and using that information the contour separating these labeled pixels optimally are found using random walk.

Instead of asking users to specify the points inside foreground and background regions and then finding the contour that separates them in some optimal sense, Mortensen and Barrett [59] ask users to specify points along the contour of the object itself and then find the shortest path between these points to output a closed curve passing through all the specified points. Although specifying points along the contour require more manual work than selecting a few points inside foreground or background regions, the segmentation output has better boundary accuracy than the point based methods discussed above [71, 83]. Snakes and level set methods make the contour based segmentation easier as they only require the users to specify a closed curve around the object which then deforms to become the actual contour [45, 108].

2.3.3 Automatic segmentation

A simple way to get rid of having to use an user defined parameter is to output a set of possible segmentations instead of one optimal decomposition of the scene. A simple example of such an approach is the hierarchical segmentation approach based on agglomerative clustering [27]. The output of a hierarchical segmentation is a tree with the root node representing the entire image as one region and the leaf nodes representing individual pixels in the image. Any intermediate node represents a cluster of connected pixels with similar properties. At each stage of the clustering process, the neighboring regions are merged depending on the similarity between them. In [76], a sophisticated hierarchical segmentation method inspired by *Algebraic Multigrid* finds salient regions at different levels of the hierarchy as those with very weak similarity with the neighboring regions at that level. In [5], an oriented watershed algorithm along with ultra metric contour map is used to generate the segmentation tree from a probabilistic contour map.

For some high level applications such as object recognition [36], content-based image retrieval [77], the entire segmentation tree can be used as it contains all the salient regions at different scales. But, once again, if the requirement is to segment an object in the scene, some form of user input is required. So, [77] extends the *Algebraic Multigrid* based formulation of [76] and uses probabilistic framework to automatically select the most salient decomposition. But, such a decomposition would only make sense for the image that has only one prominent object and thus the most salient decomposition would correspond to that of the object in the image.

2.4 Visual cues

But, in order to effectively use these regions, it is important that they are big enough to be distinctive and, not so big that they merge pixels from across depth boundaries. The decomposition should be appropriate but that needs an estimate of the scale of the objects in the scene. In most cases, one scale is good enough for the entire image as prominent objects appear at similar scales in that image or that only one object is in prominence. But, if the objects are of significantly different sizes, more than one scales are required as different objects would only appear in the segmentation corresponding to their scale. In such cases, hierarchical approach to segmentation is appropriate as it outputs several decompositions of a scene at appropriate scales.

As stated earlier, segmentation is a clustering process. So far, the discussion has been about how the clustering process stops using given user inputs. Another important factor pertains to how the similarity or dissimilarity between pixels are determined especially when different cues (e.g. static monocular cues, motion and stereo cues) are present in the scene. If only the values of static monocular cues such as such as color, texture, brightness, contrast etc are used to compare the pixels, such segmentations are called image segmentation methods. On the other hand, if flow values are used to decide if the two pixels should go together, the resulting segmentation methods are called motion segmentation. In some cases, stereo information is used to segment the region assuming the pixels with similar disparity values should belong to a region.

The problem with using stereo and motion information is that they are inaccurate at the depth boundaries in the scene which leads to inaccuracy of the region boundaries. Besides, in the case of motion, it is even hard to define what it means for two pixels to be together because the pixels from within same region can have different flow vectors (consider a spinning wheel). In any case, a brief survey of different motion segmentation approaches is given next.

2.4.1 Motion segmentation

Prior research in motion segmentation can broadly be classified into two groups:

(a) The approaches relying on 2D motion measurements only [103, 63, 14, 21]. There are many limitations in these techniques. Depth discontinuities and independently moving objects both cause discontinuities in the 2D optical flow, and it is not possible to separate these factors without 3D motion and structure estimation. Generally, dense optical flow is calculated at every point in the image and like in the image segmentation, the flow value of each pixel is used to decide similarity between the pixels which is used to cluster them into regions with consistent motion. The main problem with this approach is that the optical flow is inaccurate at the boundaries and hence the region obtained by this approach has generally poor boundaries.

To overcome this problem, many algorithms first segment the frames into regions and then merge the regions by comparing the overall flow of the two regions. The accuracy of this method is dependent on the accuracy of the image segmentation step. If a region is produced by the image segmentation step which includes parts from different objects in the scene, it cannot be corrected by the later processing of combining regions into bigger regions. To avoid that problem, some techniques over-segment the image into small regions to reduce the chances of having overlapping regions. But discriminating small regions on the basis of their overall flow is difficult.

(b) 3D approaches which identify clusters with consistent 3D motion [24, 90, 82, 60, 87, 113, 1] using a variety of techniques. Some techniques, such as [102] are based on alternative models of image formation. These additional constraints can be justified for domains such as aerial imagery. In this case, the planarity of the scene allows a registration process [94, 6, 104, 114], and un-compensated regions correspond to independent movement.

This idea has been extended to cope with general scenes by selecting models depending on the scene complexity [89], or by fitting multiple planes using the plane plus parallax constraint [73, 41]. Most techniques detect independently moving objects based on the 3D motion estimates, either explicitly or implicitly. Some utilize inconsistencies between ego-motion estimates and the observed flow field, while some utilize additional information such as depth from stereo, or partial ego-motion from other sensors. The central problem faced by all motion based techniques is that, in general, it is extremely difficult to uniquely estimate 3D motion from flow. Several studies have addressed the issue of noise sensitivity in structure from motion. In particular, it is known that for a moving camera with a small field of view observing a scene with insufficient depth variation, translation and rotation are easily confused [2].

2.4.2 Stereo segmentation

[46] combines color, texture and stereo cues to segment a binocular video into foreground and background regions. The computation of disparity values occurs simultaneously with the segmentation of the foreground. The video however should be captured with static cameras. In this paper, we segment the videos captured with a moving camera and with multiple independently moving objects in the scene. Also,

we compute the low level cues like color, texture, stereo and motion separately and use all the cues only to create a better probabilistic boundary map. The segmentation step of finding the optimal closed boundary is only affected through the probabilistic boundary map.

Chapter 3

Segmenting a fixated region

As stated earlier in chapter 1, segmenting a fixated region is equivalent to finding the "optimal" closed contour around the fixation point. This closed contour should be a connected set of boundary edge pixels (or fragments) in the edge map. However, the edge map contains both types of edges, namely, boundary (or depth) and internal (or texture/intensity) edges. In order to trace the boundary edge fragments through the edge map to form the contour enclosing the fixation point, it is important to be able to differentiate between the boundary edges from the non-boundary (e.g. texture and internal) edges.

We generate a probabilistic boundary edge map of the scene wherein the intensity of an edge pixel is proportional to its probability to be at an object (or depth) boundary. The intensity ranges from 0 to 1. In qualitative terms, the boundary edges will appear brighter (darker) than the internal and texture edges in the (inverse) probabilistic boundary edge map. All available visual cues are used to generate such an edge map. The static monocular cues (e.g. color and texture) are used to generate an initial edge map which is modified using stereo or motion cues. The detailed discussion on how we use binocular cues along with monocular cues to generate the

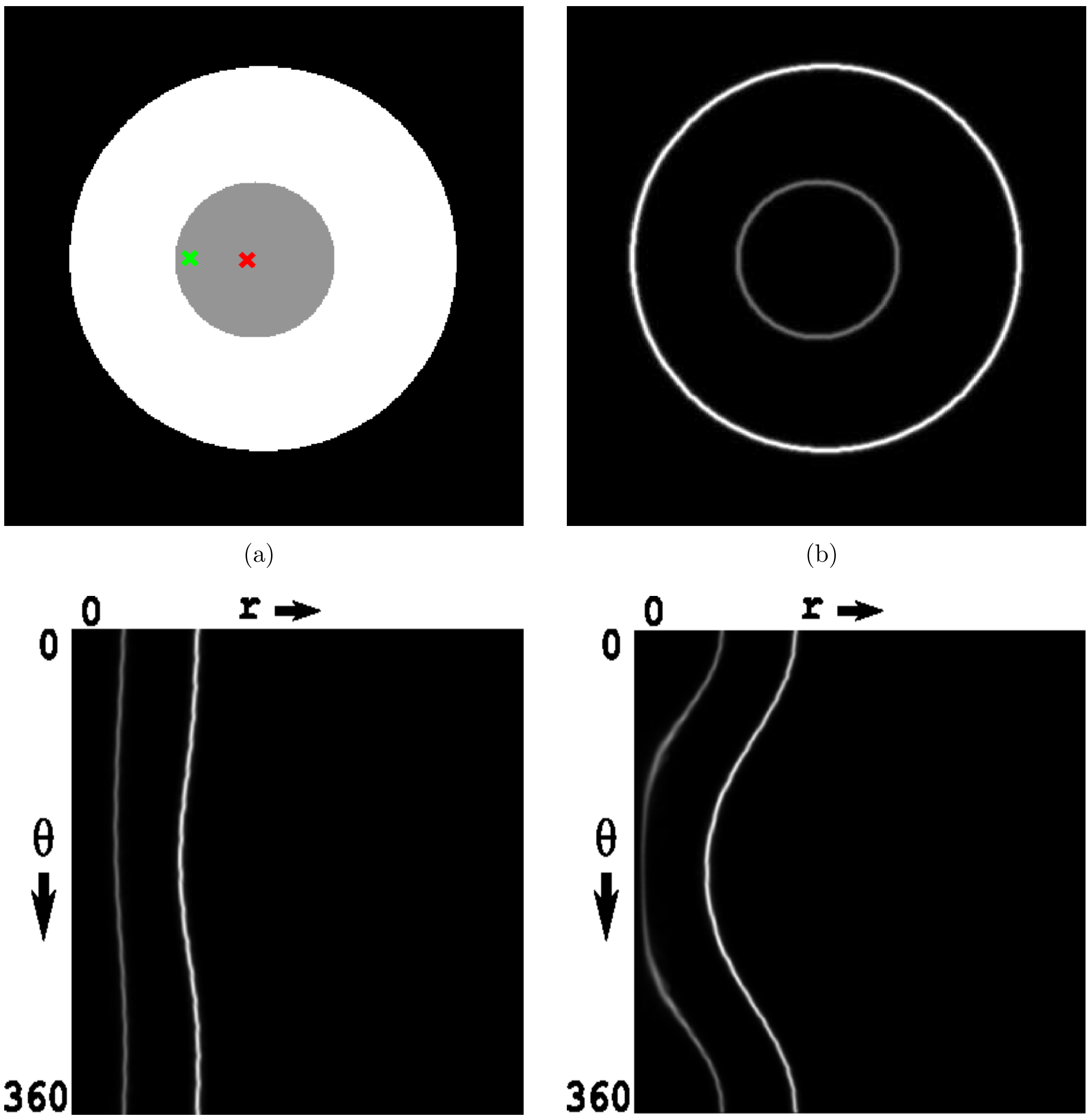


Figure 3.1: (a) The Disc. (b) The gradient edge map. (c) and (d) are the polar edge maps, generated by transforming the gradient edge map of the disc w.r.t the red and green fixations respectively.

probabilistic boundary edge map is given in section 3.2.

Now, any algorithm that traces the closed contour through the probabilistic boundary edge map in the Cartesian space inherently prefers smaller contours as the overall cost is, in essence, the product of the length of the closed contour and the average cost of tracing the edge pixels along the contour. For possible closed contours with similar average boundary probabilities for their edge pixels, the scale makes smaller contours preferable over bigger contours. We propose a solution to the scale problem as transferring the edge map from the Cartesian to the polar coordinate system (section 3.1) and segment the polar probabilistic boundary edge map to find the closed contour (see section 3.4).

3.1 Polar space is the key!

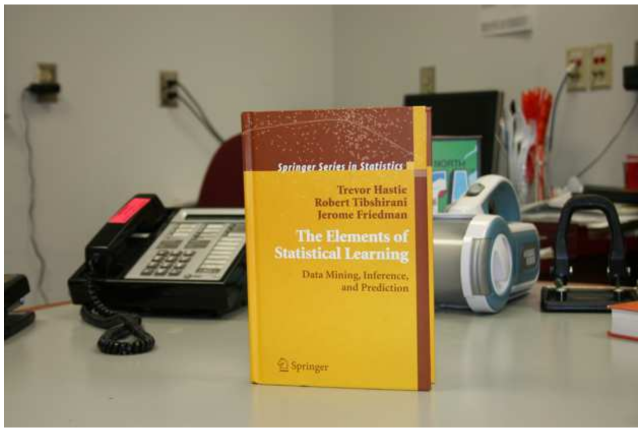
Let us consider finding the optimal contour around the red fixation point on the disc shown in Fig. 3.1a. The gradient edge map of the disc, shown in Fig. 3.1b, has two concentric circles. The big circle is the actual boundary of the disc whereas the small circle is just the internal edge on the disc. The edge map correctly assigns the boundary contour intensity as 0.78 and the internal contour 0.39 (the intensity values range from 0 to 1). The lengths of the two circles are 400 and 100 pixels. Now, the cost of tracing the boundary and the internal contour in the Cartesian space will be respectively $88 = (400 \times (1 - 0.78))$ and $61 = (100 \times (1 - 0.39))$. Clearly, the internal contour costs less and hence will be considered optimal even though the boundary contour is the brightest and should actually be the optimal contour. In fact, this problem of inherently preferring short contours over long contours has already been identified in the graph cut based approaches where the minimum cut usually prefers to take “short cut” in the image [83].

To fix this “short cut” problem, we have to transfer these contours to a space where their lengths no longer depend upon the area they enclose in the Cartesian space. The cost of tracing these contours in this space will now be independent of their scales in the Cartesian space. The polar space has this property and we use it to solve the scale problem. The contours are transformed from the Cartesian co-ordinate system to the polar co-ordinate system with the red fixation point in Fig. 3.1b as the pole. In the polar space now, both contours become open curves, spanning the entire θ axis starting from $0°$ to $360°$. See Fig. 3.1c. Thus, the costs of tracing the inner contour and the outer contour become $80.3 = 365 \times (1 - 0.78)$ and $220.21 = 361 \times (1 - 0.39)$ respectively. As expected, the outer contour (the actual boundary contour) costs the least in the polar space and hence becomes the optimal enclosing contour around the fixation.

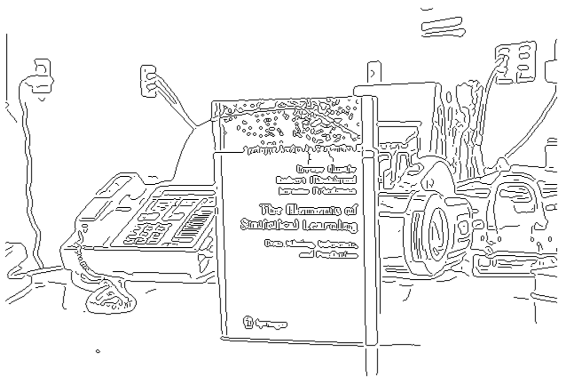
It is important to make sure that the optimal path in the polar space is stable with respect to the location of the fixation, meaning that as the fixation point moves to a new location the optimal path in the polar space for this new fixation location should still correspond to the same closed contour in the Cartesian space. For the new fixation point (the green “X”) in Fig.3.1b, both contours have changed shape (See Fig.3.1d), but the “optimal” (or brightest) contour remains the same. A detailed on the issue of stability with respect to the change in fixation location is done in section 4.3.1.

3.2 Probabilistic boundary edge map by combining cues

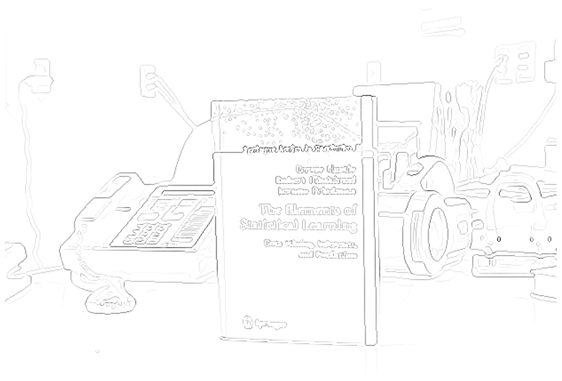
In this section, we carry out the first step of the segmentation process: generating the probabilistic boundary edge map using all available visual cues. There are two


Springer Series in Statistics
Trevor Hastie
Robert Tibshirani
Jerome Friedman
The Elements of Statistical Learning
Data Mining, Inference, and Prediction
Springer

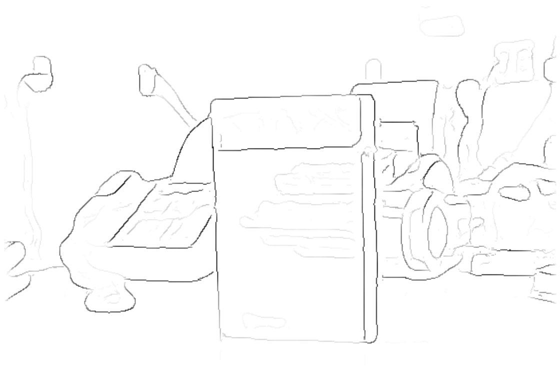
(a)



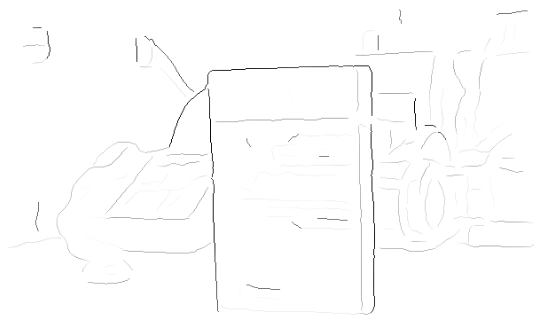
(b)



(c)



(d)



(e)

Figure 3.2: Inverse probabilistic edge maps of the color image shown in (a). Darker pixels mean higher probability. (b) The Canny edge map. (c) The gradient edge map. (d) The output of the Berkeley pb detector [53]. (e) The final probabilistic boundary edge detector on combining static monocular cues with the motion cue.

types of visual cues on the basis of how they are calculated: 1) static monocular cues, that come from just a single image; 2) stereo and motion cues, that need more than one image to be computed. The former cues such as color, intensity or texture can precisely locate the edges in the scene, but cannot distinguish between an internal texture edge from an edge at a depth discontinuity. On the other hand, stereo and motion can help distinguish between boundary and internal edge as there is a sharp gradient in disparity and flow across the former whereas no significant change occurs across the latter. But, unlike the static monocular cues, the stereo and motion cues are generally inaccurate at the boundary itself. This shows the need to use the stereo and(or) motion cues with the static monocular cues in a complementary fashion such that together they both identify and precisely locate the boundary edges in the scene.

3.2.1 Using color/intensity and texture cues only

What if we only have a single image without any motion or stereo cues to help disambiguate the boundary edges from the rest? In that case, we need some intelligent way to make the distinction between edges. Let us start with the Canny edge map (Fig. 3.2(b)) of the image (Fig. 3.2(a)). The Canny edge detector finds edges at all the locations where there is a gradient in the intensity and returns a binary edge map, meaning all edge pixels are equally important. This makes the binary edge map useless for our purpose. However, if we assign the magnitude of the gradients at these locations as their respective probability of being at the boundaries, we have a meaningful boundary edge map. But, it still has two main problems: first, the gradient magnitude is not always a good indicator of whether an edge pixels is at a boundary or not; second, Canny or similar intensity based edge detectors are unable to find boundaries between textures and, rather, create strong edge responses inside a textured region.

Recently, an edge detector has been proposed by Martin et al. [53] that learns, using a linear classifier, the color and texture properties of the pixels at region boundaries versus the edge pixels from inside the regions from human-labeled images segmentations. The learned classifier is then used to assign appropriate probability (between 0 and 1) to the computed edges for being at the region boundary. The edge detector handles texture in the image better than Canny or similar intensity based edge detectors. (See Fig. 3.2(d). The spurious texture edges from Fig. 3.2(c) have been successfully removed). However, even this edge map, though expected, has strong internal edges (See BC, CD, CF in Fig. 3.3(a)) which are not actual depth boundaries.

So, in the case of a single image, the output of the Berkeley edge detector is the probabilistic estimate of the boundary edge map. We will use it to segment the fixated regions in the image and show that we obtain segmentation accuracy close to the state of the art on the Alpert segmentation database [3]. But, in the presence of stereo or motion cues, we modify this edge map such that the probability of the edge pixels with strong gradient in either disparity or flow values are reinforced as they are supposed to be either at a depth discontinuity or at the boundary of a moving object. By the same logic, the edge pixels with no change in the disparity or flow values are the internal edges and hence their probabilities should be reduced.

3.2.2 Using stereo with color and texture cues

Let us take the case of stereo first. We compute a dense disparity map for a pair of rectified images using an algorithm proposed by Ogale and Aloimonos [65]. Let us say, the range of disparity values lies between 0 and maximum value D. Our objective is to use the disparity values to decide if an edge pixel is at a depth discontinuity.

Depth discontinuity causes a sudden change in the disparity values and the amount

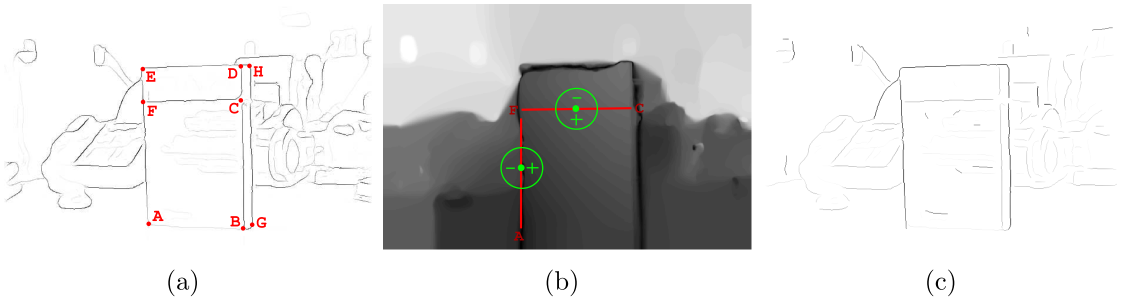


Figure 3.3: Reinforcing the depth boundaries and suppressing the internal edges in the boundary edge map generated using the monocular cue, shown in (a), to generate the final depth boundary shown in (c) using the magnitude of the flow values, shown in (b).

of change depends on the actual physical depth variation at the edge and the camera configuration. Also, the disparity values does not change across the internal edges on the object, barring small variations due to the error in the disparity map itself. So, the edge pixel with considerable change in the disparity values is considered to be a boundary edge. On the other hand, the edge pixels with a slight change in the disparity value are considered internal edges.

Using the relative disparity across the edge pixels to change its boundary probability is in agreement with the finding of the neurophysiological study [86] where they found that the depth perception in a monkey brain depends upon the relative but not absolute disparity. However, how a given amount of relative change should affect how much of the boundary probability is an important question which we cannot answer one way or the other.

By using plain logic, the amount of change in the disparity should not matter as it occurs due to relative placement of the objects in the scene. A depth boundary between two closely placed but occluding objects should be as good a depth boundary as the one between a tree against a background far away from it.

In order to know the disparity change at an edge pixel, we accumulate the disparity values in the square patches centered at an equal distance from the edge pixel in the direction perpendicular to the orientation of the edge pixel. The reason not to choose the patches from the immediate neighborhood of the edge pixel is to avoid accessing noisy values of disparity close to the depth boundary. Also, due to the presence of noise in the disparity map, computing average change in disparity values for a straight edge fragment will be more robust than calculating the disparity change for individual pixels.

Let us say Δd represents the average absolute disparity change for an edge fragment. In this work, we consider all the edge pixels along an edge fragment to be at a depth boundary if $\Delta d > 2$. Now, the actual probability of the edge pixels along the edge fragments is reinforced by adding a constant value 1.0. After analyzing all the edge fragments, the modified probabilistic boundary edge map is rescaled to lie in the range of $[0, 1]$.

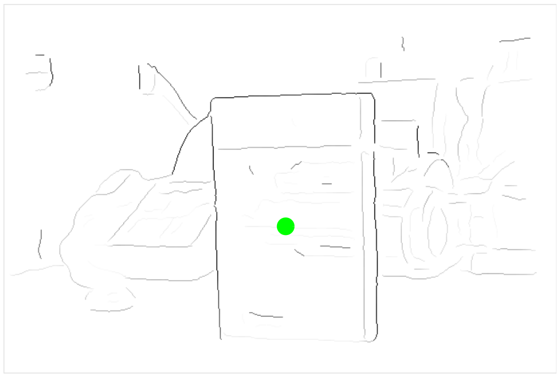
3.2.3 Using motion cues with color and texture

Motion is different from stereo for two main reasons: first, unlike stereo where a non-boundary edge does not have disparity change across it, an internal edge can also have a valid change in the flow across it. For instance, if a wheel is spinning along its axis, the flow vectors change direction across the spokes of the wheel, which are internal edges. Second, the optical flow vector representing motion information is a 2D vector whereas the disparity is a scalar quantity making it easier to calculate the change in the disparity than in the flow vector.

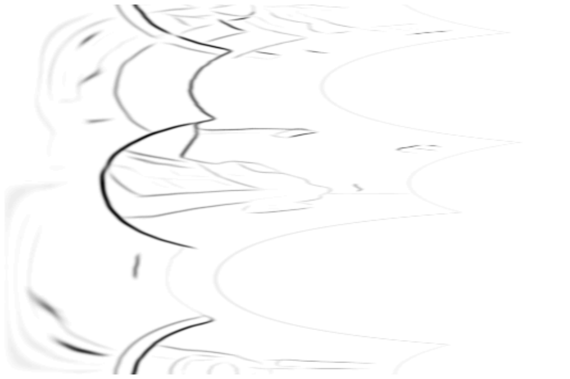
Although it is beyond the scope of this paper to define a consistently moving object in terms of the flow vectors on it, here we consider any change in the magnitude of the flow vector across an edge as a measure of depth discontinuity. This definitely holds well when the relative translational motion between an object and the camera is translation in the X-Y plane only. As the translational motion in most videos primarily involves translational motion, the assumption holds good for all of them as it is evident in our experiments.

Let us say E_{pb} and E_{int} denote the final probabilistic boundary edge map of the image using all visual cues and the static monocular cues respectively. The intensity of an edge pixel, representing the likelihood of that pixel being at the boundary, is calculated as $E_{pb}(x, y) = E_{int}(x, y) + \alpha(H(||\Delta f|| > T_f))$ for motion cues and $E_{pb}(x, y) = E_{int}(x, y) + \alpha(H(\Delta d > 2.0))$ for stereo cues where $H(.)$ is an indicator function, $||\Delta f||$ is the magnitude of the change in the average flow vectors l, Δd is the change in disparity, T_f (which is 3.0 in our case) is the threshold on the change in magnitude at depth boundaries, and α (which is 1.0 in our case) is the constant amount added to reinforce the probability of the depth boundaries. The final boundary edge map is shown in Fig.3.4d wherein the internal edges are dim and the boundary edges are bright. With the improved boundary edge map, as the algorithm traces the brightest closed contour (AGHEA shown in Fig 3.3(a)) around the fixation point, it will also be the real depth boundary of the region containing the fixation (Fig.3.4e).

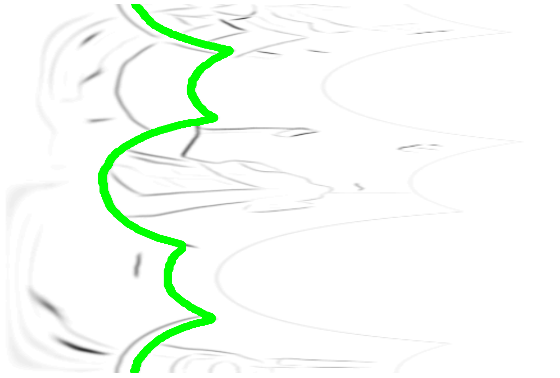
In the final boundary edge map, the image borders are also added as edges to ensure enclosedness for those fixations lying on the regions partially present in the image. For instance, see the car in the column 5 of Fig.4.1 (a part of its closed boundary is actually the left border of the image.) The intensity of the edges corresponding to image borders is kept low such that they are not preferred over the real edges in the edge map.



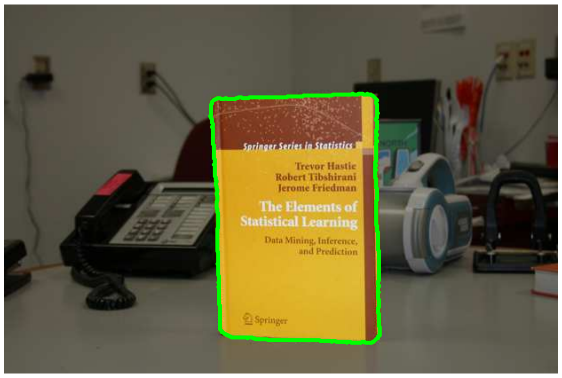
(a)



(b)



(c)



(d)

Figure 3.4: (a) The inverse probabilistic boundary edge map after combining motion cues with static monocular cues. The fixation is shown by the green circular dot. (b) The polar edge map generated using the fixation as the pole. (c) The optimal contour through the polar edge map, splitting it into two parts: inside(left) and outside(right). (d) The closed contour around the fixation when transferred back to the Cartesian space.

3.3 Cartesian to polar edge map

Let's say $E_{pb}^{pol}$is the corresponding polar plot of the probabilistic boundary edge map E_{pb} in the Cartesian space and $F(x_o, y_o)$ is the selected pole (that is the fixation point). Now, a pixel $E_{pb}^{pol}(r, \theta)$ in the polar coordinate system corresponds to a sub-pixel location $\{(x, y) : x = rcos\theta + x_o, y = rsin\theta + y_o\}$ in the Cartesian co-ordinate system. $E_{pb}(x, y)$ is typically calculated by bi-linear interpolation which only considers four immediate neighbors.

We propose to generate a continuous 2D function $W(.)$ by placing 2D Gaussian kernel functions on every edge pixel. The major axis of these Gaussian kernel functions is aligned with the orientation of the edge pixel. The variance along the major axis is inversely proportional to the distance between the edge pixel and the pole O. Let S be the set of all edge pixels. The intensity at any sub-pixel location (x, y) in Cartesian coordinates is

$$W(x, y) = \sum_{i \in S} exp(-\frac{x_i^t}{\sigma_{x_i}^2} - \frac{y_i^t}{\sigma_{y_i}^2}) \times E_{pb}(x_i, y_i)$$

$$\begin{bmatrix} x_i^t \\ y_i^t \end{bmatrix} = \begin{bmatrix} cos\theta_i & sin\theta_i \\ -sin\theta_i & cos\theta_i \end{bmatrix} \begin{bmatrix} x_i - x \\ y_i - y \end{bmatrix},$$

where $\sigma_{x_i}^2 = \frac{K_1}{\sqrt{(x_i - x_o)^2 + (y_i - y_o)^2}}$, $\sigma_{y_i}^2 = K_2$, θ_i is the orientation at the edge pixel i, $K_1 = 900$ and $K_2 = 4$ are constants. The reason for setting the square of variance along the major axis, $\sigma_{x_i}^2$, to be inversely proportional to the distance of the edge pixel from the pole is to keep the gray values of the edge pixels in the polar edge map the same as the corresponding edge pixel in the Cartesian edge map. The intuition behind using variable width kernel functions for different edge pixels is as follows: Imagine an edge pixel being a finite sized elliptical bean aligned with its orientation, and you look at it from the location chosen as pole. The edge pixels closer to the pole (or center) will appear bigger and those farther away from the pole will appear smaller.

The polar edge map $E_{pb}^{pol}(r, \theta)$ is calculated by sampling $W(x, y)$. The values of E_{pb}^{pol} are scaled to lie between 0 and 1. An example of this polar edge map is shown in Fig.3.4h. Our convention is that the angle $\theta \in [0°, 360°]$ varies along the vertical axis of the graph and increases from the top to the bottom whereas the radius $0 \le r \le r_{max}$ is represented along the horizontal axis increasing from left to right. r_{max} is the maximum Euclidean distance between any two locations in the image.

3.4 Finding the optimal cut through the polar edge map: an inside vs outside segmentation

Let us consider every pixel $p \in P$ of E_{pb}^{pol} as a node in a graph. Every node (or pixel) is connected to their 4 immediate neighbors (Fig. 3.5). A row of the graph represents the ray emanating from the fixation point at an angle (θ) equal to their row index. The first and the last rows of this graph are the rays $\theta = 0°$and $\theta = 360°$ respectively which are essentially the same ray in the polar representation. Thus, the pairs of nodes $\{(0°, r), (360°, r)\}$, $\forall r \in [0, r_{max}]$ should be connected by edges in the graph. The set of all the edges between neighboring nodes in the graph is denoted by Ω. Let us assume $l = \{0, 1\}$ are the two possible labels for each pixel where $l_p = 0$ indicates 'inside' and $l_p = 1$ denotes 'outside'. The goal is to find a labeling $f(P) \mapsto l$ that corresponds to the minimum energy where the energy function is defined as:

$$Q(f) = \sum_{p \in P} U_p(l_p) + \lambda \sum_{(p,q) \in \Omega} V_{p,q} . \delta(l_p, l_q) \tag{3.1}$$

$$V_{p,q} = \begin{cases} exp(-\eta E_{pb,pq}^{pol}) & \text{if} \quad E_{pb,pq}^{pol} \neq 0 \\ k & otherwise \end{cases} \tag{3.2}$$

$$\delta(l_p, l_q) = \begin{cases} 1 & \text{if} \quad l_p \neq l_q \\ 0 & otherwise \end{cases} \tag{3.3}$$

where $\lambda = 50$, $\eta = 5$, $k = 20$, $E_{pb,pq}^{pol} = (E_{pb}^{pol}(r_p, \theta_p) + E_{pb}^{pol}(r_q, \theta_q))/2$, $U_p(l_p)$ is the cost of assigning a label l_p to the pixel p and $V_{p,q}$ is the cost of assigning different labels to the neighboring pixels p and q.

At the start, there is no information about how the inside and outside of the region containing the fixation looks like. So, the data term $U(\cdot)$ for all the nodes in the graph except those in the first column and the last column is zero:$U_p(l_p) = 0, \forall p \in (r, \theta), 0 < r < r_{max}, 0° \leq \theta \leq 360°$. However, the nodes in the first column which correspond to the fixation point in the Cartesian space must be inside and are initialized to the label 0: $U_p(l_p = 1) = D$ and $U_p(l_p = 0) = 0$ for $p \in (0, \theta), 0° \leq \theta \leq 360°$. The nodes in the last column on the other hand must lie outside the region and are initialized to the label 1: $U_p(l_p = 0) = D$ and $U_p(l_p = 1) = 0$ for $p \in (r_{max}, \theta), 0° \leq \theta \leq 360°$. See Fig.3.5. In our experiments, we choose D to be 1000; the high value is in order to make sure the initial labels to the first and the last columns do not change as a result of minimization. We use the graph cut algorithm [17] to minimize the energy function, $Q(f)$. The binary segmentation step splits the polar edge map into two parts: left side (inside) and right side(outside). See Fig. 3.4f. The binary segmentation is finally transferred back to the Cartesian space to get the desired segmentation.

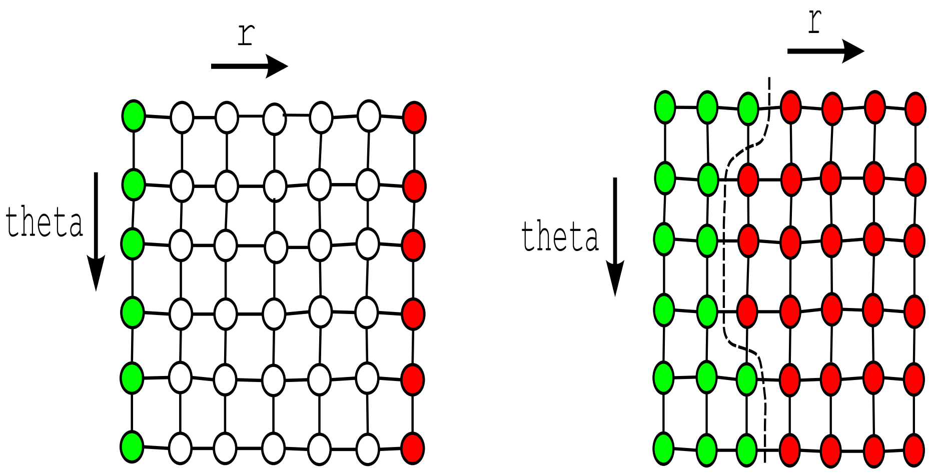


Figure 3.5: Left: The green nodes in the first column are initialized to be inside whereas the red nodes of the last column are initialized to be outside the region of interest. Right: the binary labeling output after minimizing the energy function using graph cut. Note that though the first and the last rows in our graph are connected, they are not shown connected by an edge here for the sake of clarity.

Chapter 4

Experimental Evaluation

4.1 Segmentation Accuracy

Our dataset is a collection of 20 videos with an average length of seven frames and 50 stereo pairs with respect to their ground-truth segmentation. For each sequence and stereo pair, only the most prominent object of interest is identified and segmented manually to create the ground-truth foreground and background masks. The fixation is chosen randomly anywhere on this object of interest. The videos used for the experiment are of all types: stationary scenes captured with a moving camera, dynamic scenes captured with a moving camera, and dynamic scenes captured with a stationary camera.

The segmentation output of our algorithm is compared with the ground truth segmentation in terms of the F-measure defined as $2PR/(P+R)$ where P stands for the precision which calculates the fraction of our segmentation overlapping with the ground truth, and R stands for recall which measure the fraction of the ground-truth segmentation overlapping with our segmentation.

Table 4.1 shows that after adding motion or stereo cues with color and texture

For videos	F-measure
With Motion	0.95 ± 0.01
Without Motion	0.62 ± 0.02
For stereo pairs	
With Stereo	0.96 ± 0.02
Without Stereo	0.65 ± 0.02

Table 4.1: The performance of our segmentation for the videos and the stereo pairs. See Fig.4.1

cues, the performance of the proposed method improves significantly. With color and texture cues only, the strong internal edges prevent the method from tracing the actual depth boundary. (See Fig. 4.1 Row 2). However, the motion or stereo cues clean the internal edges as described in section 3 and the proposed method finds the correct segmentation (Fig.4.1Row 3).

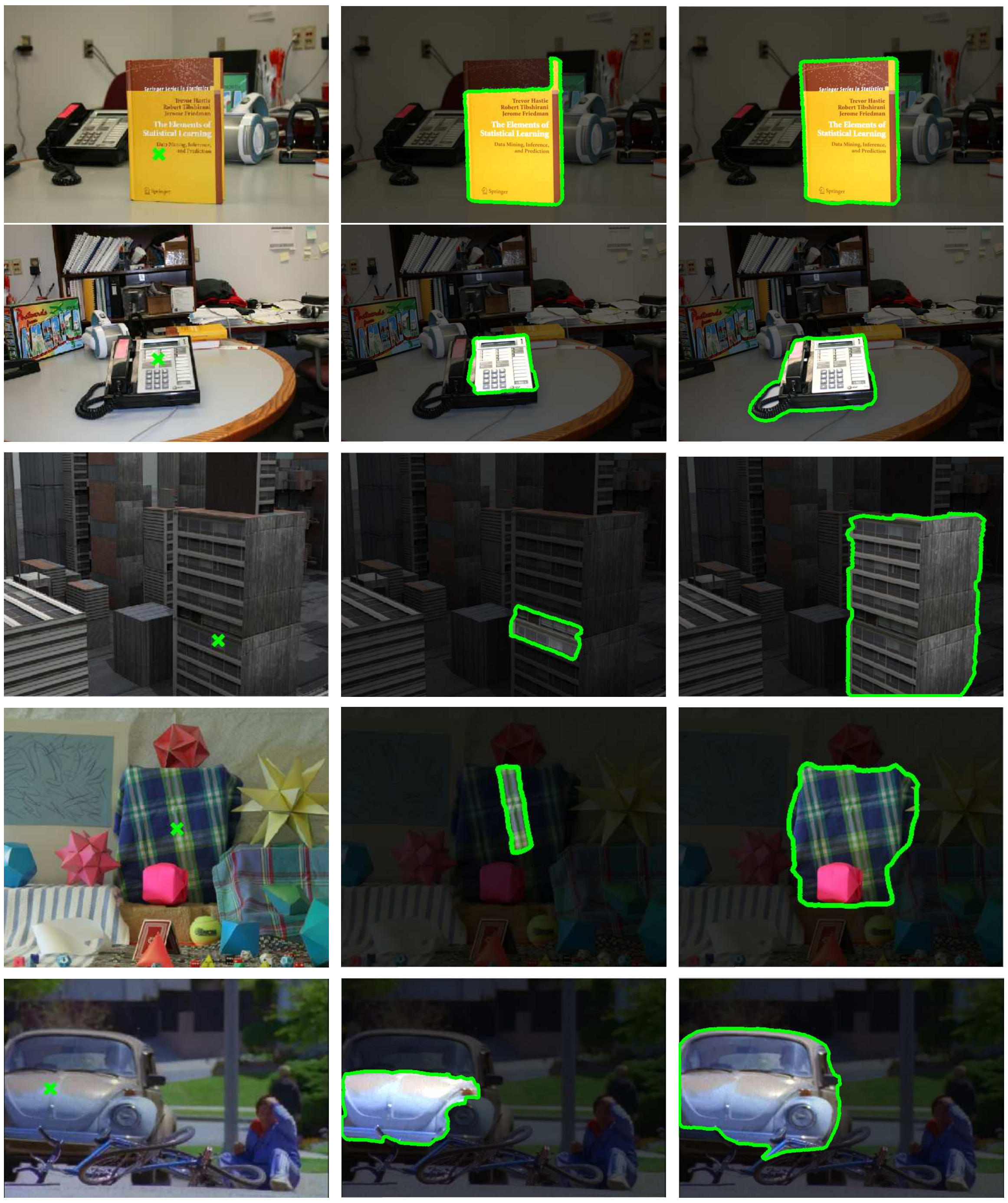


Figure 4.1: Row 1-3: a moving camera and stationary objects. Row 4: an image from a stereo pair. Row 5: a moving object (car) and a stationary camera. Column 1:the original images with fixations (the green "X"). Column 2: Our segmentation results for the fixation using static monocular cues only. Column 3: Our segmentation results for the same fixation after combining motion or stereo cues with static monocular cues.

To also evaluate the performance of the proposed algorithm in the presence of the static monocular cues only, the images from the Alpert image database [3] have been used. The Berkeley edge detector [53] provides the probabilistic boundary maps of these images. The fixation on the image is chosen at the center of the bounding box around the foreground. Our definition of the segmentation for a fixation is the region enclosed by the depth boundary. Table 4.2 shows that we perform better than [78, 96] and close to [3, 7]. The definition of segmentation in [7] is such that, for a selected seed on any of the two horses in Fig. 4.3a, both horses will be segmented. This illustrates that seed point in [7] has no significance other than selecting a good initial segment to start the processing of segmentation. In contrast, our segmentation finds only the horse being fixated, making the so called "seed point" of our algorithm a meaningful input which identifies the object of interest.

Algorithm	F-measure score
Bagon et al. [7]	0.87 ± 0.010
Alpert et al. [3]	0.86 ± 0.012
Our Method	0.83 ± 0.019
NCut [78]	0.72 ± 0.012
MeanShift [96]	0.57 ± 0.023

Table 4.2: One single segment coverage results. The scores for other methods except [7] are taken from [3].

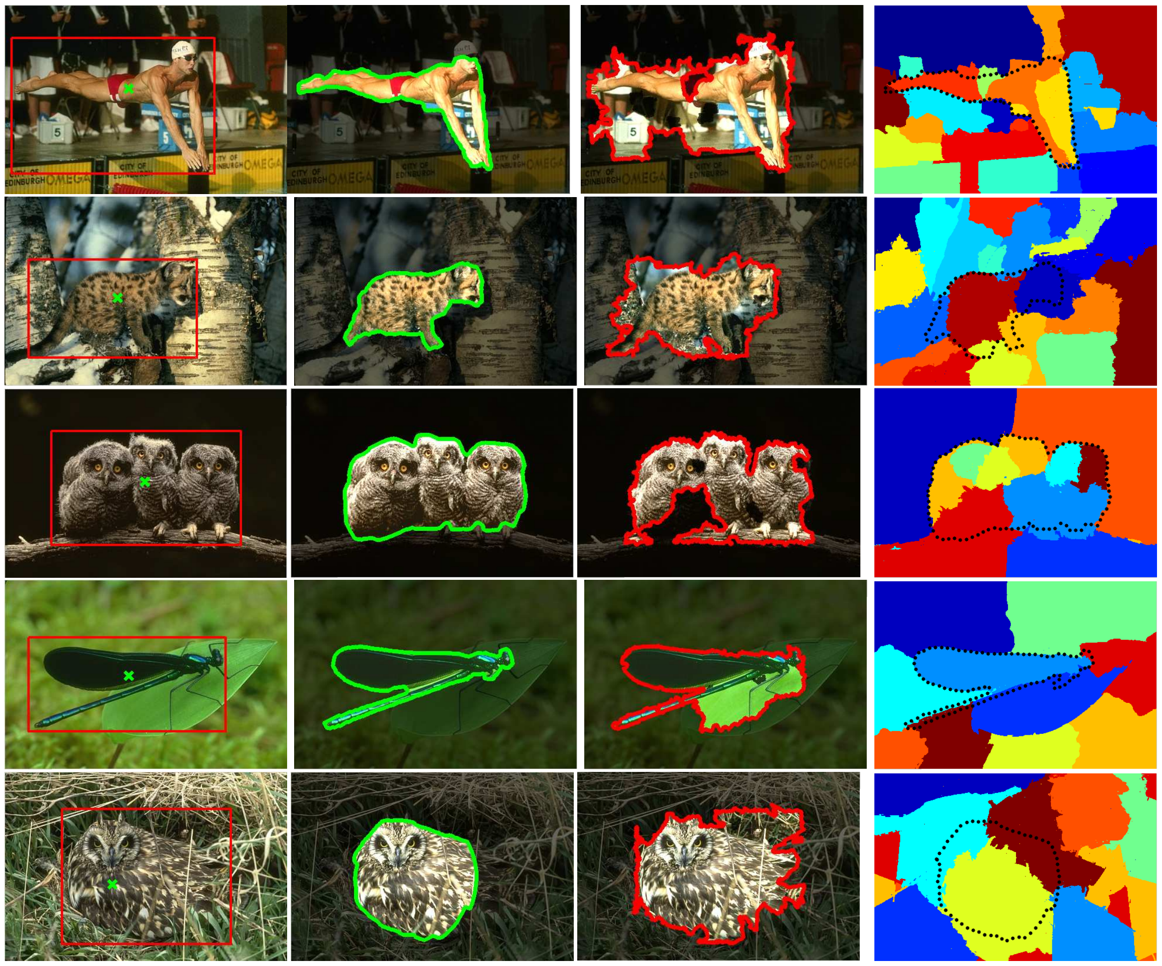


Figure 4.2: The first row contains images with the fixation shown by the green X. Our segmentation for these fixations is shown in the second row. The red rectangle around the object in the first row is the user input for the GrabCut algorithm[71]. The segmentation output of the iterative GrabCut algorithm is shown in the third row. The last row contains the output of normalized cut algorithm with the region boundary of our segmentation overlaid on it.

In Fig. 4.2, we provide a visual comparison between the output of our proposed segmentation, the interactive GrabCut algorithm[71], and Normalized Cut[78] for some of the difficult images from the Berkeley Segmentation Database[54]. For normalized cut, the best parameter (between 5 to 20) for each image is manually selected and the corresponding segmentation is shown in the last row of Fig. 4.2.

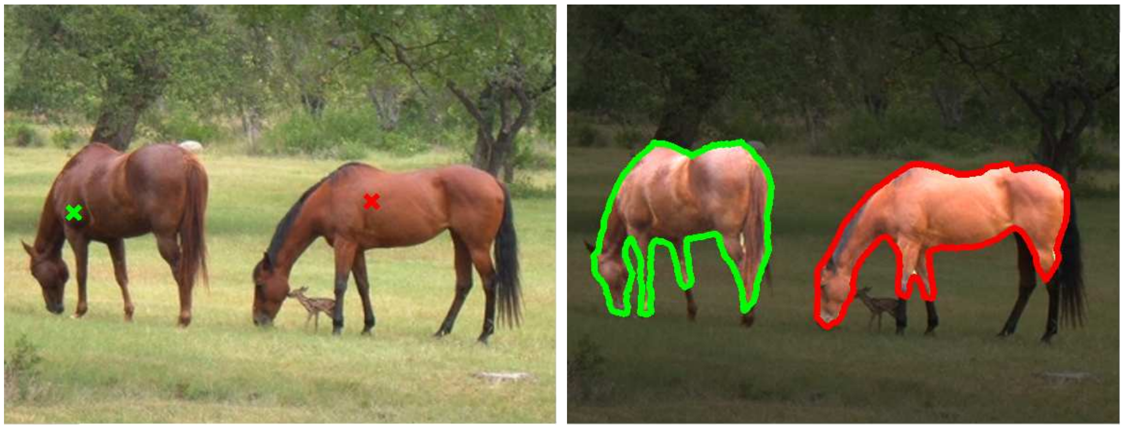


Figure 4.3: Left: an image with two fixations (the symbol "X"). Right: The corresponding segmentation for these fixations as given by the proposed framework.

4.2 Semantic significance: An empirical study

In the experiments so far, we have found that the proposed method segments the fixated region (or object) accurately and consistently, especially in the presence of both motion/binocular and static monocular cues. But, in the case of single images, only static monocular cues are present; a fixation on an object results in a segmentation which is mostly just a part of that object in the scene. What is interesting, however, is to study if there is a consistency in segmenting that part if we fixate at the same location inside that object as it appears in different images. In other words, we empirically study how semantically meaningful are the regions segmented by the proposed algorithm so that the algorithm can be used as a useful first step in the object recognition process.

For this study, we are going to use ETHZShape database [30] containing 255 images of five different objects namely Giraffes, Swans, Bottles, Mugs and Applelogos.

As, in this experiment, the final probabilistic boundary edge detector is calculated using static monocular cues only, the fixation location plays an important role in deciding what we get as the segmentation output. For instance, fixation on the neck of a giraffe would result in the segmentation of its neck. If all the internal texture edges were suppressed using say binocular cues, fixating anywhere on the giraffe would lead to the segmentation of the entire giraffe. Thus, it is important to choose the same fixation location inside the object, so that the variation due to changes in fixation location can be discounted for.

We need to make sure that we fixate at all the different parts of an object for this object. We avoid selecting these fixations manually as our selection fixation would be biased heavily by individual preference. Instead, we are going to use the shape of the object to find the salient locations to fixate and the segmented regions for these fixations are then manually labeled as a part if it appears frequently. This way, the parts are created from the low-level information and are only labeled by human subjects.

The question now is what are those salient locations to fixate and whether those fixations will be at similar locations inside the object across different instances of that object in the database? We hand segment the object in each image (we randomly select one in the image with multiple objects), and fixate at the corners and junctions in the skeleton of the binary object mask. The corners and junctions of the skeleton correspond to the prominent shape features of the object such as neck-body joint, leg-body joint etc. Although skeleton does vary, due to its susceptibility to noise, across different object instances, a corner or a junction corresponding to the major object features are still present. The noise adds extra redundant fixations that will only result in the repeated segmentations of the same region.

We fixate at all the salient locations on the objects, and collect the segmented regions corresponding to these fixations. Then, we examine these segmentations manually and labeled the segmented regions an object part if it appears repeatedly fixations at given similar locations on the object in most images in the database. See Fig. 4.4 for a sample of the parts of all five objects. Obviously, the number of parts for an object depends upon the complexity of its shape. Giraffe has the highest number of parts whereas Applelogos have the least.

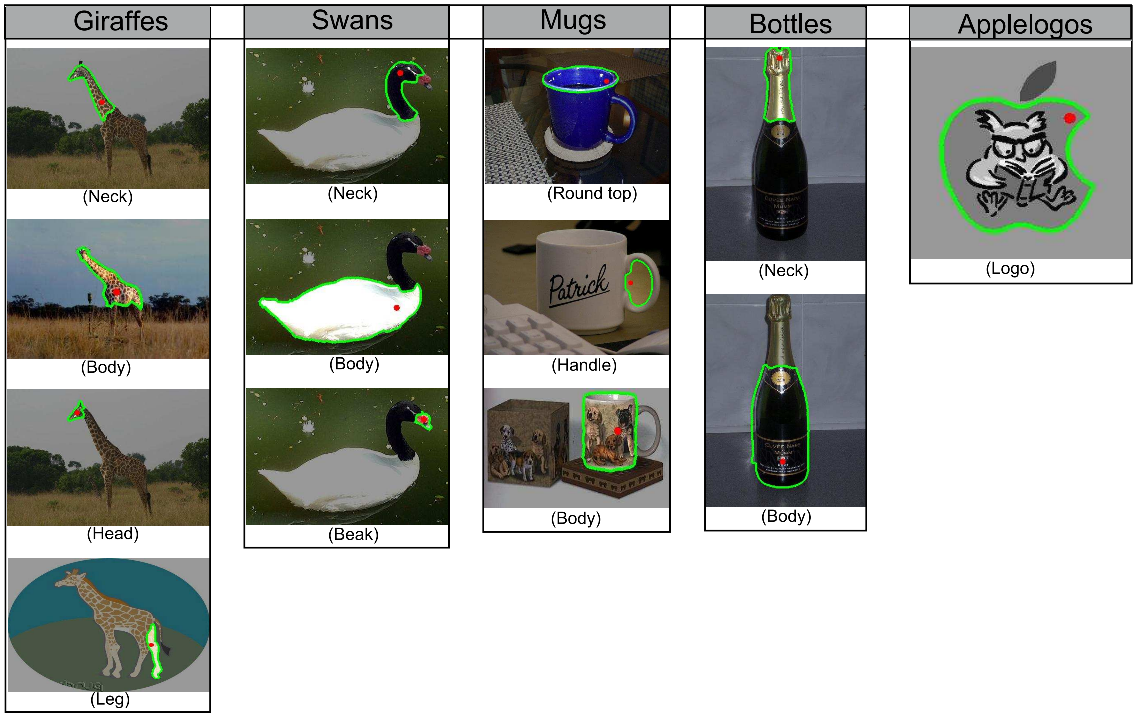


Figure 4.4: A sample of object parts. Red circular dot shows the fixation point and the green contour is the boundary of the segmented region for that fixation point. Giraffes, Swans, Mugs, Bottles, and Applelogos are found to have four, three, three, two and one part(s) respectively.

For each object, we count the number of times an object part is fixated and what percentage of the total number of fixations resulted in the segmentation of that part or the entire object in the case of compact object. We also gather similar statistics for semantically meaningless parts. These numbers are shown in the corresponding row of the table for that object. See Table 4.3. Different parts have different likelihood of being segmented on being fixated. But, some parts like the handle of a Mug, the Applelogo, the neck of a Swan etc. have high likelihood of being segmented on being fixated.

Another important statistic of interest is how often one of the fixations on the object result in the segmentation of the entire object. For that, we calculate the overlap of the biggest segmented region for each object with the hand-segmented object mask. We calculate the mean of the overlap of the biggest region over all images in the database. See table 4.4. Likelihood of segmenting an entire object is dependent upon how textured the object is. The Applelogos are segmented entirely by a single fixation, whereas bottles mostly have labels on them and are generally only segmented as its upper or lower half.

4.3 Fixation Strategy

The proposed method clearly depends on the fixation point and thus it is important to select the fixations automatically. Fixation selection is a mechanism that depends on the underlying task as well as other senses (like sound). In the absence of these cues, one has to concentrate on generic visual solutions. There is a significant amount of research done on the topic of visual attention [67, 92, 80, 99, 43, 75, 20] primarily to find the salient locations in the scene where the human eye may fixate. For our segmentation framework as the next section shows, the fixation just needs to be inside the objects in the scene. As long as this is true, the correct segmentation will be obtained. Fixation points amount to features in the scene and the recent literature on features comes in handy [49, 57]. See chapter 6 for strategies to generate

Table 4.3: shows the rate of detection of an object part given a fixation that lies on it. The first column of each table shows the name of the fixated part. The first row contains the name of the segmented regions as labeled by the human subject. The last column stands for the semantically meaningless segmented regions. Each entry (i,j) of the table is the percentage of total fixations on the part i that resulted in the segmentation of part j.

(a) **Mugs**

	Round top	Handle	Body	whole Obj	Non-parts
Round Top	90.00	0.00	0.00	0.00	10.00
Handle	98.9	0.00	0.00	0.00	1.1
Body	0.00	0.00	14.28	47.40	38.31

(b) **Giraffes**

	Head	Neck	Leg	Body	Whole Obj	Non-parts
Head	71.66	0.00	0.00	0.00	16.67	11.67
Neck	0.00	75.86	0.00	0.00	6.87	10.34
Leg	0.00	0.00	35.00	0.00	3.57	61.40
Body	0.00	0.00	0.00	26.92	25.0	48.07

(c) **Swans**

	Neck	Body	Beak	Whole	Non-parts
Neck	55.22	0	0.00	26.8	17.91
Body	0.00	28.91	0.00	37.34	33.73
Beak	0.00	0.00	87.5	0.00	12.50

(d) **Bottles**

	Neck	Body	Whole Obj	Non-parts
Neck	44.61	0.00	18.46	36.92
Body	0.00	28.16	16.90	54.92

(e) **Applelogos**

	Applelogos	Non parts
Body	91.8	8.2

Object name	**Mean overlap**
Giraffes	71.47
Swans	86.85
Mugs	84.13
Applelogos	95.13
Bottles	52.96

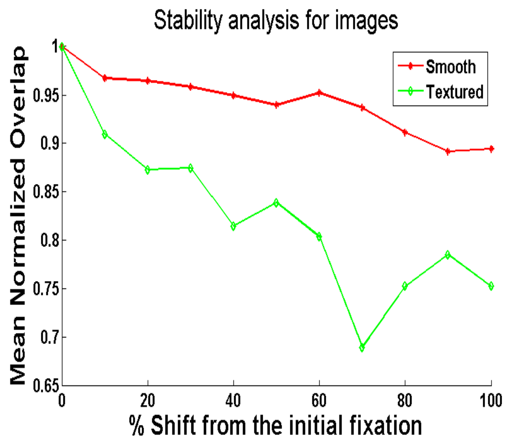
Table 4.4: The mean of the highest overlap (×100) for each image.

the potential fixations that lie inside most of the objects in a scene.

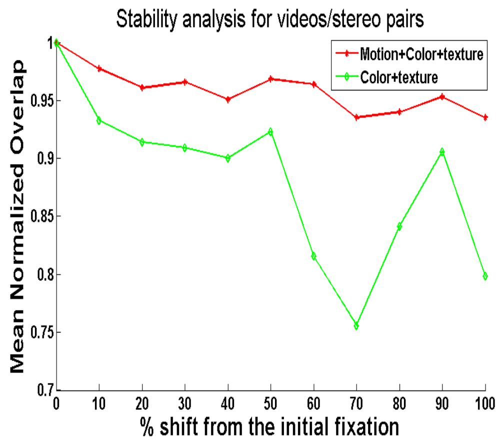
4.3.1 Stability Analysis

Here, we verify our claim that the optimal closed boundary returned by our algorithm for any fixation inside a region remains the same. The possible variation in the segmentation will occur due to the presence of bright internal edges in the probabilistic boundary edge map. To evaluate the stability of segmentation with respect to the location of fixation inside the object, we devise the following procedure: Choose a fixation roughly at the center of the object and calculate the optimal closed boundary enclosing the segmented region. Calculate the average scale, S_{avg}, of the segmented region as $\sqrt{Area/\pi}$. Now, the new fixation is chosen by moving away from the original fixation in random direction by $n \cdot S_{avg}$ where $n = \{0.1, 0.2, 0.3, ..., 1\}$. If the new fixation lies outside the original segmentation, a new direction is chosen for the same radial shift until the new fixation lies inside the original segmentation. The overlap between the segmentation with respect to the new fixation, R_n, and the original segmentation, R_o, is given by $\frac{|R_o \cap R_n|}{|R_o \cup R_n|}$.

We calculated the overlap values for 100 textured regions and 100 smooth regions from the BSD and Alpert Segmentation Database. It is clear from the graph Fig.4.5a that the overlap values are better for the smooth regions than for the textured regions. Textured regions might have strong internal edges making it possible for the original optimal path to be modified as the fixation moves to a new location. However, for the smooth regions, there is a stable optimal path around various fixations, it does not change dramatically as the fixation moves to a new location. We also calculate the overlap values for the 100 frames from the video sequences; first with their boundary edge map given by [53] and then using the enhanced boundary edge map after combining motion cues. The results are shown in Fig.4.5b. We can see that the segmentation becomes stable as motion cues suppress the internal edges and reinforce the boundary edge pixels in the boundary edge map [53].



(a)



(b)

Figure 4.5: Stability Analysis of region segmentation with respect to the fixation locations.

Chapter 5

Motion Segmentation Without Optical Flow

5.1 Introduction

Segmenting a scene into regions is a grouping problem that involves comparing pixels with their neighbors using the values of the visual cues (e.g. color, intensity, flow or disparity) defined at their respective locations. In most image segmentation work, the comparison is made using only monocular cues (color/intensity and texture) whereas, in motion segmentation, the optical flow vectors defined at the pixel locations are used to compare them. While using motion cues (optical flow) in the same way as the monocular cues (color/intensity) to compare pixels seems logical, there is a fundamental difference in the way these cues are obtained and that suggests a complementary role for these cues. Hence, the possibility arises that these cues should be handled differently.

Notwithstanding the sensor noise, the monocular cues (color/intensity) are known accurately at every pixel in the image as they are just the output of the CMOS sensors in the camera. In contrast, the optical flow vectors are obtained in a computational step by finding correspondences between pixels in the consecutive frames of a motion sequence. These correspondences can be computed unambiguously only at a few locally salient points (e.g. corners) in the scene which means only a sparse set of pixels have a valid motion information. On the other hand, all pixels have a well defined, valid color and/or intensity information.

This presents a challenge for the motion segmentation approaches that rely on comparing every pair of pixels to arrive at a segmentation. The reason is that they have to be able to compare a pair of pixels with one having a valid optical flow vector and the other not. To solve this problem, motion segmentation methods resort to either analyzing only the pixels with valid motion information or using the valid optical flow vectors at the salient locations to generate the optical flow vectors at all other non-salient locations (e.g the pixels in the smooth regions of the image).

The motion segmentation algorithms [98, 109] based on the former solution output clusters of points with consistent 2D/3D motion. In order to get the dense segmentation of the region containing a cluster of points, some sort of post processing step is employed. The latter solution to generate dense optical flow map of the image is more popular and, in fact, it is one of the most researched topics in computer vision. Despite significant progress made in designing efficient algorithms to calculate the dense flow map, the algorithms still suffer a number of limitations: poor accuracy at depth boundaries, inability to handle large (non-differential) motion, and critical dependence on texture.

In light of all the limitations of the optical flow based segmentation methods, we felt motivated to look for an alternate view to the process of segmentation. A simple observation suggests that when humans look at a video or a dynamic scene, they fixate at one of the moving objects at a time and follow it. Traditionally, the problem of motion segmentation has been defined as finding all the moving objects at once. Perhaps this is too ambitious. Instead, we formulate the problem of motion segmentation in a way that incorporates attention. Specifically, we define the motion segmentation problem as the one of segmenting only a selected object, identified by a point inside it. An attentional mechanism can provide this point. In case of robotic vision, this point is the fixation point. In case of passive video processing, this is an image point selected by the algorithm.

The approach to segment only the fixated object at a time will eventually create the segmentation of the whole scene as different objects are fixated. In the new formulation the problem is simplified in two main ways: first, only the motion of the object of interest has to be estimated as opposed to finding the exact motion of all the pixels in the image (dense optical flow map); second, the sparse motion information from the locally salient locations, in combination with dense color and texture information, suffices to segment the object of interest.

While we still use the basic framework defined previously in chapter 3, *we do not use dense optical flow map to generate the probabilistic boundary edge map.* Instead, we propose a different approach to identify the boundary edges. We estimate the motion of the fixated object using an initial segmentation and then use the estimated motion to identify the interior of the object as a set of salient points moving with that point. A rough estimate of object mask created by these points is used to generate the probabilistic boundary edge map. Our contribution lies in the way we reliably estimate the motion of the object from the initial segmentation and the method to find locally salient points moving with the estimated motion model.

5.2 Related Work

Motion segmentation based on dense optical flow can be broadly classified into two groups: (a) 2D motion segmentation: Based on the optical flow values, pixels are grouped into regions by finding discontinuities in the flow map [84, 11, 64], fitting motion models to the pixels [63, 14, 21] or using the layer representation of the flow map [100]. The flow values are noisy at different places. So, probabilistic methods are used to first group the pixels with similar motion and re-estimate the flow values of these resultant groups [103]. (b) 3D motion segmentation: 3D approaches which identify clusters with consistent 3D motion [24, 90, 82, 60, 87, 113, 1] using a variety of techniques. Some techniques, such as [102] are based on alternate models of image formation. These additional constraints can be justified for domains such as aerial imagery. In this case, the planarity of the scene allows a registration process [94, 6, 104, 114], with any un-compensated regions corresponding to independent movement.

As the optical flow map for sparsely textured scene is noisy and less inaccurate, some segmentation algorithms [101] first segment the frames into regions and then merge the regions by comparing the overall flow of the two regions. The accuracy of these methods depends on the accuracy of the image segmentation step itself. To avoid the pitfalls of using dense optical flow especially when there is a large motion in the scene, some algorithms [91, 90, 106] first calculate the sparse correspondences between consecutive frames, and group them into different motion models. The pixels are then assigned to one of these models by minimizing the overall mismatch as these pixels move to the target location given by their corresponding motion model.

Recently, [93] proposes a method to use sparse correspondences to segment the foreground by labeling regions from the segmentation computed using a complex version of the normalized cut based segmentation approach [78]. While this looks similar in spirit with what we propose in this paper, there are many fundamental differences, for example :we do not compute correspondences for the salient points in the scene and hence do not have to make any assumption about the number of possible motions to group them; furthermore, in [93], segmented regions are labeled using the motion information to form the foreground. In contrast, our approach use motion information to only modify the probabilities of the edge pixels and we never have to commit to any intermediate segmentation. They are basically different techniques.

5.3 Our approach

Given a point inside a moving object, our objective is to generate the probabilistic boundary edge map of the scene that has strong (bright) edges along the object (or motion) boundary and weak (dim) edges inside the object. The resulting boundary edge map is then used to find the optimal closed contour around the point using [58]. This point is also called fixation in [58]. So, throughout this chapter, we will use the terms "point" and "fixation point" interchangeably.

The basic outline of our approach is as follows: We first segment the region containing the given point using [58] (Fig. 5.1(d)). The probabilistic boundary edge map used to carry out the segmentation is the output of the Berkeley edge detector [53] (Fig. 5.1(b)) that uses only color and texture information to find the region boundary. Then, the motion of the segmented region is estimated (section 5.3.1). Using that estimated motion, all the salient points that belong to the object are found (Fig. 5.1(c)). Finally, the probabilistic boundary edge map is generated using the rough object mask created out of Delaunay triangulation of the set of salient points belonging to the object as found in the previous step (Figs. 5.1(e), 5.1(f)).

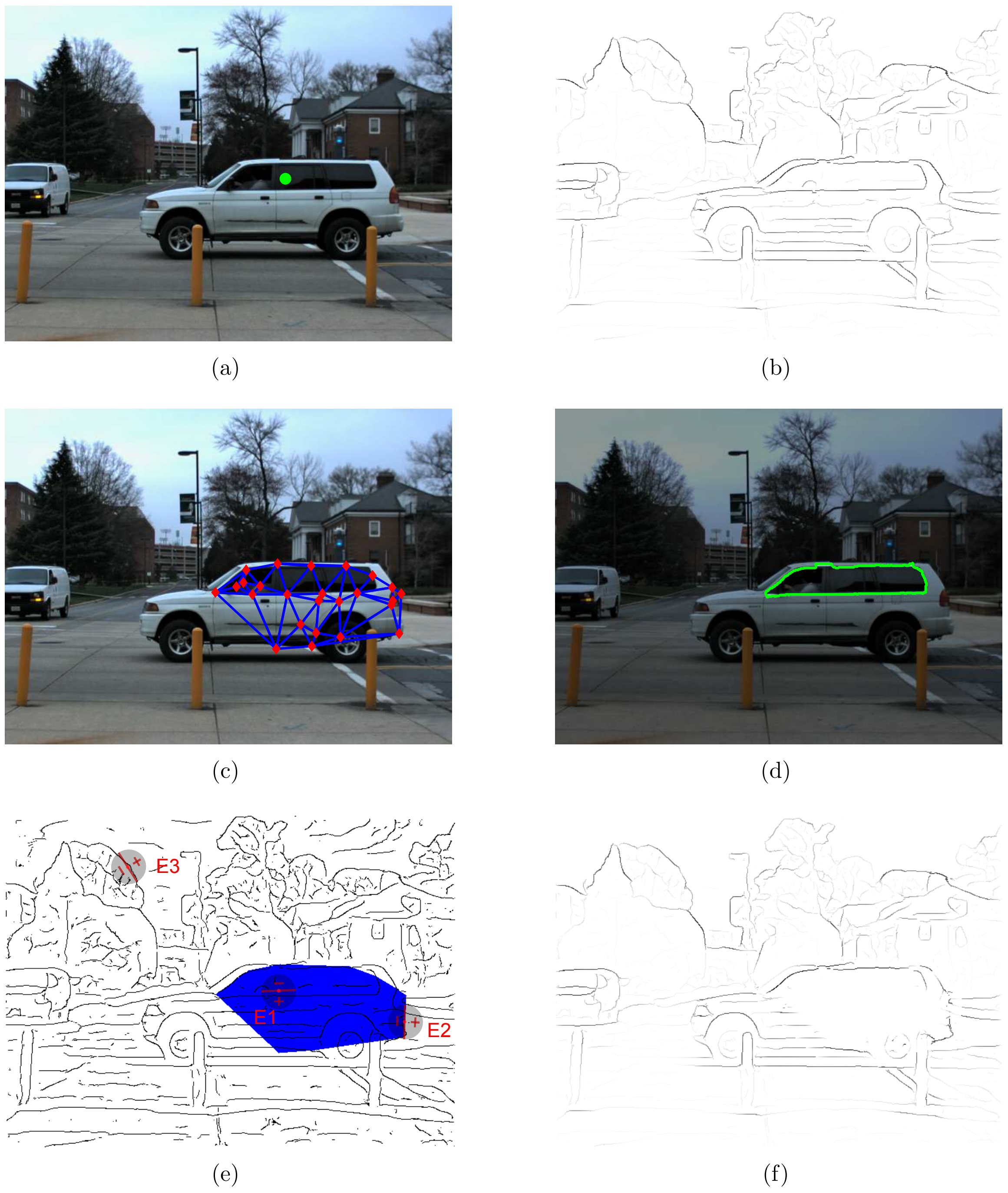


Figure 5.1: (a) A frame with the fixation point (the green dot). (b) The color and texture based probabilistic boundary edge map. (c) The salient points inside the object and the output of Delaunay triangulation. (d) Initial segmentation for the given fixation based on the color and texture based edge map. (e) The rough mask of the interior of the object overlaid on the edge map. The oriented discs of opposite polarity on two sides are used to detect the boundary edges and the internal edges resulting in the final probabilistic boundary edge map shown in (f).

5.3.1 Finding object motion

In this section, we start with the segmented region containing the given fixation point and want to estimate the motion of the object. If the entire region is fully contained inside the object, all the pixels in the region along with its closed boundary can be used to estimate the parameters of the motion model. However, as the initial segmentation is carried out using color and texture cues only, it is possible that only a part of the region and its boundary belong to the object of interest.

In fact, there are two such cases when this happens: In the first case, the pixels inside the segmented region belong to the object, but a part of the region boundary belongs to the occluding object in front of the object of interest. For instance, the boundary of the segmented region shown in Fig. 5.2(b) has such a part (the blue part along the boundary in Fig. 5.2(c)). In the second case, which arises due to weak or no intensity edge between object and the outside, a part of the region along with its boundary does not belong to the object of interest (see the red part of the region in Fig. 5.2(c)). In both cases, using the entire region and points along its boundary to estimate the motion of the region can cause a significant error. So, we need to find a way to break the region into parts such that at least one of the parts belong completely to the object of interest and thus can be used to generate the correct estimate of motion.

Breaking region into parts

Before we try to break the region to obtain parts, it is important to understand what should be the characteristic of a part: it should be distinctive enough so that the motion can be estimated accurately. Such points are the locations along the region boundary where a significant change in orientation happens abruptly, creating corner-like structures. A single such point is not sufficient to estimate the parameters of a

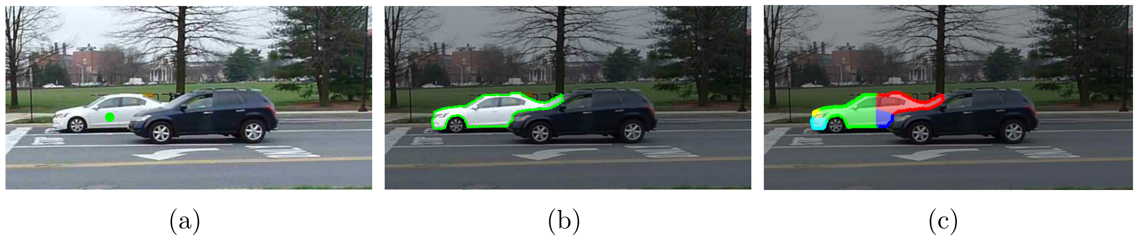


Figure 5.2: (a) A frame of a motion sequence with a given fixation (the green dot) on the white car. (b) The initial segmentation of the region using static monocular cue based edge map only. (c) The parts of the region extracted using the skeleton based method.

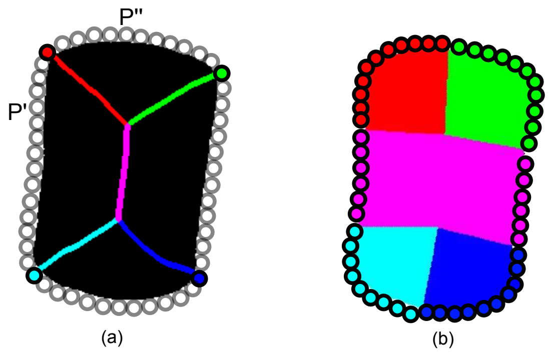


Figure 5.3: (a) The skeleton of the black region overlaid on it. The points along the boundary are shown as empty circles except the end points of the skeleton branches. (b) The region is split into parts, each corresponding to a branch in the skeleton. The part of the boundary associated with each region is filled with the same color.

motion model, so we need to associate points from inside the region with the contour fragment creating a corner like structure. The contour fragment together with the associated portion of the region will make a part. Now, the question is how do we find the fragment of the region creating a corner like structure?

The boundary of a region is a closed contour. If we extract the skeleton of the closed contour, the corner like structure along the region boundary would introduce branches in the skeleton. A skeleton has three types of branches[1]: one between an end point and a junction; between two junctions; and between two end points. In the last case, the entire skeleton has only one branch and thus do not break up the closed region into any parts. We are more interested in the skeletons with other two types of branches and those skeletons for most shapes. Having excluded the third type of the branch, we define a branch as a connected set of skeleton points between either two junctions or a junction and an end point. (See Fig. 5.3a)

Before explaining how the branches of a skeleton are used to break the corresponding closed region into parts, we are going to explain the algorithm to obtain the skeleton itself. Although there are different definitions of skeleton [44], the definition of a skeleton in this work is the Blum's medial axis of a closed region. The methods to extract the skeleton varies from simple morphological thinning process to finding ridges in the Euclidean distance function of a given mask. Although the methods have elegant formulations in the continuous space, their results in the discrete space of image are sensitive to variations along the contour of closed region which result in various spurious branches. The distance function based methods can localize the skeleton points but often change the topology of the shape of the closed region. So, in this work, we choose a robust method by Dimitrov et al. [26] which detect the

[1]We are not using the temporal aspect of when the branches of the skeleton are created as the process of medial axis transformation evolves. In [33], the timing of the branch creation is used to create shock graphs which represent the shape of the binary mask.

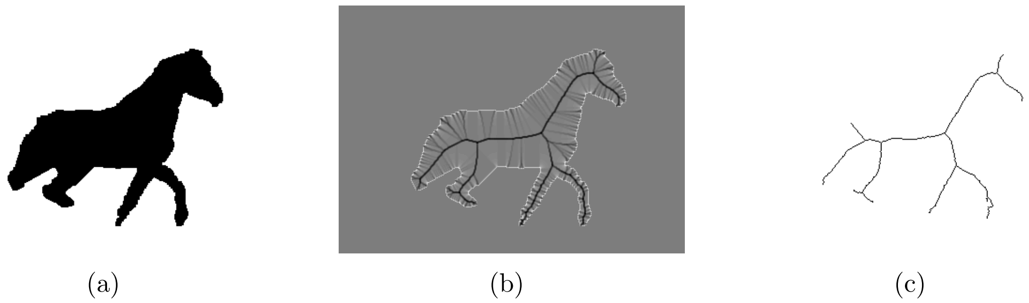


Figure 5.4: (a) A binary mask of a horse. (b) The total outward flux of the gradient vector field of the euclidian distance function. The sinks of the vector field (the dark pixels) are along the skeleton of the shape. The divergence is zero at the gray pixels. (c) The actual skeleton upon thinning the mask [26].

skeleton points as locations where the conservation principle of the net outward flux of the vector field is violated (See dark lines in Fig. 5.4(b). For instance, the skeleton extracted using this method is shown in Fig. 5.4. The end points of the extracted skeleton do not necessarily lie on the closed contour of the binary mask. So, we extend the end points in the post processing step such that they lie on the closed boundary of the mask.

A branch of the skeleton is associated with a specific fragment of the closed contour and that fragment can be extracted in the following way: The end points of the skeleton lie on the contour and break the contour into fragments. Now, let us consider the red branch of the skeleton shown in Fig. 5.3a. The points along the contour are shown by empty circles except for the end points of the branches. Starting from its end point (the red filled circle), move in the clockwise and counterclockwise direction along the contour till another end point, the green circle and the cyan circle respectively, is encountered. Among all the contour points between the red circle and the green circle, the closest point to the junction point of the red branch is found and is shown by symbol P''. Similarly, for the contour points between the red and the cyan circles, P' is such a point. Now, the connected and ordered set of contour points between

P' and P'' passing through the end point (the green circle) is the contour fragment associated with the red branch. (See Fig. 5.3b). Also, the part of the region enclosed by the contour points and the junction belong to that branch too (the red region is that portion). See Fig. 5.3b for the parts of the contour and the region associated with each of the skeleton branches.

Once we break the initial segmentation into parts, we assume that at least one of these parts belongs completely to the object of interest. Even when it is not true, the number of points from outside the object in the part would be far less than what would be in the total area of the region. So, the motion is estimated for each part and its consistency is verified for all the pixels on the region including its boundary. The motion estimate being conformed to by the highest number of pixels in the region is selected as the motion model of the object itself.

Estimating motion parameter

The objective here is to estimate the motion model M that takes the pixels of a region part R to the corresponding location in the next frame. Although M can be any warping function, we are going to assume an affine motion between two consecutive frames. Then, the parameters of the transformation M is a six-dimensional vector, $\mathbf{a} = \{a_1, a_2, a_3, a_4, a_5, a_6\}^T$. Let us denote the set of points along the part of the region boundary and the edge pixels from inside the region that belongs to the part R by O where $O = \{o_i\}_{i=1}^{N}, o_i = (x_i^o, y_i^o)$ and N is the total number data points in the set. The transformation M is defined as:

$$M(a; o_i) = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \begin{bmatrix} x_i^o \\ y_i^o \end{bmatrix} + \begin{bmatrix} a_5 \\ a_6 \end{bmatrix}$$

Let us also denote the set of all edge pixels in the next frame by $D = \{d_i\}_{i=1}^{N_d}, d_i =$

(x_i^d, y_i^d) where N_d is the total number of edge pixels in the next frame. We need to find out the affine transformation M that best aligns the data points in O to the subset of points in D. To estimate the parameter of the affine transformation, we use the iterative closest point algorithm [9] which is a two step process:

C. For each data point $o_i \in O$, we find the corresponding point $d_{\phi(i)}$ where $\phi(i) = argmin_{j \in \{1,..N_d\}} \epsilon^2(|d_j - M(a_k; o_i)|)$ and $\epsilon^2(|x|) = ||x||^2$.

E. Estimate the transformation using these correspondences:

$$a_{k+1} = argmin_{\mathbf{a}} \sum_{i=1}^{N_c} \epsilon^2(|d_{\phi(i)} - M(a; o_i)|)$$

The iteration is stopped when no new correspondences is established, meaning $a_{k+1} = a_k$. $a_0 = \{0, 0, 0, 0, t_x, t_y\}$ as the initial estimate of the motion parameter where t_x and t_y are initial estimate of the translation motion of the region calculated using phase correlation.

5.3.2 Finding salient points on the object

Using the motion model M estimated in the previous step, we find all the points in the image moving with M. These points will constitute tentatively the inside of the object. But, as discussed previously, to verify the motion of a pixel, it has to be locally salient. So, we will only be able to decide for the salient points in the image whether or not they have moved with the motion model M. Another important point is that we are interested not only in verifying if a salient point has moved with M but also in finding out whether that salient truly belongs to the object of interest or lies outside of the object moving with the expected motion model M.

We use SIFT [48] to detect locally salient points in the current and next frames.

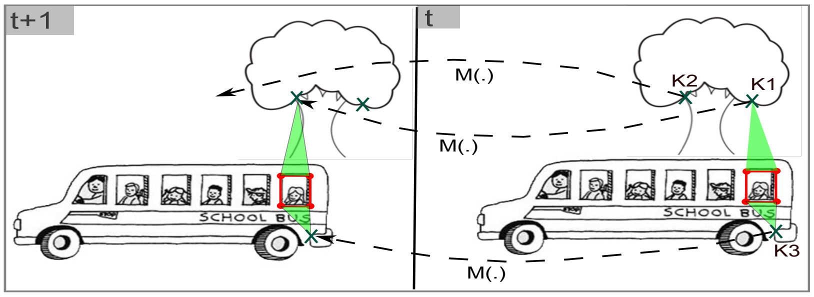


Figure 5.5: The two consecutive frames of a video containing an object (the bus) moving with respect to the other object (the tree). M is the estimated affine motion of the window region inside the bus enclosed by the red contour. $K1$,$K2$,$K3$ are three locally salient points in the current frame t.

The only criteria while choosing any keypoint detector to find the salient locations is that it should have high repeatability score. Now, the first round of elimination of possible candidates is done by moving a salient point p in the current frame with M and checking if there is a salient point detected at the expected location $M(p)$ in the next frame. The salient points not satisfying this check are eliminated from the set of possible salient points on the object. To allow for some error in motion estimation and in localization of features by the keypoint detector, we, in fact, check in a small neighborhood around $M(p)$ for any detected salient point. (In Fig. 5.5, the salient point $K2$ does not have a corresponding salient point in the next frame whereas $K1$ and $K3$ map to salient locations in the next frame. This eliminates $K2$ from consideration). Now, in the next step, we need to verify for those mapped (e.g. $K1$ and $K3$ in Fig. 5.5) to a salient point whether or not the matching salient point is indeed its corresponding location in the next frame.

In order to do that, we associate with each salient point a triangular patch Δ that is formed by connecting the salient point with the two closest points in the set $O' \in O$ where O' is a fixed number of data points sampled from the set of edge points, O, belonging to the part of the region with most consistent motion as explained previously. In Fig. 5.5, say, the four corners of the region boundary are in

O'. For $K1$ and $K3$ the patch is constructed and shown in green color. We can see the content inside the patch matches perfectly for $K1$ but not for $K3$, meaning $K1$ is on the object while $K3$ is outside of the object. More precisely, if for all the pixel $q \in \Delta, ||G^1(q) - G^2(M(q))|| < T$, then the image patch Δ matches to its projected image in the next frame, confirming that the salient point this patch belongs to is inside the object. G^1 and G^2 are the gradient magnitude maps of the frame 1 and 2 respectively.

5.3.3 Generating probabilistic boundary edge map

Finally, we want to use the salient points found to be on the object to create a rough mask of the interior of the object and then use that mask to eliminate internal edges and reinforce edges along the boundary of the object. The mask is created by Delaunay triangulation of the set of salient points and the points in the set O'. A typical output binary mask overlaid on the edge map is shown in Fig. 5.1e.

Now, the edges that lie inside the mask is eliminated whereas the edges at the border of the mask should be reinforced as they are at the object boundary. Now, to estimate how likely an edge pixel is at a depth boundary can be done by putting a disc with equal and opposite weights on its both sides at an edge pixel and calculate the response by convolving the binary mask with the disc as a 2D filter(See Fig. 5.1e). The responses at different edge pixels can then be assigned to them as their new boundary probability. The magnitude of the response is the highest at the object boundary and is the lowest inside and outside of the object (See $E2$ and $E1$ in Fig. 5.1(e) respectively). While it is clear that the response will be higher at the boundary edge pixels, a zero response does not always mean that the edge pixel lies inside the object. In fact, both for the edge pixels outside of the object and inside the object, the filter response will be close to zero. To differentiate between these two types, we use another circular disc with same weight everywhere and convolve the mask to collect the response. Now, for the pixels that are truly inside the mask will have high response whereas the points outside the mask will have low response. The response of convolving with the full disc filter and of the disc filter with opposite polarity can locate the internal edges accurately such that the are assigned zero probability while the probabilities of the pixels lying outside the mask remain unchanged as we are not certain whether the mask spans the entire object. (See $E3$ in Fig. 5.1(e)).

5.4 Experiments and Results

With a moving camera, we recorded a large number of videos of moving objects like cars, buses, and motorbikes on the road under different scenarios: fast moving objects, multiple objects, occluding objects and objects with low texture. Fig. 5.8 shows the output of our segmentation algorithm for some of these videos. The probabilistic boundary edge map is modified such that the edges that are weak (see the boundary of the girl's head in column 4, row 2 of Fig. 5.8) are made strong (column 4, row 4 of Fig. 5.8). More importantly, the strong internal edges are either removed or weakened (see the changes in the probabilistic boundary edge from row 2 to row 4).

Unlike traditional motion segmentation algorithms, our segmentation approach is not affected by texture, displacement and occlusion in any significant way. Regarding occlusion, as explained in section 5.3.1, the initial segmentation is broken into regions such that the occlusion does not affect motion estimation. Here,in the light of the above results, we are going to discuss about the other two factors: texture and displacement.

Effect of texture Texture is important because it provides the good initial correspondences to aid the dense flow estimation process. The motion segmentation algorithms report their best results for highly textured scenes. But, the accuracy drops in the presence of smooth regions as they do not have enough salient points to correctly estimate the flow vectors inside smooth regions. Our approach, on the other hand, does not try to establish correspondence between pixels but only verifies if a salient point has moved with the estimated motion or not. So, high or low amount of texture only affects the number of salient points found inside the object at the end of the process. For instance, the bus in column 2 of Fig. 5.8 has less texture on it than the girl in column 4 of Fig. 5.8 which translates into fewer salient points on the bus than on the girl even though the bus occupies for more area in the image than the girl.

Large Displacement As the proposed segmentation method relies on estimating motion parameter of the region as opposed to a pixel, large displacement is easily handled. The initial estimate of a shift in the region is obtained using phase correlation method which is then used to initialize the process of estimating affine parameters such that the actual solution lies close to the initial estimate.

5.4.1 Converging and Diverging Flow

One of the serious problems for the approaches based on using optical flow estimates to group pixels together arises when the flow is diverging or converging depending upon whether the object is moving towards or away from the camera respectively. The flow vectors for all the points inside the object are different and the pixels cannot be grouped on the basis of having similarity. See Fig. 5.6.

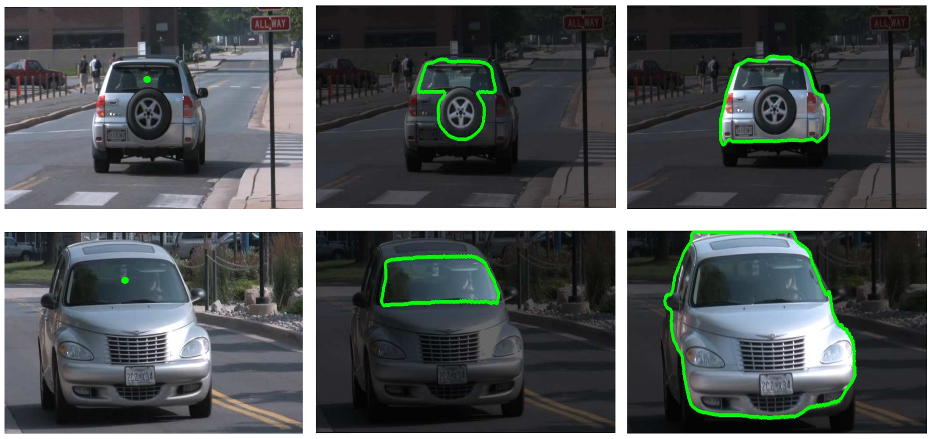


Figure 5.6: The car shown in the first row is moving away whereas the car shown in the second row is moving towards the camera. The initial color and texture based segmentation for the fixation points, shown in the first column, is just a part of the car (second column). The final segmentation is shown in the last column.

5.4.2 Effect of fixation location

As an object can have different possible sub regions contained in them, depending upon the location of fixation inside the object, a different initial segmentation is found. But, it does not affect the final outcome as the same motion parameter can be estimated using any of these initial segmentation which can then be used to segment the entire object. See Fig. 5.7 for an example, where different fixation locations inside the bus in the video captured using moving camera, although having different initial segmentation (see the middle row), result in the same final segmentation (the last row).

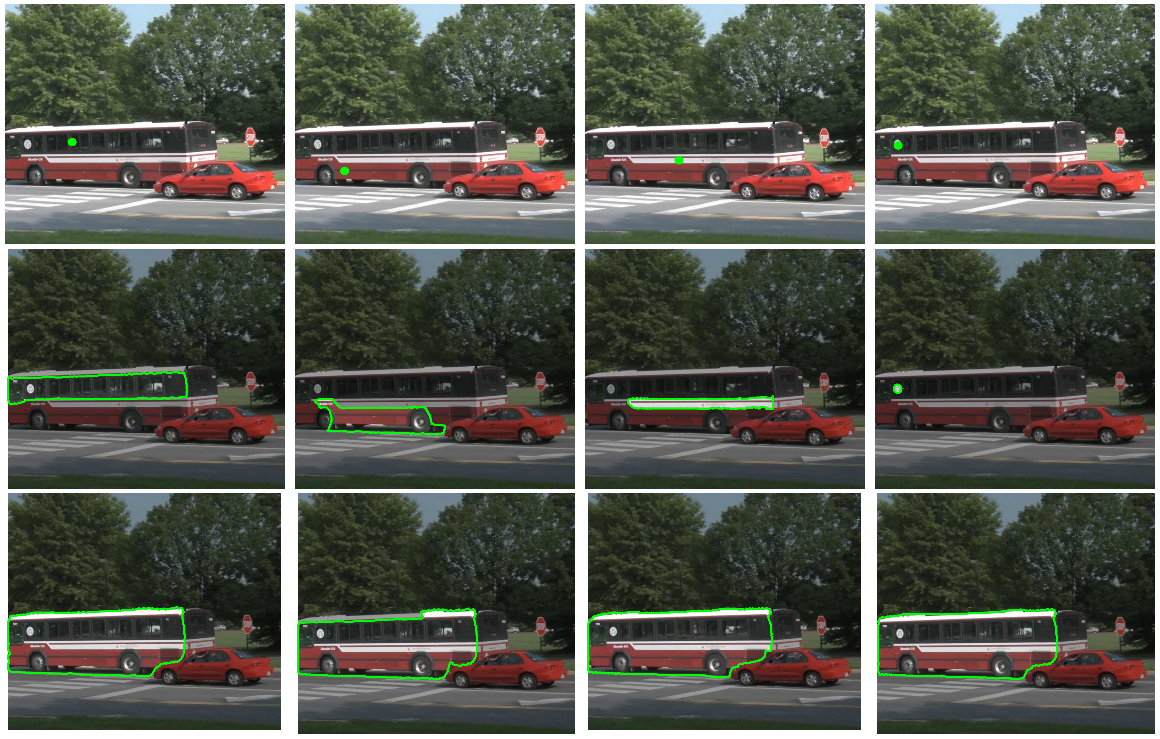


Figure 5.7: Four fixations at different locations inside the same object, shown in the first row, result in different initial segmentations (the second row) but the same final segmentations after incorporating motion information(the final row).

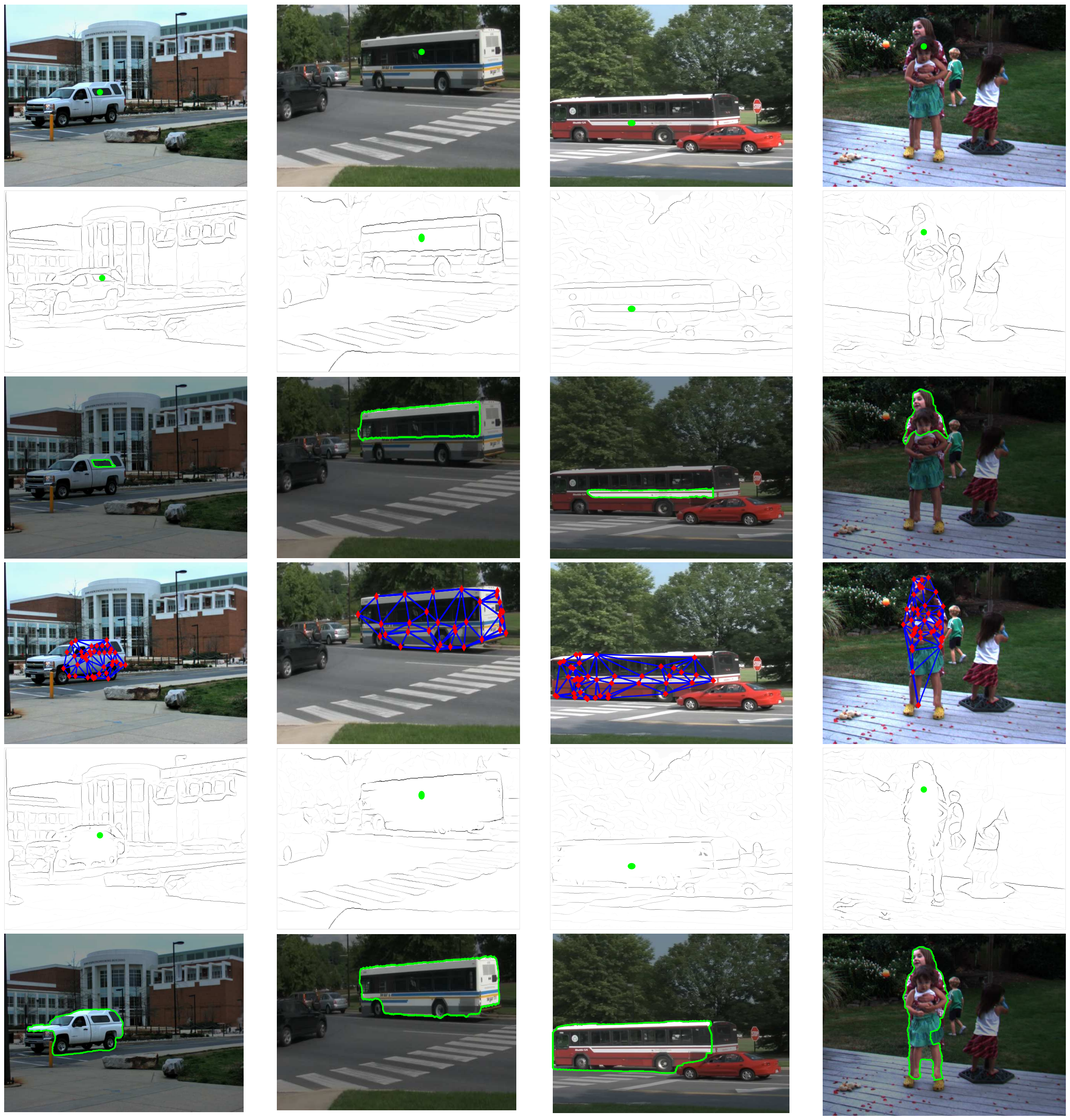


Figure 5.8: Row 1: A frame of a video and the given fixation point (the green dot) on the object. Row 2: Probabilistic boundary edge map (PBEM) based on color and texture. Row 3: Segmentation using the color-based PBEM. Row 4: The salient points (the red diamonds) inside the object found using the motion analysis. The mesh (indicated by blue lines) creates the rough estimate of the inside. Row 5: The final probabilistic boundary edge map after improving it with the estimated object mask.

Chapter 6

Fixation selection

6.1 Introduction

Attention, which is an integral part of the human visual system, are of two types of attention: covert and overt attention. The basic difference between these two types of attention is in the physical aspect of how the human visual system achieves them. The former requires the visual system to attend a certain visual stimulus in the scene without making any eye movement whereas the latter requires the physical movement of the eyes such that the visual stimulus ("fixation point") is brought to the center of the retina. It has been established that the both kinds of visual attention play a critical role in our visual perception. Although in psychophysical experiments the human subjects can attend a stimulus only covertly without actually fixating on it, in a real world, covert attention is generally followed by an overt attention to the target fixation point in the scene.

In this work, we use the output of an overt attention module as "fixation", which are basically the "important" locations in the scene. While it is hard to know the underlying process of human visual attention, there are a number of computation models that try to find the salient locations [42, 19, 37, 67, 92, 80, 75] by first extracting features from the scene and then using the extracted features to calculate a saliency map of the scene [37], which has hight intensity value for the salient locations and low for the rest of the scene.

The computational models of visual attention systems [42, 19, 37] generally involve extracting features from low-level cues such as color, brightness, texture, and motion. Motion cue has been found to be one the strongest cue that attracts human attention. Keeping that in mind, we have decided to propose two different strategies to select fixation points depending on whether the scene is dynamic or static.

6.2 Static scene

First, in the case of images (static scenes), we are going to use the visual attention method proposed by Harel et al. [37] to obtain the saliency map of the scene[1]. We observed that the saliency map has blob like features (See Fig. 6.1b), whose centroids can be the fixation points for our algorithm. We threshold the saliency map to create a binary mask such that the locally salient regions becomes a connected components in the binary mask. (The pixel intensity of the saliency map ranges from 0 to 1 and we use 0.5 as our threshold.)

The centroids of the connected components in the resulting binary mask are used as fixations (Fig. 6.1c). The segmentation given by our algorithm for each of the fixations for an example in Fig. 6.1a is shown in Fig. 6.2. It is to note that the segmented regions vary significantly in its scale showing the effectiveness of our algorithm to segment a region irrespective of its scale.

[1]Any other computation model could be used too. The basic reason to use [37] is speed.

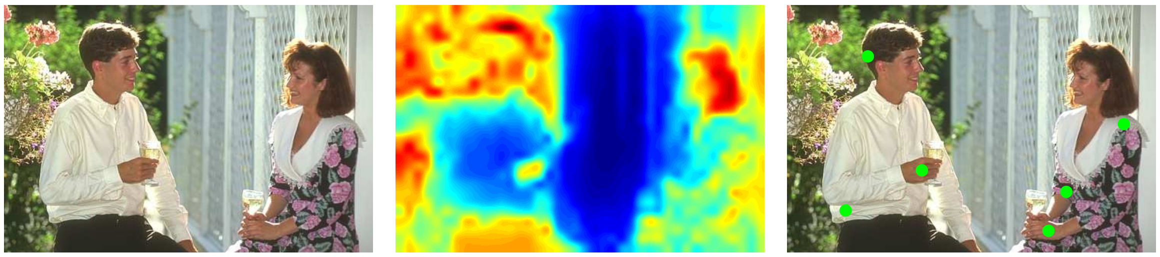


Figure 6.1: (a) RGB image. (b) Saliency Map. (c) Local maxima.

6.3 Dynamic Scene

Now, in the case of a dynamic scene, the objects are moving. To segment these objects, the attention strategy should find a point anywhere inside that object. We present a simple algorithm to find fixation points that are inside the independently moving objects in the dynamic scene.

Now, if the camera recording the video is static or in the case of a static robot, finding a point on the independently moving object is easy: track features and select any moving feature point as fixation. But, if the camera is moving, then flow in the scene is due to both the camera motion and the actual independent motion of the object. So, our strategy in such cases is to estimate camera motion and remove it from the scene such that remaining motion can be safely assumed to be only due to independently moving objects. Such an assumption is not limiting because our objective is only to locate a point on the object but not to use these flow estimates to segment the object.

Our strategy to select fixation point in the case of a dynamic scene is as follows:

1. Using phase correlation on two frames in the Cartesian representation (to find 2D translation parallel to the image plane) and in the log-polar representation (to find scale and rotation around the optical axis), we obtain a four-parameter transformation between frames (see [69]). Phase correlation can be thought

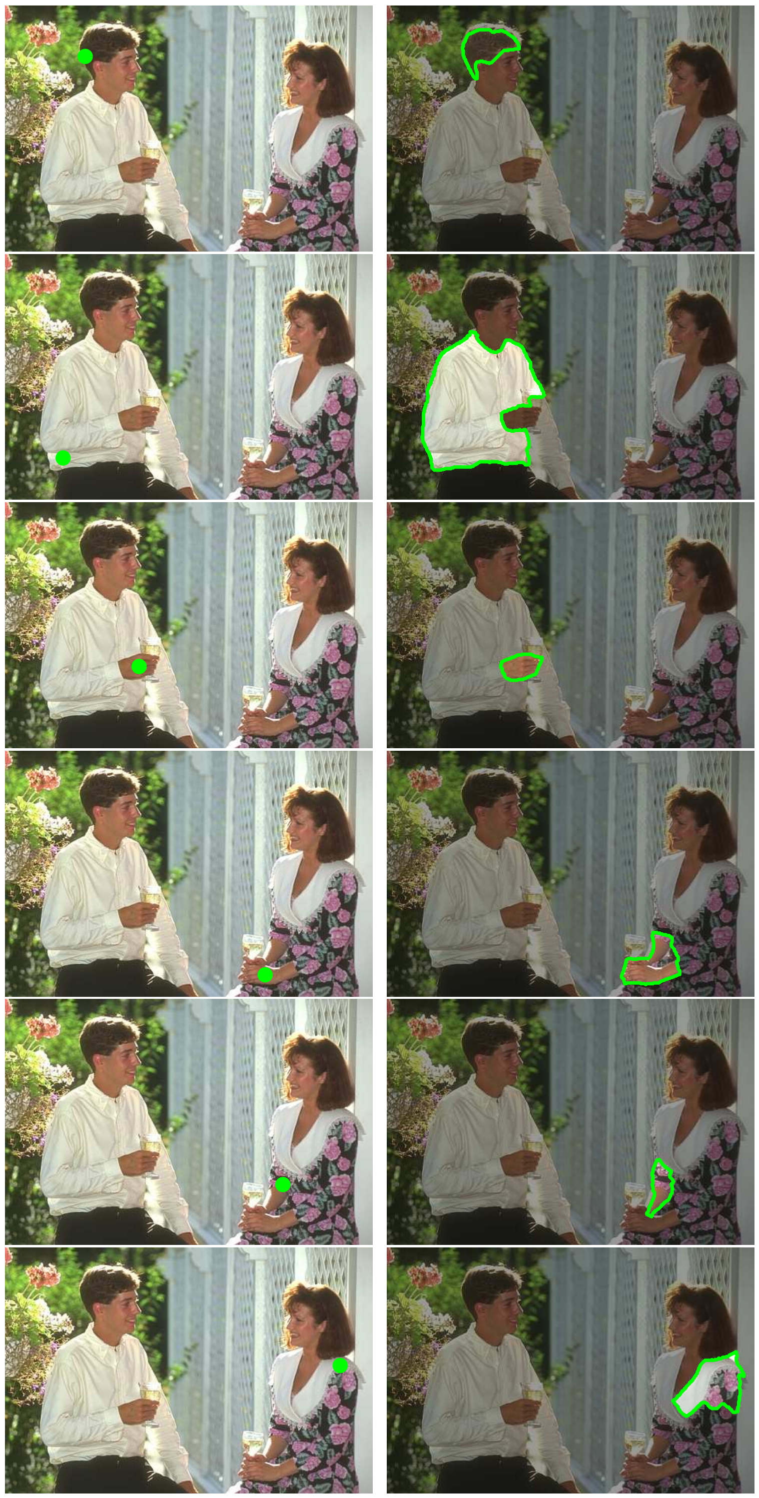
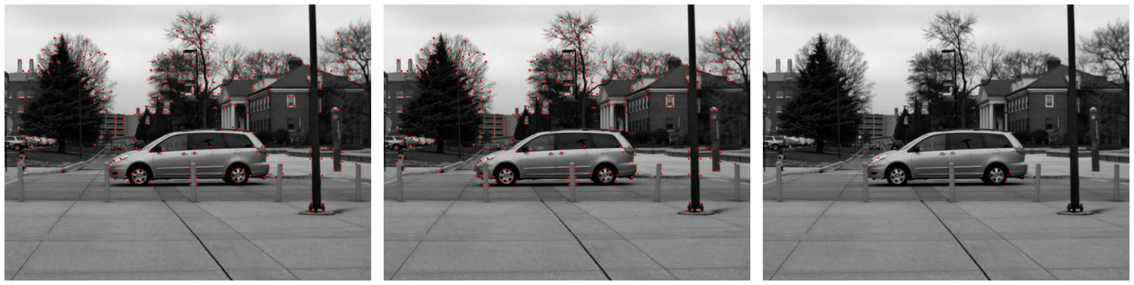


Figure 6.2: Each row contains the original image with fixation point (the green dot) and the corresponding segmentation in column 1 and 2 respectively.



(a) (b) (c)

Figure 6.3: (a) Feature points (red dots) in frame 1. (b) Tracked feature points (red dots) in frame 2. (c) A group of independently moving feature points (red dots).

of as a voting approach [31], and hence we find empirically that these four parameters depend primarily on the background motion even in the presence of moving objects. This assumption is true as long as the background edges dominate the edges on the moving objects. This four-parameter transform predicts a flow direction at every point in the image.

2. We select a set of points S in the image and find sparse correspondence using the tracking algorithm in [79]. We then compare the direction of the estimated correspondence with the estimated flow direction predicted by the phase correlation, and categorize the points in S into two groups: the ones with a difference in angle less than η degrees which are possible background candidate pixels, and those with a difference larger than η degrees, which are possible foreground candidate pixels.

Finally we cluster the points, and chose our fixation point within the cluster of points considered as possible foreground. (See Fig. 6.3)

6.4 Are fixations independent?

In this thesis so far, we have proposed a novel method to segment a fixated region in the image. But, for every fixation, the process of segmentation is carried out independently. So, an obvious question to ask is whether the segmentation process for a given fixation be influenced by the segmentation for the previous fixations? In other words, does the segmentation of some objects need more than one fixation?

Alfred L. Yarbus [110] has shown that the fixation path taken by a human subject changes depending on the high level task they are asked to perform. However, it has not been shown whether it is true even when a human subject observes a single object and makes more fixations inside a complex object than a simple object to say perceive the shape of the object.

Inspired by the possibility to connect the fixations even at the level segmentation, we performed numerous experiments and rigorous analysis of the segmentation results for more than 2000 images and came up with two cases wherein multiple related fixations are used to achieve the final goal of segmentation. The first case is segmenting objects with thin elongated shape. The reason is explained in detail in chapter 7. The other case is when we try to segment complex objects with many individual parts. Here, the basic idea is to use fixations to segment individual parts and then use the relationship such as relative scale, orientation and location between the parts detected and the part being detected to segment the currently fixated part more efficiently. We provide a preliminary study on this process in chapter 8

Chapter 7

Segmentation with multiple fixations

7.1 Relationship between fixation and segmentation

When the fixation point lies inside a homogeneous region with no strong internal textures, the exact location of the fixation with respect to the region boundary does not affect the segmentation result. It will be the same closed contour for any fixation point inside the region. However, there are scenarios when change in fixation inside of the region changes the segmentation output. It happens generally when only static monocular cues are used to generate the probabilistic boundary edge map as it leaves strong internal edges in the edge map. There are essentially three such scenarios: 1) when smaller regions are fully contained in the original region (or object); 2) in the presence of dominant internal textures and complex lighting effects; 3) when the fixated region (or object) are extremely concave and has long and thin structures.

Case 1: closed regions inside the object

Such objects (e.g a face) has smaller objects (e.g eyes and mouth) contained fully inside of it. Given the probabilistic boundary edge map (see Fig. 7.1), fixations on the smaller regions (or objects) result in the segmentation of those regions as shown in Fig. 7.1. It is intuitive to see that fixating on eyes and mouth should make the visual system see those parts of the face whereas fixation anywhere else on the face should make the entire face more perceptible. So, such variation in the segmentation with the changing fixation locations is desirable, and makes the proposed algorithm closer to how the human visual system might look at objects like faces. If, however, stereo or motion cues were used, and there is no non-rigid motion of the facial features, the internal edges on the face corresponding to the eyes and the lips would vanish and all fixations on the face would result in the same segmentation, the entire face.

But, such a probabilistic boundary edge map with strong and valid internal edges can be generated even in the presence of motion or stereo cues. For instance, consider the person, whose face we considered above, is laughing even as he moves his face. In that case, the edges along the mouth have different flow across them, making them strong boundary edges. So, the final probabilistic edge map will have strong internal edges corresponding to the boundaries of the mouth and obviously the boundary contour of the face. (Such a probabilistic edge map would be akin to the one with the static monocular cues only.). Now, once again, fixating on the mouth will segment that mouth whereas fixating anywhere else on the face outside of the mouth will give us the entire face, similar to what happened in the face example stated above. In these circumstances, not getting the same closed contour for all the fixation points inside of a contour is justified.

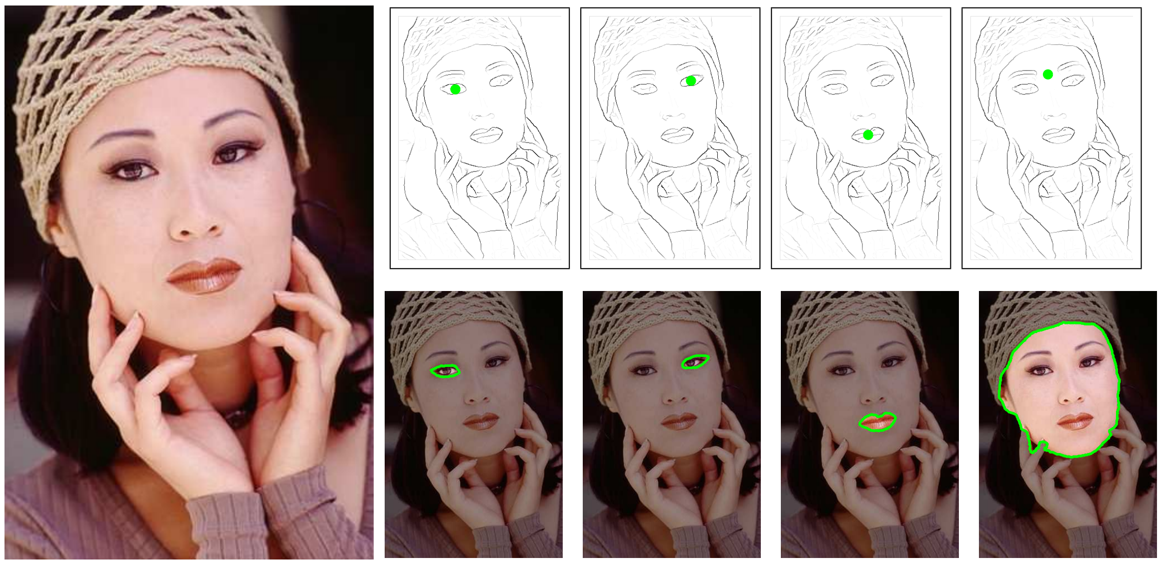


Figure 7.1: The fixations, indicated by the green circular dots, on the different parts of the face are shown overlaid on the inverse probabilistic edge map of the leftmost image. The segmentation corresponding to every fixation as given by the proposed algorithm is shown right below the edge map with the fixation.

Case 2: Texture and complex lighting effects

This case arises when we process single image only, meaning that there is no binocular or motion cues to remove the internal edges from the edge map. Although Malik et al. [53] can handle homogeneous textures using textons, non-homogeneous textures are hard to tackle and it creates spurious internal edges and disappearance of some boundary edges. Another factor contributing significant spurious internal edges is complex lighting effects on the object. See Fig. 7.2, an image of a crocodile in the wild. Its probabilistic boundary edge map clearly shows how these two factors have given rise to spurious internal and weak boundary edges, causing significant variation in the segmentation as the fixation shifts from one location to another on the body of the crocodile. Such variation in segmentation with fixation is not desirable, but it can only be fixed either using binocular and(or) motion cues as explained in section 3.2 or high level information shape information such as knowledge of how a crocodile looks

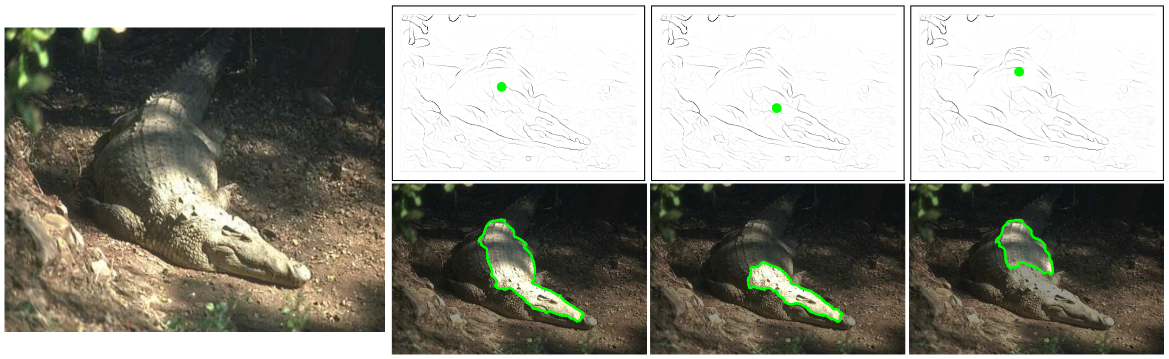


Figure 7.2: The densely textured crocodile image is shown on the left. The top row of images contain fixations at different locations on the crocodile overlaid on its inverse probabilistic boundary edge map while the bottom row of images contains the corresponding segmentation obtained using the proposed algorithm.

like and how it can deform its body.

Case 3: Concave shapes with thin structures

The location of fixation inside a concave region with thin elongated structures can affect the segmentation output as the thin structures get merged in the polar space due to fixed sampling along the angular axis. While converting the probabilistic boundary edge map from the Cartesian to the polar space is an important step of the proposed segmentation algorithm (section 3.3), it also causes a slight problem for shapes with thin structures and when the fixation lies sufficiently far away from these thin structures.

Let us understand why having a thin structure can change segmentation output with changes in the fixation location. Referring to Fig. 7.3, for the elongated part of the shape, the pair of points separated by a distance d and at a distance r away from the pole subtends an angle of θ (in radian) at the pole P such that $\theta \approx \frac{d}{r}$. If we choose the granularity to be 1° along the angular axis, the subtended angle θ should be greater than $\frac{\pi}{180}$ for the farthest point on the thin structure of any shape. In other words, for a thin structure of constant thickness d, the farthest point on the structure

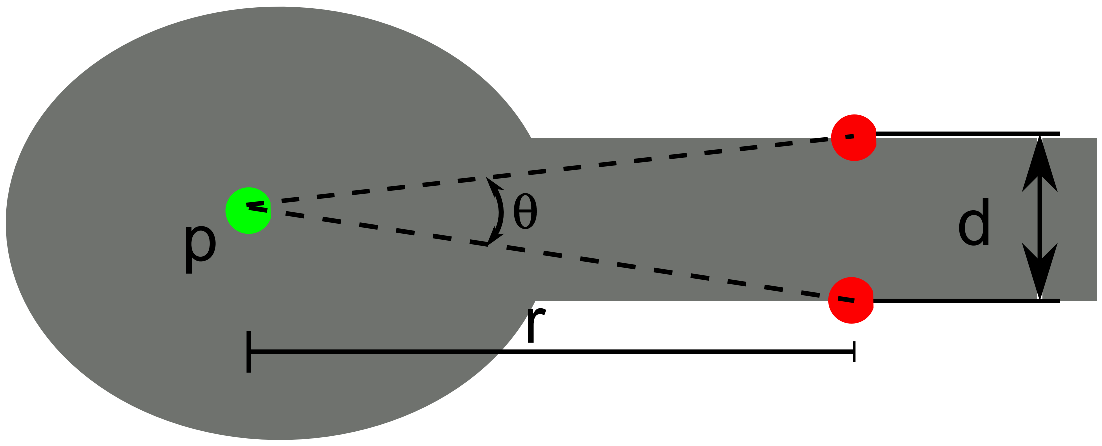


Figure 7.3: The problem of thin elongated structure along the radial axis. P is the pole (or fixation) inside the structure with an elongated part of constant width d. θ is the angle any two opposite points along the two parallel sides of the elongated structure at a distance r away from the pole. The parallel lines would appear merged to the point P if $\theta < 1°$ for the farthest point along the parallel sides of the structure.

should be at most at a distance r away from the pole to stay separated in its polar image where $r < \frac{d*180}{\pi}$.

Thin elongated structure that does not satisfy the condition stated above merges to form a line and hence the proposed segmentation method is unable to trace the boundary of the thin structure exactly. See how the fixation on the neck of the Giraffe in Fig 7.4a results in the partial detection of the rear leg as the optimal path through the polar edge map cuts in the middle of that leg (Fig 7.4b,d). Look at the blown-up image of the portion in the polar space where the path cuts the leg prematurely (Fig 7.4c) and thus an edge is hallucinated in the Cartesian space (Fig 7.4e). However, if the fixation is made close to the leg in the Giraffe in Fig 7.4, the exact contour of the leg will be revealed fully. Keeping that in mind, we propose a multiple fixation strategy to obtain the boundary of such shapes exactly.

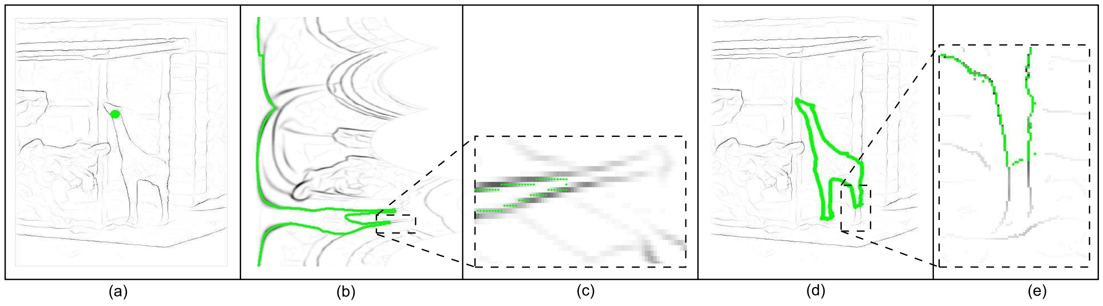


Figure 7.4: The problem of merging in the presence of a thin elongated structure. (a) The inverse probabilistic boundary edge map of an image containing a giraffe with the fixation shown by the green dot. (b) The optimal path through the polar transformation of the edge map. (c) The part of the leg merged together in the polar space is highlighted. (d) The optimal polar path in the Cartesian space. (e)The highlighted portion of the leg in the Cartesian space.

7.2 Multiple fixation based Segmentation

So far, we have obtained a segmentation for a given fixation and our objective, in this section, is to refine that segmentation by making additional fixations inside the initial segmentation to reveal any thin structures not found in the initial segmentation. Detecting these thin structures can be an expensive and complicated step if we fixate at every location inside the region. Instead, if we fixate at only the "salient" locations and incrementally refine the initial segmentations as the new details are revealed, we can be certain of not missing any complicated parts of the shape. So, where are these salient locations?

7.2.1 Locations to make additional fixations

The "salient" locations inside the segmentation correspond to those significant changes in the region boundary that results in the protrusion of the contour away from the center. Although there can be many ways to identify these locations, the simplest and fastest way to find them is through the skeleton of the segmented region. It represents the basic shape of the region boundary. We select the junctions of the skeleton as the salient locations as a junction is guaranteed to be present if the boundary has any protruding part.

Although skeleton extraction based on thinning process is generally sensitive to slight variation in the region boundary, the divergence-based skeleton extraction proposed by Dimitrov et al. [26] is stable and does not lead to spurious junctions. In fact, using the threshold on the divergence value (0.4 for all our experiments), the spurious junctions arisen due to slight change along the boundary contour can be completely avoided. Also, as the purpose of extracting the skeleton is only to select other possible fixation points inside the segmented region and not to use it to refine the segmentation per se, the exact topology of the skeleton does not matter to the task at hand. Choosing fixation points on the skeleton meets the single criterion for our segmentation algorithm to succeed: *the fixation point must lie inside the segmented region.*

Given a set of junctions in the skeleton, we choose the junction closest to the current fixation point. To avoid fixating at the same location twice during the refinement process, all the previous fixations are stored and checked against before selecting a new junction to fixate at. Also, after making a series of fixation, the closest junction is found as the one with the minimum distance from any one of the elements in the set of already fixated locations.

7.2.2 Refining the initial segmentation

Now, the question is how do we refine the initial segmentation by incorporating new details revealed by making additional fixations? There are two aspects of this process that we should emphasize at the outset. First, the fixations are made in a sequence and, in every step of the process, the boundary edge map is updated to carry the information about the part of region contours found by the valid previous fixations.

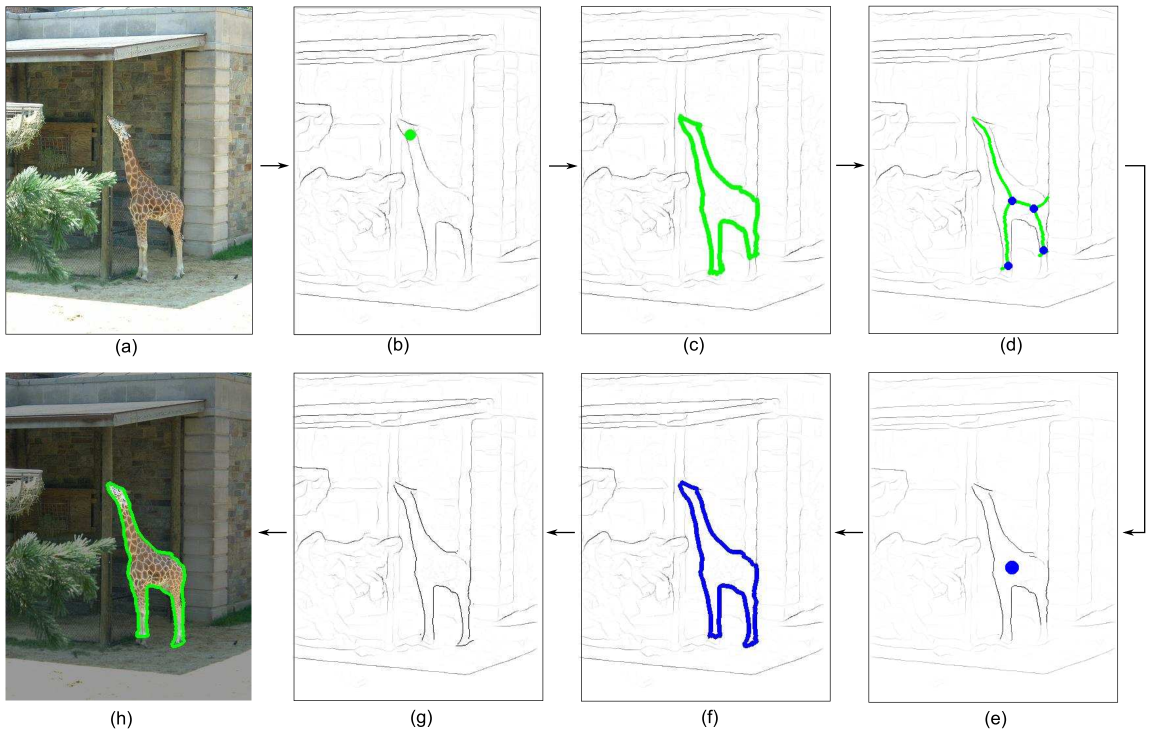


Figure 7.5: Multiple fixations to refine the initial segmentation. (a) The original color image containing the Giraffe (the object of interest). (b) The inverse probabilistic boundary edge map and the given fixation (the green circular dot). (c) The segmentation result for the given fixation. (d) The skeleton of the current segmentation with detected junctions shown by the blue circular dots. The junctions not marked are too close to the original fixation. (e) The next fixation (the blue circular dot) in the modified edge map. (f) The segmentation for the new fixation. (g) The modified edge map after incorporating additional information revealed by the new fixation. (h) The final segmentation after fixating at all the other junctions.

Second, only if the new fixation traces all the known region contours from previous steps, the additional contours revealed by the new fixation are incorporated to refine the segmentation further.

At every stage during the refinement process, there is a segmentation mask of the fixated region. The edge fragments, that lie along the region boundary and are sufficiently long (≥ 10 pixels in all our experiments), are considered the correct region contours. Accordingly, the probabilistic boundary edge map (in the Cartesian space) is modified such that all the edge pixels along these contours are assigned a probability of 1.0. For any additional fixation, this modified edge map is used to find the corresponding segmentation.

Now, if the segmentation for a new fixation does not trace almost all the known contour pixels, the corresponding segmentation is not considered valid for refining the current segmentation. Whereas, if the new segmentation traces most of the known contours, say, 95 percent (for our experiments) of all the known edge pixels along the contour, the new segmentation is combined with the current segmentation in a binary OR manner. Using the updated current segmentation, the probabilistic boundary edge map is modified to include any new contours revealed by this fixation. The process of refinement stops when all the salient locations have been fixated. For the exact details of the steps involved in the refinement process, refer to the pseudo code provided in Algorithm 1.

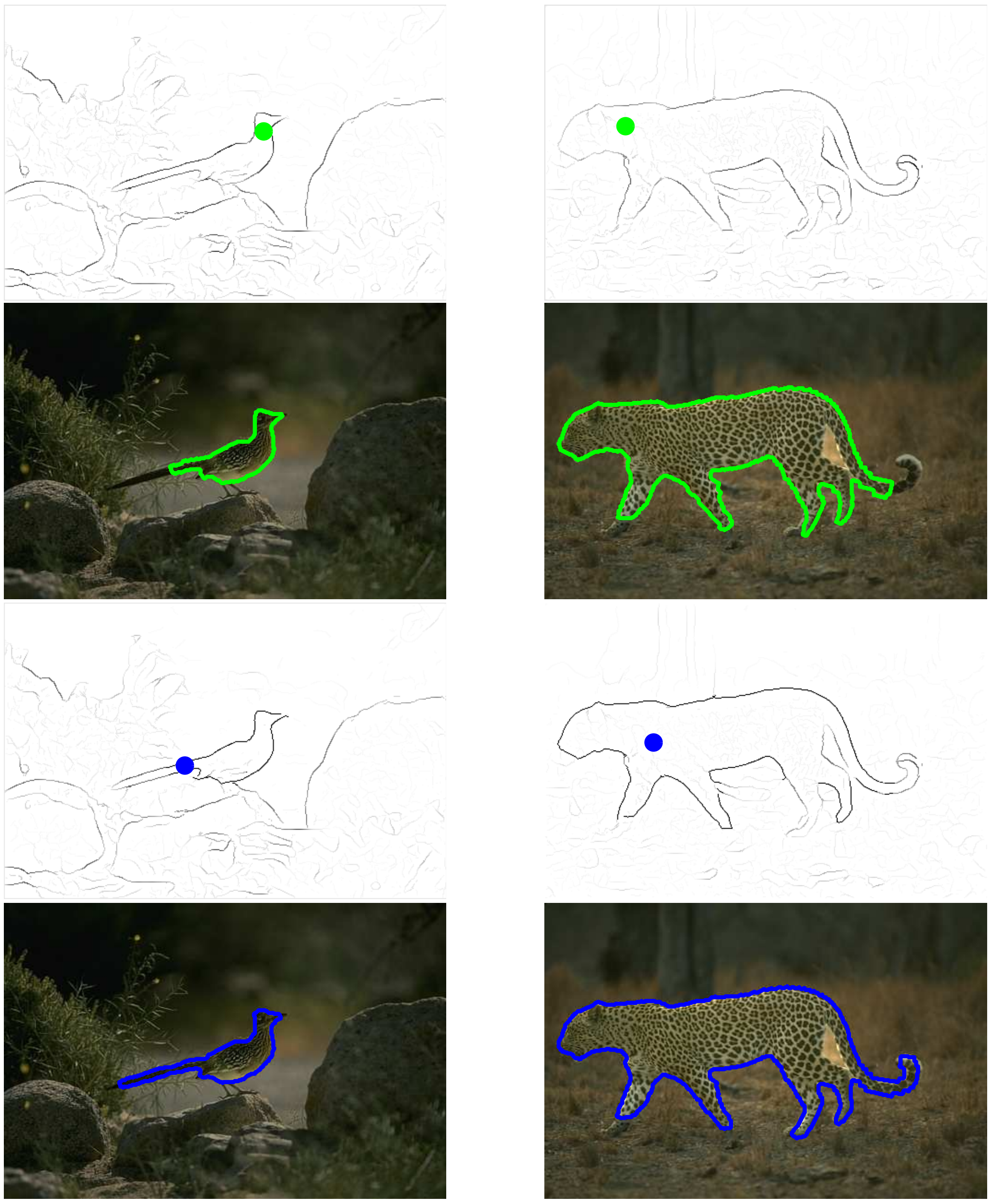


Figure 7.6: Segmentation refinement using multi-fixation segmentation strategy. Row 1: The inverse probabilistic boundary edge map with the first fixation. Row 2: Segmentation result. Row 3: The modified edge map with the next most important fixation. Row 4: segmentation result for the next fixation.

Algorithm 1 Segmentation Refinement Process

```
  $S_R$: Skeleton of region $R$
  $B$: Boundary mask {a 2D array}
  $X_j$: set of junctions in the skeleton
  $X_f$: set of fixations made so far
  $C$: a set of contour fragments $c^i$ where $c^i = \{(x^i_k, y^i_k)\}_{k=1}^{n_i}$
  and $n^i \geq 10$
  $N_C$: total number of contour fragments
Require: $R$: segmentation
  $E$: Edge map {a 2D array}
  $x^1_f$: Given fixation
Ensure: $R^{'}$: Final segmentation
  begin:
  set $B$ to 0
  set $X_f$ empty
  set $R^{'} \leftarrow R$
  Add $x^1_f$ into $X_f$
  $C \leftarrow$ extractContours(E)
  for $i = 0$ to $N_C$ do
    if $c^j$ is at region boundary then
      for $j = 0$ to $n^i$ do
        $B(x^i_j, y^i_j) = 1.0$
        $E(x^i_j, y^i_j) = 1.0$
      end for
    end if
  end for
  $S_R \leftarrow$ findSkeleton($R$)
  $X_j \leftarrow$ findJunctions($S_R$)
  $X_j \leftarrow X_j - X_f$ { Remove the locations already fixated}
  $n = 1$
  while $X_j$ not empty do
    $x^{n+1}_f \leftarrow$ findClosestJunction($X_j, X_f$)
    Add $x^{n+1}_f$ into $X_f$
    $R_{new} \leftarrow$ segmentForNewFixation($x^{n+1}_f$,$E$)
    $B_{new} \leftarrow$ extractRegionBoundary($R_{new}$)
    if count(($B_{new} \bigcap B) > 0) > 0.9 \times$ count($B > 0$) then
      $B \leftarrow B_{new}$
      $R^{'} \leftarrow R_{new}$
      $E(x, y) = 1.0 \forall (x, y)$ such that $B(x, y) > 0$
    end if
  end while
```

Chapter 8

Object Segmentation Using Shape

In this chapter, we discuss a simple approach to use multiple fixations to segment complex objects. The motivation is to establish connection between fixations to make the vision process more purposive. The work reported in this chapter however is in progress and should be read with that in mind.

8.1 Introduction

The human eye fixates at the salient regions of a scene. Conventional wisdom suggests that these regions are the peaks of a saliency map generated using low level visual cues (such as contrast, orientation, color, texture etc.) [43, 99]. Recently, however, Einh*ä*user et al. reported in their psychophysical experiments that objects predict the fixation locations better than the saliency map of the image, suggesting that the fixations usually land on the objects in the scene.

Although foveal vision is not always necessary for recognition, it is intuitively clear that eye fixates on objects to enable perceptual recognition. Is this necessary? While it is hard to answer this question conclusively, studies show a close relationship

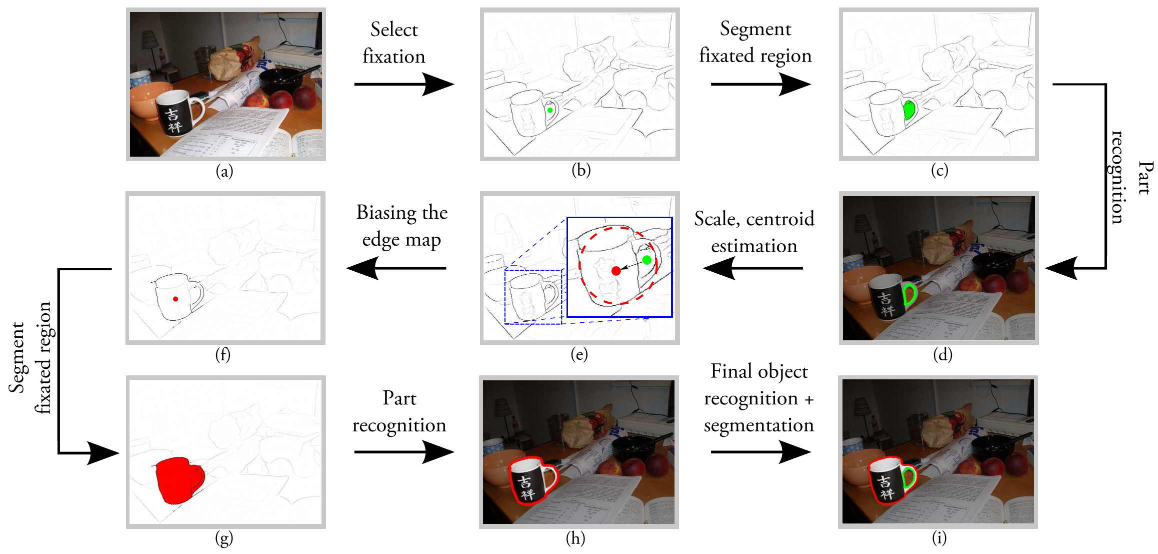


Figure 8.1: The basic steps of the algorithm to detect objects, in this case a mug, in the image. The circular dots in (b),(e),and (f) show the fixation locations for the two object parts, the handle and the full body of the mug.(a)-(d) illustrate the detection of the first part, the handle, which then predicts the centroid and the scale of the next part as shown in (e). The edge map is biased with the expected shape of the contour of the mug in (f). (g) and (h) show the segmentation of the entire mug using the modified edge map. Finally, the segmentation and detection of Mug as a collection of its part is show in (i).

between fixations and object perception. For example, Nelson and Loftus [61] show that the performance on a memory test during brief presentations of objects is best if the object is fixated directly. Hollingworth et al. [40] report that the participants could detect scene changes only when they occur in their foveal vision. This suggests the important role of fixation in recognition.

An object is a collection of parts [52, 10, 29]. Detecting an object means finding all of its parts. As the human visual system fixates to recognize, it will have to fixate on the individual parts of an object to recognize them and thus recognize the object. While simple objects can be recognized by a single fixation, complex objects will require multiple fixations to be recognized as they have multiple parts.

The series of fixations used to recognize an object's different parts is itself structured and contains information about the object. In fact, the idea that a sequence of fixations in the scene are related and encode the semantic content of the scene was proposed almost three decades ago by Norton and Stark [62]. They conducted experiments with documents and images, but not in the context of object recognition.

We do not claim that the human visual system does fixate on individual parts of an object to detect and recognize it, rather we draw inspiration from the findings of the psychophysical experiments discussed above and propose a computation model that works in that spirit.

It is important to note that we do not have at our disposal a variable resolution camera like the human eye. Thus fixation amounts to bringing the camera optical axis to a scene (image) point. In the simplified case of working with single images, fixation amounts to selecting a point in the image.

We propose a fixation based visual process that starts by making a series of fixations in the scene. For every fixation, it finds the region it is looking at, based on low-level visual cues. The identified regions are considered as object parts and described by the log-polar representation of its closed boundary contour. The description of the segmented region is matched against that of the parts of different objects, learned during a training. In the event of a match, the proposed method searches for other parts of the matching object using high level information such as scale, centroid and shape of the parts made available by the detected part of that object. For instance, (Fig. 8.1(d)) of the mug predicts the centroid and the scale of the other part (the entire mug) in Fig. 8.1(e).

We propose a new way for detecting and recognizing objects with a strong coupling between low and high level information. To our knowledge, this is the first time when the object parts are detected based on not just the low-level information in the scene but also the high-level knowledge about the shapes of the object parts. The proposed method establishes a feedback framework between low-level processing and high-level knowledge in a clear and intuitive manner.

8.2 Our Approach

8.2.1 Learning Step

The current trend in the object detection/recognition literature is to minimize the amount of supervision required to learn about the objects in the training set. It is primarily because obtaining a database with clearly segmented objects in the training images requires a lot of manual work. Although the manually segmented training images are more informative than having a bounding box around the objects (which is the case for most object detection databases), the manual work requires to generate such a training set outweighs the benefit of using it. Due to this, a good amount of research is done on how to learn about the object parts in an un-supervised manner from the training images with just the bounding boxes or even without them.

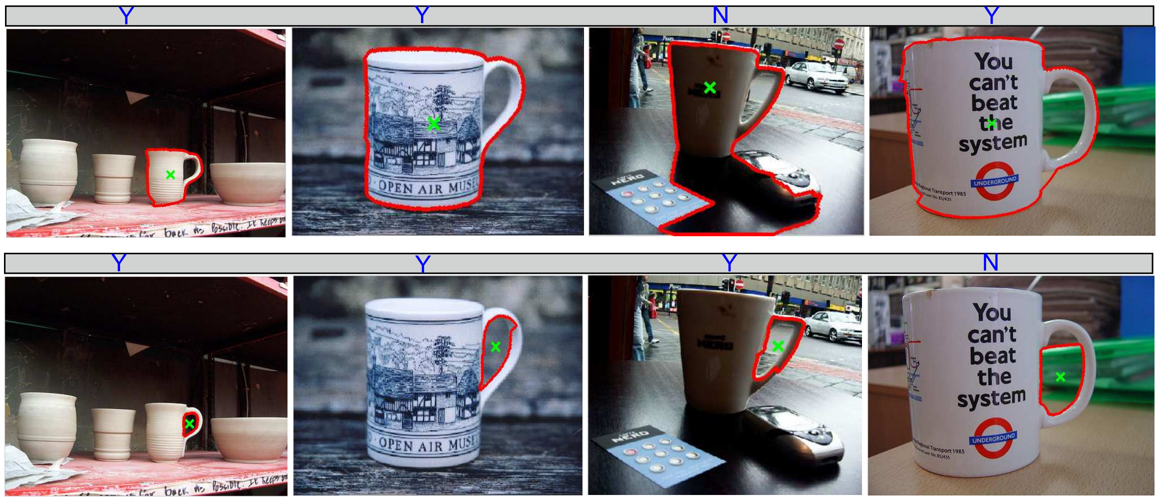


Figure 8.2: The interactive process to learn the parts of Mugs. The first row shows the first part which is the entire mug, and the second row shows the second part which is the handle of the mug. The fixation point (green X) at the center of the object part is selected by a human subject who then decides if the segmentation (the red contour) as given by the fixation based segmentation algorithm[58] for that fixation is the same as what he expected. "Y" and "N" on the top of images show some of the examples of segmentation accepted and rejected by the human subject respectively.

We, however, intend to use a fully supervised method to obtain the parts from the training images. Our procedure to obtain the segmentation of the parts require minimal manual work. We ask human subjects to look at the training images of an object and decide about the parts of the objects. The subjects are asked to split the objects into a fixed number of parts. In our experiments with the ETHZ shape database, the number of parts in an object is fixed to be 2. The subjects fixate at the center of the parts in each training image and the segmentation algorithm proposed by Mishra et al. [58] outputs a region containing that fixation. Now, if the user thinks the region is the correct segmentation of the part that he intended to segment, he allows that region to be added into the training examples for that part otherwise reject it and move to the next image in the training set.

Part Representation

It has been a constant struggle in the literature to define what is a good representation of a part of an object? Local appearance based representation [48, 57] has been most popular with researchers as it is invariant to change in illumination, scale, and viewpoint to some extent. But they cease to be useful for objects (e.g. objects in ETHZ shape database [30]) which are not related by the intensity pattern on its interior but their shapes. For such object classes, the contours of the objects are used as their parts [30]. Also, as segmentation of the images is not given, in place of closed contours the edge fragments are used to represent the parts. Recently, Chunhui et al. [36] uses regions from the segmentation as parts and its closed contour to represent the shape.

We represent a part by a compact region. The regions are the output of the fixation based segmentation algorithm [58] for different fixations in the image. Unlike [36], the log-polar representation of the closed contour is used to describe the region (or part). The reason to use the log-polar space is that the scaling and rotational variation in the shape of the region is easy to handle in log-polar space as they become translation along log(r) axis and θ axis respectively.

Let's consider $\{(x_i, y_i)\}_{i=1}^{m}$ as a set of 2D points along the closed contour and (x_c, y_c) as the centroid of a region in the Cartesian co-ordinate system. Treating the centroid as the pole for the Cartesian to polar transformation, the 2D coordinate, (lr_i, θ_i) of the corresponding location in the log-polar space is calculated as:

$$lr_i = \log_2(\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2})$$

$$\theta_i = \arctan(y_i - y_c, x_i - x_c); \theta_i \in [0, 2\pi)$$

The regions appear at their individual scales. So, to obtain the scale invariant log polar representation, the log polar coordinate calculated have to be shifted along $\log_2(r)$ axis by $\log_2(s)$ where $s = \sqrt{A/\pi}$ is the overall scale of the region and A is its area. Also, both axes of log-polar space are quantized. The $\log_2(r)$ axis ranges from -2 to 2 and the quantization step along this axis is 0.05. The θ axis ranges from $0°$ to $359°$ and each step is $1°$. The scale normalized log space coordinates (lr_i^n, θ_i^n) is given by

$$lr_i^n = \lfloor \frac{lr_i - \log_2(s)}{0.05} \rfloor$$

$$\theta_i^n = \lfloor \frac{\theta_i \times 180}{\pi} \rfloor$$

The quantization result in multiple points along the 2D contour from the Cartesian space mapping to the same location in the log-polar space. So, we pick only the unique locations of all the corresponding coordinates giving us the set of $\{lr_j^n, \theta_j^n\}_{j=1}^m$ representing the closed contour in the log-polar space.

Learning the distribution of log-polar paths

In the training phase, all the regions corresponding to the part k of the object O are collected. The log-polar representation of the closed contours of these regions form a set of locations $\{lr_j^n, \theta_j^n\}_{j=1}^L$ in the log polar space. These points are treated as identically independently distributed and a non-parametric technique is used to encode its distribution. The probability that a 2D location in the log-polar space (lr^n, θ^n) come from the distribution of the log-polar paths of the part k of the object O is given by:

$$P_O^k(lr^n, \theta^n) = \sum_{j=1}^{L} \exp(-\frac{\| lr^n - lr_j^n \|}{2\sigma^2}) \qquad (8.1)$$

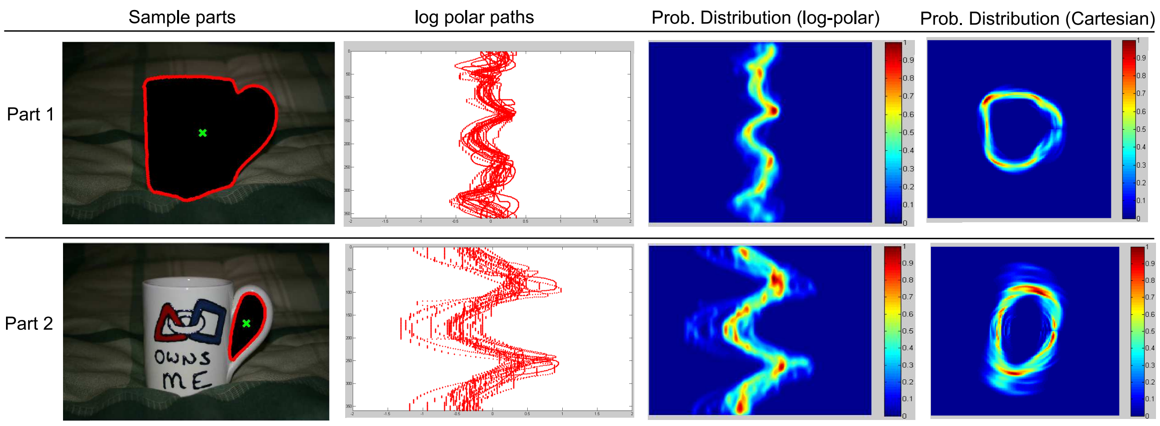


Figure 8.3: The black regions in column 1 are the parts of the Mug. Column 2 contains the collection of the log-polar paths of the region boundary as seen from the fixation point (the cross in the black region of column 1). Column 3 is the sampled output of the log-polar distribution obtained using non-parametric KDE model. Finally, column 4 contains the probabilistic estimate of the shape of the part as sampled from this distribution.

Learning Object Structure

An object is a collection of parts and these parts should be in expected relative scales and at expected relative locations. Let us say the scales and centroids of a pair of parts k_1, k_2 are s_1, s_2 and X_1^c, X_2^c respectively. The structural information is encoded in terms of relative scale and normalized relative position of say part k_2 with respect to part k_1 or vice versa. $\frac{s_2}{s_1}$ and $\frac{X_2^c - X_1^c}{s_1}$ are the relative scales and the relative position of part k_2 with respect to part k_1 respectively. We collect this information from every instances of co-occurrence of the two parts in the training set and build distributions $N(\mu^s_{k_1,k_2}, \sigma^s_{k_1,k_2})$ and $N(\mu^X_{k_1,k_2}, \Sigma^X_{k_1,k_2})$ of the relative scales and relative locations.

8.2.2 Detection Step

Given a test image, the task is now to find the objects present in the image. As our process is based on fixations in the scene, we must find all the possible locations to fixate. In the worst case, each pixel is fixated in order to find the object. But we show

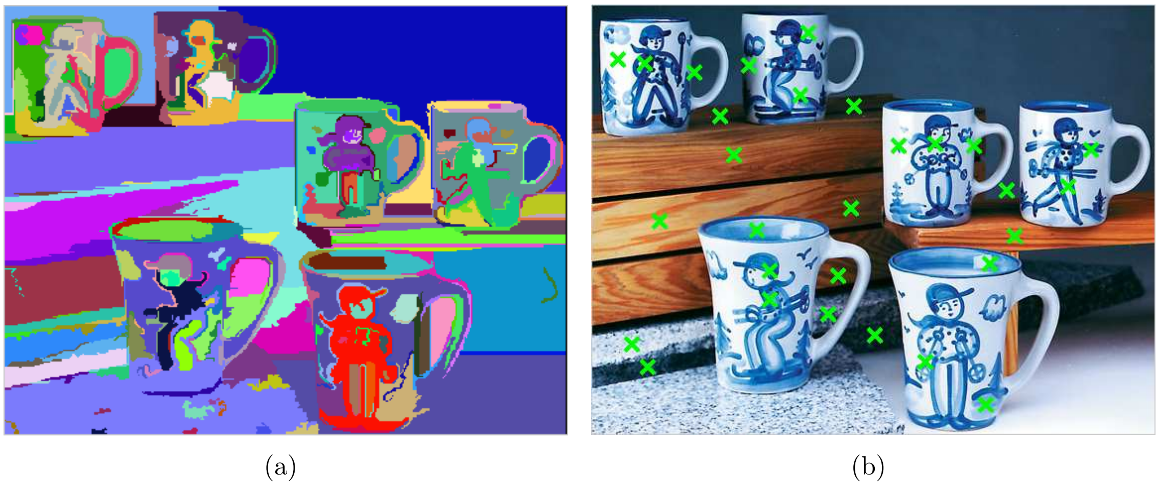


Figure 8.4: The possible fixation locations in the image are shown by green "X"s in (a). These are the centers of the regions from the color based segmentation of that image in (b)

in our experiments that a much lower number of fixations is required to detect the objects. Visual attention algorithms [43] can be used to predict the fixations in the scene (they are usually the peaks of the saliency map of the scene). But, the locations predicted by these algorithms are at or close to boundary edges. The segmentation algorithm used in this paper works better if the fixation point lies away from the edges and closer to the center of the regions.

Keeping that in mind, we use the color-based segmentation algorithm proposed by Pedro and Huttenlocher [28] to first segment the image into regions and use the centroids of these regions as all possible fixations in the image. The number of regions is significantly lower than the total number of pixels in the image. See Fig. 8.4 for an example.

Detecting an object part

Next, for every fixation, a region is segmented containing that fixation point. Using the normalized log-polar representation of these regions, they are compared with the distribution of log-paths of different parts of different objects. For any object O and its part k, the matching score, Ψ_O^k is an average of individual probabilities that each of the points along the log-polar representation comes from the learned distribution of the part k.

$$\Psi_O^k = \frac{1}{N}\sum_{i=1}^{N} P_O^k(lr_i, \theta_i)$$

If $\max_{k,O} \Psi_k^O > T_1$, the region is declared as part k of the object O in the memory where T_1 is the threshold to declare the detection of the first part of an object. As it happens, an hypothesis is made that the entire object O is present at that location. This entails the presence of the other parts of the object O at the expected locations, of the expected size and of the expected shape. Using these information, the detection of the other parts of this object become a highly constrained problem and hence, if not detected, the hypothesis about the presence of an entire object is considered wrong. In the case of occlusion, this might just mean that the parts not detected are occluded. In the ETHZ database, however, occlusion is not prevalent. So, we only accept the hypothesis if all the other parts of the object are detected after using the high-level information about the shape, location and size given by the first detected part.

Detecting Remaining Object Parts in an Active Framework

The main objective of this process is to detect the other parts of the object using the information available about them from the part that has already been detected in the previous step. For the sake of simplicity, let us assume there are only two parts of the object, k_1 and k_2. k_1 is the first part detected in the previous step. So, the task of this active process is to detect k_2.

s_{k_2}, the possible scale of the part k_2, is predicted by the distribution of the relative scale ratio, $N_1(\mu_{k_2,k_1}, \sigma_{k_2,k_1})$, learned during the training phase. Similarly, X_{k_2}, the expected location of the centroid of the part k_2, is predicted by sampling the distribution $N_2(X^m_{k_1,k_2}, \Sigma_{k_1,k_2})$. Additionally, we also know the distribution of the contour points around the centroid for the part, $P_O^{k_2}$. We use all of these information to bias the edge map such that the edge fragments forming the closed contour of the part k_2 become more prominent than other edge fragments.

Biasing the edge map to detect the part

The edge map of the image is broken into fragments of constant curvature. Let us say $\{(x_i^e, y_i^e)\}_{i=1}^{N_e}$ is the set of points along the edge fragment, e, of length N_e in the edge map. We determine the likelihood of this edge fragment lying along the contour of the part k_2 by transferring all of its points from the Cartesian to the log-polar space with X_{k_2} as the pole for the conversion. The converted log-polar coordinate is scale-normalized by shifting it by $\log_2(s_{k_2})$ along the log axis in the negative direction. The probability of each edge pixel is calculated by evaluating $P_O^{k_2}$ at its corresponding log-polar location. The overall likelihood of the edge fragment is given by

$$\frac{1}{N_e}\sum_{i=1}^{N_e} P_O^{k_2}(lr_i, \theta_i)$$

Part Recognition

As location of the pole (fixation) is important for transferring the edges from Cartesian to polar space, the biasing process is carried out from the fixation points chosen in a neighborhood of the predicted location. The fixation point with the best biasing probability is chosen as the new fixation.

Now, the segmentation is carried out with the new edge map and the resulting region is now matched with the distribution of the part expected. The closed contour is represented using its log-polar path which is normalized (shifted) by the estimated scale. Now, if the probability of the part coming from that distribution $> T_{\text{part}}$, the part is considered detected.

8.3 Experiments

As a preliminary experiments to test the strength of the proposed method, we use the ETHZ shape database containing 145 images of five objects namely Mugs, Bottles, Giraffes, Swans, and Applelogos. In our experiment, we are only going to restrict ourselves to using only non articulated objects: Bottles, Mugs and Applelogos. For each of these three objects, the total number of images is randomly divided into two equal sets for training and testing purposes. Five different sets are built to get statistically meaningful accuracy values. In each set, 70 percent of the total number of images is used as a training set and the rest as the test set. These images are selected randomly to build five different pairs of training and test sets.

Also, we assume that each object has only two parts and ask the human subject involved during the training phase to choose two prominent parts of the object. These parts can overlap but should be recurring in most of the training images. The match is considered to be found if the detected object in terms of their parts has significant overlap with the known segmentation of the objects. For our experiments, that overlap between the segmented object and the ground truth should be > 0.9. Table. 8.1 shows the mean detection accuracy and the standard deviation over the five sets of test images.

The strength of the proposed method is in using the information provided by the detected part to enhance the detection of the next part of the same object. This

	Our Algorithm	Malik [36]
Mugs	81.0 ± 2.2	76.1 ± 4.4
AppleLogos	75.8 ± 1.8	77.2 ± 11.1
Bottles	87.8 ± 2.0	90.6 ± 1.5

Table 8.1: Accuracy Results for the non-articulated objects from the ETHZ Shape database.

process helps eliminate internal edges and makes the edges corresponding to the actual boundary of the object prominent. This indicates how the information can be propagated through a sequence of fixations, in this case, to segment the object itself.

Now, event though the idea of using structural information to influence the detection of parts can make the object detection a simpler and easier problem, the definition of the part itself which is the entire closed contour is limiting. We have observed that depending upon the low-level information the segmentation of the part does not always remain the same. If the segmentation varies significantly, the centroid of the segmented region would also change. Hence, the description of the part in terms of the the log-polar path of its boundary as seen from the centroid changes as well, making it harder to match two segmentation corresponding to the same part. We need a better definition of the part which is locally invariant and can be extracted even when overall segmentation changes significantly.

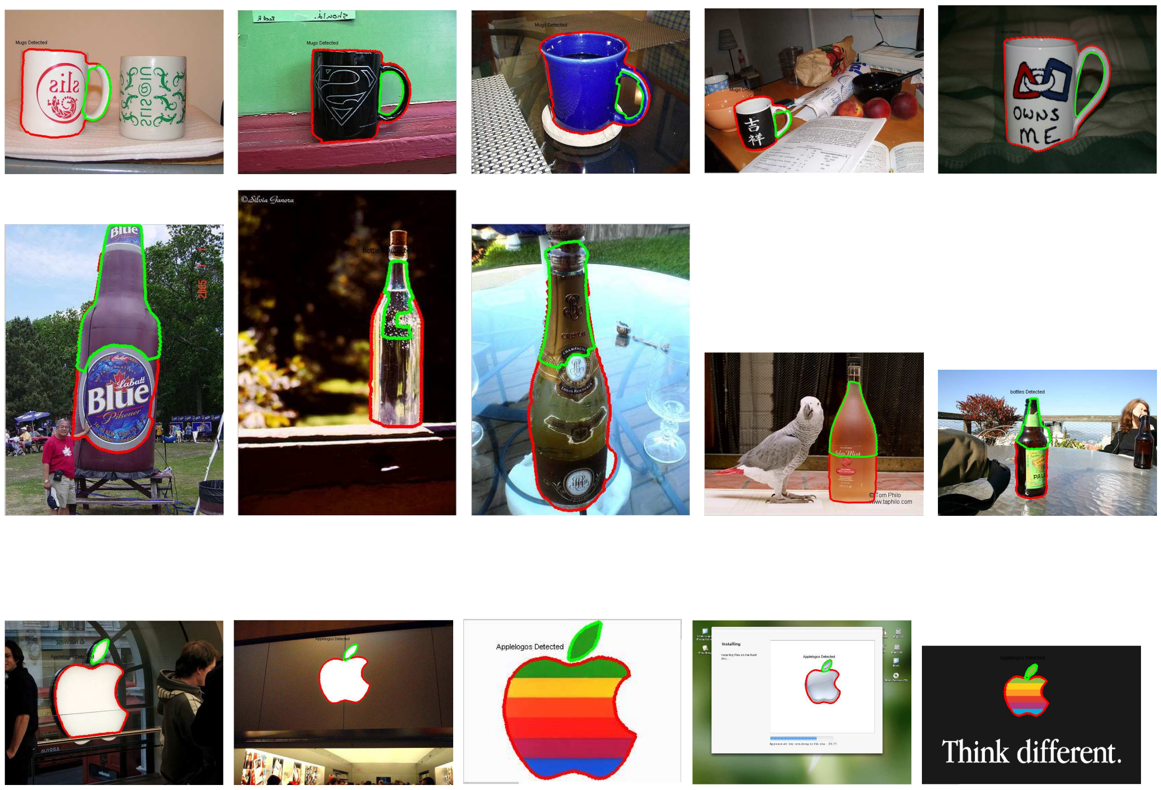


Figure 8.5: Detection results for the objects in the ETHZ Shape Database. Row 1: Mugs, Row 2: Bottles, Row 3: Apple logs

Chapter 9

Conclusion and Future work

In this thesis, a fixation based framework has been proposed that segments the region containing a given fixation point. The new fixation based framework changes the definition of segmentation from the standard one - break an entire scene/image into individual regions- to a new one - segment only the fixated region.

Incorporating fixation into segmentation has not only made the problem of general segmentation easier but also more robust. While in the standard framework the definition of what is a correct segmentation is not clear, in the new formulation fixation identifies the object of interest and the segmentation algorithm has to segment the entire object containing that fixation. More importantly, given the fixation, the region containing that is extracted automatically. Besides, choosing fixation as an input is biologically more plausible than the parameters of other image segmentation algorithms.

One of the other important aspects of the proposed algorithm is the way visual cues are used. The color cues which are dense in nature locate possible boundaries precisely in the scene and the motion or stereo cues are used to choose a subset of these possible boundaries as the real (or depth) boundaries. Due to this combination of color and motion and(or) stereo cues, the accuracy of the boundary of the segmented region is not affected by the errors in computing optical flow or disparity map. In essence, the visual cues have been used in a complementary fashion.

Since the proposed method outputs a region based on a given fixation and the gradient map of the scene, an important question arises: is the segmentation process for every fixation independent of each other? The answer to this question was explored in the thesis and the two possible ways to establish relationship between fixations were found. First, we show that sometimes a series of related fixations are required to output thin complicated shapes in the image; second, to segment an object with multiple parts, relationship between fixations can be used to do it more efficiently.

Since calculating dense optical flow leads to erroneous information and is particularly hard at the boundaries, a different version of fixation based algorithm was proposed which proceeds in a sequential fashion starting with segmenting the fixated moving object using just color and texture cues, estimating the motion information for this segmentation and then using that information to improve the segmentation. All of this is done without calculating any optical flow. This approach to incorporate motion is different than the original formulation but it has semblance to the feedback loops present in the visual cortex between layers of visual cortex.

Finally, the fixation based segmentation process seems to offer a great promise. As the future extension of this work, the algorithm can be used to segment large number of images and the extracted regions can be studied for the consistent structure across different types of scenes. Also, incorporating knowledge into the object detection/segmentation process, described in chapter 8, will be a real progress.

Bibliography

[1] G. Adiv. Determining 3d motion and structure from optical flow generated by several moving objects. *T-PAMI*, 7:384–401, 1985.

[2] G. Adiv. Inherent ambiguities in recovering 3-d motion and structure from a noisy flow field. *T-PAMI*, 11(5):477–489, 1989.

[3] S. Alpert, M. Galun, R. Basri, and A. Brandt. Image segmentation by probabilistic bottom-up aggregation and cue integration. In *CVPR*, June 2007.

[4] P. Arbelaez and L. Cohen. Constrained image segmentation from hierarchical boundaries. In *CVPR*, pages 454–467, 2008.

[5] P. Arbelaez, M. Maire, C. C. Fowlkes, and J. Malik. From contours to regions: An empirical evaluation. In *CVPR*, pages 2294–2301. IEEE, 2009.

[6] S. Ayer, P. Schroeter, and J. Bigün. Segmentation of moving objects by robust motion parameter estimation over multiple frames. In *ECCV*, pages 316–327, London, UK, 1994. Springer-Verlag.

[7] S. Bagon, O. Boiman, and M. Irani. What is a good image segment? a unified approach to segment extraction. In *ECCV*, volume 5305, pages 30–44, 2008.

[8] W. A. Barrett and E. N. Mortensen. Interactive live-wire boundary extraction. *Medical Image Analysis*, 1:331–341, 1997.

[9] P.J. Besl and N.D. McKay. A method for registration of 3-d shapes. *T-PAMI*, 14(2):239–256, 1992.

[10] T. Binford. Visual perception by computer. In *IEEE Conf. Systems and Control*, 1971.

[11] M.J. Black and P. Anandan. Robust dynamic motion estimation over time. In *CVPR*, pages 296–302, 1991.

[12] A. Blake and M. Isard. *Active contours*. Springer-Verlag, 1998.

[13] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr. Interactive image segmentation using an adaptive GMMRF model. In *ECCV*, pages 428–441, 2004.

[14] M. Bober and J. Kittler. Robust motion analysis. In *CVPR*, pages II: 947–952, 1994.

[15] Y. Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *T-PAMI*, 26:359–374, 2004.

[16] Y.Y. Boykov and M.P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *ICCV*, pages I: 105–112, 2001.

[17] Y.Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *T-PAMI*, 26(9):1124–1137, September 2004.

[18] B. Bridgeman, D. Hendry, and L. Stark. Failure to detect displacement of the visual world during saccadic eye movement. *Vision Research*, 15(6):719–22, 1975.

[19] N. Bruce and J. Tsotsos. Saliency based on information maximization. In *NIPS*, 2005.

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

[21] P.J. Burt, J.R. Bergen, R. Hingorani, R. Kolczynski, W.A. Lee, A. Leung, J. Lubin, and H. Shvayster. Object tracking with a moving camera. In *Visual Motion 1989 Proceedings Workshop on*, pages II: 2–12, 1989.

[22] M. Cerf, J. Harel, W. Einhäuser, and C. Koch. Predicting human gaze using low-level saliency combined with face detection. In *NIPS*. MIT Press, 2008.

[23] L.D. Cohen and I. Cohen. Finite-element methods for active contour models and balloons for 2-d and 3-d images. *IEEE PAMI*, 15(11):1131–1147, Nov 1993.

[24] J. Costeira and T. Kanade. A multi-body factorization method for motion analysis. In *ICCV*, page 1071, Washington, DC, USA, 1995. IEEE Computer Society.

[25] C. Currie, G.W. McConkie, L.A. Carlson-Radvansky, and D.E. Irwin. Maintaining visual stability across saccades: Role of the saccade target object. Technical report, Beckman Institute, University of Illinois, 1995.

[26] P. Dimitrov, C. Phillips, and K. Siddiqi. Robust and efficient skeletal graphs. In *CVPR*, volume 1, pages 417–423 vol.1, 2000.

[27] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[28] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, 2004.

[29] P.F. Felzenszwalb and D.P. Huttenlocher. Pictorial structures for object recognition. volume 61, pages 55–79, Hingham, MA, USA, 2005. Kluwer Academic Publishers.

[30] V. Ferrari, T. Tuytelaars, and L.V. Gool. Object detection by contour segment networks. In *Proceeding of the European Conference on Computer Vision*, volume 3953 of *LNCS*, pages 14–28. Elsevier, June 2006.

[31] D.J. Fleet. Disparity from local weighted phase-correlation. In *IEEE International Conference on Systems, Man, and Cybernetics*, 1994.

[32] C.C. Fowlkes, R. M. David, and J. Malik. Local figure/ground cues are valid for natural images. *JV*, 7(8):1–9, 2007.

[33] P.J. Giblin and B.B. Kimia. On the local form and transitions of symmetry sets, medial axes, and shocks. *IJCV*, 54(1-3):143–156, 2003.

[34] R.C. Gonzalez and R.E. Woods. *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006.

[35] J. Grimes. *Perception (Vancouver Studies in Cognitive Science)*, chapter On the failure to detect changes in scenes across saccades, pages 89–100. New York: Oxford University Press, 1996.

[36] C. Gu, J. J. Lim, P. Arbelaez, and J. Malik. Recognition using regions. In *CVPR*, June 2009.

[37] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In *NIPS*, 2006.

[38] J.M. Henderson, C.C. Williams, M.S. Castelhano, and R.J. Falk. Eye movements and picture processing during recognition. *Perception and Psychophysics*, 65:725–734, 2003.

[39] D. Hoiem, A.A. Efros, and M. Hebert. Automatic photo pop-up. *ACM Trans. Graph.*, 24(3):577–584, 2005.

[40] A. Hollingworth, G. Schrock, and JM. Henderson. Change detection in the flicker paradigm: the role of fixation position within the scene. *Memory and Cognition*, 29:296–304, 2001.

[41] M. Irani and P. Anandan. A unified approach to moving object detection in 2d and 3d scenes. *T-PAMI*, 20(6):577–589, 1998.

[42] L. Itti and P. Baldi. Bayesian surprise attracts human attention. In *NIPS*, 2005.

[43] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *T-PAMI*, 20(11):1254–1259, 1998.

[44] A.K. Jain. *Fundamentals of Digital Image Processing.* Prentice-Hall, Inc., 1989.

[45] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *IJCV*, 1:321–331, 1988.

[46] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother. Bi-layer segmentation of binocular stereo video. In *CVPR*, pages II: 407–414, 2005.

[47] D.G. Lowe. *Perceptual Organization and Visual Recognition.* Kluwer Academic Publishers, Norwell, MA, USA, 1985.

[48] D.G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, page 1150, Washington, DC, USA, 1999. IEEE Computer Society.

[49] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[50] J. Malik, S. Belongie, T. Leung, and J.B. Shi. Contour and texture analysis for image segmentation. *IJCV*, 43(1):7–27, June 2001.

[51] T. Malisiewicz and A.A. Efros. Recognition by association via learning per-exemplar distances. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[52] D. Marr and H.K. Nishihara. Representation and recognition of the spatial organization of three-dimensional shapes. *Proc. of the Royal Society of London, Biological Sciences*, B-200:269–294, 1978.

[53] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color and texture cues. *T-PAMI*, 26(5):530–549, May 2004.

[54] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, volume 2, pages 416–423, July 2001.

[55] S. Martinez-Conde, S.L. Macknik, and D.H. Hubel. The role of fixational eye movements in visual perception. *Nature Review Neuroscience*, 5:229–240, 2004.

[56] G.W. McConkie and D. Zola. Is visual information integrated across successive fixations in reading? *Perception & Psychophysics*, 25(3):221–4, 1979.

[57] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *ECCV*, pages 128–142, London, UK, 2002. Springer-Verlag.

[58] A. Mishra, Y. Aloimonos, and C.L. Fah. Active segmentation with fixation. In *ICCV*, 2009.

[59] E.N. Mortensen and W.A. Barrett. Intelligent scissors for image composition. In *SIGGRAPH*, pages I: 191–198, 1995.

[60] R.C. Nelson. Qualitative detection of motion by a moving observer. *IJCV*, 7:33–46, 1991.

[61] W.W. Nelson and G.R Loftus. The functional visual field during picture viewing. *Journal of Experimental Psychology: Human Learning and Memory*, 6:391–399, 1980.

[62] D. Noton and L. Stark. Scanpaths in Eye Movements during Pattern Perception. *Science*, 171:308–311, January 1971.

[63] J.M. Odobez and P. Bouthemy. MRF-based motion segmentation exploiting a 2d motion model robust estimation. In *ICIP*, page 3628, Washington, DC, USA, 1995. IEEE Computer Society.

[64] A. S. Ogale, C. Fermller, and Y. Aloimonos. Motion segmentation using occlusions. *T-PAMI*, 27(6):988–992, 2005.

[65] A.S. Ogale and Y. Aloimonos. A roadmap to the integration of early visual modules. *IJCV*, 72(1):9–25, April 2007.

[66] S. Osher and J.A. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations. *J. Comput. Phys.*, 79:12–49, November 1988.

[67] D. Parkhurst, K. Law, and E. Niebur. Modeling the role of salience in the allocation of overt visual attention. *Vision Research*, 42:107–23, 2000.

[68] U. Rajashekar, I.V.D. Linde, A.C. Bovik, and L.K. Cormack. Gaffe: A gaze-attentive fixation finding engine. *IEEE Transaction on Image Processing*, 17(4):564–573, April 2008.

[69] B. S. Reddy and B. N. Chatterji. An fft-based technique for translation, rotation and scale-invariant image registration. *IEEE Trans. Image Processing*, 5:1266–1271, 1996.

[70] X. Ren and J. Malik. A probabilistic multi-scale model for contour completion based on image statistics. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part I*, pages 312–327, London, UK, 2002. Springer-Verlag.

[71] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut": interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23(3):309–314, 2004.

[72] B.C. Russell, A.A. Efros, J.Sivic, W. T. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *Proceedings of CVPR*, June 2006.

[73] H.S. Sawhney, Y. Guo, and R. Kumar. Independent motion detection in 3d scenes. *T-PAMI*, 22(10):1191–1199, 2000.

[74] A. Saxena, S.H. Chung, and A.Y. Ng. 3-D depth reconstruction from a single still image. *Int. J. Comput. Vision*, 76(1):53–69, 2008.

[75] J.T. Serences and S. Yantis. Selective visual attention and perceptual coherence. *Trends in Cognitive Sciences*, 10(1):38 – 45, 2006.

[76] E. Sharon, A. Brandt, and R. Basri. Fast multiscale image segmentation. *CVPR*, 1:1070, 2000.

[77] E. Sharon, M. Galun, D. Sharon, R. Basri, and A. Brandt. Hierarchy and adaptivity in segmenting visual scenes. *Nature*, 442:810–813, 2006.

[78] J. Shi and J. Malik. Normalized cuts and image segmentation. *T-PAMI*, 22(8):888–905, 2000.

[79] J. Shi and C. Tomasi. Good features to track. In *CVPR*, 1994.

[80] C. Siagian and L. Itti. Rapid biologically-inspired scene classification using features shared with visual attention. *IEEE PAMI*, 29(2):300–12, 2007.

[81] J.F. Silverman and D.B. Cooper. Bayesian clustering for unsupervised estimation of surface and texture models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(4):482–495, 1988.

[82] D. Sinclair. Motion segmentation and local structure. *ICCV*, 93:366–373, 1993.

[83] A.K. Sinop and L. Grady. A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In *ICCV*, pages 1–8, 2007.

[84] A. Spoerri. The early detection of motion boundaries. Technical report, Cambridge, MA, USA, 1990.

[85] R.H. Steinberg, M. Reid, and P.L. Lacy. The distribution of rods and cones in the retina of the cat (*Fells domestica*). *Journal of computational neuroscience*, 148:229–248, 1973.

[86] O.M. Thomas, B.G. Cumming, and A.J. Parker. A specialization for relative disparity in V2. *Nature neuroscience*, 5(5):472–478, May 2002.

[87] W.B. Thompson and T.C. Pong. Detecting moving objects. *IJCV*, 4:39–57, 1990.

[88] S. Todorovic and N. Ahuja. Learning subcategory relevances for category recognition. In *CVPR*, pages 1–8, June 2008.

[89] P. H. S. Torr, O. Faugeras, T. Kanade, N. Hollinghurst, J. Lasenby, M. Sabin, and A. Fitzgibbon. Geometric motion segmentation and model selection [and discussion]. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, 356(1740):1321–1340, 1998.

[90] P. H. S. Torr and D. W. Murray. Stochastic motion clustering. In *ECCV*, pages 328–337, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[91] P. H. S. Torr, R. Szeliski, and P. Anandan. An integrated bayesian approach to layer extraction from image sequences. In *ICCV*, page 983, Washington, DC, USA, 1999. IEEE Computer Society.

[92] A. Torralba, A. Oliva, M.S. Castelhano, and J.M. Henderson. Contextual guidance of eye movements and attention in real-world scenes: The role of global features in object search. *Psychological Review*, 113(4):766–786, 2006.

[93] A. Toshev, A. Makadia, and K. Daniilidis. Shape-based object recognition in videos using 3d synthetic object models. In *CVPR*, 2009.

[94] B. Triggs, P.F. McLauchlan, R.I. Hartley, and A.W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *ICCV Proceedings of the International Workshop on Vision Algorithms*, pages 298–372, London, UK, 2000. Springer-Verlag.

[95] Z. Tu and S.C. Zhu. Image segmentation by data-driven markov chain monte carlo. *T-PAMI*, 24(5):657–673, 2002.

[96] Z.W. Tu and S.C. Zhu. Mean shift: a robust approach toward feature space analysis. *T-PAMI*, 24(5):603–619, May 2002.

[97] O. Veksler. Star shape prior for graph-cut image segmentation. In *ECCV (3)*, pages 454–467, 2008.

[98] R. Vidal and Y. Ma. A unified algebraic approach to 2-d and 3-d motion segmentation and estimation. 25(3):403–421, October 2006.

[99] D. Walther and C. Koch. Modeling attention to salient proto-objects. *Neural Networks*, 19(4):1395–1407, April 2006.

[100] J.Y.A. Wang and E.H. Adelson. Layered representation for motion analysis. pages 361–366, 1993.

[101] J.Y.A. Wang and E.H. Adelson. Spatio-temporal segmentation of video data. 1994.

[102] J. Weber and J. Malik. Rigid body segmentation and shape description from dense optical flow under weak perspective. *T-PAMI*, 19(2):139–143, 1997.

[103] Y. Weiss. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In *CVPR*, page 520, Washington, DC, USA, 1997. IEEE Computer Society.

[104] C.S. Wiles and M. Brady. Closing the loop on multiple motions. In *ICCV '95: Proceedings of the Fifth International Conference on Computer Vision*, page 308, Washington, DC, USA, 1995. IEEE Computer Society.

[105] L.R. Williams and D.W. Jacobs. Stochastic completion fields: a neural model of illusory contour shape and salience. *Neural Comput.*, 9(4):837–858, 1997.

[106] J. Wills, S. Agarwal, and S. Belongie. A feature-based approach for dense segmentation and estimation of large disparity motion. *Int. J. Comput. Vision*, 68(2):125–143, 2006.

[107] K.C. Winkler, R.W. Williams, and P. Rakic. Photoreceptor mosaic: number and distribution of rods and cones in the rhesus monkey. *Journal of computational neuroscience*, 297:499–508, 1990.

[108] C. Xu and J.L. Prince. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, 7(3):359–369, Mar 1998.

[109] J.Y. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. pages IV: 94–106, 2006.

[110] A. L. Yarbus. *Eye Movements and Vision.* New York: Plenum Press, 1967.

[111] S.X. Yu and J. Shi. Grouping with bias. In *NIPS*, 2001.

[112] C.T. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, 20(1):68–86, 1971.

[113] Z. Zhang, O.D. Faugeras, and N. Ayache. Analysis of a sequence of stereo scenes containing multiple moving objects using rigidity constraints. In *ICCV*, pages 177–186, 1988.

[114] Q. Zheng and R. Chellapa. Motion detection in image sequences acquired from a moving platform. In *ICASSP*, pages 201–204, 1993.