AN EFFICIENT GRAPH-THEORETICAL APPROACH FOR INTERACTIVE MOBILE IMAGE AND VIDEO SEGMENTATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

OZAN ŞENER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

MAY 2013

Approval of the thesis:

AN EFFICIENT GRAPH-THEORETICAL APPROACH FOR INTERACTIVE MOBILE IMAGE AND VIDEO SEGMENTATION

submitted by **OZAN ŞENER** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Gönül Turhan Sayan
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. A. Aydın Alatan
Supervisor, **Electrical and Electronics Eng. Dept., METU** _____

Examining Committee Members:

Prof. Dr. Gözde Bozdağı Akar
Electrical and Electronics Engineering Department, METU _____

Prof. Dr. A. Aydın Alatan
Electrical and Electronics Engineering Department, METU _____

Prof. Dr. Levent Onural
Electrical and Electronics Engineering Dept., Bilkent University _____

Prof. Dr. Fatoş Yarman Vural
Computer Engineering Department, METU _____

Assoc. Prof. Dr. Çağatay Candan
Electrical and Electronics Engineering Department, METU _____

Date: May 24, 2013

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: OZAN ŞENER

Signature :

ABSTRACT

AN EFFICIENT GRAPH-THEORETICAL APPROACH FOR INTERACTIVE MOBILE IMAGE AND VIDEO SEGMENTATION

Şener, Ozan

M.S., Department of Electrical and Electronics Engineering

Supervisor : Prof. Dr. A. Aydın Alatan

May 2013, 96 pages

Over the past few years, processing of visual information by mobile devices getting more affordable due to the advances in mobile technologies. Efficient and accurate segmentation of objects from an image or video leads many interesting multimedia applications. In this study, we address interactive image and video segmentation on mobile devices. We first propose a novel interaction methodology leading better satisfaction based on subjective user evaluation. Due to small screens of the mobile devices, error tolerance is also a crucial factor. Hence, we also propose a novel user-stroke correction mechanism handling most of the interaction errors. Moreover, in order to satisfy the computational efficiency requirements of mobile devices, we propose a novel spatially and temporally dynamic graph-cut method. Conducted experiments suggest that the proposed efficiency improvements result in significant computation time decrease. As an extension to video sequences, a video segmentation system is proposed starting after an interaction on key-frames. As a novel approach, we redefine the video segmentation problem as propagation of Markov Random Field (MRF) energy obtained via interactive image segmentation tool on some key-frames along temporal domain. MRF propagation is performed by using a recently introduced bilateral filtering without using any global texture or color model. A novel technique is also developed to dynamically solve graph-cuts for varying, non-lattice graphs. In addition to the efficiency, segmentation quality is also tested both quantitatively and qualitatively; indeed, for many challenging examples, quite significant time efficiency is observed without loss of segmentation quality.

Keywords: Image/Video Segmentation, MRF, Graph-cuts, Dynamic Algorithms

ÖZ

ETKİLEŞİMLİ GEZGİN İMGE VE VİDEO BÖLÜTLEME İÇİN ÇİZGE TEMELLİ ETKİN BİR YAKLAŞIM

Şener, Ozan

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi : Prof. Dr. A. Aydın Alatan

Mayıs 2013 , 96 sayfa

Bu tezde, imge ve videoların gezgin cihazlarda etkileşimli bolütlenmesi problem ele alınmıştir. Hali hazırda var olan ve gezgin uygulamalara uygun etkileşim metodlarının analizi, bu metodların kusurlarını ortaya çıkarmıştır. Bulunan kusurları çözmek için yeni bir etkileşim yöntemi, *boyama*, onerilmiştir. Önerilen yöntemin deneysel analizi için öznel bir değerelendirme uygulanmıştır. Gezgin cihazların küçük ekranları birçok kullanıcı etkileşimi hatasına yol açmaktadir. Bu sorunun çözümü için de yeni bir yöntem önerilmiş ve test edilmiştir. Gezgin cihazların işlemsel kapasite gereksinimleri nedeni ile, uzaysal ve zamansal dinamik cizge kesit algoritması onerilmiştir. Deney sonuçlar da önerilen yöntemin işlemsel verim artışını onaylamıştır. Tasarlanan video bölütleme sistemi, bazı ana çerçevelerin etkileşimli bölütlemesi ile başlamaktadır. Etkileşimli çerçeve bölütlemesinin ardından, bulunan Markov Rastgele Alanı (MRF) enerji fonksiyonu zamansal boyutta yayılmıştır. Yayılma operasyonu iki yönlü süzgeçlerle evrensel bir model kullanmadan gerçekleştirilmiştir. İşlemsel verimi artırmak için, iki yönlü üstsel süzgeçler kullanılmıştır. Buna ek olarak, değişen ve kafes yapısına sahip olmayan cizgelerin dinamik olarak cizge kesitlerinin bulunması icin yeni bir teknik önerilmistir. Uygulanan testlerde ciddi bir işlemsel verim artışı herhangi bir doğruluk kaybı olmadan gözlenmiştir.

Anahtar Kelimeler: İmge/Video Bölütleme, MRF, Cizgeler, Dinamik Algoritmalar

Don't Panic

ACKNOWLEDGMENTS

First and foremost I would like to thank my advisor, Prof. Dr. A. Aydın Alatan, who introduced me to the field of computer vision and multimedia. I am grateful to him for his scientific contributions to this thesis and for his great flexibility in giving me the crucial room I needed to explore my own ideas.

Throughout my master study, I had the pleasure of working closely with Nokia Research Center, Tampere. I would like to thank Dr. Kemal Uğur from Nokia Research Center for his valuable contributions to this work, sharing his valuable insight about interactive multimedia and being such a kind host in my visit to Tampere, Finland.

I would like to thank Prof. Dr. Levent Onural and Dr. Engin Tola for proofreading most of this thesis. Their suggestions improved the quality of this thesis significantly and changed my approach to scientific writing. I am also grateful to Dr. Cevahir Çıgla and Yeti Ziya Gürbüz for the valuable discussions we had about bilateral filtering and generously sharing their source code and results. I also would like to thank Özümcan Demir for the beautiful drawing of the figures in Chapter IV.

My deepest gratitude goes to Assoc. Prof. Dr. Çagatay Candan, Dr. Yang Wang and Elif Vural who motivated me to study mathematics and attack challenging theoretical problems. All the discussions we had made me aspire to learn and contribute. Thank you for sharing your enthusiasm, vision and knowledge in a most generous way possible.

I am deeply thankful to Scientific and Technological Research Council of Turkey (TUBITAK) and Nokia Research Center, Tampere for their generous financial support during my master research.

As a noisy juggler having frequent loud laughs of no reason, I would like to thank all members of Multimedia Research Group for their support and understanding. I am grateful to Yağız Aksoy and Erhan Gündoğdu for all the work we co-authored, Yeti Ziya Gürbüz and Emin Zerman for the TA work we did together, Beril Beşbınar for all the valuable discussions we had about life and academia, Ahmet Saraçoglu and Osman Serdar Gedik for their precious mentorship. Last but not least, I would like to thank Emrecan Batı for sharing his knowledge on LaTeX, programming and GNU/Linux and for all the discussions we had which caused countless hours of procrastination full of coding.

This thesis is not the product of a sole research study but the result of two years full of excitement, fun and procrastination. Special thanks go to all members of METU Juggling Club for their friendship and all the fun we had together. Indeed, juggling

was the most effective and entertaining procrastination tool I have ever experimented. I also would like to thank Pink Floyd for all the beautiful music, and Douglas Adams for the extraordinary world he created in *The Hitchhiker's Guide to the Galaxy*. As Douglas Adams stated, *"Life is paradoxically coincidental to the ironical tyranny applicable to the unparalleled definition of reverse entropy"*. I am also grateful to everyone who coincidentally become part of this experience.

Finally, nothing would have meaning if I did not have the support of my family and friends. I am grateful to my parents, Ömer Şener and Güllizar Şener, for their unconditional love and support and my brother Umut Şener for his many successful attempts to make me have a vacation and visit my home town, these vacations kept my sanity during all the research breakdowns I had.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

Next generation mobile devices have advanced display and processing technologies. Hence, generation, processing and display of the multimedia content on mobile devices are getting affordable. Even high quality user interaction is possible with the advancement of multi-touch screens. All of these advancements let computer vision, computer graphics and multimedia researchers migrate existing algorithms to the mobile domain. Extracting an object of interest from image and/or video is one of these algorithms due to its various applications, such as photo editing [67], interactive 2D/3D conversion [2], collaborative segmentation [74], video cut-outs [48] and tapestry generation [61]. However, crucial differences between requirements of classical computer based multimedia applications and mobile ones demand tailoring of these applications for mobile domain.

Although mobile devices have wide range of advantages and disadvantages over computer based systems, there are three main points need to be considered for mobile image and video segmentation problem; namely, rich interaction possibilities of touch screen devices, small screen sizes and low computational capacities.

Touch-screen devices enable many interaction scenarios; for example, it is possible to use pinch, scroll, pan, spread or rotate via touch-screen devices. On the other hand, classical mouse based interfaces either use scribbles or point selection. Hence, in order to improve user interaction quality, new interaction scenarios should be proposed.

Small screens of the mobile devices decrease the precision of the interaction quality significantly. When the user aims for a specific pixel in screen, there is significant uncertainty around the interaction point. Therefore, algorithms should tolerate a significant amount of interaction error.

Although computational power of the mobile devices have been significantly improved during the last years, they are still far from being comparable against desktop computer systems. Hence, the proposed algorithms should have low computational complexity.

In summary, interaction quality, error-tolerance and computational efficiency are three main factors need to be considered for an successful mobile segmentation application. In addition to these, segmentation quality is still an important factor for user experience quality and successful mobile multimedia applications.

1.2 Scope of the Thesis

In this thesis, segmentation is revisited from the perspective of mobile touch-screen usage. All three factors explained in Section 1.1 has been considered for both image and video segmentation problem, separately.

From the perspective of user interaction, image and video segmentation problems are considered to be equivalent. Interaction for the video segmentation problem has been applied to initial frame of the video only. Hence, user interaction for video segmentation algorithm is actually interactive image segmentation method. After the analysis of existing interaction techniques, a novel interaction methodology, *coloring*, is proposed.

Error-tolerance problem is solved with the proposed pre-processing step. Before feeding user interaction to the segmentation method, erroneous interaction is discarded by the proposed method. Even correct user interaction is estimated as a result of graph based search problem.

Graph-theory and Markov random fields (MRF) are common mathematical formulation frequently used in the literature for the segmentation problem. Hence, the image segmentation problem has been formulated as inference on MRF's and solved by using graph-based approaches. Furthermore, video segmentation problem is defined as MRF energy propagation throughout the frames. Propagation is realized via bilateral filters, and resulting energies are minimized via graph-based method. For computational efficiency, novel dynamic and iterative graph-cut solutions have been proposed for both image and video segmentation problems. Information permeability filter is also used for efficiency increase.

1.3 Outline of the Thesis

MRF energy functions and graph theory are two mathematical formulations used to explain and analyse the proposed segmentation method. In Chapter 2, required background on MRF energy functions and graph theory is summarized. For the case of binary segmentation problem, min-cut/max-flow algorithm is an efficient method to compute MAP estimate on MRF energies; hence, min-cut/max-flow method has been explained in detail. Necessary and sufficient conditions on exact MAP estimation is also explained with discussions on submodularity.

Chapter 3 is devoted to interactive image segmentation problem. All three factors stated in Motivation has been analysed for existing algorithms in the literature. Novel improvements are also proposed for all three factors. An interaction method, *coloring*, is proposed and subjectively analysed. An error-tolerance method is also proposed and tested. In order to decrease computational complexity, a spatially dynamic and iterative graph-cut method is also proposed and experimented.

In Chapter 4, available literature on interactive and automatic video segmentation is summarized briefly. Existing algorithms have been experimented with the metric of segmentation quality and execution time. A novel video segmentation algorithm based on MRF energy propagation via bilateral filter is proposed. In order to increase

the computational efficiency, information permeability/bi-exponential (IP/BE) filters are used. Furthermore, necessary background on IP/BE filters are also presented. In order to further increase the efficiency, min-cut/max-flow problem has been revisited for varying graph-linear filtering scenario. A novel dynamic min-cut/max-flow method is also presented. Proposed interactive video segmentation is extended for automatic video segmentation problem as a speed-up tool. Both interactive and automatic video segmentation methods has been experimentally compared for both computational efficiency and segmentation quality.

Finally, in Chapter 5, thesis is summarized and the conclusions of the experiments are presented. Future research directions and existing open problems are also presented in detail. Results of the computation time experiments are discussed both theoretically and practically. Although all of the methods are discussed by using big O notation, effect of implementation and data dependency is also discussed. Although proposed method shares the same worst case complexity with many other methods in the literature, performed experiments suggest a significant computation time decrease.

CHAPTER 2

GRAPH THEORETIC FORMULATION OF INTERACTIVE SEGMENTATION

Human visual system and design of imaging devices induce a graph structure on sampling and processing of visual information. Sampling of the visual space is usually performed in a space-aware manner due to the structure of visual system and imaging sensors. In this sense, image sensors has non-uniform Cartesian like geometries. On the other hand, higher vertebrate visual systems are commonly based on space-aware sampling [37]. Hence, exploiting space aware methods in processing of visual information is consistent with sampling of the visual information.

First usage of graph theory in computer vision was for clustering feature points in order to obtain Gestalt clusters [79]. Since then, graph theory has become a standard tool for many low-level computer vision tasks such as segmentation [29, 64, 46, 12, 60, 79, 35] and depth estimation [13, 62]. Graph clustering is also a widely studied problem in mathematics [63]. Since segmentation is inherently a clustering problem, it is easy to adopt available graph clustering methods for the segmentation problem.

Within the scope this thesis, we are mainly interested in the interactive segmentation problem. The differences between interactive segmentation and the automatic segmentation problems are two folds. First; interactive segmentation aims to differentiate foreground and background. Hence, this problem is binary graph clustering problem. Second; there exist prior information to model foreground and background in interactive segmentation problem due to the user interaction.

In the following sections, we will first create the graphical model for this problem. Then, we will convert the MAP estimation problem to a well-known combinatorial optimization problem, namely *min-cut/max-flow* [31]. Finally, we will summarize and analyse the Ford & Fulkerson algorithm with Boykov & Kolmogorov heuristic [13], which is the main tool used to solve min-cut/max-flow problem within the scope of this thesis, in detail.

Before explaining details of the models and methods, we need to explain basic notations and definitions of combinatorial entities used in these models and methods; A graph, which is a combinatorial structure combining nodes and edges, is represented as $G(\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. Nodes are represented as $v_i \in \mathcal{V}$, and edges are represented as $e_{ij} \in \mathcal{E}$ which combines node v_i and v_j. We also use $e_{v_i v_j}$ to represent edge between node v_i and v_j. Weighted graph is a special type of graph in which every edge is assigned a weight and represented as $w(e_{i,j})$. Indeed, un-

weighted graph is a specific case of a weighted graph where $w(e_{ij}) = 1 \; \forall e_{ij} \in \mathcal{E}$. Hence, all the theory developed for weighted graphs are applicable to unweighted graphs, as well. The common representation for the number of nodes is n and the number of edges is m; hence, we also use this notation.

2.1 Graphical Model for Interactive Image Segmentation

As we explained at the beginning of Section 2, within an interactive image segmentation scenario there should be some prior information. Let us assume that this information is given as a parametric prior model. It should be noted that this selection is for the simplicity of the representation of the resultant model. The formulated mathematical model is also applicable to non-parametric model case; hence, similar network can easily be constructed for the non-parametric case. Let us consider that there exists a set of parameters in a vector form θ. These parameters can be in the form of any global color, texture or shape model like Gaussian mixture models (GMM) or kernel density estimation (KDE). In addition to such a prior information, both biological model of ganglion [37] and image sensor geometry suggest that relation between nodes should be in immediate neighbours level. Hence, we can safely use the Markovian property. In other words; given the parameter vector θ and its immediate neighbours, label of any pixel is independent of the rest of the image. Graphical model satisfying the Markovian property is defined as Markov random field (MRF). Hence, the dependency graph suggested by interactive image segmentation problem is an MRF. The resultant graphical model is illustrated in Figure 2.1.

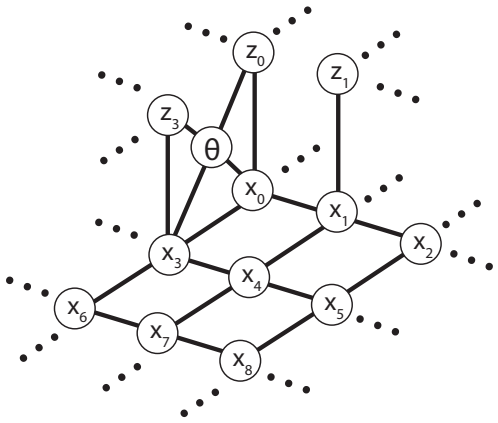


Figure 2.1: Dependency network for the image segmentation problem. In order to make the figure more comprehensible, some edges are not drawn. Node θ is connected to all x_i's and z_i's. Moreover, some of the z_i's are discarded. Every label node x_i has an observation node z_i. It should also be noted that, edges are given as undirected nodes; however, directions can also be added.

In the representation given by Figure 2.1, x_i's are the labels of each node v_i, z_i is the feature vector (color pixel value or extracted texture information) of node v_i, and, θ is the parameter vector of the prior parametric model of the foreground and the

background. Furthermore, labels are binary values as $x_i \in \{0, 1\}$ such that 0 represents the background and 1 represents the foreground.

MRF models are generally factored using graph structure and inference problems are solved using this factorization [59]. On the other hand; there are many cycles in the resultant dependency network. Hence, it is not reasonable to use factorization over the dependency network. Instead, we can use the Hammersley–Clifford theorem [16]. The Hammersley-Clifford theorem states that probability distribution defined on a Markov network is strictly positive if and only if it is a Gibbs random field [16]. Indeed, Gibbs random fields are a specific form of Markov random field which has a clique factorization whose probabilities are formulated as a Gibbs measure.

In our formulation, every labelling of the graph nodes is possible; therefore, probability of every state of the resultant network is strictly positive. Non-negativity of the probabilities of the states suggest a Gibbs random field and Gibbs measure for the interactive image segmentation problem. Gibbs random fields have the following clique factorization. In other words, probability of any state can be written as the product of probabilities over cliques as

$$p(x|z, \theta) = \prod_{c \in \mathcal{C}} F_c(x_c|z_c, \theta). \tag{2.1}$$

In this notation $\mathcal{C}$ represents the set of maximum cliques of edges. Clique is a fully connected set of nodes, and maximal clique for an edge is a clique including that specific edge and having maximum number of nodes. In our representation there are two types of cliques, cliques having two immediate neighbour nodes as x_i, x_j, and cliques including θ, x_i and z_i. Since distribution is a Gibbs random field, $p(x|z, \theta)$ is also a Gibbs measure. A Gibbs measure can be written as an exponential energy function as,

$$p(x|z, \theta) = \frac{1}{Z} e^{-E(x,z,\theta)}. \tag{2.2}$$

In this notation, Z represents the normalization term, and $E(x, z, \theta)$ is a Gibbs energy defined over cliques. Hence, maximizing $p(x|z, \theta)$ becomes equivalent to minimizing $E(x, z, \theta)$. Indeed, since $E(x, z, \theta)$ is a Gibbs energy, it can also be decomposed over cliques as,

$$E(x, z, \theta) = \sum_{v_i \in \mathcal{V}} U(x_i, z_i, \theta) + \sum_{e_{ij} \in \mathcal{E}} V(x_i, x_j). \tag{2.3}$$

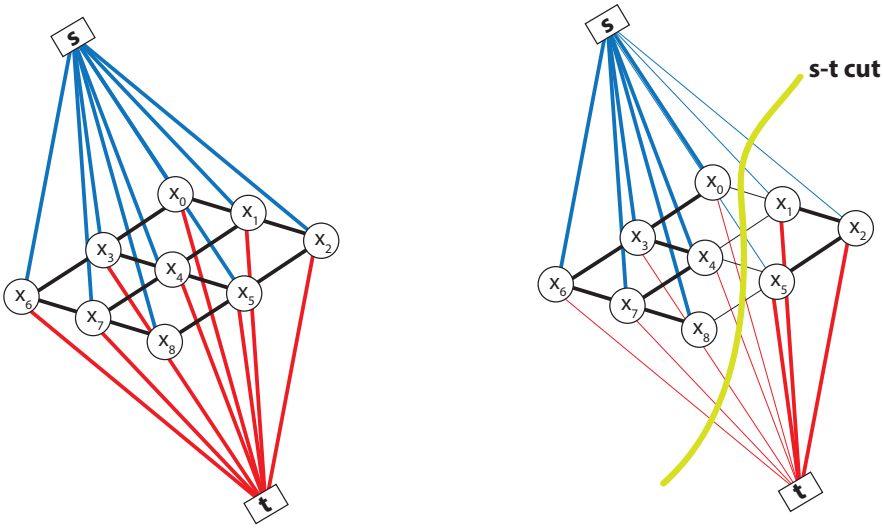
In this representation, U represents the unary energy defined over (x_i, z_i, θ) cliques and V represents the binary energy defined over (x_i, x_j) cliques.

We are interested in the state (labels of all nodes) having the highest probability among all states; i.e. MAP (maximum posterior) solution. Moreover, this state is the state minimizing the Gibbs energy. At this point, we define our fundamental optimization problem. In the next section, we will construct a special type of a graph which will convert the defined optimization problem into a combinatorial optimization problem.

2.2 Relation Between MAP Estimation Problem and Min-Cut/Max-Flow Problem

General minimization of MRF energies is NP-hard (Non-deterministic polynomial hard) and can not be solved in polynomial time [42]. On the other hand, there exist approximate solutions which finds a non-optimal solution in polynomial time such as simulated annealing [40] or belief propagation [78]. On the other hand, specific instances of the minimization problem in (2.3) can be solved in polynomial time [42]. Graph representable binary energies are one of the important examples of these sub problems which we can find optimal solution in polynomial time. Within the scope of these thesis, we are interested only in graph representable energies of the form given by (2.3). We first give the necessary and sufficient condition for binary energy function to be graph representable, and, construct a graph representation of the binary energy function. Before moving to details of the representation, let us define a graph representable energy function.

For the binary energy minimization problem, once can construct a special graph. We introduce two auxiliary nodes to the graph namely source *(s)* and terminal *(t)*. Then, we connect each node to both source and terminal nodes. Finally, we choose edge weights such that energy of any label vector x is equivalent to a specific *s-t cut* in the graph. Any cut of the graph which separates source and terminal is defined as an *s-t cut*. A sample constructed graph and an *s-t cut* is shown in Figure 2.2.



(a) A sample constructed graph (b) A sample *s-t cut* on a constructed graph

Figure 2.2: Visualization of (a) a constructed graph and (b) an *s-t cut*.

Let us consider the following binary energy over a Markovian network:

$$E(x) = \sum_{v_i \in \mathcal{V}} E^i(x_i) + \sum_{e_{ij} \in \mathcal{E}} E^{ij}(x_i, x_j). \tag{2.4}$$

We slightly change the notation, since a constructed graph is much general than the energy function given by (2.3). For this energy minimization problem, Kolmogrov and

Zabih show the necessary and sufficient condition to be graph representable [42]. Any *sub-modular* binary energy function over a Markovian network in the form of (2.4) is graph representable and its global minimum can be found in polynomial time [42]. Sub-modularity constraint can be stated as,

$$E(x \wedge x') + E(x \vee x') \leq E(x) + E(x') \quad \forall x, x' \in \{0,1\}^n. \tag{2.5}$$

For the case of energy functions in the form of (2.4), this constraint is equivalent to

$$E^{ij}(0,0) + E^{ij}(1,1) \leq E^{ij}(0,1) + E^{ij}(1,0) \quad \forall v_i, v_j \in \mathcal{V}. \tag{2.6}$$

Formal proof of the above statement for the binary energy minimization over Markovian network can be found in [42]. Indeed, a general result of sub-modularity is available in [72]. Here, we restate the constructive graph creation for a binary energy function. We limit ourselves to an energy function defined over two binary variables x_1, x_2. Indeed, for any set of binary random variables, one can construct the graph by using the same idea. First, let us state the closed form energy over two variables, as:

$$E(x_1, x_2) = \sum_{i \in 1,2} E^i(x_i) + \sum_{e_{ij} \in \mathcal{E}} E^{ij}(x_i, x_j) \tag{2.7}$$

$$= E^1(x_1) + E^2(x_2) + E^{12}(x_1, x_2). \tag{2.8}$$

While constructing the graph, we consider the unary and binary terms separately by using the additivity theorem [42]. The additivity theorem states that if two energy functions are graph representable, their sum is also graph representable. Moreover, the resultant graph is the graph having edge weights as the sum of corresponding edge weights in the summand graphs. Let us first consider the unary energy of x_1. If x_1 is 0, it should be connected to s node, and cut should have $E^1(0)$ cost, if x_1 is 1, it should be connected to t node and cut should have $E^1(1)$ cost. Similar graph can also be constructed for x_2. Hence, the resultant graph is shown in Figure 2.3.

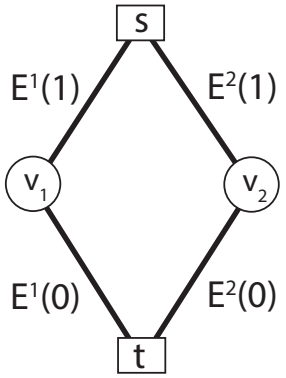


Figure 2.3: Construction of a graph representing unary energy

For the binary energy term, consider the binary energy cost table in Table 2.1.

In this representation, A is a constant term and can be discarded since adding constant term to an optimization problem does not change the solution. Moreover, all the other three terms can be represented as graphs. It should be noted that edge weights

Table2.1: Binary energy factorization for two nodes Markovian network. [42]

$E^{12}(0,0)$	$E^{12}(0,1)$
$E^{12}(1,0)$	$E^{12}(1,1)$

$=$

A	B
C	D

$= A +$

0	0
C-A	*C-A*

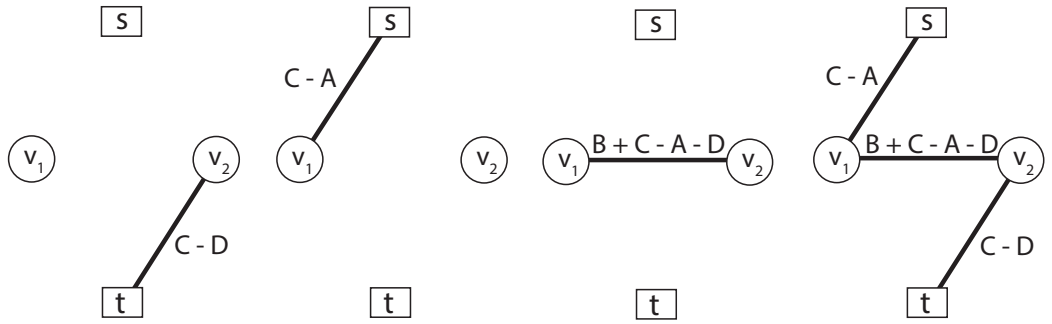
$+$

0	*D-C*
0	*D-C*

$+$

0	*B+C-D-A*
0	0

should be non-negative in order to find min-cut in polynomial time, since majority of the combinatorial algorithms are optimal under the condition of non-negative edge weights. In this factorization, we assume that $D \geq C$ and $C \geq A$. However, if this is not the case, a similar factorization can be achieved via selecting $C - D$ and/or $A - C$ as edge weights. On the other hand, non negativity of $B + C - D - A$ is the result of the sub-modularity (2.6). Hence, the constructed graph of the resultant energy is shown in Figure 2.4.



(a) (b) (c) (d) Final graph

Figure 2.4: Construction of a graph representing binary energy over a two-node Markovian network. (a) is the graph representing the first table in Table 2.1, (b) is the graph representing the second table in Table 2.1 and (c) is the graph representing the third table in Table 2.1. (d) is found via the additivity theorem. It should be noted that $C \geq D$ and $C \geq A$ is assumed; however, similar graph can also be constructed for the inverse inequality cases.

This construction is applicable to any energy function in the form of (2.3) satisfying sub-modularity condition (2.6). By constructing this graph, an energy minimization problem is converted to a problem of finding minimum cut separating source(s) and terminal(t) nodes. Although constructed graph and formulated theory is much more general, we will construct special graph of interest using widely used Ising model [32]. Within the scope of the problems in computer vision, the binary energy term is usually selected as the Ising model [32]. In Ising model, only the label disagreements are punished for the smoothness penalty. In other words, binary energy for the two network node is chosen as in Table 2.2;

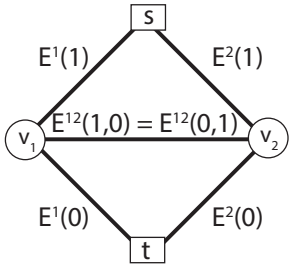
Table2.2: Binary energy factorization for Ising model over two nodes Markovian network.

$E^{12}(0,0)$	$E^{12}(0,1)$
$E^{12}(1,0)$	$E^{12}(1,1)$

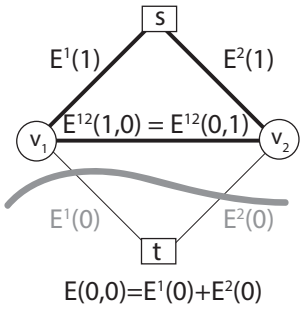
$=$

0	K
K	0

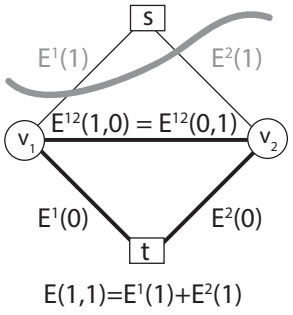
Hence, the constructed graph is much simpler. In Figure 2.5, the constructed graph for the Ising model is visualized and all possible cuts are shown. Extending this two-node graph to a graph of pixels is indeed trivial.



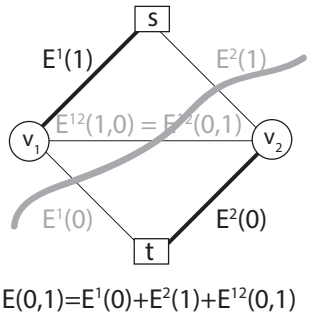
(a) Two nodes graph using Ising model



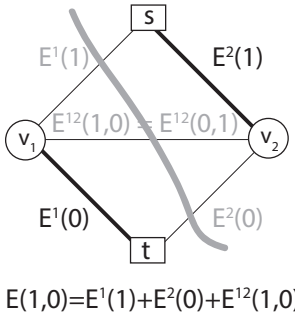
(b) cut corresponding to E(0,0) energy



(c) cut corresponding to E(0,0) energy



(d) cut corresponding to E(0,0) energy



(e) cut corresponding to E(0,0) energy

Figure 2.5: Visualization of the constructed graph for the Ising model [32] and all the possible cuts. In (a) a two-node Ising model Markovian network is shown. In (b), (c), (d) and (e), all possible label configurations are shown and their corresponding cut values are also visualized. It should be noted that the cost of every cut is the corresponding value of the energy function. Hence, the proposed binary energy function in (2.4) is represented with the constructed graph.

2.3 Min-Cut/Max-Flow Algorithms

The problem of interactive segmentation over a Markovian network is converted into finding the minimum cut on a graph, in Section 2.2. How to find a minimum cut on a directed graph is a widely studied problem in combinatorial optimization [31]. To convert an undirected graph into a directed graph in Figure 2.2 is a trivial task: the edges between the source and nodes should be replaced with edges from source to nodes. Furthermore, edges between nodes and terminal should be replaced with the edges from nodes to the terminal. Finally, edges between immediate nodes should be replaced with two edges for each direction. Then, the problem is expected to be transformed into a combinatorial min-cut problem on a directed graph.

The min-cut/max-flow equivalence theorem [31] states that the problem of finding the minimum cut separating the source and the terminal is equivalent to the problem of finding the maximum flow from the source to the sink. When linear programming (LP) definitions of the problems over the directed graph is formulated, the two problems are the dual of each other. The complete proof and formal discussion can be found in [18]. Since formal discussion of min-cut/max-flow is beyond the scope of this thesis, we try to summarize the basic idea behind the theory. We also give the sketch of the specific case of the theorem applicable to our constructed graph.

An *s-t cut* is defined as a set of edges which separate source and terminal nodes, completely. Separate means that after removing the edges of the *s-t cut*, the source and the terminal would be disconnected. Moreover, two nodes are disconnected means there exist no path connecting the two nodes. On the other hand, the maximum flow problem is the problem of assigning a set of flows to the edges of the graph such that 1)flow of any edge is non-negative and less than edge weight; 2)conversation of flows is not violated. In other words, for any node the sum of the input and output flows are equal; 3)net flow from the source to the sink is maximum. Formally, max-flow problem can be formulated as

$$\begin{aligned} \max_{f(e)} \quad & \sum_{e\in\delta^+(s)} f(e) - \sum_{e\in\delta^-(s)} f(e) \\ \text{s.t.} \quad & \sum_{e\in\delta^+(v)} f(e) = \sum_{e\in\delta^-(v)} f(e) \quad \forall v \in \mathcal{V} - \{s,t\} \\ & f(e) \leq w_e \quad \forall e \in \mathcal{E} \\ & 0 \leq f(e) \quad \forall e \in \mathcal{E}. \end{aligned} \tag{2.9}$$

In this formulation, $\delta^+(v)$ represents the edges leaving node v and $\delta^-(v)$ represents the edges entering to node v. Moreover, $f(e)$ represents the flow of edge e. In our constructed graph, no edge enters to the source node, hence, $\delta^-(s)$ is empty set and $\sum_{e\in\delta^-(s)} f(e)$ is 0 and can be discarded. $\sum_{e\in\delta^+(v)} f(e) = \sum_{e\in\delta^-(v)} f(e) \ \ \forall v \in \mathcal{V} - \{s,t\}$ represents the conservation of flows. Equality states that input flow to any node is equal to output flow, except the source and the terminal nodes. Finally, $f(e) \geq 0 \ \ \forall e \in \mathcal{E}$ is for the non-negativity of the flows and $f(e) \leq w_e \ \ \forall e \in \mathcal{E}$ states that flows are bounded with the edge weights.

The main intuition behind the equivalence of problems (2.3) and (2.9) can be summarized via a *residual graph*. Indeed, residual graph formulation is widely used in most of the contributions in this thesis. After defining a set of flows over a graph (not necessarily maximum flow), a residual graph is the graph having the same structure as the original graph with the following edge weights:

$$r_e = w_e - f(e) \quad \forall e \in \mathcal{E}. \tag{2.10}$$

In this representation r_e represents the weight of the edge in the residual graph. It should also be noted that definition of the flow implies that r_e would be non-negative since $f(e) \leq w_e$.

After defining the residual graph, one can state the main claim behind the equivalence theorem. Assume that the max-flow solution to (2.9) is obtained, then the source and the terminal should be disconnected, when the edges having $r_e = 0$ are removed.

Sketch of this proof is rather straightforward, i.e., if there exist a source to terminal path having non-zero residual weights, minimum of these residual weights could be added to the flows without violating any constraint. Hence, the flow would not be maximum and will result in a contradiction. Clearly, there is no path between the source and the terminal; moreover, edges satisfying r_e is the minimum cut, since the cost of this cut is equal to 0.

The final step is to show that resulting minimum cut founded after pushing the maximum flow is equal to the minimum cut of the initial graph. We show only this step for our specific constructed graph topology, although the claim is much general. The general proof can be found in [31]. We show that pushing any flow through the available graph only adds a constant term to the optimization problem. Hence, pushing any flow does not change the solution. We show this claim by the analysis of our two-node Markovian network. There are two types of flows and we will analyse them separately. First type of flow is from source to sink through single node and the second type is from source to sink through two nodes. Both of them are visualized in Figure 2.6

For the case of flows going through a single node as shown in Figure 2.6, $E'^{i}(0) = E^{i}(0) - \alpha$ and $E'^{i}(1) = E^{i}(1) - \alpha$ in the resultant graph. Hence, $E'(x) = E(x) - \alpha$. This result shows that pushing any valid flow from source to sink through a single node only adds a constant term to the optimization problem. Hence, it does not change the solution.

For the case of flow going through two nodes; as visualized in Figure 2.6, the energy function is updated as;

Table2.3: Energy function after pushing flow over two nodes Markovian network.

$E'^{12}(0,0)$	$E'^{12}(0,1)$
$E'^{12}(1,0)$	$E'^{12}(1,1)$

$=$

$E^{12}(0,0)+\alpha$	$E^{12}(0,1)+\alpha$
$E^{12}(1,0)+\alpha$	$E^{12}(1,1)+\alpha$

$=$

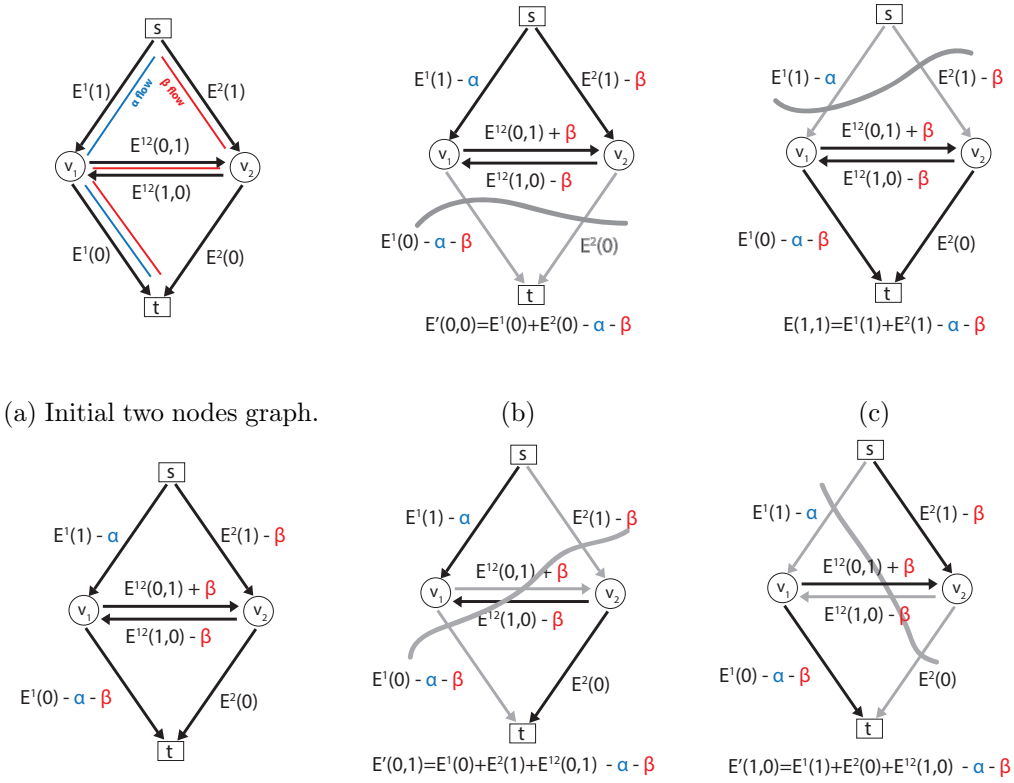
$E^{12}(0,0)$	$E^{12}(0,1)$
$E^{12}(1,0)$	$E^{12}(1,1)$

$+\ \alpha$

Hence, $E'(x_1, x_2) = E(x_1, x_2) + \alpha$. And, this concludes that pushing any flow from source to sink only adds a constant term to the optimization problem. Therefore, the minimum cut before pushing the set of flows and after pushing the set of flows are equivalent.

To find the minimum cut of a resultant residual graph, after finding the maximum flow is straightforward. Union of edges having $r_e = 0$ should be selected as the minimum cut since it has a 0 cost. Hence, solving for the maximum flow from the source to the terminal inherently solves the problem of finding the minimum cut separating the source and the terminal for our graph of interest. It should be noted that we only show this equivalence for our specific graph of interest. However, this claim is much more general. Furthermore, its proof and some applications can be found in [18].

An interactive segmentation formulation over a Markovian network results in a Gibbs energy minimization procedure, and Gibbs energy minimization is equivalent to the problem of finding a minimum cut over a specially constructed graph. Moreover, to find the minimum cut is equivalent to the problem of finding the maximum flow. Hence, we finally convert the problem of interactive segmentation to the problem of finding the maximum flow for which there exist many polynomial time, globally optimum algorithms to find the solution. Some of these algorithms and their corresponding



(a) Initial two nodes graph. (b) (c)

(d) After pushing flows. (e) (f)

Figure 2.6: Visualization of two types of flows applicable to a two-node Markovian network. α represents the flow from the source to the terminal via single node and β represents the flow from the source to the terminal via two nodes. Flows and original graph are shown in (a). Flows are pushed and residual graph is visualized in (b). In (c), (d), (e) and (f), costs of the cuts for all possible labels are shown. For all label selections, the values of α and β are subtracted from the cost values. Hence, the resulting optimization problem is added only a constant value $(-\alpha-\beta)$ i.e. $E'(x) = E'(x) - \alpha - \beta$. Therefore, the solution to the minimum cut problem does not change after pushing any valid flow from the source to the terminal.

computational complexities are summarized in Table 2.4 in a chronological order. In complexity formulas, n represents the number of nodes of the graph, and m is the number of the edges of the graph and U is the bound of the edge weights.

Table2.4: Computational Complexities of Some of the Existing Max-flow Algorithms.

Due to	Computational Complexity
Ford & Fulkerson [30]	$O(nmU)$
Edmonds & Karp [26]	$O(nm^2)$
Dinic [23]	$O(n^2m)$
Goldberg & Tarjan [33]	$O(nmlog(n^2/m))$
Orlin [56]	$O(nm)$, $O(n^2/logn)$ if $m = O(n)$

The algorithms mentioned in Table 2.4 are general algorithms applicable to all types of graphs. Furthermore, their complexities are the worst case complexities. However, the graphs arising in the computer vision problems are 2D grids or 3D grids. Hence, topology of the graph should be taken into consideration for the selection of the max-flow algorithms. According to the experimental comparison presented in [13], Ford & Fulkerson algorithm [30] with an additional heuristic outperforms all other available methods. Hence, within the scope of this thesis we only explain Furd & Fulkerson algorithm with Boykov & Kolmogorov heuristic.

2.3.1 Ford & Fulkerson Algorithm with Boykov & Kolmogorov Heuristic

Ford & Fulkerson algorithm is the straightforward and basic extension of the equivalence proof. Since pushing any flow does not change the solution, one can push as much as possible flows via arbitrary selected paths until max-flow is reached. Furthermore, definition of the maximum flow is a combination of set of flows until there exist no possible flow in the graph. Hence, the main idea of the algorithm is to start with an empty flow and to find any valid flow from the source to the terminal until there exist no possible flow in the graph. In other words, the algorithm can be summarized as follows:

Algorithm 1 Ford & Fulkerson Algorithm.

1: **Initialization:** $f(e) = 0 \;\; \forall e \in \mathcal{E}, r(e) = w(e) \;\; \forall e \in \mathcal{E}$
2: **while** $\exists p(s\text{-}t \textit{ path})$ s.t. $r(e) > 0 \;\; \forall e \in p$ **do**
3: Chose a valid $p(s\text{-}t \textit{ path})$ s.t. $r(e) > 0 \;\; \forall e \in p$
4: Push the maximum possible flow through the $p(s\text{-}t \textit{ path})$. i.e.
5: $f_{max} = \min_{e \in p} r(e)$
6: $f(e) = f(e) + f_{max} \;\; \forall e \in p.$
7: $r(e) = r(e) - f_{max} \;\; \forall e \in p.$
8: **end while**

Computation time of Ford & Fulkerson algorithm significantly depends on the selection of the *s-t path*. Hence, selection of the *s-t path* requires some heuristics and prior knowledge of the graph topology. One of the widely used ideas is the shortest path heuristic of Dinic [23]. It starts with finding a shortest path from the source to the terminal over at most k edges via breadth first search [22]. When there exist no

valid path having k edges, k is increased to $k + 1$. k is generally started with a small number. In terms of the search strategy, Dinic's heuristic is similar to Iterative Deepening method widely used in graph search [44]. This shortest path selection makes the worst case complexity of the algorithm $O(mn^2)$.

The main computational drawback of this algorithm is performing search over trees from scratch at every iteration. Boykov & Kolmogorov heuristic is mainly developed to solve this redundancy. It starts with a tree expansion until an *s-t path* is found. After the augmentation stage, Boykov & Kolmogorov heuristic recover the tree. The main drawback of this heuristic is the lack of shortest path guarantee since recovering some nodes may change the expansion order of the nodes and violate the optimality of breadth first search [22]. The worst case computational complexity of the algorithm becomes $O(mn^2C)$ where C is the minimum cut. On the other hand, the experimental comparison suggests that average run time of Boykov & Kolmogorov heuristic outperforms other existing algorithms [13]. Hence, Ford & Fulkerson algorithm with Boykov & Kolmogorov is the main method of interest in this thesis.

Since the main idea of the Ford & Fulkerson algorithm using Boykov & Kolmogorov heuristic is the re-usage of the trees, nodes need to be tracked throughout the iterations. Nodes in the graph are divided into three groups: active nodes, passive nodes and free nodes. Furthermore, two trees are expended and kept throughout the iterations: one rooted at the source and the one rooted at the terminal. The nodes which are not a member of either trees are called as free nodes, and the nodes at which trees can continue to grow are called as active nodes [13]. Finally, internal nodes are denoted as passive nodes. These nodes are visualized in a sample graph in Figure 2.7.

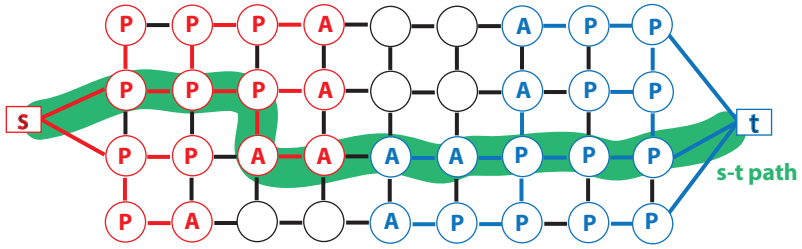


Figure 2.7: Visualization of active, passive and free nodes. Active nodes are the nodes in which trees might grow via inserting free nodes to the trees. Passive nodes are the internal nodes of the trees where there is no free node neighbour. Finally, free nodes are the nodes not included in any tree. When two nodes, one from source tree and one from terminal tree, become neighbours, an *s-t path* can obtained via following parents of the neighbour nodes. A sample *s-t path* is illustrated in the green path.

The algorithm simply expands both trees until two active nodes of trees become neighbours; i.e. trees touch each other. Then, an *s-t path* from the source to the terminal is computed and augmented. The augmentation is simply the update of the flow values and the residual weights of the graph according to the selected *s-t path* and maximum possible flow value over the path. After the augmentation stage, trees become forest since some internal edges may saturate, i.e., they have zero residual edge weight. Hence, some of the nodes loose their parents. Moreover, these nodes are called as orphan nodes [13]. Finally, the trees are recovered from the forests. Let us represent trees as S and T, set A as active nodes and set O as orphan nodes. Then, the resulting

algorithm can be summarized as in Algorithm 2:

Algorithm 2 Ford & Fulkerson Algorithm with Boykov & Kolmogorov Heuristic.

Initialization: $S = s,\ T = t,\ A = s, t,\ O = \varnothing,\ f(e) = 0\ \ \forall e \in \mathcal{E},\ r(e) = w(e)\ \ \forall e \in \mathcal{E},$
while $\exists p$(*s-t path*) **do**
  Grow: Expand S and T until $\exists v_i \in S, v_j \in T, v_i \in \mathcal{N}(v_j)$
  Finds the *s-t path* via v_i and v_j.
  Augment: Updates residual weights
  $f_{max} = \min_{e \in p} r(e)$
  $f(e) = f(e) + f_{max}\ \ \forall e \in p.$
  $r(e) = r(e) - f_{max}\ \ \forall e \in p.$
  Adopt: Find a valid parent for orphan nodes.
end while

Grow stage is accomplished via breadth-first [22] manner. At each stage, for each active node, free neighbour nodes are added to the corresponding trees and the parents are set accordingly. This stage is continued until there exist two nodes, one from the source tree and one from the terminal tree, which are immediate neighbours.

After the growth stage, an *s-t path* is obtained by starting with neighbour nodes of S and T trees and adding parents of them to the path, incrementally. When the roots (s and t) are added to the path, the *s-t path* would be obtained. Then, the minimum residual weight among these edges need to be selected as the maximum possible flow through the path. Finally, this path is pushed through the path. The residual weights and flow values are updated according to the pushed flow.

Adoption stage actually requires more elaboration than the growth and augmentation stages. The main idea of the algorithm is to try to find a proper parent for each orphan node. If the proper parent is not determined, node is left as a free node. Hence, the resultant algorithm is summarized in Algorithm 3.

In Algorithm 3, *tree*(.) is a function which returns the current state of the node. In other words, it checks whether the node is active(A), passive(P) or orphan(O). The algorithm processes every orphan node; it first tries to find a neighbour node having a non-negative residual edge weight. If selected parent has a different state than the orphan node in terms of active/passive/free status, the orphan node is added as an active node. If not the status of the node is not changed.

Details on the implementation of the Ford & Fulkerson method with Boykov & Kolmogorov heuristic can be found in [13]. Different algorithms and heuristics for max-flow are also compared experimentally for computer vision applications in [13]. Within the scope of this thesis some novel improvements are introduced over Ford & Fulkerson method with Boykov & Kolmogorov heuristic and an extensive computation time analysis is performed in Chapters 3 and 4.

Algorithm 3 Finding parents for orphan nodes.

```
1: for all p ∈ O do
2:   for all q ∈ 𝒩(p) do
3:     if q ∈ S and r(e_pq > 0) then
4:       S = S ∪ p
5:       parent(p) = q
6:       if tree(q) ≠ tree(p) then
7:         A = A ∪ p
8:       end if
9:     end if
10:    if q ∈ T and r(e_qp > 0) then
11:      T = T ∪ p
12:      parent(p) = q
13:      if tree(q) ≠ tree(p) then
14:        A = A ∪ p
15:      end if
16:    end if
17:    if no valid parent (q) is founded then
18:      tree(q) = free
19:    end if
20:  end for
21:  O = O − p
22: end for
```

CHAPTER 3

INTERACTIVE IMAGE SEGMENTATION

Extracting the object of interest from the non-trivial background is a crucial step in many multimedia applications. Although, fully automatic image segmentation algorithms have been improved significantly, it is still not possible to apply an automatic image segmentation algorithm with a guaranteed performance in a general setup. Therefore, interactive image segmentation algorithms are becoming more popular for commercially related applications. [12, 60, 49, 35, 20].

Problem of interactive image segmentation has many perspectives. First of all, there is an interaction part, which need to be handled considering application device and target audience. After the interaction stage, there are variety of algorithms depending on the required computational complexity and the segmentation accuracy to model the features of the image and perform segmentation. Selection of the algorithm heavily depends on the application scenario. Before going into the details of the proposed method, some of the options for interaction, modelling and segmentation are summarized in the following section.

3.1 Related Work

Interactive image segmentation work flow can be divided into three main parts. First, user interacts with an image to be segmented. Then, some low level features, such as color, texture and shape of foreground and background are modelled. Next, an optimization problem defined via selected features. Finally, optimization problem is solved to obtain final segmentation. Hence, the existing algorithms are analysed in terms of these three sub-blocks, namely interaction, modelling and optimization, in the following section. Some options for each of these sub-blocks have been summarized in Figure 3.1.

There exist three main methods for user interaction, to select a bounding box around the object of interest [60, 34], to draw scribbles on foreground and/or background [55, 12, 14, 50], and to draw approximate boundary of the object of interest [49, 54] are the most common and widely used interaction methods in the literature. Each of these interaction methods have their own advantages and disadvantages, and their usage depends heavily on the application scenario. For example, professional image editing applications require near optimal segmentation quality; hence, computational complexity drawback can be tolerated. On the other hand, consumer electronic appli-

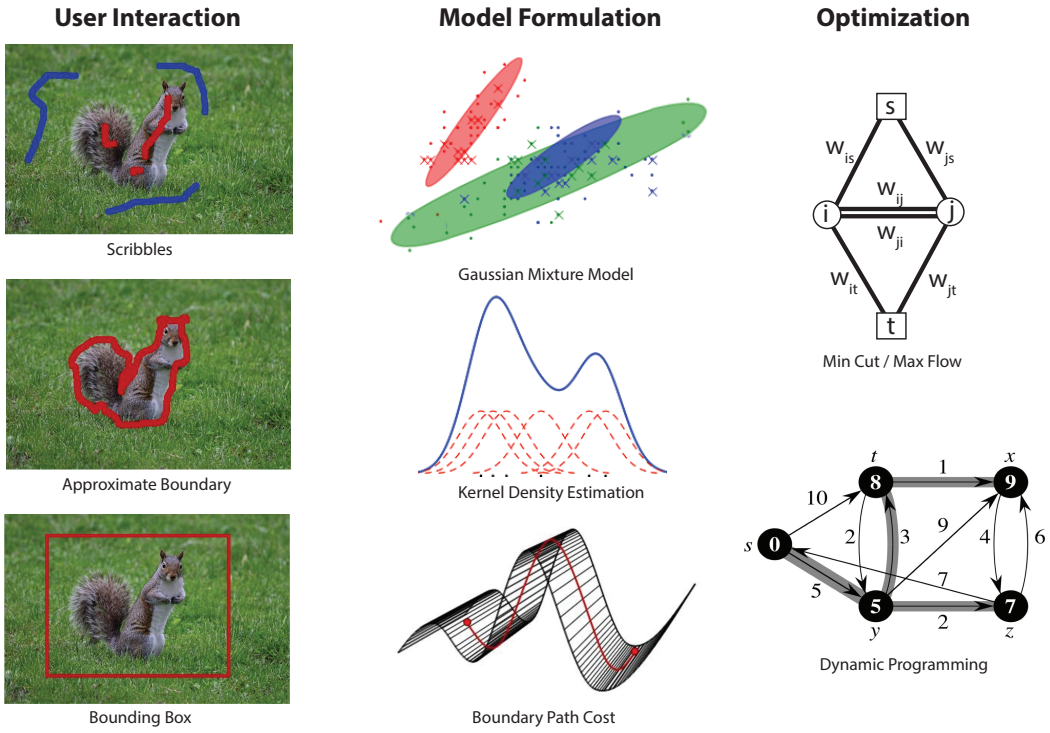


Figure 3.1: Main sub-blocks used in any interactive image segmentation algorithm. Selection of the interaction method, model formulation and optimization method depends on the application. Most of the available interactive image segmentation methods in the literature are different combinations of these mentioned sub-blocks. For example, combination of Bounding Box, GMM and Min-Cut/Max-Flow is the Grabcut algorithm [60]. On the other hand, combination of approximate boundary, path cost and dynamic programming is intelligent scissors algorithm [54].

cations require near real-time performances and robustness to interaction errors.

Within the scope of this thesis, attention is mainly given to interactive mobile multimedia applications. Moreover, literature on mobile interactive image segmentation is quite limited. Indeed, almost all the mobile methods are the extensions of classical interactive image segmentation algorithms. Hence, they do not satisfy the interaction quality requirement of mobile multimedia applications. In this thesis, user interaction is revisited and a novel interaction method is proposed and explained in Section 3.2.1.

The energy function is modelled via set of low level features depending on the optimization method and the interaction method. For example; if scribbles or bounding boxes are used for interaction, global models are favoured, since there exist enough global information. On the other hand, if an approximate boundary used as an interaction method, the resulting problem is inherently local. Hence, local costs such as cost of the boundary separating foreground and background is utilized. Within existing models in the literature, Gaussian Mixture Models (GMM) is mainly used as the global model and boundary path cost is mainly used as the local model. Usage scenario of GMM in segmentation problems is explained in Section 3.1.2, whereas boundary cost

approach is explained in Section 3.1.3. Moreover, optimization method is also a constraint for the selection of the low level models. For example, some family of functions -submodular functions- can efficiently be minimized via min-cut/max-flow methods. Hence, using such functions for foreground/background modelling is desired.

For the optimization sub-block, dynamic programming is used for local models. Since interaction is approximate boundary, one can assume that true boundary is close to the interacted boundary. Therefore, global optimum is close to the initial point. We can assume that energy function is convex around the interacted boundary and local methods results in global optimum point. However, erroneous user interaction, small size of mobile screens and high amount of color variation and/or noise makes this assumption invalid and the local search finds a local minimum of the energy function. Detailed theoretical and experimental analysis of local search is also performed in Section 3.1.3 and Section 3.3.

On the other hand, global models can be solved optimally, if some constraints are introduced like sub-modularity of the energy function. Although some local optimization techniques, such as simulated annealing [40] and belief propagation [78] methods are utilized for the interaction image segmentation problem, extensive analysis of the probabilistic modal make the global optimization possible for most of these cases. As explained in Chapter 2, it is possible to find the global optimum of the MRF energy defined over the image via an efficient combinatorial algorithm, namely min-cut/max-flow algorithm. Indeed, quality difference between local and global methods are significant due to the erroneous user interaction, small size of mobile screens and high amount of color variation and/or noise [69].

Grabcut [60] and intelligent scissors [54] algorithms are two main example of global and local methods. Moreover, their detailed analysis explain the details of all main sub-blocks used in the literature and stated in Figure 3.1. Furthermore, there exist some heuristic based region grow/merge type methods which achieve high quality segmentation. Although heuristic methods are out of the scope of this thesis, they will be summarized for the sake of completeness.

3.1.1 Region Grow/Merge Type Methods

User interaction can actually be formulated as providing seeds to an algorithm. Consider the interaction type of bounding box or scribbles; clearly scribbles are the seeds for foreground or background. On the other hand, bounding box of an object is also a seed for the background. Naive idea is to start with this seeds and to apply a region grow procedure [27] by using a fixed or an adaptive threshold. However, finding a robust threshold or an adaptive algorithm is not possible in general. Hence, algorithms developed via this perspective are ad-hoc methods relying on a greedy metric.

There exist a strong relationship between graph-based methods and region grow-merge like methods. A metric which needs to be optimized via region grow/merge operations can be formulated as a measure on a grid graph defined over image pixels. For example, cutting weakest edge at each iteration is just a local search to find the minimum cut of the graph. Indeed, most of the region grow/merge type of algorithms are just greedy versions of the graph clustering algorithms. Moreover, it might be possible to solve

the resultant optimization problem globally. On the other hand, if you consider the max-flow algorithm explained in Section 2.3, at each iteration a maximal flow is pushed through an *s-t path* and at least one of the edges of the path is saturated; i.e. removed from the graph. This operation can be observed as a region grow/merge type move with guaranteed global optimality. Hence, most of the region grow/merge like methods in the literature are mainly replaced with graph theoretical methods.

On the other hand, greedy algorithms are quite useful in graph theory when their convergence and optimality is properly considered [18]. It is a widely used approach to use greedy algorithms to approximately solve many NP-Hard problems in the literature. Main advantage of greedy algorithms is their low computational complexity. When the size of the problem is large, greedy algorithms might be the only reasonable solution [18]. Indeed, some greedy algorithms such as Kruskal's minimal spanning tree algorithm [45], are proven to be optimal. Hence, with proper selection of the greedy move algorithm, it is possible to achieve an acceptable segmentation accuracy in almost real-time. For example, maximal similarity based region merge algorithm [55] is a high quality segmentation method which relies on this formulation.

3.1.1.1 Maximal Similarity Based Region Merge

Maximal similarity based region merge [55] starts with a highly redundant representation of the image (set of oversegments) and merge every node with its maximally similar node, if it is also the maximally similar node of the node to be merged. This definition of maximal similarity is more powerful than the thresholding methods. As a similarity measure, Bhattacharyya coefficient [9] of the histograms is used. Moreover, initial over-segmentation is obtained via the mean shift algorithm [17].

One should first define the similarity measure used in the method, namely *Bhattacharyya coefficient*. Initially features are extracted from the regions in terms of RGB histograms. RGB space is divided into 16x16x16 = 4096 bins. Then, a histogram consisting of 4096 bins is extracted from the regions. Let the normalized histogram of the region R be represented as $Hist_R$. Similarity measure between histograms is selected as *Bhattacharyya coefficient*. Formally;

$$\rho(R,Q) = \sum_{u=1}^{4096} \sqrt{Hist_R^u \times Hist_Q^u}. \tag{3.1}$$

Geometric interpretation of *Bhattacharyya coefficient* is the cosine between the angles of unit vectors

$$\left(\sqrt{Hist_R^1} \dots \sqrt{Hist_R^{4096}}\right)^T \text{ and } \left(\sqrt{Hist_Q^1} \dots \sqrt{Hist_Q^{4096}}\right)^T.$$

Hence, *Bhattacharyya coefficient* is higher when regions are similar to each other and lower if they are different. Extraction of RGB histograms requires high computation time. However, we consider it as a preprocessing step. Moreover, one can say that algorithm works near real-time. After the merge operations, the histograms are updated. However, histogram updates can efficiently be handled since histograms are monoids [38].

The maximal similarity based region merge algorithm starts with the over-segmentation of the image via the meanshift [17] algorithm. After over-segmentation, interaction is performed via drawn scribbles for foreground and background. Then, region merge operations are handled for each superpixel separately. Initially, a merge operation is performed for each superpixel in the foreground set. Then, for each superpixel in the background set, a merge operation is performed. Finally, the merge operation is performed for all other superpixels. These processes are iteratively repeated throughout the segmentation until there is no possible merge operation in the image. Indeed, convergence of the maximal similarity based merging is proven in [55].

The merging operation for a superpixel is performed by using maximum similarity. Assume that a candidate superpixel is being searched to be merged with a superpixel R. Initially, we need to find a neighbour superpixel Q which has the highest similarity to R. In other words;

$$Q = MaxSim(R) = \underset{S \in \mathcal{N}(R)}{\arg\min}\, \rho(R, S). \tag{3.2}$$

Then, we check the maximum similarity condition. If the region having highest similarity to Q is also R then we merge them. In other words,

$$T = MaxSim(Q) = \underset{S \in \mathcal{N}(Q)}{\arg\min}\, \rho(Q, S) \tag{3.3}$$

If $T = R$, then Q and R is merged. Moreover, maximal similarity based region merging algorithm can be summarized in Algorithm 4.

Algorithm 4 Maximal Similarity Based Region Merging [55].

$\mathcal{F} = \emptyset, \mathcal{B} = \emptyset$
Over-segment the image
Make user draw scribbles on foreground and background (populate $\mathcal{F}, \mathcal{B}$)
while $\exists$Merge Operation **do**
  for all $v_i \in G \cap (\mathcal{F} \cup \mathcal{B})$ **do**
    $v_j = \arg\max_{v \in \mathcal{N}(v_i)} \rho(v_i, v)$
    $v_k = \arg\max_{v \in \mathcal{N}(v_j)} \rho(v_j, v)$
    if $v_k = v_i$ and $(v_i, v_j \in \mathcal{F}$ or $v_i, v_j \in \mathcal{B})$ **then**
      Merge $v_i \& v_j$
    end if
  end for
  for all $v_i \in G \cap (\overline{\mathcal{F} \cup \mathcal{B}})$ **do**
    $v_j = \arg\max_{v \in \mathcal{N}(v_i)} \rho(v_i, v)$
    $v_k = \arg\max_{v \in \mathcal{N}(v_j)} \rho(v_j, v)$
    if $v_k = v_i$ and $(v_j, v_i \notin \mathcal{F}$ or $v_j, v_i \notin \mathcal{G})$ **then**
      Merge $v_i \& v_j$
    end if
  end for
end while

Maximal similarity based region merging algorithm [55] requires computationally expensive preprocessing due to the histogram extraction and the meanshift algorithm; therefore, it is not desirable in mobile scenarios. Indeed, in this method, user is required to give interaction input for both foreground and background. As explained in

(a) Interaction for Lemon Data [55] (b) Output for Lemon Data [55] (c) Interaction for Girl Data [55] (d) Output for Girl Data [55]

(e) Interaction for Mona Lisa Data [55] (f) Output for Mona Lisa Data [55] (g) Interaction for Toucan Data [55] (h) Output for Toucan Data [55]

Figure 3.2: Interactions and segmentation results for Maximal Similarity Based Region Merging algorithm [55]. Interactions are shown by using the color notation of blue as background and green as foreground. Oversegment boundaries are also visualized.

Chapter 1, in mobile touch screen scenarios this type of interaction is not favoured. Therefore, we did not include maximal similarity based merging algorithm during the experimental comparisons. However, to give an idea about the segmentation accuracy of the algorithm, we include some results of the algorithm in Figure 3.2.

3.1.2 Grabcut

Grabcut algorithm [60] is based on the iterative estimation of global color models and the solution of min-cut/max-flow. Furthermore, Grabcut algorithm [60] is the first algorithm that uses only background information. Algorithm can efficiently and accurately segment an image given only bounding box of the object of interest. Grabcut algorithm [60] is also the first algorithm that utilizes color statistics via Gaussian mixture models (GMM).

Grabcut [60] algorithm is formulated on a pixel-based representation. As explained in Section 2, oversegmenting an image into a set of superpixels is a common technique in the literature. Indeed, for graph theoretical approaches, there is no difference between superpixel-based and pixel-based representations in terms of graph formulation and optimization. Hence, it is reasonable to formulate the algorithms in the superpixel domain to satisfy the efficiency requirement of mobile devices. Therefore, Grabcut

algorithm [60] will be explained in a superpixel setting, although the original algorithm is based on a pixel representation.

Grabcut [60] starts with a bounding box of the object of interest supplied by the user. The outside of this box is considered as background and GMM for the background is estimated. Then, inside of the box is used as the foreground model. After this coarse initialization, min-cut/max-flow based binary segmentation is performed. After the segmentation, the resultant foreground and background regions are used to estimate the GMM's; and, the segmentation procedure is applied iteratively. The process is finalized, when the background/foreground region converges and does not further change throughout the iterations.

An input image is represented as a color vector of the form $\mathbf{z} = (z_1, ..., z_n, ..., z_N)$. In this representation, z_i is the concatenated color vectors of the pixels of superpixel i. Then, segmentation of the image is represented as a binary vector of form $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_n, ..., \alpha_N)$ with $\alpha_i = 1$, if the superpixel i is foreground and $\alpha_i = 0$, if the superpixel i is background.

GMM is used to model the color information; therefore, there is also an additional GMM vector, which assign a unique GMM to each superpixel $\mathbf{k} = (k_1, ..., k_n, ..., k_N)$. The parameters of the GMM for the color values are also stored in a vector θ. Then, the energy function, whose minimum corresponds to a segmentation result that is guided by color models and having coherent foreground and background, is formulated. This energy is in the form of a Gibbs energy [32] and it is represented as

$$\mathbf{E}(\boldsymbol{\alpha}, \theta, \mathbf{z}, \mathbf{k}) = U(\boldsymbol{\alpha}, \theta, \mathbf{z}, \mathbf{k}) + V(\boldsymbol{\alpha}, \mathbf{z}).$$

In this energy function, $U(\boldsymbol{\alpha}, \theta, \mathbf{z}, \mathbf{k})$ corresponds to a fit measure of the image data $\mathbf{z}$ by the estimated parameters θ and $\mathbf{k}$ to the segmentation mask $\boldsymbol{\alpha}$. This term is defined as the sum of the fitness terms of each superpixel. Moreover, fitness terms of each superpixel is represented by the mean of fitness of its pixels. Hence, if z_{n_i} represents the color vector of i^{th} pixel of n^{th} superpixel and $\|z_n\|$ represents the number of pixels in the superpixel n, $U(\boldsymbol{\alpha}, \theta, \mathbf{z}, \mathbf{k})$ can be written as

$$U(\boldsymbol{\alpha}, \theta, \mathbf{z}, \mathbf{k}) = \sum_n \frac{1}{\|z_n\|} \sum_{i \in n} D(\alpha_n, \theta, z_{n_i}, k_n),$$

where $D(\alpha_n, \theta, z_{n_i}, k_n)$ is the inverse of conditional-log-likelihood of the color vector z_{n_i} as

$$D(\alpha_n, \theta, z_{n_i}, k_n) = -\log p(z_{n_i} | \alpha_n, k_n, \theta),$$

which up to a constant can be written as:

$$D(\alpha_n, \theta, z_{n_i}, k_n) = \frac{1}{2} \log det \Sigma(\alpha_n, k_n) +$$

$$\frac{1}{2}(z_{n_i} - \mu(\alpha_n, k_n))^T \Sigma(\alpha_n, k_n)^{-1} (z_{n_i} - \mu(\alpha_n, k_n)).$$

On the other hand, $V(\boldsymbol{\alpha}, \mathbf{z})$ corresponds to the coherency of the segmented foreground and background. This term is a typical smoothness term and enforce smooth foreground and backgrounds. Therefore, it is defined as

$$V(\boldsymbol{\alpha}, \mathbf{z}) = \gamma \sum_{(m,n) \in \mathbf{C}} [\alpha_m \neq \alpha_n] e^{-\beta dis(m,n)},$$

where $[\psi]$ is the indicator function, which gives 1, if the ψ is true and 0 if ψ is false. $\mathbf{C}$ represents the neighboring superpixels. Moreover, $dis(m,n)$ represents the distance between superpixels and this distance is taken as the Euclidean distance between mean color vectors of each superpixel. Finally, the constant β is selected as [13]

$$\beta = (2 < dis(m,n) >)^{-1}$$

where $< . >$ denotes average over the entire image. It should be noted that, this coherency is the Ising model explained in Section 2.3.

At any step, current foreground and background regions are used to estimate the GMM parameters, $\theta = (\mu(\alpha,k), \Sigma(\alpha,k))$, by the expectation maximization (EM) algorithm. In the initialization stage, outside of the supplied box is used as a background and inside of the box is used as a foreground.

This energy function is submodular and can be minimized with the min-cut/max-flow method, as explained in Section 2.3. Furthermore, Grabcut [60] is explained in Algorithm 5. It should be noted that Grabcut [60] includes a further correction stage which let the user corrects the erroneous regions by drawing scribbles on them. Since the proposed application area -mobile multimedia- puts a significant emphasis on the user interaction, we did not include any user correction stage for any algorithm.

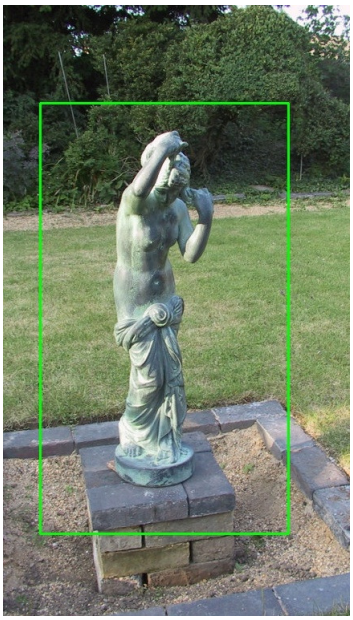
Algorithm 5 Grabcut [60] algorithm without user correction.

1: $F = \emptyset, B = \emptyset$
2: Oversegment the image
3: User draws the bounding box of the object
4: $\mathcal{F} = \{v_i | i \in Box\}$, $\mathcal{B} = \{v_i | i \notin Box\}$
5: **while** $\mathcal{F}$ and $\mathcal{B}$ Changes **do**
6: Expectation Maximization to learn Foreground and Background GMMs (learn θ)
7: Assign most possible gaussian (estimate k_i)
8: Solve min-cut (minimize $\mathbf{E}(\boldsymbol{\alpha}, \theta, \mathbf{z}, \mathbf{k}) = U(\boldsymbol{\alpha}, \theta, \mathbf{z}, \mathbf{k}) + V(\boldsymbol{\alpha}, \mathbf{z})$)
9: $F = \{v_i | \alpha_i = 1\}$, $B = \{v_i | \alpha_i = 0\}$
10: **end while**

In order to visualize the process further, we also include the intermediate results of the algorithm for a sample input. Iteration of the Grabcut is visualized in Figure 3.3. Figure 3.3.a is the interaction supplied by the user in the form of a bounding box. Moreover, subsequent subfigures are the results of the iterations in Figure 3.3.b, 3.3.c, 3.3.d, 3.3.e, 3.3.f.

3.1.3 Intelligent Scissors

Intelligent scissors algorithm [54] is one of the earliest and widely used interactive segmentation methods in the literature. It is also included in a widely used image editing software Adobe Photoshop [67]. Intelligent scissors [54] is a dynamic process. Interaction method of the intelligent scissors [54] is based on an approximate boundary drawn around the object of interest. User starts from some point on the boundary and draws the approximate boundary while moving the mouse. Throughout the interaction,



(a) Interaction

(b) 1st Iteration

(c) 2nd Iteration

(d) 3rd Iteration

(e) 4th Iteration

(f) 5th Iteration

Figure 3.3: Interaction and intermediate segmentation results for Grabcut [60] algorithm. In (a) rectangle supplied by the user is shown. In (b), (c), (d), (e) and (f) segmentation result of the 1st, 2nd, 3rd, 4th and 5th iteration of the algorithm is shown, respectively. Fast convergence of the algorithm is clearly visible.

boundary of the object is computed and updated dynamically. To explicitly add some points to the boundary, user can click on these points. Intelligent scissors algorithm is built on the idea of local optimization. Indeed, the main idea is snapping the drawn approximate boundary to the nearest strong and consistent edge simultaneously and

locally.

In order to explain the algorithm, we first need to define local energy function minimized via dynamic programming. A boundary should have a high gradient magnitude, a consistent gradient direction with the rest of the path and should be at Laplacian zero crossings of the image. Hence, the cost function for the boundary element connecting pixel p and q is the weighted sum of these functionals:

$$l(p,q) = \omega_z f_z(q) + +\omega_d f_d(p,q) + \omega_g f_g(q). \tag{3.4}$$

In this formulation, $f_z(q)$ represents the Laplacian zero crossing such that $f_z(q)$ is 1, if q is at zero crossing and 0 otherwise:

$$f_z(q) = \begin{cases} 0 & \text{if } \bigtriangledown I(q) = 0 \\ 1 & \text{if } \bigtriangledown I(q) \neq 0 \end{cases} \tag{3.5}$$

However, a regular image has very few non-zero Laplacian crossings, if this definition of Laplacian zero crossing is used. Hence, definition is replaced with the sign changes of neighbour nodes. If the sign of the Laplacian of two neighbour nodes are different, the node having the smaller Laplacian is selected as the Laplacian zero crossing of the image.

Laplacian crossing gives equal weights to each edge. The gradient magnitude is used in order to differentiate between the strong and the week edges. f_g represents the gradient magnitude such that,

$$f_g(q) = 1 - \frac{\sqrt{I_x(q)^2 + I_y(q)^2}}{max(\sqrt{I_x^2 + I_y^2})} \tag{3.6}$$

On the other hand, f_d represents the consistency of the gradient direction. Moreover, it is used for the smoothness of the resultant path. Let $D(p)$ be a vector perpendicular to the gradient direction at pixel p such that $D(p) = (I_y(p), -I_x(p))$. Moreover, assume that $L(x,y)$ represents the bidirectional link between pixels p and q such that

$$L(p,q) = \begin{cases} \mathbf{q} - \mathbf{p} & \text{if } \mathbf{q} \geq \mathbf{p} \\ \mathbf{p} - \mathbf{q} & \text{if } \mathbf{p} \geq \mathbf{q} \end{cases} \tag{3.7}$$

Then, the smoothness penalty can be written as

$$f_d(p,q) = \frac{1}{\pi}(\cos[D(p)^T L(p,q)]^{-1} + \cos[L(p,q)D(q)^T]^{-1}). \tag{3.8}$$

This cost function gives high and positive values if the gradient directions are not similar to each other and low values, if they are similar. Then, the problem is to find the minimum path boundary. However, there exist a boundary having zero length (all image is foreground or all image is background). In order to solve this problem, one should use some seed points. While user is moving the mouse, user can select a point as a seed point by clicking on it. Moreover, the selected point is required to be included in the resultant boundary. Then, the problem is equivalent to problem of finding a minimum path going through these seed points. Indeed, this problem can easily be solved via Dijkstra algorithm [22]. Some results of the algorithm can be seen in Figure 3.12, 3.13, 3.14 and 3.15.

3.1.4 Isoperimetric Graph Partitioning

Images and videos inherently impose a grid graph structure. Hence, statistical inference is achieved in terms of energy minimization over grid graphs. Isoperimetric graph partitioning [35] approaches to the problem from a different perspective: an object is defined as the segment having the minimum isoperimetric constant.

Isoperimetric constant is defined as the ratio of a closed surface to an enclosed volume for any regular manifold. In other words, isoperimetric constant of any closed manifold h is;

$$h = \inf_{S} \frac{|\partial S|}{Vol_S} \tag{3.9}$$

Where S is a region on the manifold, Vol_S defines the volume of region S and $|\partial S|$ is area of the boundary of region S. Moreover h is the infimum of the ratio over all possible regions. In other words, isoperimetric graph partitioning algorithm [35] searches for the segment having the minimum area and the maximum volume simultaneously. To convert the graph clustering problem into a problem of finding a isoperimetric cut, one can define the area as the difference between segment and rest of the image, whereas define the volume as within segment similarity. This problem can be defined over a weighted graph as follows.

If we consider the weighted graph $G = (V, E)$, the isoperimetric constant can be computed as [53]:

$$h_G = \inf_{S} \frac{|\partial S|}{Vol_S} \tag{3.10}$$

$$|\partial S| = \sum_{e_{ij} \in \partial S} w(e_{ij}) \tag{3.11}$$

$$Vol_S = \sum_{i, \forall v_i \in S} d_i \tag{3.12}$$

where $w(e_{ij})$ is the weight of the edge ij and d_i is the degree of the node i computed as $d_i = \sum_{e_{ij}, \forall e_{ij} \in E}$. Moreover, ∂S is the s-t cut and S is the set of nodes labelled as foreground.

In order to show the usefulness of the isoperimetric constant, we need to define our graph first. Consider the graph of superpixels or pixels of the image; we connect each node to 4-connected or 8-connected neighbours for the pixel graph case and assign color/intensity difference as edge weights. Similarly, we also connect each superpixel to its immediate neighbours and assign mean color/intensity difference as the edge weight. We also define the degree of a node as the sum of its edge weights. Hence, if a pixel/superpixel is dissimilar from its neighbours, its degree is higher. When this graph is defined, circumference of the cut is the sum of the edges it cuts. Moreover, its volume is the sum of the degrees of the nodes included in the boundary (a foreground pixels).

Representing isoperimetric cuts as segments is similar to Shi and Malik's [64] definition of average cut. However, there are still significant differences between these formulations. Isoperimetric segment can be considered as the segment having the maximum

volume (inter similarity) and the minimum area (intra similarity). However, finding the isoperimetric cuts on general graphs are proven to be NP-hard [53]. Hence, method of isoperimetric segmentation [35] can be considered as an heuristic to find isoperimetric cuts.

In order to formulate the solution, we need to represents the isoperimetric minimization functional in terms of combinatorial structures. Consider a partition vector as a binary vector over nodes such that

$$x_i = \begin{cases} 0 & \text{if } v_i \in \bar{S} \\ 1 & \text{if } v_i \in S \end{cases} \tag{3.13}$$

Furthermore, *Laplacian* or *admittance* matrix L is defined as:

$$L_{v_i v_j} = \begin{cases} d_i & \text{if } i = j \\ -w(e_{ij}) & \text{if } e_{ij} \in E \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

We can also represent the degrees of nodes in vector from as vector d. Then, isoperimetric minimization equation can be converted into the following relation:

$$h_G = \inf_S \frac{|\partial S|}{Vol_S} = \inf_x \frac{x^T L x}{x^T d}. \tag{3.15}$$

Isoperimetric segmentation [35] uses the constant volume hypothesis. If we fix the volume of the segment we search for ($x^T d = k$). Then, the minimization problem becomes equivalent to the minimizing $x^T LX$ with constraint $x^T d = k$. If we apply Lagrange multiplication, the cost function becomes

$$Q(x) = x^T L x - \Lambda(X^T d - k). \tag{3.16}$$

After differentiating with respect to x and equating to 0, the following relation is obtained:

$$2Lx = \Lambda d. \tag{3.17}$$

Scalar terms can be ignored, since the solution will be extracted from x in scale independent form. Furthermore, L is rank deficient and non-invertible since all rows and columns of L sum to 0. At this point, interaction comes into consideration: assume c nodes are selected by the user as the foreground. Then, corresponding rows and columns can be removed from the matrix. These c nodes are assumed to be foreground. Foreground is the set of nodes having $x_i = 0$. Physical analogy from circuit theory can be used to explain this selection. If you consider the electrical circuit, (3.17) represents the Kirchoff's Voltage/Current Law, and, selecting c nodes is analogous to the selection of a ground node. Without selecting a ground node, voltages can not be obtained. After removing c rows and columns corresponding to ground nodes, $n - c$ by $n - c$ matrix will be full rank and invertible. Let us call this matrix L_0, the corresponding label vector x_0 and the degree vector d_0. Then, (3.17) becomes

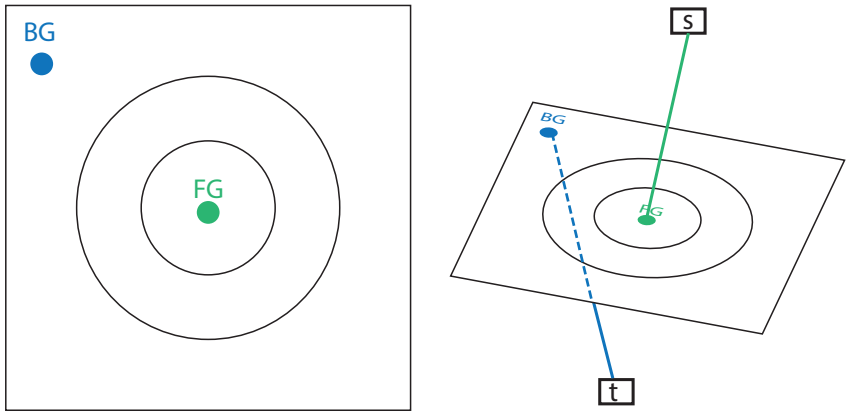
$$L_0 x_0 = d_0. \tag{3.18}$$

Moreover, this linear system can be solved efficiently. After finding x_0, x can be obtained via inserting 0's to the entries corresponding to the ground nodes. However, the resultant x vector is not binary since we did not put any constraint on x. Indeed, it is not possible to solve for the binary vector in polynomial time. Hence, the best

segmentation can be computed after trying every possible threshold in linear time. Some indicator results of the algorithm is shown in Figure 3.11.

It should be noted that originally iso-perimetric image segmentation method is proposed as an automatic image segmentation method. The ground node is selected as the node having the highest degree and partitioning is applied until the segments converge.

Isoperimetric graph cut is fundamentally different than Gibbs energy formulation. However, it also has some similarities. We want to explicitly state the relation between isoperimetric segmentation and graph-cut procedure. Consider the simplified case of graph-cut where only one node is selected as a foreground and one node is selected as a background. These two nodes are connected to the source and the terminal with infinite/zero weights as shown in Figure 3.4. Binary weights are assigned as color differences. Then, the problems become similar to each other. A cost function minimized via isoperimetric segmentation is $\frac{x^T L X}{x^T d}$. However, the cost function minimized by graph-cut is $x^T L x$.



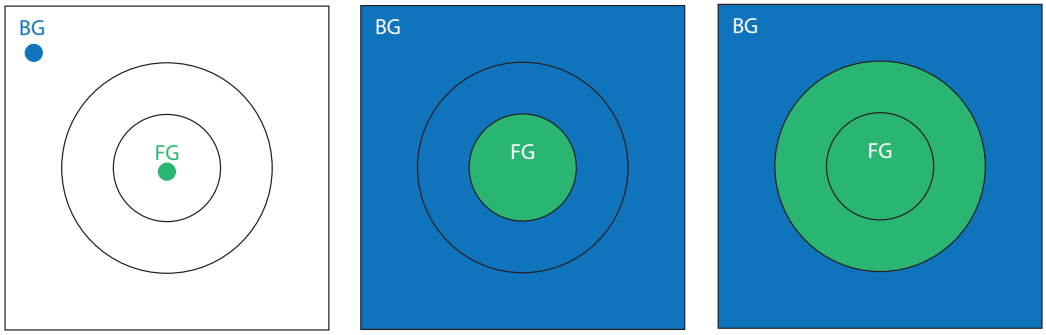
(a) Input image with interacted nodes

(b) Simplified graph for min-cut/max-flow

Figure 3.4: Toy example image and interactions are shown in (a) to compare isoperimetric segmentation and min-cut/max-flow. In (b), a simplified version of the s-t is visualized. It should be noted that unary terms are discarded in this simplified graph. Hence, it is only constructed to compare the algorithms.

Within this simplified comparison case, graph-cut based approach finds the minimum cut; hence, it finds the smallest object-like region. On the other hand, the isoperimetric segmentation finds the largest object like region. This difference is visualized in a toy example. Consider the nested circles in Figure 3.5, foreground and background nodes are selected as shown in Figure 3.5. Graph-cut finds the inner circle, since its objective is to find the minimum cut. On the other hand, isoperimetric segmentation finds the outer circle since it has the highest volume with the smallest circumference. A practical consequence is as follows: for a quite small amount of interaction, a low amount of noise and a multi-color object profile graph-cut will only find the smallest object. On the other hand, the isoperimetric segmentation will find the largest segment which has a higher possibility of being the object of interest. However, for many

practical situations, unary energy obtained via interaction and global models is much more representative than the binary energy. Since the isoperimetric segmentation does not utilize unary energy properly, in practical scenarios the segmentation accuracy of isoperimetric segmentation procedure is limited.



(a) Input image with interacted nodes.

(b) Output of simplified min-cut/max-flow.

(c) Output of isoperimteric segmentation.

Figure 3.5: Input image to isoperimteric segmentation and simplified min-cut/max-flow methods are visualized in (a). In (b) the result of the simplified min-cut/max-flow is shown. Moreover, in (c) result of the isoperimetric segmentation is shown. Isoperimetric segmentation selects the object-like region with the highest area to circumference ratio. On the otherhand, min-cut/max-flow prefers the object like region having the minimum circumference.

3.2 Proposed Method

All of the algorithms explained in Section 3.1 are mainly developed for mouse/keyboard based interfaces. Since algorithms are interactive, application scenario is an important aspect to consider. None of these algorithms are natively applicable to mobile touch-screen devices. Apart from the interaction method, the computational power of mobile devices is also another serious limitation for such methods. The main motivation of the proposed method is to solve the drawbacks of the existing algorithms that make them inapplicable to mobile scenarios.

Three main drawbacks of the existing interactive image segmentation algorithms are discussed in Section 1.1. Mainly, user centred interaction, interaction error robustness and computational efficiency are the three fundamental issues to be handled. The proposed method deals with all of these problems. A novel interaction method based on coloring gesture is explained in Section 3.2.1. Coloring is a dynamic process requiring only foreground scribbles. Moreover, it is specifically tailored for mobile touch screen applications. Apart from this improvement, a heuristic procedure to improve the error robustness is also proposed in Section 3.2.2. The most common interaction error that occurs in mobile scenarios, that is called "overfills", is handled via the proposed method. Finally, in Section 3.2.3, a novel spatially and temporally dynamic graph-cut method is proposed to improve the computational efficiency further.

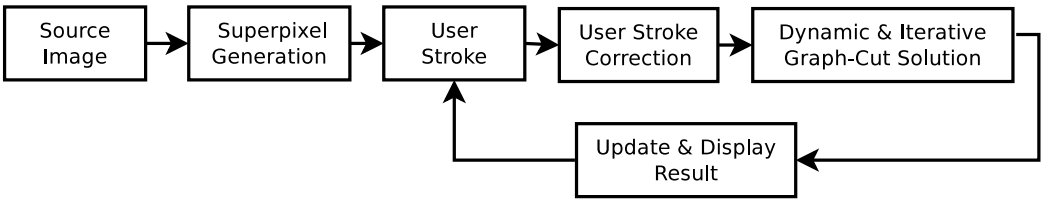


Figure 3.6: Block diagram of the overall proposed algorithm.

The proposed interactive and dynamic algorithm is summarized in Figure 3.6 as a block diagram. The coloring method is explained in Section 3.2.1. Details of the user stroke error correction sub-block is explained in Section 3.2.2. Moreover, the dynamic and iterative graph-cut sub-block is discussed in Section 3.2.3. A block diagram is used to visualize the dynamic nature of the proposed method and interaction procedure. It is observed that, to keep the user in the process throughout the entire procedure is promising idea to increase interaction quality. Indeed, superior interaction quality of the proposed method is shown via subjective user evaluation results in Section 3.3.

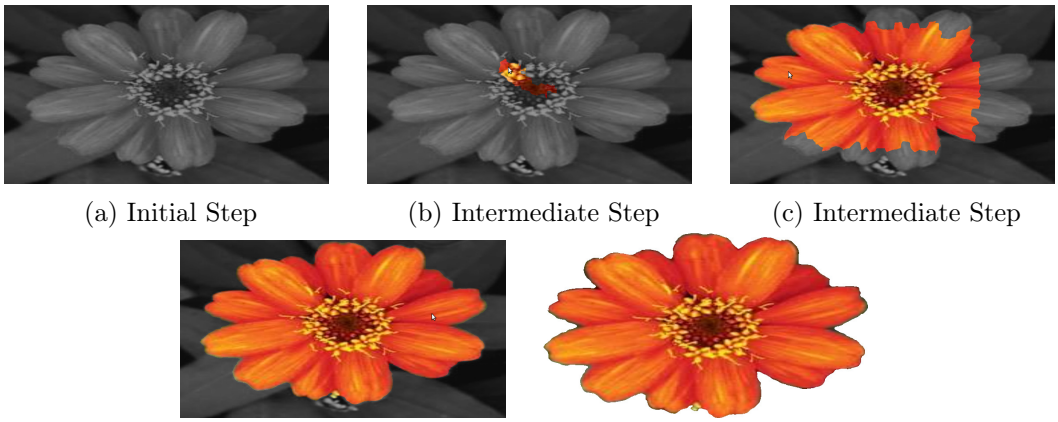
It should be noted that formulation of the energy function is based on Grabcut algorithm [60]. The same MRF energy formulation and iterative GMM estimation procedure are also used in the proposed method.

3.2.1 Segmentation via Coloring

Existing image segmentation algorithms in the literature are generally batch processes. User interacts with an input image, then runs the segmentation. If the segmentation fails, the user need to update the result by an extra interaction or need to start from scratch. Furthermore, common interaction types are scribbles on foreground and/or background, bounding box of the object of interest and approximate boundary of the object of interest. Although such approaches are reasonable for professional editing applications, mobile applications should be more interactive for a better user/consumer experience. Hence, we revisited the problem of interaction for the interactive image segmentation problem.

The proposed interaction method is a dynamic process. In the proposed interaction scenario, when the user selects a color image, a gray scale version of the image is initially displayed to the user. Then, the user starts to colourize the object of interest by the finger strokes on the screen. In order to differentiate a gray region, we also decrease the contrast of the image by adding a constant white layer via alpha matting. Hence, the user might even colorize a gray region and see the result.

By each stroke, a global segmentation is performed, and the result on the display is updated in real-time. It should also be noted that only the foreground scribbles are used. To discard background scribbles actually increase the interaction quality, significantly. Since the only interaction tool is finger strokes on mobile touch screen devices, there should be a button to switch between foreground and background scribbles, if both foreground and background scribbles are needed to be used. Coloring interaction is more appropriate for a better user experience, since it does not require processing pressing a button or a key. Moreover, coloring is a well known action; hence, the



(a) Initial Step (b) Intermediate Step (c) Intermediate Step

(d) End of Interaction (e) Final Result

Figure 3.7: Sample illusturation of user interaction. Initially, gray-scaled version of the image is shown to the user as in (a). User starts to colorize the object of interest -a flower in this case-. With each stroke, the result is updated and displayed to the user as shown in (b), (c), (d). When the user is satisfied with the result, the segmentation result is shown to the user as visualized in (e).

proposed segmentation process is as intuitive as the process of coloring a color book.

In order to structure the proposed interaction in a more intuitive form, we restrict the effect of scribbles in a local window around the interacted region. Although the segmentation is computed for the entire image, the result of the local region around the interaction is presented to the user. Although this increases the segmentation time, it also increases the interaction quality. By keeping the effect of the interactions local, we simply aim to provide the gesture of coloring a color book. Furthermore, size of the local window is also enlarged to decrease the segmentation time throughout the process.

The proposed interaction method is also visualized in Figure 3.7. Initially, the user starts with an arbitrary region around the center of the flower. Then, the user continues to interact. After each action, the segmentation result is computed and displayed to the user. The user might stop, when he/she is satisfied by the segmentation result. Final segmentation result is also presented in Figure 3.7.

As explained in Section 1.1, due to the small screen sizes of the mobile devices, the algorithms should tolerate user interaction errors. From the error correction point of view, in the proposed method, a user always has a chance to correct the foreground segments classified as the background. Indeed, segmentation process becomes more engaging by showing the result to the user dynamically. Moreover, the process implicitly guides the user to correct segmentation errors. Error correction process is a part of the interactive segmentation. On the other hand, the user might not have a chance to correct background segments classified as foreground without restarting the algorithm from scratch. Therefore, it is better to prevent these types of errors before they occur. The procedure to handle the interaction errors is explained in Section 3.2.2.

The proposed interaction method is also experimented by a subjective user experience test. We compare the crucial factors of user interaction for the mobile segmentation

problem with the state-of-the-art interaction methods in the literature. Superior performance of the coloring and the details of the experimentation procedure is discussed in Section 3.3.

3.2.2 Error Correction

Regardless of the interaction quality or the computational complexity, the segmentation accuracy is the most crucial factor for all interactive segmentation algorithms. Most of the algorithms in the literature use global models and globally optimum optimization methods. Hence, given accurate and adequate interaction, it is possible to reach near optimal segmentation accuracy. Lack of interaction is solved by the help of proposed interaction method. User can provide foreground scribbles until he/she is satisfied with the result. Hence, we can assume there is always enough amount of interaction. However, erroneous interaction is still a problem. Therefore, interaction quality is the crucial factor affecting the segmentation accuracy.

Due to the small screens of the mobile devices, the users generally make finger stroke errors during interaction and these errors typically occur around the boundary of the object of interest. Overfills are the most common example of this type of errors. User generally left the boundary of the object of interest accidentally and either left the finger or went back to the object of interest. In order to eliminate such interaction errors, we propose a correction method which is summarized in Algorithm 6. We simply keep track of the low level features of the region where user interacted, and try to correct, if interaction is not consistent with the features of the region.

We assume that user starts interaction within the object. Then, the algorithm accumulates the color statistics of the current region in a single RGB Gaussian model. When the user moves from current superpixel to a new one, the algorithm checks this new superpixel. If the new superpixel fits to the learned model, the algorithm accepts this new superpixel. If not, the algorithm stores the superpixel which the user left the object. Then, the new superpixels are stored in a temporary queue and not inserted to the algorithm. In the mean time, color model of these new superpixels are stored in another temporary single RGB Gaussian model.

When yet another superpixel is examined, if this new superpixel fits to the previously learned color model, superpixels accumulated in the temporary queue are discarded, and the correct path between the superpixel user left and the superpixel user returned back to the object is calculated and inserted in to the dynamic graph-cut. This situation corresponds to the case that user left the object accidentally and goes back to the object of interest. If the next interaction does not fit to the previous model and fits to the current temporary model, the algorithm continues to add this next superpixel to a temprorary queue and update current temporary Gaussian model. If the next superpixel fits neither to the previous nor to the current model, it means that the object has a multi color profile and interaction in the temporary queue may or may not be correct. Hence, algorithm tries to find the correct path between the starting of temporary queue and end of the temporary queue. As a fit measure, Euclidean distance between the color GMM means are used; i.e. $dist(\text{NewOS,ColorModel}) = |z_{NewOS} - \mu_{ColorModel}|$ and scalar multiple of standard deviation ($k\sigma_{ColorModel}$) is used as a threshold.

In order to find the correct path between any two superpixels which user left the object and returned to object, a minimum cost path finding problem is defined and solved. The correct path is assumed to be the geodesic path between these two points. Correct path should have minimum color variation and minimum Euclidean distance. Hence, one can define the cost of the path as weighted some of color and spatial difference as,

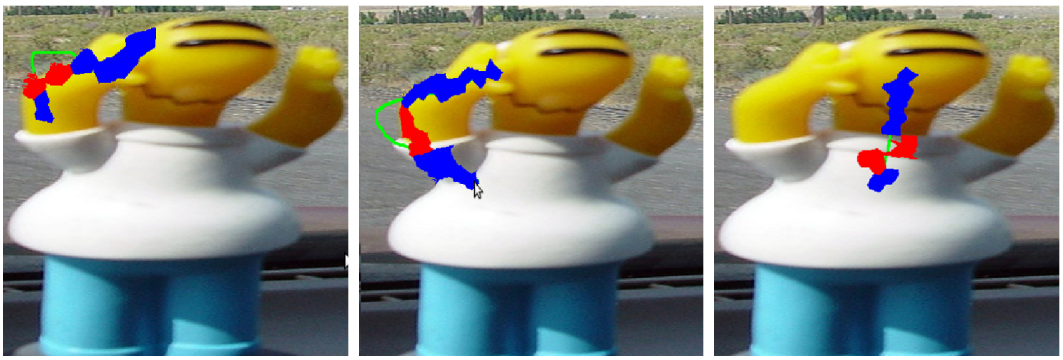
$$Cost(path) = \sum_{u,v \in path} |\overline{x_u} - \overline{x_v}| + \lambda |\overline{I_u} - \overline{I_v}|, \tag{3.19}$$

where, u and v are the nodes incident to the same edge in a path, $\overline{x_i}$ is the mean position vector of superpixel i and $\overline{I_i}$ is the mean RGB vector of superpixel i. This problem can actually be converted to a minimum path finding problem over a graph. Each superpixel is connected to its neighbours, with edge weights, as $|\overline{x_u} - \overline{x_v}| + \lambda |\overline{I_u} - \overline{I_v}|$. Then, problem is equivalent to find the minimum path connecting two specified node over this graph. Since all edge weights are Euclidean distances, they are positive and minimum path can be obtained via Dijkstra's algorithm [22] over the superpixel graph efficiently in polynomial time. λ is the parameter which sets the relative importance of spatial and color distances. Furthermore, throughout the experiments, we have fixed the value of parameter $\lambda = 0.5$. If any prior information exists about the application domain, λ value can be found empirically or via Bayesian learning [11].

Algorithm 6 User input correction algorithm

Initialization: $PossibleError \leftarrow 0$, **clear** TempQueue
InsertNewOversegment(NewOS):
if not *PossibleError* **then**
  if $dist$(NewOS,CurrentMdl) $\leq k\sigma_{current}$ **then**
    insert NewOS **to** Dynamic-Graph-Cut
    update CurrentMdl **with** NewOS
  else
    $PossibleError \leftarrow 1$, LeftOS$\leftarrow$NewOS, PrevMdl $\leftarrow$ CurrentMdl
  end if
else
  if $dist$(NewOS,PrevMdl) $\leq k\sigma_{prev}$ **then**
    CurrentMdl $\leftarrow$ PrevMdl, $PossibleError \leftarrow 0$, FindPath(LeftOS,NewOS)
    insert found path **to** Dynamic-Graph-Cut
  else if $dist$(NewOS,CurrentMdl) $\leq k\sigma_{current}$ **then**
    insert NewOS **to** TempQueue
    update CurrentMdl **with** NewOS
  else
    $PossibleError \leftarrow 0$, FindPath(LeftOS,NewOS)
    insert founded path **to** Dynamic-Graph-Cut
  end if
end if

In order to visualize the performance of the proposed error correction algorithm, we summarized the three main error correction scenarios in Figure 3.8.a, 3.8.b and 3.8.c on a sample image. In Figure 3.8.a, 3.8.b and 3.8.c, blue superpixels are the ones accepted as the correct user interaction, whereas the green line is the discarded user interaction (considered as an interaction error). Moreover, red superpixels are obtained from the minimum cost path solution. Therefore, only the blue and the red regions are used for



(a) Error - 1 color (b) Error - 2 colors (c) False Positive

Figure 3.8: Visualization of the proposed error correction algorithm for three main interaction error scenarios. In all figures the green line is the discarded user interaction, blue is the accepted user interaction and red is the estimated correct path. In (a), (b) user leaves the object and returns back for the single color and the multi color case. In (c) false positive of the error correction algorithm is shown. Although the correct user interaction is discarded, the resultant path is still correct. In summary, the proposed method successfully handles all three cases for this example.

the dynamic and iterated graph-cuts algorithm. It should be noted that these lines are not presented to the user as part of the algorithm. These markers are only drawn to explain the algorithm. In the final version, only the current segmentation result is shown to the user.

In Figure 3.8.a, the user first accidentally left the object boundary and then returned back to object of interest. The proposed algorithm discards the erroneous interaction and finds the correct path. This case is considered to be of primary interest, since this type of interaction errors are the main set of errors which we aim to solve. It should also be noted that even when there are only a few correct interactions in the starting phase, the proposed method still effectively learn the color statistics of the region and detects the inconsistency. Indeed, this behaviour is result of over-segmenation. Even a single over segment includes enough number of pixel to accurately estimate the color distribution.

In Figure 3.8.b, the user left the object accidentally from the yellow coloured region, then returned to a white coloured region. The proposed algorithm also handles this case successfully. Since all the paths connecting these two nodes need to cross the white/yellow boundary, they all need to pay the cost of this boundary. Hence, the proposed algorithm selects the one which is most consistent with the white/yellow regions and with the minimum length.

In Figure 3.8.c, user left the yellow region and then continued along the white region (multi-color case). When the user enters the region with shadow, the algorithm detects the color change and solve the path finding problem. Indeed, the resulting path is also correct. In other words, in case of false alarm, the resulting minimum path is still expected to be contained in the object; therefore, false alarm has no side effects for the algorithm. Indeed, if λ is not too small, the resultant path will not diverge due to the cost of Euclidean distance. Hence, if this false positive is not close to the boundary of

the object of interest, the resultant path will stay inside the object.

3.2.3 Dynamic Graph-Cut

Mobile devices generally have significantly less computational power when compared against standard computers. Hence, we need to decrease the computation complexity of the method to make it applicable to mobile scenarios. When computation of different blocks of the algorithm is considered, there exist room for improvement. For oversegmentation, SLIC [1] is utilized and it is a simple implementation of constrained K-means algorithm. Global model used in the system is a GMM; GMM is generally slow to converge, however, after the initialization stage, the algorithm is quite fast. One can claim that parameters of the GMM does not change significantly with each scribble. Hence, we can update the GMM throughout the interaction efficiently in less iteration. One can safely state that global feature modelling and over-segmentation steps are efficient enough. However, for min-cut/max-flow, there is significant room for improvement. At each stroke, min-cut/max-flow is recomputed from scratch. However, one can use previous solution to increase the efficiency. Furthermore, a dynamic graph-cut is developed entirely based on this idea [41]. In this section, we first explain this method. Then, we further improve its efficiency by the proposed novel spatially dynamic graph-cut.

In addition to dynamic improvements, there exist faster graph-cut methods in the literature, too. Hierarchical graph-cut [51] segmentation starts from the pyramidal representation of the graph and uses the previous coarse solution to warm start the fine solution. Band-level graph cut [52] also starts with an approximate boundary and only solves the graph-cut around this boundary. All of these solutions are approximate and local solutions and they might not be equal to the globally optimum solution. Hence, these approximate solutions are discarded.

The dynamic graph-cut algorithm is explained first in detail in Section 3.2.3.1. Then, the proposed novel improvement is explained in Section 3.2.3.2.

3.2.3.1 Temporally Dynamic Graph-Cut [41]

Consider the problem of video segmentation. If we consider the pixel grid as a graph, the structure of the graph does not change throughout the video. Number of nodes is the same and the edges are at exactly same locations (incident to same nodes). We can also state that unary and binary energies are not changing significantly, since most of the video is stationary. Then, idea of dynamic graph-cut is to reuse the flows of previous frame in the current frame. Although our problem is not video segmentation, each iteration of the algorithm can be considered as a different frame. Hence, formulation of temporally dynamic graph-cut is applicable to our problem. It should be noted that the name of the algorithm in [41] is dynamic graph-cut. Since we also proposed another dynamic graph-cut, we call our novel algorithm as the temporally dynamic graph-cut.

We start with formally defining the algorithm [41]. There exist two minimization problems with the same graph structure. Hence, one can consider both problems to

be defined on the same graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. Consider the weights of graph-1 and graph-2 as w_{ij}^1 and w_{ij}^2. The main idea is to reuse all the flows of the solution of graph-1 in the solution of graph-2. We can use the definition of the residual weights in order to accomplish this task. The residual weights of graph-1 can be written as,

$$r_{ij}^1 = w_{ij}^1 - f_{ij}^1, \tag{3.20}$$

where f_{ij} represents the flow at edge e_{ij} and w_{ij} represents the weight of the edge e_{ij}. If the same flows are used in graph-2, the resultant residual graph would be equal to

$$r_{ij}^2 = w_{ij}^2 - f_{ij}^1 \tag{3.21}$$

$$r_{ij}^2 = w_{ij}^2 - (w_{ij}^1 - r_{ij}^1) \tag{3.22}$$

$$r_{ij}^2 = r_{ij}^1 + (w_{ij}^2 - w_{ij}^1). \tag{3.23}$$

Hence, instead of solving the graph having weights w_{ij}^2, we can find the updated residual graph via formula $r_{ij}^2 = r_{ij}^1 + (w_{ij}^2 - w_{ij}^1)$ and solve the updated residual graph. The main advantage of this approach can be stated as follows: Since two energy functions will be similar to each other, most of the weights will be either 0 or close to 0. Hence, updated graph can be solved with much small number of flows. Therefore, the convergence is expected to be much faster.

The main problem, which can arise during this formulation, is the edges having negative weights. Min-cut/max-flow procedure is optimal, if all edge weighs are non-negative. Moreover; in its standard(non-dynamic) representation, graph-2 will not have any negative edge weights. Moreover, applying an arbitrary flow does not change the optimization problem. Hence, the resulting updated graph-2 can be converted to a graph having no negative edge weights via set of additional flows.

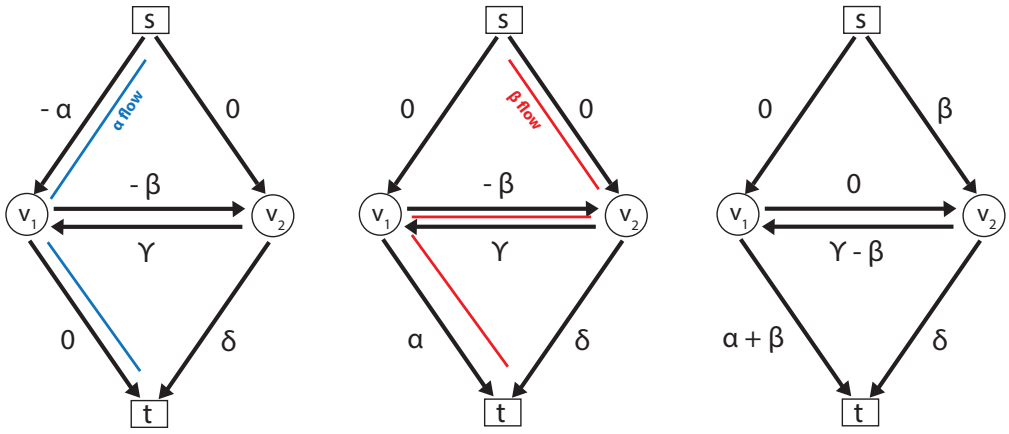
To make the algorithm optimal, one need to propose an algorithm to find required additional flows. These flows should not violate the conservation of the flow and should replace the negative edge weight with non-negative ones. If weight of any edge within terminal and non-terminal node is negative, one can push flow with magnitude equal to the magnitude of negative edge weight via the source node, problematic node and the terminal node. Assume w_{si} is negative $(-\alpha)$; then, one can push flow of α through s, i and t. Let w_{ij} be negative $(-\beta)$; then, one can push flow of β through s, i, j and t to make edge weights non-negative. These flows are visualized and explained in Figure 3.9. This procedure is named as the re-parametrization of the graph [41].

Re-parametrization for the terminal weights are quite trivial as shown in Figure 3.9; however, in order to validate the correctness of the re-parametrization of the non-terminal weights (β flows), we need an extra constraint. Since non-terminal edges are two sided, an increase in one direction should correspond to a decrease in the other direction. Hence, we need to show that re-parametrization results in non-negative edge weights for both directions. This constraint comes from the principle of sub-modularity. The sub-modularity principle states that:

$$E(1,0) + E(0,1) \geq E(1,1) + E(0,0) \tag{3.24}$$

$$E^{12}(1,0) + E^{12}(0,1) \geq E^{12}(1,1) + E^{12}(0,0) \tag{3.25}$$

$$E^{12}(1,0) + E^{12}(0,1) \geq 0. \tag{3.26}$$



(a) Initial residual graph with α re-parametrization

(b) β re-parametrization

(c) Resultant graph having non-negative edge weights

Figure 3.9: Re-parametrization of the updated residual graph having negative edge weights. In (a), an updated graph having negative edge weights at both terminal and non-terminal edges is represented with a required terminal weight re-parametrization. In (b), graph after terminal re-parametrization is visualized with required non-terminal weight re-parametrization. In (c), re-parametrized graph having non-negative edge weights is shown. Non-negativity of $\gamma - \beta$ is the result of sub-modularity property.

Constraint in (3.26) proves that the re-parametrization always results in non-negative edge weights. For example, consider the graph in Figure 3.9.a. By sub-modularity,

$$E^{12}(1,0) + E^{12}(0,1) \geq 0 \tag{3.27}$$

$$-\beta + \gamma \geq 0. \tag{3.28}$$

Hence, in the final re-parametrized graph, $\gamma - \beta$ is always non-negative. Correctness of re-parametrization concludes the temporally dynamic graph-cut formulation.

3.2.3.2 Proposed Spatially and Temporally Dynamic Graph-Cut

Temporally dynamic solution [41] improves the computation efficiency significantly but there is still some room for additional improvement. In the proposed interaction method, user colourize the object of interest locally; therefore, the required solution should also be a local one. However, min-cut/max-flow solution is determined for the entire graph; therefore, there must be some redundant processing. A straightforward remedy to this problem is to solve the sub-graph including the user interaction. However, performed experiments showed that it is not possible to find a generic size for this sub-graph. Therefore, an adaptive method to find an appropriate size of this subgraph is proposed.

First, we need to relate the spatially and temporally dynamic graph-cut to the temporally dynamic graph-cut. Consider the sub-graph at time t, and enlarged sub-graph at time $t+1$; One should show that flows in time t can be re-used for the graph in time $t+1$. If the edges are divided into two groups, as edges exist in both graphs and

the edges exist only in graph in time $t+1$, for the first group of edges we can reuse the flows as

$$r_{ij}^2 = r_{ij}^1 + (w_{ij}^2 - w_{ij}^1). \tag{3.29}$$

Since there exist no flow for the second group, we can find the residual weight as

$$r_{ij}^2 = w_{ij}^2. \tag{3.30}$$

It should be noted that this equation is analogues to the temporally dynamic graph-cut, if we assume that the edges in the second group also exist in the first graph with edge weights equal to zero. Since their flow value equals to zero, their residual weights should also be zero. If we put these values into the temporally dynamic graph-cut equation, we obtain

$$r_{ij}^2 = r_{ij}^1 + (w_{ij}^2 - w_{ij}^1) \tag{3.31}$$

$$r_{ij}^2 = 0 + (w_{ij}^2 - 0) \tag{3.32}$$

$$r_{ij}^2 = w_{ij}^2. \tag{3.33}$$

Hence, starting with a small graph and enlarging it is equivalent to the case of assigning zero weights to the edges not included in the sub-graph at time t. Since the number of edges and nodes in the subgraph will be much smaller than the number of edges and nodes in the original graph, the solution to the resultant optimization problem should be much faster. The only remaining problem is to adaptively find the size of this sub-graph. If we can find this size, one can solve the graph-cut for this sub-graph and in the next iteration, one can enlarge this graph and update the residual sub-graph as in the case of temporally dynamic graph-cut.

In order to find the size of this sub-graph, one should define the desired properties of this sub-graph. After a set of scribbles drawn for foreground, the sub-graph definitely needs to include the interacted region. Hence, if we assume a rectangular sub-graph; it should include the bounding box of the interaction. Segmentation result of this sub-graph should be robust. In other words, if we enlarge the sub-graph further in the current iteration, the resultant segmentation for the initial sub-graph should not change. In other words, the rest of the graph should not change the solution of the sub-graph. Finally, the resultant sub-graph should be the minimum of sub-graphs satisfying other requirements to have the highest computational efficiency.

The proposed algorithm starts with the smallest possible sub-graph and checks for the robustness requirement. The smallest possible sub-graph is the bounding box of the interaction. Moreover, we need to formalize the requirement of robustness in terms of graph properties. Starting with the smallest possible rectangle and enlarging it until robustness requirement is satisfied, we always find the smallest sub-graph satisfying the robustness requirement. At this point, we need to formalize only the robustness requirement.

First, consider a robustness requirement for a set of connected nodes. Assume there exist a set of connected nodes R, such that all nodes of set R is either foreground or background. Then, assume that a sub-graph including nodes in R is chosen and min-cut/max-flow operation is performed and the residual graph is obtained. Then, the solution for a subset of connected nodes R having the same segmentation result can not be changed simultaneously by the external flow, if the conditions in (3.34a)

and (3.34b) are satisfied. Simultaneous change corresponds to the flipping the label of all nodes in region R. Robustness condition can be formulated by using residual weights as:

If R is foreground (connected to source)

$$\sum_{i \in R} w_{iS} - w_{iT} > \sum_{iR, j \notin R} w_{ij} \tag{3.34a}$$

If R is background (connected to sink)

$$\sum_{i \in R} w_{iT} - w_{iS} > \sum_{j \notin R, i \in R} w_{ji} \tag{3.34b}$$

where w_{iS} and w_{iT} denote the terminal weight of node i with source and sink, respectively.

This condition holds since the cost of changing the solution (cutting terminal edges) is larger than the cost of cutting all the non-terminal edges. It should be noted that solution to some of the nodes in R might still change; however, the result for the entire R can not change. In other words, only some part of the nodes in region R might flip their labels.

On the other hand, this condition can also be defined in terms of edge weights between the sub-graph and the rest of the global graph. It should be noted that there is no available path within the sub-graph since this would conflict with the augmenting paths algorithm as explained in Section 2.3. Therefore, all the paths which change the solution should go through edges between the sub-graph and the rest of the global graph. Moreover, if the sum of the possible flows through these paths is less than the terminal weights of the nodes, the resultant labelling can not be changed via enlargement of the sub-graph; because, cost of the changing the solution (cutting only the terminal edges) is larger than cutting all edges between sub-graph and the rest of the graph.

The condition for the sub-graph case can be formulated as: if N is the set of nodes neighbour to the nodes in the sub-graph, and $\exists Path(i, j)$ indicates the existence of a path between i and j; then the following condition should hold for robustness:

If R is foreground (connected to source)

$$\sum_{i \in R} w_{iS} - w_{iT} > \sum_{\substack{i \in R, j \in N \\ \exists Path(i,j), e \in E \cap Path(i,j)}} min(w_e), \tag{3.35a}$$

If R is background (connected to sink)

$$\sum_{i \in R} w_{iT} - w_{iS} > \sum_{\substack{i \in R, j \in N \\ \exists Path(j,i), e \in E \cap Path(j,i)}} min(w_e). \tag{3.35b}$$

Minimum weight in the path is used as a flow value, since it corresponds to the maximum amount of the flow which can go through the path. It should be noted that when the conditions in (3.34a) and (3.34b) are satisfied, labels of all the nodes in region R can not change simultaneously.

If R is taken as a single super-pixel and all the superpixels in the selected sub-graph satisfy the condition in (3.34a) and (3.34b), the solution of any node in the graph can not change when sub-graph is enlarged. However, it is not efficient to check this condition. Indeed, this condition is too strict to be satisfied.

One can relax this condition by using clustering. Instead of superpixels, we can use the cluster of superpixels. If one can find some reliable clustering, and (3.34a) and (3.34b) are satisfied for each of these clusters, one can safely claim that sub-graph is robust. Indeed, this clustering is already supplied by the GMM algorithm. Moreover, in our experimental analysis, all sub-graphs satisfying (3.34a) and (3.34b) give the exact same segmentation results compared to the global solution. Hence, we can safely use the GMM clustering and conditions given by (3.34a) and (3.34b).

The resulting algorithm first starts with a bounding box of the user interaction, and then enlarges this sub-graph until the condition in (3.34a) and (3.34b) are satisfied for all GMM clusters.

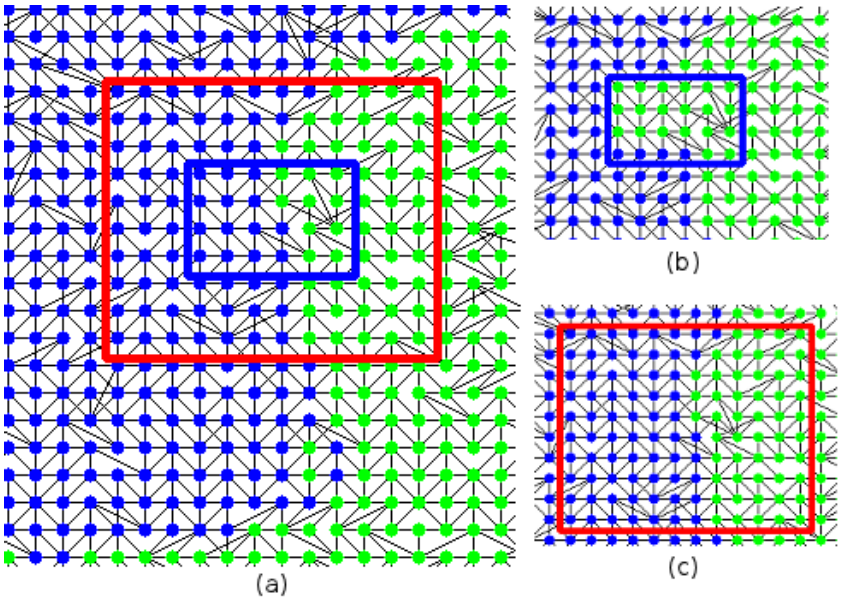


Figure 3.10: Visualization of the automatic sub-graph finding procedure. In (a) ground truth foreground and background information given as green and blue nodes, respectively. Blue rectangle is the initial bounding box of the interaction. Moreover, red rectangle is the enlarged subgraph obtained via the proposed algorithm. The result of the blue rectangle is shown in (b) and result of red rectangle shown in (c) via color coding of green for resultant foreground and blue for resultant background. Although the initial bounding box results in an erroneous segmentation, enlarged graph results in a more accurate segmentation.

Consider the example in Figure 3.10.a, in which the green nodes represent the ground-truth for the foreground, whereas the blue nodes represent the ground-truth for the

background nodes. In Figure 3.10.a, the blue rectangle is the bounding box of the interacted superpixels, and this rectangle does not satisfy the condition. Enlarged version, which satisfies the condition, is computed via the proposed algorithm and shown with the red rectangle. The corresponding solutions are given in Figure 3.10.b and 3.10.c, respectively. In Figure 3.10.b and 3.10.c, the green and blue nodes represent the computed foreground and background nodes, respectively. As presented in this figure, although the initial solution is erroneous, the enlarged rectangle leads to the correct solution.

3.3 Experimental Results

Many aspects of the proposed algorithm is tested by using an extensive dataset with various color and texture profiles. Tests are conducted to compare the segmentation accuracy of the proposed method, as well as to experimentally validate the improvements achieved via the novel coloring interaction method, the stated error tolerance algorithm and the proposed spatially and temporally dynamic graph-cut algorithm.

3.3.1 Analysis of Segmentation Accuracy

Segmentation accuracy of the resultant method is compared against the intelligent scissors [54], grabcut [60] and isoperimetric segmentation [35] algorithm. The main reason for the selection of these algorithms is their interaction methods. Interactions, such as right-click and key press, are not desirable in touch-screen mobile application interfaces. Moreover, these algorithms are the only state-of-the-art options which do not require such types of interactions. Therefore, these methods are assumed to be the only natively applicable ones to the mobile touch-screen based scenarios. In [60], a bounding box drawn around the object of interest is used as the interaction. In [54], a roughly drawn boundary of the object is used for the interaction. Moreover, in [35], scribbles on foreground are used as the interaction.

Usage of isoperimteric image segmentation as an interactive tool did not yield promising results. As explained in Section 3.1.4, isoperimteric segmentation is not originally a binary segmentation algorithm. It performs multi-object segmentation and some of them actually corresponds to the object of interest. However, binary segmentation results are generally erroneous. This problem is visualized in Figure 3.11. In Figure 3.11, some interactions, binary segmentation results and automatic multi object segmentation results are shown. As shown in Figure 3.11, although the isoperimetric segmentation algorithm is an accurate automatic segmentation tool, its interactive extension does not produce accurate segmentation results. Indeed, selection of a ground node is not a strong prior for an interactive image segmentation scenario.

Algorithms are visually compared by some images having different level of textures and color profile. Interactions and their corresponding results are presented in Figure 3.12, 3.13, 3.14 and 3.15.

As it can be observed from the results in Figure 3.12, 3.13, 3.14 and 3.15, intelligent scissors algorithm [54] requires many seed points for a robust operation. Even with

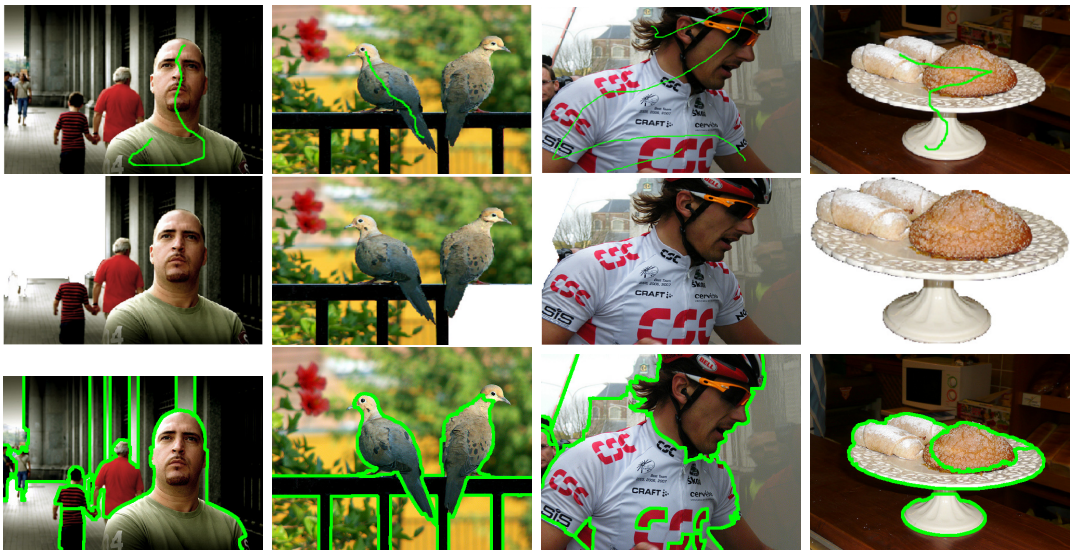


Figure 3.11: Comparison of automatic and interactive segmentation scenario of isoperimetric image segmentation algorithm [35]. In the first row, the interaction is visualized. In the second row, the interaction is fed as the ground nodes to the linear system and the binary segmentation results are shown. In the final row, no interaction is used and multi-object automatic segmentation results are shown.

(a) Interaction for grabcut [60]

(b) Interaction for intellgient scissors [54]

(c) Interaction for the proposed method

(d) Result for grabcut [60]

(e) Result for intellgient scissors [54]

(f) Result for the proposed method

Figure 3.12: Comparison of the interactive segmentation methods.

dense input seed points, the performance is quite limited, if there is local texture around the boundary. For all the test images, the resulting segmentation is neither smooth nor correct. Even for the strong boundary in Figure 3.15, the intelligent scissors algorithm [54] fails to find correct object boundary. Furthermore, the results are not smooth at all.

Grabcut [60] algorithm yields acceptable results, if the object of interest has a different

(a) Interaction for grabcut [60] (b) Interaction for intellgient scissors [54] (c) Interaction for the proposed method

(d) Result for grabcut [60] (e) Result for intellgient scissors [54] (f) Result for the proposed method

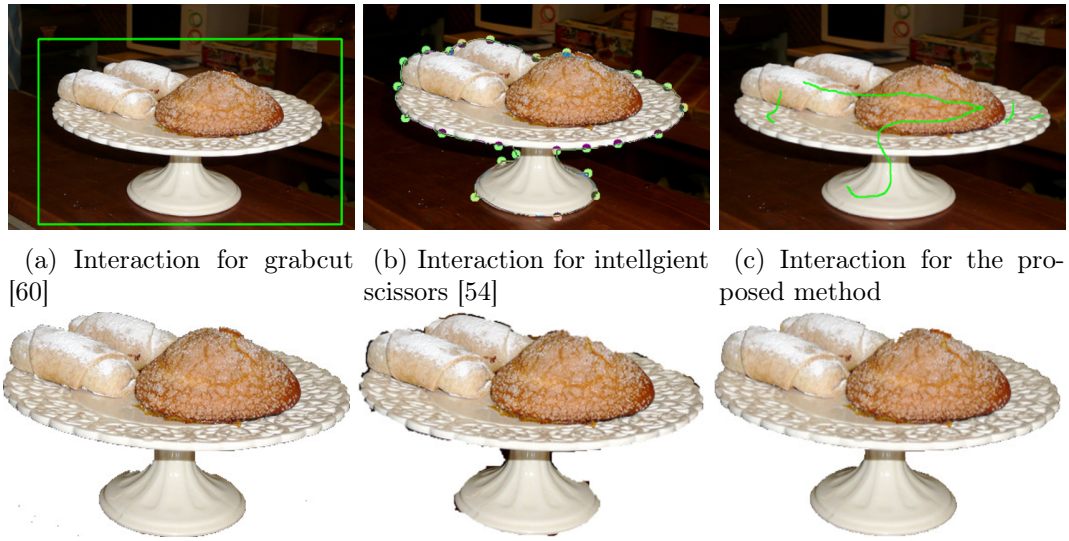
Figure 3.13: Comparison of the interactive segmentation methods.

(a) Interaction for grabcut [60] (b) Interaction for intellgient scissors [54] (c) Interaction for the proposed method

(d) Result for grabcut [60] (e) Result for intellgient scissors [54] (f) Result for the proposed method

Figure 3.14: Comparison of the interactive segmentation methods.

colour characteristics than the background as in the case of Figure 3.15. However, the performance of the Grabcut algorithm is limited in other cases. In Figure 3.12, there is a second object in the rectangle apart from the object to be segmented. The algorithm could not separate the objects, since both objects are contained within the rectangle. In Figure 3.13, the color characteristics of the head of the bird is similar to the background; therefore, the head of the bird is segmented as background. Finally, in the Figure 3.14, there is not enough background information; therefore, Grabcut fails to segment the object.



(a) Interaction for grabcut [60] (b) Interaction for intellgient scissors [54] (c) Interaction for the proposed method

(d) Result for grabcut [60] (e) Result for intellgient scissors [54] (f) Result for the proposed method

Figure 3.15: Comparison of the interactive segmentation methods.

The proposed method has superior performance for all the test examples visualized in Figure 3.12, 3.13, 3.14 and 3.15. In Figure 3.12 and 3.15, the proposed method yields much smoother and more accurate segmentation results. In Figure 3.13, the foot of the bird is not segmented correctly due to the limitations of touch screen. However, the proposed method still outperforms the others. In Figure 3.14, there is an erroneous additional small head around the shoulder of the cyclist. This artifact is caused by the *shrinking bias*. Some small objects having strong edges around the boundary may be segmented erroneously by min-cut/max-flow methods, since the sum of terminal weights (the sum of background likelihoods) might be less than the coherence penalty of the boundary. This phenomenon is called as *shrinking bias*. Indeed, solving this problem is possible by empirically adjusting the parameter which controls the effect of coherence penalty or by designing a learning based inference algorithm if there exist a training data.

3.3.2 Analysis of Interaction Quality

In order to analyse the performance of the proposed interaction method -coloring- experimentally, we designed a subjective user evaluation procedure.

We have compared 3 different interaction methods, namely Intelligent Scissors [54], Grabcut algorithm [60] and the proposed method. These methods are given to the user with anonymous names as Algorithm A, B and C. In order to evaluate the algorithms, 4 different evaluation metrics are used: *segmentation performance*, *entertainment*, *easiness* and *overall satisfaction*.

At the beginning of the experiment, each subject is shown a tutorial about the usage of each algorithm. Then, users are provided with a sample image to segment by using each algorithm. This instructive sample image is selected as the same image in all

experiments and is not used in the rest of the experiment.

After the initial tutorial and segmentation of a sample image, the user is asked to segment 4 images randomly selected from the dataset composed of 10 images with various difficulty. For each image, the algorithms are applied in a random order.

The user is asked to rate each algorithm for each of these 4 metrics. Metrics are explained to the user with the questions: "How much are you satisfied the resultant segmentation accuracy ?", "How entertaining was the overall process of segmentation ?", "How easy was to segment the image ?" and "What is your overall rating of the segmentation algorithm ?" Rating is conducted by grading at the level of 1-5. The tests were conducted with a capacitive mobile touch screen. 15 subjects, composed of undergraduate engineering students, have been participated in the tests.

For the analysis, median ratings of each metric for each algorithm, as well as interquartile ranges (IQR) and standard deviations (STD) are summarized in Table 3.1. Dependent ANOVA test is applied to find p-values and p-values are the same for each metric and equal to 0.0005 [5].

Table3.1: Interaction quality subjective evaluation results in the format of *Median (IQR, STD)*, For each metric, p-values are obtained via dependent ANOVA test and they all are equal to 0.0005.

	Performance	Easiness	Entertainment	Overall
Proposed Met.	**5 (1, 0.45)**	**4 (0, 0.86)**	**5 (1, 0.74)**	**4 (1, 0.45)**
GrabCut[60]	3 (2, 0.92)	**4 (1, 0.75)**	2 (1, 0.61)	3 (1, 0.77)
Intelligent Scissor[54]	3 (1, 0.51)	2 (1, 0.74)	3 (2, 0.89)	2 (1, 0.76)

The proposed method has the best segmentation performance result. Superior segmentation quality of the proposed method is also visible in Figure 3.12, 3.13, 3.14 and 3.15. Although the results of intelligent scissors is much close to the correct segmentation result when compared to the Grabcut, it has the have same segmentation accuracy when compared to the Grabcut subjectively. We think that this behaviour is a result of the non-smoothness of the results of intelligent scissors.

Grabcut algorithm requires only a rectangle around the object so we expect it to be the easiest one. However, the proposed method is thought to be as easy as Grabcut. We conclude that, this result is due to the intuitive coloring gesture used in the algorithm. Intelligent scissors has the worst result in terms of easiness, since the selection of landmarks and movement around the boundary is quite unattractive. Most of the subjects actually commented about the unattractiveness of intelligent scissors.

The proposed method has the best entertainment result. This result is expected, since coloring is an entertaining process. On the other hand, Grabcut has the worst entertainment result. This result is surprising, since comments of the users suggest that intelligent scissors is an unattractive process. We argue that this surprising result is due to the time spent for the interaction. When the user spends more time, he/she likes the process more.

We can easily conclude that the proposed technique has the best overall satisfaction result. With its superior performance and interaction quality, this result was expected.

3.3.3 Analysis of Computational Efficiency

Proposed spatially and temporally dynamic graph cut algorithm is compared against the conventional min-cut/max-flow solution [13], as well as the temporally dynamic graph-cut [41] solution. Proposed method is dynamic; hence, all of the interaction given by the user until the end of the current optimization stage, is fed to the system. It is not fair to compare algorithms independently, since amount of the interaction at any step might differ significantly. It is also not fair to compare total interaction time, since it is dependent on the speed of the user. Fair comparison is only possible by feeding the same input to all algorithms.

In the proposed experimentation scenario, a complete segmentation is obtained from a user. Then, interaction throughout the segmentation is divided into equal parts; introduction of each new k superpixels is considered as a new interaction. Hence, all algorithms are performed after each k new superpixels are fed into the system. This experiment is both fair and realistic. Input is real interaction obtained from a subjective evaluation and all algorithms are fed with same inputs. We select k as 3 for all the experiments.

For the segmentation of a single image, computation times of three different algorithms are calculated for each iteration (each k new superpixels). Execution times are computed on a mobile device having 600 MHz ARM-Cortex-A8 CPU and Maemo 5 GNU/Linux operating system. Then, plot of the computation time throughout the entire process is obtained as in Figure 3.16. At any time, only the computation time for the current iteration is shown.

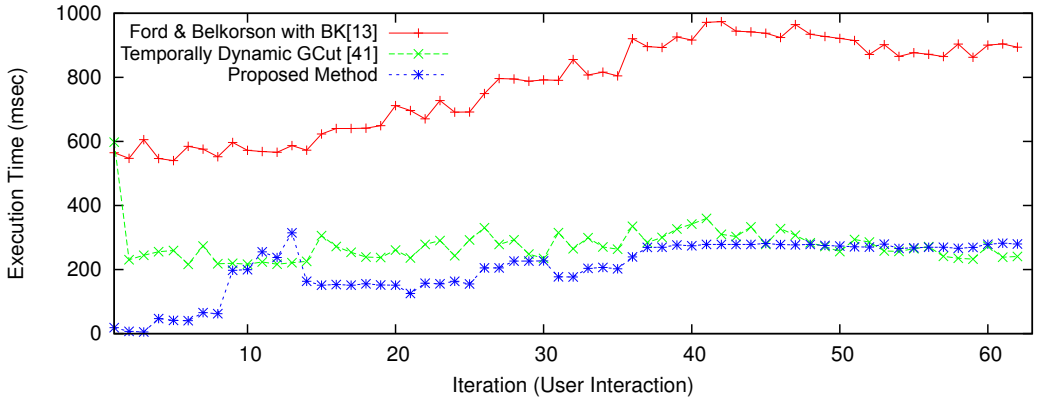


Figure 3.16: Execution times of each iteration of the algorithms for Ford& Belkorson algorithm with Boykov& Kolmogrov heuristic [13], Temporally Dynamic Graph-Cut [41] and the proposed method. Each iteration corresponds to the introduction of 3 new superpixels via user interction. Exactly the same inputs are fed into each algorithm at each iteration.

Obviously the original min-cut/max-flow algorithm [13] has the highest computation time, since it is the base algorithm for the other two. Dynamic graph-cut [41] starts with the full solution of graph-cut. After the initial iteration, dynamic graph-cut only updates the residual solution; hence, it has a lower computation time. At the initial iteration, the computation times of the dynamic graph-cut and the original min-cut/max-flow are equal. The only time difference is caused by the extra bookkeeping

Table3.2: Average computation times per iteration for Boykov& Kolmogrov [13], Kohli& Torr [41] and proposed method

Boykov&Kolmogrov [13]	Kohli&Torr [41]	Proposed Method
771 msec	278 msec	201 msec

required for the dynamic graph-cut. Moreover, the proposed algorithm initially starts with a small sub-graph and it is much more efficient than the other methods at the starting phase of the algorithm. Throughout the process, the computation time of the proposed method increases, since interacted region and the resulting sub-graph gets larger. Eventually, the sub-graph converges to the full graph and the computation time of the proposed method converges to that of the dynamic graph-cut.

Since the proposed algorithm enlarges the sub-graph until (3.35a) and (3.35b) are satisfied, there are points which enlargement and augmenting flow algorithm are performed many times. At those iterations, the proposed method spends more time compared to that of the dynamic graph-cut, as expected. This situation is observable in Figure 3.16 at the 13th iteration. Since enlargement is performed many times at these iterations, the algorithm converges much faster for the rest of the iterations. For all other iterations, superior time performance of the proposed method is visible in Figure 3.16.

As a second test, a total of 50 segmentation procedures for different images are performed and recorded. User interactions for these segmentation procedures are divided into discrete steps including interaction of $k = 3$ superpixels. Then, these iterations are fed to all the tested algorithms. Moreover, average computation time of each algorithm for each iteration is computed. The average computation time per iteration of these 3 algorithms is tabulated in Table 3.2. It is also clear from Table 3.2 that the proposed algorithm yields the best computation time.

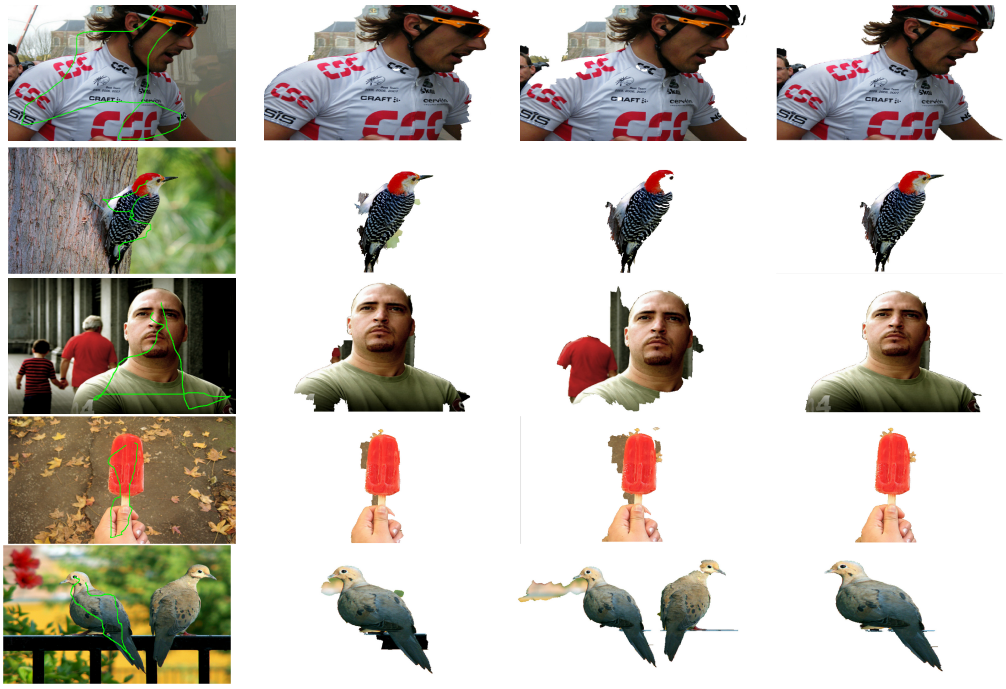
3.3.4 Analysis of Error Correction

As explained in Section 1.1, due to the small size of the mobile touch screens, there will always be some errors during user interaction. In order to prevent such interaction errors, a method is proposed in Section 3.2.2. Furthermore, the proposed error correction mechanism is experimented for such scenarios.

The proposed error correction algorithm corrects errors before the optimization stage. On the other hand, correction can also be incorporated within the graph-cut formulation. If hard constraints (terminal edges with infinite weight) are replaced with soft constraints (terminal edges with weights calculated from GMM), graph-cut framework supposed to handle these interaction errors. However, these hard labels on the foreground are semantically meaningful, and the proposed algorithm tries to utilize these interactions.

We compare three main scenarios. The first scenario trusts user interaction and handling errors before the segmentation stage; i.e. the proposed method. The second scenario does not trust user interaction at all, and assigns soft weights (likelihood values) instead of infinite weights i.e. soft label graph-cut. The final scenario trusts the user and does not do any error correction i.e. hard-label graph-cut. Non-terminal

weights are selected as the same in all methods. The results of this experiment are shown in Figure 3.17.



(q) Interaction (r) Hard-label graph cut (s) Soft-label graph cut (t) Proposed method

Figure 3.17: Comparison of the proposed error correction method against the hard-label graph-cut and the soft-label graph-cut. Columns show the interaction and corresponding results for the hard-label graph-cut, soft-label graph-cut and the proposed method, respectively. For hard-label graph-cut, no error correction is utilized in the system. Moreover, for soft-label graph-cut, infinite weights at the terminal edges of nodes user interacted are replaced by the GMM likelihood values.

Figure 3.17 suggests that the proposed algorithm has a superior error correction performance. Hard-labeled graph-cut leads to a solution with extra erroneous regions, since no error correction is utilized. In the soft-label graph-cut case, discarding spatial information related to the user interaction results in a much worse segmentation performance.

In the first and the third row of Figure 3.17, the input image has highly complex color structure; furthermore, the color profiles of foreground and background are similar to each other. Therefore, to use only the likelihood terms fail to properly segment the image. In the last row, there are two objects in the image and the user is interested with only one of them. Soft-label graph-cut fails to distinguish between these two objects since both of the objects have almost the same color profile. Indeed, it also erroneously classify part of the background as foreground due to the color similarity. In the second row, there is no interaction around the head of the bird. Furthermore, color of the head is also available in the background. Moreover, in the fourth row, color of the hand and the ground are similar to each other. Therefore, resulting foregrounds areas found by the soft-label graph-cut algorithm are erroneous in both cases. It is surprising that soft-label graph-cut algorithm yields worse performance than no error

correction (hard-label graph cut). This result actually shows the importance of the hard-labels (infinite terminal weights).

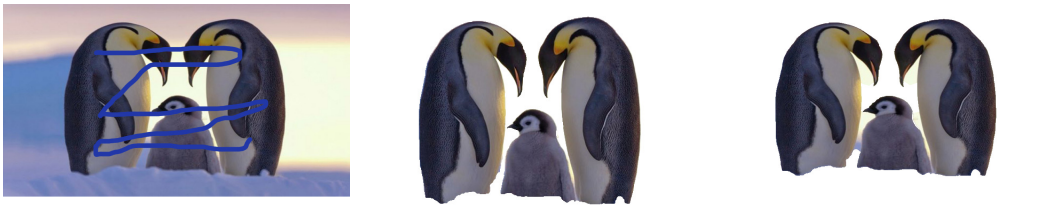
It can be argued that the main reason for the performance degradation of soft-label graph-cut is the lack of spatial information. Since all models used in the system are global color models, spatial information is not used in the energy definition. Only available spatial information is the nodes having infinite weights and this is discarded in the soft-label graph-cut.

In addition to the explained method, there exist an error correction mechanism which is recently proposed. In [66], energy definition and probabilistic model are altered to overcome interaction errors. All hard weights are replaced by the soft weights (likelihood values). In order to overcome the problem associated with the lack of spatial information, unary energies are not changed; however, binary energies are changed. Every node is connected to all other nodes in the graph. In other words, grid graph is replaced by a fully connected graph. Connections are set as sum of absolute differences (SAD) of image patches. Even if the interaction is erroneous, strong connections between the similar nodes does not let nodes having erroneous interactions to have erroneous labels. Furthermore, in order to achieve an acceptable efficiency, image patch differences are approximated over a manifold embedding computed via multi dimensional scaling [19] algorithm.

Graph formulation of [66] makes the method inapplicable to our framework due to the computational complexity of optimization and distance computation. However, for the sake of completeness, we also include experimental comparison of the proposed method and the method given in [66]. In Figure 3.18, sample images with user interactions are visualized together with their corresponding results. In all of the examples, the proposed method has a comparable error correction accuracy.

(a) Interaction for Bike Data. (b) Result of [66]. (c) Result of the proposed method.

(d) Interaction for Cow Data. (e) Result of [66]. (f) Result of the proposed method.



(g) Interaction for Penguin Data. (h) Result of [66]. (i) Result of the proposed method.

(j) Interaction for Skeleton Data. (k) Result of [66]. (l) Result of the proposed method.

Figure 3.18: Comparison of the proposed method and binary accurate segmentation algorithm [66]. Interactions and results of the algorithm given in [66] and the proposed method are shown in each column, respectively. Both methods have comparable segmentation accuracies and error tolerances.

CHAPTER 4

INTERACTIVE VIDEO SEGMENTATION

Similar to image segmentation, video segmentation is another crucial step in many multimedia and computer graphics applications. Hence, problem of video segmentation has been widely studied in many application scenarios. Moreover, both automatic and interactive video segmentation problems have been dealt with by many researchers.

A video can be seen as a sequence of frames; hence, one can clearly claim that given a reliable segmentation procedure for each frame, the data associations among frames can easily be handled. Therefore, any image segmentation method can be converted into a video segmentation algorithm. However; as explained in Section 3, there is no automatic image segmentation method with a guaranteed performance in general. Although interactive segmentation algorithms are more successful, interacting with each frame is not reasonable. Even the extension of image segmentation methods to 3D is not reasonable, since the information among the temporal direction is highly correlated and requires special attention. Therefore, video segmentation problem is significantly different than that of images and need to be solved independently.

When an interactive multimedia application is considered, efficiency, segmentation quality and quality of user interaction are all equally crucial. Therefore, the interaction should be at minimum level, whereas the algorithm should be efficient and the results should be accurate. Indeed, these requirements pose a serious dilemma for video segmentation problems in multimedia applications. Current state-of-the-art algorithms are analysed in terms of these three aspects in Section 4.1. A novel algorithm which overcomes the drawbacks of the existing algorithms is explained in Section 4.2. Finally, the proposed algorithm and the existing state-of-the art methods are compared in terms of explained metrics in Section 4.3.

4.1 Related Work

Automatic image segmentation is considered to be an ill-posed problem due to the ambiguous definition of an "object" and lack of prior information for a successful object segmentation. On the other hand, when temporal consistency of the objects in a video is considered, it is possible to re-define object segmentation as moving object segmentation. By the help of extraction of accurate motion information from a video, it is possible to accurately segment a moving objects in the video [3, 73, 65]. However, general multimedia or computer graphics applications do not put any

constraint on the object of interest, the object of interest might not be moving in the video at all. Main ambiguity is caused by the definition of an object. If the object is defined properly, it is possible to incorporate this information and resolve the ambiguity. Definition of the object is revisited by the help of a saliency metric [4, 39]. It is possible to incorporate saliency based definition of object, motion information and statistical graph-based methods to obtain accurate segmentations automatically [47]. Although fully automatic video segmentation is out of the scope of this thesis, details of the *Key-Segments* algorithm [47] is explained in Section 4.1.1 for the sake of completeness.

Interactive video segmentation algorithms start with an interaction stage on a single frame [8, 20, 48] or along the spatio-temporal domain [75]. After the interaction stage, as in the case of graph-based interactive image segmentation method, a set of low-level features are extracted from the video. Motion information, color, texture and shape are the most common features used in video segmentation scenarios. Then, an either two-dimensional (spatial) or three dimensional (spatio-temporal) energy function is defined. Moreover, this energy function is minimized via discrete optimization methods for binary or multi-label cases. Selection of the dimension depends on the modelling of the features and their relations, as well as, computational efficiency requirements. Higher order models are more reliable in terms of temporal coherency; however, optimization over a two dimensional grid is much more complicated. In Rotobrush [8], interaction in the form of scribbles is combined with spatial information, motion information and energy minimization which are explained in Section 4.1.2.

Although energy minimization methods and global color/texture models are commonly used in the literature, it is not possible to reach real-time performances by these approaches. Classical implementation of a min-cut/max-flow energy minimization requires 200 millisecond by a conventional desktop PC having 2.8 GHz Intel Core i7 processor . In order to solve the computational efficiency drawback, there exist fast heuristic-based methods in the literature. For example, geodesic video matting [20] uses fast marching algorithm and computationally simple matting update rule. Details of the geodesic video matting procedure is explained in Section 4.3.

In addition to the energy minimization based methods, there exist graph-clustering based approaches. In a typical energy minimization-based approach, interaction is assumed to be in the form of foreground/background regions or boundaries. However, it is also possible to design a general clustering algorithm dependent on some parameters and let the user interact with the system through these parameters. In [36], a spatio-temporal graph clustering algorithm is developed via motion information, over-segmentation and color features. Then, parameter to adjust object sizes is left as a free parameter to the user. Finally, a multi-object segmentation is performed via graph clustering. However, it requires a huge memory size and a computation capacity; entire video or a large chunk of frames is required to be stored into the memory. Moreover, spectral clustering and optical flow computation requires high computational power. Hence, the algorithm [36] is not applicable to the mobile multimedia scenarios.

There are also heuristic-based extensions of the interactive image segmentation methods. These methods [48, 76] start with an interaction in a key-frame and use global shape and color models for the entire video. Next, segmentation is performed for the video using this extracted global model. However, these methods discard the motion

information and spatial information. Therefore, their segmentation results are reliable only for a few frames.

4.1.1 Key-Segments Algorithm for Automatic Video Segmentation

Key-segments algorithm [47] consider the problem of video segmentation as an automatically finding object-like regions in each frame of the video. Then, these object-like regions are assigned an objectness score via saliency information. Final segmentation is obtained via processing of these objectness scores.

Possible object-like regions are found via category independent object proposals [28] method. Category independent object proposals method starts with the over-segmentation of the image obtained via hierarchical over-segmentation. Then, possible regions are obtained by giving each over-segment as a seed to the binary segmentation algorithm. Therefore, a large number of regions are extracted from the image by using many seed points.

After finding possible object like regions, the algorithm needs to find top-regions that possibly correspond to the objects. This ranking is performed in two parts. First, a set of features are extracted from each region r and regression based approach is used to find an objectness score $A(r)$ for each region. Regression is performed in a category independent manner [28]. The second score is obtained via motion information. Histograms of the optical flow of region r and its outside r' are computed in L_1 normalized form. Then, a motion-based objectness measure is computed in terms of χ^2 distance as

$$M(r) = 1 - e^{-\chi^2_{flow}(r,r')}. \tag{4.1}$$

This equation can be considered as the motion saliency information. Final objectness $S(r)$ is the combination of appearance and motion objectness scores by the relation:

$$S(r) = M(r) + A(r). \tag{4.2}$$

After the region proposals are found and a score is assigned to each region, a set of top scoring regions from each frame are gathered in a pool redundantly covering the video. Since these regions are the most likely object like regions, all objects are expected to be the combinations of these regions. Hence, these regions are needed to be clustered into sets of object proposals. Clustering is performed via the spectral graph clustering method. Each pair of regions in the top scored regions set is assigned a distance with a metric utilizing color and size differences. The distance metric is selected as a χ^2 distance of unnormalized color histogram as:

$$K(r_m, r_n) = e^{-\frac{1}{\Omega}\chi^2_{color}(r_m,r_n)}, \tag{4.3}$$

where Ω represents the mean of χ^2 distance over entire image. Finally, this similarity matrix, K, is used to find clusters. Each eigenvector of K corresponds to a single cluster. Then, resulting eigenvectors are also ranked according to their objectness value $S(r)$. For now, a set of object hypothesis, with their objectness score, is obtained.

Key-segments algorithm [47] considers the highest ranking region group as foreground model. Moreover, shape and color models are estimated from this top ranking region.

Finally, a spatio-temporal graph is constructed by using the 4-neighbour spatial arrangement and temporal links obtained from optical flow information. Energy function using color and shape features is constructed over the spatio-temporal graph. Finally, min-cut/max-flow energy minimization is used to segment the entire video.

Entire process of the key-segments algorithm is summarized in the Figure 4.1. Some key segments, shape and color likelihood maps are also presented in Figure 4.1.

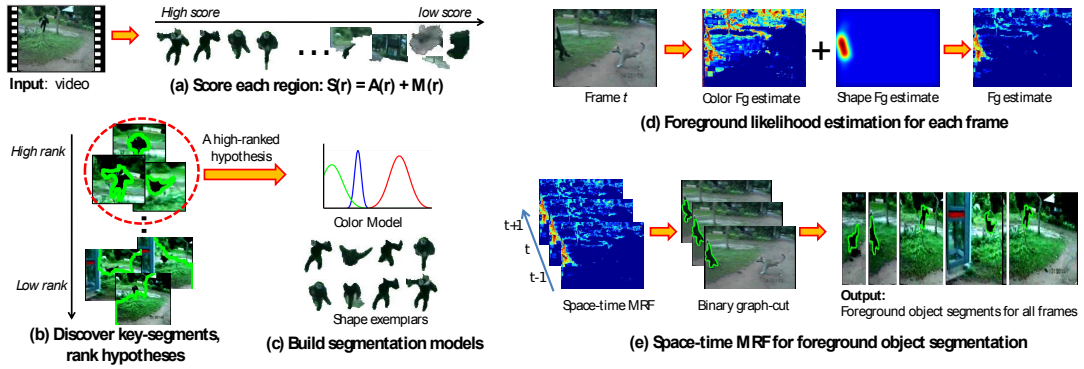


Figure 4.1: Key-segments [47] algorithms work flow [47].

Although the results of key-segments algorithm look quite promising, to find motion estimation and saliency measures are computationally expensive. Therefore, it is almost impossible to use them in any interactive or mobile multimedia application. On the other hand, the resultant segmentation framework is entirely based on graph-theoretical formulation. Moreover, our proposed efficiency improvements are applicable to key-segments [47] algorithm. Therefore, we extend the key-segments [47] by using the proposed MRF energy propagation as explained in Section 4.2.1. An extended version of key-segments is explained in Section 4.2.4; moreover, efficiency of the key-segments algorithm and the proposed extensions are analysed in Section 4.3.

4.1.2 Video Snap Cut (RotoBrush)

Video Snap-Cut [8] is a state-of-the-art high quality interactive video segmentation tool with wide range of users. After its introduction; due to its accuracy and interaction quality, video snap-cut was included in a widely used video editing tool, namely Adobe After Effects [68] by the name of *Rotobrush*.

Interaction scenario of Rotobrush [8] depends on interactions at some key frames of the video. Furthermore, any interactive segmentation tool can be used to initialize Rotobrush [8]. After the interactive segmentation stage, algorithm iteratively solves for all other frames of the video. Moreover, every frame of the video is processed independently. Independent processing of frames has two reasons: time and memory efficiency. Solving two dimensional optimization problems are much more efficient than solving their three dimensional counterparts. Moreover, keeping all frames in the memory is not possible for most of the long videos.

It is not possible to use the formulation of the interactive image segmentation during

interactive video segmentation scenario with a guaranteed segmentation accuracy. This result is due to the fact that in most of the interactive image segmentation methods only the color information is used and the spatial information is discarded. Moreover, most of the interactive image segmentation algorithms use global models, such as Gaussian mixture models (GMM) or Kernel Density Estimation (KDE). Rotobrush [8] is introduced two solve these two main problems.

In the formulation of Rotobrush [8], boundary of the object of interest is obtained via localized classifiers. The boundary of the object of interest is represented via a set of overlapping local windows. Consider Figure 4.2; jumping baseball player is the object of interest and it is represented via a set of rectangular overlapping classifiers. Moreover, every local classifier has a probability density function (p_{fg}) in terms of color and shape features. These feature sets are combined to obtain a final probability map. Finally, the resultant probability map is used to obtain a final segmentation mask as shown in Figure 4.2.

Initialization of the algorithm starts with a uniform sampling of the initial object boundary at the interacted key frame. After the initial interaction, a set of constant size overlapping local windows covering the entire boundary are obtained via a uniform sampling. Each rectangle overlaps one-third of its size by the next rectangle. Then, motion information for each local rectangle is obtained via a dense optical flow estimation procedure [10]. Such moved boundary is used to update color and shape models for the next frame.

While moving local classifier among temporal direction, color models are estimated in terms of GMM in *Lab* color space [8] and shape features are modelled in a non-parameteric manner. Gaussian windows are used at each boundary point to model the shape features via Parzen window approach [24]. Variance of the Gaussian windows are selected by considering the color information. If the color profile is discriminative enough, shape prior is not needed. Therefore, variances of the Gaussian windows should be high. If the color profile is not discriminative, i.e. foreground and background have similar color profiles, shape prior should be given more importance. Therefore, variances of the Gaussian windows should be low. In order to define these models formally, we should start with color modelling. Probability of being foreground for a single pixel in terms of only color information can be stated as

$$p_c(x) = \frac{p_c(x|F)}{p_c(x|F) + p_c(x|B)} \tag{4.4}$$

where $p_c(x|F)$ and $p_c(x|B)$ are the corresponding GMM likelihoods. Furthermore, color confidence value can be formulated as expected value of the error on estimated probability. Since model is initially generated for a known segmentation result for the keyframe, the desired $p_c(x)$ values are equal to 1 for the known foreground and 0 for the known background. Expected value of the drift from the desired values can be considered as a confidence. By given more importance to the pixels near the object boundary, the color confidence can be formulated as [8]

$$f_c = \frac{\int_{W_k} |L(x) - p_c(x)| \cdot e^{-\frac{d^2(x)}{\sigma_s^2}}}{\int_{W_k} e^{-\frac{d^2(x)}{\sigma_c^2}} dx} \tag{4.5}$$

where W_k is the local region used by the classifier, $d(x)$ is the distance to the object boundary and $L(x)$ is the desired binary probability map. Moreover, σ_c is fixed to be the half of the window size.

Then, the color confidence value inversely affects the variance of Gaussians used to model the shape feature as [8]

$$\sigma_s = \begin{cases} \sigma_{min} + \frac{\sigma_{max}-\sigma_{min}}{(1-f_{cutoff})^r}(f_c - f_{cutoff})^r & f_{cutoff} \leq f_c \leq 1 \\ \sigma_{min} & 0 \leq f_c \leq f_{cutoff} \end{cases} \tag{4.6}$$

where σ_{min}, σ_{max} and f_{cutoff} values are tunable parameters. For the detailed analysis of the behaviour of the algorithm in terms of these parameters, the reader is referred to the original paper [8].

After finding the standard deviation of Gaussians are found, shape confidence for each pixel can be computed as [8]

$$f_s(x) = 1 - e^{-\frac{d^2(x)}{\sigma_s^2}}. \tag{4.7}$$

Shape confidence is combined by a shape prior to obtain the final a priori shape information. Shape prior is just the transformed local boundary by using the obtained optical flow information. Transformation is selected as the mean optical flow vector of the foreground pixels of the local window. Only foreground pixels are used, since we are interested in modelling the foreground object.

After these models are generated, the final probability map is obtained via a linear combination of color model, shape confidence and shape priors as shown in Figure 4.2.

The resultant probability map is combined by the Ising model [32] to define an MRF energy function. Moreover, this energy function is used to obtain the final segmentation mask. If it is desired, this resultant segmentation mask can be re-used to model color and shape. Moreover, the process can be continued until the convergence, as in the case of Grabcut [60].

Segmentation quality of the Rotobrush [8] is extensively discussed in Section 4.3 together with its computation time analysis. Although the results of the Rotobrush [8] is accurate, due to the usage of a computationally complex optical flow subprocedure and GMM estimation, Rotobrush [8] is highly inefficient. Therefore, it is not applicable to mobile scenarios. Even its Graphical Processing Unit (GPU) implementation is far from being real-time.

4.1.3 Geodesic Interactive Video Matting

As explained in Section 4.1.2, the main drawback of the interactive image segmentation methods in a video segmentation scenario is not exploiting the spatial information properly. The main idea of Geodesic Video Matting method [7] is to design a fast method which is solely based on spatial information. Indeed, geodesic video matting [7] is the direct extension of geodesic image segmentation.

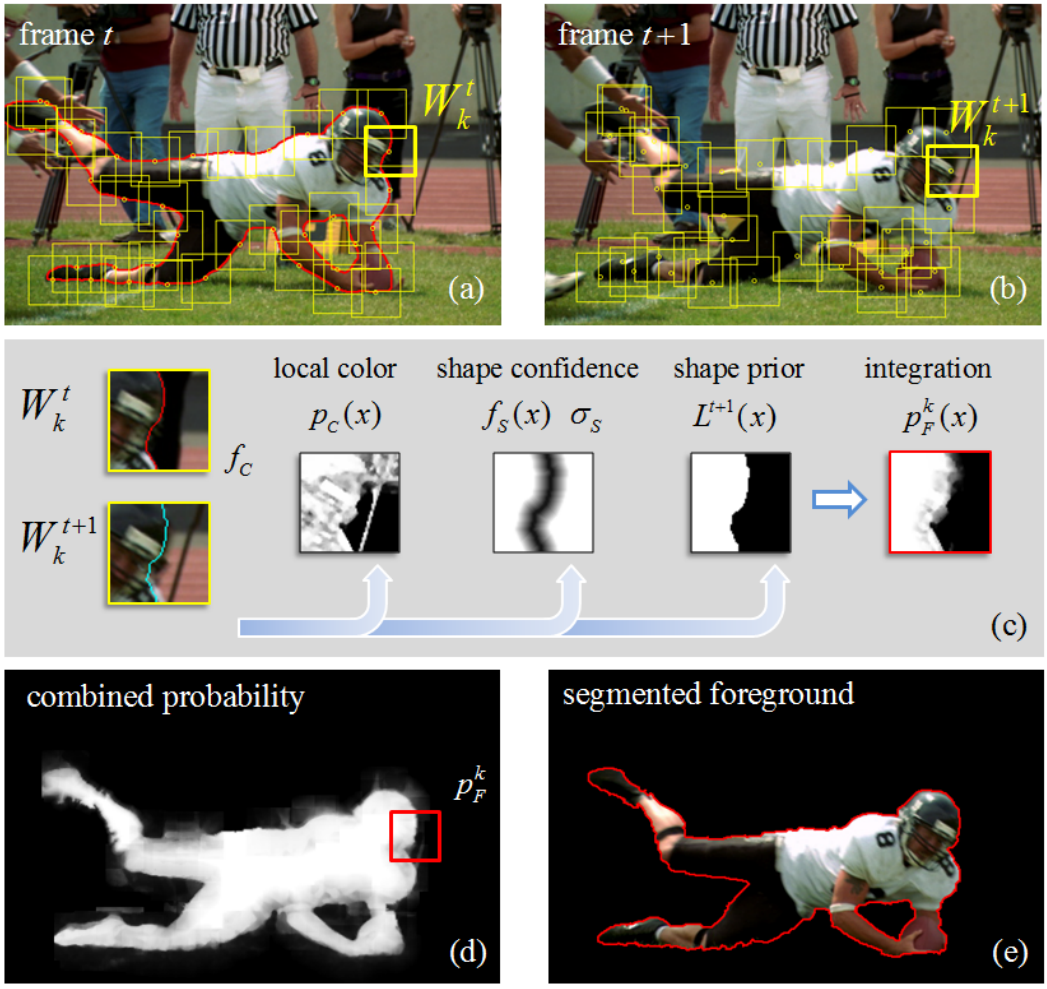


Figure 4.2: Visualization of the work flow of Rotobrush [8]. Local classifiers are the shown rectangles in time t (a). Then, local classifiers are moved to the corresponding positions in time $t+1$ by using motion information (b). Final probability map is obtained as weighted sum of the local classifiers (c) and minimized via min-cut/max-flow method (d) [8].

Consider the input image as a graph of nodes. It should be noted that final geodesic matting is an $O(N)$ algorithm. Hence, to convert the pixel based image representation into a superpixel-based image representation is not reasonable. It should be noted that computation time of oversegmentation is generally much higher than the time improvements obtained via a superpixel based representation.

Geodesic matting algorithm starts with the extraction of the set of low level features from the input image. By using the interactions and these low level features, one can estimate some global models and probability of being foreground for each pixel. GMM or KDE of color or texture features are widely used examples. A prior probability of being foreground can be computed as [7]

$$p_F(x) = \frac{p(x|F)}{p(x|F) + p(x|B)}. \tag{4.8}$$

After this computation, the main novelty of geodesic mating is to obtain actual prob-

ability values by considering spatial information. Probability of being foreground is modelled as the inverse distance to the foreground. Distance is defined as the geodesic distance to the interacted pixels via a weight graph of prior probabilities. Weight (W) is assumed to be the gradient of the prior probability map; hence

$$W(x) = \nabla p_F(x). \tag{4.9}$$

Then, this distance is defined over the weight graph as:

$$d'(s_1, s_2, C) = \int_{s_1}^{s_2} |W(x) \cdot \dot{C}_{s_1,s_2}(x)| dx, \tag{4.10}$$

where $C_{s_1,s_2}(x)$ is the path connecting s_1 and s_2, and $\dot{C}(x)$ is the edge on the path $C(x)$. Hence, the distance over a path is defined as the sum of edge weights of the edges of the path. Then, a distance between two points is defined as the minimum of these distances. In other words, the distance between two points is the geodesic distance between points [20] by the following relation:

$$d(s_1, s_2) = min_C d'(s_1, s_2, C). \tag{4.11}$$

In other words, the distance between two points is the minimum of distances of all possible paths connecting these points. Furthermore, distance to foreground is defined as the minimum of distances to the foreground scribbles. Similarly, distance to background is defined as minimum of distances to the background scribbles as

$$D_l(x) = min_{s \in \Omega_l} d(x, s) \quad l \in F, B \tag{4.12}$$

where Ω_l is set of foreground or background scribbles depending on l. Finally, the probability of being foreground can be computed as

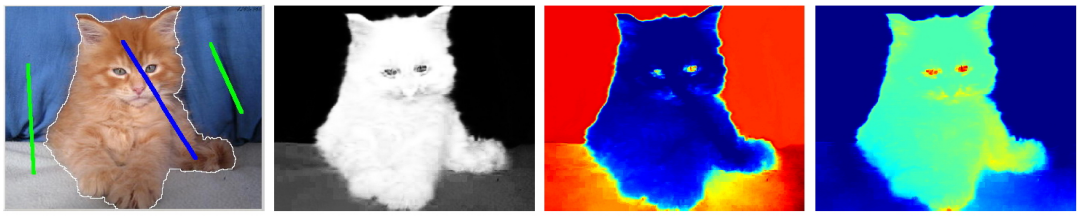
$$f_F(x) = \frac{D_F(x)}{D_F(x) + D_B(x)}. \tag{4.13}$$

If we assume KDE is used to model the global color feature, and explained definition of foreground and background is used, the resultant probability values can be used to either find matte values or binary segmentation results, trivially. $f_F(x)$ is the matte value for pixel x and $f_F(x) > f_B(x)$ is the binary decision rule for the binary segmentation case. A sample input image with scribbles, probability maps for foreground and background, as well as binary segmentation results are shown in Figure 4.3.

Idea of geodesic distance implicitly use spatial information. Since the image is not smooth enough, even for the paths over the same image, the longer path will have higher cost than the shorter path. Furthermore, one can increase the importance of spatial information by modifying the weight function as

$$W(x) = \nabla P_F(x) + \lambda. \tag{4.14}$$

Then, every edge in the path will induce a fixed λ cost. The value of λ parameter can be tuned to control the relative weight of color difference and spatial information. Moreover, computation of the geodesic path is efficient, if fast marching algorithm [77] is used. It is possible to obtain the geodesic distance of all nodes in a liner time via fast marching algorithm [77]. Details of the geodesic distance estimation is beyond the scope of this thesis. Therefore, the reader is referred to the original manuscript [77].



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

Figure 4.3: Visualization of input image, scribbles and distance values. In (a) input image is shown with binary segmentation result and input scribbles. In (b) prior probability map $p_F(x)$ is visualized where white represents high probability and black represents low probability. Moreover, in (c) and (d) computed distances to the foreground and background scribbles are visualized respectively where red represents high distance and blue represents low distance.

Extension of geodesic matting to video segmentation case is rather trivial. However, using global models in the frame other than the key-frame is not reasonable since color profile may change significantly due to occlusions. The basic solution is to use Euclidean color differences to find weights. Therefore, in the spatio-temporal graph, the spatial edges are given the weight of gradient of the prior probability map and the temporal edges are given the weight of Euclidean color differences between pixels.

The only problem arising in this extension is due to the occlusions. Consider the case in Figure 4.4; region A is part of the foreground and B is the part of the background. Moreover, they occlude each other at some frame between two key frames. Since the distance is 0 when objects occlude each other, these distances will propagate and region B will be erroneously classified as foreground. In order to solve this issue, initial strategy is not to let paths in the negative time directions. This solution can be accomplished via assigning infinite weights to the edges going in negative temporal directions. This approach solves the tube going in backward direction as shown in Figure 4.4.c. In order to solve the tube going in forward temporal direction, another key frame is used. Consider a key frame in future time; then segmented object can be back propagated and all operations can be applied in the reverse direction. Hence, the resulting erroneous tube will disappear.

The experimental results for the segmentation accuracy and computation of geodesic matting [7] are analyzed in Section 4.3. The main drawback of the geodesic matting is the unreliable behavior of the geodesic distance in noisy images. Indeed, the proposed method also utilize the idea of geodesic distance as explained in Section 4.2. However, our method keeps the temporal support region limited and puts an extra coherency penalty to get rid of this drawback.

4.2 Proposed Method

As explained in Section 4.1, the main dilemma of the interactive video segmentation problem is the trade-off between computation time and segmentation accuracy. If spa-

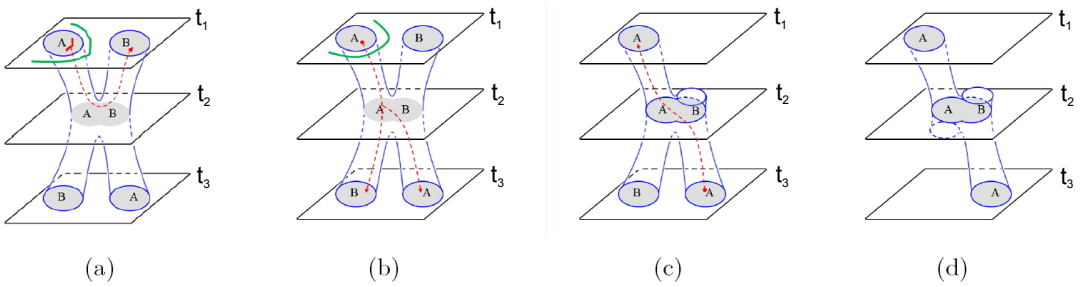


Figure 4.4: Visualization of the problem caused due to the occlusions. In (a) and (b), A is selected as foreground and B is selected as background in key frame t_1. After an occlusion in t_2, foreground leaked to the region B in all frames. By putting forward temporal direction constraint, erroneous foreground region between time t_1 and t_2 is solved. Moreover, by selecting region A as foreground in t_3 and applying same procedure backwards in time, the erroneous region between time t_2 and t_3 is also solved.

tial information and motion information are discarded, near real-time performances are possible. However, segmentation accuracy significantly drop. On the other hand, if motion information is included, segmentation accuracy increases; however, computation time increases. Geodesic video matting is a method, whose objective is to exploit spatial information without extracting motion information. However, geodesic distance is not robust for long temporal directions. In summary, computation time and segmentation accuracy requirements create a serious dilemma for the interactive video segmentation problem.

For the proposed method, in order to solve this dilemma, we redefine the video object segmentation problem as an estimation of the foreground probability density function of each frame in terms of probability density functions of the previous frames. By the help of Markovian property assumption, one can define such a problem as the estimation of the MRF energy of each frame in terms of MRF energy for the previous frames. By this definition, we constraint the explicit support of any frame only to the subsequent frame. Long term support is implicitly possible after correction in each frame. Hence, the main objective of the proposed method is to combine robustness and accuracy of the energy minimization methods by the efficiency of the local graph search, such as geodesic distance computation.

In our framework, we assume that MRF-based energy function for the first frame is already known. In practice, this unknown energy function can be obtained via user interaction. We propose an efficient method to propagate energy distributions throughout the frames via bilateral filters. The weights of these filters are selected by using texture similarities and spatio-temporal relations. Moreover, an increase in the segmentation quality is expected by the help of simultaneous usage of spatial and texture cues. In order to increase the efficiency of the method further, we also propose a dynamic algorithm for the solution of the min-cut/max-flow problem for bilateral filtering scenario. The proposed method does not use any global shape or color model; it does not use motion information, either. We compensate for the lack of motion information via extensive usage of spatial information during the calculation of the

bilateral filter weights. Indeed, for the videos having limited amount of motion, the proposed method results in a high quality segmentation result even for the non-rigid objects. The proposed method is general enough to convert any MRF-based interactive image segmentation algorithm to an interactive video segmentation method. Moreover, it can also be used to speed-up any automatic video segmentation tool.

The proposed algorithm is explained in three main parts: In Section 4.2.1, a general method to estimate the MRF energy of a frame by using MRF energy of the previous frame is proposed via spatio-temporal bilateral filtering. In Section 4.2.2, an efficient approximation of geodesic bilateral filter is introduced to increase the computational efficiency. Finally, in Section 4.3.3, a dynamic method to efficiently solve graph-cuts in linear filtering scenarios is proposed.

4.2.1 Video Segmentation as MRF Energy Propagation

The proposed framework solves video segmentation problem for each frame separately in order to reduce the dimension since solving 2D graph-cuts is more efficient than solving 3D spatio-temporal graph-cuts. Hence, one can define video segmentation problem as estimating the MRF energy of the current frame by utilizing the MRF energy of the previous frame. In other words, the proposed estimation method propagates the energy functions throughout the frames.

Before starting to explain the details of this propagation method, we should summarize the usage scenario for such a propagation. The proposed method can either be used as an interactive video segmentation tool or speed-up tool for an automatic video segmentation procedure. Any MRF energy minimization-based interactive image segmentation method [60, 21, 12] can be used to generate MRF energy of the initial frame. Then, the obtained energy function can be propagated via the proposed algorithm. Hence, the overall algorithm can be used as an interactive segmentation tool. For an automatic video segmentation case, any optical flow based method that exploits MRF energies can be used to segment the initial frame [36]; then, the resultant energy can be propagated by the proposed method to speed-up the algorithm for the rest of the video.

In order to explain the method, which estimates MRF energies in terms of the MRF energy of the previous frame, we first need to state the explicit form of the energy function. As explained in Section 2.3, there exist a particular class of MRF energies which can be optimized efficiently via min-cut/max-flow method; namely sub-modular energy functions [42]; we are specifically interested in this class of energy functions. Generally, these energies composed of two main parts; the first part is the unary term which represents the consistency of the segmentation labels against some predefined models. The other term is the binary one and represents the coherency of the segmentation result. MRF energy over a graph, $G(\mathcal{V}, \mathcal{E})$, can be represented as

$$E(\alpha) = \sum_{v_i \in \mathcal{V}} U(\alpha_i, z_i) + \sum_{v_i \in \mathcal{V}} \sum_{v_j \in N(v_i)} V(z_i, z_j)\phi[\alpha_i \neq \alpha_j]. \tag{4.15}$$

We will refer to the nodes as regions in the image; these regions might correspond

to pixels or over-segments depending on the application. In this representation, α_i represents the label of the region i, z_i represents the color, shape and/or motion information related to region i and $N(v_i)$ represents the neighbour regions of region i. $\phi[x]$ is an indicator function, yields one, if x is true and zero otherwise. $U(\alpha_i, z_i)$ is the unary energy corresponding to feature vector z_i and label α_i. Moreover, $V(z_i, z_j)$ term represents the penalty associated for giving different labels to two neighbour regions, where α represents the concatenation of labels of each region in the image. As a clarification to the notation, $\mathcal{V}$ corresponds to set of nodes (set of v_i) and $V(v_i, v_j)$ corresponds to the binary energy terms associated with node i and j.

When we consider the limited amount of object motion, we can safely assume that there exists a matching region (i.e. a region with a similar appearance) in the previous frame for each region in the current frame, except for the occlusion regions. Therefore, one can argue that MRF energy of the current frame can be approximately predicted in terms of the MRF energy of the all regions in the previous frame.

In order to proceed further, one should assume that there exists a distance metric between each region in the current frame and the previous frame. This distance metric should correspond to the spatio-temporal distance and texture similarity, simultaneously. Among alternatives, geodesic distance is a suitable metric to be used in such a setting. Geodesic distance can be considered as the minimum cost among the paths connecting two specified regions. In the proposed framework, this cost can be defined as the weighted sum of the color differences along the spatio-temporal path and length of the path. In the proposed method, an approximation to the geodesic distance is utilized; the details of this distance metric are explained in Section 4.2.2.

The representation of the energy function in terms of the previous energy function is defined separately for U and V terms. For the unary terms (U), the unary energy of each region in the current frame t is estimated by the weighted sum of the energy of each region in the previous frame $t-1$. Moreover, these weights are selected as inversely proportional to distances between regions. In other words, for any region in t, unary term is written in terms of the weighted sum of the unary terms in $t-1$ as;

$$U^t(\alpha_i^t, z_i^t) = \frac{1}{\gamma_i^t} \sum_{v_j^{t-1} \in \mathcal{V}^{t-1}} U^{t-1}(\alpha_j^{t-1}, z_j^{t-1}) e^{-dis(z_i^t, z_j^{t-1})}, \tag{4.16}$$

where superscripts represent the time instants and $dis(z_i^t, z_j^{t-1})$ represents the aforementioned distance metric between region i in time t and region j in time $t-1$. γ_i^t is used for normalization and it can be computed as

$$\gamma_i^t = \sum_{v_j^{t-1} \in \mathcal{V}^{t-1}} e^{-dis(z_i^t, z_j^{t-1})}. \tag{4.17}$$

The main rationale behind the relation in (4.16) can be explained by considering each region in the previous frame as a model, and assuming the current setup as a mixture of these already calculated models. As in the case of mixture models, the resultant unary energy is a weighted sum of the each model in the mixture. Another rationale can be put forward by considering each region in the previous frame as a moving object. Therefore, every region in the current frame will either corresponds to an object in the previous frame or alpha matting of different overlapping objects. When the small

segments generated by an over-segmentation algorithm or even pixels are considered as regions, every region in the current frame should have many matches in the previous frame; therefore, instead of exact matches, linear combination of all matches can be used. The selection of these matches and computation of their corresponding weights are implicitly performed by the help of the proposed distance metric.

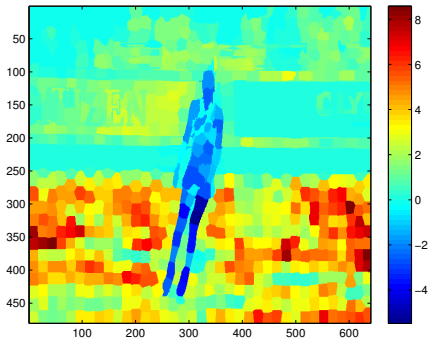
For the binary terms (V), the conventional approach typically utilizes the frequently preferred Potts Model; $V(z_i, z_j) = e^{-\frac{|z_i - z_j|}{\beta}}$ (β is used for normalization). This penalty is generally considered as inversely proportional to the color differences in order to generate coherent segmentation results. However, if we use this conventional approach, it is impossible to use the proposed dynamic optimization method explained in Section 4.3.3, since both unary and binary energy terms of the current frame are assumed to be linearly dependant on unary and binary energy terms of the previous frame. Therefore, we slightly change the coherence penalty; we consider the graph of edges (i.e. dual graph of nodes) and apply the same propagation rule to this graph. Therefore, the relation between the current and previous binary energies are represented as:

$$V^t(z_i^t, z_j^t) = \frac{1}{\gamma_{ij}^t} \sum_{v_k \in \mathcal{V}^{t-1}} \sum_{v_l \in N(v_k)} e^{-dis(z_i^t, z_k^{t-1})} e^{-dis(z_j^t, z_l^{t-1})} V^{t-1}(z_k^{t-1}, z_l^{t-1}). \quad (4.18)$$

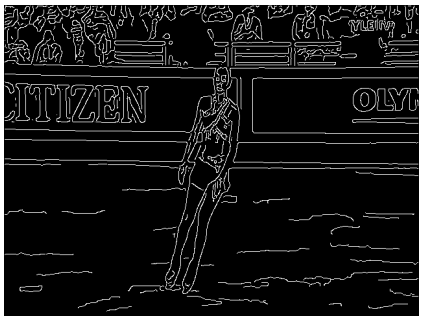
This definition of the binary energies can actually be related to smoothing the conventional binary terms. For the case of linear and piecewise linear penalty functions, it can be shown that the definition of the proposed binary penalty is equivalent to applying spatio-temporal edge-aware smoothing to video frames, and then, computing conventional binary terms. In other words, defined binary energy is equivalent to the conventional binary energy used in [13] with an extra prior smoothing operation. Hence, defined binary energy can be considered as color smoothness penalty with an additional temporal smoothness. Furthermore, this equivalence can easily be proven via straightforward application of linearity.

Main advantages of the proposed estimation method are generality in terms of distance function selection and extended support region. Although the proposed system is based on geodesic distances, the proposed energy propagation can also be accomplished by a variety of distance functions accompanying color, motion and shape. Moreover, in this formulation, each region in the previous frame actually supports every region in the current frame. Therefore, any region in the current frame is supported by all the regions of the previous frame. Such an approach relaxes limited motion assumption significantly, and eliminates the necessity to use any global model.

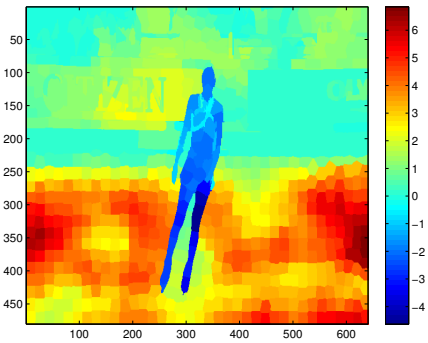
In order to visualize energy propagation, we apply the proposed method to a typical example. The energy of the first frame is computed via the interactive image segmentation method [21], whereas the energy of the 5^{th} frame is estimated via the proposed method. As in the case of [21], over-segments are obtained by using SLIC algorithm [1]. The resultant propagation is illustrated in Figure 4.5, where it suggests that the estimated unary terms are much smoother than the original terms due to the wide support region. It can also be observed that utilization of (4.18) for binary weights is feasible and the result is consistent with the edge map.



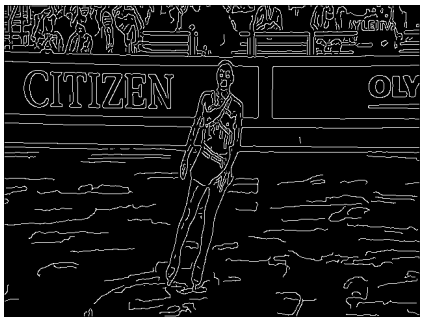
(a) $U^1(\alpha_i^1, z_i^1)$



(b) $V^1(z_i^1, z_j^1)$



(c) $\hat{U}^5(\alpha_i^5, z_i^5)$



(d) $\hat{V}^5(z_i^5, z_j^5)$

Figure 4.5: Visualization of Energy Propagation. Unary $U^t(\alpha_i^t, z_i^t)$ (a) and Binary energy $V^t(z_i^t, z_j^t)$ (b) terms of frame t=1 is computed via user interaction, whereas energy terms of frame t=5 are estimated via the proposed energy propagation method. Estimated binary energy (d) is consistent with the real edge map of the image. Estimated unary energy (c) is also consistent with the actual foreground/background probabilities of the regions. Furthermore, estimated unary energy (c) is much smoother than the unary energy found via user interaction (a) due to the wide support region.

4.2.2 Information Permeability Filter / Bi-exponential Edge Preserving Smoother

Energy propagation method, explained in Section 4.2.1, estimates the MRF energy of each frame via the formulation using the bilateral filtering. Hence, segmentation results can be obtained via minimization of the corresponding energy function. However, the proposed energy propagation method is not applicable to interactive and mobile multimedia applications due to high computational complexity of bilateral filters and computation of geodesic distances. It should be noted that the relations in (3.16) and (3.18) correspond to a cross filtering by a bilateral filter [15]. Naive implementation of a bilateral filter requires $O(n^2)$ operations; indeed, each operation requires a computation of a geodesic distance which has a linear time complexity via a fast marching algorithm [77]. Therefore, the overall complexity of the estimation is $O(n^3)$ which is far from being acceptable. In order to solve this efficiency problem, we utilize the recently proposed approximation of geodesic bilateral filter [15, 71].

Bilateral filters are widely used for edge-preserving smoothing operations in vision research [6]. They can be considered as smoothing filters whose weights are obtained from a Gaussian kernel. Later, bilateral filters are also attributed to anisotropic-diffusion filters [58]. However, their high computation cost made them unfavourable until some recent improvements of the computation time are achieved. In [25], bilateral filter is accelerated by using quantization and piecewise linear approximations. Alternatively, in [57], bilateral filtering is performed by using linear convolutions in higher dimensions. Recently, a constant time bilateral filter is proposed independently by Cigla et. al. [15] and Thevenaz et. al. [71] under different names as *Information Permeability (IP)* and *Bi-Exponential (BE) filter*. For the sake of fairness, we refer to this filter as IP/BE.

In IP/BE filter, bilateral filtering is performed by using a dual one-tap recursive filter (IIR) for each dimension in the 2D spatial domain (vertical and horizontal). By the help of the recursive filter, one can achieve full image support for any pixel by a Gaussian kernel. 1D one tap recursion filter is applied in two directions (both horizontal and vertical) separately. For a sequence $x[n]$ of length N, the recursion is applied as follows;

For positive direction (left to right, [15, 71]):

$$\hat{x}_1[k] = x_1[k] + \hat{x}_1[k-1]r(x[k], x[k-1])$$

whereas, for negative direction (right to left):

$$\hat{x}_2[k] = x_2[k] + \hat{x}_2[k+1]r(x[k], x[k+1]).$$

In this representation, $r(x[k], x[k-1])$ is the filter weight and can be computed as $r = e^{-\frac{|x[k]-x[k-1]|}{\sigma^2}}$. Moreover, this recursion should be initialized as $x_1[n] = x_2[n] = x[n]$. The values after the recursion needs an extra normalization operation. During this normalization, the constants are computed via the same recursion. Consider a sequence of 1's with length N, as $1_1[n]$ and $1_2[n]$; then the normalization constants are, for positive direction,

$$\hat{1}_1[k] = 1_1[k] + \hat{1}_1[k-1]r(x[k], x[k-1]),$$

and, for negative direction,

$$\hat{1}_2[k] = 1_2[k] + \hat{1}_2[k+1]r(x[k], x[k+1]),$$

The final smoothed values are computed as [15, 71]:

$$y[k] = \frac{\hat{x}_1[k] + \hat{x}_2[k]}{\hat{1}_1[k] + \hat{1}_2[k]}.$$

These 1D recursion filters can be applied to horizontal, spatial and temporal directions separately. Moreover, these results can be combined as explained in [15, 71]. It should be noted that, IP/BE filter is explained for smoothing scenario. In the proposed method, IP/BE filter is used as a cross filter. $x[k]$ values correspond to the unary and the binary energy terms at $t-1$, and, $\hat{x}[k]$ values correspond to unary and binary energy terms at t. Moreover, the weight values are obtained via the color values of the image instead of the energy terms. In other words, $r(x[k], x[k-1])$ is replaced by $r(z_k, z_{k-1})$.

It should also be noted that these 1D filters are applied along horizontal, vertical and temporal dimensions separately. Hence, their ordering changes the result. We apply the filters in both orders in the spatial domain, then in the temporal domain (horizontal, vertical, temporal and vertical, horizontal, temporal), and use the average of the results as the final propagated energy.

In [15], it is argued that IP/BE filter approximates a geodesic bilateral filter. Although the weights do not correspond to the geodesic distances, they correspond to the cost of the path having a single horizontal and a single vertical piece. Moreover, the quality of this approximation is demonstrated in an edge preserving smoothing problem [15, 71]. Although this approximation could have some failure cases, it can still be used in most of the practical cases.

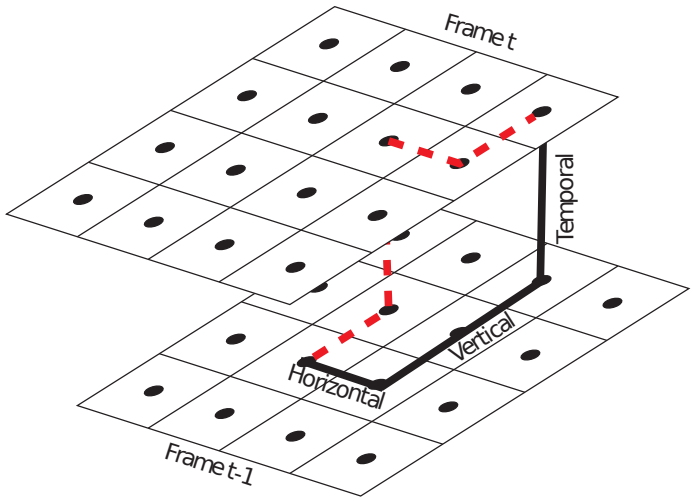
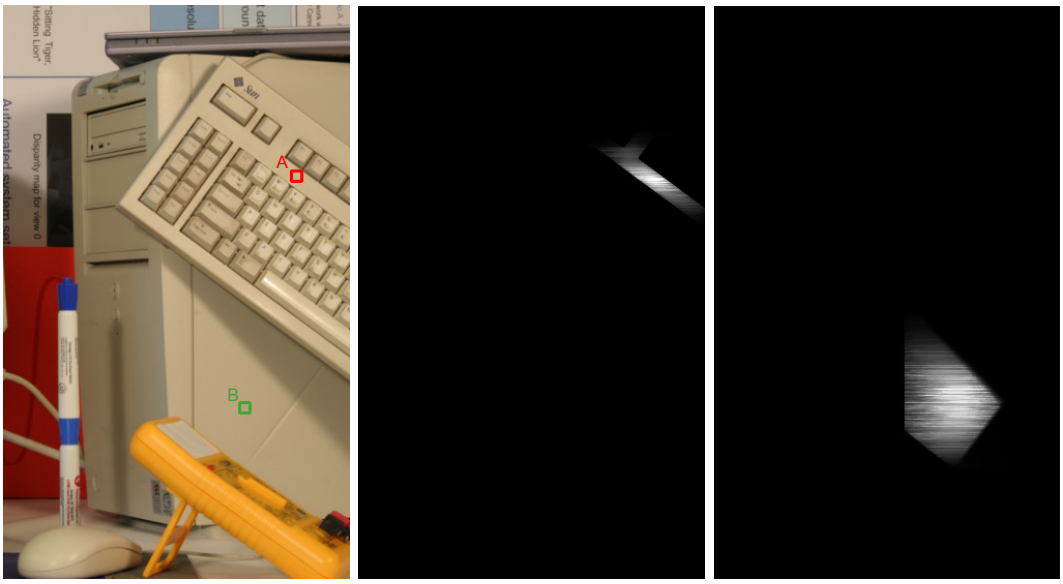


Figure 4.6: Illustration of step-like paths used in IP/BE filter. Instead of a geodesic path shown as red dashed line (arbitrarily selected), a path having single horizontal, vertical and temporal components is used. Distance between two nodes is computed as the sum of color differences along this step-like path via the IP/BE filter.

When such filtering is used, the resulting MRF energy for the current frame is equivalent to the energy defined in Section 4.2.1 by $dis(z_i^t, z_j^{t-1})$ as the cost of the step like path shown in Figure 4.6. In other words, instead of taking the distance as the minimum of the sum of the color differences among the paths connecting point z_i^t and z_j^{t-1}, this distance is taken as the sum of the color differences along the path having three components on each dimension (x, y and t). For the videos not having serious level of noise or high amount of motion, the proposed cost approximates the geodesic distance [15].

For the visualization of the effectiveness of the approximation, a sample image is shown in Figure 4.7. A set of sample pixels of the image are selected. Moreover, filter weights for this sample point is visualized in Figure 4.7. As shown in Figure 4.7, only the pixels of the object affects the selected pixel. Hence, the resultant filter have wide and accurate spatial support. It should also be noted that the filter adaptively adjusts its support region as shown in Figure 4.7.

When the proposed method is analysed, the number of operations is linear with the



(a) Selected pixels (b) Support region for -A- (c) Support region for -B-

Figure 4.7: Input image with two sample selected pixels is shown in (a). The support region for the selected pixel -A- (center of the red square) is shown in (b), whereas the support region for the selected pixel -B-(center of the green square) is shown in (c). Adaptive size of the filter support and its accurate shape consistent boundary of the object are clearly visible in (b) and (c).

number of regions in the frames, if the number of overlapping regions is bounded by a constant. Indeed, most of the over-segmentation algorithms have a parameter which controls the number of superpixels or the maximum size of a superpixel. Therefore, maximum number of overlapping regions can be adjusted. Hence, by proper selection of over-segmentation parameters, the proposed method runs in linear time. In summary; it is possible to approximate MRF energy propagation explained in (3.16) and (3.18) via an algorithm having linear time complexity by using the IP/BE filter.

4.2.3 Dynamic Graph-Cut for Varying Graph Structure

By using IP/BE filter and linear filtering based MRF energy propagation scheme, we propose a linear-time method to estimate MRF energy of a frame by using the previous frame. Moreover, for each frame, this energy function can be minimized via the min-cut/max-flow method optimally [13]. However, due to our challenging efficiency requirements, we need to further improve the min-cut/max-flow approach to increase the overall efficiency. Hence, we propose a method to recycle residual flows in a bilateral filtering scenario for non-lattice graphs.

Solution methods for MRF energy minimization problems via the min-cut/max-flow based approaches are explained in Section 2.3. Moreover, a dynamic version of the problem for the case of non-varying graph structure is explained in Section 3.2.3.1. However, this method is not applicable to a more general case, since the structure of the graph might change significantly due to superpixel-based representation. Over-

segment positions and sizes might change in each iteration of the over-segmentation method. Therefore, we propose another dynamic version of the min-cut/max-flow algorithm for such a varying graph structure case in linear filtering scenario.

In order to apply the conventional dynamic method to varying graph structure case, a computationally expensive graph matching problem needs to be solved between graph of the current frame and that of the previous frame. However, in the proposed method, there exists a linear relation between each node in the current frame and each node in the previous frame. Moreover, edge weights of the graph of the current frame are defined in terms of the edge weights of the graph of the previous frame, as in (3.16) and (3.17). Therefore, this relation needs to be exploited.

We propose to propagate the flows in the previous frame to the current frame via bilateral filter that is computed during the estimation step of MRF energies. More interestingly, we show that if the flows in the previous frame are propagated and pushed to the current frame, the resultant residual graph will be equivalent to applying the same bilateral filter (IP/BE cross filter) to the residual graph of the previous frame. In other words, in order to find the updated residual graph in time t, we simply apply the proposed bilateral filter to the residual graph in time $t-1$. Moreover, we claim that minimum cut result will be same. This claim is also explained in Figure 4.8 by an example. In order to keep the results as general as possible, we prove this claim for the general bilateral filtering scenario.

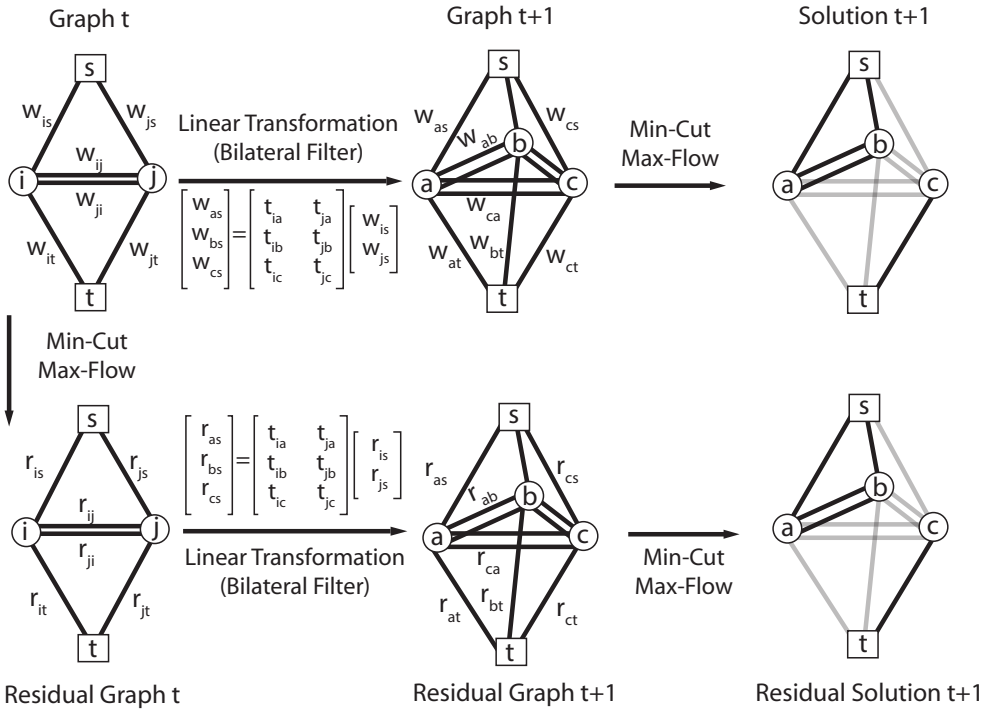


Figure 4.8: Example for Proposition 1. The solution at $t+1$ is obtained by applying a linear transformation to the graph at t and solving the graph at $t+1$. Note that the graph structure significantly changes at $t+1$ (a new node is inserted). On the other hand, residual graph at $t+1$ is obtained by applying the same linear transformation to the residual graph at t that is already computed while solving the graph at t. Moreover, the residual solution for $t+1$ is obtained by solving the residual graph at $t+1$. Proposition states that the solution at $t+1$ and the residual solution at $t+1$ are equal to each other. Indeed, computation time for the minimization of residual graph at $t+1$ is much less than the minimization of graph at $t+1$. (In order to make the illustration less dense $w_{ba}, w_{bs}, w_{ac}, w_{bc}$, $w_{cb}, r_{ba}, r_{bs}, r_{ac}, r_{bc}$ and r_{cb} are not shown in the figure. Linear transformations are also only displayed for node to source edges, while the rest can be computed trivially).

Proposition 1. *Binary labels obtained by minimizing the MRF energy, resulted after applying the bilateral filter on the energy function which is defined via the residual graph, is equivalent to the minimization of the MRF energy obtained via application of the bilateral filter on the original energy function.*

The proof of this proposition is deferred to Appendix. By using Proposition 1, instead of propagating MRF energy completely, we only propagate residual graphs throughout the frames. This dynamic propagation results in significant computation time reduction as explained in Section 4.2. Efficiency obtained via dynamic graph-cut can be crucial factor to increase the interaction quality for interactive multimedia applications. In [21], a similar dynamic extension of the graph-cut is used for mobile interactive image segmentation experiment. The result of the subjective evaluation shows that the increase in the efficiency leads to a significant improvement in the interaction quality specifically for the mobile applications [21]. On the other hand, the proposed

technique is the only dynamic min-cut/max-flow method which is applicable to the graphs whose structures are also time varying.

In summary, the proposed video segmentation method starts with an already known MRF energy for the first frame, solves the min-cut/max-flow problem; then, propagates the residual energy to the next frame via the proposed linear time algorithm. The proposed method continues to iterate through the frames.

4.2.4 Automatic Video Segmentation Extension

Although the proposed method is not applicable to automatic video segmentation problem directly, it can be used as a speed-up tool. Consider a video segmentation tool which process each frame or small chunk of frames, separately. Due to the high computation cost of motion information extraction, these methods are far from being efficient. Moreover, some of them use MRF energy based formulation after extracting a set of low level features from the video. For example, key-segments [47] algorithm uses motion and saliency information to produce high quality segmentation results using a high computation time (around 8 minutes per frame for standard definition consumer video).

The main idea is to use inefficient and high quality video segmentation tool for the initial frames or chunk of frames. Next, the available MRF energy function is used to initialize the proposed method and segment the rest of the video, efficiently. For videos with a long temporal length, the resulting time difference can be significant. The analysis of the computation time and the accuracy is performed in Section 4.3.

In addition to initialization via any motion aware-method and blind application of the proposed method, a feedback mechanism can also be added to the system in order to detect the scene changes and re-initialize the proposed method. It is possible to compute the confidence level of the min-cut/max-flow optimization easily. The most basic approach is to use the margin of the MRF solution as the threshold. Max-margin of any MRF energy can be computed efficiently as explained in [70]. Therefore, the resultant automatic video segmentation algorithm starts with an initialization by the video segmentation procedure, and continues with an inefficient and high accuracy motion aware segmentation method. Moreover, initialization is fed to the proposed MRF energy propagation to obtain segmentation for the next frames. Furthermore, after segmentation of each frame, a MRF margin is computed and compared with a specific threshold. If the margin is higher than the threshold, the algorithm is re-initialized via the motion aware segmentation method.

4.3 Experimental Results

Proposed MRF energy propagation method can be used either as an interactive video segmentation algorithm with an existing interactive image segmentation method or as a speed-up method for an automatic video segmentation algorithm. Hence, we have experimented the proposed method for both of these scenarios. In Section 4.3.1 and 4.3.2, we utilize recently proposed mobile and dynamic image segmentation tool [21] in

order to experiment with the interactive video segmentation scenario. Moreover, segmentation quality and computational efficiency results are compared against existing interactive video segmentation algorithms from the literature. On the other hand, in Section 4.3.4, we use the proposed method to speed-up a recently proposed automatic video segmentation tool [47]. We compare the segmentation quality and computational efficiency of the original method and propose a speed-up extension.

4.3.1 Analysis of Segmentation Quality for Interactive Video Segmentation

For the evaluation of the interactive video segmentation scenario, we utilize the "coloring-based" interactive image segmentation technique [21] to segment the initial frame. Then, we use the proposed method to propagate the initial segmentation results. Throughout the video, only the initial frame is interacted; no other user interaction is applied for the other frames. We compare the proposed method against two high-performing methods from the literature [7, 8]. The first one is a geodesic segmentation method [7]. Due the the fact that the permeability filter [15] actually approximates the geodesic distances, our proposed method approximates a bilateral filter using the geodesic distances as the filter weights. Therefore, our algorithm can be accepted as an extension of [7] with a larger support region, additional coherency term and smoother energy propagation. The other algorithm is a local classifier based segmentation method [8] which is included in Adobe After Effects CS5 [68] as the *roto-brush tool*. We use exactly the same interaction for the segmentation of initial frames for all three algorithms. Then, without any further interaction, we use these algorithms for the entire video sequence.

For the subjective comparison of the algorithms, we have used the dataset used in [36]. For the *IceSkater* sequence, the resultant segmentation and input video are presented in Figure 4.9. It can be observed in Figure 4.9, that *roto brush tool* [8] and the proposed method have similar performances. On the other hand, geodesic video matting tool [7] fails to propagate the segmentation results. The main reason for this performance drawback is its single pixel support region. Furthermore, the motion information is discarded in the geodesic video matting tool. For the first 50 frames, both the proposed method and *roto brush* [8] give near optimal segmentation results. Then, both algorithms segment part of the background as foreground erroneously. In addition to these, at frames 76 and 91, the region around legs of the ice skater are segmented as foreground by *roto brush*. The main reason for this observation is due to the shrinking bias of MRF energies [13]. However, the proposed method handles shrinking bias successfully due to its wide support region. Hence, one might conclude that the segmentation quality of the proposed method is slightly superior than that of the *roto brush* [8].

For the quantitative comparison of the algorithms, we have used *SegTrack* dataset [73]. The initial frame is segmented by using the same user interaction, and the algorithms are executed for the rest of the video. Then, pixel-based precision and recall values are computed for each frame via the following relations: $precision = \frac{tp}{tp+fn}$ and $recall = \frac{tp}{tp+fp}$; where tp is true positive, fp is false positive and fn is false negative. The resulting precision-recall curves for each video sequence is plotted in Figure 4.10.

Table4.1: Overall Computation Time per Frame

Geodesic Segmentation [7]	Roto Brush [8]	Proposed Method
2,7 sec	2,3 sec	1,3 sec

It can be observed from Figure 4.10 that in all videos, both the proposed method and *roto brush* [8] have superior performances against the *geodesic method* [7]. The main reason for this result is due to the incorporation of motion information in [8] and implicit usage of spatial information and wide support region (whole frame) for the proposed method. This result clearly indicates the benefit of the spatial information usage in video segmentation problem.

For *BirdFall* sequence, the proposed algorithms have better recall values compared to *roto brush* [8]. For the points having recall value higher than 0.6, the algorithms show almost the same performance. For the *Cheetah* sequence, *roto brush* and the proposed method have again similar performances. For the *Girl* sequence, *roto brush* [8] outperforms the proposed method. This behaviour is due to the high motion blur in the sequence. For the motion blur case, the color differences are low even at the object boundaries; therefore, the filter coefficients are not computed properly. An example of the motion blur in the dataset is shown in Figure 4.11. Hence, in *Girl* sequence, there is almost no edge information to be used in the frames due to the motion blur; therefore, the proposed method fails.

For *Monkey* sequence, the proposed method outperforms *roto brush* [8]. The motion occurs around the frame boundary; therefore, there is no enough local color information to be used for *roto brush* [8]. This characteristic of the video is also visualized in Figure 4.11. For *Penguin* sequence, utilized interactive image segmentation algorithm fails to segment the initial frame. Hence, the performance drawback is not due to the proposed energy propagation method. An example image is also shown in Figure 4.11. Finally, for the *Parachute* sequence, both *roto brush* [8] and the proposed method have precision and recall values higher than 0.9; both algorithms have similar precision values; however, the proposed method has better recall values.

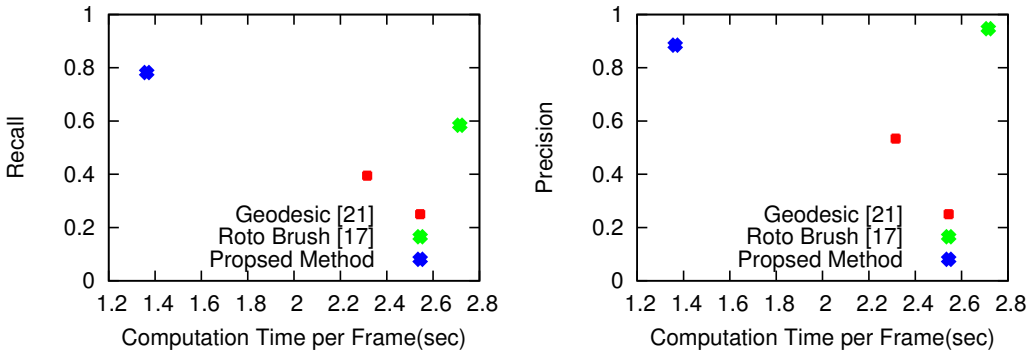
4.3.2 Analysis of Computational Efficiency for Interactive Video Segmentation

In order to compare the computational efficiencies of the algorithms, we have used the *SegTrack* [73] dataset. Each sequence in the dataset is rescaled to 640x480 resolution, and, segmentation time for each frame is computed for all sequences. The mean computation time for each frame is summarized in Table 4.1. These execution times are obtained using a standard desktop PC having processor of 2.4 Ghz and memory of 4Gb.

As it can be observed from the Table 4.1, the proposed method has the highest computational efficiency. It should also be noted that the significant part of the computation time for the proposed method is consumed by SLIC algorithm [1]. Convergence of SLIC algorithm takes approximately 0.9 second per frame. Hence, by the introduction of efficient over-segmentation algorithms, it is possible to reach a much better

computation time for the proposed method.

In order to visualize the performance vs time trade-off for all the algorithms, we have also performed performance vs time comparisons. For the *SegTrack* [73] dataset, we have computed mean precision and mean recall values, as well as the corresponding computation times. Precision vs computation time and recall vs computation time graphs are presented in Figure 4.12.



(a) Recall vs Computation Time (b) Precision vs Computation Time

Figure 4.12: Recall and Precision versus Computation Time (Top and left indicates a better performance). The results suggest that the proposed method reaches twice the efficiency with superior recall and comparable precision values. It should also be noted that, significant part of the computation time (0.9 second) is consumed by SLIC algorithm [1]. Hence, much better computation time is possible via faster over-segmentation algorithms.

As it can be observed from Figure 4.12.a, the proposed method has better recall and computation time, when it is compared against other algorithms. In Figure 4.12.b, *roto brush* [8] has slightly better precision values with a higher computational burden. One can conclude that the proposed method reaches segmentation quality of *roto brush* [8] with much higher computational efficiency.

4.3.3 Dynamic Bilateral Graph-Cut

In order to experiment the computational improvements obtained by dynamic computation of graph-cut in filtering scenario, we have performed MRF energy propagation via both conventional min-cut/max-flow method [13] and the proposed dynamic-graph cut. As explained in Section 4, dynamic graph-cut presented in [41] is not applicable to our scenario since the structure of the graph in the proposed method changes significantly due to the SLIC [1] over-segmentation method.

For the conventional min-cut/max-flow, we propagate MRF energy via the proposed method and apply the min-cut/max-flow method [13]. For the dynamic graph-cut, we use the proposed dynamic method. Only the graph-cut execution times for each frame are computed and plotted in Figure 4.13.

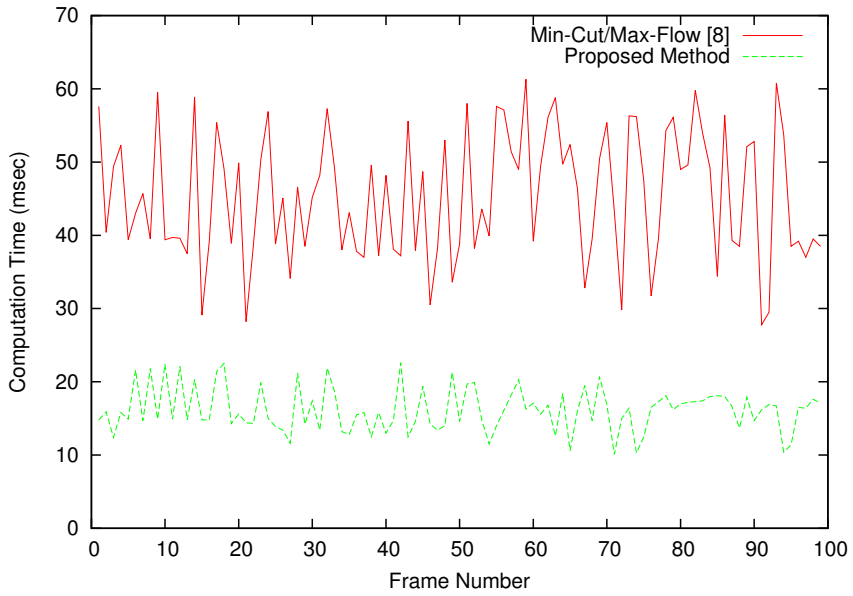


Figure 4.13: Computation times for dynamic graph-cut of each frame. It should be noted that only the minimization times are plotted. Moreover, the proposed improvements result in around 3 times efficiency increase.

As shown in Figure 4.13, the proposed improvements result in a significant reduction in the computation time for the min-cut/max-flow procedure.

4.3.4 Analysis of Automatic Video Segmentation

The proposed energy propagation tool can actually be used to speed-up any automatic video segmentation tool. We have experimented on a recently proposed automatic segmentation algorithm, Key-Segments [47]. In Key-Segments algorithm, graph-based 2D clustering of frames is performed to generate as many hypothesis as possible. Then, these hypothesizes are ranked by using a saliency-like measure and this measure is computed by using both shape, texture and motion information. Both extraction of optical flow vectors and computation of saliency-like measure are computationally expensive. In our setup, Key-Segments [47] method is applied only to initial k frames (k=3). Then, the resultant MRF energy used by key-segments [47] is used to initialize the proposed MRF energy propagation. In summary, computationally expensive video segmentation tool is used to initialize the proposed algorithm for automatic video object segmentation scenario for initial few frames; then efficient energy propagation tool is used to compute the video object segmentation for the rest of the video.

We have used *SegTrack* [73] dataset in order to analyse the performance of these speed-up procedures. As explained in [47], *Penguin* dataset is discarded due to the ambiguous ground truth. For the initial 3 frames, Key-Segments algorithm is performed and for the rest of the video, the proposed algorithm is used. The computation time and precision-recall values for each frame of each video are recorded and the resulting precision-recall curves are plotted in Figure 4.14. The average computation time for each frame is also computed and summarized in Table 4.2. For a fair

Table4.2: Computation Time per Frame (in MATLAB)

Key-Segments [47]	Speed-up Key-Segments (Proposed Method)
260.6 sec	4.0 sec

comparison, MATLAB source code distributed with [47] is used, and the proposed method is reimplemented in MATLAB. The execution times in Table 4.2 are obtained in MATLAB framework for both methods.

When the precision recall curves in Figure 4.14 are considered, except for the *Girl* sequence, the proposed speed-up method performs comparable against the original Key-Segments [47] method. In *Birdfall* sequence, the proposed method has better recall values than the original Key-Segments [47]. Moreover, in *Cheetah* and *Monkey* sequences, both algorithms perform very similar to each other. In *Parachute* sequence, both algorithms perform near optimal and the performance drawback of proposed method is mostly due to the superpixel errors caused by SLIC [1] algorithm. Finally, in *Girl* sequence the proposed algorithms perform much worse than the original Key-Segments [47]. The main reason of this performance degradation is due to the high amount of motion blur in the sequence. The edge information is used extensively by the permeability filter; therefore, lack of edges results in a serious performance degradation. Motion blur in the *Girl* sequence is visualized in Figure 4.11, and discussed in Section 4.3.1. In conclusion, except for the case of very high motion blur, the proposed speed-up results in similar segmentation quality, when compared to the original segmentation algorithm.

On the other hand, when computational complexity is considered, the proposed algorithm results in around 65 times speed-up, compared to the original algorithm. With its slight performance drawback and enormous speed-up ratio, the proposed speed-up method is a perfect candidate for any multimedia application which requires an efficient and automatic video object segmentation procedure.

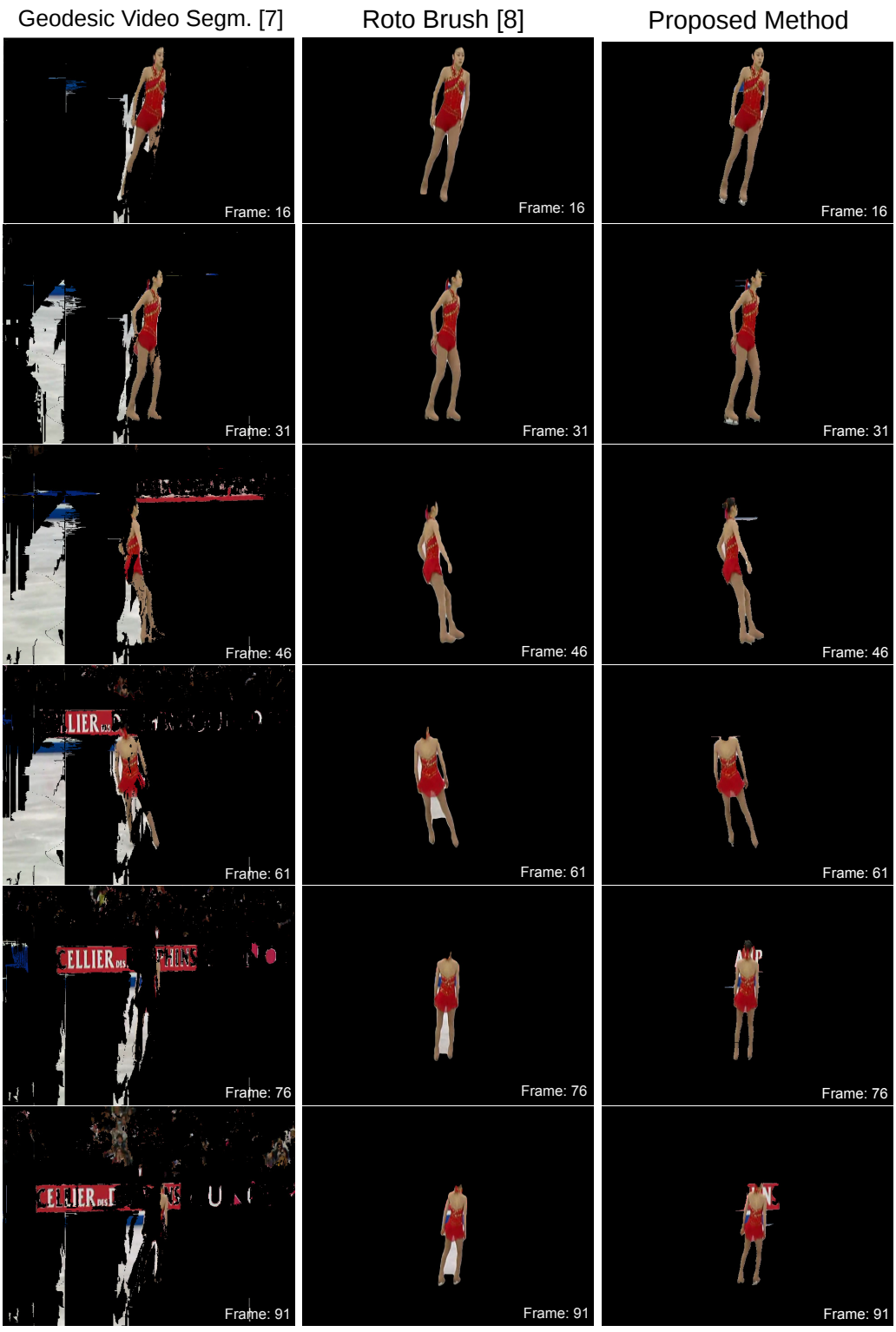
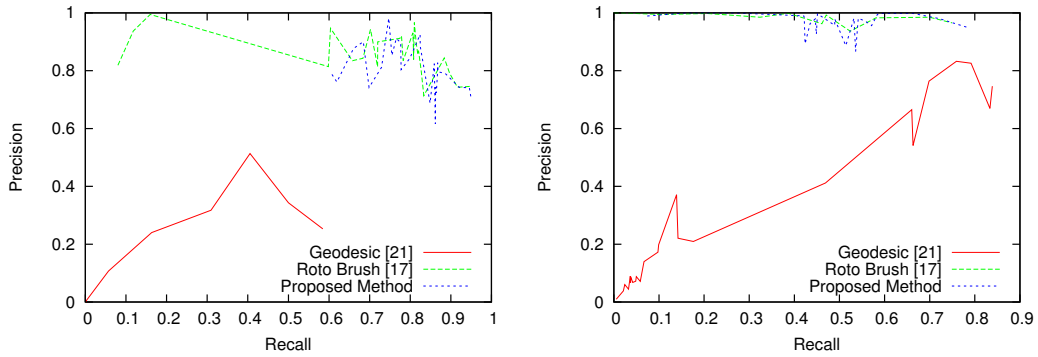
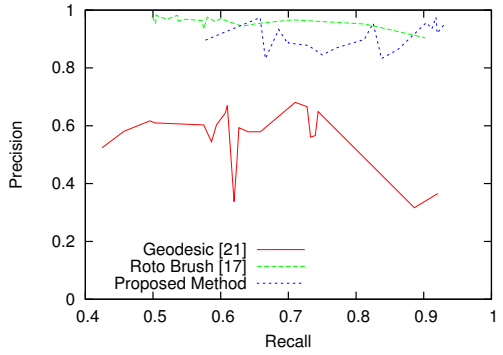


Figure 4.9: Visual comparison of interactive video segmentation algorithms for *IceSkater* sequence. The left column shows the result of *geodesic video segmentation* tool [7], the middle column shows the result of *roto brush* tool [8], and the right column shows the result of the proposed algorithm. Superior performance of the *roto brush* and proposed method against *geodesic video segmentation* is clear in all frames. Superior performance of the proposed method against *roto brush* can also be observed in Frame 61 and 76.

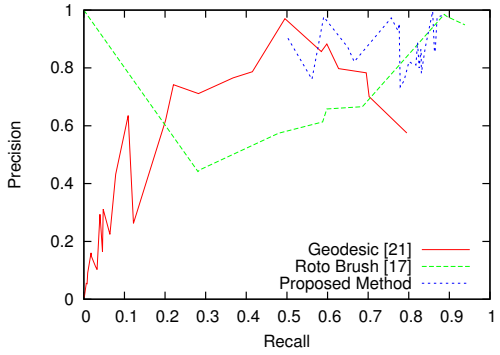


(a) Birdfall

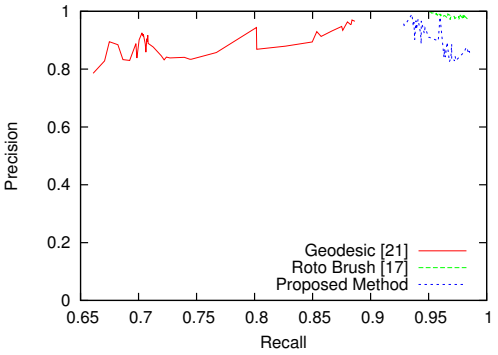
(b) Cheetah



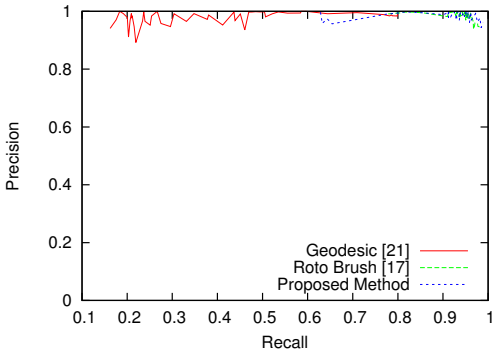
(c) Girl



(d) Monkey



(e) Penguin

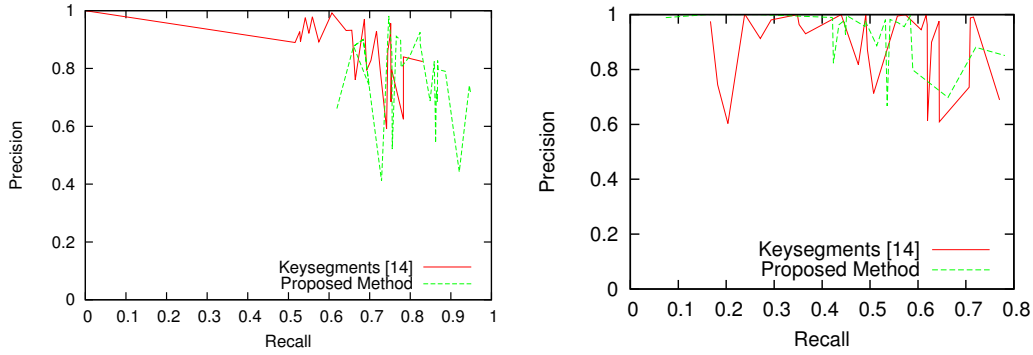


(f) Parachute

Figure 4.10: Precision-Recall curves for SegTrack[73] dataset (A curve near to top-right corner indicates better performance). Precision-recall curves suggest that for the interactive video segmentation problem, the proposed method shows either superior or comparable performances against the available methods in the literature. The main performance degradation in *Girl* sequence is due to the motion blur in the video.

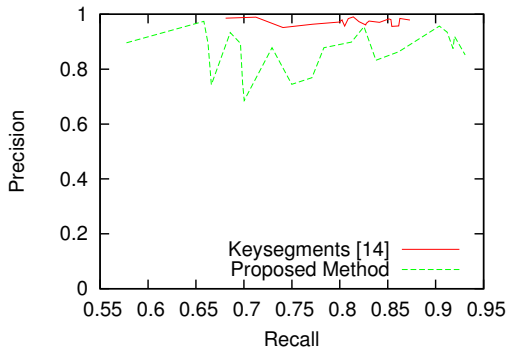
(a) Girl frame 16 (b) Monkey frame 19 (c) Penguin frame 1

Figure 4.11: Visualization of the motion blur and color characteristic in SegTrack [73] dataset that causes performance drawback in the proposed method and *roto brush*. Motion blur in the *Girl* sequence is clearly visible in (a). The black monkey in *Monkey* sequence is at the image boundary; therefore, local windows have no color information. Moreover, in *penguin* dataset, the ground truth is selected as a single penguin. Hence, interactive image segmentation algorithm fails due to the repetitive structure of the image.

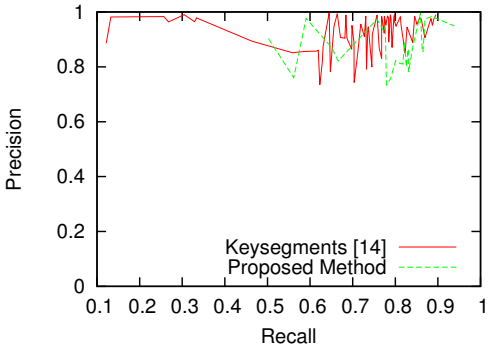


(a) Birdfall

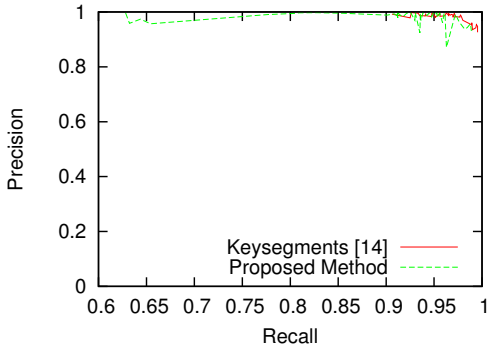
(b) Cheetah



(c) Girl



(d) Monkey



(e) Parachute

Figure 4.14: Precision-recall curves for automatic video segmentation on SegTrack [73] (A curve near to top-right corner indicates better performance). The proposed method does not cause significant performance drawback in any of the videos except *Girl* sequence. The reason for this performance drawback is the motion blur visualized in Figure 6.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Summary

Within the scope of this thesis, we consider the problem of interactive image and video segmentation on mobile touch-screen devices. We attack three main perspectives of the problem, namely, interaction quality, error robustness and computational efficiency.

After making an analogy by the gesture of coloring a color book, we have proposed a novel interaction method. Effectiveness of the proposed interaction method is subjectively studied for general mobile multimedia scenarios. Subjective evaluation suggests that proposed interaction method is easier and more user-friendly to use when compared to state-of-the-art interaction methods.

For the correction of user interaction errors, we have further proposed an error handling mechanism based on consistency via Gaussian mixture models. Effectiveness of the proposed method is also experimentally validated.

Dynamic graph-cut is an iterative and dynamic process that is introduced to decrease the computational complexity of the min-cut/max-flow optimization algorithm. Within the scope of this thesis, the dynamic graph-cut algorithm is further improved with the addition of a novel spatially dynamic graph-cut extension.

For the video extension, the interactive segmentation problem is reformulated as the propagation of the MRF energies that are obtained via an interactive image segmentation procedure on some video key-frames. A bilateral filtering based algorithm is proposed to propagate MRF energies efficiently throughout the frames. Efficiency of the proposed algorithm is further increased with a bi-exponential edge preserving smoother. A general case of minimization of linearly related energy functions on arbitrary and varying graphs is studied. Furthermore, a novel dynamic graph-cut algorithm is proposed and formally proven to have globally optimum result. The segmentation accuracy of the proposed propagation and the computational efficiency are also investigated both subjectively and objectively through simulations.

Finally, the proposed MRF propagation method is reformulated as a speed-up tool for automatic video segmentation methods. Significant time efficiency increase (around 65 times) is obtained without any loss in segmentation quality.

5.2 Conclusion and Future Directions

Interactive image/video segmentation problem has many aspects, such as interaction, error robustness and computational efficiency. Furthermore, these aspects are highly correlated. Therefore, it is not reasonable to study these aspects separately. The experiments suggested that time efficiency and error robustness have significantly positive effect on interaction quality for users. On the other hand, by proper selection of the interaction method, it is possible to achieve high error robustness. Coloring implicitly solves false negatives of the segmentation algorithm. Moreover, ordering of the interaction induced by coloring ease error correction. In addition to these, utilization of a common gesture of coloring a colorbook increases the user interaction quality significantly. Performed subjective evaluation indicates that the resultant user interaction method is easier, more user-friendly to use and leads more accurate segmentation results.

Images have many spatial redundancies and analysis of these redundancies results in significant time improvements. The proposed spatially dynamic graph-cut and over-segmentation are two basic solutions for the removal of these redundancies. By using the locality of the problem, we decreased the computation time of the min-cut/max-flow algorithm by half. In order to achieve this efficiency improvement, we show that it is possible to find a sub-graph of an image grid which gives approximately the same result by the global solution. Although the theoretical analysis only guarantees an approximate equivalence, the proposed algorithm gives the same result with the global graph for all test examples utilized during the experimental procedure.

Superior performance of the proposed error prevention method implies the importance of the spatial information included within user interaction. If global models are used for color and texture modelling, the coordinates of the user interaction is the only feature which carry spatial information. Hence, this information should be exploited, and error prevention should be handled prior to the min-cut/max-flow solution.

It is possible to formulate the video segmentation problem as the estimation of the MRF energies along the temporal dimension. A linear estimator formulated via reliable spatio-temporal distance metric results in high quality segmentation results. Furthermore, filtering via spatio-temporal distance metrics overcomes the limitations of the generative global models, since global models are not capable of utilizing spatial information. Spatial information is specifically important for video segmentation problem, since consecutive frames have strong spatial relations; hence, exploiting spatially aware distances, such as the geodesic distance have huge potential for the segmentation problem.

Conducted experiments suggest that it is possible to compensate the lack of motion information via reliable spatio-temporal distance metric. It is possible to compute the approximate spatio-temporal geodesic distance efficiently by using the bi-exponential/information permeability filters. Hence, overcoming the necessity of motion information usage via bi-exponential/information permeability filters reduces the computation time significantly. Furthermore, the experimental results suggest that approximation does not cause any loss in the segmentation quality.

Min-cut/max-flow algorithm is a linear and iterative algorithm. Therefore, efficiency of the min-cut/max-flow algorithm can be significantly increased by a linear problem definition. Hence, the proposed linear dynamic min-cut/max-flow algorithm results in significant computation time decrease. Moreover, the proposed algorithm is general enough to be applicable to any linear estimation problem.

Exploiting graph structure of the min-cut/max-flow problem might be a promising future research direction. Super-pixel based representation of images induce a graph structure different than grid graphs. Moreover, properties of the super-pixel graphs are not yet studied enough. One can study the graph theoretical properties of super-pixel graphs, such as degree of the nodes and maximum cliques. Furthermore, by the proper analysis of the graph structure, it might be possible to achieve better accuracy and time efficiency. Such an analysis could also be useful in order to design an over-segmentation algorithm which guarantees some graph-theoretical properties.

Min-cut/max-flow is just a one interpretation of the problem. There is also a linear programming based representation of the optimization problem, as well. By applying the recently proposed Lagrange duality [43], it might be possible to divide the min-cut/max-flow problems into a set of related optimization problems. One can also use such a formulation to propose a parallel algorithm for the solution of the energy minimization problem.

REFERENCES

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. SLIC Superpixels. Technical report, EPFL, 2010.

[2] Y. Aksoy, O. Sener, A. Alatan, and K. Ugur. Interactive 2D-3D image conversion for mobile devices. In *IEEE International Conference on Image Processing*. IEEE, 2012.

[3] A. Alatan, L. Onural, M. Wollborn, R. Mech, E. Tuncel, and T. Sikora. Image sequence analysis for emerging interactive multimedia services-the European COST 211 framework. *Circuits and Systems for Video Technology, IEEE Transactions on*, 8(7):802 –813, nov 1998.

[4] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 73–80. IEEE, 2010.

[5] F. Anscombe. The validity of comparative experiments. *Journal of the royal statistical society. series A (General)*, 111(3):181–211, 1948.

[6] V. Aurich and J. Weule. Non-linear Gaussian filters performing edge preserving diffusion. In G. Sagerer, S. Posch, and F. Kummert, editors, *DAGM-Symposium*, Informatik Aktuell, pages 538–545. Springer, 1995.

[7] X. Bai and G. Sapiro. A Geodesic Framework for Fast Interactive Image and Video Segmentation and Matting. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 2007.

[8] X. Bai, J. Wang, D. Simons, and G. Sapiro. Video snapcut: robust video object cutout using localized classifiers. In *ACM SIGGRAPH 2009 papers*, SIGGRAPH '09, pages 70:1–70:11, New York, NY, USA, 2009. ACM.

[9] A. Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bull. Calcutta Math. Soc*, 35(99-109):4, 1943.

[10] M. J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer vision and image understanding*, 63(1):75–104, 1996.

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

[12] Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. *Proceedings ICCV 2001*, 1(July):105–112, 2001.

[13] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE PAMI*, 26(9):1124–37, Sept. 2004.

[14] F. Calderero and F. Marques. Region merging techniques using information theory statistical measures. *IEEE Transactions on Image Processing*, 19(6):1567–86, June 2010.

[15] C. Cigla and A. A. Alatan. Information permeability for stereo matching. accepted for publication in *Signal Processing: Image Communication*,2013.

[16] P. Clifford. Markov random fields in statistics. *Disorder in physical systems*, pages 19–32, 1990.

[17] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619, 2002.

[18] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. *Combinatorial optimization*. John Wiley & Sons, Inc., New York, NY, USA, 1998.

[19] T. F. Cox and M. A. Cox. *Multidimensional scaling*, volume 88. CRC Press, 2001.

[20] A. Criminisi, T. Sharp, and A. Blake. Geos: Geodesic image segmentation. In *ECCV '08*, pages 99–112. Springer-Verlag, 2008.

[21] O. Şener, K. Uğur, and A. A. Alatan. Error-tolerant interactive image segmentation using dynamic and iterated graph-cuts. In *International Workshop on Interactive Multimedia for Mobila and Portable Devices*, 2012.

[22] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[23] E. A. Dinic. Algorithm for Solution of a Problem of Maximum Flow in a Network with Power Estimation. *Soviet Math Doklady*, 11:1277–1280, 1970.

[24] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. Wiley-interscience, 2012.

[25] F. Durand and J. Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '02, pages 257–266, New York, NY, USA, 2002. ACM.

[26] J. Edmonds and R. M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *J. ACM*, 19(2):248–264, Apr. 1972.

[27] H. Emrah Tasli and K. Ugur. Interactive 2D-3D image conversion method for mobile devices. In *3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON), 2011*, pages 1–4. IEEE, 2011.

[28] I. Endres and D. Hoiem. Category independent object proposals. In *Computer Vision–ECCV 2010*, pages 575–588. Springer, 2010.

[29] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59(2):167–181, Sept. 2004.

[30] L. R. Ford and D. R. Fulkerson. Maximal Flow through a Network. *Canadian Journal of Mathematics*, 8:399–404.

[31] L. R. Ford, Jr. and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

[32] S. Geman and D. Geman. *Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images*, pages 452–472. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[33] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, STOC '86, pages 136–146, New York, NY, USA, 1986. ACM.

[34] L. Grady, M.-P. Jolly, and A. Seitz. Segmentation from a box. In *IEEE International Conference on Computer Vision*, pages 367–374, 2011.

[35] L. Grady and E. L. Schwartz. Isoperimetric graph partitioning for image segmentation. *IEEE Trans. on Pat. Anal. and Mach. Int*, 28:469–475, 2006.

[36] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph based video segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[37] A. Hughes, A. N. University, and J. C. S. of Medical Research. *The Topography of Vision in Mammals of Contrasting Life Style: Comparative Optics and Retinal Organisation*. Handbook of sensory physiology. Australian National University, 1977.

[38] M. Izbicki. Gausian distributions form a monoid. http://izbicki.me/blog/gausian-distributions-are-monoids. Accessed: 2013-04-03.

[39] J. Kim and K. Grauman. Boundary preserving dense local regions. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, pages 1553–1560, Washington, DC, USA, 2011. IEEE Computer Society.

[40] S. Kirkpatrick, D. G. Jr., and M. P. Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[41] P. Kohli and P. H. S. Torr. Dynamic graph cuts for efficient inference in Markov random fields. *IEEE PAMI*, 29(12):2079–2088, 2007.

[42] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–59, Mar. 2004.

[43] N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[44] R. E. Korf. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial intelligence*, 27(1):97–109, 1985.

[45] J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50, 1956.

[46] S. Kwok and A. Constantinides. A fast recursive shortest spanning tree for image segmentation and edge detection. *Image Processing, IEEE Transactions on*, 6(2):328–332, 1997.

[47] Y. J. Lee, J. Kim, and K. Grauman. Key-segments for video object segmentation. In D. N. Metaxas, L. Quan, A. Sanfeliu, and L. J. V. Gool, editors, *ICCV*, pages 1995–2002. IEEE, 2011.

[48] Y. Li, J. Sun, and H.-Y. Shum. Video object cut and paste. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 595–600, New York, NY, USA, 2005. ACM.

[49] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum. Lazy snapping. *ACM Transactions on Graphics*, 23(3):303, Aug. 2004.

[50] D. Liu, Y. Xiong, L. G. Shapiro, and K. Pulli. Robust interactive image segmentation with automatic boundary refinement. In *IEEE Conference on Image Processing*, pages 225–228, 2010.

[51] H. Lombaert, Y. Sun, L. Grady, and C. Xu. A multilevel banded graph cuts method for fast image segmentation. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1 - Volume 01*, ICCV '05, pages 259–265, Washington, DC, USA, 2005. IEEE Computer Society.

[52] H. Lombaert, Y. Sun, L. Grady, and C. Xu. A multilevel banded graph cuts method for fast image segmentation. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 259–265. IEEE, 2005.

[53] B. Mohar. Isoperimetric numbers of graphs. *Journal of Combinatorial Theory, Series B*, 47(3):274–291, 1989.

[54] E. N. Mortensen and W. a. Barrett. Intelligent scissors for image composition. *SIGGRAPH '95*, 84602(801):191–198, 1995.

[55] J. Ning, L. Zhang, D. Zhang, and C. Wu. Interactive image segmentation by maximal similarity based region merging. *Pattern Recognition*, 43(2):445–456, Feb. 2010.

[56] J. B. Orlin. Max flows in o (nm) time, or better.

[57] S. Paris and F. Durand. A fast approximation of the bilateral filter using a signal processing approach. *Int. J. Comput. Vision*, 81(1):24–52, Jan. 2009.

[58] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:629–639, 1990.

[59] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[60] C. Rother, V. Kolmogorov, and A. Blake. Grabcut": Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23:309–314, 2004.

[61] C. Rother, S. Kumar, V. Kolmogorov, and A. Blake. Digital tapestry [automatic image synthesis]. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 589–596. IEEE, 2005.

[62] S. Roy. Stereo without epipolar lines: A maximum-flow formulation. *International Journal of Computer Vision*, 34(2-3):147–161, Aug. 1999.

[63] S. E. Schaeffer. Survey: Graph clustering. *Comput. Sci. Rev.*, 1(1):27–64, Aug. 2007.

[64] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 1997.

[65] J. Shi and J. Malik. Motion segmentation and tracking using normalized cuts. In *Computer Vision, 1998. Sixth International Conference on*, pages 1154–1160. IEEE, 1998.

[66] K. Subr, S. Paris, C. Soler, J. Kautz, et al. Accurate binary image selection from inaccurate user input. In *Computer Graphics Forum*, volume 32, 2013.

[67] ADOBE SYSTEMS. Adobe Photoshop 6.0 User Guide. 2009.

[68] ADOBE SYSTEMS. Creative Suite 5 User Guide. 2010.

[69] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(6):1068–1080, 2008.

[70] M. Szummer, P. Kohli, and D. Hoiem. Learning CRFs using graph cuts. In *Computer Vision–ECCV 2008*, pages 582–595. Springer, 2008.

[71] P. Thévenaz, D. Sage, and M. Unser. Bi-exponential edge-preserving smoother. *IEEE Transactions on Image Processing*, 21(9):3924–3936, September 2012.

[72] D. M. Topkis. Minimizing a submodular function on a lattice. *Operations Research*, 26(2):305–321, 1978.

[73] D. Tsai, M. Flagg, and J. M.Rehg. Motion coherent tracking with multi-label MRF optimization. *BMVC*, 2010.

[74] S. Vicente, C. Rother, and V. Kolmogorov. Object cosegmentation. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2217–2224. IEEE, 2011.

[75] J. Wang, M. Agrawala, and M. F. Cohen. Soft scissors: an interactive tool for realtime high quality matting. In *ACM Transactions on Graphics (TOG)*, volume 26, page 9. ACM, 2007.

[76] J. Wang, P. Bhat, R. A. Colburn, M. Agrawala, and M. F. Cohen. Interactive video cutout. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 585–594. ACM, 2005.

[77] L. Yatziv, A. Bartesaghi, and G. Sapiro. O(n) implementation of the fast marching algorithm. *Journal of Computational Physics*, 212:393–399, 2005.

[78] J. S. Yedidia, W. T. Freeman, Y. Weiss, et al. Generalized belief propagation. *Advances in Neural Information Processing Systems*, pages 689–695, 2001.

[79] C. T. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Trans. Comput.*, 20(1):68–86, Jan. 1971.

APPENDIX A

PROOF OF PROPOSITION 1

Proposition 1. *Binary labels obtained by minimizing the MRF energy, resulted after applying the bilateral filter on the energy function which is defined via the residual graph, is equivalent to the minimization of the MRF energy obtained via application of the bilateral filter on the original energy function.*

Proof. We prove this proposition by constructing sets of flows at time t. The flows are constructed by applying bilateral filter to the flows at time $t-1$. Moreover, we show that application of these flows converts the propagated energy function to a propagated residual energy.

We consider the flows in terminal weights and non-terminal weights, separately. Furthermore, our proof is based on edge flows; not source-sink flows. Since source-sink flows are combination of flows on edges, these flows can be obtained via a concatenation of edge flows.

For the flows on terminal edges, consider f_{iT}^{t-1} (T is either source or sink) as flows at time $t-1$:

$$f_{jT}^{t} = \frac{1}{\gamma_i^t} \sum_{v_i \in V^{t-1}} f_{iT}^{t-1} e^{-dis(z_i^{t-1}, z_j^t)}. \tag{A.1}$$

For the non-terminal flows, consider f_{ij}^{t-1} as flows at time $t-1$:

$$f_{ij}^{t} = \frac{1}{\gamma_{ij}^t} \sum_{v_k \in V^{t-1}} \sum_{v_l \in N(v_k)} f_{kl}^{t-1} e^{-dis(z_i^t, z_k^{t-1})} e^{-dis(z_j^t, z_l^{t-1})}. \tag{A.2}$$

As explained in Section 2, γ values are normalization constants. Moreover, for the clarification on the notation, superscript t represents time and sub-script T represents the terminals of the graph. It is shown that the application of any flow through the graph does not change the solution to the minimum cut problem [13]. Therefore, the solution after applying these flows does not change. The residual weights are obtained after applying these flows as:

$$r_{iT}^t = U^t(t, z_i^t) - f_{iT}^t \tag{A.3}$$

$$= \frac{1}{\gamma_i^t} \sum_{v_j^{t-1} \in V^{t-1}} U^{t-1}(t, z_j^{t-1}) e^{-dis(z_i^t, z_j^{t-1})} - \frac{1}{\gamma_i^t} \sum_{v_j \in V^{t-1}} f_{jT}^{t-1} e^{-dis(z_i^t, z_j^{t-1})} \tag{A.4}$$

$$= \frac{1}{\gamma_i^t} \sum_{v_j^{t-1} \in V^{t-1}} (U^{t-1}(t, z_j^{t-1}) - f_{jT}^{t-1}) e^{-dis(z_i^t, z_j^{t-1})} \tag{A.5}$$

$$= \frac{1}{\gamma_i^t} \sum_{v_j^{t-1} \in V^{t-1}} r_{jT}^{t-1} e^{-dis(z_i^t, z_j^{t-1})}. \tag{A.6}$$

A similar relation can also be obtained for the non-terminal weights as well:

$$r_{ij}^t = V^t(z_i^t, z_j^t) - f_{ij}^t \tag{A.7}$$

$$= \frac{1}{\gamma_{ij}^t} \sum_{v_k \in V} \sum_{v_l \in N(v_k)} e^{-d(z_i^t, z_k^{t-1})} e^{-d(z_j^t, z_l^{t-1})} V(z_k^{t-1}, z_l^{t-1}) \tag{A.8}$$

$$- \frac{1}{\gamma_{ij}^t} \sum_{v_k \in V^{t-1}} \sum_{v_l \in N(v_k)} f_{kl}^{t-1} e^{-dis(z_i^t, z_k^{t-1})} e^{-dis(z_j^t, z_l^{t-1})} \tag{A.9}$$

$$= \frac{1}{\gamma_{ij}^t} \sum_{v_k \in V} \sum_{v_l \in N(v_k)} e^{-d(z_i^t, z_k^{t-1})} e^{-d(z_j^t, z_l^{t-1})} r_{kl}^{t-1}. \tag{A.10}$$

This result corresponds to the application of the bilateral filter to the weights in the residual graph. In summary, we prove that propagating flows used in the min-cut/max-flow and recycling them for the next frame, corresponds to propagating residual graph via bilateral filters. The only question that might arise is the non-negativity of the weights. If the weights of the bilateral filter are positive, all edge weights in the propagated residual graph are also required to be non-negative. □