Graphical Methods for Image Compositing and Completion

by

Ahmed Fathi Mohammad Al-Kabbany

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the Ph.D. degree in
Electrical and Computer Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

This thesis is concerned with problems encountered in image-based rendering (IBR) systems. The significance of such systems is increasing as virtual reality as well as augmented reality are finding their way into many applications, from entertainment to military. Particularly, I propose methods that are based on graph theory to address the open problems in the literature of image and video compositing, and scene completion.

For a visually plausible compositing, it is first required to separate the object to be composed from the background it was initially captured against, a problem that is known as natural image matting. It aims, using some user interactions, to calculate a map that depicts how much a background color(s) contributes to the color of every other pixel in an image. My contributions to matting increase the accuracy of the map calculation as well as automate the whole process, by eliminating the need for user interactions. I propose several techniques for sampling user interactions which enhance the quality of the calculated maps. They rely on statistics of non-parametric color models as well as graph transduction and iterative graph cut techniques. The presented sampling strategies lead to state-of-the-art separation, and their efficiency was acknowledged by the standard benchmark in the literature. I have adopted the Gestalt laws of visual grouping to formulate a novel cost function to automate the generation of interactions that otherwise have to be provided manually. This frees the matting process from a critical limitation when used in rendering contexts.

Scene completion is another task that is often required in IBR systems. This document presents a novel image completion method that overcomes a few drawbacks in the literature. It adopts a binary optimization technique to construct an image summary, which is then shifted according to a map, calculated with combinatorial optimization, to complete the image. I also present the formulation with which the proposed method can be extended to complete scenes, rather than images, in a stereoscopically and temporally-consistent manner.

To my grandparents, my parents, my uncle and my aunt, my nieces and my nephews,

and my teacher, Eric Dubois,

Ink always falls short of expressing our deep love and sincere gratitude.

Acknowledgements

First and foremost, I would like to thank Canada; I will always be indebted to this country, my second home country. Represented by the Natural Sciences and Engineering Research Council of Canada, and the University of Ottawa, Canada provided me with thousands of dollars and unlimited resources of knowledge in order to become a better educated person, and it paid off.

I would like to thank my dear supervisor, Prof.Eric Dubois, for everything he did for me, for his support when I fell and failed, for his unconditioned financial support, for the several last-minute paper reviews, for bearing my unprofessionalism many times, for his knowledge and time, and for the doors he opened for me. In fact, any success I has achieved or will achieve throughout my life will bear the signature of Eric Dubois.

I would like to thank Ms.Laura Roach; her beautiful smile, sense of humour, and willingness to help had always been calming and inspiring. I also thank Ms.Michele Roy, Ms.Lily Murariu and Mrs.Sarah Duncan very much for their help.

I thank Mina Rafi Nazari and Roy Wang for always being there. Lending me their PC for unlimited amount of time when I needed it, their technical and life advices, and their heart healing chats amidst tough times, were just a small part of what they did for me.

I thank Alan Brunton, Houman Rastgar and Luis Gurrieri, for their help, for their advices, and for the good times.

I thank Prof.Claude D'Amour and Prof.Pierre Payeur.

My sincere thanks to the examiners of my thesis, Prof.Anthony Whitehead, Prof.James Clark, Prof.Jochen Lang, and Prof.Robert Laganiere, for their thorough review and insightful comments.

I thank my colleagues and lab-mates, Behrooz Abbaszadeh, Hamid Bazargani, Ismail Pasandi, Andres Montero and Olexa Bilaniuk, and I wish them all a thriving career.

I thank my friend Ammar Elhosainy, and my professors and colleagues at the Arab Academy for Science and Technology in Egypt.

Last but not least, I thank everyone who gave me a moment of happiness during the course of my Ph.D. studies.

Contents

List of Figures

List of Tables

Acronyms

BG Background. 43

BoSP Bag of Significant Patches. xi, 59

FG Foreground. 43

FVV Free Viewpoint Video. 1

GMM Gaussian Mixture Model. 47

GPU Graphical Processing Unit. 44

IBR Image-based Rendering. 1

KNN K-nearest Neighbours. 37

LDA Linear Discriminant Analysis. 36

LLE Locally-linear Embedding. 36

NVS Novel View Synthesis. 1

PCA Principal Component Analysis. 36

SSD Sum of Squared Differences. 61

Chapter 1

Introduction

From education to urban planning, and from entertainment to military training, 3D modelling is at the heart of many applications nowadays. The interest in creating a virtual world that would resemble a certain environment, a city's touristic attractions or even a body part, is on the rise, and it thus has been the scope of intensive and highly-paced research. For a system which enables a user to walk/fly through a virtual world, the modelling of that world acts like the backbone of such a system. Moving from one location to another, it is expected to render novel views to the user. The previous statement mentions one of the so-called 'mother problems' in computer vision and computer graphics, namely, Novel View Synthesis (NVS). Those problems encompass a multitude of other 'component problems'. These sub-problems surface at every stage in the pipeline of a system built to solve the mother problem.

Novel view synthesis is a well-studied area of research with several application-driven approaches. The research in this document is concerned with a particular approach, namely, Image-based Rendering (IBR). In IBR systems, the input is a set of images of a particular environment, captured by a set of calibrated cameras, and the desired output is non-physical (virtual) views. Those views represent what would have been seen by the user if he/she was present physically at that particular location in the real environment. This is how a high-quality immersion is realized. Numerous applications and platforms today rely on IBR systems, including 3DTV, Free Viewpoint Video (FVV) and NAVIRE systems[1] (Virtual Navigation in Image-based Representations of Real World Environments). An

[1] http://www.site.uottawa.ca/research/viva/projects/ibr/

illustration of what is a virtual view as compared to physical ones is depicted in Fig. 1.1.

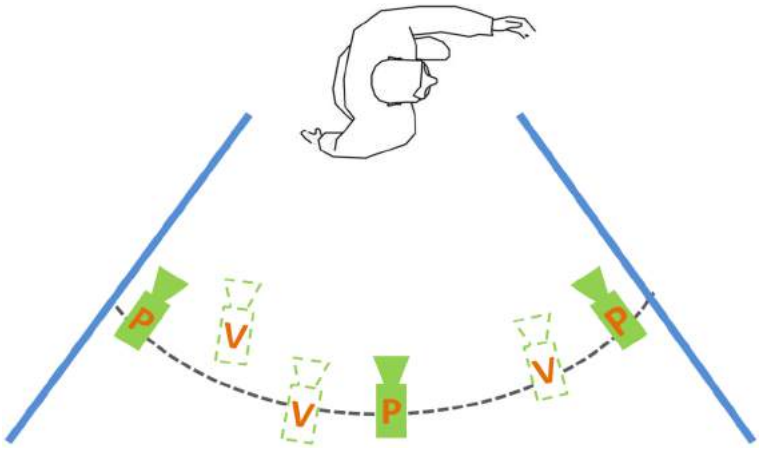


Figure 1.1: An illustration of the notion of physical vs. virtual views. Physical views are depicted as cameras with the letter 'P' while the virtual views are depicted as cameras with the letter 'V'.

A widely-used criterion to classify the IBR techniques is their dependency on the scene geometry to accomplish view synthesis. This is depicted by the chart in Fig. 1.2. Starting from the right-most side of the spectrum, we find techniques that explicitly use the scene geometry for rendering; texture mapping is a member of that family. Creating novel views is thus a trivial task, since we already have the 3D model of the environment. However, the acquisition of that model for an arbitrary real-world environment puts the generality of the whole approach in question. The other extreme of the spectrum is occupied by methods that do not rely on any geometric information during synthesis. Obviously, capturing the scene arrangement thus comes at the expense of an arduous image acquisition step along with significant retrieval/storage/communication burdens.

The research in this thesis is concerned with a few challenges faced by the techniques that exist in the middle of the chart in Fig. 1.2. Particularly, I am concerned with the robustness of view interpolation methods. Those methods are arguably the most prevalent approaches for IBR. The typical pipeline of a view interpolation system is depicted in Fig. 1.3. This figure is inspired by a similar figure in [1]. It starts by gathering information about the scene structure, through depth computation, before it proceeds to synthesizing novel views. Unless other equipment are available (like range sensors or Microsoft's Kinect sensor), this depth is usually measured passively by means of disparity estimation from

multi-view stereo. Establishing dense or sparse correspondences between a pair of views is used to calculate disparity maps, which are then employed to create intermediate views by means of 3D warping.

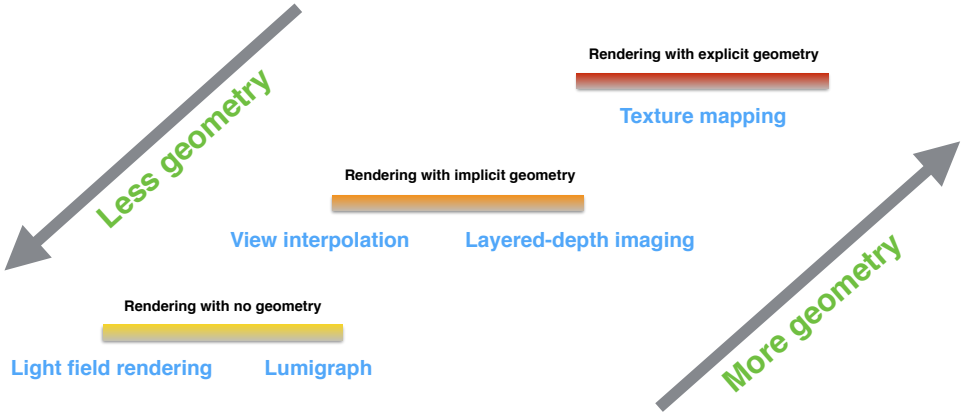


Figure 1.2: A schematic showing one possible classification of IBR systems according to their dependency on scene geometry.

Solving a collection of problems in computer vision is what holds the hope for such an IBR system to work robustly. These problems include but are not limited to:

1. Stereo matching or disparity estimation and a few associated open problems
   (a) Large disparities of thin and small structures
   (b) Edge bleeding of patch-based techniques (usually the fastest)
   (c) Computational efficiency of segment-based and pixel-based techniques
2. Scene segmentation
   (a) Soft segmentation: calculation of a partial opacity map of scene pixels
   (b) Hard segmentation: segregation of the scene into color-coherent (or generally feature-coherent) regions
   (c) Semantic segmentation: segregation of the scene into meaningful objects, e.g., a chair's backrest made of wooden and metal parts (with substantially different colors and textures) is expected to be warped (or more generally, manipulated) using the same parameters, as a single object.
3. Depth-guided scene completion

4. Structure-preserving warping

5. Point cloud matching and registration for 3D surface reconstruction

Each of these component sub-problems is a research area on its own, and usually represents the overlap between IBR and other vision-based applications.

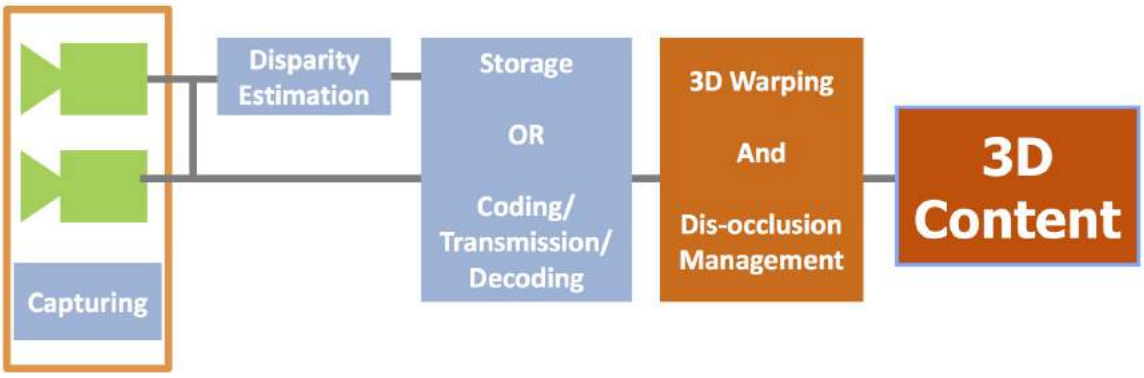


Figure 1.3: The view interpolation pipeline.

This thesis focuses on two of the problems mentioned above, namely, *soft segmentation* and *depth-guided scene completion*. Soft segmentation is more commonly known as natural image matting or alpha matting; it is also referred to as the image compositing problem, which is the eventual goal of the segmentation step – that is to be able to render novel composites of foreground/background objects. Depth-guided scene completion is sometimes referred to as depth-guided inpainting or dis-occlusion management. This thesis proposes novel strategies and frameworks to address a few challenges in the literature of the aforementioned problems. The performance of the techniques that represent the contributions of this research is compared with the performance of the state-of-the-art (SoA) techniques, and proves to be on-par, and superior in some aspects. The thesis concludes by providing a multitude of new horizons for future research directions.

1.1 Problem Statement

1.1.1 Background and Motivation of Natural Image Matting

Image compositing is a frequently used operation in image editing, TV broadcasting and film industry. To be able to make composites of high visual quality, by seamlessly

superimposing different foreground objects over various backgrounds, an opacity map discriminating (or soft-segmenting) the foreground object from the background should be acquired. This fuzzy map assigns a value, an alpha value, for every pixel in the image. According to a convex-linear model of image formation, the fractional alpha value that belongs to $[0, 1]$ determines how much the foreground/background contributes to the overall color of a certain pixel.

For constrained scenarios, where the foreground object is known a priori and is shot against a constant or an almost-constant backing color, with subtle illumination changes and limited camera poses, the well-established technique of blue screen matting [2], also known as chroma keying, suffices for extracting accurate alpha maps. One method for calculating an alpha map in such a constrained environment is discussed in section A.2. Natural image matting, however, involves a more general setup of the problem. In that setup, the foreground object is not known beforehand, and neither is the background. In addition, an alpha map (also called an alpha matte in the literature) indicating the contribution of the foreground/background at each pixel is required. Consequently, the linear convex model that expresses the color of each pixel in the composite as a combination of a fraction of a foreground color and a fraction of a background color defines an incompletely specified problem, with three equations and seven unknowns, namely, the un-composited foreground color vector, the background color vector and the alpha value. Hence, an infinite number of solutions exists unless more information is provided, to prune the solution space. Fig. 1.4 illustrates the difference between blue-screen matting (or chroma keying) and the general image compositing problem, and the output of a natural image matting algorithm.

The main goal of an IBR system is to provide the user with a high-quality navigation experience in real-world environments. Those environments are usually imposed rather than chosen. For instance, a tele-collaboration (or a video conferencing) system with virtual view rendering functionality is expected to be a plug-and-play platform, in any location and with any collaborators. Consequently, adopting chroma keying and similar approaches to acquire information about scene structure is not applicable. Rather, this scenario represents an instance of the general setup of the image compositing problem: For realistic composites in arbitrary environments, a partial opacity map of the image/scene

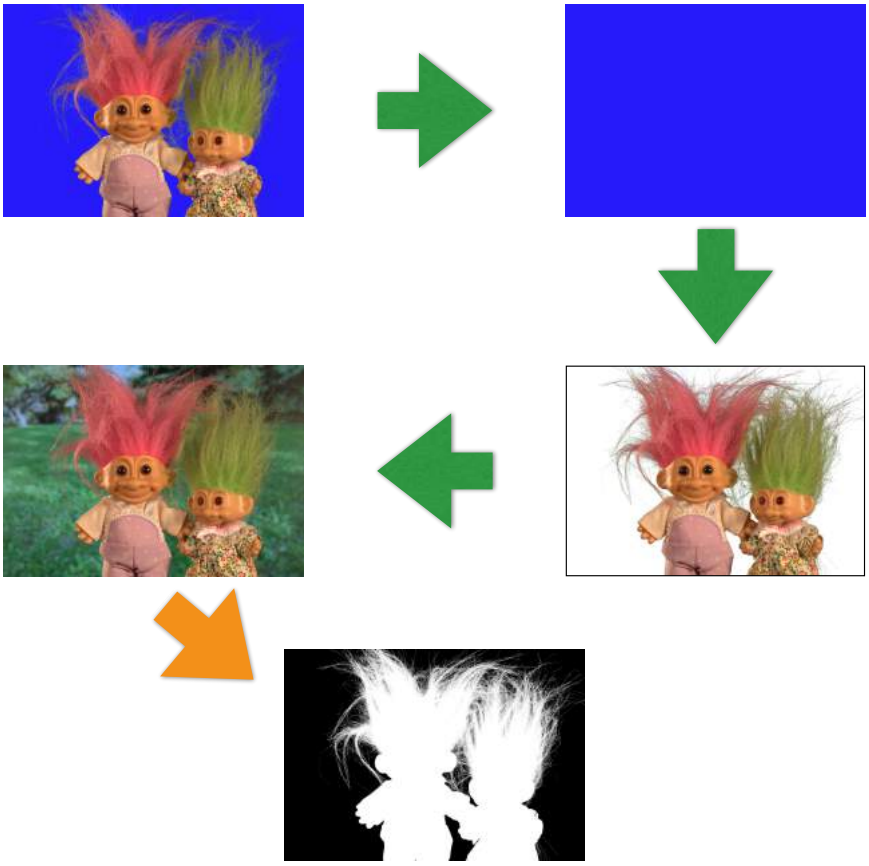


Figure 1.4: Starting from the upper left corner, this figure reads as follows: if a fuzzy object is shot against backgrounds whose colors are constant (1st image), using this prior information (2nd image), that object could be trivially separated from the background (3rd image). Then, it becomes possible to make composites by replacing the blue background with various (usually more pleasant) backgrounds (4th image). Natural matting problem though starts from that latter stage, where the object would have been shot against a complex background that is unknown beforehand, and the eventual goal is to compute an alpha map like the one pointed to by the orange arrow (5th image). The image is from the standard matting dataset [3].

is required.

Figure 1.5 shows an example [4][2] of a rendered view from a video conferencing system, where the novel view shows clear artifacts at the participant's head boundaries. Hence, it can be argued that the quality of the output would have been enhanced if alpha maps (or alpha mattes) of the scene were incorporated in the rendering pipeline. This is closely related to 'co-matting' [5], a recently-coined term for jointly calculating alpha maps from multiple views.

[2]http://www.robots.ox.ac.uk/%7Eojw/software. htm

(a) Input Views (b) Novel View

Figure 1.5: An example of a left and a right view, in a video conferencing (tele collaboration) context, and a synthesized view from both the input views. The black arrows point to regions of low plausibility in the synthesized views, which can be greatly enhanced using high-quality compositing. The figure highlights the role which can be played by matting in the novel view synthesis problem. The images are re-printed from [4], in accordance with the IEEE rules regulating the re-usage of copyrighted material by individuals working on theses. A printed copy of the permission grant is attached in section A.9. Copyright ©2008, IEEE.

1.1.2 Background and Motivation of Image Completion

The need to fill a hole, i.e., a region of missing pixels, in an image or a video could arise in a variety of contexts. Image retouching, object removal from an image or a video, or scene re-arrangement, where the positions of people and/or objects in a scene are required to be changed, are some examples for such contexts. A plethora of approaches do exist for using the information in the non-hole regions to fill a hole in an image, a video or a scene. One of those approaches is to diffuse the colors of the pixels surrounding the hole to the inside of it, which usually suffices for holes of small sizes. As the size of the hole gets larger, and the content of the non-hole regions becomes richer, with complex textures and/or structures, for example, simple diffusion is not efficient enough, and better formulations for the problem of hole filling become necessary. Section 1.3 will highlight the most recent approaches for image/video completion in the literature, and in Chapter 3, more details on particular methods that adopt those formulations will be covered. *The problem of image completion, by definition, has no ground-truth or reference, to which the completed/output image would be compared. Thus, the subjective evaluation of performance is a prevalent*

characteristic in the literature, as will also be seen in Chapter 3.

When the user of an IBR system chooses to be moved to a particular location in the virtual environment, 3D warping of the physically acquired scenes takes place. To visualize this process, one can imagine a rectangular lattice on which the physical scenes are defined; this lattice resembles a mesh. 3D warping is the manipulation of that mesh using the camera calibration parameters. In an ideal FVV system, for instance, the user position should not be constrained. Hence, for some locations, the image/scene function will not be defined on the warped mesh, resulting in large missing holes in the rendered scene. The problem here is two-fold. For the half-occluded scene regions, which are not visible in some physical views but visible in others, direct copy-pasting the pixel values from where they are seen will not guarantee high-quality renderings [6]. More importantly, for the fully-occluded scene regions, which are not visible in any of the acquired views, an algorithm should be engineered to hallucinate those regions in a visually-plausible way. Moreover, the hole-filling algorithm should employ the acquired disparity maps to complete the scene in a stereoscopically-consistent (or stereoscopically and temporally-consistent) manner. Figure 1.6 depicts an example of a dis-occlusion on the 'Ballet' sequence[3], a case where a warped virtual scene includes large areas of missing pixels. The literature review that will be presented in Chapter 3 will show that *at the heart of every depth-guided video/stereo-pair/scene completion method, there is a single-image completion algorithm that represents the core of the pipeline, and that heavily affects its performance and the quality of its output. This is why the efficiency of single-image completion methods is central to developing capable dis-occlusion management systems.*

Natural image matting and image completion are the scope of the research presented in this thesis. The next section highlights the wide variety of applications where my research can be of value.

1.2 Significance and applicability of this research

Since they communicate the transparency of objects in a scene, alpha maps find their way to every application that involves the creation of image composites. This includes interactive image editing, augmented reality, real-environment video games, cinematic

[3]http://research.microsoft.com/en-us/downloads/5e4675af-03f4-4b16-b3bc-a85c5bafb21d/

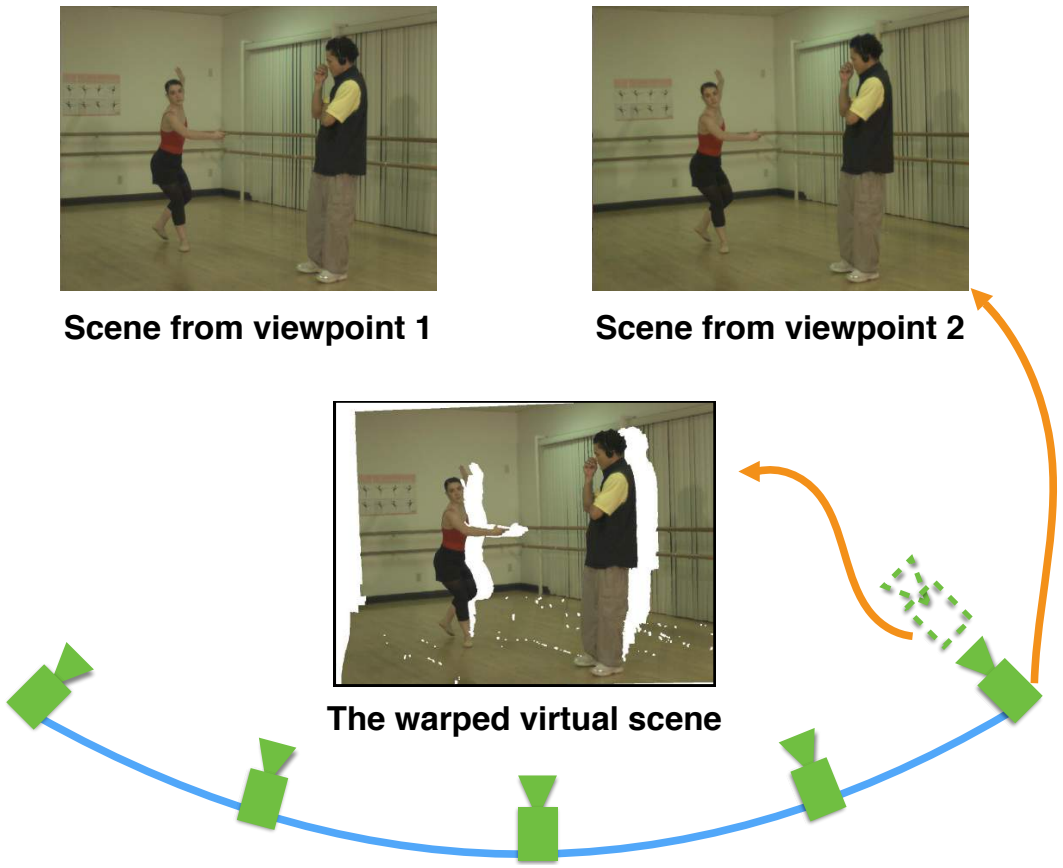


Figure 1.6: An illustration of the disocclusion problem on the 'Ballet' sequence.

reality, and many other applications that have been appearing in the media and technology industry everyday.

Current matting techniques still suffer from numerous limitations, even for interactive image editing, which is the least demanding matting application in the sense that the background is still static. New platforms like the Oculus Rift [4], for which the consumer version has been released since April 2015, have paved the way to dramatic new horizons. For the time being, it is just a head-mounted display, which offers the user a 3D gaming experience in a virtual world. The next envisaged milestone is the real-environment video games, where the user will be allowed to capture the environment where the game takes place as well as the characters to be overlaid on that environment. Superimposing arbitrary characters and objects on arbitrary backgrounds will definitely benefit from robust matting techniques. The more complex and the fuzzier the objects, the more accurate mattes will be needed to render the object efficiently in different contexts. Cinematic real-

[4]https://www.oculus.com/

ity is another revolution taking place in the field of augmented reality. Recently launched startups, like Magic Leap [5], aim at building platforms that can composite light-field sculptures on real-world environments, and the spectrum of applications is enormous. Crossing the gap of possible composites from limited pre-modelled objects to on-the-fly modelling of arbitrary structures of the user's choice, for educational purposes for example, will also benefit from natural image matting.

Image and scene completion, the second major scope of this research, is not confined to managing dis-occlusions in IBR systems. In fact, it is one of the key techniques in image and video editing, and it represents a pivotal stage in the pipeline of many tasks such as object removal and scene re-arrangement. In addition, my research involved the development of a new technique for image skimming, to be used as an integral part of the scene completion pipeline. Image skimming techniques are envisaged to benefit many areas of multimedia research. Statistics on skimmed videos can provide valuable insights about video content, which can be employed at a later stage for video indexing, video retrieval and re-identification, all of which are very active research areas.

1.3 Summary of the existing techniques and the open challenges

The literature of natural image matting can be classified into two groups or families of techniques, namely, propagation-based and sampling-based methods. Propagation-based or affinity-based matting techniques build and then use an affinity matrix, whose entries represent the similarity between neighbouring pixels, to propagate the partial opacity values (alpha values) from the known regions to the unknown regions in the trimap. A trimap is an image with the same size as the image for which the matte is required. In that trimap, some pixels are marked as definite foreground (using a white brush) with $\alpha = 1$, others are marked as definite background (using a black brush) with $\alpha = 0$, and the rest of the pixels with alpha values to be calculated are marked with a grey brush, hence the name. Pixels with unknown alpha values will be called the 'unknown pixels' throughout the rest of this document. Also, the abbreviations 'FG' and 'BG' will be used

[5] http://www.magicleap.com/

interchangeably with the words 'foreground' and 'background' respectively.

A critical limitation of many members in this family is the localness of the affinity matrices they construct. Contrary to sampling-based techniques, propagation-based matting does not rely on sifting particular foreground/background samples from the trimap to calculate an alpha value for every unknown pixel. However, some of them assume image continuity, for the propagation to be valid, which may or may not exist. Other group members adopt a non-local affinity matrix construction; however, they still suffer from a characteristic disadvantage of this family, namely, the considerable computational demands, memory and run-time. Recent algorithms which leverage preconditioned conjugate gradient methods, for solving large sparse linear systems, are still much slower than sampling-based methods. The latter have attained interactive rate matting; and are well-posed for parallelization which makes them eligible for GPU acceleration and real-time computation as well [7, 8].

Sampling-based matting approaches adopt a convex-linear compositing model. This model represents every unknown pixel in the trimap as a combination of a FG pixel and a BG pixel, sampled from the known regions in the trimap. Sampling-based matting aims at finding the FG/BG pair that best-describes the color of the unknown pixel under consideration. This category of matting techniques can be further divided into two major groups, namely, parametric and non-parametric methods. The former group builds color distribution models (GMM for example [9, 10]) for the known pixels in the trimap. These models are then used, at a later stage, to guide the sampling process. Non-parametric methods, however, try to robustly choose a sample pair without assuming any underlying color model. In general, highly-textured regions and overlapping color distributions are the most critical traps for all sampling-based methods (this becomes more severe with the parametric methods)[9, 11]. Many techniques have been suggested to address the challenges of sampling-based alpha matting, mainly through the adoption of:

1. A robust sampling strategy that hopefully can result in a comprehensive pool of foreground/background pairs for each unknown pixel.

2. A more distinctive feature space that can complement the classical color feature.

3. A distance function that can efficiently pick the best pair from the comprehensive

pool of foreground/background pairs.

The main competition in the sampling-based matting research area has been focused on proposing efficient trimap sampling strategies. A multitude of sample gathering procedures can be found in the literature [7, 8]. Some techniques propose to collect samples by ray shooting [7, 12, 13], others recommend to include all the pixels on the borders of the trimap [8], and others suggest an adaptive augmentation of the pairs pool size according to certain criteria. All of them, however, fail to deal efficiently with the variety of scenarios and challenges involved in sample gathering [11]. Thus, the problem of efficient trimap sampling has remained open for further development.

Since the main target of the research presented in this thesis is the enhancement of the quality of IBR systems, another pivotal, yet application-dependent problem, is the automatic acquisition of interactions that are normally provided by a user (trimaps, scribbles, etc...). Without this property, the user should somehow provide the system with a trimap, ideally, for every frame. This is a well-known problem for all video matting techniques [9, 14]. Although many of them count on the propagation of user scribbles provided on key frames, they still work within the limitations of the scribbles propagation quality, which can be threatened remarkably [14] in several situations including, for example, motion blur. The automatic acquisition of trimaps is thus a direction where a lot of room for improvements is still available.

Existing techniques for single image completion as well as 3D scene completion still cannot generalize well and fall short of dealing with the large pool of challenges involved in such operations. In the rest of this section, the limitations of the existing methods for single image completion and depth-guided completion will be discussed.

The literature of single image completion is quite extensive. Historically, image inpainting [15] targeted the problem of filling tiny holes and the removal of scribble-like attacks on images. The earliest family is the diffusion-based class of techniques [15, 16, 17, 18, 19]. These methods perform poorly with large hole sizes and cannot reproduce texture well, even though some of them can preserve image structures. The problem with these methods has an obvious resemblance with the shortcomings of propagation-based matting, since a diffusion-based filling technique also propagates colors to missing regions. They were followed by the exemplar-based family [20, 21, 22, 23]. The terms ‘an

exemplar' or 'a patch' will be used interchangeably throughout the rest of this document, and they will refer to a square window in an image, unless otherwise specified.

Exemplar-based techniques are the prevailing family in the literature to date, thanks to their better capability to reproduce textured regions plausibly, as compared to the diffusion-based methods. This has an impact on their generality in terms of filling small and large missing regions equally well. They can be further divided into matching-based techniques, graph-based techniques and hybrid techniques. Matching-based techniques fill the hole regions by explicitly comparing patches from the known (non-hole) regions in an image with patches from the hole region. Prior to Wexler's trilogy [23, 24, 25], patch matching and filling was done in a greedy manner, which can fail easily in case of, for example, multiple intersecting linear structures passing through the hole. Nevertheless, the global coherence measure proposed in [23] could fail to synthesize visually plausible completions due to its sensitivity to initialization and optimization strategy [26]. A detailed mathematical modelling for matching-based techniques, as well as techniques associated with other categories will be presented in Chapter 2, and further detailed in Chapter 3.

Graph-based methods [27, 28, 29] avoid, to a large extent, the demanding process of patch matching; rather, they seek a globally optimal solution. *The optimality here refers to the process of minimizing or maximizing an energy function for the completion.* Thus, graph-based methods do not suffer from sensitivity to initialization, given proper boundary conditions. However, this comes at the expense of the computational complexity of solving a graph labelling problem, whose optimization complexity is linear in the number of labels. The shift-map formulation [28, 26] is perhaps the most efficient graph-based framework, in terms of the quality of completions. Nevertheless, methods which adopt it still suffer from inherent limitations that hinder their ability to deal efficiently with the various parameters involved in the problem (large hole size, large image size, etc...). Particularly, their approaches of constructing the bag of significant patches (BoSP), which will be covered in Chapters 2 and 3, hamper the success of the hole filling procedure. Hybrid matching-graph-based methods use patch matching and graphical model-based optimization at consecutive stages along their pipelines [30].

The more information and cues one can acquire, the more plausible completions one can get. In situations where the task is to fill a 3D scene, not just a single image,

considering the acquired depth information side-by-side with color features is appropriate. The approaches adopted by the various depth-based inpainting techniques in the literature for utilizing depth information are fundamentally different. Some of these methods pre-process the warped depth maps so that they can be used to control the color/texture filling process at a later stage. Other approaches, simultaneously, fill the holes in the depth maps and the synthesized views. Major problems in the existing techniques include their reliance on pre-smoothing the depth maps [31] which results in noticeable geometric distortions in the filled views. Another problem is the direct copying of half-occluded regions from the view where they are seen. This leaves its imprint as poor visual quality near depth discontinuities [32] in the filled view. A third problem, which is not less severe, is the adoption of greedy filling algorithms and/or an objective function with multiple-local minima which fails to produce consistently faithful completions [6, 33]. Some approaches even proposed to manipulate the distances (calculated from depth maps) in a way that avoids the presence of missing pixels in the rendered views. This model obviously puts the viewer satisfaction at risk, since it basically renders a different view than what was intended by the viewer in a walkthrough or a generic virtual navigation. Hierarchical approaches [1] that propagate color values across scales of an image pyramid usually perform well with slim holes that result from narrow baseline stereo setups. However, real-life wide baselines require a higher level of scene structure understanding. It is then necessary to find an approach which:

1. Determines a comprehensive bag of significant patches, from the non-hole region, with which the 3D scene can be completed.

2. Formulates the task of depth-guided completion as a global optimization framework that yields consistently visually-plausible fillings. This should be done while maintaining the number of labels (and thus the computational complexity) as low as possible, without compromising the quality of completions.

3. Deals efficiently with the completion of the warped depth maps, within the limitations of disparity estimation methods, and eliminates the need for pre-smoothing and similar pre-processing operations.

1.4 Novelty and Contributions

[6] This research revolves around a central theme: Developing new techniques for image/video matting and completion, which can overcome limitations of the state-of-the-art (SoA) methods, while lending themselves well to IBR systems, aiming at better quality of novel view synthesis.

For natural image matting, the research presented in this thesis focuses on two problems, namely, the development of robust trimap sampling strategies, and the automatic acquisition of interactions (dense trimaps) that are normally provided by a user. I developed new sampling strategies which avoid several prevalent problems in the literature of sampling-based matting. All the proposed matting techniques were evaluated according to an online matting benchmark[7]. A detailed discussion on this benchmark is given in section A.1. The first proposed method [11] attained reasonable results according to the metrics of the benchmark (in addition to three top scores out of ninety-six on the 5^{th} of January 2014). The second proposed method [8] was placed eighth according to the MSE metric, over all the techniques in the literature, on the 23^{rd} of May 2015, and is among the top performing sampling-based matting techniques. The methods presented in this thesis were also compared objectively with the SoA sampling-based matting techniques on a training dataset of images, provided by the webpage of the benchmark. They scored the least MSE in 14 images out of 27 images.

I also developed a new algorithm that incorporates the Gestalt laws of visual grouping to generate dense trimaps in a fully-automatic fashion. The suggested framework is not the first in the literature to adopt the Gestalt laws for foreground/background segmentation. Nevertheless, my research formulates a novel objective function that quantifies those laws [34][9]. On one side, this research contributes to image matting, in the context of image/video editing, by making it more user-friendly through the elimination of the tedious process of supplying the interactions manually. On the other side, harnessing the power of soft segmentation in synthesizing plausible novel views is now more applicable, since the trimaps are not required to be provided by the user frame by frame, nor is the

[6] The content of the fourth, the fifth and the sixth Chapters of this thesis which overlaps with prior publications [34, 11, 35] is copyright SPIE 2014-2015

[7] http://alphamatting.com

[8] http://www.site.uottawa.ca/%7Eaalka046/TspsMatting/index.html

[9] http://www.site.uottawa.ca/%7Eaalka046/spie2015matting/index-spie15matting.html

user vulnerable to the trimap propagation techniques.

Another published contribution of this thesis is a graph-based framework for single image hole-filling using a near-globally optimal shift map. As mentioned earlier, the core of every scene (set of panoramas) completion pipeline is a single image completion technique. Thus, the proposed method represents a cornerstone, that is on-par with the SoA single image completion methods [23, 24, 25, 26], overcomes their inherent limitations and lends itself well for further developments that involve the incorporation of disparity maps for depth-guided scene completion. Along the way, a novel algorithm for image skimming was developed to construct a comprehensive bag of significant patches to fill the hole; this algorithm can find its way to a plethora of video editing, summarization and indexing applications.

1.5 Thesis Organization

The next Chapter will present the theoretical background of the problems addressed in the thesis as well as the techniques adopted in my research. Chapter 3 is dedicated to reviewing the literature of natural image matting and scene completion. For each problem, a detailed discussion of the recently-proposed techniques will be presented. This is expected to cover the problem formulation they used, their mathematical model, what they enhanced over their predecessors, in which aspects they are challenged and the prospective areas for improvement. This paves the way to the presentation of the contributions of the thesis which will take place in Chapter 4 through Chapter 7. Chapter 4 and Chapter 5 will include the contributions of the thesis to sampling-based matting. Chapter 6 is devoted for discussing the suggested framework for automatic trimap generation. Finally, Chapter 7 highlights the contribution to single image completion. The last Chapter reiterates the thesis structure and the contributions, and is concluded by foreseeable future research directions.

Chapter 2

Theoretical Background

The proposed research borrows theories and concepts, and benefits from various knowledge areas, namely, signal processing, graph theory, mathematical optimization, machine learning and visual perception. This chapter is devoted to laying the theoretical foundation upon which the thesis is based. It also serves to introduce briefly how the problems addressed by my research can benefit from the advances in the aforementioned knowledge areas. The chapter will start off by presenting the mathematical modelling of the problems addressed by the thesis. This will be followed by a discussion of the techniques used in my research. This discussion involves a few low-level and high-level image cues, and is comprised of a presentation of the relationship between the Gestalt psychology and some basic behaviours of human visual perception, in addition to a review of a recently proposed image decomposition technique. Afterwards, I will highlight some basic concepts in graph theory such as matrix representations of graphs in addition to graph cuts, a combinatorial optimization method that has gained remarkable attention from the computer vision research community since 2001. The Chapter will then be concluded by explaining a few concepts and techniques from manifold learning theory and compact representations of data. For the sake of clarity-of-presentation, I summarize the Chapter's map below.

1. Mathematical Modelling of Image Matting and Image Completion
2. Image Cues
    - The Gestalt laws of perceptual grouping

- Cartoon-texture image decomposition

3. Graph Theory and Mathematical Optimization

   - Matrix representations of graphs
   - Graph cuts for computer vision applications

4. Manifold Learning and Label Propagation

   - Dimensionality reduction using locally-linear embedding (LLE)
   - Learning by transduction

For the rest of this document, I deal with images that are sampled on a rectangular sampling structure. The sampling locations are given the symbol $\underline{x}_i$, where $i \in \{1, 2, \cdots, M\}$ for an image with M pixels. A pixel in an RGB color image will be given the symbol I_i which refers to the vector $[I_R(\underline{x}_i)\ I_G(\underline{x}_i)\ I_B(\underline{x}_i)]^T$. In the following sections, sampling locations could be denoted by horizontal and vertical indices (so a pixel could be given the symbol $I_{i,j}$); however, using a single index (I_i) is the default, unless otherwise stated.

2.1 Mathematical modelling of the problems addressed by the thesis

This section starts by modelling the matting problem formally, and highlighting briefly, in the light of this modelling, the areas of competition among the state-of-the-art techniques. Afterwards, I will proceed to modelling the depth-guided inpainting problem. In particular, different models of image inpainting, with varying levels of robustness and generality, will first be explained. Then, the specific model of interest, which is adopted in my research, will be discussed together with the 'how to' of incorporating the depth information in its pipeline. The open challenges in matting and hole filling will then conclude this section.

Image matting, in its own right, is a general framework of the image compositing problem. Nevertheless, it is a branch of a more general application, namely, image blending.

It is for this reason that alpha compositing is, classically, dubbed as alpha blending in the computer graphics literature. Alpha blending is modelled as a convex linear combination of the RGB values of a source image and the RGB values of a destination image. The general alpha blending model can thus be formulated as:

$$I_{comp} = A_{src} \times I_{src} + (1 - A_{src}) \times I_{dst}, \tag{2.1}$$

where I_{comp} is the composite image, I_{src} is the source image, I_{dst} is the destination image and A_{src} is the alpha channel indicating the translucency/transparency of every pixel in the source image. If $A_{src} = 0$ at a certain site (pixel), this indicates total transparency and thus the color of that pixel, in I_{comp}, is the color of its respective site in I_{dst}. On the other hand, if $A_{src} = 1$ at a certain site, this indicates complete opaqueness and the pixel's color in I_{comp} will be the color of the source image at that site. Fractional values in A_{src} results in mixing the colors of the source and destination images as given by Eqn. 2.1. In the usual compositing setup, those destination and source images contain, respectively, a particular required background and the figural objects to be superimposed on it. Hence, a more common form of Eqn. 2.1 is:

$$I = \alpha \times F + (1 - \alpha) \times B, \tag{2.2}$$

where I is the composite image, F stands for the un-composited foreground, B stands for background and α stands for the alpha map which is identical to A_{src}. For the clarity of presentation, it is worth mentioning at this point that in section A.2 of the appendix, which discusses the process of calculating ground truth alpha maps for matting datastes, the notation of [2] was used, where the un-composited foreground color was given the symbol C_o, while the backing color (or the background color) was given the symbol C_k. That notation though will not be used elsewhere throughout the thesis except in section A.2. If the alpha map has been already acquired, the color of a particular pixel I_i in I can be written in terms of the respective pixels in F and B as:

$$I_i = \alpha_i \times F_i + (1 - \alpha_i) \times B_i, \tag{2.3}$$

where α_i is the partial opacity of the pixel I_i. However, in the natural image matting problem, the alpha map is required, not given. Accordingly, every pixel's alpha value needs to be estimated first. Since only one equation is available, the problem is incompletely specified (three equations and seven unknowns for an RGB feature), and thus an infinite number of solutions exist unless more information is provided. Basically, all parameters in Eqn. 2.3 have to be given. This is usually done using user guidance that takes the form of either sparse scribbles or dense trimaps. Given a certain pair of foreground/background pixels (F_u, B_v), known from the given user interactions (sparse scribbles or dense trimap), we estimate the alpha value at pixel I_i as:

$$\hat{\alpha_i} = \frac{(I_i - B_v) \cdot (F_u - B_v)}{\|F_u - B_v\|^2}. \tag{2.4}$$

Handling the case where $F_u = B_v$ will be covered in sub-section 3.1.7. The suitability of the chosen pair, and thus the correctness of the estimated alpha value, is then judged by calculating the following color cost function (sometimes dubbed color energy or fitting error or chromatic distortion):

$$\xi_{color} = \|I_i - (\hat{\alpha_i} F_u + (1 - \hat{\alpha_i}) B_v)\|, \tag{2.5}$$

which can be illustrated using the color-line model shown in Fig. 2.1. The figure is adapted from previous work in the literature [36]. This figure shows that a certain foreground/background pair can better represent a pixel's color if it is nearer to the straight line joining that pair in a particular color coordinate system. The problem of choosing good samples can also be formulated in a Bayesian framework [37]. Equation 2.5 is embedded in the likelihood term, and the priors are calculated by fitting parametric models to the known regions in the trimap. Gaussian mixture models (GMM) are a prominent example. The overall MAP problem is given by:

$$\underset{\alpha_i, F_u, B_v}{\operatorname{argmax}} P(\alpha_i, F_u, B_v / I_i) \tag{2.6}$$

$$\underset{\alpha_i, F_u, B_v}{\operatorname{argmax}} \underbrace{L(I_i/\alpha_i, F_u, B_v)}_{Likelihood} + \underbrace{L(F_u) + L(B_v)}_{Priors}; \tag{2.7}$$

where $L(\cdot) = \log P(\cdot)$ and the likelihood term is the chromatic distortion given in Eqn. 2.5.

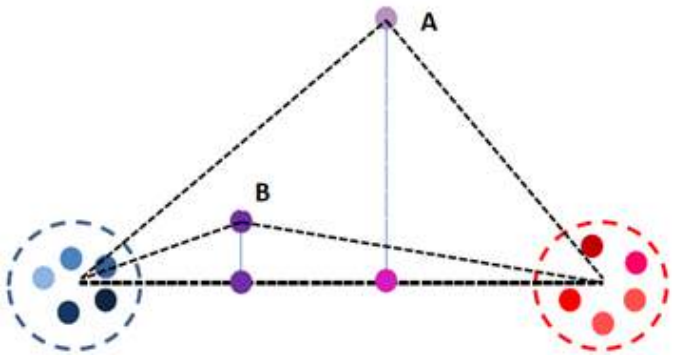


Figure 2.1: A couple of foreground and background pixel clusters are more suitable to represent the pixel B than pixel A since the former is nearer to the line joining the two clusters in the color space. The figure is adapted from previous work in the literature [36].

In Eqn. 2.4, just one pair of foreground/background pixels, namely (F_u, B_v) was mentioned. This brings to the surface one of the challenges which arise while solving the matting problem, that is: How many pairs should be considered? Taking into account all the given pairs in the known (white and black) regions of the trimap is computationally prohibitive, and impacts the quality of the calculated maps negatively as well, as will be seen in Chapter 3 and Chapter 4. That is why researchers have kept proposing new sampling strategies, according to which, they can limit the considered pairs for every pixel, by adopting certain criteria. This is the fundamental difference between the various sampling-based matting techniques.

Another main competition area, that is pertinent to the problem modelling, is the adopted cost functions, like the chromatic distortion in Eqn. 2.5. Many forms (or kernels) of chromatic distortion functions have been used to date, and each has shown particular merits and drawbacks. In addition, some algorithms have adopted cost functions with a variety of terms that weigh other aspects beside the color feature while ranking a certain FG/BG pair. Detailed differences between sampling-based matting techniques will be given in the literature review in Chapter 3. The family of matting techniques which do not rely on pair sampling, but on alpha value propagation from known regions to unknown regions in the trimap (propagation-based matting techniques) are generally out of the scope of my research so far, mainly due to their computational demands.

Compared to the matting problem, the mathematical models used for image completion are more diverse and arguably dependent on the problem formulation. If the filling

process is formulated as the diffusion of pixels values from known regions to missing regions in the image, then the problem boils down to solving a system of PDEs or similar systems of equations [15]. This technique deals efficiently with, for example, restoration of old images and image retouching applications in which the size of missing regions is quite small. Algorithms in this family are iterative, and they define the restored image at certain iteration I^{n+1} in terms of the image at the previous iteration I^n as:

$$I^{n+1}(i,j) = I^n(i,j) + \Delta t\, I_t^n(i,j) \qquad \forall (i,j) \in \Omega, \tag{2.8}$$

where (i,j) represents the pixel coordinates, Δt is the improvement rate and $I_t^n(i,j)$ is the update for the image $I^n(i,j)$. The symbols Ω, $\partial\Omega$ and $\bar{\Omega}$ are the hole region, the hole region's boundary (or the fill front) and the known region respectively. Fig. 2.2(a) depicts the fundamental symbols and terminology in the hole-filling literature. Diffusion-based algorithms rely on the propagation of the lines which hit $\partial\Omega$. This requires determining the information to be propagated and the direction of propagation. To guarantee a smooth propagation of information across $\partial\Omega$, smoothness estimators are embedded in $I_t^n(i,j)$, and the direction of propagation $\vec{N}$ is the direction of the 'isophote'. This term is defined as the normal to the direction of the largest spatial change. Based on the iterative nature of the algorithm, the completed image is often referred to as I^∞. As will be highlighted in the literature review; in the context of depth-guided hole-filling, the usage of this family of techniques has been limited to fill holes in the disparity maps, where most of the information is expected to be piece-wise constant, linear and/or smooth. This is quite different from the case of texture-rich RGB images.

Exemplar-based filling techniques are named after their main motif: filling Ω with exemplars from $\bar{\Omega}$, shown in Fig. 2.2(b). The term 'exemplar' is often used in lieu of 'patch' which refers to a rectangular area in an image. The majority of these algorithms are greedy, best-first search algorithms. This was alleviated by adopting a certain filling order dictated by a priority function $M(\mathcal{I}_s, \mathcal{C}_o)$: $\mathbb{R}^2 \rightarrow \mathbb{R}$. Starting from the fill front $\partial\Omega$, the algorithm iteratively 'peels' the hole commencing from the location with the highest priority. That location is defined as the incomplete patch with the maximum product of isophote strength $\mathcal{I}_s$ and count of known (non-hole) pixels $\mathcal{C}_o$ [21]. An example of a filled patch is shown in Fig. 2.2(b). Although not optimal, this strategy had been used

by many recently proposed depth-guided filling techniques [38].

Wexler **et al.** [23, 24] suggested an optimized framework for the patch-matching process. This was realized using a coherence measure given by:

$$d_{coherence} = \sum_{P \in \Omega} \min_{Q \in \bar{\Omega}} \|P - Q\|^2, \tag{2.9}$$

where P is a patch in the hole region, Q is a patch in the non-hole region and $\|.\|^2$ is a distance function. This measure penalizes completions which involve patches whose best matches are not similar to them. It is usually adopted with a multi-scale strategy, by building an image pyramid, and then optimized in an expectation-maximization (EM) fashion. This technique has been recently extended to depth-guided inpainting [6] by dealing with the input views as $RGBD$ (D stands for depth or disparity) images, and augmenting Eqn. 2.9 with a term that compares disparities of P and Q in addition to their RGB patterns.

Hole-filling can also be formulated as a quest for an optimal shift-map [28, 39]. A justification for this notion is the observation that Ω can be 'painted' by shifting $\bar{\Omega}$ in a direction which maximizes (thus it is dubbed an optimal direction) the coherency of the completed image I^{∞}. An example of this process is shown in Fig. 2.2(c). With such formulation, a globally-optimal shift-map can be calculated using a pair-wise energy of the form

$$E = \underbrace{\sum_{s_i \in \Omega} E_d(s_i, l_{m_1})}_{Data\ Term} + \underbrace{\sum_{(s_i, s_j)\, \forall\, s_i \in \Omega, s_j \in \Omega} E_s(s_i, s_j, l_{m_1}, l_{m_2})}_{Smoothness\ Term}, \tag{2.10}$$

where s_i and s_j are two 4-connected sites or pixels in the graph of Ω, while l_{m_1} and l_{m_2} are two labels from the pool of possible M shifts. The data term in such energies is usually a constraint of the form

$$E_d = \begin{cases} 0 & \text{if } l_{m_1} \text{ is a valid shift for } s_i \\ \infty & \text{otherwise,} \end{cases} \tag{2.11}$$

where the validity of a certain shift for a particular hole site is determined using some criteria. These criteria are one fundamental difference among the various proposed energies in the literature, and in my proposed technique as well. While the data term dictates a

cost to every possible shift-pixel (site-label) assignment, the smoothness term is designed to penalize the shift-pixels assignments that lead to incoherent seams. The coherency in this context is sometimes achieved by minimizing the color difference between neighbouring pixels in Ω, or the color+gradient difference between them. More details on different energies will be discussed in Chapter 3. Although it has has been shown to be effective, in terms of the visual quality of the computed image completions, the optimal shift-map approach has not been used for depth-guided scene completion before.

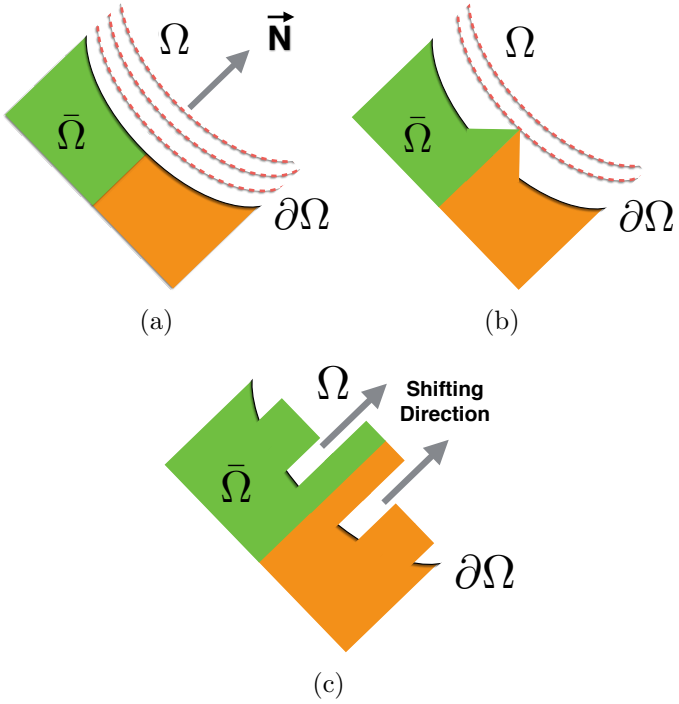


Figure 2.2: Declaration of hole-filling symbols and basic terminology. The symbols Ω, $\partial\Omega$ and $\bar{\Omega}$ are the hole region, the hole region's boundary (or the fill front) and the known region respectively. (a) Hole filling can be formulated as a diffusion of pixel color values from the source region to the hole region in the image, starting from the fill front. (b) It can also be achieved using exemplar-based inpainting where patches from the source region are used to fill the hole. (c) Shifting the source region in a certain direction, indicated by an optimal shift map, can also be used to fill the hole region.

2.2 The Gestalt laws of perceptual grouping

The Gestalt laws of grouping are a set of principles in a theory of mind named as Gestalt psychology. As a theory which tries to decode and explain the process of perception and response to external stimuli, it is not limited to visual perception; however, this latter part

is clearly what is focused on during this discussion. Among the fundamental pillars of this theory is an argument stating that humans tend to perceive a scene in a simplistic manner. This pursuit for simplicity shows up as an inclination to interpret what is seen in terms of regular patterns, symmetry, organized objects, etc. Since the 1920s, scholars embracing the theory of Gestalt psychology have been refining and decomposing its fundamental concepts into a set of rules. Gestalt laws are a sub-group of these rules which account, to a large part, for the nature of foreground-background assignment and scene organization. Along the same lines of the innate pursuit for simplistic scene interpretations is the holistic perception of objects, the tendency to perceive a whole object in lieu of the sum of its parts which are organized and structured according to those set of laws.

The Gestalt principles of grouping are used extensively in art and design. Other uses that are more pertinent to the subject of this thesis include the design of graphical user interfaces and the development of human computer interaction platforms. Recently, those laws have found their way to an increasing number of computer vision applications. These applications include saliency modelling [40], figure-ground segmentation [41], and natural image/video matting [34], [14]. Figure 2.3 depicts a few principles such as:

The law of proximity: This law expresses the tendency of a mind to perceive objects that are close to each other as one entity, and vice versa.

The law of symmetry: This law claims that humans are inclined to interpret symmetric entities as a single object formed around a centre point.

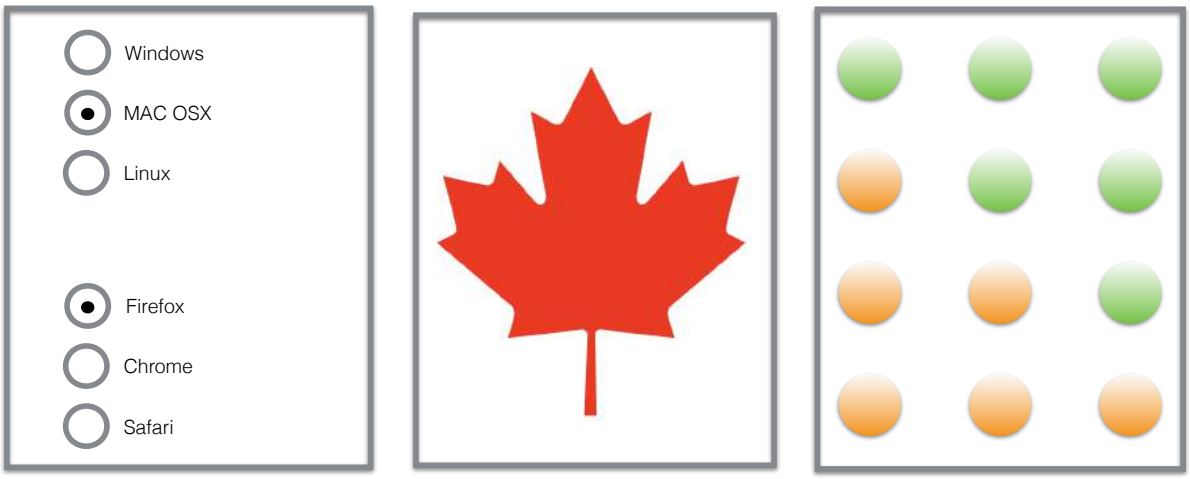
The law of enclosure: This law states that regions that are enclosed and surrounded are more likely to be perceived as figural objects.

The law of similarity: Perhaps this is the least controversial law among them all. It states that we tend to group parts that are similar in appearance as one object, and vice versa.

To keep the presentation concise and to-the-point, I haven't reported other laws that are not adopted in my research. I refer to [42] and [43] for a more detailed discussion of this subject.

These cues do correlate, and their quantification in a certain scene/image is not necessarily independent or a per-law process. In fact, as will be pointed out in Chapter 5, a metric or a distance function designed to evaluate the proximity score of an image can

give information on its enclosure score as well. Another example for this relates to the law of symmetry and the law of good continuation. The Gestalt laws of grouping have been used in this thesis to formulate a new cost function for generating dense trimaps in a fully-automatic fashion. This helps in closing the gap between natural image matting and the applications that do not tolerate user interactions, like novel view synthesis.



(a) Proximity (b) Symmetry and Enclosure (c) Similarity

Figure 2.3: Some examples for the Gestalt laws of grouping

Since these principles highlight abstract information about scene organization, rather than pixel-based values or features, they are affiliated with a diverse pool of high-level image cues in the literature. The next section discusses a low-level image feature which has been exploited in the research on image matting presented in Chapter 4, and which is obtained using a relatively-recent technique for signal decomposition, namely, the cartoon-texture decomposition.

2.3 Cartoon-texture image decomposition

Signal decomposition is a cornerstone in all signal-processing-based applications and technologies. It aims at decomposing a signal into its main constituents or 'building blocks', a process which has proved to be of vital importance for a diversity of applications including signal compression and feature extraction. Cartoon-texture decomposition (sometimes dubbed structure-texture and geometry-texture decomposition) is an additive decomposition model of the form $I = c + t$ which aims at analyzing the signal into a piece-wise

smooth (cartoon) component and an oscillatory (textural) component. For this particular type of decomposition applied to 2D images, the classical linear decomposition techniques, realized using a high-low-pass filter pair, have two critical disadvantages [44]. First, this process takes away contours and edges from the cartoon part to the the textural part, although they are not considered textured areas; at the same time it keeps blurred textures (out-of-focus areas for instance) with the low-frequency part, although they are not considered a cartoon part. The work in [45] was the first to propose a variational framework to address this shortcoming. They formulated an energy minimization problem of the form

$$(c_{opt}, t_{opt}) = \underset{(c,t)\in X_1\times X_2}{\operatorname{arginf}} \{F_1(c) + \lambda F_2(t) : I = c + t\}, \tag{2.12}$$

which picks the cartoon part from the space of functions with bounded variation, and the textural part from the space of oscillatory distributions [46]. Equation 2.12 reads as follows: for the functionals F_1 and F_2, single out the (c, t) pair which minimizes $F_1(c) + \lambda F_2(t)$, such that the summation of this (c, t) pair yields the image to be decomposed; λ is a tuning parameter, and X_1, X_2 are the spaces of functions for which $F_1(c)$ and $F_2(t)$ are exclusively bounded, i.e., $F_1(c) < \infty$ and $F_2(t) < \infty$, if and only if $(c, t) \in X_1 \times X_2$. Some details on the choice of X_1, X_2, F_1 and F_2 are deliberately skipped to keep the presentation concise.

Later, other techniques were proposed to obtain a cartoon-texture separation of signals [47], [48]; an extensive comparison of them was presented in [46]. However, I am particularly interested in the efficient approach of [46] for three main reasons: its low time complexity, publicly available code and few tuning parameters (actually only one, which determines the texture scale). This approach gives an approximate solution for the original variational problem, and is based on the simple intuition that a textural part in an image demonstrates both high local total variation (LTV) and high decay rate of such a variation under image smoothing. Using the notation of [46], the LTV of a pixel at position x in image I is defined as

$$LTV_\sigma(x)(I) \coloneqq G_\sigma * |\nabla I|(x), \tag{2.13}$$

where G_σ is a Gaussian kernel and σ is its standard deviation. The relative decay rate of

the LTV at x is given by:

$$\lambda_\sigma(x) \coloneqq \frac{LTV_\sigma(x)(I) - LTV_\sigma(x)(L_\sigma * I)}{LTV_\sigma(x)(I)}, \tag{2.14}$$

where $L_\sigma * I$ is a low-pass filtered version of I. A high value of $\lambda_\sigma(x)$ (close to 1) carries strong indications of having a textural patch at x. If a low-high-pass filter pair can extract the textural part of an image efficiently, with some 'impurities' of edges and contours, can the above LTV-based observation be exploited to locally-guide the filter pair so that the structural (cartoon) part of I can be kept unchanged? Of course, that guided filter pair cannot be called linear anymore. The authors of [46] proposed an answer to the aforementioned question in the form of a non-linear filter pair that is constructed using a weighted average of I and $L_\sigma * I$, where the weights are a function of λ_σ. The resulting components are computed on a per-pixel basis and thus $c(x)$ and $t(x)$ augment the extensive pool of low-level image features available in the literature. In Chapter 4, the cartoon-texture decomposition of images will be used as the feature vector to locate good FG/BG samples in the trimap for every unknown pixel. This helps in such cases where the color distributions of the FG regions and the BG regions overlap, as it becomes necessary to adopt another more distinctive feature to represent the image pixels.

So far, two types of image cues have been discussed. Whether they are low-level or high-level features, a common approach is to attach such features to a node or a vertex in a graph (the node can represent a pixel, a region, an object, etc.) and then formulate the task at hand as a graph labelling problem. This paves the way to harness the power of the well-established knowledge in graph theory and spectral graph theory in solving vision-related problems. I am devoting the next section to review some basic concepts in graph theory.

2.4 Matrix representations of graphs

The literature review and the proposed research on natural image matting will highlight three kinds of matrix representations of graphs, namely, the adjacency matrix, the degree matrix and the Laplacian matrix. The following paragraphs explain the construction of the three representations and their relation to graph clustering, particularly the undirected

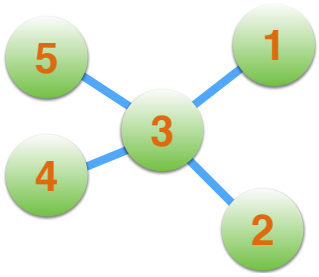
graphs, since this is the type of networks I use throughout my research.

A graph $\mathcal{G}$ is a mathematical structure which consists of a set of vertices $\mathcal{V}$ linked by a set of edges $\mathcal{E}$, most commonly denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The number of vertices, denoted as $|\mathcal{V}|$, defines the graph's order, while $|\mathcal{E}|$, the cardinality of $\{\mathcal{E}\}$, is referred to as the graph's size. Thinking of a graph as a network of nodes and pipes, every edge in the graph bears a weight which reflects the intensity of flow between the two vertices at its ends. When this weight is independent of the direction of flow, i.e. the weight $\mathcal{W}_{ij} = \mathcal{W}_{ji}$, the graph is undirected or simple. If the set $\{\mathcal{V}\}$ can be sub-divided into two independent groups of vertices, where every edge in the graph should not have both its ends at members of the same sub-group, the graph is called a bi-partite graph. To visualize a bipartite graph, let's assume a graph with two groups of vertices, each of which has been given a unique color; in this case, a graph is bipartite if each and every edge in it has two different-colored ends. Bipartite graphs are commonly denoted as $\mathcal{G} = (\mathcal{U}, \mathcal{V}, \mathcal{E})$, where $\mathcal{U}$ and $\mathcal{V}$ refer to the two parts of the graph while $\mathcal{E}$ represents its edges.

For an unweighted graph $\mathcal{G}$, with $|\mathcal{V}| = N$, the adjacency matrix, as its name implies, is an $N \times N$ matrix that describes the adjacency relations between the graph's vertices, while the $N \times N$ diagonal degree matrix indicates the degree of every vertex in the graph; this is the number of edges connected to it. I will refer to the adjacency and degree matrices as A and D respectively. Figure 2.4 depicts an example of a graph together with its adjacency and degree matrices.

The benefits of such matrices go far beyond mere representations since they can be used to infer the properties of a graph. For instance, a principal insight that can be acquired about a graph from its adjacency matrix is its connectedness. This property is a common constraint that arises in many graph-modelled applications, among which is the natural image matting. One possible approach to check the connectedness of a graph is the Dulmage-Mandelsohn decomposition of its adjacency matrix A. During this process, a bi-partite graph is further sub-divided into subsets (or sub-components) of vertices with the constraint that the members of a sub-component should share an edge in a perfect matching of a graph [49][1]. Since a graph is said to be connected if every pair of its

[1]A matching of a graph is a subset of its edges that do not share a common vertex. A vertex is said to be matched if it is one of the endpoints of the edges in the matching. If all the vertices in a graph are matched, the graph is said to be perfectly matched.



(a)

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ -1 & -1 & 4 & -1 & -1 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 0 & 1 \end{pmatrix}$$

(b) (c) (d)

Figure 2.4: A graph and its (b) adjacency, (c) degree and (d) Laplacian matrices respectively

vertices is connected, the number of sub-components indicates whether the whole graph is connected or not.

Another key representation of graphs that is often mentioned in the literature of natural image matting is the graph Laplacian matrix L. This matrix, whose entries are denoted as L_{ij}, is defined as the difference between the degree matrix and the adjacency matrix, $L = D - A$. Since D is a diagonal matrix and A_{ij} is zero wherever nodes i and j are not adjacent, the entries L_{ij} can be expressed as

$$L_{ij} = \begin{cases} D_{ij} & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } A_{ij} = 1 \\ 0 & \text{otherwise.} \end{cases}$$

An example of the Laplacian matrix is shown in Table 2.4. For constructing the Laplacian matrix, the adjacency matrix serves to indicate the spatial neighbourhood of two vertices. If the notion of spatial neighbourhood is replaced with 'appearance neighbourhood' which means the degree of similarity in some feature space (a color coordinate system for example), the adjacency matrix will be replaced by the so-called affinity matrix. The similarity between two vertices is quantified using a kernel function k and thus the entries of the

affinity matrix can be expressed as

$$A_{ij} = \begin{cases} k(i,j) & \text{if } i \neq j \text{ and } \quad j \in \mathcal{N}_i \\ 0 & \text{otherwise,} \end{cases}$$

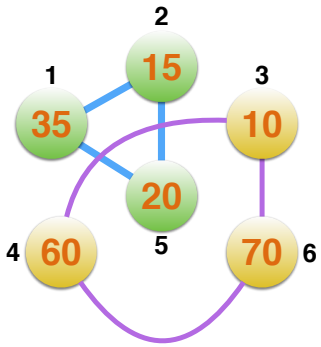
where $\mathcal{N}_i$ represents the neighbourhood of the vertex with index i. If the K nearest neighbours are calculated for every vertex, so only the entries corresponding to those neighbours in the affinity matrix will take the value of the kernel $k(i,j)$, and the rest of the entries will be set to zero. Figure 2.5 shows an example of a graph Laplacian constructed using an affinity matrix. The kernel of this matrix is the absolute difference between the neighbour vertices (with similar color) and the entries of the diagonal matrix D are expressed as $D_{ij} = \sum_j A_{ij}$. Non-neighbour vertices are actually linked with a zero weight, but such links are omitted from the graph illustration to keep the figure clear. To make the matrices meaningful, let the indices of the vertices labelled $\{35, 15, 10, 60, 20, 70\}$ be $\{1, 2, 3, 4, 5, 6\}$ respectively. It can be seen in both examples of the Laplacian matrix that it serves as a discrete Laplace operator, which is the homologue of its continuous version on discrete grids (graphs); it quantifies the difference at a certain site in the graph as compared to its neighbours, in the general sense of neighbourhood. In fact, the entries of both Laplacian matrices are very similar to the most commonly used kernels of 2D 3×3 Laplacian filters, shown below.

-1	-1	-1
-1	8	-1
-1	-1	-1

0	-1	0
-1	4	-1
0	-1	0

These kernels, shown above, are meant to approximate the second derivative of a signal in vertical and horizontal dimensions, with the kernel centred on a particular site in the signal lattice.

The concepts presented in this section belong to the field of spectral graph theory, which revolves around the study of the properties of graphs through the spectra of their representative matrices. It involves other concepts that fall out of the scope of this thesis. Hence, this section has just covered what suffices to clarify my presentation of the proposed research and the related work in the literature.



(a)

$$\begin{pmatrix} 0 & 20 & 0 & 0 & 15 & 0 \\ 20 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 50 & 0 & 60 \\ 0 & 0 & 50 & 0 & 0 & 10 \\ 15 & 5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 60 & 10 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 35 & 0 & 0 & 0 & 0 & 0 \\ 0 & 25 & 0 & 0 & 0 & 0 \\ 0 & 0 & 110 & 0 & 0 & 0 \\ 0 & 0 & 0 & 60 & 0 & 0 \\ 0 & 0 & 0 & 0 & 20 & 0 \\ 0 & 0 & 0 & 0 & 0 & 70 \end{pmatrix} \quad \begin{pmatrix} 35 & -20 & 0 & 0 & -15 & 0 \\ -20 & 25 & 0 & 0 & -5 & 0 \\ 0 & 0 & 110 & -50 & 0 & -60 \\ 0 & 0 & -50 & 60 & 0 & -10 \\ -15 & -5 & 0 & 0 & 20 & 0 \\ 0 & 0 & -60 & -10 & 0 & 70 \end{pmatrix}$$

(b) (c) (d)

Figure 2.5: A graph, (b) its affinity matrix, (c) its corresponding diagonal matrix and (d) its Laplacian

2.5 Graph cuts for computer vision applications

Graph cuts is an optimization technique which lies at the intersection between combinatorial optimization and graph theory. For over a decade, it has become a standard tool in many early vision tasks, with significant advantages and inherent drawbacks. This section presents a discussion on the basics of graph cuts and its incorporation in binary optimization problems.

Perhaps the backbone of the theory underlying graph cuts, and arguably one of the most important theorems in combinatorial optimization [50] is the Ford-Fulkerson theorem. My interpretation of this theorem is the following: Imagine that we have a network flow model, namely, a set of pipes, a source node and a sink node. A *cut* in this model is by definition an interruption in its flow. Let's assume that we need to know what is the minimum number of pipes which, if removed, can interrupt the flow in the model. To calculate this number, we have to *saturate* the network first, i.e. open all the taps shown in Fig. 2.6. Basically, these taps symbolize the source node; similarly, all the green pipes on the left of the figure are assumed to be connected to the sink node of the network

model. In the same figure, it can be seen that removing the blue pipe terminates the proper flow in the network. At the same time, this blue pipe carries the maximum flow from the source pipes on the right to the sink pipes on the left. This is the Ford-Fulkerson theorem: '*The maximum flow from the source to the sink in a network is equal to the min-cut of this network*'. The word 'min-cut' means the cut of minimal cost, where its cost is the sum of the flows of the pipes severed by the cut.

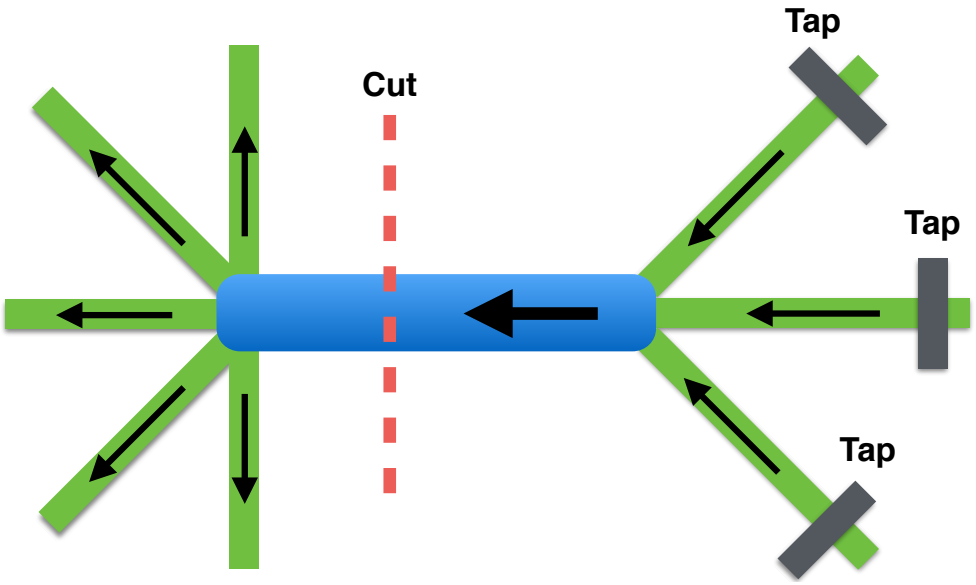
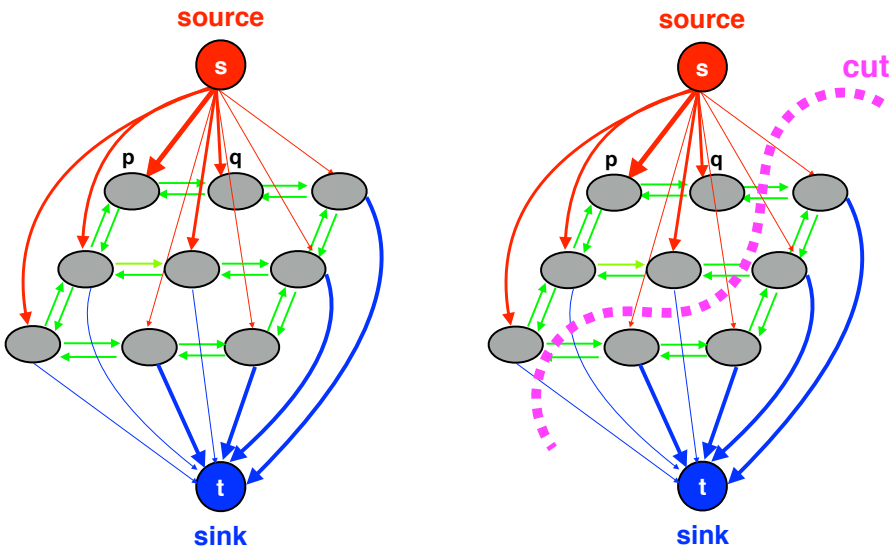


Figure 2.6: Ford-Fulkerson min-cut max-flow theorem [51]

The baseline for a certain problem to benefit from graph cuts is its possession of an inherent graphical model, like the one shown in Fig. 2.7(a), and its viability to be cast as an energy minimization problem. In this graph construction, there is a set of vertices $V = \{s, t\} \cup P$, where s snd t are the source and the sink nodes, and P is the set of all other gray vertices. The cut shown in Fig. 2.7(b) partitions the graph into two *disjoint* groups $\mathcal{S}$ and $\mathcal{T}$. To cross the gap between Fig. 2.6 and Fig. 2.7, the Ford-Fulkerson theorem comes into the picture. In order to infer the min-cut which partitions the graph to $\mathcal{S}$ and $\mathcal{T}$, it is necessary and sufficient to calculate the max-flow of this graph. To the best of my knowledge, the algorithm in [52], [53] is the most efficient for calculating the max-flow on grid graphs.

With the gray nodes representing pixels in Fig. 2.7, and source and sink nodes representing labels (foreground/background, cow/grass, moving/stationary), the graph cuts is well-posed for binary labelling problems. In such types of problems, assigning a label to a node depends on the two following aspects. The first is the similarity between the



(a) Graph construction (b) A cut on the graph

Figure 2.7: The iconic graph construction from Greig et al.[54]. Edge costs are illustrated by the thickness of the arrows. The figure is adapted from Fig.1 in [50].

feature descriptor of the label and that of the node. One possible functional form for measuring such a similarity is the Euclidean norm. This similarity, in the case of pixels, depends on the observed pixel values. Hence, this part of the objective function is called the observed data term. Moreover, in the large majority of image analysis applications, spatial smoothness is a favourable characteristic in the output labelling. This means that for a node/pixel to be assigned a certain label, the dissimilarity between its label and the label of its neighbours should incur a penalty. Hence, the second term in the objective function is called the smoothness term. The overall energy (or objective) function thus takes the form

$$E(f) = \sum_{p \in P} D_p(f_p) + \sum_{(p,q) \in \mathcal{N}} V_{pq}(f_p, f_q), \tag{2.15}$$

where the first summation is over all the graph nodes/sites (pixels or super-pixels in an image for example), while the second summation is over all the set of pairs of neighbouring nodes in a neighbourhood system, 4 or 8-connected neighbourhood, for example, in grid graphs. $D_p(f_p)$ is the cost of assigning the label f_p to node p and $V_{pq}(f_p, f_q)$ is the cost of assigning the labels f_p and f_q to the neighbouring nodes p and q. Referring to the violet cut in Fig. 2.7, by recalling the aforementioned definition for a graph min-cut, the one-one correspondence between the given labels and the min-cut becomes obvious. Last but not least, graph cuts is guaranteed to find the globally-optimum solution for binary problems

if and only if the smoothness terms are sub-modular functions [55, 54], i.e., they satisfy the triangle inequality. For binary problems, this is formulated as

$$V_{pq}(0,0) + V_{pq}(1,1) \leq V_{pq}(0,1) + V_{pq}(1,0). \tag{2.16}$$

To the best of my knowledge, only two algorithms [56], [57] have been proposed to address the problem of optimizing non sub-modular graphs. These algorithms will resurface during the discussion of the thesis contributions with regard to image completion.

Graph cuts is adopted in this thesis to accomplish near-optimal labelling for matting and image completion. First, it is a part of the pipeline of the matting technique discussed in Chapter 5 where it serves to assign each unknown pixel (pixel with unknown alpha value) a FG/BG pair that well-describes its color. In addition, graph cuts is used to compute a near-globally optimal shift-map to fill holes in images. The technique in [56] will be used to accomplish a task called 'image skimming' which precedes the computation of an optimal shift-map and provides the graph cuts algorithm with the labels that will be used in the optimization. It is worth mentioning that I give a brief presentation on the convergence and the optimality properties of graph cuts in section A.8. Although unnecessary, I strongly recommend reading it before proceeding to Chapters 4 through 6.

During the review of the concepts of graph theory and graph optimization, there has been a common process taking place, namely, the evaluation of a certain distance function between the graph vertices, intuitively, to quantify their similarity. Whether this quantified similarity will fill an entry in a representative matrix, or be a data cost or a smoothness cost in a graph labelling problem, there is always an interest in a better similarity metric to better represent a graph. Better similarity assessment can be achieved by either adopting a better or more insightful representation of the data and/or formulating a robust distance function. The rest of this Chapter will shed light on a few techniques that serve both approaches.

In computer vision, like many other fields of science, the acquisition of more features may result in a better representation (and thus exploration) of the data at hand. The previous discussion on graph-cuts has revealed that the output of the graph optimization process will be heavily affected by the choice of data (likelihood) and smoothness terms. The better the ability to quantify similarity/dissimilarity between neighboring sites, the

better the result one can get. This discriminative power is obviously related to the descriptors of the graph sites, whether they are pixels, patches, voxels, objects, etc. When the classical color feature falls short of the expectations, the intuitive direction is to augment the vertex descriptors with other features. Whether they are dense or sparse, this results in a higher-dimensional feature space, the handling of which is more demanding in terms of time and computational complexity. The provoked curse of dimensionality would then heavily impact the performance of matching algorithms, from relatively-small scale applications like disparity estimation to giant search engines. It thus becomes very appealing to come up with a compact representation for the data being studied. There is a vast literature on compact data representation techniques such as hashing and dimensionality reduction. The next section will explain one of the non-linear dimensionality reduction techniques that has been successfully adopted recently in many labelling problems. Chapter 2 will be concluded by the discussion of transduction – a semi-supervised learning approach for similarity assessment and label propagation using graph matrices.

2.6 Dimensionality Reduction Using Locally-linear Embedding

Locally-linear Embedding (LLE) is a member of the family of non-linear dimensionality reduction (NLDR) techniques. Although they do not in general outperform their linear ancestors such as Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), etc. [58], they have proven to be superior in a diversity of labelling problems [59] recently. In many vision tasks, typically like numerous sensory phenomena which involve complex interactions and perception mechanisms, the 'data' is believed to lie near or close to a much lower dimensional manifold, compared to the original representation of the data. For example, we can observe millions of rotated and scaled versions of an object (as images) and figure out this redundancy immediately; but how would a robot know it, just from this large feature space represented by all the images and their RGB values? This is the embedding that all dimensionality reduction techniques compete to infer effectively.

The LLE approach commences from the intuition that a data vector $\vec{X}_i$ in a feature

space $\mathbf{F}$ of dimension D can be reconstructed from a linear combination of its neighbours, given that the manifold of the data is well-sampled, i.e., there are enough data points available. The neighbours of each data point can be determined in various ways; one of these is to confine the neighbourhood to the n-sphere of radius r centred at the data point, or simply to find the K-nearest Neighbours (KNN) for every $\vec{X}_i$. The second task in the LLE pipeline is to determine the weight matrix W which defines the contribution of each neighbour $k_j \in \mathcal{N}_i$ to the vector $\vec{X}_i$, where $\mathcal{N}_i$ refers to the set of neighbours of $\vec{X}_i$. For $i = 1, 2, \ldots, N$ data points, and $j = 1, 2, \ldots, K$ neighbours defined for each point, the $N \times K$ weight matrix W, with entries w_{ij}, is expected to capture the geometric structure of the feature space's manifold. Those weights in W are computed by solving the following constrained optimization problem:

$$W \coloneqq \underset{w_{ij}}{\operatorname{argmin}} \sum_{i=1}^{N} \left| \vec{X}_i - \sum_{j=1}^{K} w_{ij} \vec{X}_j \right|^2 \quad \textbf{s.t.} \quad \sum_{j=1}^{K} w_{ij} = 1 \ \textbf{and} \ w_{ij} = 0 \ \forall \ j \notin \mathcal{N}_i. \tag{2.17}$$

Accordingly, LLE can be described as a technique which aims at modelling manifolds through local linear fits [60].

The method of LLE reduces the dimensionality of data as follows: According to the mathematical field of group theory, the calculated weights exhibit a key symmetry, or invariance to certain transformations (rotation, scaling, translation), for a particular data point and its neighbourhood. Thus, the weights lying on any single row of W characterize the intrinsic geometry of the corresponding neighbourhood in $\mathbf{F}$ whose dimension is D. Now, assume there is a lower dimensional manifold $\mathbf{f}$ of dimension $d \ll D$, that is constructed by a series of scalings, rotations and translations (a linear mapping), and that well-approximates $\mathbf{F}$; it will be dubbed 'the embedded manifold'. Since W is resilient to such transformations (by symmetry), then the characterization power of the w_{ij} for the local geometry of $\mathbf{F}$ should be equally valid for $\mathbf{f}$, i.e., a single row in W_i can reconstruct $\vec{X}_i$ in $\mathbf{F}$ and, equally well, its coordinate $\vec{x}_i$ in $\mathbf{f}$.

The LLE pipeline is concluded by the step of finding that $\mathbf{f}$. In that stage, a linear mapping $Y : \mathbf{F} \to \mathbf{f}$ is sought. This mapping is calculated from the weight matrix W and is thus a neighbourhood-preserving mapping. Every sought d-dimensional $\vec{Y}_i$ represents the global embedded manifold coordinate of $\vec{X}_i$ and is chosen to minimize a quadratic cost

that is similar to the one in Eqn. 2.17. This means that it depends on the minimization of the construction error; however, this time, we fix the weights and solve for $\vec{Y_i}$ as:

$$\underset{Y_i \in \mathbf{f}}{\operatorname{argmin}} \sum_{i=1}^{N} \left| \vec{Y_i} - \sum_{j=1}^{K} w_{ij} \vec{Y_j} \right|^2, \tag{2.18}$$

which is accomplished by solving an $N \times N$ sparse linear system.

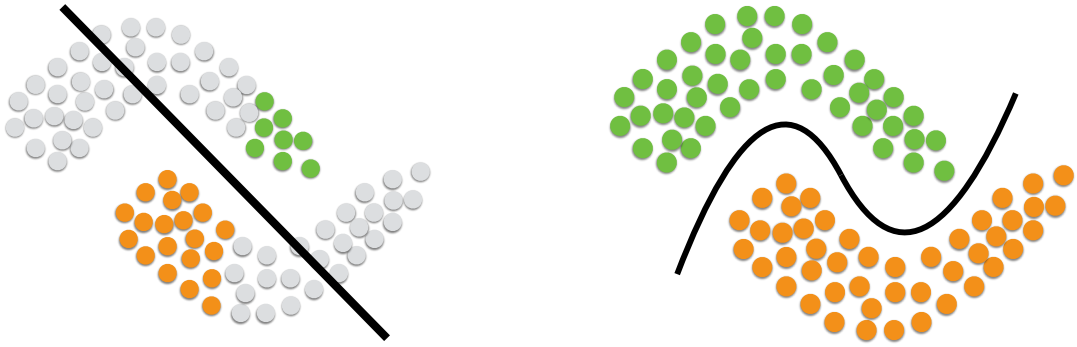
LLE has recently shown appealing results in image editing tasks that involve label propagation, such as image/video re-coloring. Chapter 4 will feature intensive statistics that were collected from the image matting standard datasets, based on which LLE was used to reconstruct the alpha values of unknown pixels from those of their neighbours. As will be seen, this is very favourable in the context of sampling-based matting. The next section is devoted to highlight another, yet very close technique of label propagation, namely, the transductive inference.

2.7 Learning by Transduction

Assuming that some training data (with known class/label) is available, the classical inductive model for inference uses the labelled data points to construct a predictive model or a mapping function. The task of labelling new test points thus becomes trivial; however, the labelling is accurate so long as the inferred mapping is representative. For data lying on complex manifolds, even powerful discriminative model construction approaches, Adaboost [61] and SVM for example, may fail to crystallize a generic model that works equally well with the labelled and the out-of-sample data points. This problem was mentioned in [62] and was attributed to the fact that the inductive model tries to learn a general rule, using only the known data points. In specific cases or contexts, the necessity of learning a general rule can be avoided, and both the labelled and the unlabelled data can be used to classify the unlabelled points; this is transduction.

Figure 2.8 depicts an instance of the two half-moons configuration. As shown, all the observed data points, labelled and unlabelled, are available beforehand. In Fig. 2.8(a), the green and orange points are the labelled data (training points), and the goal is to label the rest of the gray points (testing points). A learning algorithm that uses the

labelled points may fit a hyperplane as shown in the figure. Obviously, classifying the data points on the right of the hyperplane as green, and the ones to the left as orange will result in wrong labels. However, if there is a high confidence that the points are well-separated in the feature space, a function might be learnt from *all the observed data points* such that it should pass through the low-density regions in the feature space; this is shown as the black curve in Fig. 2.8(b). Although it looks very appealing, transduction cannot be used in the case of streaming data, and a high-margin feature space should exist to guarantee the availability of low-density regions. In machine learning, the word 'margin' refers to the distance (Euclidean or any other distance) between a datapoint and the decision (or separating) boundary in the feature space. The separating boundary is shown in Fig. 2.8(a) as a black line segement, and in Fig. 2.8(b) as a section of a smooth function in the feature space.



(a) (b)

Figure 2.8: Iconic illustration of transduction on a two half-moons dataset

Transductive inference can be done using a variety of methods. In this thesis, I use the graph Laplacian-based transduction algorithm that was discussed in [63] and developed in [64]. Figure 2.8 shows that the goal of transduction is to find a smooth mapping f that varies only in regions of low density in the input space, and simultaneously maps every training point to its associated (or a very close) label, i.e., $f(X_i) = Y_i$, where Y_i is the label of the training point X_i. The previous requirements can be formulated as an optimization problem given by

$$\min_f \sum_{i \in T} c_i [Y_i - f(X_i)]^2 + \int_M \|\nabla f\|^2 p^s \mathrm{d}V. \tag{2.19}$$

The first term is a summation over the set of training points T. The purpose of this

term is to penalize the deviation between the labels of the training points and the output of the sought mapping function. The cost of deviation is controlled by the confidence parameters c_i which reflect how certain we are about the labels of the training points, or alternatively, how much fitness is required between every X_i and its known label Y_i. The integral term is meant to favour the mapping functions that vary only in the low-density regions of the input space, according to the parameter s. The input points are assumed to lie on M, which is a sub-manifold of the Euclidean space, and p is the density of the input probability distribution in the canonical measure of M (Lebesgue measure if M is the Euclidean space). Hence, the parameter 's' controls how low the density should be to allow large variations in f; the higher it is, the more confident we are that the data points are well-separated in the feature space. In [65], the authors presented a discrete alternative to the optimization problem in Eqn. 2.19. Their approach adopts graph Laplacian methods that are based on a discrete approximation of the s-weighted Laplacian operator (in the integral term). These methods construct a graph with nodes representing the training and testing points, and the weights of that graph are induced using a kernel (often an exponential kernel) that quantifies the affinities between of the nodes of the graph. The proposed approximation for the problem in Eqn. 2.19 is given by

$$\min_{F \in \mathbb{R}^n} \sum_{i \in T} c_i (Y_i - F_i)^2 + F^T L\, F, \tag{2.20}$$

where L is the graph Laplacian (discussed in sec. 2.4). Equation 2.20 can thus be reduced to:

$$\min_{F \in \mathbb{R}^n} (F - Y)^T C (F - Y) + F^T L\, F, \tag{2.21}$$

where C is the diagonal $n \times n$ matrix in which the i^{th} diagonal element is c_i for a labelled point, and 0 for a test point, Y is the n-dimensional vector in which the i^{th} element is Y_i for a labelled point, and 0 for a test point. The n-dimensional vector F can then be obtained by solving the linear system given by

$$(L + C)\, F = C\, Y. \tag{2.22}$$

For the two half-moons example in Fig. 2.8, F should be thresholded to obtain the binary

labelling of the testing data points.

Recently, transductive inference was introduced to many problems in computer vision, particularly, the problems that satisfy the aforementioned constraints (well separated data points that are all available beforehand) and that involve label propagation. These problems include segmentation [64], matting [66] and people tracking in video sequences [67]. In Chapter 4, I will present a transductive-inference-based approach for robust sampling of trimaps. In that approach, every unknown pixel will be proposed FG or BG pixel from the known regions in the trimap, and the binary labelling will indicate if the unknown pixel accepts the proposed known pixel as one of its 'two half-pairs' (the FG half-pair and the BG half-pair) or not.

Chapter 3

Literature Review

The problems addressed by this thesis have quite an extensive body of research. To keep the discussion focused and concise, its scope will be limited to those methods which are most-recent, closely related to my presented contributions and ranked among the top performers. This ranking is either based on a benchmark (for the matting part) or based on mere subjective plausibility (for the image completion part).

The first section of this chapter presents a literature review for the natural image matting problem. It starts off by highlighting the motivation behind the interest in sampling-based matting, rather than the propagation-based framework, in this thesis; this will be followed by listing its primary challenges. Afterwards, I discuss four state-of-the-art methods. This involves discussing the different stages in their pipeline, their strengths and their inherent limitations. The first section will be concluded by enumerating a few open problems in sampling-based matting.

The literature review of image and scene completion will be discussed in the second section. Hole filling has been tackled in a wide variety of applications, each of which represents a distinct context or problem setup. Thus, the discussion will start off by highlighting all such contexts given in the literature. Among all those contexts, I consider novel view synthesis to be the most general one, i.e., it involves all the challenges that might be encountered in the other contexts. For this reason, and for the sake of an inclusive discussion, I have chosen a few representative techniques to show the following: first, the diversity of the problem setups, second, the diversity of effectiveness levels even in the very recent literature, and finally the room available for improvement, even in single

image completion (which is the simplest setup of hole filling).

3.1 Recent advances in Sampling-based Matting

The following few paragraphs refresh the basics of sampling-based image matting, which have been already mentioned in the second Chapter; they are reproduced here for the reader's convenience. For this Chapter and the next Chapter as well, I will use FG and BG as a shorthand for foreground and background respectively.

The fundamental equation in sampling-based matting is the compositing equation given by:

$$I_i = \alpha_i \times F_i + (1 - \alpha_i) \times B_i, \tag{3.1}$$

where α_i is the partial opacity of the pixel I_i, while F_i and B_i are the Foreground (FG)/Background (BG) pair which contributes to the color of that pixel. Since we seek that α_i for every pixel in the image, and we do not know beforehand which FG/BG pair results in I_i, Eqn. 3.1 defines an under-determined problem. This is resolved by suggesting some pixels (FG/BG pairs) to the algorithm. Those given pixels are then searched by the matting algorithm to find the pair with the best color fitness (or least chromatic distortion), given by:

$$\xi_{color} = \|I_i - (\hat{\alpha_i} F_u + (1 - \hat{\alpha_i}) B_v)\| \quad \text{where,} \tag{3.2a}$$

$$\hat{\alpha_i} = \frac{(I_i - B_v) \cdot (F_u - B_v)}{\|F_u - B_v\|^2}. \tag{3.2b}$$

In the above equations, (F_u, B_v) is a particular pair among those pairs suggested to the algorithm. The chromatic distortion in Eqn. 3.2a determines the suitability of a FG/BG pair by measuring the perpendicular distance between the unknown pixel and the line spanned by the FG/BG pair (this is illustrated in Fig. 2.1). In the beginning of the next Chapter, there will be a discussion on cases where a particular FG/BG pair would minimize the chromatic distortion, yet does not yield the correct alpha value. Such a pair, in addition to any FG/BG pair that does not simultaneously minimize the chromatic distortion and yields the correct alpha value will be called **a wrong pair** throughout

the document. In addition, a FG/BG pair that simultaneously minimizes the chromatic distortion and yields the correct alpha value will be called **a suitable pair** or **a good pair**, throughout the rest of this document. Moreover, in all the following Chapters of this thesis, I will be calling a pixel with unknown alpha value an ***unknown pixel***, and a suggested FG pixel (or BG pixel) a ***half-pair***. Also, the terms ***pair space*** or ***pair pool*** will be used to refer to all the suggested pairs, unless otherwise mentioned. If that space is further reduced in size using certain criteria, it will be dubbed ***short-listed pair space***. The gist of sampling-based matting is to single out the best pair for every I_i in a computationally-efficient manner.

3.1.1 Motivation behind the interest in sampling-based matting

This thesis looks for new ways to benefit from matting in IBR systems. Even though the offline processing of acquired images is allowed in some scenarios (real-time processing is not a constraint), the speed of processing and the computational cost is always an issue. In my case, it was the main decision aspect. Most of the speed-ups that are being reported nowadays, in the computer vision research community, have the same root, namely, Graphical Processing Unit (GPU)-based accelerations. Until the time of writing these lines, I affirm that *there are no methods, reported in the literature, to solve large sparse linear systems, of the sizes commonly encountered in propagation-based matting, on the GPU, in real-time.* The next few lines present the general framework of all propagation-based matting techniques. Afterwards, I will briefly highlight two recent publications that addressed the acceleration of sparse linear solvers on the GPU [68], [69].

Propagation-based matting relies on the propagation of alpha values from the known regions to the unknown regions in the trimap (the meaning of the term 'trimap' was mentioned in the first Chapter). This is done by constructing an affinity matrix between the image pixels, from which the Laplacian matrix can be computed (please refer to Chapter 2). Since the affinity is constrained to a 3×3 local window or to KNN non-local neighbours, both the affinity and the Laplacian matrices are sparse. The per-pixel alpha values can then be obtained by solving a sparse linear system to minimize a quadratic cost in α [70], [71], [72]. The affinity and the Laplacian matrices are of size $N \times N$, where N is the number of image pixels. For an image of size 400×300 pixels (we often have larger

images), the Laplacian matrix has a row/column size of approximately 10^5 elements.

Speeding up matrix computations using GPUs is a research area on its own, and the speed-up factor is controlled by numerous aspects. In [68], the performance varied considerably with different pre-conditioners, the format of the sparse matrix-vector product kernel, and the CPU-GPU synchronization in case some steps are faster to be performed on the CPU. Arguably, the high-performance-computing research community has not reached yet the real-time computations phase; it is in the stage of trying to parallelize the large sparse matrix computations. In [69], the authors reported excellent speedups, up to 10 times faster than a cluster of 12 CPUs, with number of unknowns that ranged up to 10^7. No real-time performance was reported though, and the required computing capabilities are far beyond what is commonly accessible (a cluster of 12 Nvidia Tesla GPUs, 500 dollars each). On the other hand, there is already a real-time sampling-based matting technique in the literature, and its only computational bottleneck is the 'optional' post-processing step which involves a large sparse linear solver [7].

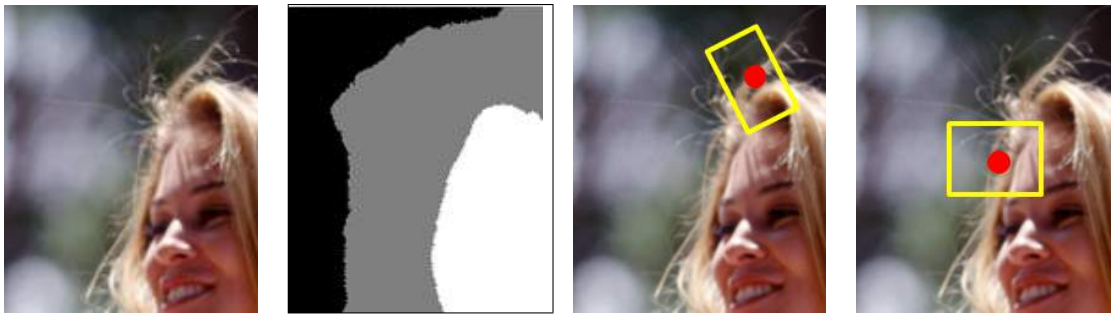
Hence, I chose to focus my research on a matting paradigm that lends itself more easily to fast computation and parallelization, a paradigm that is employed by existing real-time matting techniques; this is what suits IBR systems the most, even though a real-time implementation of my own technique has not be implemented yet. In the following paragraphs, I discuss the primary challenges in sampling-based matting.

I start by presenting my own view for the unifying pipeline of all sampling-based methods; this is comprised of three stages, namely, *descriptor selection*, *sampling strategy* and FG/BG *pair assessment*. In the first stage, one decides how the pixels will be described; is it the color feature only? Which color coordinate system? Is there any other feature that can favourably augment the color feature? etc. In the second stage, given that a comprehensive pair space is huge, one tries to narrow it down, particularly, to sample the most relevant pairs out of it; this is the sampling strategy of a matting technique. Last but not least, the third stage is meant to pick the best FG/BG pair out of the short-listed pair space. Researchers have challenges at each stage of those three.

Selecting a robust feature descriptor could be the main advantage of a matting algorithm, but it could be its main computational bottleneck as well. This is because a high-dimensional feature will probably require a method of dimensionality reduction to

mitigate its burden. Pair assessment may take the chromatic distortion into account, but it may weigh other aspects as well, such as the spatial distance, color distribution fitness and other possible statistics. Weighing those aspects, which lie in different ranges, is another challenge. A chief challenge among them all, that is attributed to the second stage of the pipeline (mentioned in the previous paragraph), is the *distant-but-true* problem. This problem has triggered many techniques for a more efficient sampling strategy, and the challenge still exists. I will elaborate on this problem below due to its significance in my research.

Even when images are captured using a special setup such as the standard dataset of the online benchmark [3] or the datasets used in earlier work [70], one may encounter a variety of challenges, from fuzzy and complex structures to isolated BG regions, in addition to other challenges. One of these challenges is to have the correct FG or BG not nearby. I have observed many cases among the aforementioned datasets where the nearby FG is not the correct FG, or the nearby BG is not the correct BG. One example is shown in Fig. 3.1 where a near FG can yield an incorrect alpha value. Having the red circle representing one unknown pixel, Fig. 3.1(c) depicts a correct case where the nearby FG is the hair while the unknown pixel is clearly a combination of the hair and the BG. On the other hand, Fig. 3.1(d) shows a case in which the unknown pixel is a hair-and-a-BG while the nearest FG is the forehead.



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

Figure 3.1: (a)The original image [70], (b) one possible trimap, (c) true nearby samples and (d) false nearby FG

The rest of this section sheds light on an assortment of the state-of-the-art sampling-based matting techniques. Afterwards, in the light of the limitations of those techniques, the section will be concluded by discussing the open problems in the literature.

3.1.2 Image matting using FG/BG pair pool sharing

Prior to the algorithm proposed in [7], sampling strategies were confined to collecting spatially-near FG/BG samples, which yields an insufficient short-listed pair pool, and thus poor results. It is worth mentioning that among the few trials to improve this local pair pool was the method proposed in [10]. Basically, they proposed to adopt a geodesic distance, that is a function of a parametric color model, rather than a Euclidean spatial distance while searching for 'near' samples. They start by building a Gaussian Mixture Model (GMM) for the spatially-near FG/BG regions, then they construct their short list from the regions that are most likely to generate the color of the unknown pixel under consideration. Although it was a good step towards the enhancement of the acquired pair pool, it did not succeed to generate a rich-enough short list, since it was generated from the spatially-near regions only. This is illustrated in Fig. 3.2(a).

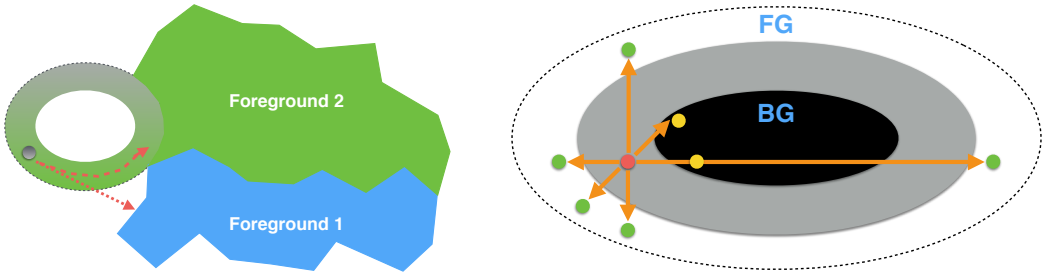
The pipeline of shared matting [7] involves three main stages, namely, sample gathering, sample refinement and local smoothing. The main theme of the whole algorithm is making the most out of a 'cross-talk' that takes place among the neighboring pixels. During sample gathering, the authors proposed a ray-shooting-based sampling strategy. Starting from an unknown pixel p, they shoot k_g rays at certain angles in the image domain [7], so that every pixel collects ${k_g}^2$ FG/BG pairs; particularly, every ray brings to p the closest FG and BG half-pairs to it. This is illustrated in Fig. 3.2(b). To assess the ${k_g}^2$ pairs, they adopted an objective function with photometric, spatial and probabilistic terms. The first two favour the pairs that minimize the chromatic distortion and that are near to the unknown pixel. The last term acknowledges (or weighs) the choice of a specific pair by quantifying the probability of a certain pixel to be affiliated to a FG (or BG). If the chromatic distortion resulting from a specific pair dictates that α_p is close to 1, while its probability of being of affiliated to a FG region is low, the two terms will interact destructively and the objective function will deviate from a minimum value. The objective function is given by:

$$O_p(F_i, B_j) = C_p(F_i, B_j)^{e_C} \times D_p(F_i)^{e_F} \times D_p(B_j)^{e_B} \times A_p(F_i, B_j)^{e_A}, \tag{3.3}$$

where the first three terms are the chromatic distortion and the spatial distance of the

FG/BG pair from the unknown pixel, while the last term is the probabilistic term.

After sample gathering, every unknown pixel checks the FG/BG pairs selected by its k_r (where $k_r > 3$) spatial neighbours during a process called sample refinement. The three FG/BG pairs that result in the least chromatic distortion values are averaged and become the new FG/BG pair of the unknown pixel under consideration. As a local smoothing final step, the FG/BG pairs of the spatially-closest 100 pixels to the pixel under consideration are then averaged with weights derived from a Gaussian kernel and the result of the averaging becomes the final FG/BG pair of the unknown pixel.



(a) (b)

Figure 3.2: (a)According to the geodesic distance (which is a function of a Gaussian mixture color model), the pixel colored in grey is nearer to foreground 2 than foreground 1, although the latter is spatially nearer to it. (b) an illustration of the ray-shooting mechanism proposed by shared matting

3.1.3 A global FG/BG pair space for robust matting

Following [7], other non-local sampling strategies were proposed, each with its own extent of non-locality, as will be clarified later. Global sampling matting [8] represents one of these strategies, and the authors were the first in the literature to point out the *distant-but-true* problem.

Given the trimap, global sampling matting constructs a pair space that is comprised of all the FG/BG samples lying at the boundaries of the trimap. The collected FG and BG samples are shown in green and red respectively in Fig. 3.3(c). For every unknown pixel, the algorithm then searches for the best pair in the space shown in Fig. 3.3(d). Due to the huge size of the pair space, the authors leveraged the advances that have been achieved recently in calculating approximate nearest neighbour fields (ANN) to help them expedite the best-pair selection process. Particularly, they used the efficient PatchMatch algorithm, that was first introduced in [25], to speed up the process of matching patches

for image editing operations.

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

Figure 3.3: (a)The original image[3], (b) one possible trimap, (c) FG(green) and BG(red) dictionaries and (d) the pair-space with a square depicting a particular pair in it.

Beside the new sampling strategy, the algorithm uses a new objective function to assess every pair in the constructed space. This objective function takes in consideration the classical RGB color feature in addition to the distance between the FG/BG samples and the unknown pixel; the nearer the sample, the more privileged it is to be nominated. Their objective function is given by:

$$\xi_{color,distance} = \xi_{color} + \xi_{D_F} + \xi_{D_B}. \tag{3.4}$$

where the first term ξ_{color} is given by Eqn. 3.2a and the second term ξ_{D_F} is meant to favor near FG samples. For a particular FG sample F_i, ξ_{D_F} is given by $\|\frac{\mathbf{x_{F_i}} - \mathbf{x_I}}{D_F}\|$ where the numerator is the Euclidean distance between the FG sample and the unknown pixel, and the denominator is the nearest distance between the unknown pixel and the foreground boundary. The third term is meant to favor near BG samples in the same way as the second term does with FG samples.

Although it was dubbed 'global', this sampling strategy suffers when the true FG (or BG) half-pair is not included in the boundary of the trimap, which may happen often. This has left room for further improvements. More drawbacks of this technique will be highlighted in sub-section 3.1.6.

3.1.4 Weighted color and texture for robust pair selection

The challenge of having a FG neighbouring a BG with an overlapping color distribution can cause any method that relies solely on the color feature to fail, especially if the true FG (or BG) does not exist nearby. The algorithm in [12] addressed this problem by proposing a new texture descriptor as a feature for enhancing the acquired samples. In

that sense, that work enhances the criterion of sampling, while the strategy for sampling remains identical to that of [7]. To acquire a texture descriptor, the authors used a 36×1 feature vector constructed from the 2-level Haar wavelet decomposition of the three color channels of the image. The feature vector is given by

$$FV_T := \{A_{l,c}^{grad}, A_{l,c}^{var}, A_{l,c}^{mean}, H_{l,c}^{mean}, V_{l,c}^{mean}, D_{l,c}^{mean}\} \tag{3.5}$$

where A, H, V and D refer to the approximation, horizontal, vertical and diagonal sub-images, l is the decomposition level, c is the color channel and *mean*, *var*, *grad* refers to the local mean, variance and gradient of the sub-images respectively. To speed up the matching process, later on during the sample selection, they went through a two-step linear dimensionality reduction process using both principal component analysis (PCA) and linear discriminant analysis (LDA). The final output of the texture descriptor calculation is a 3×1 'scaled texture image' that will be used jointly with the original image during the sample selection process.

As mentioned earlier, they collect samples by ray shooting, akin to [7], and nominate the best FG/BG pair by adopting an objective function which gives the texture feature an adaptive weight. This weight is dependent on the degree of overlap between the color distributions of the FG/BG samples. Significantly overlapping color distributions provoke higher texture weight during sample selection, to augment the color feature. Their objective function is given by

$$O = (C_\alpha)^{e_C} \times (T_\alpha)^{e_T} \tag{3.6}$$

where C_α is a function of the chromatic distortion (chromatic distortion embedded in an exponential kernel). T_α serves to authenticate the estimated alpha from the color information and is given by:

$$T_\alpha = \hat{\alpha} \times PF_z^T + (1 - \hat{\alpha}) \times PB_z^T, \tag{3.7}$$

where

$$PF_z^T = \|B_T - T_z\|/(\|B_T - T_z\| + \|F_T - T_z\|) \tag{3.8a}$$

$$PB_z^T = \|F_T - T_z\|/(\|B_T - T_z\| + \|F_T - T_z\|). \tag{3.8b}$$

B_T and F_T are the background and the foregorund texture samples respectively, while T_z is the value of the pixel z in the texture space. If the estimated alpha is close to 1 so the probability of the pixel under consideration being affiliated to the FG should be high in the texture space, otherwise C_α and T_α will interact destructively and devalue the tested pair. The weighting exponents e_C and e_T are functions of the degree of overlap (OL) between the normalized histograms of FG and BG samples in color (OL_c) and texture (OL_T) feature spaces. The significance of overlap between normalized histograms is defined as:

$$OL(H^F, H^B) = \frac{\sum_{i=1}^{n} H^F(i) \times H^B(i)}{\sum_{i=1}^{n} (H^F(i)^2 + H^B(i)^2)/2}, \tag{3.9}$$

where H^F and H^B are the foreground and the background normalized histograms, each with n bins. When the foreground candidate samples and the background candidate samples have the same distribution, the overlap is 1, and if they are distinct, the overlap is 0. In order to limit the effect of the texture feature to the cases where color distributions overlap significantly, the weighing exponents e_C and e_T are determined by the degree of overlap of foreground and background samples in color and texture feature spaces as follows:

$$e_C = e^{-\frac{OL_C}{(OL_T + OL_C)}} \tag{3.10a}$$

$$e_T = e^{-\frac{2 \times OL_T}{(OL_T + OL_C)}}, \tag{3.10b}$$

where OL_C and OL_T are the overlap of foreground and background normalized histograms in color and texture feature spaces, respectively.

3.1.5 Constructing a comprehensive FG/BG pair pool

In all the previously-discussed matting techniques, the sampling was non-parametric, i.e., there was no color modelling taking place. Similar to Rhemann **et al.** [10], the authors of [73] adopted a parametric modelling approach. Their method mainly aims to address the shortcomings of the global sampling matting by collecting a few samples from every color distribution in the known FG and BG regions. Since the samples are gathered from everywhere in the trimap, in lieu of the trimap boundaries only, this method is meant to construct a comprehensive pair space. A fundamental contribution in this method is their sampling strategy. They proposed to adaptively augment the number gathered samples based on the distance from the unknown pixel to the trimap boundary; the larger the distance, the more the gathered samples. For instance, if the unknown pixel is very close to the FG boundary, FG samples will be collected from a very thin stripe that exists right after the FG boundary. The farther the unknown pixel from the boundary, the thicker is the stripe used to collect the FG samples. The same holds for the BG samples. This is illustrated in Fig. 3.4.

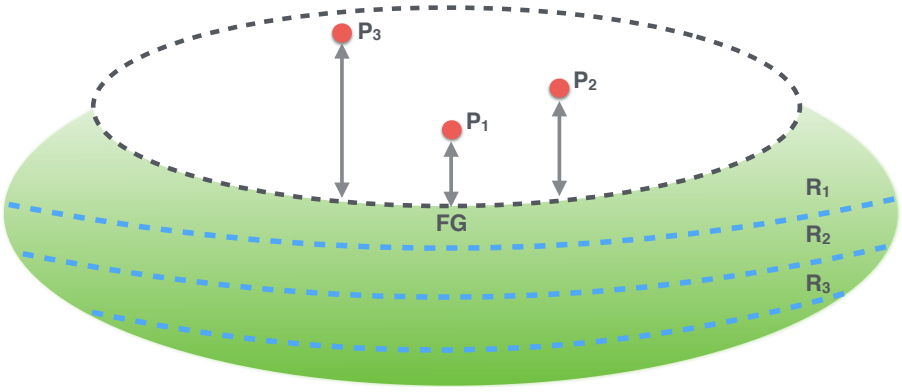


Figure 3.4: An illustration of the sampling strategy of [73]. P_1 will collect samples from the first region only (R_1), P_2 will collect samples from R_2 ($R_1 \subset R_2$) since it is farther from the FG boundary, and P_3 will collect samples from R_3 ($R_1 \subset R_2 \subset R_3$).

The authors retained the color feature and augmented it with the spatial term of [8], in addition to a term calculated from color statistics. The objective function is given by

$$O_z(F_i, B_j) = K_z(F_i, B_j) \times S_z(F_i, B_j) \times C_z(F_i, B_j). \tag{3.11}$$

This equation evaluates a particular pair (F_i, B_j) for an unknown pixel z using the chromatic distortion presented by the term $K_z(F_i, B_j)$, a function of the Euclidean spatial

distance $S_z(F_i, B_j)$ from F_i and B_j to z, and lastly the term $C_z(F_i, B_j)$ which favours pairs from well-separated color distributions, where the overlap between the distributions is measured by Cohen's d-value embedded in C_z. The comprehensive sampling of [73] was later combined with the color/texture feature descriptor and the objective function of [12] in [13]. The same procedure was used in [74] for video matting with an additional term in the objective function that is added to encourage the temporal coherence of the calculated alpha values. This latter term is simply a function of the absolute difference between α_t and α_{t-1} weighted by the confidence they have in α_{t-1} and the accuracy of the optical flow estimation used to get the alpha map of frame F_t from that of frame F_{t-1}.

So far, I have discussed three matting techniques that collect non-local samples. Shared matting was shooting rays in the spatial domain, global sampling matting was considering all pairs on the boundaries of the trimap, and lastly the comprehensive sampling matting which gathers the samples from everywhere in the trimap but augments the scope of every unknown pixel according to its distance from the boundary. I consider the latter approach to be the one that benefits the most from the non-local sampling approach.

3.1.6 Open problems in the literature

The discussion on the open problems in the literature will be comprised of the two following aspects. First, I will highlight some drawbacks for the techniques presented in the previous sub-sections. Afterwards, in the light of those drawbacks, I will enumerate the prospective directions of improvements, in sampling-based matting in general, and its application to the IBR systems in particular.

Even though the sampling strategy of [73] has succeeded to guarantee a comprehensive pair space, their short-listed pair space can still be deficient. In general, I found that relying on the spatial nearness between the unknown pixel and its most suitable pair is not necessarily efficient. This becomes more evident when we have a highly-textured background. More illustrations will be given while presenting my research in Chapter 4. What is not less important is that when the unknown pixel is far from the boundaries of the trimap, this technique may include half-pairs from completely irrelevant regions, which adds more workload that is unneeded to the pair-assessment stage, as will be seen in Chapter 4.

The sampling strategy of [12] and [7] are almost the same, hence, they share the same drawback which is the dependence of the collected samples on the structure of the trimap. Ray shooting is an efficient way to guarantee the diversity of the short list, but it tends to collect local samples (especially with FG regions) since each ray is allowed to fetch one FG or BG pixel, which is the nearest one it hits. The sampling strategy of [8] fails if we need half-pairs that are far from the trimap boundaries; this may happen often.

Continuing my discussion on [8], their pair assessment strategy may pick a wrong FG/BG pair that minimizes the chromatic distortion, yet does not yield the correct alpha value, *the color ambiguity problem*. This stems from the fact that they offer the 'whole' pair space (all pixels on the edges of the trimap) for every unknown pixel; the more the alternatives the more probable it is to pick an incorrect pair. The strategy of [12] uses an efficient descriptor, which may alleviate the color ambiguity problem. Nevertheless, this comes at a significant computational cost of the texture descriptor which involves an eigenvalue decomposition of a 36×1 descriptor for all the pixels in the image; for video matting, this is a huge amount of computation.

Obviously, the door is still wide open for more improvements, in all the stages of the sampling-based matting pipeline. I discuss below my suggestions about the prospective research directions:

- A more flexible, yet robust, sampling strategy is still missing, a strategy that is able to handle backgrounds with varying complexities and diverse textures, and to handle trimaps with various structures. Flexibility facilitates the construction of a sufficient short-listed pair space, while robustness refer to the aspects in the system that ensure minimal redundancy and irrelevancy in the short-listed samples. Example cases for the deficiency of the sampling strategies in the literature will be given in Chapter 4.

- Spatially-near unknown pixels tend to share the same FG/BG pair, since their alpha values are close to each other. The necessity to go through the pair-assessment process for each of them, as well as the possibility of using the locally-linear assumption to ameliorate the computational burden, have not been explored so far.

- The FG/BG best-pair-search is done simultaneously so far in the literature, i.e., we

search for the best pair at the same time. The benefit of searching for an appropriate half-pair first and then match it to a complement half-pair (so that they together can well-describe the color of the unknown pixel) has not been explored before.

- By adopting the composition model in Eqn. 3.1, current techniques in the literature are not able to deal with cases where one half-pair that would minimize the chromatic distortion cannot be found in the whole image. This is mostly the FG half-pair since the background is always opaque. For example, a plastic bag or any similar transparent surface will be marked wholly as gray in a trimap. However, a known 'opaque plastic' region may not exist among the known FG regions in the image, only the opaque background can be seen through. To deal with such a challenge, a better composition model is required. In general, this model has to support *multiplicative composites*, in addition to the current linear-convex composites. Moreover, it has to allow an unknown pixel under consideration to express itself on the right-hand side of Eqn. 3.1 if it did not find a pair that yields a low chromatic distortion. Some instances of this challenge will be shown in the beginning of the next Chapter.

- By adopting the composition model in Eqn. 3.1, the process of FG/BG pair assessment and best-pair nomination is pixel-wise by definition, and this necessitates a post-processing smoothing step to ensure that similar and spatially-near pixels have similar alpha values. Every sampling-based matting technique in the literature applies a post-smoothing step for the calculated alpha maps, whether is it Laplacian-based [75] smoothing [8, 12, 73] or guided filtering-based [76] smoothing [8]. This means that all sampling-based matting techniques are actually hybrid techniques. First, they calculate alpha values by gathering samples from the known regions in the trimap, then they smooth it by propagation. Relying solely on the gathered samples to produce smooth and accurate alpha maps, without resorting to solving a huge number of linear equations, is still an open problem.

- Trimaps are, so far, manually generated, which is a painstaking task for video matting and undesirable in IBR systems. Can the trimap generation be automated?

In Chapter 4, Chapter 5 and Chapter 6, I address some of the aforementioned

questions and challenges. Chapter 4 and Chapter 5 highlight three variants of a new trimap sampling strategy that overcomes critical drawbacks in the current methods, while Chapter 6 highlights a new formulation for the automatic generation of trimaps using laws of perceptual grouping.

3.1.7 Common Practices in The Literature And Comments on The Benchmark

1. For sampling-based matting techniques, the step of assessing the gathered FG/BG pairs consists of the three following steps. First, the cost function (chromatic distortion for example) is calculated for every FG/BG pair, for which there is a corresponding alpha value. A tolerance is allowed in the calculated alpha values; this tolerance has a value of 0.2 in the implementation of [12] and [73]. The values of the cost function that correspond to alpha values beyond that tolerance ($\alpha > 1.2$ or $\alpha < -0.2$) are set to infinity/zero if the cost function is to be minimized/maximized. The final alpha value is then taken to be the alpha value corresponding to the minimum/maximum value of the cost function, clipped at the value of 0 or 1. If the array of the calculated cost function values contains more than one occurrence of the minimum/maximum value, the algorithm picks the alpha value corresponding to the first occurrence only. This procedure has consequences which will constitute a part of the motivation for the matting techniques that will be presented in the next Chapter.

2. The size of the matting dataset is small compared to the standard datasets available for other computer vision tasks, like the object segmentation dataset for example. In addition, there are some challenges that exist in the training dataset, but do not exist in the testing dataset, and vice versa. At the same time, only the performance on the testing dataset is what determines an algorithm's position in the benchmark ranking. The majority of the matting researchers do not provide their code online, and thus their performance on the training dataset and the challenges in its images cannot be analyzed.

3. The subjective superiority of a proposed approach, compared to earlier techniques in

the literature, would be demonstrated on patches in particular images in the training dataset without referring to the quantitative and the qualitative performance on the whole training dataset.

4. Addressing a challenge is not always correlated with the position in the benchmark. A contribution to sampling-based matting could be along one or more of these directions: feature selection, sample gathering, pair assessment (objective function). One would propose a sampling strategy that brings good pairs to an unknown pixel. However, assessing those pairs with an objective function that is not as efficient would deprive the proposed sampling method from attaining a top position in the benchmark. This does not mean that the proposed sampling method is not novel. Publishing the proposed sampling method, in this case, would serve as time-stamping, because the competition is very fierce.

5. Another point that is related to the previous one: An algorithm could address an open problem that does not exist often enough (in the testing dataset) to move the average rank of the algorithm up in the benchmark table. This is not necessarily because the challenge is cursory or superficial; it could be due to the fact that the dataset itself is very small.

The next section is devoted to highlight the recent advances in two closely-related problems in computer vision, namely, hole filling in images and scene completion.

3.2 Recent advances in hole filling and dis-occlusion management

Chapter 1 featured an abstract presentation about dis-occlusion management. Before proceeding to its literature review, I present a more detailed picture of that problem in the following few lines. The term 'hole filling' has been brought to the surface, in the literature, in four different contexts which are: single image hole filling, video completion, depth-guided inpainting and stereoscopic inpainting. In the first context, the hole is usually the result of an object removal or scene re-arrangement operation, a kind of task often needed in image editing. The information available in such a context is the

image only, the spatial information. In video completion, the reason for having a hole is usually the same as in single image completion, but now the temporal information is also available in addition to spatial information. In depth-guided inpainting, one starts to have another source of holes, other than object removal, which is the dis-occlusions that result from scene warping to a virtual viewpoint. The information available might be the spatial+depth information or the spatial+depth+temporal information; however, in both cases, stereoscopic consistency is overlooked. The depth information *is* exploited for hole filling, but every view or frame is completed independently. Finally, in stereoscopic inpainting, we find the stereoscopic consistency guaranteed, but the temporal information is not considered. This is the current situation in the literature, and the applications are limited to the image editing-type of tasks. Figure 3.5 depicts the aforementioned classification that, despite its importance, is not at all mentioned in the literature. I believe that Novel View Synthesis (NVS) involves all the aforementioned research areas as sub-problems. Consequently, I devoted this section to shed light on the recent advances in these contexts, present the strengths of the proposed methods, and finally, pinpoint the open problems in the literature.

Being the simplest completion scenario, the advances in single image hole filling have triggered many improvements in other hole filling applications such as stereoscopic inpainting and depth-guided video completion. Hence, the following sub-sections will start off the section by discussing the three most-widely used exemplar-based techniques for single image hole filling. For those techniques, I will go through the pipeline that was mentioned in Chapter 1 and Chapter 2, which is the unifying pipeline in the large majority of such techniques. Particularly, I will state the criteria used in constructing the Bag of Significant Patches (BoSP), before proceeding to the different objective functions adopted to single out the proper patch from that aggregated bag. Following the discussion on single-image hole filling, I will proceed to present an assortment of dis-occlusion management techniques. They are chosen to well-represent both the most recent and the most relevant literature. In Table 3.1, I summarize a few symbols that will appear often throughout the rest of this section.

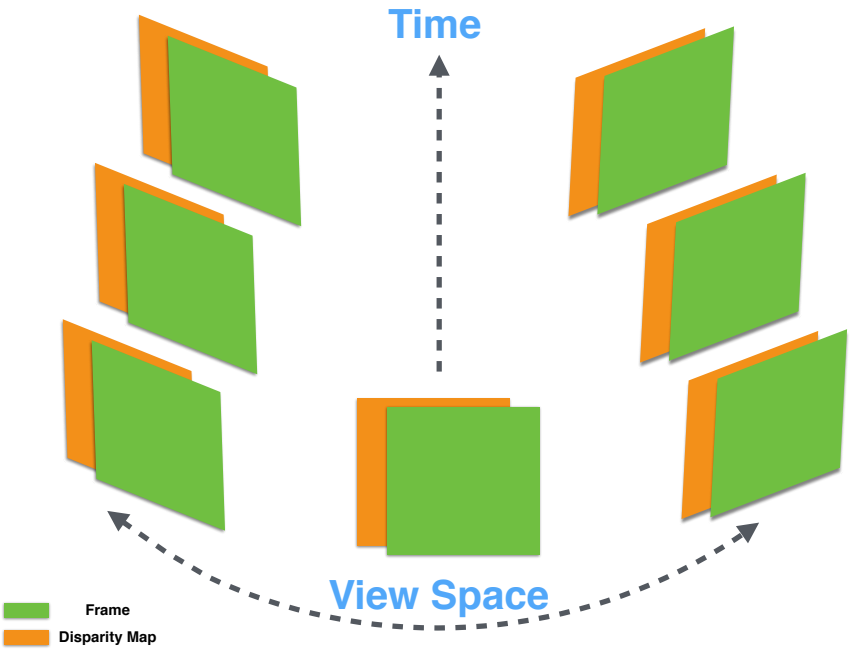


Figure 3.5: An illustration of the different contexts of hole filling, indicating the possible domains of completion and BoSP construction; it can be the spatial domain only in single image hole filling (only one green frame is used for completion), the temporal domain only in video completion (green frames across time are available), view-space+depth only in stereoscopic inpainting (green and orange frames are available for different viewpoints without temporal information), time+depth only in depth-guided video completion, and finally view-space+time+depth in novel view synthesis.

Table 3.1: Symbols that will be frequently used during the review of image completion techniques

Symbol	**Meaning**
Ω	The hole region in an image
$\bar{\Omega}$	The known/source (non-hole) region in an image
$\partial\Omega$	The boundary of the hole region
p_b	A pixel on the boundary of the hole region
Ψ_{p_b}	A patch centred on p_b in a Criminsi-based approach
$\Psi_{p_b^*}$	The patch with the highest filling priority in a Criminsi-based approach
$\mathcal{P}(p)$	An image patch centred on an image pixel p
$\mathcal{P}(p+s)$	An image patch centred on an image pixel p and shifted by s pixels

3.2.1 Criminsi's priority-based concentric hole filling

The method proposed by Criminsi et al. [21] was the first patch-based hole filling approach. Following it, the use of patches (exemplars) for such a task has become the mainstream. It is a non-optimal (greedy) algorithm that has been followed by more efficient techniques. The motivation to review it, however, stems from the fact that it is adopted by many recently-proposed depth-guided inpainting techniques. The stages of the pipeline will be discussed below.

The algorithm takes as an input a hole (target) region Ω and sets the source region $\bar{\Omega}$ as $\mathcal{L}_{\bar{\Omega}} = \mathcal{L} - \mathcal{L}_{\Omega}$; this is the BoSP in this algorithm. $\mathcal{L}$ is the set of all pixel locations in the image, $\mathcal{L}_{\Omega}$ is the set of all pixel locations in the hole region and $\mathcal{L}_{\bar{\Omega}}$ is the set of all pixel locations in the source region. Starting from $\partial\Omega$, which is the boundary of Ω (or the fill front), and using a pre-determined patch size, the algorithm proceeds in an iterative, best-first synthesis strategy. Since the fill front is comprised of pixels affiliated to soft and/or textured regions as well as regions with local structure (an edge for example), a priority-based filling approach is adopted, which gives local structures the priority to be inpainted first. The priority function is calculated for every pixel on $\partial\Omega$ and is given by

$$\mathrm{Pro}(p_b) = C(p_b) \times D(p_b) \tag{3.12}$$

where p_b is a pixel on the hole boundary, $Pro(\cdot)$ is the priority of the argument, and $C(\cdot)$ and $D(\cdot)$ are the confidence and data terms respectively; they are given by

$$C(p_b) = \frac{\sum\limits_{q \in \Psi_{p_b} \cap \bar{\Omega}} C(q)}{|\Psi_{p_b}|} \tag{3.13a}$$

$$D(p_b) = \frac{|\nabla I^{\perp}_{p_b} \cdot n_{p_b}|}{\alpha}, \tag{3.13b}$$

where Ψ_{p_b} is a patch centred at p_b, $|\Psi_{p_b}|$ is its area (number of pixels), n_{p_b} is a unit vector orthogonal to $\partial\Omega$ at p_b, and finally $\perp$ denotes the orthogonal operator. For initializing the concentric hole filling process, the confidence $C(x)$ is set as follows: $C(x) = 1\ \forall x \in \bar{\Omega}$ and $C(x) = 0\ \forall x \in \Omega$. Hence, the first equation dictates that the filling should start

from where we have the largest amount of information (non-hole pixels), while the second equation dictates that the peeling should start from where the strength of the linear structure (also called the isophote) hitting $\partial\Omega$ is large. The priority is calculated for every Ψ_{p_b} centred on every p_b, and once the patch with the highest priority (centred on ${p_b}^*$) is determined, its best match patch from $\bar{\Omega}$ is copied to its place; the best match is determined based on the Sum of Squared Differences (SSD) metric. With a part of the hole now filled, $\partial\Omega$ is updated and the whole process is then iterated. I will use ${p_b}^*$ and $\Psi_{p_b^*}$ throughout the discussion of every Criminsi-inspired approach to refer to the hole boundary pixel of the highest priority and the patch centred on it respectively.

3.2.2 Content-aware filling

As opposed to the greedy approach of Criminsi et al., the methods proposed in [23] and [24] have adopted an optimized framework that minimizes an objective function, called the coherence measure, which is given by

$$d_{coherence} = \sum_{P \in \Omega} \min_{Q \in \bar{\Omega}} \|P - Q\|^2, \tag{3.14}$$

where P is a patch in the hole region, Q is a patch in the non-hole region and $\|.\|^2$ is a distance function. The role of this objective function is to penalize the completions that involve 'alien' patches, i.e., the patches whose best matches in $\bar{\Omega}$ are not similar to them. Equation 3.14 is optimized using a multi-scale strategy and an expectation-maximization (EM) procedure. In the expectation step, the best match Q is sought for every $P \in \Omega$, while in the maximization step, the color of each hole pixel is reconstructed by a voting procedure, since every hole pixel is covered by multiple overlapping patches. Following the construction of an image pyramid, the aforementioned EM procedure starts at the coarsest level and the result at every level is interpolated to the next finer level. The number of levels in the image pyramid is usually determined by setting a lower bound for the hole size, i.e., keep down-sampling the image until the hole is less than or equal 100×100 pixels. This technique was the first optimized hole filling framework to be adopted in depth-guided inpainting applications [6].

The most computationally expensive stage in the pipeline of all exemplar techniques

is the patch matching step. Consequently, it has been very favourable to combine those techniques with other strategies that yield more efficient matching operation, e.g., by speeding up finding correspondences and/or narrowing the number of comparisons. The PatchMatch algorithm proposed in [25] has succeeded to do this task. Basically, it calculates an approximate nearest neighbour (ANN) field using a randomized search strategy. Assuming the eventual goal is to find patch correspondences in image B for every patch in image A, the algorithm starts from a random shift map that defines a shift between the centres of every patch in A and its best patch in B. This random initialization is then updated iteratively through the following two processes.

First, good initial offsets are propagated to neighbouring patches. This is done by letting each patch check whether the offsets of its neighbouring patches would bring it a better match or not, and to adopt their offset if so. Afterwards, each patch searches *again* for a better match in concentric regions around its best offset; those regions have radii that start from the size of the image, and then are halved each time until reaching the value 1. In a nutshell, the algorithm is comprised of three stages, namely, initialization, propagation and search, where the latter two stages are iterated until convergence. Since its proposal, this randomized correspondence algorithm has been adopted in a multitude of image analysis and vision applications, and is combined with [23] and [24] in Adobe Photoshop's Content-aware Filling functionality.

3.2.3 Hole filling using statistics of patch offsets

In the technique proposed in [28], the authors presented a unifying framework for various image editing operations. They crystallized tasks like hole filling, retargeting and scene rearrangement as a quest of an optimal shift-map. Recalling the naming convention in Chapter 2 and Table 3.1, if the known pixel p' in the known region $\bar{\Omega}$ is the best pixel to fill the hole pixel p in the hole region Ω, they sought the value $s = p - p'$ for the pixel p. Towards this goal, they arranged the image pixels on the nodes of a graph and sought the optimal map using hierarchical graph cuts, with color and gradient-based smoothness terms. Their BoSP thus comprised all the patches in the image. While their approach is versatile, and proved to be useful in multiple tasks, it is computationally demanding since it considers all the possible shifts for every hole pixel. Moreover, for the hole filling task

in particular, offering a hole pixel all the possible shifts tends to insert degenerate seams especially near the hole boundary, which results in violating the image's linear structures. Hence, there was a necessity for finding a method to refine that BoSP.

Inspired by [28], the authors of [26] proposed to construct the BoSP from the dominant offsets between similar patches. Particularly, they observed that those dominant offsets are sparse and include all the required shifts to fill the missing regions in an image. Their method starts by calculating the approximate nearest neighbour (ANN) field for all the patches in an image. For every patch $\mathcal{P}$ centred at p, they nominate a shift s which is defined as

$$s(p) = \underset{s}{\operatorname{argmin}} \ \|\mathcal{P}(p+s) - \mathcal{P}(p)\|^2. \tag{3.15}$$

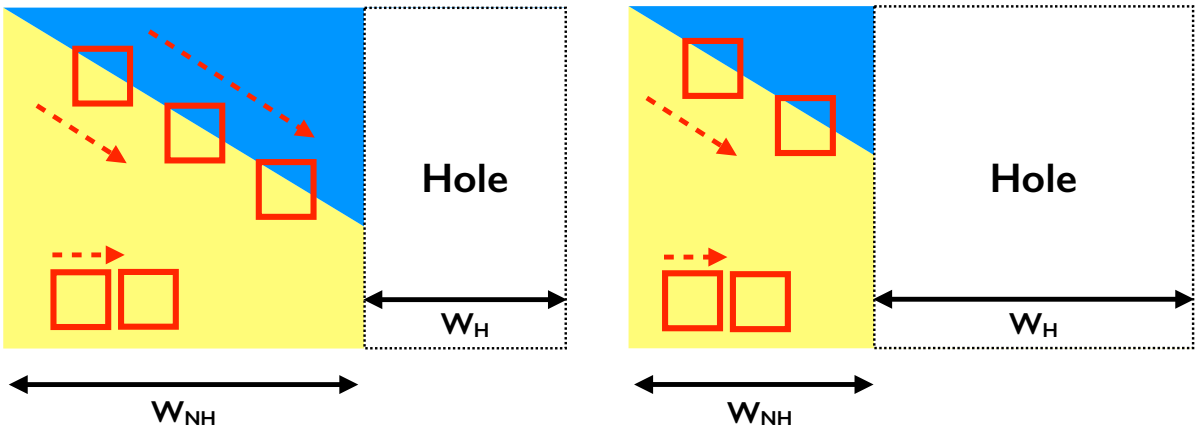
Afterwards, they compute a histogram of all the shifts from which they single out K modes (or peaks of probability mass function) to complete the hole. The essence is that every hole pixel/patch will find its best match within one of those calculated shift modes. This is illustrated in Fig. 3.6(a). With those K mode shifts as labels, they seek the optimal shift map by solving a graph labelling problem whose energy is given by:

$$E(L) = \sum_{p_i \in \Omega} E_d(L(p_i)) \quad + \sum_{(p_i,p_j)\,:\,p_i \in \Omega, p_j \in \Omega} E_s(L(p_i), L(p_j)), \tag{3.16}$$

where $L(p_i)$ is the shift assigned to p_i and the data term $E_d(L(p_i))$ is a constraint assigning a zero cost to a shift (label) which moves the hole pixel p_i to a known pixel location, and ∞ cost to a shift if it moves the hole pixel to another hole pixel or to outside the image lattice. The smoothness cost $E_s(L(p_i), L(p_j))$ for the two arbitrary shifts s_a and s_b, between the neighbouring pixels p_i and p_j (4-connected neighbourhood) is given by:

$$\|I(p_i + s_a) - I(p_i + s_b)\|^2 + \|I(p_j + s_a) - I(p_j + s_b)\|^2, \tag{3.17}$$

where $I(\cdot)$ is the color of the image at the specified attribute, $s_a = L(p_i)$ and $s_b = L(p_j)$. If s_a and s_b are not equal, p_i and p_j will be moved by different offsets and the smoothness cost will increase *unless* the shifting results in coherent seams, i.e., similar RGB values between the shifted pixels. Despite being fast and successful in many cases, the algorithm can fail easily if the statistics of large soft regions dominate the statistics of much smaller



(a) (b)

Figure 3.6: (a) The dominant patch offsets are expected to reveal the structural regions as well as the soft regions in the image, which represents an important cue for completion. Only three examples of possible dominant offsets are shown as red arrows, with the length and the direction of the arrow representing the magnitude and the direction of the shift respectively. Part (b) shows a failure case for the statistics of patch offsets where the size of the hole, whose width is W_H, is larger than the calculated shifts from the non-hole region, whose width is W_{NH}.

structured regions in an image, i.e., it will complete the structures with the dominating soft regions. It would also fail if the hole dimensions are larger than the shifts computed. An example of this case is illustrated in Fig. 3.6(b).

3.2.4 An observation regarding the literature

This sub-section does not only serve the inclusiveness of the discussion on the literature of depth-based inpainting. It also makes it easier for the reader to comprehend the reason behind the considerable differences among the frameworks covered in the coming sub-sections.

Arguably, there are a few principal milestones in the literature of image completion. They commenced by the method proposed in [15], which relies on the diffusion of pixel color values, evolved to the exemplar-based methods of [21] then [23, 25], before reaching the good performance of the methods that involve graph optimization such as [26] and [35]. Nowadays, it is widely accepted in the image completion research community that the capabilities of the early methods such as [15] are limited to specific inpainting tasks, and thus the new contributions are built over the most recent, which are the more efficient. The main building block of any depth-guided inpainting technique is an image completion method. Hence, it is expected to find the recent literature of dis-occlusion management

adopting the more robust image completion approaches. Actually this is not the case. Even the very recent literature of depth-guided inpainting and dis-occlusion management carries the traces of all kinds of hole filling approaches, the ones who are able to deal with a variety of challenges as well as the ones who can manage fewer challenges.

I preferred to share with the reader an assortment of approaches that are affiliated to a wide variety of frameworks, to show how diverse the literature is. Last but not least, a similar observation was made in the research areas of background subtraction in video sequences and natural image matting, and a good step was taken: an experimental comprehensive survey and an online benchmark [77].

3.2.5 Hierarchical hole filling

Hierarchical hole filling [1] is a pyramidal approach which diffuses color values, across image scales, from a hole-free low-resolution estimate of the warped image to its full resolution. The algorithm starts by building an image pyramid of N levels, each of which involves a *Reduce* operation. In each *Reduce* iteration, a down-sampled version of the image is obtained by means of an average filter, where the calculations only involve the non-hole pixels in the sliding filter kernel. The authors kept N a variable, i.e., they stop *Reducing* the image only when no more hole pixels exist in the respective down-sampled image; hence, N is dependent on the hole size. A reduced image at the n^{th} pyramid level is denoted R_n.

Starting from the hole-free R_N, the coarsest image, the algorithm iteratively applies an *Expand* (linear interpolation) operation across the pyramid levels. Taking R_N as an input, the expansion produces a hole-free image E_{N-1}, which is used to fill the holes in R_{N-1}, and so on. This way, the algorithm avoids the depth map filtering operations adopted by previous techniques [78] which result in noticeable geometric distortions in the warped images. For further improvement around depth discontinuities, and to keep the depth information in the loop, the authors proposed a depth-adaptive version of their hierarchical hole filling approach. This version of the algorithm counts on the observation that the missing regions, most likely, belong to a background region. It starts by warping the reference frame and its associated depth map, before a weight is calculated for every pixel in the warped color image. This weight increases as the disparity of the pixel

decreases, so that the background pixels are given higher weights than the foreground ones. The warped color image is then weighted according to the calculated map, which emphasizes the effect of the background regions during the *Reduce* and *Expand* operations. Finally, the same hierarchical hole filling procedure is applied on the weighted color image.

3.2.6 A new data term for better concentric hole filling

In contrast to hierarchical hole filling, the method in [38] uses exemplars (patches) from the source region to fill the hole (target) region, by adopting the approach of [21]. It was not the first method to manage dis-occlusions using an exemplar-based approach since it was preceded by [79] and [80]. However, those latter approaches had many drawbacks, chiefly their assumption that the depth maps of the virtual views are known a priori, which limits their applicability in applications like Free Viewpoint Video (FVV). The main contribution of [38] is a robust procedure for calculating the filling priority and a new objective function for nominating the best patch from the BoSP; nevertheless, it does not consider any temporal information.

The filling priority in this technique is calculated using the same formula adopted in Criminsi et al.'s approach. Since the classical method in [21] gives equal priorities for all the pixels on the boundary of the source region, it gives equal importance for foreground and background pixels. However, based on the observation that the hole regions are usually a part of the background, a new method for initializing the confidence term was proposed. The adopted initialization sets the confidence of the foreground regions to zero and those of the background regions to one. Hence, their fill front is comprised of the background pixels lying on the hole boundary. Finally, the classification of hole boundary pixels to foreground (or background) pixels is done by thresholding the warped depth map. For the second term of the filling priority equation, the data term, a new method that is more stable (than Eqn. 3.13b) in estimating the direction of local image structure was proposed. Basically, the new data term is a function of the eigenvalues of a 2×2 Hessian matrix of a decision window, centred around every pixel in the hole boundary. It is designed so that it yields a low priority in case of homogeneous and texture-rich regions, and to give a high priority for the continuation of local image structures.

The last stage of the algorithm involves the nomination of the best patch in the BoSP

for every $\Psi_{p_b^*}$ centered at every p_b^*, where the BoSP is built from a rectangular region surrounding $\Psi_{p_b^*}$. The objective function used for nomination is comprised of two terms; the first of these is the sum of the absolute differences between the compared patches, and the second term favours the patches that lie on the same row as the incomplete patch under consideration. This second term is weighed by a free parameter in the algorithm.

3.2.7 Space-time hole filling with random walks

Similar to the method presented in the previous sub-section, the technique proposed in [81] capitalizes on Criminsi et al.'s approach for concentric hole filling [21]. Their perspective of hole filling is similar to what I referred to in the beginning of this section as depth-guided inpainting, i.e. they use the depth information while filling occlusions, however, they fill each frame independently, which violates the stereoscopic consistency. This drawback was mentioned explicitly in their article. In the following few paragraphs, the two main stages of their method will be explained, namely, the sample space selection (BoSP construction) and the space-time hole filling procedure.

Given the video frames and their corresponding depth maps, the algorithm starts by classifying the boundaries of the holes, which result from view warping, as either affiliated to the foreground or the background. This is done based on the sign of the warped depth map Laplacian, i.e., they calculate the Laplacian of the depth map of the reference frame, warp it, and then check the sign of the hole boundary pixels in that warped image. The purpose of this process is to assign a zero priority for the foreground hole boundary pixels during the hole filling process later on. In this aspect, they took an exactly similar approach to the method in [38] by defining only one 'fill front', which is the background part of the hole boundary.

The sample space construction commences by a binary segmentation step of the warped depth map into foreground and background regions. The random walker segmentation method of [82] was adopted. This segmentation technique takes as an input some seeds for the foreground and background regions, generated from the hole boundary classification step, and propagates the labelling to the rest of the pixels in the depth map. It is a transduction operation (recall Chapter 2) on the depth map's weighted undirected graph. However, instead of using the un-normalized version of the Laplacian, which was

presented in Chapter 2, the random walk version of the Laplacian is used. In [64], a discussion on both variants of the Laplacian in the context of image segmentation can be found. This random walk segmentation is done on a frame-by-frame and a pixel-by-pixel basis (super-voxels are often used nowadays) which explains the high time complexity of this method. Going along lines with constraining the fill front to the background region only, the search space, which will be the input to the hole filling step, is defined as the segmented background region only. This is the primary BoSP and it will be further downsized during hole filling.

The hole filling step starts at the background fill front of the hole boundary and propagates textural information to the interior of the hole region. To keep the depth information in the loop of calculating the filling priority, the authors proposed to add a term to Eqn. 3.12 whose purpose is to favor candidate patches lying on a homogeneous depth plane. Moreover, even though the depth map segmentation is done for each frame independently, the process of filling priority computation (Eqn. 3.12) takes a temporal, as well as spatial, neighbourhood into account. Their final BoSP is thus a space-time cuboid around every $\Psi_{p_b^*}$.

Now, the remaining task is to single out the best matching patch (to the patch centred at the hole boundary pixel of highest priority) in a space-time volume (cuboid) that is much smaller than the original space-time tube. This downsized problem stimulated the idea of carrying out another graph transduction operation. This time, the graph nodes are the patches within the cuboid, and the affinity function is the MSE between the patches embedded in a Gaussian kernel. Hence the output of this process can be interpreted as a probability, the closer the matching between a patch in the space-time cuboid and the hole patch under consideration, the higher the probability the former patch has. Following the graph transduction, each candidate patch in the cuboid is now assigned a corresponding matching probability; by minimizing the SSD between the candidate patches and $\Psi_{p_b^*}$, weighted by their corresponding probabilities, the best (filling) patch is then determined.

3.2.8 Joint color-depth stereoscopic inpainting

The term *stereoscopic inpainting* was used for the first time by Wang **et al.** [32]. Their framework was more focused on hole filling resulting from object removal, so they were

mainly concerned about stereoscopic image editing rather than novel view synthesis. This has left its imprint in their methodology as will be clarified below. It is worth mentioning that the importance of their work in the context of this thesis stems from the fact that stereoscopic image editing was the application that linked optimal hole filling frameworks [25], [23] to the scene (multi-view) completion problem [6].

Using their notation, the algorithm's pipeline expects three inputs, namely, the image pair $\{I_L, I_R\}$, their depth maps $\{D_L, D_R\}$ and their occlusion maps $\{O_L, O_R\}$. To get ready for the object removal step, they had to fill the occlusion regions first. This was done using the widely-used ***SegPln*** approach [4]. Basically, the image pair is segmented, and a depth plane is fitted to every super-pixel, based on the known depth values among its members. For segments where the known depth values are less than a threshold, the authors adopted a greedy algorithm that minimizes the objective function given by:

$$E(t,s) = E_{clr}(t,s) + \lambda_{adj} E_{adj}(t,s) + \lambda_{vis} E_{vis}(t,s) \tag{3.18}$$

where t refers to a segment with a known disparity plane and s refers to a segment where the number of pixels with known disparity values is less than a threshold. The terms E_{clr}, E_{adj} and E_{vis} refer to color, adjacency and visibility [83] constraints, and all λ's are empirically-set constants.

If a certain object is to be removed or relocated in the scene, the missing pixels in $\{I_L, I_R\}$, that were occupied by the removed object, and which are also missing in the occlusion-filled disparity maps $\{\bar{D}_L, \bar{D}_R\}$, are inpainted by adopting the following two-step process. The first step is the backbone of the algorithm, and it directly copies the half-occluded pixels from one frame to the other, using 3D warping. The rest of the missing pixels, in the image pair and their depth maps, are filled jointly and iteratively using a modified version of the method presented in [21]. A filling order that gives higher priority to linear structures, followed by textures, is adopted to each image independently, i.e. the color and depth are completed simultaneously but the left half-pair is completed independently of the right half-pair. To fill a patch in the hole, the algorithm proposed in [21] compared the color pattern of the patches and nominated the 'filler patch' using

the objective function given by

$$\underset{\mathcal{P}_k \in \bar{\Omega}}{\operatorname{argmin}} F(\mathcal{P}_k, \mathcal{P}_h) \quad | \quad \mathcal{P}_h \in \Omega. \tag{3.19}$$

While using the same objective function, the joint color-depth filling algorithm augmented the patch distance function with disparity and visibility-based terms to be

$$F(\mathcal{P}_k, \mathcal{P}_h) = F_{clr}(\mathcal{P}_k, \mathcal{P}_h) + F_{disp}(\mathcal{P}_k, \mathcal{P}_h) + F_{vis}(\mathcal{P}_k, \mathcal{P}_h). \tag{3.20}$$

Since the left and the right images are inpainted independently, mis-matches can occur leading to inconsistent stereoscopic completions. To eliminate these artifacts, the consistency check given by

$$|I_L(x, y) - I_R(x - D_l(x, y), y)| < \epsilon \tag{3.21a}$$

$$|I_R(x, y) - I_L(x + D_R(x, y), y)| < \epsilon, \tag{3.21b}$$

is done for the completed images after each iteration of the aforementioned completion procedure. Only the consistent pixels are marked as filled after each iteration. If a predetermined maximum number of iterations was reached with some pixel still failing the consistency check, they take their best label through all the iterations.

3.2.9 PatchMatch-based stereoscopic inpainting

The method proposed in [6] was the first attempt to import a state-of-the-art single image hole filling method to the problem of stereoscopically-consistent scene completion. They were motivated by the negative impacts of the direct copying of half-occluded pixels, which was adopted by [32], on the quality of completions. Particularly, they pointed out the degradation of completions near depth discontinuities. Even though it may be acceptable in image editing applications, it will be more felt while synthesizing novel views. Their method has involved two main contributions: a diffusion-based inpainting for holes in the disparity maps using a coupled system of partial differential equations, and a generalization of the PatchMatch algorithm for an efficient cross-view search of suitable candidate filling patches.

The first stage of the algorithm is similar to that in [32]; depth and occlusion maps are calculated and the occlusions are filled using the ***SegPln*** approach. Following the step of object manipulation in the stereo pair, which can be a removal or a re-placement, the algorithm starts to fill the resulting holes in the depth maps. The authors capitalize on the fact that disparity maps are textureless, mostly piece-wise constant or piece-wise smooth, which makes filling by simple diffusion a suitable candidate for the task. Towards this goal, they adopted the PDE given by

$$\frac{\partial D}{\partial t} = \nabla L \cdot \nabla_{\perp} D \tag{3.22}$$

where $L = \nabla^2 D$ is the Laplacian of the disparity map. Particularly, to maintain a stereoscopically-consistent filling of both disparity maps, they used a coupled pair of PDE's, for I_L and I_R, which are given by

$$\frac{\partial D_L}{\partial t} = \nabla L_L \cdot \nabla_{\perp} D_L + \lambda \rho_L \tag{3.23a}$$

$$\frac{\partial D_R}{\partial t} = \nabla L_R \cdot \nabla_{\perp} D_R + \lambda \rho_R \tag{3.23b}$$

where the first term in both equations is the inpainting term , similar to Eqn. 3.22, while the second term enforces the mutual consistency between the filled maps. Similar to [32], the consistency between the depth maps was enforced using the weak consistency constraint of [83].

With both disparity maps hole-free, they are now ready to be fed, as a constraint, to the last stage of the algorithm, namely, texture matching and synthesis. The authors proposed a texture-fill objective function that extends the coherence measure of [23] with regards to the following three aspects. First, the cross-view patch search, which refers to the procedure of including all the known patches in $\bar{\Omega}_L$ in addition to the patches in $\bar{\Omega}_R$ while filling Ω_R, and vice versa. Second, by including the depth information, in addition to the RGB pattern, during the assessment of the compatibility of two patches. Third, by requiring the final completion to minimize the distance between every hole patch and

its corresponding patch in the other half-pair. The overall objective function is given by

$$d_{tot}(\bar{\Omega}_R, \bar{\Omega}_L, \Omega_R, \Omega_L) = \sum_{\mathcal{P}_t \in \{\Omega_R \cup \Omega_L\}} \min_{\mathcal{P}_s \in \{\bar{\Omega}_R \cup \bar{\Omega}_L\}} d(\mathcal{P}_s, \mathcal{P}_t) + \sum_{\mathcal{P}_t \in \Omega_L} d(\mathcal{P}_t, C_{LR}(\mathcal{P}_t)) + \sum_{\mathcal{P}_t \in \Omega_R} d(\mathcal{P}_t, C_{RL}(t)) \tag{3.24}$$

where $d(\cdot)$ is the Euclidean norm, $\mathcal{P}_s$ and $\mathcal{P}_t$ are patches in the source and hole regions respectively, $C_{LR}(t)$ in the second term is the corresponding patch for t in the right image, and $C_{RL}(t)$ in the third term is the corresponding patch for t in the left image.

The framework proposed in [84] made some improvements over [6], chiefly a new method for quantifying the color-depth compatibility of the filling patches. Even though they demonstrated their results on a wider variety of stereo editing applications, compared to inpainting only in [6], both frameworks are very similar. For that reason, in addition to the fact that the two methods were not visually compared by [84], and that the authors were more focused on editing applications rather than view interpolation, I found it of superficial importance to give more details about their method.

3.2.10 Open problems in the literature

During my discussion on single image completion, I highlighted three state-of-the-art methods, each of which has its inherent limitations. The approach of Criminsi et al. for concentric hole filling is widely used, and has proved its efficiency in many contexts. However, being a greedy approach limits its application considerably. Depth-guided inpainting techniques that are based on that technique such as [38] will, obviously, suffer from its limitations. In addition to being slow, which is a characteristic of iterative approaches, the technique proposed in [81] also does not guarantee stereoscopic consistency.

Content-aware filling [23], [24], [25] has more favourable characteristics than Criminisi's approach [21] since it is an optimized approach, in addition to its speed, thanks to the PatchMatch correspondence algorithm. Nevertheless, its results are very sensitive to the initialization strategy and to the adopted optimization technique. Although it is very powerful, it has left room for improvement. The same conclusion can be drawn on the depth-guided inpainting techniques that adopt this approach. It is worth mentioning that the results reported in [6] are not as visually-plausible as the results obtained by

the method in [32], although they do have less noise at the depth discontinuities. This affirms that obtaining visually-plausible and stereoscopically-consistent completions is a goal that has not been yet attained using the aforementioned methods.

In the graph-based image completion techniques [28] and [26], the open problems in the methods found in the literature are two-fold. First, with regards to single image completion, it is highly desirable to find a hole filling method that:

- Considers a few possible shifts for hole completion, as opposed to [28], which contributes to the computational efficiency as well as the accuracy of completions.
- As opposed to [26], avoids the dependence on the dominant patch offsets in the known region, which may or may not suffice to complete the hole.

Second, the literature, so far, has not included a depth-guided inpainting technique, which can combine the speed, non-sensitivity to initialization, and the optimality of graph-based methods, while at the same time, guaranteeing stereoscopic consistency and exploiting the temporal+depth information.

The research that will be presented in Chapter 6 addresses the first couple of the aforementioned deficiencies starting from the base of the pyramid, the context of single image completion. In section A.7 in the appendix, I also show how to adopt my technique to deal with the very general context of dis-occlusion management for novel view synthesis.

Chapter 4

Towards Efficient Alpha Matting Using New Strategies for Trimap Sampling

In Chapter 3, two classes of challenges were discussed in regards to the application of natural image matting in the context of novel view synthesis. The first class of problems stems from the inefficiency of the matting techniques proposed in the literature so far. Specifically, the pipeline of the sampling-based techniques still requires improvement at each of its stages, namely, feature selection, pair pool shortlisting and best-pair nomination. In this chapter, I present new techniques for trimap sampling which address inefficiencies in pair pool shortlisting.

In sec. 3.1, I explained what constitutes 'a good pair' or 'good samples' in the quest of an accurate alpha value for every unknown pixel in the trimap. The main premise of the trimap sampling techniques proposed in this chapter is the benefit of adopting a sequential pair selection strategy, to bring good samples for every unknown pixel. The existing techniques, that were discussed in Chapter 3, shortlist the pool of all pairs in the trimap, then evaluate one whole pair at a time in the final stage of the pipeline. On the contrary, the proposed strategy aims at deciding a suitable half-pair first, and then searching for a suitable complement. To highlight the merits of this procedure, I devote the following section to discuss the motivation behind sequential pair selection, before proceeding to the discussion of the proposed techniques.

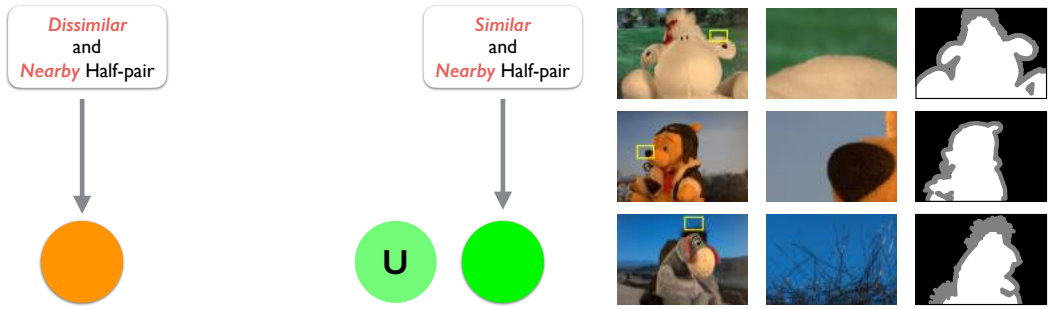
4.1 The motivation behind sequential pair selection

The presentation given in the previous Chapter about the existing matting techniques showed that good samples that best describe the color of an unknown pixel do not necessarily exist spatially-nearby to that unknown pixel. However, the matting bands (gray parts in the trimaps or pixels with unknown alpha values) in general exist at the discontinuities between a foreground object and the background. The colors of the pixels in these regions are the result of a local interaction between the foreground and the background. So, if the alpha maps are assumed to be sparse [70], i.e., most pixels are expected to be either fully foreground or fully background [36], at least one half-pair should lie nearby in space to the unknown pixel under consideration.

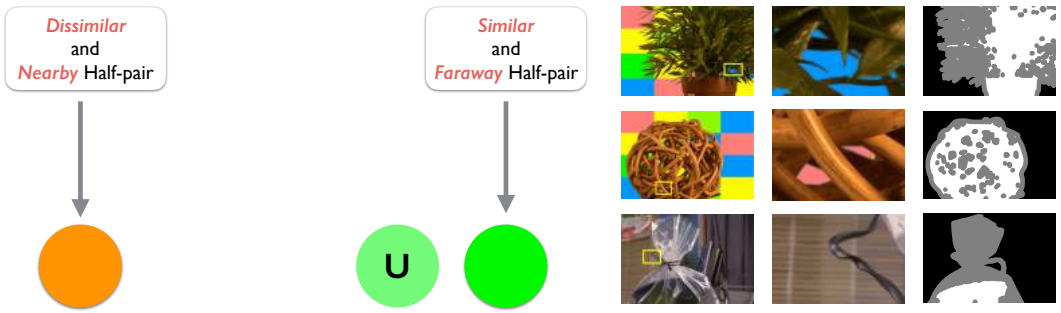
If an unknown pixel is mostly FG, throughout the rest of this document, the FG half-pair associated with it will be called the similar (in feature) half-pair, whether that similar half-pair is nearby in space or not. For that same pixel that is mostly foreground, the BG half-pair associated with it will be called the dissimilar half-pair, throughout the rest of this document. Since the matting bands are the result of local interaction between an object and its background, and the alpha maps are sparse, two main cases are encountered during trimap sampling. They are shown in Fig. 4.1.

In Fig. 4.1(a), three examples on the right depict the case illustrated by the diagram on the left, where the unknown pixels in the matting band find their FG/BG pair nearby; the similar and the dissimilar half-pairs exist spatially-close to the unknown pixels. For example, in the first row, the unknown pixels at the intersection between the ear of the white monkey and the grass in the background are expected to have colors that result from the linear-convex composition model adopted in natural image matting. In this sense, a half-pair from the nearby background and another half-pair from the near FG region (the ear of the white monkey) will constitute a *suitable pair*, since they will simultaneously minimize the cost function (Eqn. 3.2a) and yield an accurate alpha value.

Figure 4.1(b) shows three examples on the right that depict the case illsutrated by the diagram on the left. In this case, the unknown pixels find only one *suitable half-pair* close in space, while the other half-pair is faraway, even though the faraway half-pair is the similar one. For example, in the first row, the blue background pixels that are labelled as unknown in the trimap (on the right) will not find a similar half-pair nearby in space, since



(a) Case 1: The unknown pixels in the matting band have a similar half-pair and a dissimilar half-pair, both nearby in space. Three examples that depict this case are shown on the right.



(b) Case 2: The unknown pixels in the matting band have a dissimilar half-pair nearby in space, but the similar half-pair exists faraway in space, or does not exist at all as shown in the third row. Three examples that depict this case are shown on the right.

Figure 4.1: Cases encountered during trimap sampling. In part (a) and (b), a diagram illustrating the case is shown on the left. On the right, three examples are shown to depict each corresponding case. The first column is the original image with the region under consideration surrounded by a yellow rectangle. The second column shows those regions enlarged, and finally the third column shows one possible trimap for every image.

the nearest known background is yellow. However, assigning a blue pixel to a nearby FG pixel from the leaves region would suffice to calculate an accurate alpha value for the blue pixel (using the linear compositing model) if the matting algorithm succeeded to locate the faraway blue BG region, in the lower-left corner of the image. A pixel from the leaves is called a *suitable half-pair*, even though it is dissimilar, because it is quite distinctive from the background. It is worth mentioning that the approach presented in [73] would suffer in a case like the one shown in the first row of 4.1(b). This is because it augments the size of the bag of considered FG(or BG) samples as a function of the spatial distance between an unknown pixel and the nearest FG (or BG) to it. As the figure shows, the depicted unknown area is very close to a BG region, but its color is quite different from the BG samples required to calculate an accurate alpha value.

The third example shown on the right of Fig. 4.1(b) depicts another challenge. It is

a case where an opaque black ribbon (definite foreground) is all marked as grey in the trimap, however, there is no other known region in the trimap from the same ribbon. Dealing with this challenge requires:

- Adopting a better composition model that allows an unknown pixel under consideration to express itself if it did not find a pair that yields a low chromatic distortion. This challenge was mentioned among the open problems in the literature, at the end of the previous Chapter.

- Adopting a sampling strategy that assigns the unknown pixels of the ribbon to a suitable half-pair (the nearest BG in space, in this case). Otherwise, if the known background regions in the image include black objects (similar to the ribbon), the algorithm would select a half-pair from one of those black objects, which will yield a wrong alpha value (close to zero), only because that half-pair is a part of a FG/BG pair that minimizes the chromatic distortion.

The discussion so far has highlighted two problems related to the existing trimap sampling strategies, and an observation. The first problem is that suitable half-pairs may exist faraway in space. Second, the dependence on the spatial distance to decide the size of the bag of samples is not reliable, since it depends on the structure of the trimap, and would perform poorly with trimaps generated using wide brushes. The observation states that: at least one suitable half-pair should lie nearby in space.

The bag of samples gathered from the known FG and BG regions occupy a 2D space (Fig. 3.3). Choosing a suitable pair for every unknown pixel is a search process in that 2D space. Hence, the problem with considering all the FG and BG pixels in the trimap is primarily the computational burden associated with searching in a large space. However, there is another problem which may arise whenever the search space gets larger in size, namely, **the color ambiguity problem**. That is: the algorithm may occasionally pick a wrong pair (the term 'wrong pair' was defined in Chapter 3, after Eqn. 3.2a) only because it minimized the chromatic distortion. This problem was reported by the authors of [8]. It may get more severe with the approaches that rely on spatial distance to determine the number of considered samples, since some unknown pixels may lie faraway from both the BG and the FG regions, in which case there will be a large pool of pairs to choose from.

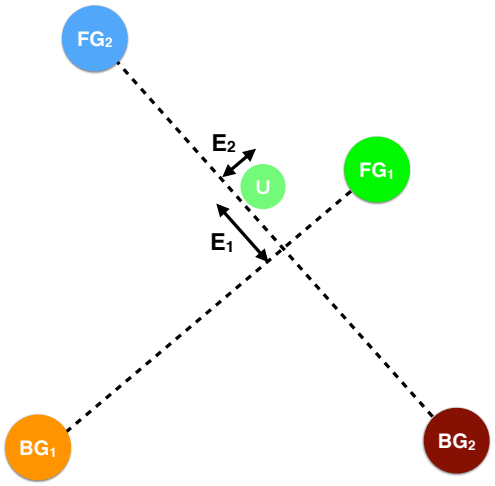


Figure 4.2: Considering a large pool of pairs may result in picking a wrong pair that minimizes the cost function (chromatic distortion) but yields a wrong alpha value. The case depicted in this figure shows that the pair FG_2/BG_2 would be picked instead of the pair FG_1/BG_1 because E_2 is less than E_1.

By recalling the discussion in the first point in sub-section 3.1.7 on the procedure for assessing the gathered pairs, the case shown in Fig. 4.2 may be encountered. The figure depicts a case where the algorithm picks the wrong pair because it minimized the cost function, i.e., if $E_2 < E_1$; the inefficiency here is attributed to the cost function and to the method adopted for trimap sampling and for constructing the shortlisted pair space (for every unknown pixel) as well. However, if the unknown pixel U was initially paired with FG_1, the probability of encountering that color ambiguity problem would be much lower (if not eliminated) even if a large number of BG half-pairs would be evaluated to see the best match for FG_1. The best match means the BG half-pair that minimizes the chromatic distortion, given FG_1. The same logic applies if the unknown pixel U was initially paired with BG_1, then a FG complement half-pair is sought for it. This limits the search space to a streak within the 2D pair space that is comprised of all the FG and BG pixels in the trimap. A *streak* is a stripe in that 2D pair space, and is comprised of all the pairs constructed from the FG (or BG) half-pair and all the available complement half-pairs gathered from the BG (FG) regions in the trimap.

To demonstrate the significance of the color ambiguity problem, I gathered a large pool of pairs that is still only a small subset of the known pixels in the trimap, and I calculated the alpha maps of a subset of the training dataset of [3] using that pool. The pairs were gathered using the following procedure:

1. I calculated the edge maps of the images using the method in [85].

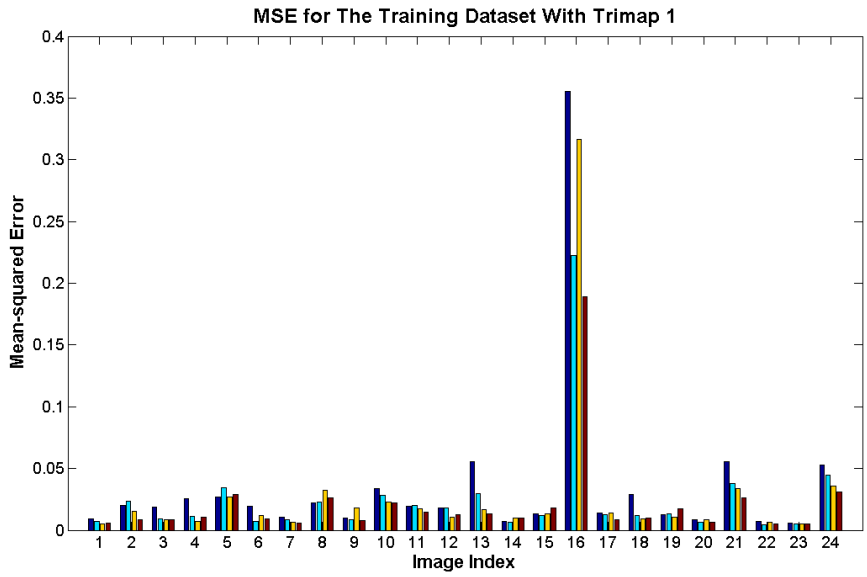
2. Then, I calculated the Simple Linear Iterative Clustering (SLIC) super-pixels [86, 87] (region size=5 and regularizer=0.1) of the images.

3. The final pool of FG and BG samples is then constructed as the mean color values of the spatially-nearest known super-pixel (identified by its centroid) to every edge super-pixel in the edge map. A 'known super-pixel' is a super-pixel that does not contain any unknown pixels and an 'edge super-pixel' is a super-pixel that contains one or more edge pixels. I used *Trimap1* only in my experiments, due to time and computational limitations.

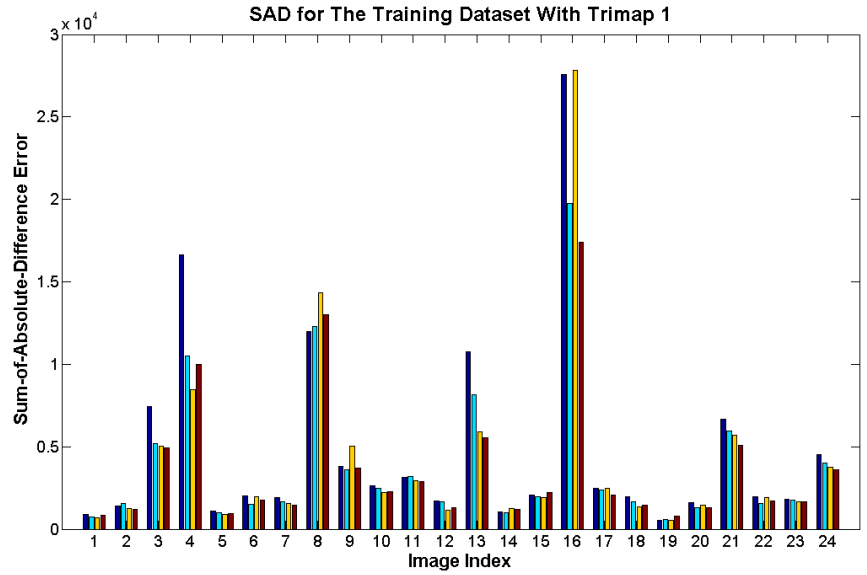
4. Afterwards, I looped over the unknown pixels and nominated, for each, the FG/BG pair that minimizes the chromatic distortion.

Since the pool of pairs is large, e.g., compared to the pool comprised of the spatially near super-pixels only, it could be thought that the alpha maps should be of high quality, i.e. relative to the ground truth, achieving lower MSE and SAD than the techniques in the literature. I plotted the MSE and SAD statistics of the above naive matting procedure in Fig. 4.3 for twenty-four images out of the twenty-seven images in the training dataset. I also compared the statistics I got with those of two other matting methods in the literature, namely, the method of [12] and the method of [73], in addition to the matting technique that will be discussed in sec. 4.3. For MSE, that is shown in Fig. 4.3(a), the naive matting method achieved the highest MSE in 13 images out of 24 images. For SAD, that is shown in Fig. 4.3(b), the naive matting method achieved the highest SAD in 15 images out of 24 images. These statistics show the significance of the color ambiguity problem, and also show that the size of the considered pool of pairs is not necessarily proportional to the accuracy of the alpha maps.

For an unknown pixel, given that at least one half-pair should lie nearby: Deciding a suitable half-pair to start from, and then searching for its best match, will be called the **sequential pair selection** sampling strategy, throughout this document. In the rest of this Chapter, two approaches will be presented to realize the sequential pair selection, taking into consideration the two open problems in the literature, namely, the faraway suitable



(a)



(b)

Figure 4.3: Statistics showing the (a)MSE and (b)SAD of the alpha maps for a matting technique that considers all FG and BG samples near image edges (a naive matting technique). The performance of the naive technique (blue bar) was compared with other matting techniques in the literature, namely the method of [12] (yellow bar) and the method of [73] (red bar), in addition to the matting technique that will be discussed in sec. 4.3 (cyan bar). The statistics demonstrate the significance of the color ambiguity problem, and show that the size of the considered pool of pairs is not necessarily proportional to the accuracy of the alpha maps. Please see text for more details.

half-pairs and the unreliability of the spatial distance as a factor to augment/downsize the shortlisted pair space.

Last but not least, this section is meant to stress the fact that the presented methods target a real problem that has not been addressed sufficiently in the literature so far. In the light of a previous discussion in 3.1.7, the small size of the standard dataset and/or the frequency of occurrence of the discussed challenges in the testing dataset (which decides the rank of any algorithm on the benchmark) do not belittle the importance of the addressed challenges.

4.2 Sequential pair selection by quantifying overlap between color distributions

The algorithm starts by an over-segmentation step for the input image and an expansion step for the trimap. Throughout this document, the terms 'region', 'segment' and 'super-pixel' will be used interchangeably to refer to an area in an image with coherency between the features of its constituent pixels – the output of the segmentation step. To segment the image, I used the publicly-available Edison implementation[1] of mean-shift [88] segmentation. The trimap expansion was done using the same condition adopted in [73] and is given by

$$(D(p, F_i) < E_{threshold}) \wedge (\|I_p - I_{F_i}\| \leq (C_{threshold} - D(p, F_i))), \tag{4.1}$$

which means that an unknown pixel p will be considered as a definite FG if the Euclidean distance $D(p, F_i)$ between it and another pixel $F_i \in \mathbf{F}$ ($\mathbf{F}$ is all the definite FG pixels in the trimap) is less than $E_{threshold}$ and if a norm of their chromatic difference is less than $C_{threshold} - D(p, F_i)$. $E_{threshold}$ and $C_{threshold}$ are both constants in the spatial domain and the color coordinate system respectively, and they were empirically set to 9. The same condition is applied for comparing the unknown pixels with the background pixels.

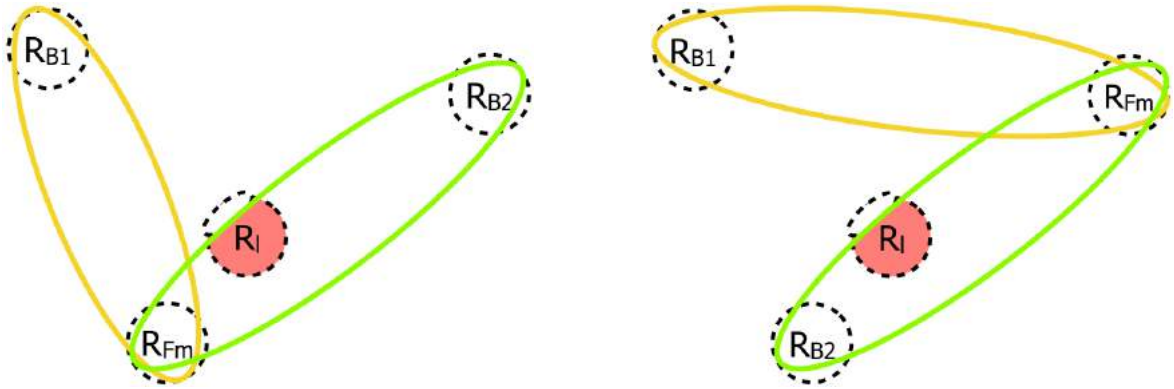
In the second stage of the algorithm, I determine whether the spatially-nearest FG super-pixel (identified by its centroid) to every unknown super-pixel is a suitable half-pair

[1] http://coewww.rutgers.edu/riul/research/code/EDISON/

for it or not. It is worth mentioning that, in the following paragraphs, the procedure of the algorithm will be explained for the case where the known super-pixel belongs to the foreground; however, the same logic is applicable and the same procedure was followed for the BG super-pixels as well. In other words, I also checked for every unknown super-pixel if the spatially-nearest BG super-pixel to it is a suitable half-pair or not.

To determine, for an unknown super-pixel, if the near FG region $\mathbf{R_{F_m}}$ is a suitable half-pair or not, a variety of approaches could have been taken. For example, I could have built a parametric model (GMM [10] for example) for the color distribution of that near FG region and check its suitability based on the likelihood that the colors of the members of that unknown super-pixel were generated by such a distribution. To lower the computational burden, I have adopted a different approach inspired by the work in [73].

By recalling the illustration provided in Fig. 2.1, there should be an overlap between an unknown super-pixel and the color distribution represented by the union of the pixels in the FG super-pixel and the BG super-pixel that best-describe the colors of the members of that unknown pixel. An illustration for this claim is shown in Fig. 4.4, where the unknown region, the spatially-nearest FG region to it, and two arbitrary BG super-pixels in the image are given the symbols $\mathbf{R_l}$, $\mathbf{R_{F_m}}$, $\mathbf{R_{B_1}}$ and $\mathbf{R_{B_2}}$ respectively. The dashed circles in that illustration symbolizes the color distribution of each region. The orange oval and the green oval symbolize the distribution represented by the union of the pixels in $\mathbf{R_{F_m}}$ and $\mathbf{R_{B_1}}$, and $\mathbf{R_{F_m}}$ and $\mathbf{R_{B_2}}$ respectively. If $\mathbf{R_{F_m}}$ is a suitable half-pair for $\mathbf{R_l}$, at least one union distribution among those that exist between $\mathbf{R_{F_m}}$ and all the BG super-pixels in the image should be overlapping with the distribution of $\mathbf{R_l}$. The amount of overlap is shown in red in Fig. 4.4. It is worth mentioning that the distributions depicted as green and orange ovals in Fig. 4.4 are not assumed to be Gaussian; the figure is just a schematic. In fact, the formulation of quantifying the overlap between the union distributions and an unknown region under consideration does not put a prior on the type of those union distributions. The premise of this formulation is that whatever the shape of these union distributions, the overlap of an unknown pixel with a union distribution containing a good half-pair for it will be larger than another distribution that does not include a good half-pair for it.



(a) (b)

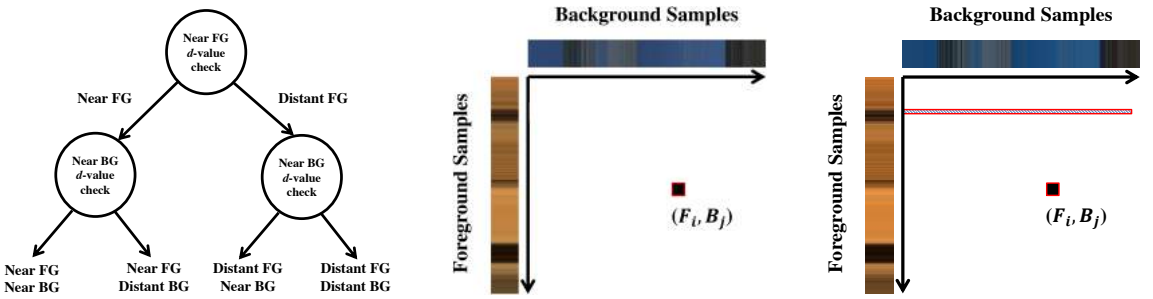
Figure 4.4: The color distributions of an unknown region ($\mathbf{R_l}$), a near FG region ($\mathbf{R_{F_m}}$) and two arbitrary BG regions ($\mathbf{R_{B_1}}$ and $\mathbf{R_{B_2}}$) in the image. The figure illustrates how a FG region is decided to be a suitable half-pair for an unknown region, whether it is the similar half-pair (a) or the dissimilar one (b). Please see text for details.

Assume that the output of the initial over-segmentation step is the following: M FG regions each with the symbol $\mathbf{R_{F_i}}$ where $i = 1, \ldots, m, \ldots, M$, N BG regions each with the symbol $\mathbf{R_{B_j}}$ where $j = 1, \ldots, n, \ldots, N$ and L unknown regions each with the symbol $\mathbf{R_k}$ where $k = 1, \ldots, l, \ldots, L$. To decide the suitability of $\mathbf{R_{F_m}}$ for the particular unknown region $\mathbf{R_l}$, I construct N distributions, each of which is given the symbol $\mathbf{R_{U_j}}$, and contains the union of the pixels in $\mathbf{R_{F_m}}$ and one of the BG regions $\mathbf{R_{B_j}}$. I then calculate N Cohen's d-values between the color distribution of $\mathbf{R_l}$ and the color distribution of each of the N $\mathbf{R_{U_j}}$s, to quantify the overlap between those distributions. If any of the calculated d-values is less than $d_{threshold}$ (set to 2 in my experiments), FG samples will be collected from $\mathbf{R_{F_m}}$ for all the pixels in $\mathbf{R_l}$. The value $d_{threshold}$ was set empirically using exemplar cases.

In my implementation, I did not calculate all the N d-values. I considered only five of them, the five distributions whose mean is the nearest (in the feature space using the Euclidean distance) to the mean of $\mathbf{R_l}$. The d-value is given by

$$d(\mathbf{R_{U_j}}, \mathbf{R_l}) = \frac{\mu_{\mathbf{R_{U_j}}} - \mu_{\mathbf{R_l}}}{\sqrt{\frac{(N_{\mathbf{R_l}}-1)\sigma^2_{\mathbf{R_l}} + (N_{\mathbf{R_{U_j}}}-1)\sigma^2_{\mathbf{R_{U_j}}}}{N_{\mathbf{R_l}} + N_{\mathbf{R_{U_j}}} - 2}}}, \tag{4.2}$$

where μ, σ and N are the mean, the standard deviation and the cardinality respectively. An unknown region could be spatially-near to more than one FG region. The d-value was



(a) A Decision tree (b) A Pair space (c) A streak in the pair space

Figure 4.5: (The decision tree that is followed for every unknown region. It indicates how the FG and BG samples will be collected. Figure (b) depicts a pair space with one point representing a single FG/BG pair, and (c) depicts a streak in the same pair space. The term 'streak' is defined during the discussion of Fig. 4.2.

thus measured with the near h FG regions struck by $H = 8$ rays in an almost-similar approach to [7], where $h \leq H$. It is important to mention that the rays emanated in my algorithm are meant to collect local information, which is the task they usually excel in.

For the unknown region under consideration $\mathbf{R_l}$, and for every unknown region $\mathbf{R_k}$, the decision tree shown in Fig. 4.5(a) is then followed leading to one of four possible leaves. The first leaf from the left 'Near FG - Near BG' means that 'S' samples will be taken from the particular near FG region $\mathbf{R_{F_m}}$ and the particular near BG region $\mathbf{R_{B_n}}$. The second leaf 'Near FG - Distant BG' means that FG samples exist nearby while non-local BG samples will be collected from a pool containing all the known BG pixels in the trimap using the *k-nearest neighbors* (KNN) algorithm. The third leaf 'Distant FG - Near BG' means that BG samples exist nearby while non-local FG samples will be collected from a pool containing all the known FG pixels in the trimap using the KNN algorithm. If an unknown region, occasionally, reached the fourth leaf, I check both nearby FG samples with non-local BG samples and vice versa; this is a computational bottleneck in the algorithm.

The Matlab function `knnsearch` was used for seeking the nearest neighbors. For the second and the third leaves in the decision tree, the KNN search was carried out for each pixel in the unknown region under consideration $\mathbf{R_l}$. I have followed previous work in the literature [89] and used the feature vector $RGBXY$ for the KNN search, where RGB is the pixel color and XY is the pixel coordinate. Nevertheless, I observed that tying the color feature with the spatial coordinates results in spatially near samples to

the unknown pixel; sometimes, this over-influences and compromises the color part of the feature vector. I present more insightful statistics that justify this observation amid my discussion on the second proposed matting technique in the next section. Hence, I collected half of the samples using the vector $RGBXY$ and the other half with the RGB vector only, giving the opportunity for spatially-far samples to be nominated if they yield low chromatic distortion. The gathered FG/BG pairs were assessed using an objective function similar to Eqn. 3.4.

The performance of this algorithm was evaluated according to the online matting benchmark [3]. Even though it succeeded, at the time of its proposal, to achieve a few top scores in the benchmark, i.e. to attain the least MSE or SAD in a particular trimap for a particular image, I consider the contribution of this algorithm is conceptual, rather than practical. In other words, I proposed a new idea which addresses an open problem with regards to efficient trimap sampling; however, the final alpha maps are not good enough to put the proposed method on-par with the state-of-the-art techniques. This is why I preferred to devote more space to analyze the performance of the other matting methods proposed by this thesis, which will be discussed in the rest of Chapter 4 and Chapter 5. At the end of chapter 5, and in chapter 8 as well, I compare the performance of all the proposed matting methods, with respect to one another and with respect to the SoA, and I present some recommendations based on these comparisons.

Since I determine a suitable starting point along one dimension in the pair space, and then search for a suitable complement, my shortlisted pair space is limited to a streak in the whole pair space, such as the one that is depicted in Fig. 4.5(c). I argue that this reduces the influence of the color ambiguity problem. Moreover, I did not augment my gathered samples based on the distance of the unknown pixel from the boundary of the trimap, rather, I check the suitability of the near regions using color-space analysis, and if they are not suitable for sampling, the unknown region will be assigned to a distant FG (or a distant BG), and its FG (or BG) samples will be collected non-locally.

Nevertheless, the current implementation of the presented method suffers from critical disadvantages. Due to the adopted feature vector, the RGB feature, the algorithm is sensitive to the case where an overlap between the color distributions of FG and BG regions exists. Moreover, even though the size of the pair pool has been reduced to a

streak in the pair space, the size of the shortlisted pool-of-pairs is large compared to the SoA techniques. The alpha maps obtained using this method can be found in sec. A.5.

4.3 Detecting the best half-pair using graph transduction

4.3.1 Choosing delegates for image regions

Throughout the literature of image matting, the shortlisted pair space was used to calculate the alpha of every unknown pixel. Spatially-nearby unknown pixels were allowed to share the same pair space, however, their inter-dependencies were not leveraged to cut back the number of computations during the best-pair selection process. Along parallel lines, I have observed that the combination of the spatial coordinates and the color feature (RGBXY or HSVXY for every pixel), that is widely used in the literature especially in propagation-based matting, results in local (close in the spatial domain) neighbours if used for a KNN search. In fact, the majority of the neighbours found for every pixel lie in its same super-pixel, i.e., with RGBXY as the feature descriptor, long-range correspondences between pixels in the image domain (or faraway pixels in the terminology of [72]) can not be established. In the following discussion, I will use the terms 'region', 'segment' and 'super-pixel' and the abbreviation SP interchangeably.

To check how local the samples collected using the space-color feature are, I calculated the Simple Linear Iterative Clustering (SLIC) super-pixels [86, 87] (region size=20 and regularizer=1) of all the training dataset of [3], then I calculated KNN ($K = 15$) for every pixel, and finally I counted the 'true positives' and the 'false positives'. The former is the case where a certain pixel falls in the same super-pixel with all its neighbours. The latter is that case where one or more of the neighbours of a certain pixel does not share the same super-pixel with it. These statistics are shown in Fig. 4.6 and it shows that almost 90% of the pixels have all their KNNs inside their own SLIC pixel; this is consistent if the statistics are calculated for all the pixels in the image or the mixed pixels only (recall that we have the ground truth of the training dataset [3]). I also calculated the same statistics for the case where the size of the super-pixel is reduced to be of radius=10 pixels instead

of 20; I got almost similar statistics as depicted in Fig. 4.7(a). The same pattern can not be observed if the feature used for determining the neighbours is RGB instead of RGBXY as shown in Fig. 4.7(b).

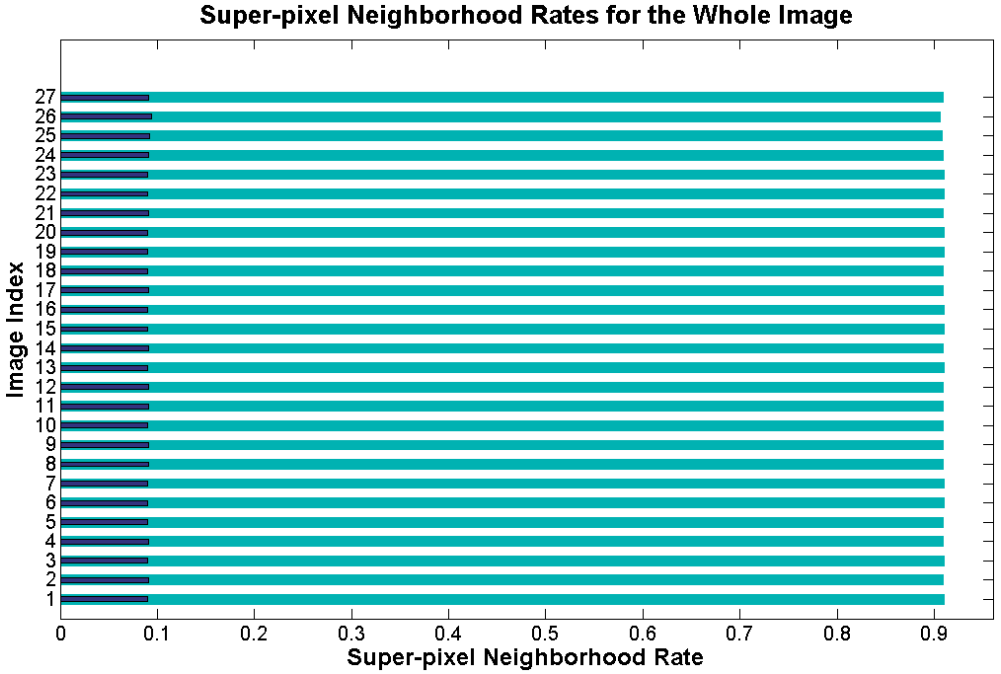
In the light of these statistics, I propose to nominate a few delegates from every super-pixel. If a certain super-pixel is known according to the trimap, we will consider its delegates only (rather than all its members) as FG or BG proposals. Similarly, if it is unknown, the best-pair nomination process (the last stage in the pipeline of sampling-based matting) will be carried out only for its delegates rather than all its members. Algorithm 1 shows the pseudocode of my proposed algorithm for the nomination of delegates. For every super-pixel, I calculate the mean RGB color vector, then I sort the members of the region according to their deviation from the mean. The whole range of deviation-from-mean is then divided into N intervals (or subgroups, $N = 10$ in my experiments) of equal width and S_{S_g} samples are picked evenly from every subgroup as a function of a budget B that is given by

$$S_{S_g} = \left\lceil B \times \frac{N_{S_g}}{N_{Tot}} \times \frac{(MAD)_{S_g}}{(MAD)_{Tot}} \right\rceil, \tag{4.3}$$

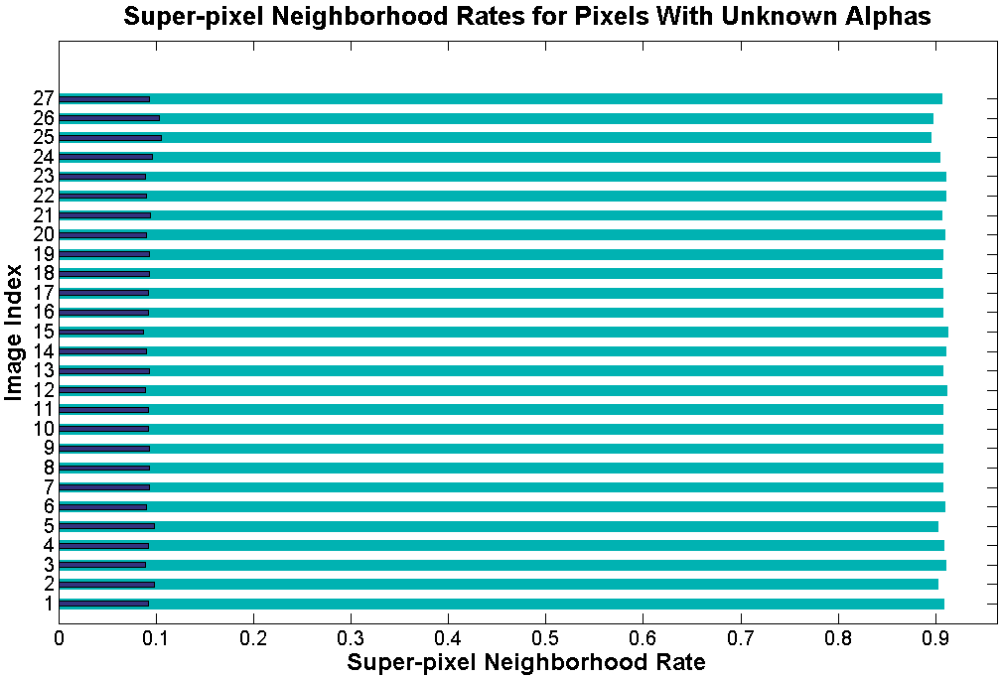
where B is the budget ($B = 40$ in my experiments), N_{S_g} and N_{Tot} are the number of members in the subgroup and the whole super-pixel respectively, $(MAD)_{S_g}$ is the mean absolute deviation from the mean in the subgroup and $(MAD)_{Tot}$ is the mean absolute deviation from the mean in the whole super-pixel. Using this heuristic, the percentage of delegates for the images in the training dataset in [3] is around 10% of the total number of pixels in the image. Once the delegates are determined for every region, I calculate a weighting matrix which indicates how the rest of a region's members can be obtained from its delegates. This matrix is calculated using the same procedure of [60] which was discussed in Chapter 2. It can be expressed as:

$$W \coloneqq \underset{w_{ij}}{\operatorname{argmin}} \sum_{i=1}^{N_{Tot}} \left| \vec{X}_i - \sum_{j=1}^{K} w_{ij} \vec{X}_j \right|^2 \quad \textbf{s.t.} \quad \sum_{j=1}^{K} w_{ij} = 1, \tag{4.4}$$

where N_{Tot} is the total number of pixels in a super-pixel, K is the number of delegates and $\vec{X}_i$ (and $\vec{X}_j$) is a pixel's feature vector. This procedure simply applies the local linearity principle within every SLIC super-pixel. Once the alpha values of the delegates

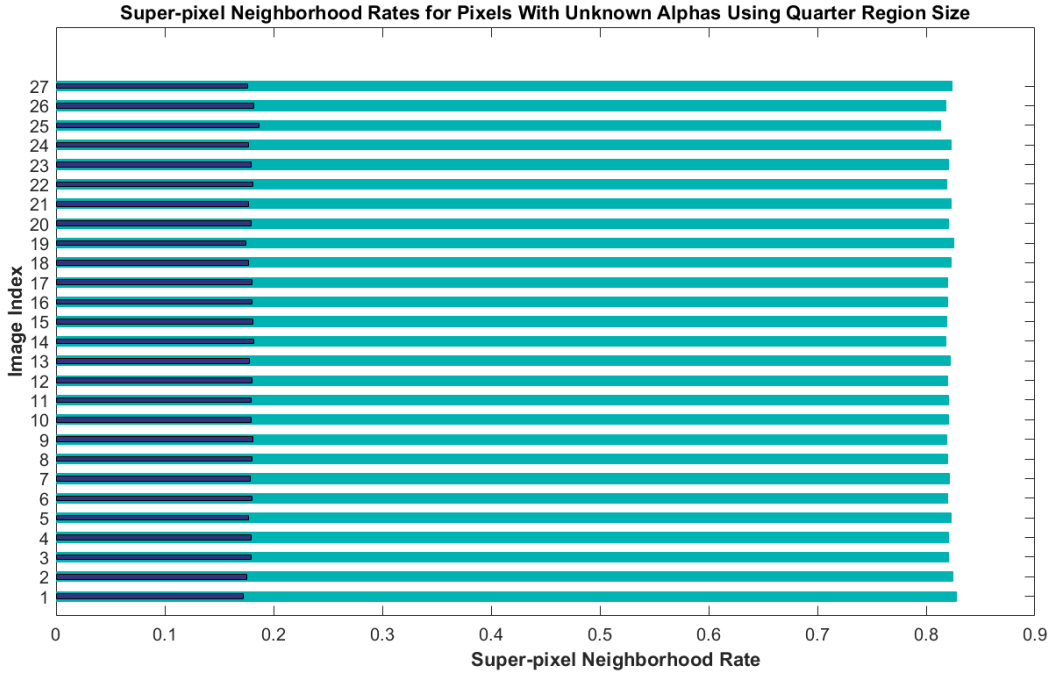


(a)

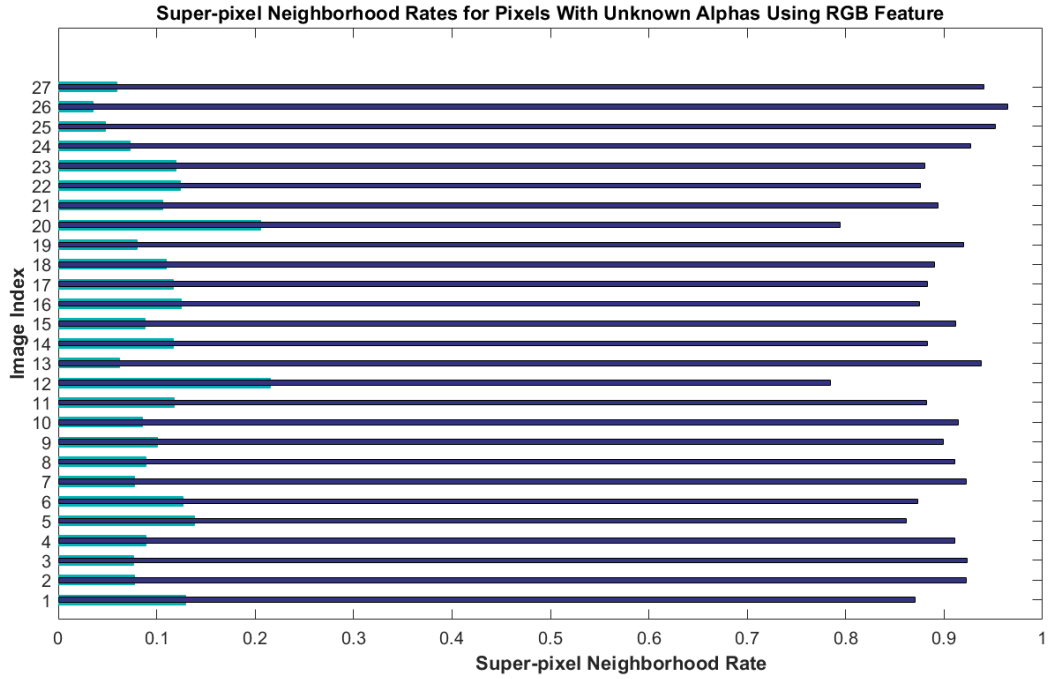


(b)

Figure 4.6: SLIC super-pixel neighbourhood rates for the pixels in the whole image (a), and the mixed pixels only (b). The cyan bar indicates the percentage of the true positives (same super-pixel neighbourhood rate) and the blue bar indicates the percentage of the false positives (different super-pixel neighbourhood rate).



(a)



(b)

Figure 4.7: (a) SLIC super-pixel neighbourhood rates for the mixed pixels in the image using the feature RGBXY when the super-pixel radius is reduced to 10 pixels, (b) and the neighbourhood rates for the mixed pixels in the image using the feature RGB only. The cyan bar indicates the percentage of the true positives (same super-pixel neighbourhood rate) and the blue bar indicates the percentage of the false positives (different super-pixel neighbourhood rate).

of an unknown region are calculated, the alpha values of the rest of the members can be computed accordingly.

Algorithm 1 A heuristic to nominate delegates from every SLIC super-pixel

1: **procedure** GETDELEGATES(*Image, SLIC Super-pixels, N, Budget,* S_{S_g})
2: *loop over SLIC segments*:
3: Calculate super-pixel mean
4: Calculate deviation from mean for all members
5: Sort members based on their deviation from mean
6: Calculate deviation range and divide it into N intervals
7: *loop over N intervals*:
8: Pick S_{Sg} samples from each interval as a function of a budget B
9: **end procedure**

4.3.2 Determining good half-pairs for unknown super-pixels

The process of determining a suitable half-pair for every unknown pixel starts by calculating the cartoon-texture decomposition [46] of the input image[2]. The decomposition was computed at scale = 5. Following their nomination, the delegates of the super-pixels are represented by their cartoon-texture decomposition, and I then solved a binary graph transduction problem, akin to Duchenne **et al.** [64], to find the best half-pair (a FG or a BG super-pixel) for every unknown SP. This graph is shown in Fig. 4.8. The cartoon-texture decomposition was used in this stage in lieu of the color to avoid the ambiguity that may arise if the color distributions overlap.

I start by looping over the unknown super-pixels in the image, and determine those that are not farther than 50 pixels away from any of the known FG and BG super-pixels. Every unknown SP will then be offered one of its nearby known super-pixels (proposals) at a time. To decide whether the unknown SP under consideration accepts a proposal or not, I adopt the following procedure. Considering an unknown SP and one of its nearby proposals, their delegates are used to build a graph. The entries of the Laplacian matrix

[2]I used the publicly-available code at: http://www.ipol.im/pub/art/2011/blmv_ct/

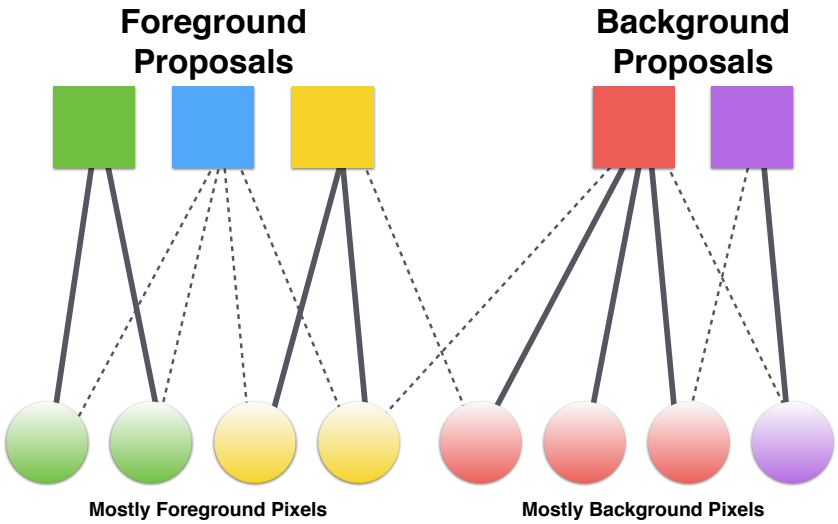


Figure 4.8: An illustration of the graph model that is used in the transduction process to determine the best half-pair for every unknown SLIC super-pixel. The proposals are depicted as squares while the unknown super-pixels are depicted as circles. The affinities between the proposals and the unknown regions are depicted by the lines connecting them; the solid lines symbolize high affinity (or appearance similarity) while the dashed lines symbolize low affinities. An unknown super-pixel is expected to accept a proposal that is similar in appearance to it.

of this graph are calculated using the kernel function given by

$$k(X_i, X_j) = \frac{\widetilde{k}(X_i, X_j)}{[\widetilde{d}(X_i)\ \widetilde{d}(X_j)]^{\lambda}} \qquad \textbf{where} \tag{4.5a}$$

$$\widetilde{k}(X_i, X_j) = \mathrm{e}^{-\frac{\|X_i - X_j\|^2}{2\sigma^2}} \qquad \textbf{and} \tag{4.5b}$$

$$\widetilde{d}(X_i) = \sum_{j=1}^{n} \widetilde{k}(X_i, X_j). \tag{4.5c}$$

In the above equations, σ is the standard deviation of the Gaussian kernel and is fixed to the value of 20 throughout the experiments, n is the dimension of the Laplacian (square) matrix and X_i (or X_j) are defined as $X_i \coloneqq \{C_R, C_G, C_B, T_R, T_G, T_B\}$ which is the RGB cartoon-texture feature vector. By recalling the discussion in Chapter 2, a graph transduction is carried out by minimizing an objective function and solving a corresponding

linear system which are given, respectively, by

$$\min_{F \in \mathbb{R}^n} (F - Y)^T C (F - Y) + F^T L\, F, \tag{4.6a}$$

$$(L + C)\, F = C\, Y, \tag{4.6b}$$

where C is the diagonal $n \times n$ matrix in which the i^{th} diagonal element is c_i (c_i=10 in my experiments) for a labelled point, and 0 for a test point, Y is the n-dimensional vector in which the i^{th} element is Y_i for a labelled point, and 0 for a test point. After obtaining F, I threshold it to 0 or 1. If at least 30% of the delegates in the unknown pixel accept the proposal, it will assigned as its best half pair, i.e. the number of ones in F should be at least 30% of its length. An unknown SP may accept more than one of its nearby proposals, so the loop over the proposals is interrupted once an unknown SP accepts a proposal. In other words, an unknown SP is assigned to the first known SP (either FG or BG) it accepts. This is mainly determined by the closeness of their cartoon-texture features which is depicted by the weight of the links between the nodes in Fig. 4.8 between the proposals (squares) and the unknown delegates (circles). Obviously, an unknown pixel is expected to accept a proposal if they have quite similar appearance.

Since we considered the delegates of every SP only, rather than all its members while constructing the Laplacian, the size of the Laplacian matrix is equal to the sum of the number of the foreground (or background) delegates *plus* the number of delegates of the unknown region under consideration. Thus, I end up working with small matrices. I loop over the unknown regions and the size of the Laplacian does not exceed 200×200 entries in every loop. Moreover, once an unknown region accepts a proposed super-pixel, it is not offered more proposals. Both of these reasons contribute to the computational efficiency of the matte calculation. There is a rare case where an unknown SP contains known FG pixels, known BG pixels and a few unknown pixels; it is still considered as an unknown SP. Such a SP would accept a nearby FG and a nearby BG, and in this case, I gave the priority to the nearby FG super-pixel in my implementation.

The unknown SPs that are farther than 50 pixels away from the known regions will be considered in the following stage of the algorithm. I continued the best half-pair computation in propagation fashion. The unknown super-pixels that have been already assigned a

half-pair will represent the 'labels' themselves, and an exactly similar transduction-based procedure will be followed in an attempt to assign an already-paired unknown SP to one or more of the remaining unknown SPs that are not farther than 50 pixels away from it. This propagation stops once all the unknown SPs are paired or when the propagation itself fails to proceed inside the matting band (the gray band in the trimap); this may happen if an unknown pixel refused all the available proposals.

I had two strategies to deal with those remaining unknown SPs, which could not be paired during half-pair propagation. The first strategy takes into consideration that an unknown SP may be comprised of a mixture of unknown pixels and known, yet dissimilar, pixels. This happen often in the case of an isolated BG region for example. In this case, the isolated BG will not accept the nearby FG proposals through transduction because they are dissimilar; the nearby FG pixels would constitute a good half-pair, though, because they are quite distinctive from the isolated BG region (recall the discussion in the beginning of this Chapter). To cope with this case, if a remaining unknown SP contains known FG pixels, it will be assigned to itself as the best half-pair, and similarly if it contains known BG pixels.

The last strategy to deal with those remaining unknown SPs is a voting procedure. I calculate the spatially-nearest seven paired unknown SPs, and I retrieve their half-pairs. If the majority of their half-pairs are FG labels, the remaining unknown SP under consideration is assigned the half-pair of the spatially-nearest paired unknown SP among those which were assigned to FG half-pairs. Similarly, if the majority of their half-pairs are BG labels, the remaining unknown SP under consideration is assigned the half-pair of the spatially-nearest paired unknown SP among those which were assigned to BG half-pairs.

It is worth mentioning that in the graph in Fig. 4.8, the unknown super-pixels (circles) have not been connected because the transduction was done for every unknown super-pixel independently. A question that may surface at this point is: should not the similar super-pixels share the same best half-pair? Should not the circles in the graph be connected to guarantee this assumption? My approach actually took this assumption into consideration, but implemented it differently. I was keen on solving a small graph problem in every loop, and then let the neighbouring super-pixels share their best half-pairs.

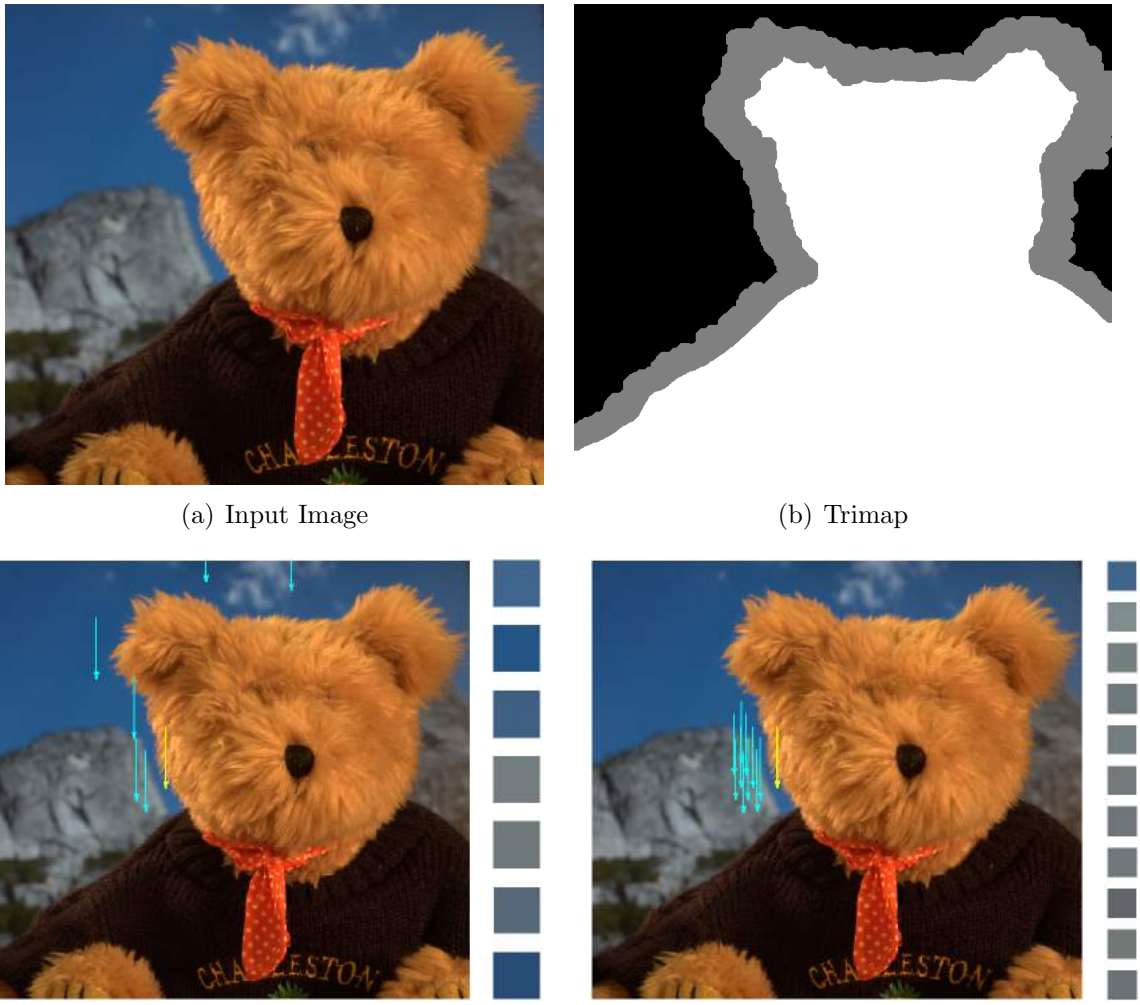
Half-pair sharing is done as follows: I calculate the mean color-cartoon-texture feature

for all the unknown SPs in the image, in addition to those of the BG labels in the image. I also determine which of the unknown SPs has more than or equal 25% of its constituent pixels already known in the trimap; I give them the symbol U_{SN} – the unknown SPs with significant number of known pixels. Every unknown SP is then allowed to share its half pair with the five spatially-nearest SPs, the five most similar SPs according to the mean color-cartoon-texture feature and the five most similar BG labels to account for isolated BGs. Finally, every unknown SP is also given access to the known pixels in the five most similar SPs among U_{SN}, if any. To determine the most similar SP among U_{SN}, I used the joint color-cartoon-texture-xy feature.

The merit of the proposed technique with regards to gathering good half-pairs is demonstrated using the two following methods. First, I picked a challenging case, in the training dataset of the matting benchmark, where the classical technique for gathering spatially near samples fails. I compared the gathered BG half-pairs with that classical technique with those gathered using the proposed method. This is shown in Fig. 4.9, and I considered a maximum of 10 gathered half-pairs to plot this figure. In part (c) and (d) of Fig. 4.9, the unknown super-pixel under consideration is pointed to by a yellow arrow, while its gathered half-pairs from the BG regions are pointed to by cyan arrows. The former shows the gathered half-pairs using the proposed method while the latter shows the spatially-nearest BG half-pairs. On the right of part (c), I show the mean color values of the gathered half-pairs as a palette. The upper-most square in the palette is the mean of the unknown region under consideration, while the rest represent the mean color values of its corresponding half-pairs. The same information is shown in part (d). It is clear that the proposed method brought more similar samples to the unknown region if compared with the classical approach.

The second method that was adopted to demonstrate the merit of the proposed technique with regards to gathering good half-pairs aims at assessing the goodness of the gathered half-pairs objectively. I carried out the following experiment on the whole dataset of the matting benchmark (27 images with Trimap 1):

1. Calculate the SLIC pixels of the input image with region size = 5 and regularizer = 0.1. Note that this over-segmentation step uses a smaller region size than the one adopted in our half-pair computation. This over-segmentation is completely



(a) Input Image (b) Trimap

(c) Half-pairs With Graph Transduction (d) Spatially-near Half-pairs

Figure 4.9: A demonstration of the benefit of using the proposed algorithm to determine a suitable half-pair for every unknown super-pixel. In (c), the unknown super-pixel under consideration is pointed to by a yellow arrow, while its gathered half-pairs from BG are pointed to by cyan arrows. Part (d) shows the case where the gathered half-pairs are the spatially-near half-pairs only, which is the prevalent method in the literature. On the right of part (c), I show the mean color values of the gathered half-pairs as a palette. The upper-most square in the palette is the mean of the unknown region under consideration, while the rest represent the mean color values of its corresponding half-pairs. The same information is shown in part (d). It is clear that the proposed method brought more similar samples to the unknown region if compared with the classical approach. It is worth mentioning that I gathered similar number of half-pairs in both cases; however, some of the gathered half-pairs in part (c) are repeated.

independent of the main algorithm, and that region size was used only in this experiment for the purpose explained below.

2. According to this over-segmentation, determine the FG labels and the BG labels, and compute their means. These means will constitute my whole dictionary FG samples and BG samples.

3. Given the data of best half-pair computation, loop over the unknown labels in the image, and for each, determine its computed best half-pair. I will assume that the best half-pair was a FG label, but the same logic holds if it was a BG label.

4. Determine the color feature of the delegates of the best half-pair.

5. Loop over the delegates of the unknown pixel.

6. For every delegate, we have its ground-truth alpha value (in the training dataset), and we have the features of the delegates of its best half-pair.

7. Assuming that the best half-pair is a FG label, and recalling Eqn. 3.2a and given the ground-truth alpha value, calculate an ideal 'other half-pair'; the other half-pair is BG pixels in our case since the half-pair is a FG. Actually, we will have a number of ideal other half-pairs that is equal to the number of the delegates of the FG half-pair.

8. Given the BG dictionary calculated from the over-segmentation step, calculate the minimum Euclidean distance between the color feature of the computed ideal half-pairs and that dictionary.

9. For all the delegates of the unknown labels, and for all the labels, compute the mean minimum Euclidean distance in the whole image.

For a particular image, if the computed half-pairs are good, its mean minimum Euclidean distance should be close to zero. I calculated that value (the mean minimum Euclidean distance) for each of the 27 images in the training dataset, then subtract them from 1 to be able to plot a presentable bar chart. This bar chart is shown in Fig. 4.10. Ideally, all bars should be close to 1; however, the results are reasonable in

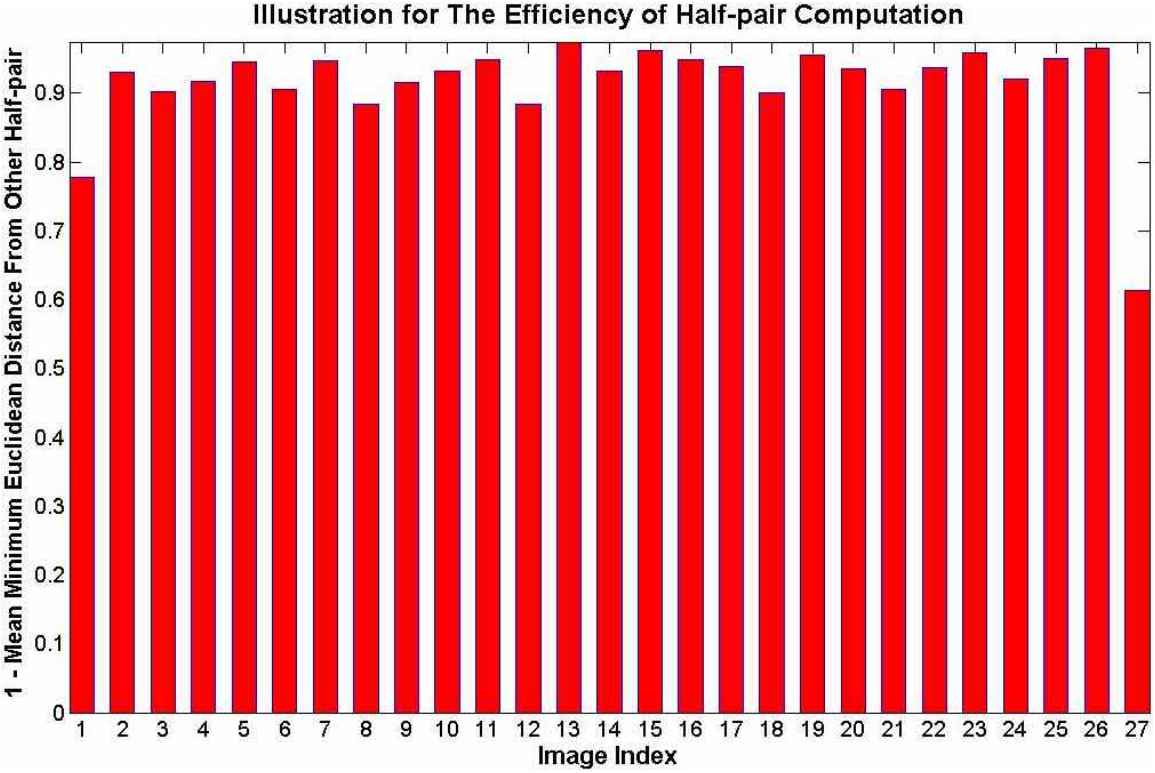
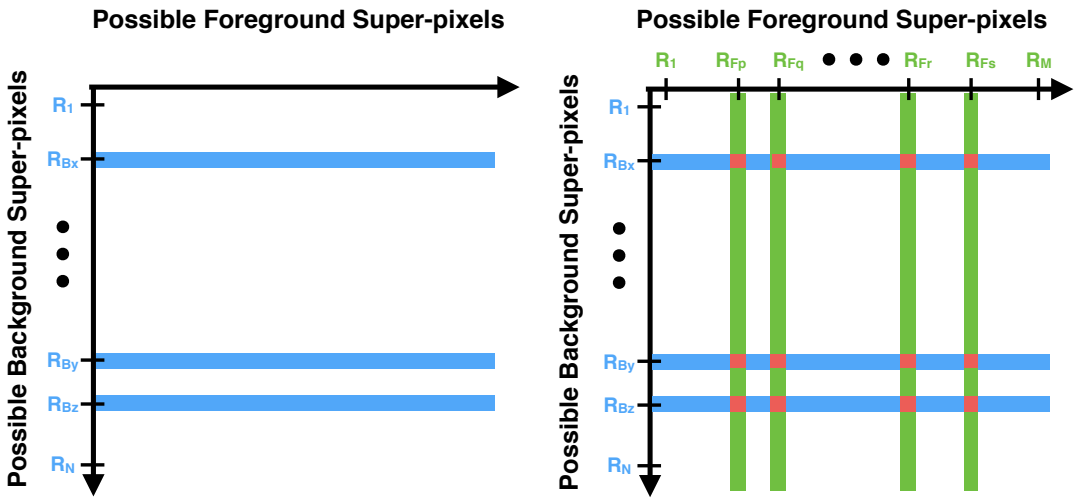


Figure 4.10: Objective assessment of the goodness of the computed half-pairs using the proposed method. To verify the goodness of the computed half-pairs, we used the 27 images of the training dataset in the matting benchmark. Given the ground-truth alpha values and the computed half-pairs, we determine an ideal complement half-pair. Then we calculate the minimum distance between that ideal complement half-pair and all the complement half-pairs in the image. This distance should be small if the computed half-pair is suitable. The height of every bar in the figure equals to 1− the mean minimum distance, where the mean is computed over all the unknown super-pixels' delegates in the image.

the sense that for 25 images out of the 27 images, the mean minimum Euclidean distance is less than 0.2.

4.3.3 Punching the pair space

To proceed to the following stage in the pipeline, after the best half-pair computation, I illustrate in Fig. 4.11(a) the result of the best-half-pair computation step for a particular unknown SP (U_{sp}); this is shown on a 2D space that depicts all the FG and BG super-pixels in the image. The same logic holds for the rest of the unknown super-pixels. Just as an example, in this figure, U_{sp} is assigned to three (out of the say N available) BG super-pixels in the image: R_{B_x} which is its best half-pair, in addition to R_{B_y} and R_{B_z} that it shares with its neighbours. It is worth mentioning that if the unknown SP under consideration would have preferred FG super-pixels as its best half-pairs, we would have



(a) Best Half-pair Computation (b) Pair space Punching

Figure 4.11: An illustration of the best half-pair computation and the punching steps for a single unknown super-pixel.

vertical streaks in Fig. 4.11(a).

Instead of the streaks in Fig. 4.11(a), I need to further narrow-down the search space to a few *patches* (or parts of streaks) in this space. Otherwise, the pair assessment step would be computationally inefficient as in [11]. If I can determine the FG super-pixels that best-suit U_{sp}, given its best half-pairs, then those sought *patches* are the intersections between the previously-found BG super-pixels (best half-pairs) and their most suitable complement among the FG super-pixels. An example of those patches are shown as red squares in Fig. 4.11(b). My final pair-space (*shortlisted pair space*) for that particular unknown SP will thus be the FG/BG pairs (pixels) in those red squares. I name this step: *punching* the pair-space.

Let's recall the chromatic distortion that was mentioned in Chapter 1 and 3; that is the Euclidean distance between the color of a pixel and the color produced by a given FG/BG pair and the corresponding alpha value for the same pixel. It is given by:

$$\xi_{color} = \|I - (\hat{\alpha}F_i + (1 - \hat{\alpha})B_j)\|. \tag{4.7}$$

In order to determine the most-suitable complement among the FG super-pixels for U_{sp}, I calculate the mean color feature for all the FG super-pixels in the image; these will represent the Fs in Eqn. 4.7. I also have the best half-pairs for U_{sp} (all of them are BG super-pixels in our example); these will represent the Bs in Eqn. 4.7. I retain the

K foreground super-pixels that result in the least K values for the chromatic distortion (Eqn. 4.7); $K = 10$ in my experiments. At the end of this stage, I should have acquired for U_{sp}: its best BG half-pairs (R_{B_x}, R_{B_y} and R_{B_z}) and its most-suitable complement FG half-pairs (R_{F_p}, R_{F_q}, $R_{F_r}$and R_{F_s}). The same procedure is taken for all the unknown super-pixels. It is worth mentioning that CIELab feature performed better than RGB feature in this step. The pair space punching is the most computationally demanding stage in the current implementation of the algorithm. This is because, given a BG half-pair for example, I try to find the best complement among **all** the FG SPs in the image. It can also be attributed to the fact that, for an unknown SP under consideration, I perform the punching for each of its delegates, and then determine the complement half-pairs as the K most frequent SPs among the bucket of complement half-pairs computed for all the delegates of U_{sp}. One possible variant of the implementation is to use the mean of U_{sp} instead of looping over its delegates, but I have not tried this variant.

4.3.4 FG/BG pair assessment

The algorithm's pipeline is then concluded by the pair assessment stage. In this step, the FG/BG pair that best describes every delegate in U_{sp} is sought among the delegates of the super-pixels that are depicted as red squares in Fig. 4.11(b). The alpha values of the non-delegate members of U_{sp} were reconstructed using Eqn. 4.4 from the alpha values of the delegates. Since the good nearby samples have already been included in the *shortlisted pair space* while determining the best half-pair for every SP, there was no need to include a proximity term [8, 73] in the objective function. I experimented with three objective functions for pair assessment. Their mathematical formulation and the results obtained by each of them will be presented in sub-section 4.3.5 and sub-section 4.3.6.

4.3.5 Pre and post-processing, results and discussion

All the experiments were implemented using Matlab®, and were run on a PC with Intel Core2Quad 2.66GHz processor and 4GB of RAM. Moreover, in all the experiments, the trimaps used were first expanded using the same method in [73] according to the condition that is given by Eqn. 4.1.

The first adopted cost function is the chromatic distortion in Eqn. 4.7. After calculat-

ing the alpha maps, they undergo a post-smoothing step using the same technique as in [7] and [73]. A modified version of the smoothing module available in the publicly available code of [73] has been used. To smoothen the alpha map, a quadratic cost function in α is minimized. This function is the right-hand side of the equation given by:

$$E = \alpha^T L\alpha + \gamma(\alpha - \hat{\alpha})^T \Gamma(\alpha - \hat{\alpha}) + \lambda(\alpha - \hat{\alpha})^T \Sigma(\alpha - \hat{\alpha}), \tag{4.8}$$

where α is a vector containing the values in the alpha map, $\alpha^T L\alpha$ is a smoothness term that encodes the smoothness constraints of [75] in the Laplacian matrix L. The other two terms in the function serve as data terms. The vector $\hat{\alpha}$ is the values of the alpha map to be smoothed, $\gamma = 10^{-1}$ encodes the relative importance of the data and smoothness terms, Γ is a diagonal matrix whose zero entries for the known foreground and background pixels, and a confidence value f for the unknown pixels. The results presented in this thesis were obtained with $f = \beta \times E_{min}$, where $\beta = 10^7$ and E_{min} is the minimum value attained by the cost function for a particular unknown pixel. The last term involves Σ which is a diagonal matrix with zero entries for unknown pixels and a value of 1 for the known pixels, while λ is a weighting parameter ($\lambda = 100$ in our experiments).

The performance of the proposed method with this objective function was evaluated according to [3] on *May 23, 2015*, and its ranking is available on the benchmark's website under the temporary name '*Anonymous SP_Lett_Subm*'. The tables summarizing the objective performance according to four metrics (MSE, Connectivity, SAD and Gradient) were obtained from the benchmark website and included in Fig. 4.12 through Fig. 4.15.

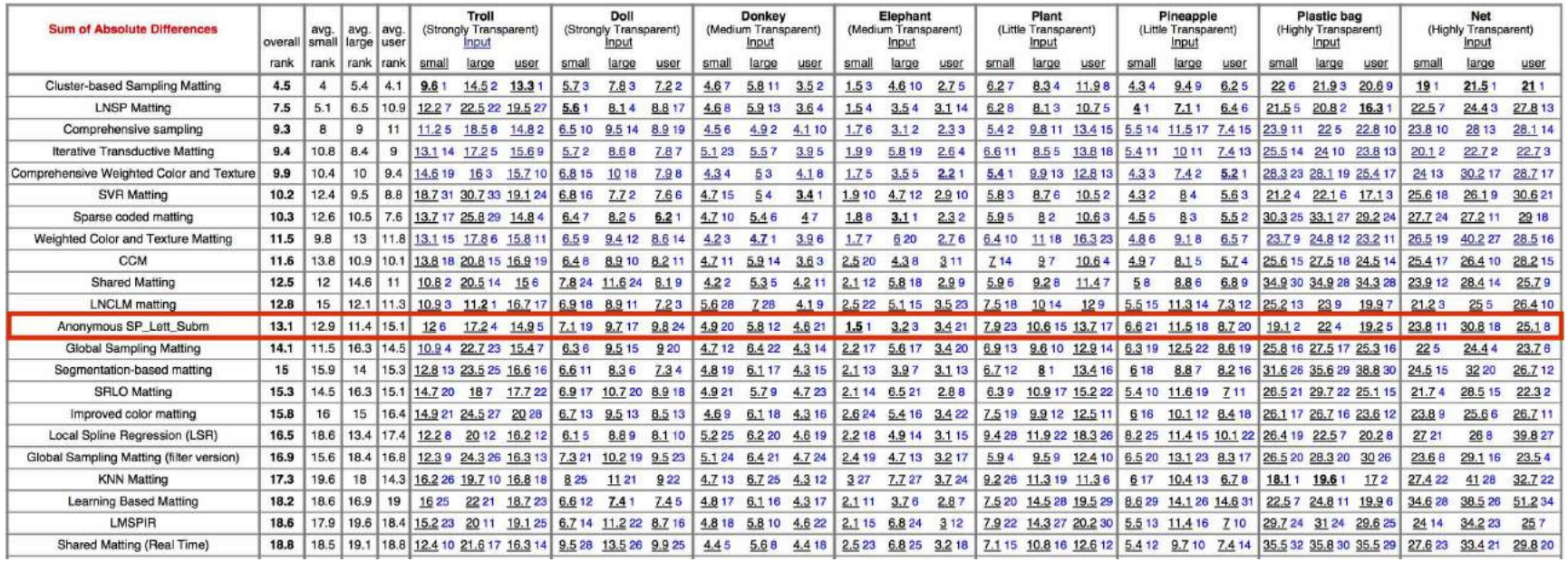
The most remarkable merits according to [3] are: For the connectivity metric, the overall rank of the presented technique is better than that of [12] and [73]. For the gradient metric, the performance of the presented technique is better than [12]; it has also achieved a top score over all the techniques in the benchmark (the second trimap of the image named 'Plastic bag'), and in 25% of the cases, the performance is better than [73]. In 50% of the cases, the performance is better than [13]. For the MSE, the presented technique has achieved a top score over all the techniques (the second trimap of the image named 'Elephant') and its rank is higher than [12]. In 10 cases and 11 cases (out of 24), my algorithm performs better than [73] and [13] respectively. Finally, for the SAD metric, the presented technique has achieved a top score over all the techniques in

Mean Squared Error	overall rank	avg. small rank	avg. large rank	avg. user rank	Troll (Strongly Transparent) Input small	Troll large	Troll user	Doll (Strongly Transparent) Input small	Doll large	Doll user	Donkey (Medium Transparent) Input small	Donkey large	Donkey user	Elephant (Medium Transparent) Input small	Elephant large	Elephant user	Plant (Little Transparent) Input small	Plant large	Plant user	Pineapple (Little Transparent) Input small	Pineapple large	Pineapple user	Plastic bag (Highly Transparent) Input small	Plastic bag large	Plastic bag user	Net (Highly Transparent) Input small	Net large	Net user
Cluster-based Sampling Matting	**6.3**	5.6	6.4	6.9	**0.3** 1	0.6 2	**0.5** 1	0.3 4	0.4 3	0.5 7	0.3 11	0.4 14	0.2 5	0.1 4	0.2 10	0.2 15	0.5 11	0.5 6	1 10	0.3 6	0.7 10	0.5 8	1.5 7	1.4 5	1.2 8	**0.7** 1	**0.8** 1	**0.8** 1
LNSP Matting	**6.6**	4.8	5.5	9.6	0.5 9	1.9 24	1.2 26	**0.2** 1	**0.4** 1	0.5 11	0.3 5	0.4 8	0.2 3	**0** 1	0.1 2	0.2 7	0.4 9	**0.5** 1	0.8 4	**0.2** 1	**0.3** 1	0.4 6	1.4 6	1.2 2	**0.8** 1	1 6	1.1 5	1.5 19
CCM	**8.5**	10.9	8.1	6.6	0.5 12	1.2 12	0.8 12	0.3 7	0.5 11	0.5 8	0.3 3	0.4 6	**0.2** 1	0.1 13	0.2 3	0.1 2	0.5 17	0.6 8	0.7 3	0.3 5	0.4 3	0.3 2	1.7 14	1.8 15	1.5 15	1.2 16	1.2 7	1.3 10
Comprehensive sampling	**9.3**	8.8	9	10	0.4 5	1.2 13	0.8 5	0.3 10	0.6 15	0.6 21	0.3 15	0.3 3	0.3 13	0.1 7	0.2 4	0.1 3	**0.2** 1	0.5 5	0.9 7	0.3 10	0.9 16	0.5 9	1.6 11	1.5 6	1.5 11	1 11	1.2 10	1.3 11
SVR Matting	**9.8**	12.9	8.1	8.5	1.1 28	2.7 32	1.2 25	0.4 23	0.4 2	0.4 5	0.3 16	0.3 4	0.2 2	0.1 6	0.2 7	0.2 11	0.3 4	0.5 4	0.7 2	0.2 3	0.4 4	0.3 3	1.4 5	1.4 3	0.9 3	1.3 18	1.2 9	1.5 17
Comprehensive Weighted Color and Texture	**10.4**	10.4	10.9	10	0.7 20	0.8 3	0.8 13	0.4 17	0.7 20	0.5 10	0.3 4	0.3 2	0.2 6	0.1 5	0.2 8	**0.1** 1	0.3 2	0.7 13	1 13	0.2 2	0.4 2	**0.3** 1	2.2 23	2.1 21	1.7 20	1 10	1.7 18	1.4 16
Sparse coded matting	**12.3**	14	12.9	10	0.6 18	2.7 31	0.8 11	0.3 9	0.4 6	**0.3** 1	0.3 21	0.4 16	0.3 16	0.1 12	0.2 6	0.1 4	0.4 5	0.5 2	0.9 5	0.2 4	0.5 5	0.3 4	2.3 24	2.6 26	2.1 24	1.3 19	1.3 11	1.4 15
Anonymous SP_Lett_Subm	**12.5**	13	9.6	14.8	0.4 3	0.9 4	0.7 3	0.4 20	0.6 16	0.7 23	0.3 20	0.4 7	0.3 19	**0** 3	**0.1** 1	0.3 24	0.7 25	0.7 15	1.1 16	0.6 23	0.8 13	0.8 21	1.2 2	1.5 7	1.1 7	1 8	1.3 14	1 5
Weighted Color and Texture Matting	**12.6**	11.4	13.8	12.6	0.5 13	0.9 6	0.8 4	0.4 13	0.6 17	0.6 18	0.3 2	**0.3** 1	0.2 4	0.1 16	0.5 25	0.2 19	0.5 12	0.8 17	1.5 22	0.3 8	0.5 6	0.5 7	1.6 10	1.6 10	1.5 13	1.2 17	2.6 28	1.4 14
Global Sampling Matting	**12.9**	9.3	15.8	13.8	0.4 4	2.3 27	0.9 15	0.3 5	0.6 14	0.6 19	0.3 8	0.4 20	0.3 7	0.1 9	0.3 16	0.2 14	0.5 13	0.6 7	1 11	0.4 14	1 19	0.7 18	1.9 16	2 20	1.7 19	0.9 5	1 3	1.1 7
Iterative Transductive Matting	**13.5**	14.5	12.3	13.8	0.6 15	0.9 7	0.8 10	0.3 3	0.7 18	0.6 17	0.4 25	0.4 10	0.3 12	0.1 17	0.5 27	0.2 17	0.6 19	0.6 11	1.4 21	0.4 18	0.8 14	0.7 17	1.9 17	1.6 9	1.5 14	0.8 2	0.9 2	0.9 2
LNCLM matting	**13.7**	15.9	12.9	12.3	0.4 2	**0.4** 1	0.9 16	0.4 16	0.5 10	0.4 2	0.4 30	0.6 29	0.3 17	0.1 25	0.3 14	0.2 20	0.6 23	0.8 16	1.1 14	0.4 12	0.9 18	0.5 10	1.8 15	1.6 11	1.1 6	0.8 4	1.1 4	1.3 13
Shared Matting	**14.1**	13.6	15.9	12.8	0.5 6	1.6 18	0.9 14	0.5 25	0.9 26	0.5 13	0.3 6	0.4 5	0.3 15	0.1 18	0.4 20	0.2 9	0.4 7	0.6 10	0.9 6	0.4 11	0.6 8	0.5 11	2.9 29	2.8 28	2.7 28	1 7	1.3 12	1 6
KNN Matting	**14.6**	16.8	14.8	12.3	0.8 23	1 9	0.8 8	0.4 15	0.5 13	0.5 6	0.4 24	0.5 23	0.3 26	0.1 21	0.3 17	0.2 21	0.7 26	0.9 22	0.9 9	0.3 9	0.5 7	0.4 5	**1.1** 1	**1.1** 1	0.9 2	1.2 15	2.3 26	1.6 21
Sampling Based Image Matting Using NMF	**15.3**	11.8	17	17.1	0.7 19	0.9 5	0.8 6	0.4 19	0.8 24	0.8 28	**0.2** 1	0.7 33	0.3 8	0 2	0.2 5	0.5 33	0.4 8	0.7 12	**0.6** 1	0.3 7	0.7 12	0.7 16	1.7 13	2 18	1.9 22	1.6 25	2.4 27	2 23
Improved color matting	**16**	16.9	16.1	14.9	0.8 25	2.4 28	1.5 29	0.3 8	0.5 9	0.5 12	0.3 7	0.4 19	0.3 9	0.1 19	0.3 15	0.2 12	0.7 24	0.7 14	0.9 8	0.4 13	0.7 11	0.7 19	2 19	1.9 17	1.4 10	1.3 20	1.5 16	1.5 20
Segmentation-based matting	**16**	16.6	15	16.5	0.5 14	2.2 25	1.1 20	0.3 11	0.4 4	0.4 3	0.3 13	0.4 18	0.3 10	0.1 8	0.2 9	0.2 5	0.6 20	0.5 3	1.2 19	0.4 17	0.7 9	0.8 23	2.7 27	3.3 31	3.6 30	1.4 23	1.9 21	1.6 22
Improving Sampling Criterion	**17.8**	16.6	19.4	17.5	0.5 7	1.3 14	0.8 7	0.6 28	1 27	0.7 24	0.3 18	0.5 26	0.3 24	0.3 33	0.8 32	0.4 31	0.5 15	0.9 21	1.2 17	0.6 25	1.1 21	0.8 22	1.4 4	1.5 8	1.5 12	0.8 3	1.1 6	0.9 3
SRLO Matting	**17.9**	17.9	18.6	17.3	0.7 22	1 8	1 17	0.4 21	0.8 25	0.7 22	0.4 23	0.4 12	0.3 25	0.1 22	0.6 28	0.2 16	0.4 10	0.8 18	1.5 23	0.4 16	1 20	0.6 14	2 20	2.3 23	1.7 17	1 9	1.4 15	1 4
Local Spline Regression (LSR)	**18.7**	19.5	17.3	19.3	0.5 8	1.5 15	1 18	0.3 6	0.5 12	0.6 15	0.4 22	0.4 22	0.3 18	0.1 14	0.3 13	0.2 13	1.2 31	1.3 29	2.2 28	0.8 29	1.1 22	1.1 25	2.1 22	1.7 12	1.3 9	1.5 24	1.3 13	3 28
Learning Based Matting	**19.2**	19.9	18.6	19	0.8 26	1.6 17	1.3 27	0.4 18	0.4 7	0.5 9	0.3 14	0.4 17	0.3 14	0.1 15	0.2 11	0.2 8	0.5 18	1.2 25	2 26	0.9 31	1.7 30	2 32	1.6 9	1.7 13	1 4	2.2 28	2.7 29	4.2 32
LMSPIR	**19.4**	19	19.8	19.4	0.8 24	1.1 10	1.2 23	0.4 14	0.8 23	0.6 20	0.3 19	0.4 9	0.3 21	0.1 20	0.5 24	0.2 18	0.6 22	1.2 28	2 27	0.4 15	0.9 17	0.6 12	2.3 25	2.3 24	2.2 25	1.1 13	2 23	1.2 9

Figure 4.12: Ranking of the matting algorithms according to the MSE metric on the alpha matting benchmark [3], on the *23rd* of *May 2015*. The presented technique, shown under the name 'Anonymous SP_Lett_Subm', achieved the first position in the second trimap of the image named 'Elephant'.

Connectivity error	overall rank	avg. small rank	avg. large rank	avg. user rank	Troll (Strongly Transparent) Input small	Troll (Strongly Transparent) Input large	Troll (Strongly Transparent) Input user	Doll (Strongly Transparent) Input small	Doll (Strongly Transparent) Input large	Doll (Strongly Transparent) Input user	Donkey (Medium Transparent) Input small	Donkey (Medium Transparent) Input large	Donkey (Medium Transparent) Input user	Elephant (Medium Transparent) Input small	Elephant (Medium Transparent) Input large	Elephant (Medium Transparent) Input user	Plant (Little Transparent) Input small	Plant (Little Transparent) Input large	Plant (Little Transparent) Input user	Pineapple (Little Transparent) Input small	Pineapple (Little Transparent) Input large	Pineapple (Little Transparent) Input user	Plastic bag (Highly Transparent) Input small	Plastic bag (Highly Transparent) Input large	Plastic bag (Highly Transparent) Input user	Net (Highly Transparent) Input small	Net (Highly Transparent) Input large	Net (Highly Transparent) Input user
Random Walk Matting	**4.2**	3.4	6	3.3	0.8 2	0.8 4	0.8 3	0 1	**0** 1	0 1	0.2 18	0.2 14	0.2 17	0 1	**0** 1	0 1	0.1 2	0.4 25	**0.1** 1	0 1	0 1	0 1	**0.2** 1	**0.2** 1	**0.6** 1	**1.2** 1	**1.2** 1	**1.2** 1
Transfusive Weights	**6.3**	4.3	4.7	10.3	0.8 3	0.8 5	0.8 4	0 2	0 2	0.1 2	0.2 19	0.2 15	0.2 18	0 3	0	0	**0.1** 1	0.1 5	0.7 18	0 2	0 2	0 2	0.2 2	0.2 2	0.7 2	1.2 2	1.2 2	3.4 26
CCM	**7.4**	8.1	7.8	6.4	**0.7** 1	0.8 3	0.7 2	0.4 18	0.6 15	0.6 22	0.2 3	0.2 5	0.2 3	0 8	0 3	0 6	0.1 9	0.1 3	0.4 6	0.1 4	0.3 6	0.1 5	15.4 18	12 21	0.7 3	1.4 4	1.5 6	1.4 4
Closed-Form Matting	**8**	8.3	7	8.8	0.8 4	0.8 7	**0.7** 1	0.2 4	0.3 5	0.3 7	0.2 9	0.2 4	0.2 11	0 12	0 8	0 13	0.1 10	0.2 9	0.4 9	0.2 17	0.4 12	0.2 14	0.5 3	1.7 4	1 9	1.6 7	1.6 7	1.5 6
LNSP Matting	**8.3**	10.3	8	6.5	0.8 5	**0.7** 1	0.8 6	0.2 5	0.3 4	0.2 4	0.2 5	0.2 3	0.2 2	0 10	0 9	0 12	0.1 6	**0.1** 1	0.1 2	0.2 9	0.5 14	0.1 6	25.9 27	12.3 23	1.1 10	2.1 15	1.8 9	1.9 10
Large Kernel Matting	**11.3**	13.1	9.6	11	1 24	0.9 14	0.9 20	0.2 7	0.4 8	0.3 6	0.2 11	0.2 13	0.2 16	0 16	0 7	0 8	0.1 14	0.2 11	0.5 11	0.2 18	0.4 11	0.2 11	3.3 5	1.6 3	0.8 5	1.8 10	1.8 10	2 11
SVR Matting	**12.1**	13.9	10.1	12.4	1.2 28	1 16	1.1 27	0.3 10	0.5 11	0.4 11	0.3 26	0.2 20	0.2 23	0 7	0 6	0 18	0.1 8	0.2 12	0.2 4	0.1 5	0.1 3	0.1 3	20.3 22	4.8 9	1 8	1.5 5	1.3 4	1.4 5
Improved color matting	**12.4**	14	10.8	12.4	0.8 7	0.8 8	0.8 12	0.3 13	0.6 12	0.5 15	0.2 14	0.2 10	0.2 10	0 13	0 5	0 9	0.1 16	0.2 8	0.6 16	0.1 3	0.1 4	0.1 4	29.5 29	15.6 28	1.8 16	2.3 17	1.9 11	2.4 17
Local Spline Regression (LSR)	**13.8**	18	13.1	10.3	0.9 15	0.7 2	0.8 5	0.2 6	0.3 6	0.3 10	0.2 22	0.2 16	0.2 13	0 19	0 13	0 15	0.1 17	0.2 16	0.4 7	0.2 19	0.3 8	0.2 13	39.2 37	32.8 36	1.2 11	1.6 9	1.7 8	1.6 8
Anonymous SP_Lett_Subm	**14.6**	14.9	16	13	0.9 17	1.1 22	0.9 21	0.3 11	0.4 9	0.3 9	0.3 29	0.3 23	0.2 25	0 4	0.1 16	0 4	0.1 19	0.2 14	0.4 8	0.2 11	0.8 21	0.2 8	13.7 15	3.7 6	1.6 17	2 13	2.2 17	2.1 12
Cell-based matting Laplacian	**14.6**	16.4	12.4	15.1	0.9 14	0.8 9	0.8 14	0.4 16	1.3 26	0.5 17	0.2 15	0.2 11	0.2 12	0 30	0.1 19	0.1 32	0.1 13	0.1 6	0.6 13	0.4 28	0.3 7	0.3 16	11.5 12	8.5 18	1.3 13	1.2 3	1.3 3	1.2 2
Learning Based Matting	**15.3**	16	15	14.9	0.8 9	0.8 10	0.8 7	0.1 3	0.2 3	0.2 3	0.2 8	0.2 2	0.2 8	0 15	0 11	0 11	0.6 31	1.1 32	2.1 33	0.2 8	0.3 5	0.2 7	22.5 23	13.3 24	1.3 14	4.3 31	5.9 33	8.5 36
Segmentation-based matting	**16.8**	17.9	16.8	15.9	0.8 12	1.7 29	0.8 8	0.3 9	0.4 7	0.4 12	0.2 12	0.2 7	0.2 9	0 17	0 4	0 10	0.1 20	0.3 19	1.1 26	0.2 16	0.3 9	0.3 20	29.5 30	38.5 37	1.8 20	3.1 27	2.8 22	3 22
KNN Matting	**17**	19.9	14.6	16.5	3.4 33	2.8 33	3.1 35	0.3 15	0.5 10	0.4 13	0.2 17	0.2 16	0.2 22	0 21	0 2	0.1 28	0.1 11	0.2 7	0.2 3	0.3 22	0.7 18	0.4 24	17.8 21	8 16	0.8 4	2.6 19	2 13	1.3 3
Improving Sampling Criterion	**17**	16	17.9	17.1	0.8 6	0.9 15	0.9 16	0.3 14	0.8 18	0.3 8	0.2 6	0.2 9	0.2 6	0 29	1.1 33	0 21	0.3 26	0.6 28	1.2 29	0.3 21	0.7 19	0.2 15	5.3 6	4.2 7	4.1 26	2.4 18	2.1 14	2.4 16
Global Sampling Matting	**17.4**	16.8	19	16.4	0.8 11	0.8 6	0.8 11	0.2 8	0.6 16	0.4 14	0.2 10	0.2 8	0.2 5	0 20	0.4 22	0 14	0.1 12	0.3 20	0.6 15	0.2 15	1.1 29	0.3 17	32.7 33	14.5 25	5 27	3 25	3.4 26	3.4 28
Sampling Based Image Matting Using NMF	**17.4**	15.5	18.4	18.4	1.1 27	1.2 23	1 26	0.3 12	0.6 13	0.3 5	0.2 7	0.3 25	0.2 24	0 2	0 10	0 2	0.2 27	0.2 13	0.2 5	0.2 12	0.8 20	0.4 25	12.5 13	14.7 27	7.9 33	2.9 24	2.2 16	3.4 27
High-res matting	**18.4**	17.8	21.8	15.6	0.8 13	2.5 31	0.8 10	0.5 24	1.1 23	0.7 26	**0.2** 1	**0.2** 1	**0.2** 1	0 14	0.7 28	0 5	0.2 23	0.4 26	2.8 35	0.4 30	1.1 28	0.3 22	8.4 8	5.2 10	1 7	3.6 29	3.5 27	2.9 19
LNCLM matting	**18.7**	19.5	17.9	18.6	1 21	1 17	0.8 15	0.5 23	0.6 14	0.5 18	0.3 31	0.3 29	0.3 31	0 27	0.1 17	0.1 26	0.1 3	0.1 4	0.5 10	0.2 10	0.4 10	0.2 10	16.1 19	19.7 32	1.8 19	2.8 22	2.8 20	2.9 20
Weighted Color and Texture Matting	**19.1**	17.1	22	18.1	1 22	1 21	1 24	0.4 17	1 22	0.5 21	0.2 24	0.3 21	0.2 26	0 24	1 32	0 3	0.1 22	0.3 22	0.8 22	0.1 7	0.6 17	0.3 19	11 10	8.4 17	1.7 15	2 11	3.2 24	2.4 15
Comprehensive Weighted Color and Texture	**19.1**	16.1	21.8	19.5	1 25	1.2 24	1 25	0.5 22	0.8 20	0.6 23	0.3 27	0.3 27	0.3 30	0 5	0.3 20	0 17	0.1 5	0.2 17	0.7 17	0.2 13	0.6 16	0.2 12	24.4 26	19 31	2.6 23	1.5 6	2.5 19	1.8 9
Sparse coded matting	**19.4**	20.3	17.8	20.3	1.4 30	1 20	1.3 29	0.6 25	0.9 21	0.5 20	0.3 33	0.4 32	0.3 34	0 23	0.1 15	0 24	0.1 7	0.1 2	0.5 12	0.2 14	1 26	0.2 9	14.3 16	5.5 11	2.4 21	2.1 14	2.2 15	2.3 13
Cluster-based Sampling Matting	**19.5**	19	18.5	21.1	1 23	0.9 13	0.9 19	0.4 19	0.8 19	0.6 24	0.3 30	0.3 28	0.2 20	0 18	0 14	0 25	0.1 4	0.2 10	0.7 20	0.1 6	0.5 13	0.3 16	31.8 31	18.3 30	2.6 24	2.8 21	2.8 21	2.9 21
Shared Matting	**20.1**	19	23	18.3	0.9 16	1.4 27	0.9 18	0.6 27	1.4 29	0.8 27	0.2 4	0.2 12	0.2 7	0 26	0.7 29	0.1 27	0.1 15	0.2 15	0.8 21	0.3 20	1 27	0.4 26	28.2 28	19.8 33	0.9 6	2.1 16	1.9 12	2.4 14
SPS matting	**22**	20.3	24.9	20.8	0.8 8	1.3 25	0.8 13	0.6 26	1.1 24	0.5 16	0.2 16	0.3 26	0.2 14	0 28	1.5 35	0 20	0.2 26	0.4 24	1 25	0.3 23	1.2 30	0.4 23	7.4 7	3.7 5	5.8 30	3.3 28	3.9 30	3.4 25
Comprehensive sampling	**22**	21.8	22	22.1	1 20	1 18	1 22	0.5 20	0.8 17	0.5 19	0.2 21	0.3 22	0.2 21	0 11	0 12	0 19	0.1 21	0.3 21	0.9 23	0.3 24	0.9 23	0.3 21	23.2 24	20.1 34	1.8 18	4.6 33	3.8 29	5 34

Figure 4.13: Ranking of the matting algorithms according to the connectivity metric on the alpha matting benchmark [3], on the *23rd* of *May 2015*. The proposed method is shown under the name 'Anonymous SP_Lett_Subm'.



Sum of Absolute Differences	overall	avg. small	avg. large	avg. user	Troll (Strongly Transparent) Input			Doll (Strongly Transparent) Input			Donkey (Medium Transparent) Input			Elephant (Medium Transparent) Input			Plant (Little Transparent) Input			Pineapple (Little Transparent) Input			Plastic bag (Highly Transparent) Input			Net (Highly Transparent) Input		
	rank	rank	rank	rank	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user
Cluster-based Sampling Matting	**4.5**	4	5.4	4.1	**9.6** 1	14.5 2	**13.3** 1	5.7 3	7.8 3	7.2 2	4.6 7	5.8 11	3.5 2	1.5 3	4.6 10	2.7 5	6.2 7	8.3 4	11.9 8	4.3 4	9.4 9	6.2 5	22 6	21.9 3	20.6 9	**19** 1	**21.5** 1	**21** 1
LNSP Matting	**7.5**	5.1	6.5	10.9	12.2 7	22.5 22	19.5 27	**5.6** 1	8.1 4	8.8 17	4.6 8	5.9 13	3.6 4	1.5 4	3.5 4	3.1 14	6.2 8	8.1 3	10.7 5	**4** 1	**7.1** 1	6.4 6	21.5 5	20.8 2	**16.3** 1	22.5 7	24.4 3	27.8 13
Comprehensive sampling	**9.3**	8	9	11	11.2 5	18.5 8	14.8 2	6.5 10	9.5 14	8.9 19	4.5 6	4.9 2	4.1 10	1.7 6	3.1 2	2.3 3	5.4 2	9.8 11	13.4 15	5.5 14	11.5 17	7.4 15	23.9 11	22 5	22.8 10	23.8 10	28 13	28.1 14
Iterative Transductive Matting	**9.4**	10.8	8.4	9	13.1 14	17.2 5	15.6 9	5.7 2	8.6 8	7.8 7	5.1 23	5.5 7	3.9 5	1.9 9	5.8 19	2.6 4	6.6 11	8.5 5	13.8 18	5.4 11	10 11	7.4 13	25.5 14	24 10	23.8 13	20.1 2	22.7 2	22.7 3
Comprehensive Weighted Color and Texture	**9.9**	10.4	10	9.4	14.6 19	16 3	15.7 10	6.8 15	10 18	7.9 8	4.3 4	5 3	4.1 8	1.7 5	3.5 5	**2.2** 1	**5.4** 1	9.9 13	12.8 13	4.3 3	7.4 2	**5.2** 1	28.3 23	28.1 19	25.4 17	24 13	30.2 17	28.7 17
SVR Matting	**10.2**	12.4	9.5	8.8	18.7 31	30.7 33	19.1 24	6.8 16	7.7 2	7.6 6	4.7 15	5 4	**3.4** 1	1.9 10	4.7 12	2.9 10	5.8 3	8.7 6	10.5 2	4.3 2	8 4	5.6 3	21.2 4	22.1 6	17.1 3	25.6 18	26.1 9	30.6 21
Sparse coded matting	**10.3**	12.6	10.5	7.6	13.7 17	25.8 29	14.8 4	6.4 7	8.2 5	**6.2** 1	4.7 10	5.4 6	4 7	1.8 8	**3.1** 1	2.3 2	5.9 5	8 2	10.6 3	4.5 5	8 3	5.5 2	30.3 25	33.1 27	29.2 24	27.7 24	27.2 11	29 18
Weighted Color and Texture Matting	**11.5**	9.8	13	11.8	13.1 15	17.8 6	15.8 11	6.5 9	9.4 12	8.6 14	4.2 3	4.7 1	3.9 6	1.7 7	6 20	2.7 6	6.4 10	11 18	16.3 23	4.8 6	9.1 8	6.5 7	23.7 9	24.8 12	23.2 11	26.5 19	40.2 27	28.5 16
CCM	**11.6**	13.8	10.9	10.1	13.8 18	20.8 15	16.9 19	6.4 8	8.9 10	8.2 11	4.7 11	5.9 14	3.6 3	2.5 20	4.3 8	3 11	7 14	9 7	10.6 4	4.9 7	8.1 5	5.7 4	25.6 15	27.5 18	24.5 14	25.4 17	26.4 10	28.2 15
Shared Matting	**12.5**	12	14.6	11	10.8 2	20.5 14	15 6	7.8 24	11.6 24	8.1 9	4.2 2	5.3 5	4.2 11	2.1 12	5.8 18	2.9 9	5.9 6	9.2 8	11.4 7	5 8	8.8 6	6.8 9	34.9 30	34.9 28	34.3 28	23.9 12	28.4 14	25.7 9
LNCLM matting	**12.8**	15	12.1	11.3	10.9 3	**11.2** 1	16.7 17	6.9 18	8.9 11	7.2 3	5.6 28	7 26	4.1 9	2.5 22	5.1 15	3.5 23	7.5 18	10 14	12 9	5.5 15	11.3 14	7.3 12	25.2 13	23 9	19.9 7	21.2 3	25 5	26.4 10
Anonymous SP_Lett_Subm	**13.1**	12.9	11.4	15.1	12 6	17.2 4	14.9 5	7.1 19	9.7 17	9.8 24	4.9 20	5.8 12	4.6 21	**1.5** 1	3.2 3	3.4 21	7.9 23	10.6 15	13.7 17	6.6 21	11.5 18	8.7 20	19.1 2	22 4	19.2 5	23.8 11	30.8 18	25.1 8
Global Sampling Matting	**14.1**	11.5	16.3	14.5	10.9 4	22.7 23	15.4 7	6.3 6	9.5 15	9 20	4.7 12	6.4 22	4.3 14	2.2 17	5.6 17	3.4 20	6.9 13	9.6 10	12.9 14	6.3 19	12.5 22	8.6 19	25.8 16	27.5 17	25.3 16	22 5	24.4 4	23.7 6
Segmentation-based matting	**15**	15.9	14	15.3	12.8 13	23.5 25	16.6 16	6.6 11	8.3 6	7.3 4	4.8 19	6.1 17	4.3 15	2.1 13	3.9 7	3.1 13	6.7 12	8 1	13.4 16	6 18	8.8 7	8.2 16	31.6 26	35.6 29	38.8 30	24.5 15	32 20	26.7 12
SRLO Matting	**15.3**	14.5	16.3	15.1	14.7 20	18 7	17.7 22	6.9 17	10.7 20	8.9 18	4.9 21	5.7 9	4.7 23	2.1 14	6.5 21	2.8 8	6.3 9	10.9 17	15.2 22	5.4 10	11.6 19	7 11	26.5 21	29.7 22	25.1 15	21.7 4	28.5 15	22.3 2
Improved color matting	**15.8**	16	15	16.4	14.9 21	24.5 27	20 28	6.7 13	9.5 13	8.5 13	4.6 9	6.1 18	4.3 16	2.6 24	5.4 16	3.4 22	7.5 19	9.9 12	12.5 11	6 16	10.1 12	8.4 18	26.1 17	26.7 16	23.6 12	23.8 9	25.6 6	26.7 11
Local Spline Regression (LSR)	**16.5**	18.6	13.4	17.4	12.2 8	20 12	16.2 12	6.1 5	8.8 9	8.1 10	5.2 25	6.2 20	4.6 19	2.2 18	4.9 14	3.1 15	9.4 28	11.9 22	18.3 26	8.2 25	11.4 15	10.1 22	26.4 19	22.5 7	20.2 8	27 21	26 8	39.8 27
Global Sampling Matting (filter version)	**16.9**	15.6	18.4	16.8	12.3 9	24.3 26	16.3 13	7.3 21	10.2 19	9.5 23	5.1 24	6.4 21	4.7 24	2.4 19	4.7 13	3.2 17	5.9 4	9.5 9	12.4 10	6.5 20	13.1 23	8.3 17	26.5 20	28.3 20	30 26	23.6 8	29.1 16	23.5 4
KNN Matting	**17.3**	19.6	18	14.3	16.2 26	19.7 10	16.8 18	8 25	11 21	9 22	4.7 13	6.7 25	4.3 12	3 27	7.7 27	3.7 24	9.2 26	11.3 19	11.3 6	6 17	10.4 13	6.7 8	**18.1** 1	**19.6** 1	17 2	27.4 22	41 28	32.7 22
Learning Based Matting	**18.2**	18.6	16.9	19	16 25	22 21	18.7 23	6.6 12	7.4 1	7.4 5	4.8 17	6.1 16	4.3 17	2.1 11	3.7 6	2.8 7	7.5 20	14.5 28	19.5 29	8.6 29	14.1 26	14.6 31	22.5 7	24.8 11	19.9 6	34.6 28	38.5 26	51.2 34
LMSPIR	**18.6**	17.9	19.6	18.4	15.2 23	20 11	19.1 25	6.7 14	11.2 22	8.7 16	4.8 18	5.8 10	4.6 22	2.1 15	6.8 24	3 12	7.9 22	14.3 27	20.2 30	5.5 13	11.4 16	7 10	29.7 24	31 24	29.6 25	24 14	34.2 23	25 7
Shared Matting (Real Time)	**18.8**	18.5	19.1	18.8	12.4 10	21.6 17	16.3 14	9.5 28	13.5 26	9.9 25	4.4 5	5.6 8	4.4 18	2.5 23	6.8 25	3.2 18	7.1 15	10.8 16	12.6 12	5.4 12	9.7 10	7.4 14	35.5 32	35.8 30	35.5 29	27.6 23	33.4 21	29.8 20

Figure 4.14: Ranking of the matting algorithms according to the SAD metric on the alpha matting benchmark [3], on the *23rd* of *May 2015*. The presented technique, shown under the name 'Anonymous SP_Lett_Subm', achieved the first position in the first trimap of the image named 'Elephant'.

Gradient error	overall	avg. small	avg. large	avg. user	Troll (Strongly Transparent) Input			Doll (Strongly Transparent) Input			Donkey (Medium Transparent) Input			Elephant (Medium Transparent) Input			Plant (Little Transparent) Input			Pineapple (Little Transparent) Input			Plastic bag (Highly Transparent) Input			Net (Highly Transparent) Input		
	rank	rank	rank	rank	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user
LNSP Matting	**8.1**	6.3	7	11	0.2 3	0.3 6	**0.2** 1	**0.2** 1	**0.2** 1	0.3 14	0.4 3	0.5 11	0.4 3	**0.3** 1	0.5 2	0.6 18	1.8 11	1.9 6	2.7 15	1.2 3	1.6 4	2 14	1.4 16	1.3 9	0.9 11	0.6 12	0.8 17	0.6 12
Comprehensive sampling	**8.4**	8.5	7.6	9	0.2 5	0.2 3	0.2 5	0.2 11	0.3 14	0.4 22	0.4 5	0.4 3	0.4 4	0.3 6	0.5 3	0.4 3	1.1 2	1.7 1	2.3 2	1.5 12	2.4 12	1.9 11	1.3 6	1.3 10	0.9 6	0.7 21	0.7 15	0.7 19
Cluster-based Sampling Matting	**9.9**	10	9.9	9.8	**0.2** 1	0.2 2	0.2 2	0.2 16	0.3 15	0.3 16	0.4 19	0.5 12	0.4 19	0.3 2	0.7 12	0.5 8	1.8 10	2 9	2.7 13	1.2 4	2.1 10	1.8 8	1.5 20	1.4 14	0.8 4	0.5 8	0.6 5	0.5 8
CCM	**9.9**	12	9.3	8.4	0.2 6	0.3 8	0.2 4	0.2 7	0.3 5	0.3 8	0.4 14	0.5 13	0.4 12	0.4 18	0.6 5	0.5 5	2.1 18	2.3 14	2.5 8	1.3 6	1.7 5	1.6 4	1.3 5	1.3 6	0.8 3	0.7 22	0.8 18	0.7 23
SVR Matting	**10**	12.1	10.6	7.3	0.3 25	0.4 28	0.3 16	0.3 23	0.3 8	0.3 9	0.4 7	0.5 10	0.4 2	0.4 9	0.6 9	0.5 6	1.5 7	2.1 11	2.5 7	1.2 5	1.6 3	1.5 3	1.3 3	1.2 4	0.8 2	0.7 18	0.7 12	0.6 13
Sparse coded matting	**10.4**	11.9	9.1	10.3	0.3 24	0.4 21	0.3 24	0.3 17	0.3 9	0.3 13	0.4 17	0.5 6	0.4 14	0.3 7	**0.5** 1	**0.4** 1	1.4 6	1.7 4	2.3 4	1.1 2	1.6 2	1.3 2	1.5 19	1.6 27	1.1 21	0.5 3	0.5 3	0.4 3
Segmentation-based matting	**10.7**	13.5	9.1	9.4	0.2 8	0.3 4	0.2 6	0.2 5	0.2 2	0.3 2	0.4 10	0.5 18	0.4 8	0.4 10	0.6 4	0.5 7	2.1 19	1.7 3	2.5 9	1.9 18	2.2 11	2.5 19	2 34	1.7 29	1.2 23	0.5 4	0.5 2	**0.4** 1
Global Sampling Matting	**11**	10.9	11.8	10.3	0.2 2	0.3 12	0.2 3	0.2 3	0.3 3	0.3 5	0.4 8	0.5 14	0.4 6	0.4 15	0.8 15	0.6 19	2.1 20	2.1 13	2.7 12	2 20	2.8 22	2.6 20	1.4 12	1.3 7	0.9 10	0.5 7	0.6 8	0.5 7
Shared Matting	**11.2**	11.3	11.9	10.4	0.2 4	0.3 9	0.2 7	0.3 27	0.6 27	0.5 29	0.4 4	0.4 5	0.4 5	0.4 12	0.8 16	0.5 4	1.6 9	2 10	2.3 3	1.4 7	1.9 6	1.8 7	1.5 17	1.5 18	1.1 22	0.6 10	0.6 4	0.4 6
Improved color matting	**11.3**	12.6	11.8	9.6	0.2 9	0.3 7	0.2 9	0.2 9	0.3 7	0.3 3	0.4 9	0.5 17	0.4 11	0.5 21	0.7 14	0.5 15	2.5 24	2.5 20	2.7 14	1.9 17	2.5 16	2.4 18	1.4 11	1.4 12	0.9 5	**0.4** 1	**0.4** 1	0.4 2
Comprehensive Weighted Color and Texture	**13.4**	13	15.1	12	0.4 30	0.4 29	0.5 31	0.4 29	0.5 23	0.5 24	0.3 2	**0.4** 1	**0.4** 1	0.3 5	0.6 7	0.4 2	1.1 1	1.8 5	2.4 5	1 1	1.4 1	1.2 1	1.7 31	1.8 31	1.2 27	0.5 5	0.9 24	0.4 5
Anonymous SP_Lett_Subm	**14.8**	16.6	10.8	17	0.2 7	0.3 5	0.2 8	0.2 15	0.3 11	0.3 17	0.5 24	0.4 4	0.4 20	0.3 3	0.6 6	0.7 25	2.7 26	2.6 23	3.1 21	2.4 24	2.7 20	2.7 22	1.4 9	**1.1** 1	0.9 8	0.8 25	0.7 16	0.6 15
Global Sampling Matting (filter version)	**15.8**	14.6	15.4	17.3	0.2 10	0.4 27	0.3 17	0.3 25	0.4 19	0.4 23	0.5 26	0.5 20	0.5 27	0.6 27	0.8 18	0.7 26	1.3 4	1.7 2	**2.2** 1	2 19	2.7 19	2.2 16	1.3 4	1.2 5	1.2 24	0.4 2	0.7 13	0.4 4
Weighted Color and Texture Matting	**16.1**	15.1	15.6	17.5	0.2 14	0.3 13	0.2 10	0.3 24	0.4 20	0.5 26	0.4 6	0.4 2	0.4 7	0.5 22	1.1 25	0.7 27	1.9 14	2.4 16	3.3 25	1.5 11	2 8	1.9 10	1.4 15	1.4 15	1.1 19	0.6 15	0.9 26	0.6 16
LNCLM matting	**16.1**	17.9	15.9	14.6	0.2 11	**0.2** 1	0.3 13	0.3 22	0.4 18	0.3 12	0.5 28	0.6 30	0.5 25	0.5 20	0.7 11	0.6 16	2.1 21	2.6 22	2.9 16	1.5 10	2.4 13	2 12	1.6 22	1.6 22	0.9 9	0.6 9	0.6 10	0.6 14
Iterative Transductive Matting	**17.4**	18.5	16.3	17.4	0.3 19	0.4 22	0.3 14	0.2 13	0.6 26	0.5 28	0.5 30	0.5 7	0.4 21	0.3 8	1.1 24	0.5 14	2 17	2.1 12	3 20	1.8 16	2.5 17	2.3 17	1.6 28	1.4 11	1 15	0.7 17	0.6 11	0.5 10
Sampling Based Image Matting Using NMF	**17.5**	10.4	18.4	23.6	0.3 16	0.4 20	0.4 28	0.2 12	0.4 16	0.4 20	**0.3** 1	0.6 29	0.4 17	0.3 4	0.9 20	1.6 36	1.9 15	2.4 15	2.5 10	1.6 14	2.4 14	2.8 24	1.3 7	1.3 8	1.3 30	0.6 14	0.9 25	0.7 24
SPS matting	**17.5**	17	19.4	16.3	0.2 12	0.3 18	0.3 12	0.4 30	0.7 30	0.5 25	0.5 21	0.5 22	0.5 26	0.6 29	1.4 30	0.7 23	1.3 5	2 8	2.5 6	2.1 22	2.7 21	2.1 15	**1** 1	1.2 3	**0.8** 1	0.6 16	0.8 23	0.7 22
SRLO Matting	**18.3**	19	19.8	16.1	0.3 28	0.4 24	0.3 26	0.3 28	0.6 28	0.6 30	0.5 23	0.5 8	0.4 9	0.4 17	1.2 26	0.5 9	1.2 3	1.9 7	2.6 11	1.4 8	2.5 18	1.8 6	1.6 25	1.9 33	1 17	0.7 20	0.7 14	0.7 21
Learning Based Matting	**18.8**	17.6	19.1	19.8	0.3 22	0.3 15	0.3 22	0.2 6	0.3 4	0.3 4	0.4 16	0.5 23	0.4 18	0.4 13	0.6 8	0.5 13	1.9 13	2.5 19	3.2 24	3.1 32	3.6 31	3.9 32	1.4 10	1.6 24	1 14	0.9 29	1.1 29	1.2 31
LMSPIR	**18.9**	18.1	19.1	19.4	0.3 27	0.4 25	0.3 25	0.3 21	0.7 29	0.5 27	0.4 20	0.5 9	0.4 16	0.4 11	1.2 27	0.5 11	1.5 8	2.5 18	3.2 23	1.6 13	2.4 15	1.9 9	1.6 26	1.6 21	1.2 26	0.7 19	0.6 9	0.6 18
Local Spline Regression (LSR)	**19.1**	20.4	19.1	17.9	0.2 13	0.3 10	0.3 11	0.2 4	0.3 12	0.3 11	0.4 15	0.5 15	0.4 10	0.4 14	0.7 13	0.5 12	3.4 32	3.5 29	4.5 29	2.9 28	3 24	3.2 26	1.6 29	1.7 28	1 18	0.8 28	0.8 22	0.9 26

Figure 4.15: Ranking of the matting algorithms according to the gradient metric on the alpha matting benchmark [3], on the *23rd* of *May 2015*. The presented technique, shown under the name 'Anonymous SP_Lett_Subm', achieved the first position in the second trimap of the image named 'Plastic bag'.

the first trimap of the image named 'Elephant', and in 13 cases, 6 cases and 10 cases (out of 24), my algorithm performs better than [12], [73] and [13] respectively.

Fig. 4.16 depicts a few cases where the proposed method is more efficient than the SoA techniques [73] and [12]. In the eleven cases shown, the subjective quality (compared to the ground truth), as well as the objective quality (measured in terms of MSE) are better/comparable to the other techniques. The patches are chosen to demonstrate the effectiveness of the proposed technique in many cases; nevertheless, there are other cases where it was worse than both.

4.3.6 Other cost functions for pair assessment

Following the criterion of [36] and [10] to determine a confident pair, the first adopted objective function was extended to involve two other terms that quantify the separation between the two half-pairs of a pair under consideration (also called the robustness of a pair), and promote the sparsity [70, 36] of the calculated alpha maps as well. Throughout the rest of this document, the second adopted objective function will be referred to as the *extended cost/objective function*. The formulation used in this thesis for the sparsity-promoting term is different from the formulations used in [36] and [10]. The extended cost function is given by:

$$\xi_{rs} = \frac{\|I - (\hat{\alpha}F_i + (1-\hat{\alpha})B_j)\|}{\|F_i - B_j\|} \times \min\left\{\frac{W_1}{W_1+W_2}, \frac{W_2}{W_1+W_2}\right\} \quad \text{where,} \tag{4.9a}$$

$$W_1 = \exp(-\|I - F_i\|), \quad W_2 = \exp(-\|I - B_j\|), \tag{4.9b}$$

where ξ_{rs} is the cost of the function that reflects the robustness of the pairs and encourages the sparsity in alpha maps. The numerator of the first term is the chromatic distortion that was adopted as our first objective function, while the denominator serves to favour FG/BG pairs that come from widely separated color distributions. This criterion facilitates an accurate estimation of the α value using the composting equation.

The sparsity property states that most pixels are expected to be either fully foreground or fully background. If so, one half-pair should be close in color space to the unknown pixel under consideration. The formulation of the sparsity-promoting terms in [36] and [10]

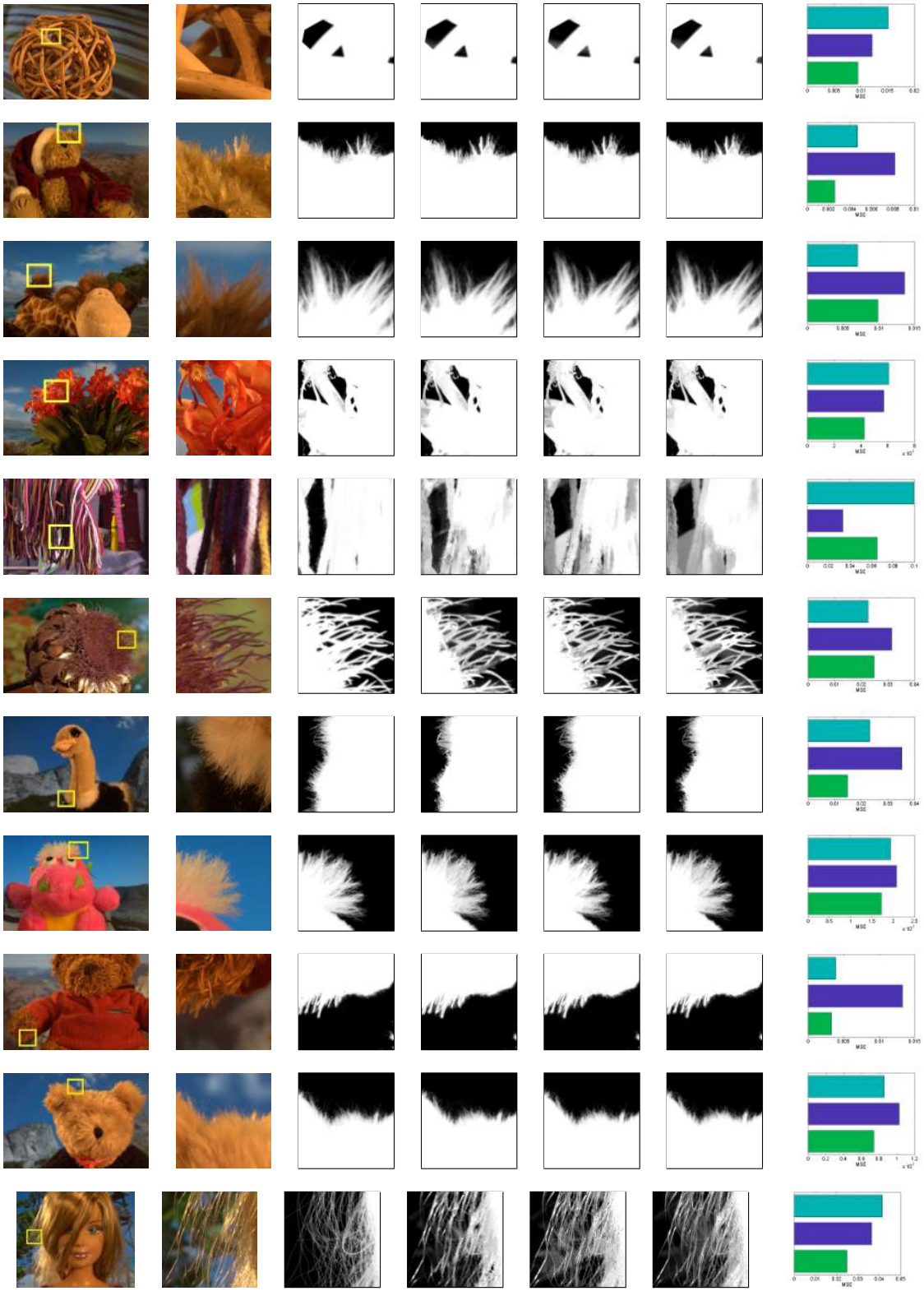


Figure 4.16: A few comparisons with the SoA sampling-based matting techniques. From the left, the columns depict the original image with the patch under consideration highlighted by a yellow rectangle, the patch enlarged, its ground truth alpha map, the result of [12] (WCT), the result of [73] (CS), our result and the MSE of the three techniques; the green, purple and turquoise bars are ours, [12] and [73] respectively. Please see text for more details.

involves two weights that quantify the closeness of the unknown pixel under consideration to both the FG samples and the BG samples. In this thesis, for every FG/BG pair, a weight is calculated as a function of the Euclidean distance (in color space) between the unknown pixel and each half-pair (W_1 and W_2). Two ratios are then calculated for each of the calculated weights to their sum, and the minimum of each of these ratios is embedded as a weight in the cost function. The minimum of the two ratios corresponds to the half-pair that is farther in the feature space from the unknown pixel, which may sound contradictory with the formulations in [36] and [10]. However, the rationale for the formulation used in this thesis is: No matter which half-pair is a FG and which is a BG, if a large separation between the two half-pairs (H_1^p,H_2^p) is guaranteed by the denominator of the first term in the cost function, and a large separation is guaranteed between the unknown pixel I and only one of the half-pairs (let it be H_2^p) by the second term, then there will be one of the following two cases. Either the unknown pixel and H_1^p are also far away from each other in the feature space (like I and H_2^p), and in this case the chromatic distortion would be high and the pair would be dis-qualified, or the unknown pixel and H_1^p are nearby in the feature space, which is what the sparsity-promoting terms are meant to do.

The performance of the presented sampling strategy with the extended cost function in Eqn. 4.9a consistently gave better results on the benchmark than using the chromatic distortion only. It was evaluated according to [3] on *February 20, 2016*. The tables summarizing the objective performance according to four metrics (MSE, Connectivity, SAD and Gradient) were obtained from the benchmark website and included in Fig. 4.17 through Fig. 4.20. The ranking of the presented method appears in the tables under the temporary name '*TSPS Robust Sampling*'.

The most remarkable merits for using the extended cost function, according to [3] are: For the connectivity metric, the overall rank of the presented technique is better than that of [12] and [13] and [73]. Its rank is also six-positions higher than the rank of the performance with the cost function minimizing the chromatic distortion only. For the gradient metric, the presented technique has achieved a top score over all the techniques in the benchmark (the second trimap of the image named 'Plastic bag'); in 11 cases out of 24 cases, the performance is better than [13], and in 25% of the cases, the performance is

Mean Squared Error	overall rank	avg. small rank	avg. large rank	avg. user rank	Troll (Strongly Transparent) Input			Doll (Strongly Transparent) Input			Donkey (Medium Transparent) Input			Elephant (Medium Transparent) Input			Plant (Little Transparent) Input			Pineapple (Little Transparent) Input			Plastic bag (Highly Transparent) Input			Net (Highly Transparent) Input		
					small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user
anonymous_submission	**5.1**	4.3	3.8	7.3	0.5 9	0.7 4	0.5 2	**0.2** 1	0.3 2	0.4 3	0.2 2	0.3 5	0.2 5	0.1 8	0.2 7	0.1 4	0.3 4	**0.4** 1	1.2 22	0.2 4	0.4 5	0.4 5	1.3 5	1.2 2	0.9 4	**0.6** 1	0.9 4	1.2 13
LNSP Matting	**8.8**	6.4	7.8	12.1	0.5 12	1.9 30	1.2 32	0.2 2	0.4 3	0.5 15	0.3 7	0.4 12	0.2 3	**0** 1	0.1 2	0.2 10	0.4 11	0.5 3	0.8 4	**0.2** 1	**0.3** 1	0.4 7	1.4 8	1.2 3	**0.8** 1	1 9	1.1 8	1.5 25
Cluster-based Sampling Matting	**9.3**	8.6	9.6	9.8	**0.3** 1	0.6 2	0.5 3	0.3 10	0.4 7	0.5 10	0.3 14	0.4 20	0.2 9	0.1 9	0.2 15	0.2 20	0.5 14	0.5 10	1 12	0.3 9	0.7 14	0.5 11	1.5 10	1.4 8	1.2 12	0.7 2	**0.8** 1	**0.8** 1
Trajectory	**9.8**	7.4	8.8	13.1	0.5 13	1.9 31	1.2 33	0.2 3	0.4 4	0.5 16	0.3 8	0.4 13	0.2 4	0 2	0.1 3	0.2 11	0.4 12	0.5 4	0.8 5	0.2 2	0.3 2	0.4 8	1.4 9	1.2 4	0.8 2	1 10	1.1 9	1.5 26
KL-Divergence Based Sparse Sampling	**11.8**	11.6	9.9	13.8	0.4 6	0.9 7	0.7 6	0.3 8	0.5 12	0.5 14	0.3 26	0.4 8	0.3 15	0.1 7	0.2 12	**0.1** 1	0.4 8	0.4 2	1.2 20	0.4 16	0.6 12	0.6 18	1.7 18	2 24	2.1 31	0.8 4	0.8 2	0.9 5
CCM	**12**	15.3	11.8	9	0.5 17	1.2 16	0.8 18	0.3 13	0.5 16	0.5 11	0.3 4	0.4 10	**0.2** 1	0.1 19	0.2 5	0.1 3	0.5 21	0.6 12	0.7 3	0.3 8	0.4 4	0.3 2	1.7 17	1.8 19	1.5 20	1.2 23	1.2 12	1.3 14
Anonymous_TIP_submission	**12.5**	13.6	12.1	11.8	0.5 19	1.5 22	0.8 9	0.3 7	**0.3** 1	0.3 2	0.3 18	0.4 16	0.3 13	0 4	0.2 6	0.1 7	0.5 15	0.5 5	0.9 7	0.2 5	0.6 10	0.5 10	2.2 29	2.4 31	1.9 29	1 12	1.1 6	1.3 17
Comprehensive sampling	**13.2**	12.4	13	14.3	0.4 7	1.2 18	0.8 10	0.3 16	0.6 21	0.6 27	0.3 20	0.3 3	0.3 20	0.1 12	0.2 8	0.1 5	**0.2** 1	0.5 9	0.9 9	0.3 13	0.9 21	0.5 12	1.6 14	1.5 9	1.5 16	1 16	1.2 15	1.3 15
SVR Matting	**13.4**	17.5	11.6	11.1	1.1 35	2.7 39	1.2 31	0.4 30	0.4 5	0.4 7	0.3 21	0.3 4	0.2 2	0.1 11	0.2 11	0.2 16	0.3 5	0.5 8	0.7 2	0.2 6	0.4 6	0.3 3	1.4 7	1.4 6	0.9 5	1.3 25	1.2 14	1.5 23
Comprehensive Weighted Color and Texture	**14.2**	14.4	14.9	13.4	0.7 26	0.8 5	0.8 19	0.4 24	0.7 27	0.5 13	0.3 5	0.3 2	0.2 10	0.1 10	0.2 13	0.1 2	0.3 2	0.7 18	1 15	0.2 3	0.4 3	**0.3** 1	2.2 30	2.1 27	1.7 25	1 15	1.7 24	1.4 22
TSPS Robust Sampling	**14.3**	14.5	11.8	16.6	0.4 3	0.9 9	0.6 4	0.4 19	0.5 17	0.7 28	0.3 23	0.3 7	0.3 18	0 6	0.1 4	0.3 32	0.7 29	0.7 16	1.1 16	0.5 25	0.7 16	0.7 24	1.2 3	1.3 5	1 7	0.9 8	1.4 20	0.9 4
LocalSamplingAndKnnClassification	**15.3**	17.4	13.3	15.4	0.5 14	0.7 3	**0.5** 1	0.3 4	0.4 6	0.5 18	0.3 6	0.3 6	0.2 8	0.1 14	0.2 16	0.2 15	0.9 33	1 28	1.7 30	0.6 30	1 25	0.9 29	1.8 19	1.5 11	1.2 11	1.1 19	1.2 11	1.2 11
Anonymous SP_Lett_Subm	**16.6**	17.1	13.4	19.4	0.4 4	0.9 6	0.7 7	0.4 27	0.6 22	0.7 30	0.3 27	0.4 11	0.3 26	0 5	**0.1** 1	0.3 30	0.7 30	0.7 20	1.1 19	0.6 29	0.8 18	0.8 26	1.2 2	1.5 10	1.1 10	1 13	1.3 19	1 7
Sparse coded matting	**16.9**	19.4	17.8	13.5	0.6 24	2.7 38	0.8 17	0.3 15	0.4 10	**0.3** 1	0.3 28	0.4 22	0.3 23	0.1 18	0.2 10	0.1 6	0.4 6	0.5 6	0.9 6	0.2 7	0.5 7	0.3 4	2.3 31	2.6 33	2.1 30	1.3 26	1.3 16	1.4 21
Weighted Color and Texture Matting	**17**	15.9	18.1	17	0.5 18	0.9 10	0.8 8	0.4 20	0.6 23	0.6 24	0.3 3	**0.3** 1	0.2 6	0.1 22	0.5 32	0.2 24	0.5 16	0.8 22	1.5 27	0.3 11	0.5 8	0.5 9	1.6 13	1.6 14	1.5 18	1.2 24	2.6 35	1.4 20
Global Sampling Matting	**17.4**	13.1	21.1	18	0.4 5	2.3 34	0.9 21	0.3 11	0.6 20	0.6 25	0.3 11	0.4 26	0.3 11	0.1 15	0.3 23	0.2 19	0.5 17	0.6 11	1 13	0.4 18	1 24	0.7 22	1.9 21	2 26	1.7 24	0.9 7	1 5	1.1 9
LNCLM matting	**18.1**	20.3	17.4	16.8	0.4 2	**0.4** 1	0.9 22	0.4 23	0.5 15	0.4 4	0.4 37	0.6 36	0.3 24	0.1 32	0.3 21	0.2 26	0.6 27	0.8 21	1.1 17	0.4 15	0.9 23	0.5 13	1.8 20	1.6 15	1.1 9	0.8 6	1.1 7	1.3 19
Iterative Transductive Matting	**18.3**	19.5	16.9	18.5	0.6 21	0.9 11	0.8 16	0.3 9	0.7 25	0.6 23	0.4 32	0.4 15	0.3 19	0.1 23	0.5 34	0.2 22	0.6 23	0.6 15	1.4 26	0.4 23	0.8 19	0.7 21	1.9 22	1.6 13	1.5 19	0.8 3	0.9 3	0.9 2
Shared Matting	**18.8**	17.9	21.3	17.4	0.5 8	1.6 24	0.9 20	0.5 32	0.9 33	0.5 19	0.3 9	0.4 9	0.3 22	0.1 24	0.4 27	0.2 13	0.4 9	0.6 14	0.9 8	0.4 14	0.6 11	0.5 14	2.9 36	2.8 35	2.7 35	1 11	1.3 17	1 8
KNN Matting	**19.2**	21.9	19.4	16.4	0.8 29	1 13	0.8 14	0.4 22	0.5 19	0.5 9	0.4 31	0.5 30	0.3 33	0.1 27	0.3 24	0.2 27	0.7 31	0.9 27	0.9 11	0.3 12	0.5 9	0.4 6	**1.1** 1	**1.1** 1	0.9 3	1.2 22	2.3 32	1.6 28

Figure 4.17: Ranking of the matting algorithms according to the MSE metric on the alpha matting benchmark [3], on the *20th* of *February 2016*. The proposed method is shown under the name 'TSPS Robust Sampling'.

Connectivity error	overall rank	avg. small rank	avg. large rank	avg. user rank	Troll (Strongly Transparent) Input			Doll (Strongly Transparent) Input			Donkey (Medium Transparent) Input			Elephant (Medium Transparent) Input			Plant (Little Transparent) Input			Pineapple (Little Transparent) Input			Plastic bag (Highly Transparent) Input			Net (Highly Transparent) Input		
					small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user
Random Walk Matting	**5**	3.9	7.4	3.8	0.8 2	0.8 5	0.8 3	**0** 1	**0** 1	**0** 1	0.2 21	0.2 17	0.2 21	**0** 1	**0** 1	**0** 1	0.1 3	0.4 32	0.1 1	**0** 1	**0** 1	**0** 1	**0.2** 1	**0.2** 1	**0.6** 1	**1.2** 1	**1.2** 1	**1.2** 1
Transfusive Weights	**7.5**	4.8	5.7	12.3	0.8 3	0.8 6	0.8 4	0 2	0 2	0.1 2	0.2 22	0.2 18	0.2 22	0 3	0	0	0.1 2	0.1 8	0.7 23	0 2	0 2	0 2	0.2 2	0.2 2	0.7 2	1.2 2	1.2 2	3.4 31
CCM	**9**	9.9	9.3	7.8	**0.7** 1	0.8 4	0.7 2	0.4 25	0.6 21	0.6 29	0.2 3	0.2 6	0.2 4	0 9	0 3	0 7	0.1 13	0.1 5	0.4 8	0.1 4	0.3 7	0.1 5	15.4 20	12 22	0.7 3	1.4 4	1.5 6	1.4 4
LNSP Matting	**9.2**	11.6	8.8	7.1	0.8 5	**0.7** 1	0.8 6	0.2 5	0.3 4	0.2 4	0.2 5	0.2 3	0.2 2	0 12	0 10	0 16	0.1 7	**0.1** 1	0.1 2	0.2 10	0.5 17	0.1 6	25.9 31	12.3 24	1.1 10	2.1 18	1.8 10	1.9 11
Closed-Form Matting	**9.9**	10.1	8.5	11	0.8 4	0.8 9	**0.7** 1	0.2 4	0.3 6	0.3 9	0.2 10	0.2 5	0.2 14	0 16	0 9	0 18	0.1 14	0.2 13	0.4 13	0.2 22	0.4 15	0.2 17	0.5 3	1.7 4	1 9	1.6 8	1.6 7	1.5 7
Trajectory	**10.2**	12.6	9.8	8.1	0.8 6	0.7 2	0.8 7	0.2 6	0.3 5	0.2 5	0.2 6	0.2 4	0.2 3	0 13	0 11	0 17	0.1 8	0.1 2	0.1 3	0.2 11	0.5 18	0.1 7	25.9 32	12.3 25	1.1 11	2.1 19	1.8 11	1.9 12
Large Kernel Matting	**13.9**	15.9	11.9	13.9	1 28	0.9 16	0.9 23	0.2 9	0.4 12	0.3 8	0.2 12	0.2 16	0.2 20	0 20	0 7	0 12	0.1 18	0.2 15	0.5 15	0.2 24	0.4 14	0.2 14	3.3 5	1.6 3	0.8 5	1.8 11	1.8 12	2 14
SVR Matting	**14.7**	16.6	12.4	15.1	1.2 34	1 21	1.1 34	0.3 14	0.5 15	0.4 15	0.3 32	0.2 24	0.2 28	0 8	0 6	0 23	0.1 10	0.2 16	0.2 5	0.1 5	0.1 3	0.1 3	20.3 25	4.8 10	1 8	1.5 5	1.3 4	1.4 5
TSPS Robust Sampling	**15.3**	15.6	17	13.3	0.9 16	1.1 27	0.8 12	0.2 10	0.3 8	0.3 10	0.2 27	0.2 25	0.2 25	0 5	0 18	0 6	0.1 12	0.1 9	0.4 9	0.2 19	0.7 24	0.2 11	16.1 22	4.5 8	1.6 17	2 14	2.1 17	2.2 16
Improved color matting	**15.3**	17.4	13	15.6	0.8 8	0.8 10	0.8 15	0.3 17	0.6 16	0.5 20	0.2 17	0.2 12	0.2 12	0 17	0 5	0 13	0.1 21	0.2 12	0.6 21	0.1 3	0.1 4	0.1 4	29.5 35	15.6 32	1.8 19	2.3 21	1.9 13	2.4 21
anonymous_submission	**15.5**	14	15.3	17.3	1.1 32	1.1 29	1 32	0.2 11	0.3 9	0.3 13	0.2 16	0.2 8	0.2 15	0 24	0 16	0 8	**0.1** 1	0.1 7	0.4 12	0.1 6	0.2 5	0.2 8	9.8 10	12.6 26	5.9 37	1.9 12	2.3 22	1.9 13
LocalSamplingAndKnnClassification	**16.7**	19.8	15.8	14.5	0.8 10	0.8 8	0.8 11	0.2 7	0.3 7	0.3 6	0.2 15	0.2 13	0.2 13	0 30	0 15	0 11	0.1 26	0.2 21	0.7 24	0.2 23	0.4 13	0.3 21	32.9 40	22 41	2 24	1.6 7	1.7 8	1.5 6
Local Spline Regression (LSR)	**17**	21.9	16.3	12.9	0.9 18	0.7 3	0.8 5	0.2 6	0.3 10	0.3 14	0.2 25	0.2 19	0.2 17	0 23	0 17	0 20	0.1 22	0.2 20	0.4 10	0.2 25	0.3 9	0.2 16	39.2 44	32.8 43	1.2 12	1.6 10	1.7 9	1.6 9
Anonymous SP_Lett_Subm	**18**	17.9	20.1	15.9	0.9 20	1.1 28	0.9 25	0.3 15	0.4 13	0.3 12	0.3 35	0.3 28	0.2 30	0 4	0.1 21	0 4	0.1 24	0.2 18	0.4 11	0.2 13	0.8 27	0.2 10	13.7 16	3.7 6	1.8 20	2 16	2.2 20	2.1 15
Cell-based matting Laplacian	**18.1**	20	15.4	19	0.9 17	0.8 11	0.8 17	0.4 21	1.3 33	0.5 23	0.2 18	0.2 14	0.2 16	0 37	0.1 25	0.1 39	0.1 17	0.1 10	0.6 18	0.4 34	0.3 8	0.3 23	11.5 13	8.5 19	1.3 14	1.2 3	1.3 3	1.2 2
Learning Based Matting	**18.3**	19.3	17.8	17.9	0.8 11	0.8 12	0.8 8	0.1 3	0.2 3	0.2 3	0.2 9	0.2 2	0.2 10	0 19	0 13	0 15	0.6 38	1.1 39	2.1 40	0.2 9	0.3 6	0.2 9	22.5 27	13.3 27	1.3 15	4.3 38	5.9 40	8.5 43
Segmentation-based matting	**20.8**	21.9	20.8	19.9	0.8 14	1.7 36	0.8 9	0.3 13	0.4 11	0.4 16	0.2 13	0.2 9	0.2 11	0 21	0 4	0 14	0.1 25	0.3 26	1.1 33	0.2 20	0.3 10	0.3 26	29.5 36	38.5 44	1.8 23	3.1 33	2.8 26	3 27
KNN Matting	**20.9**	24.5	17.8	20.4	3.4 40	2.8 40	3.1 42	0.3 20	0.5 14	0.4 17	0.2 20	0.2 21	0.2 27	0 26	0 2	0.1 35	0.1 15	0.2 11	0.2 4	0.3 28	0.7 22	0.4 31	17.8 24	8 17	0.8 4	2.6 23	2 15	1.3 3
Improving Sampling Criterion	**21**	19.9	21.8	21.3	0.8 7	0.9 17	0.9 19	0.3 19	0.8 25	0.3 11	0.2 7	0.2 11	0.2 7	0 36	1.1 40	0 27	0.3 35	0.6 35	1.2 36	0.3 27	0.7 23	0.2 18	5.3 6	4.2 7	4.1 32	2.4 22	2.1 16	2.4 20
Sampling Based Image Matting Using NMF	**21.3**	18.9	22.8	22.4	1.1 33	1.2 30	1 31	0.3 16	0.6 17	0.3 7	0.2 8	0.3 30	0.2 29	0 2	0 12	0 2	0.2 34	0.2 17	0.2 6	0.2 14	0.8 26	0.4 32	12.5 14	14.7 31	7.9 40	2.9 30	2.2 19	3.4 32

Figure 4.18: Ranking of the matting algorithms according to the connectivity metric on the alpha matting benchmark [3], on the *20th* of *February 2016*. The proposed method is shown under the name 'TSPS Robust Sampling'.

Sum of Absolute Differences	overall rank	avg. small rank	avg. large rank	avg. user rank	Troll (Strongly Transparent) Input			Doll (Strongly Transparent) Input			Donkey (Medium Transparent) Input			Elephant (Medium Transparent) Input			Plant (Little Transparent) Input			Pineapple (Little Transparent) Input			Plastic bag (Highly Transparent) Input			Net (Highly Transparent) Input		
					small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user
anonymous_submission	**6.4**	7	4.6	7.6	12.3 13	15.5 3	13.5 3	5.6 2	7.4 3	7.2 5	4.2 2	5.3 5	3.8 8	1.8 12	3.4 5	2.6 7	6.5 14	9 11	13.9 22	4.7 8	7.7 4	6.3 6	20.5 4	19.8 2	17.9 6	**18.5** 1	23.2 4	22.7 4
Cluster-based Sampling Matting	**6.6**	6.1	8.3	5.5	**9.6** 1	14.5 2	13.3 2	5.7 8	7.8 5	7.2 3	4.6 10	5.8 16	3.5 2	1.5 5	4.6 15	2.7 8	6.2 8	8.3 7	11.9 10	4.3 6	9.4 13	6.2 5	22 9	21.9 7	20.6 13	19 2	**21.5** 1	**21** 1
LNSP Matting	**9.8**	6.9	9.1	13.5	12.2 9	22.5 28	19.5 33	5.6 3	8.1 6	8.8 22	4.6 11	5.9 18	3.6 4	1.5 6	3.5 7	3.1 17	6.2 9	8.1 4	10.7 5	**4** 1	**7.1** 1	6.4 8	21.5 7	20.8 4	**16.3** 1	22.5 9	24.4 5	27.8 18
Anonymous_TIP_submission	**10.4**	11.1	10.4	9.6	12.6 16	20.5 20	14.8 8	5.7 7	**7.3** 1	6.4 2	4.5 8	5.3 8	3.7 6	**1.4** 1	3.3 4	2.3 4	6.3 12	**7.9** 1	11.1 7	4.2 3	8.3 8	6.4 7	28.7 30	31.3 31	27.1 26	23.6 12	25.1 10	27.3 17
Trajectory	**10.8**	7.9	10.1	14.5	12.2 10	22.5 29	19.5 34	5.6 4	8.1 7	8.8 23	4.6 12	5.9 19	3.6 5	1.5 7	3.5 8	3.1 18	6.2 10	8.1 5	10.7 6	4 2	7.1 2	6.4 9	21.5 8	20.8 5	16.3 2	22.5 10	24.4 6	27.8 19
KL-Divergence Based Sparse Sampling	**11.2**	11.1	9.4	13.1	11.6 7	17.5 8	14.7 5	5.6 5	8.5 11	8 12	4.9 26	5.3 6	3.7 7	1.5 8	3.5 6	**2.1** 1	5.8 4	8.3 6	14.1 23	5.6 19	9.3 12	8 19	24.6 16	27.7 24	28.9 30	20.7 4	22.7 2	23.9 8
Comprehensive sampling	**12.8**	10.8	12.4	15.1	11.2 5	18.5 12	14.8 6	6.5 16	9.5 19	8.9 25	4.5 7	4.9 2	4.1 15	1.7 10	3.1 2	2.3 5	5.4 2	9.8 15	13.4 18	5.5 17	11.5 22	7.4 18	23.9 14	22 9	22.8 14	23.8 15	28 18	28.1 20
Iterative Transductive Matting	**13.1**	15	12	12.3	13.1 20	17.2 7	15.6 15	5.7 6	8.6 13	7.8 10	5.1 29	5.5 10	3.9 9	1.9 14	5.8 26	2.6 6	6.6 15	8.5 8	13.8 21	5.4 14	10 15	7.4 16	25.5 19	24 14	23.8 18	20.1 3	22.7 3	22.7 3
Comprehensive Weighted Color and Texture	**13.5**	14.3	13.5	12.9	14.6 25	16 4	15.7 16	6.8 22	10 25	7.9 11	4.3 5	5 3	4.1 13	1.7 9	3.5 9	2.2 2	**5.4** 1	9.9 17	12.8 15	4.3 5	7.4 3	**5.2** 1	28.3 29	28.1 25	25.4 22	24 18	30.2 22	28.7 23
SVR Matting	**13.6**	16.6	13	11.1	18.7 38	30.7 40	19.1 30	6.8 23	7.7 4	7.6 8	4.7 20	5 4	**3.4** 1	1.9 15	4.7 17	2.9 13	5.8 3	8.7 9	10.5 2	4.3 4	8 6	5.6 3	21.2 6	22.1 10	17.1 4	25.6 24	26.1 14	30.6 28
Sparse coded matting	**14**	17.4	14.1	10.5	13.7 23	25.8 36	14.8 9	6.4 13	8.2 9	**6.2** 1	4.7 14	5.4 9	4 11	1.8 13	**3.1** 1	2.3 3	5.9 6	8 3	10.6 3	4.5 7	8 5	5.5 2	30.3 32	33.1 34	29.2 31	27.7 31	27.2 16	29 24
TSPS Robust Sampling	**15.1**	13.1	14.9	17.4	11.4 6	17.5 9	13.7 4	6.8 21	9.7 24	9.5 30	4.7 15	5.6 13	4.3 23	1.5 2	3.6 10	3.5 28	7.7 25	10 18	13.3 17	6.1 23	10.8 18	8.5 23	18.9 2	20.7 3	17.7 5	23.4 11	31.4 24	24 9
CCM	**15.9**	18.9	15.5	13.3	13.8 24	20.8 21	16.9 25	6.4 14	8.9 15	8.2 16	4.7 16	5.9 20	3.6 3	2.5 26	4.3 13	3 14	7 18	9 10	10.6 4	4.9 10	8.1 7	5.7 4	25.6 20	27.5 23	24.5 19	25.4 23	26.4 15	28.2 21
LocalSamplingAndKnnClassification	**15.9**	18.6	13.1	16	12.6 15	16 5	**12.4** 1	5.8 9	8.1 8	8 13	4.5 9	5.5 11	4.1 12	2.2 22	5.1 20	3.4 26	8.1 29	10.5 20	15.6 28	7.3 29	12.3 27	9.4 26	24.1 15	21.8 6	19.7 9	24.7 21	24.8 8	25.9 13
Weighted Color and Texture Matting	**16**	13.9	17.5	16.5	13.1 21	17.8 10	15.8 17	6.5 15	9.4 17	8.6 19	4.2 4	**4.7** 1	3.9 10	1.7 11	6 27	2.7 9	6.4 13	11 24	16.3 29	4.8 9	9.1 11	6.5 10	23.7 12	24.8 16	23.2 16	26.5 26	40.2 34	28.5 22
Shared Matting	**16.8**	15.6	19.5	15.1	10.8 2	20.5 18	15 11	7.8 31	11.6 31	8.1 14	4.2 3	5.3 7	4.2 16	2.1 17	5.8 25	2.9 12	5.9 7	9.2 12	11.4 9	5 11	8.8 9	6.8 12	34.9 37	34.9 35	34.3 35	23.9 17	28.4 19	25.7 12
LNCLM matting	**17**	19.3	16.6	15.1	10.9 3	**11.2** 1	16.7 23	6.9 25	8.9 16	7.2 4	5.6 35	7 35	4.1 14	2.5 28	5.1 21	3.5 29	7.5 22	10 19	12 11	5.5 18	11.3 19	7.3 15	25.2 18	23 13	19.9 11	21.2 5	25 9	26.4 14
Anonymous SP_Lett_Subm	**17.4**	17	15.5	19.8	12 8	17.2 6	14.9 10	7.1 26	9.7 23	9.8 31	4.9 25	5.8 17	4.6 28	1.5 3	3.2 3	3.4 25	7.9 28	10.6 21	13.7 20	6.6 27	11.5 23	8.7 25	19.1 3	22 8	19.2 8	23.8 16	30.8 23	25.1 11
Global Sampling Matting	**18.7**	15.6	21.6	18.8	10.9 4	22.7 30	15.4 12	6.3 12	9.5 20	9 26	4.7 17	6.4 28	4.3 20	2.2 23	5.6 24	3.4 24	6.9 17	9.6 14	12.9 16	6.3 24	12.5 28	8.6 24	25.8 21	27.5 22	25.3 21	22 7	24.4 7	23.7 7
Segmentation-based matting	**19.9**	21.1	18.9	19.6	12.8 19	23.5 32	16.6 22	6.6 17	8.3 10	7.3 6	4.8 24	6.1 23	4.3 21	2.1 18	3.9 12	3.1 16	6.7 16	8 2	13.4 19	6 22	8.8 10	8.2 20	31.6 33	35.6 36	38.8 37	24.5 20	32 26	26.7 16

Figure 4.19: Ranking of the matting algorithms according to the SAD metric on the alpha matting benchmark [3], on the *20th* of *February 2016*. The proposed method is shown under the name 'TSPS Robust Sampling'.

Gradient error	overall rank	avg. small rank	avg. large rank	avg. user rank	Troll (Strongly Transparent) Input			Doll (Strongly Transparent) Input			Donkey (Medium Transparent) Input			Elephant (Medium Transparent) Input			Plant (Little Transparent) Input			Pineapple (Little Transparent) Input			Plastic bag (Highly Transparent) Input			Net (Highly Transparent) Input		
					small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user
anonymous_submission	**7.2**	6.5	6.8	8.3	0.2 15	0.2 4	0.2 9	0.2 4	0.3 7	0.3 2	**0.3** 1	0.4 8	**0.3** 1	0.3 2	**0.4** 1	**0.4** 1	1.2 3	**1.3** 1	2.5 8	1.1 4	1.6 2	1.3 3	1.4 12	1.4 17	0.9 12	0.5 11	0.7 14	0.7 30
KL-Divergence Based Sparse Sampling	**10.1**	9	7.8	13.5	0.2 3	**0.2** 1	0.2 5	0.2 3	0.2 5	0.3 23	0.4 6	0.4 4	0.4 5	0.3 6	0.6 10	0.4 2	1.7 11	1.9 7	2.9 21	1.7 19	2.3 15	2.3 21	1.4 11	1.3 9	1 19	0.6 13	0.6 11	0.5 12
Anonymous_TIP_submission	**10.2**	8.6	7.9	14.1	0.2 16	0.3 6	0.2 4	0.2 6	0.2 4	0.4 29	0.4 11	0.4 5	0.4 10	**0.2** 1	0.5 2	0.4 6	1.9 17	1.9 8	2.8 18	1.1 3	1.8 8	1.8 10	1.4 14	1.5 26	1.2 28	**0.4** 1	0.5 4	0.5 8
LNSP Matting	**10.5**	8.3	9.8	13.5	0.2 4	0.3 9	**0.2** 1	**0.2** 1	**0.2** 1	0.3 17	0.4 4	0.5 17	0.4 7	0.3 3	0.5 4	0.6 22	1.8 13	1.9 9	2.7 16	1.2 5	1.6 5	2 16	1.4 20	1.3 11	0.9 13	0.6 16	0.8 22	0.6 16
Comprehensive sampling	**11.5**	11.9	10.6	12	0.2 7	0.2 5	0.2 8	0.2 16	0.3 20	0.4 28	0.4 10	0.4 3	0.4 9	0.3 11	0.5 6	0.4 5	1.1 2	1.7 2	2.3 2	1.5 15	2.4 16	1.9 13	1.3 6	1.3 13	0.9 6	0.7 28	0.7 20	0.7 25
Trajectory	**11.5**	9.3	10.8	14.5	0.2 5	0.3 10	0.2 2	0.2 2	0.2 2	0.3 18	0.4 5	0.5 18	0.4 8	0.3 4	0.5 5	0.6 23	1.8 14	1.9 10	2.7 17	1.2 6	1.6 6	2 17	1.4 21	1.3 12	0.9 14	0.6 17	0.8 23	0.6 17
Cluster-based Sampling Matting	**13.1**	13.4	13.9	12	**0.2** 1	0.2 3	0.2 3	0.2 22	0.3 21	0.3 20	0.4 25	0.5 19	0.4 26	0.3 5	0.7 18	0.5 11	1.8 12	2 13	2.7 14	1.2 7	2.1 13	1.8 9	1.5 25	1.4 18	0.8 4	0.5 10	0.6 6	0.5 9
CCM	**13.6**	16.3	13.4	11.1	0.2 9	0.3 13	0.2 7	0.2 12	0.3 10	0.3 10	0.4 20	0.5 20	0.4 18	0.4 24	0.6 8	0.5 8	2.1 22	2.3 18	2.5 9	1.3 9	1.7 7	1.6 5	1.3 5	1.3 7	0.8 3	0.7 29	0.8 24	0.7 29
SVR Matting	**13.8**	16.5	14.8	10	0.3 31	0.4 35	0.3 22	0.3 30	0.3 14	0.3 12	0.4 13	0.5 16	0.4 6	0.4 14	0.6 14	0.5 9	1.5 8	2.1 15	2.5 7	1.2 8	1.6 4	1.5 4	1.3 3	1.2 5	0.8 2	0.7 25	0.7 15	0.6 18
Sparse coded matting	**13.8**	15.8	12.6	13.1	0.3 30	0.4 28	0.3 31	0.3 23	0.3 15	0.3 16	0.4 23	0.5 11	0.4 20	0.3 12	0.5 3	0.4 3	1.4 7	1.7 5	2.3 4	1.1 2	1.6 3	1.3 2	1.5 24	1.6 33	1.1 26	0.5 5	0.5 3	0.4 3
Segmentation-based matting	**14.3**	18	12.1	12.6	0.2 11	0.3 7	0.2 11	0.2 10	0.2 3	0.3 3	0.4 16	0.5 25	0.4 14	0.4 15	0.6 7	0.5 10	2.1 23	1.7 4	2.5 10	1.9 22	2.2 14	2.5 23	2 41	1.7 35	1.2 29	0.5 6	0.5 2	**0.4** 1
Global Sampling Matting	**14.7**	14.6	16.4	13	0.2 2	0.3 19	0.2 6	0.2 8	0.3 8	0.3 7	0.4 14	0.5 21	0.4 12	0.4 20	0.8 22	0.6 24	2.1 24	2.1 17	2.7 13	2 24	2.8 27	2.6 24	1.4 16	1.3 8	0.9 11	0.5 9	0.6 9	0.5 7
Improved color matting	**15**	16.8	16.3	12.1	0.2 12	0.3 12	0.2 14	0.2 14	0.3 12	0.3 4	0.4 15	0.5 24	0.4 17	0.5 27	0.7 21	0.5 18	2.5 28	2.5 25	2.7 15	1.9 21	2.5 20	2.4 22	1.4 15	1.4 15	0.9 5	0.4 2	**0.4** 1	0.4 2
Shared Matting	**15.2**	15.3	16.5	13.8	0.2 6	0.3 15	0.2 12	0.3 34	0.6 34	0.5 36	0.4 9	0.4 10	0.4 11	0.4 17	0.8 23	0.5 7	1.6 10	2 14	2.3 3	1.4 10	1.9 9	1.8 8	1.5 22	1.5 22	1.1 27	0.6 14	0.6 5	0.4 6
Comprehensive Weighted Color and Texture	**16.8**	16.5	19.1	14.9	0.4 37	0.4 36	0.5 38	0.4 36	0.5 30	0.5 31	0.3 3	**0.4** 1	0.4 2	0.3 10	0.6 11	0.4 4	**1.1** 1	1.8 6	2.4 5	1 1	**1.4** 1	**1.2** 1	1.7 37	1.8 38	1.2 33	0.5 7	0.9 30	0.4 5
TSPS Robust Sampling	**18.1**	20.4	13	21	0.2 8	0.3 11	0.2 10	0.3 26	0.3 13	0.4 27	0.5 28	0.4 7	0.4 25	0.3 8	0.6 12	0.7 32	2.7 31	2.3 19	3 24	2.2 29	2.6 23	2.7 27	1.3 9	**1.1** 1	0.9 8	0.7 24	0.7 18	0.5 15
Anonymous SP_Lett_Subm	**19.3**	21.4	14.9	21.5	0.2 10	0.3 8	0.2 13	0.2 21	0.3 17	0.3 21	0.5 31	0.4 9	0.4 27	0.3 7	0.6 9	0.7 30	2.7 30	2.6 28	3.1 26	2.4 30	2.7 25	2.7 26	1.4 10	1.1 2	0.9 9	0.8 32	0.7 21	0.6 20
LocalSamplingAndKnnClassification	**19.9**	21.4	19.3	19	0.2 20	0.3 18	0.2 16	0.2 5	0.3 6	0.3 6	0.4 8	0.4 6	0.4 4	0.4 21	0.7 17	0.5 19	2.9 35	3.1 32	3.9 35	2.5 31	3.2 33	2.9 31	1.6 29	1.5 23	1 20	0.7 22	0.7 19	0.6 21
Global Sampling Matting (filter version)	**20.1**	18.4	20.1	21.8	0.2 13	0.4 34	0.3 23	0.3 32	0.4 26	0.4 30	0.5 33	0.5 27	0.5 34	0.6 34	0.8 25	0.7 33	1.3 5	1.7 3	**2.2** 1	2 23	2.7 24	2.2 19	1.3 4	1.2 6	1.2 30	0.4 3	0.7 16	0.4 4
LNCLM matting	**20.5**	22.6	20.5	18.5	0.2 14	0.2 2	0.3 19	0.3 29	0.4 25	0.3 15	0.5 35	0.6 37	0.5 32	0.5 26	0.7 16	0.6 20	2.1 25	2.6 27	2.9 19	1.5 13	2.4 17	2 14	1.6 27	1.6 28	0.9 10	0.6 12	0.6 12	0.6 19

Figure 4.20: Ranking of the matting algorithms according to the gradient metric on the alpha matting benchmark [3], on the 20^{th} of *February 2016*. The presented technique, shown under the name 'TSPS Robust Sampling', achieved the first position in the second trimap of the image named 'Plastic bag'.

better than [73]. For the MSE, the overall rank of the presented technique is higher than that of the method in [12]. In 13 cases (out of 24), my algorithm performs better than [13] and [73]. Finally, for the SAD metric, in 10 cases (out of 24), my algorithm performs better than [73] and [13]. The use of the extended cost function resulted in consistent enhancement for the rank of the presented algorithm across all the metrics.

The performance of the presented techniques using the initial and the extended cost function was compared to that of the methods in [12] and [73]. The bar-charts and the tables depicting the comparison are shown in Fig. 4.21, Fig. 4.22, Fig. 4.23 and Fig. 4.24. For the bar charts in Fig. 4.21 and Fig. 4.22, errors for image '16' and image '25' were omitted because they are much higher than the errors in the rest of the images for all the algorithms; including them would impact negatively the clarity of presentation of the rest of the results. Nevertheless, the results of image '16' and image '25' can be found in the tables shown in Fig. 4.23 and Fig. 4.24.

Further experiments were also carried out on the use of cartoon-texture distortion, instead of the chromatic distortion. The cartoon-texture distortion was formulated as follows: Following the cartoon-texture decomposition model, a pixel I_i in the input image I can be expressed as:

$$I_i = C_i^s + T_i^s, \tag{4.10}$$

where C_i^s is the cartoon component of the pixel I_i calculated at the texture scale s, and T_i^s is the texture component of the pixel I_i calculated at the texture scale s. The matting compositing equation for the same pixel is given by

$$I_i = \alpha F_i + (1-\alpha) B_i. \tag{4.11}$$

Thus, the matting compositing equations for the cartoon and texture components at scale s are given by

$$C_i^s = \alpha F_i^{C^s} + (1-\alpha) B_i^{C^s}, \tag{4.12a}$$

$$T_i^s = \alpha F_i^{T^s} + (1-\alpha) B_i^{T^s}, \tag{4.12b}$$

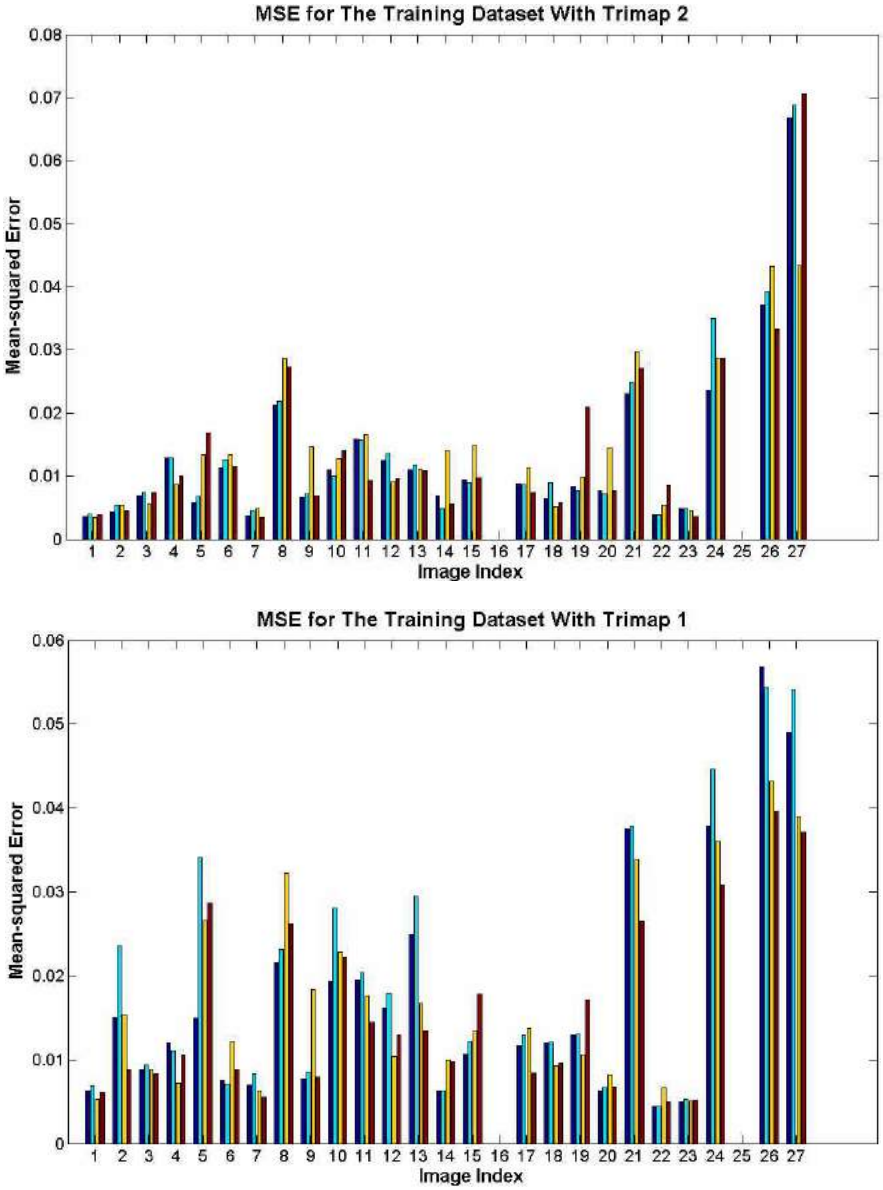


Figure 4.21: Objective comparison (MSE) of the performance of the presented matting techniques with the performance of the approaches in [12] and [73], on the training dataset of [3]. The blue, cyan, yellow and red bars represent my method with the extended cost function, my method with the chromatic distortion cost function, the method of [12] and the method of [73] respectively.

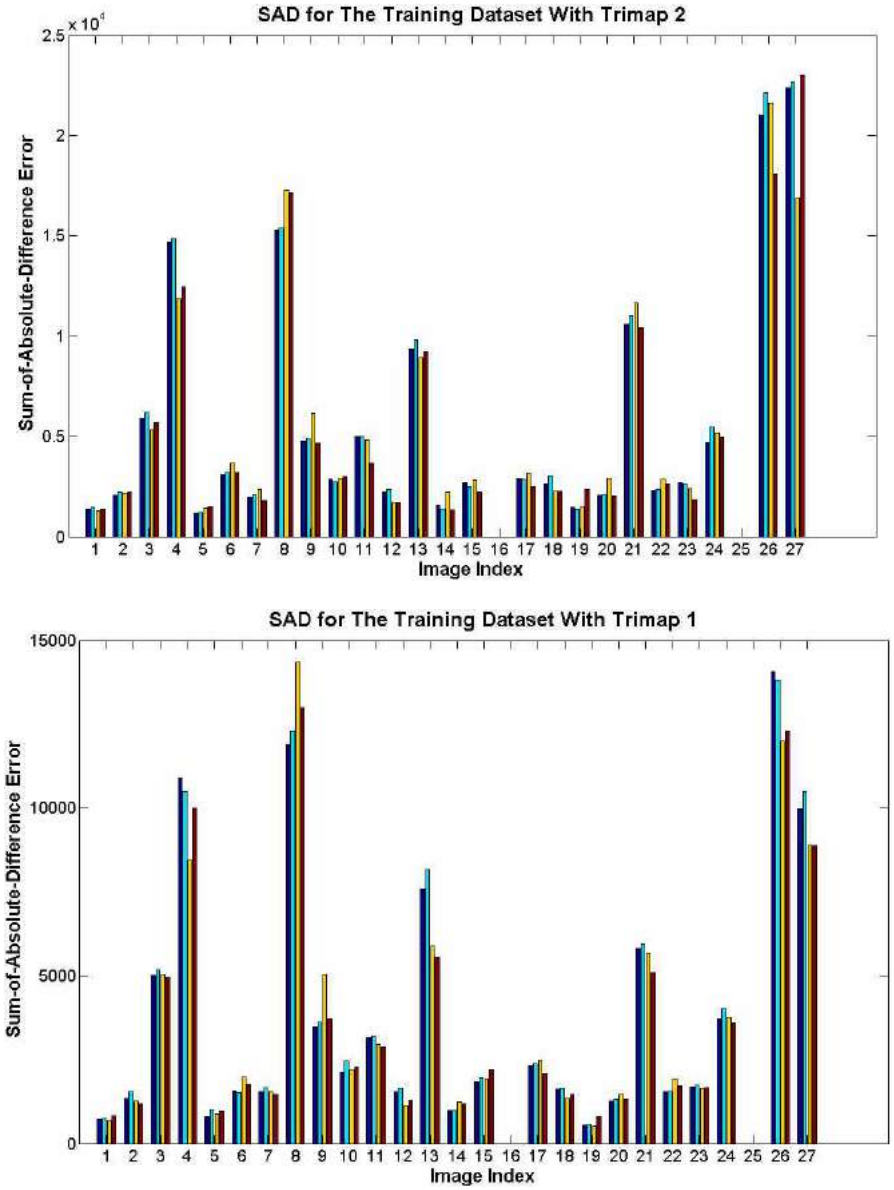


Figure 4.22: Objective comparison (SAD) of the performance of the presented matting techniques with the performance of the approaches in [12] and [73], on the training dataset of [3]. The blue, cyan, yellow and red bars represent my method with the extended cost function, my method with the chromatic distortion cost function, the method of [12] and the method of [73] respectively.

MSE, Trimap 2			
TSPS, Cost function 2	TSPS, Cost function 1	Weighted Color-Texture	Comprehensive Sampling
0.0036	0.004	0.0035	0.0039
0.0042	0.0054	0.0054	0.0045
0.0069	0.0075	0.0056	0.0074
0.0129	0.0129	0.0086	0.01
0.0059	0.0068	0.0134	0.0168
0.0114	0.0126	0.0134	0.0115
0.0037	0.0045	0.0048	0.0035
0.0213	0.0219	0.0286	0.0273
0.0066	0.0072	0.0146	0.0069
0.011	0.01	0.0127	0.0142
0.0158	0.0157	0.0166	0.0093
0.0124	0.0136	0.009	0.0096
0.0109	0.0117	0.0111	0.0108
0.0068	0.0048	0.0141	0.0057
0.0095	0.0089	0.0149	0.0097
0.3191	0.3191	0.3215	0.3229
0.0088	0.0087	0.0114	0.0074
0.0065	0.0089	0.0051	0.0058
0.0084	0.0077	0.0099	0.021
0.0077	0.0071	0.0145	0.0078
0.0231	0.0249	0.0298	0.0271
0.0038	0.0039	0.0053	0.0086
0.0048	0.0048	0.0045	0.0036
0.0237	0.035	0.0287	0.0287
0.0835	0.0902	0.0678	0.109
0.0371	0.0393	0.0433	0.0334
0.0668	0.0687	0.0435	0.0705
9	5	7	6

(a)

MSE, Trimap 1			
TSPS, Cost function 2	TSPS, Cost function 1	Weighted Color-Texture	Comprehensive Sampling
0.0063	0.0069	0.0053	0.0061
0.015	0.0236	0.0154	0.0088
0.0089	0.0093	0.0088	0.0083
0.012	0.0111	0.0072	0.0105
0.0149	0.0341	0.0267	0.0287
0.0075	0.0071	0.0122	0.0089
0.007	0.0084	0.0063	0.0056
0.0217	0.0231	0.0322	0.0263
0.0077	0.0086	0.0183	0.0079
0.0193	0.0281	0.0228	0.0223
0.0194	0.0203	0.0176	0.0145
0.0162	0.0179	0.0104	0.0129
0.0249	0.0295	0.0167	0.0135
0.0063	0.0063	0.0099	0.0098
0.0107	0.0122	0.0135	0.0178
0.214	0.2227	0.3164	0.1891
0.0117	0.0129	0.0137	0.0084
0.012	0.0121	0.0092	0.0096
0.013	0.013	0.0106	0.0171
0.0063	0.0067	0.0083	0.0068
0.0376	0.0379	0.0339	0.0265
0.0044	0.0045	0.0067	0.0049
0.005	0.0053	0.005	0.0051
0.0379	0.0446	0.0361	0.0308
0.1312	0.134	0.0811	0.1217
0.0568	0.0544	0.0432	0.0396
0.049	0.0541	0.039	0.0371
8	2	6	11

(b)

Figure 4.23: MSE of the presented matting approach with the two suggested cost functions and the approaches in [12] and [73], on the training dataset of [3]. The first, second, third and fourth coloumns represent my method with the extended cost function, my method with the chromatic distortion cost function, the method of [12] and the method of [73] respectively. The results with Trimap 2 is shown in (a), while the results with Trimap 1 is shown in (b). The last row is a count of the number of images where each technique attained the least error.

SAD, Trimap 2			
TSPS, Cost function 2	TSPS, Cost function 1	Weighted Color-Texture	Comprehensive Sampling
1379.8	1486.7	1319.1	1399.2
2070.9	2238.4	2201.3	2248
5889.9	6180.1	5348.1	5686.2
14672	14839	11846	12446
1149.9	1197.7	1416.8	1527.1
3067.3	3204.5	3694.7	3211.6
1963.2	2107.7	2358.6	1814.1
15273	15401	17268	17129
4785.2	4938.8	6149.9	4668.8
2864.1	2745	2929.2	3010.8
5004.5	4987.3	4834	3697.2
2231	2379.6	1713.7	1737.3
9342.3	9819.8	8913.5	9218
1605.7	1388.2	2230.4	1330.2
2708	2504.7	2828.9	2241.6
32468	32468	33214	32289
2928.6	2895.9	3155.8	2491.2
2660.4	3032.6	2257.7	2290.5
1468.2	1375.7	1532.5	2350.8
2075.4	2087.5	2903.4	2035.5
10566	11003	11658	10395
2329.1	2368.7	2874.6	2655.9
2694.1	2678.4	2395.6	1862.2
4709.6	5487.8	5163.9	4940.8
12755	13564	10473	15781
21028	22122	21615	18089
22383	22670	16874	23015
6	2	8	11

(a)

SAD, Trimap 1			
TSPS, Cost function 2	TSPS, Cost function 1	Weighted Color-Texture	Comprehensive Sampling
731.66	760.29	709.33	825.74
1329.6	1554.6	1254.2	1176.2
5018.1	5176.7	5047.7	4945.3
10909	10484	8452.9	9997.7
814.86	1010.4	888.51	949.26
1561.8	1525.1	1990.7	1788.3
1536.5	1681.5	1547.1	1459.1
11890	12299	14344	12988
3485.7	3604.8	5059.3	3707.4
2133.1	2462.1	2212.7	2276.4
3145.7	3218.2	2946.8	2863.6
1545.3	1650.3	1130.7	1285.6
7570.7	8165.1	5880.4	5546.8
993.68	988.63	1239.3	1190.8
1860.1	1948.6	1921	2210.9
19650	19736	27834	17420
2311.4	2398.6	2502.2	2075.7
1618.8	1645.3	1331.8	1476
557.39	563.38	512.97	801.6
1275.6	1320.4	1463.2	1328.7
5807.6	5951.5	5677.2	5090.3
1547.2	1546.5	1930.3	1713.1
1703.3	1754.1	1649.2	1661.7
3725.7	4035	3759.1	3601.1
11435	11640	8403.9	12206
14075	13799	11993	12296
9968.5	10498	8870.5	8870.6
6	3	9	9

(b)

Figure 4.24: SAD of the presented matting approach with the two suggested cost functions and the approaches in [12] and [73], on the training dataset of [3]. The first, second, third and fourth coloumns represent my method with the extended cost function, my method with the chromatic distortion cost function, the method of [12] and the method of [73] respectively. The results with Trimap 2 is shown in (a), while the results with Trimap 1 is shown in (b). The last row is a count of the number of images where each technique attained the least error.

which bring another constraint to the calculation of the best alpha for every pixel I_i, instead of the chromatic distortion only. Thus, the cartoon-texture objective function to be minimized can be expressed as:

$$\xi = \|C_i^s - \alpha F_i^{C^s} + (1-\alpha) B_i^{C^s}\| + \|T_i^s - \alpha F_i^{T^s} + (1-\alpha) B_i^{T^s}\|. \tag{4.13}$$

By varying the parameter s, more constraints can be brought to the problem of picking the best FG/BG pair for the pixel I_i in image I. However, the experiments were carried out on a single value for s only. The motivation behind adopting this distortion and its mathematical formulation were presented for the sake of completeness. However, the cartoon-texture distortion when adopted with the training dataset (of the matting benchmark) tended to produce alpha maps with much higher mean-squared error rates compared to the chromatic distortion, and thus presenting the alpha maps obtained with this error function (cartoon-texture distortion) was seen to be of minimal benefit.

This chapter started by highlighting the motivation behind sequential pair selection – the core contribution of this thesis to the area of efficient sampling of trimaps. Following the motivation, I presented the first variant of sequential pair selection. Since it relied on the quantification of the overlap between the color distributions of the shortlisted pair space and the unknown region under consideration, this method suffered when the FG and the BG color distributions overlap. Evaluating a large number of pairs at the last stage of the algorithm's pipeline is also another negative aspect in this variant. Objectively, this method attained the 34^{th} and the 28^{th}, according to the SAD and the MSE metrics respectively, over the 43 methods in the matting benchmark; this ranking was reported on the 1^{st} of August 2016, leaving a lot of room for improvement. Hence, the second variant of sequential pair selection was presented. I proposed to find a suitable half-pair for every unknown super-pixel by solving a binary graph transduction problem. The cartoon-texture decomposition of a few delegates, rather than all the constituent pixels, of a super-pixel were considered in this stage, overcoming the problem of overlapping color distributions, and cutting back the computational burden of considering all the pixels in an image. Following the pair space punching, the gathered pairs were then assessed using the classical chromatic distortion and a newly-proposed sparsity-promoting cost function. The performance of the second variant of sequential pair selection was then evaluated

subjectively and objectively on the dataset of the matting benchmark. It showed that its performance is close to the SoA, while overcoming critical drawbacks in the literature such as the color ambiguity problem.

Compared with the first variant of sequential pair selection, the performance of the second method is remarkably better. Nevertheless, according to the method's ranking on the benchmark, more room for improvement is still available. This constitutes the motivation for the third technique, which is the subject of the next chapter.

Chapter 5

Natural Image Matting Using Iterative Graph Cuts With Half-pair Constraints

Even though the research presented in this Chapter is quite pertinent to the scope of research in the previous one, I discuss it separately for the sake of clarity of presentation. In this Chapter, I highlight a novel pipeline for sampling-based matting which yielded, in some cases, better results than the methods explained in Chapter 4. Based on the main premise of this thesis with regards to sampling-based matting, namely, the benefit of sequential pair selection, I formulate the problem of FG/BG pair assessment (which involves assigning each unknown pixel a FG/BG pair) as a graph labelling problem. In such a problem, I can encode the half-pair constraints, that were discussed in the beginning of Chapter 4, as smoothness constraints, and then solve it using multi-label graph cuts. Through the sections of this chapter, I explain each stage of the proposed pipeline that is depicted in Fig. 5.1.

5.1 Choosing Delegates for Super-pixels

I start the algorithm by calculating the Simple Linear Iterative Clustering (SLIC) super-pixels [86, 87] (region size=10 and regularizer=0.1) and choosing a few delegates to represent each of them. Towards the goal of constructing a $1 \times N$ vector that represents a

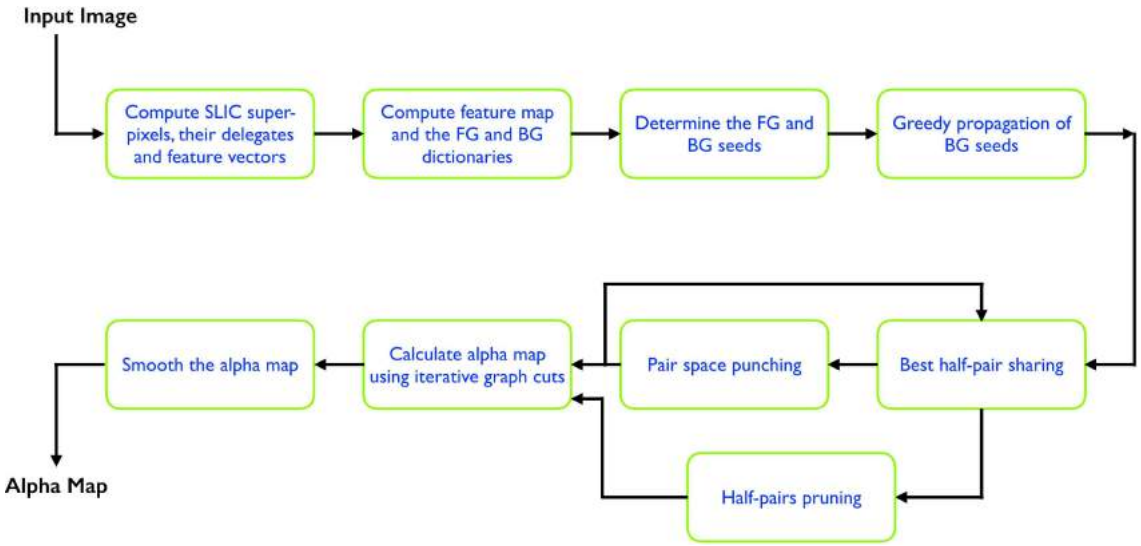


Figure 5.1: The pipeline of the proposed algorithm for natural image matting.

feature vector for every SP, where N is the number of delegates, I sought an equal number of delegates for every SP. Thus, the delegates nomination procedure that was discussed in sub-section 4.3.1 could not be adopted. It was controlled by a budget (a ceiling) but in general it yielded a variable number of delegates depending on the variety of the color feature of the member pixels of a SP.

I chose a fixed number of delegates ($N = 10$ in my experiments) for every super pixel using the following procedure. Throughout this procedure and the rest of the pipeline, the CIEL*a*b feature was used. I measured the variance across the three color channels for every member of the super-pixel under consideration, and I sorted them accordingly; this implies a sorting index $I^m_{s_1}$ on the members. I also measured the variance across each channel; this implies a sorting index $I^c_{s_2}$ on the channels. I multiplied the two indices, which implies a composite index for every color channel, for every member of the super-pixel, $I^{mc}_{s_3}$. Then, I summed the composite index, over the three color channels, for every member and sorted the result, which yields the final sorting index $I^m_{s_4}$ on the members of the super-pixel. This can be formally written as:

$$I^m_{s_4} := \text{sort}\left\{\sum_{j=1}^{3} I^{mc_j}_{s_3}\right\}, \tag{5.1}$$

where c_1, c_2 and c_3 are the three color channels respectively. Finally, I divided the sorted

members into N subgroups and nominated the first member of every sub-group to be a delegate. This constitutes the N delegates of every super-pixel. The feature vector of every super-pixel is then constructed as the $N \times 3$ CIEL*a*b vector.

5.2 Constructing the FG and BG Dictionaries

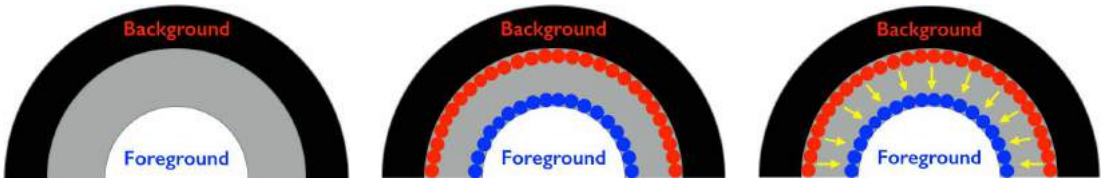
I adopted a procedure for downsizing the two sets of all the known FG and BG pixels in the trimap; the output of this step is the FG and BG dictionaries. The procedure was guided by the feature maps of [85], which were computed using their publicly available code.

The generated feature map is an image whose size is equal to that of the original image, and each pixel has a value representing the probability of being an edge pixel. I thresholded these feature maps (threshold is 0.2 in my implementation; the higher it is, the smaller the dictionaries). I then loop over every label (SLIC super-pixel) that contains one or more edge pixels and add to the FG(BG) dictionary the spatially-nearest FG(BG) super-pixel to it. The reduction in the number of FG and BG labels using the aforementioned procedure is key to alleviating the computational load of the pair space punching step, which was previously discussed in sub-section 4.3.3.

5.3 Good Half-pair Computation and Sharing

In this stage of the algorithm, I seek a good half-pair for every unknown super-pixel. Since the certainty about the suitability of a half-pair is inversely proportional with the distance of the unknown pixel under consideration from the borders of the trimap, I started by computing a suitable half-pair for SPs that lie nearby in space to the borders of the trimap. Then I propagate this information to the deeper regions of the matting band. This is illustrated in Fig. 5.2 and I elaborate on it below.

Starting with the FG and BG dictionaries, and all the unknown super-pixels in the image, I calculate the 15 most similar known labels to every unknown label. The similarity is measured as the Euclidean distance between the super-pixels' feature vectors; the construction of the super-pixel feature vector was described in sec. 5.1. I then take a vote for every unknown super-pixel regarding whether it favours FG super-pixels or BG



(a) Example of a Trimap (b) Seed Computation (c) Half-pair Propagation

Figure 5.2: An illustration of the stages of half-pair computation: FG/BG seed determination and BG seed propagation.

super-pixels. The consensus factor in my experiments is equal to 13 (out of the computed 15 neighbours). Only those super-pixels which chose 13 or more FG (or BG) super-pixels are nominated to be *preliminary seeds*.

I then loop over the preliminary seeds and follow this procedure:

- If a preliminary seed (an unknown SP) preferred FG super-pixels, I check whether there is a member of the BG dictionary lying nearby in space to it or not (a maximum of 20 pixels apart in my experiments, because the radius of the SLIC pixels is 10 pixels). If this conditions holds, I assign that BG dictionary member to the preliminary seed under consideration as its suitable half-pair; note that this would be the dissimilar nearby half-pair.

- If a preliminary seed (an unknown SP) preferred FG super-pixels and there is not a BG dictionary member lying nearby, I check whether there is a member of the FG dictionary lying nearby in space to it or not. If this condition holds, I calculate the Euclidean distance D between the feature of the nearby FG super-pixel and that of the most similar FG super-pixel (among the 15 neighbours computed initially) to the preliminary seed under consideration. If $exp(-D) > 0.85$, I pair the preliminary seed under consideration to the most similar FG super-pixel to it (among the 15 neighbours computed initially).

The same logic holds for the preliminary seeds that preferred BG super-pixels. At the end of this stage, two sets of already-paired unknown super-pixels are acquired, namely, the FG seeds and the BG seeds, which are shown in Fig. 5.2(b) as blue and red circles respectively.

This seed information is propagated greedily to the inside of the matting band, the

process that is depicted as yellow arrows in Fig. 5.2(c). In the current implementation, I allowed the propagation from the BG seeds only (but not the FG seeds), which was enough to bring good half-pairs to the super-pixels inside the matting band. In the last sentence, and at this stage, the goodness of the computed half-pair was judged subjectively; nevertheless, a procedure similar to the one followed in sub-section 4.3.2 can be adopted to assess the computed half-pairs objectively.

As mentioned earlier, the BG seed propagation takes place in a greedy fashion. Basically, I loop over the BG seeds and determine whether there are any unknown super-pixels, nearby in space, that have not been paired yet. The same 20 pixels limit was used again in this stage because the radius of the SLIC pixels is 10 pixels. If this condition holds, all the nearby un-paired super-pixels are assigned the same half-pair of the neighbouring BG seed. This procedure is iterated until one of two conditions is satisfied, either no more un-paired super-pixels left, or when more iterations fail to pair the un-paired. The latter case happened in my experiments but the probability of its occurrence (measured as a percentage of the total number of unknown super-pixels) is very low. If that case took place, I loop over those super-pixels which could not be paired using the aforementioned procedure and pair every one of them to its most similar super-pixel among those that were successfully labelled during the propagation step.

With all the unknown super-pixels now paired to a suitable half-pair, I then allowed half-pair sharing according to the following criteria. In addition to its half-pair, every super-pixel among the FG(BG) seeds is paired with the half-pairs of the most similar M neighbours among the FG(BG) seeds. The similarity is measured as the Euclidean distance between the super-pixels' feature vectors. For the super-pixels that are non-seeds, i.e., they get their half-pairs during the propagation step, they are paired with the half-pairs of the most similar $\frac{M}{2}$ neighbours among the BG seeds and the most similar $\frac{M}{2}$ neighbours among the non-seed super-pixels.

5.4 Punching the Pair Space and Half-pair Pruning

The algorithm adopted for punching the pair space will be explained for a case where the best half-pairs, that were computed in the first stage, for the unknown region under

consideration are FG labels. Nevertheless, the same logic holds for the case where the half-pairs are BG labels.

For the unknown super-pixels whose half-pairs are FG labels, I need to punch the 2D pair space along the BG direction, the vertical direction in Fig. 4.11 as an example. Towards that goal I computed the mean CIEL*a*b value of every label in the BG dictionary; each mean will represent a B_j in Eqn. 5.2a. It is worth mentioning that this is the extended cost function used in Chapter 4. I also computed the mean color value of the unknown super-pixel under consideration; this will represent I in Eqn. 5.2a. Finally, the color feature of the delegates of all the FG half-pair labels will represent the F_i in Eqn. 5.2a. For every FG delegate, I determined the BG label that minimizes the right-hand side of Eqn. 5.2a and I chose the 5 most frequent (mode) BG labels in the set containing all the best BG labels of all the delegates.

$$\xi = \frac{\|I - (\hat{\alpha}F_i + (1-\hat{\alpha})B_j)\|}{\|F_i - B_j\|} \times \min\left\{\frac{W_1}{W_1+W_2}, \frac{W_2}{W_1+W_2}\right\} \quad \text{where,} \tag{5.2a}$$

$$W_1 = \exp(-\|I - F_i\|), \quad W_2 = \exp(-\|I - B_j\|). \tag{5.2b}$$

As a quick recall, the above cost function is the chromatic distortion divided by a denominator that encourages well-separated FG and BG labels, and multiplied by a term that encourages sparsity in alpha maps, by ensuring that either F_i or B_j are very close (in the feature space) to the mean color feature of the unknown super-pixel under consideration.

Following the pair space punching step, I pruned the initially calculated set of half-pairs. So far, for every unknown super-pixel, I have had $M + 1$ half-pairs, that is: its best half-pair plus the M shared ones, in addition to 5 labels computed from punching the pair space. Assuming the best half-pairs are FG labels, each of them will be given the symbol F_h, and the BG labels computed from the punching step will be given the symbol B_p. Along the same lines of encouraging well-separated FG and BG labels, I retained only one FG label from the F_h that maximizes the function given by

$$F^* := \underset{F_h}{\operatorname{argmax}} \frac{2}{P} \sum_{p=\frac{P}{4}+1}^{\frac{3P}{4}} \parallel F_h - B_p \parallel \quad \forall\, p = 1, 2, ..., P. \tag{5.3}$$

The above function is the inter-quartile mean of the Euclidean distances between one member of the F_h and all the B_p associated with the unknown pixel under consideration.

5.5 Calculating Alpha Maps Using Iterative Graph Cuts

In this stage, I use the information about suitable half-pairs and punching labels that were computed along the pipeline so far to solve a graph labelling problem for pair assessment. While constructing the graph, I encode the half-pair constraints in the smoothness term of the energy function to be minimized as will be shown below.

I start by tiling the image with semi-overlapping square patches of size 100×100. Then, I determine (using the trimap) which tiling patches contain unknown pixels. I loop over those patches, construct a graph for each of them, solve a graph labelling problem using multi-label graph cuts [90, 55, 52, 91] and compute an alpha map for every patch. Since the patches are overlapping, some unknown pixels will be covered more than once, so I averaged the alpha values I get from all the labelling problems containing a particular unknown pixel. In the following paragraphs, I explain the graph construction of a single patch; this process is done in the same way for all the patches.

While constructing a patch's graph, I considered only the unknown pixels in it as the graph sites. I tried to include the known pixels as well, if any, and fix their label (as a constraint or a boundary condition), but the computational cost was high since the complexity of graph cuts is $\mathcal{O}(NL)$ where N are the number of sites and L are the number of labels. After determining the unknown pixels in the patch, I constructed a 4-neighbourhood system. The labels of that graph problem are the FG/BG pairs of delegates that represent the half-pair labels + punching labels of every unknown super-pixel within the patch under consideration. Basically, I check the SLIC label of every unknown pixel in the patch, retrieve its best half-pair and punching labels, then I construct graph labels by concatenating the feature of the delegates of the half-pair and the feature of the delegates

of the punching super-pixels (so, a graph label is a 6×1 vector). The graph sites share all the available graph labels. The energy function that I minimize is given by

$$E(L) = \sum_{p_i \in \Omega} E_d(L(p_i)) \quad + \sum_{(p_i,p_j)\,\forall\, p_i \in \Omega, p_j \in \Omega} E_s(L(p_i), L(p_j)), \tag{5.4}$$

where the first summation is over all the unknown pixels in the patch Ω, while the second summation is over all the set of pairs of 4-neighbouring pixels among the unknown pixels in the patch. The term E_d is the chromatic distortion, while the term E_s is a Potts cost, i.e. a fixed cost whenever any neighbouring pixels are assigned a different label (FG/BG pair). Setting E_s is a critical issue in the implementation; I elaborate on it below.

Using Potts model is a hard half-pair constraint. It should suffice to make sure that the two labels assigned to any two neighbouring pixels have half-pairs of the same affiliation, i.e., both half-pairs are BG pixels or both are FG pixels. I tried this formulation in the smoothness cost, but the results were of low visual quality. So, I adopted a Potts cost which inherently implies a half-pair constraint, i.e. labels of neighbouring pixels not just should have the same half-pair or the affiliation of the half-pair (the label of its mother super-pixel), they should be the same for E_s to be zero. Last but not least, while constructing E_d, I set the cost of any FG/BG pairs that yield $\alpha < -0.2$ or $\alpha > 1.2$ with a particular graph site to a very high cost. This way, I hamper the optimization algorithm to assign those costly pairs to that particular graph site.

5.6 Pre and Post-processing, Results and Discussion

All the experiments were implemented using Matlab®, and were run on a PC with Intel Core2Quad 2.66GHz processor and 4GB of RAM. I have adopted the same pre-processing steps that I discussed in sub-section 4.3.5 regarding trimap expansion. I have also adopted the same post-processing method with one exception which is the setting of the confidence value f. In this algorithm, f of a particular unknown pixel under consideration was set as the mean of the chromatic distortions computed for that particular unknown pixel in all the labelling problems that involved it (in all the tiling patches that cover it).

Prior to discussing the results, it should be noted that the algorithm suffers from high time complexity; some large images, especially with wide-brush trimaps, took close to

one hour to finish. Nevertheless, it has some notable merits that I will show through the discussion of the results. All alpha maps that were acquired by this algorithm for the testing dataset of [3] are presented in sec. A.6 of the appendix. Throughout the following discussion, I will refer to the method in this Chapter as *GHC* matting – a shorthand for Graph Cuts With Half-pair Constraints.

I start my presentation by showing the performance of the algorithm on one of the most challenging images in the matting dataset, the 'Net' image. What justifies my claim that 'Net' is one of the most challenging images are the MSE and SAD values attained by the SoA methods on it, compared to the MSE and SAD values on other images. Figure 5.7 depicts a visual comparison (will be succeeded by objective evaluation) between the performance of the proposed algorithm and the method presented in Chapter 4. The first row of the figure shows the original image. The second row shows Trimap 1 and the results of the aforementioned techniques before smoothing, the image on the right is the result of GHC matting. The third row shows results after smoothing, while the fourth and the fifth rows show the results for Trimap 3. On the smoothed results of the method discussed in Chapter 4, visually obvious artifacts are pointed to by white arrows. The results of GHC matting do not contain such artifacts. It is worth mentioning that for those two trimaps (Trimap 1 and Trimap 3), the MSE ranking of GHC matting is **worse** than the method of Chapter 4, even though the former is visually better.

In Fig. 5.4 through Fig. 5.6, I present the ranking tables of the benchmark for the connectivity, SAD and MSE metrics respectively. The ranking table of the gradient metric was not included because the ranking of GHC matting was bad enough that the position of the algorithm is far down the table, and to show it, the table would not fit in the page. Nevertheless, according to the connectivity metric, GHC tops the method in Chapter 4 and several SoA techniques. The proposed matting technique has also succeeded to achieve the lowest SAD (1^{st} position) over all the techniques in the literature in Trimap 1 of the image 'elephant'. Moreover, compared to the method in Chapter 4, GHC matting attained less SAD in 11 cases out of the total of 27 testing cases, and for the MSE metric, it was better in 8 cases out of the total of 27 testing cases. Last but not least, Fig. 5.7 shows an objective performance comparison, on the training dataset of [3], between GHC matting and the method in Chapter 4 with its two cost functions used for pair assessment

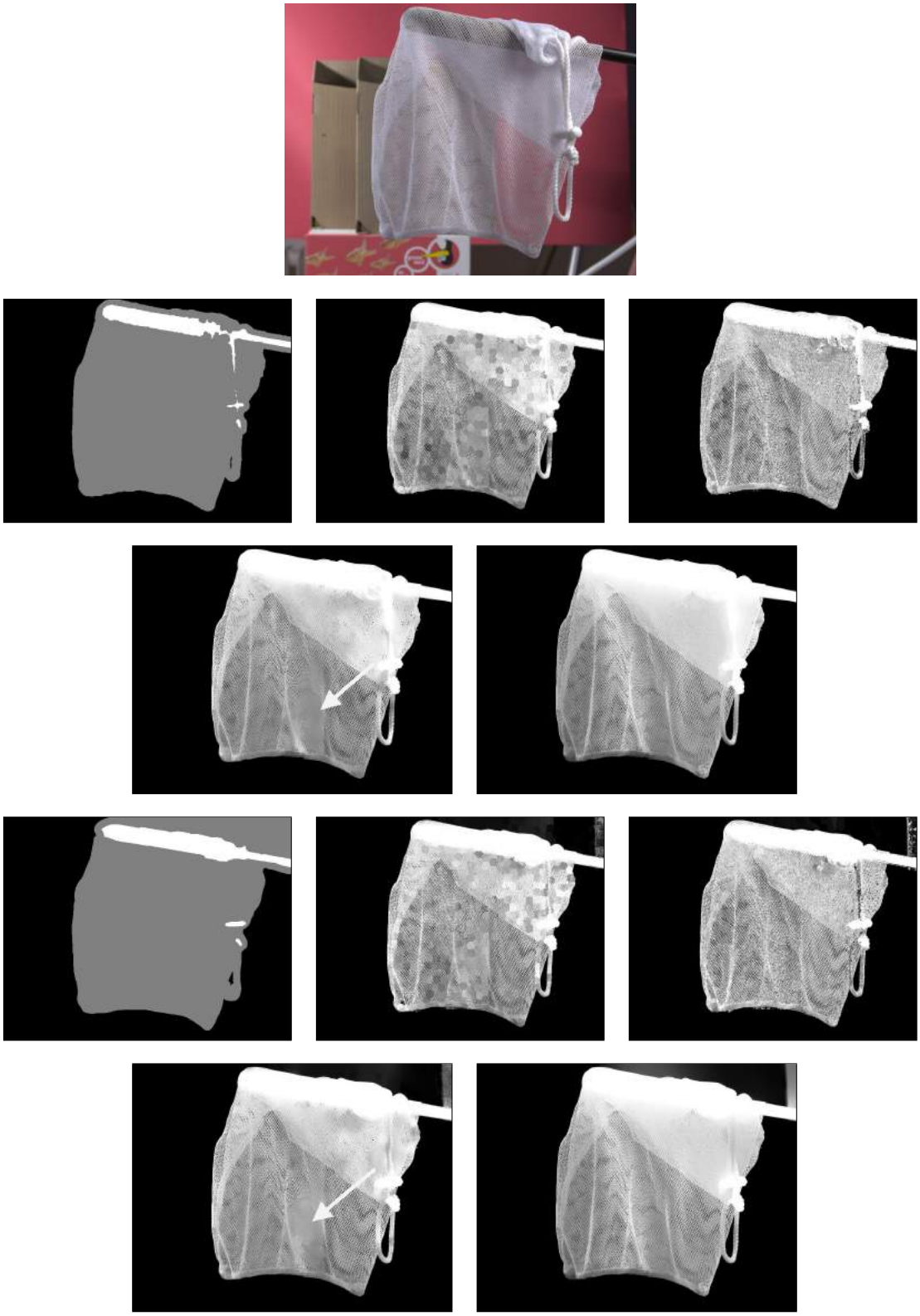


Figure 5.3: The challenging image 'Net' of [3] and a visual comparison between the results of the matting technique proposed in Chapter 4 with the chromatic distortion as the cost function and the GHC matting. The first row shows the original image, second row shows Trimap 1 and the results of the aforementioned techniques before smoothing, third row shows results after smoothing, fourth row shows the results for Trimap 3 and fifth row shows its smoothed alpha maps for the two techniques. Obvious artifacts are pointed to by a white arrow. Please see text for more details.

(chromatic distortion and the extended version of it, please check sub-section 4.3.6). I computed the MSE and SAD for the 27 images (with Trimap 1) and excluded image 16 and image 25 for the clarity of the bar chart – for those two images, the SAD and MSE values are higher than other images, and plotting the bars corresponding to them results in very short bars for the other images. By analyzing the bar charts, it can be seen that GHC matting attained the least SAD and MSE in 40% of the cases (10 cases out of 25).

In this chapter, I proposed another variant of the sequential pair selection strategy. First, foreground and background dictionaries are gathered by allowing the feature map and the trimap of the input image to communicate. Iterative and confidence-driven half-pair computation then took place, followed by pair space punching. The pipeline was then concluded by an iterative graph cuts step where half-pair constraints were embedded in the smoothness term of the energy function to be minimized. Because the neighbouring pixels were penalized if chose different FG/BG pairs (labels), the final alpha maps computed using this method were in some cases better than those computed using the second variant of sequential pair selection – the method that was discussed in the previous chapter. Nevertheless, the overall performance of the third variant, as indicated by the objective evaluation on the training and testing datasets of the matting benchmark, was worse than the second variant. Hence, the second variant is recommended over the other two variants. Future research directions will be highlighted in Chapter 8.

Trimaps so far are assumed to be available prior to computing the alpha map. To harness the power of matting methods in rendering visually plausible composites in NVS and IBR applications, trimaps generation have to be automated. In Chapter 6, I propose an automatic trimap generation technique.

Connectivity error	overall rank	avg. small rank	avg. large rank	avg. user rank	Troll (Strongly Transparent) Input small	Troll (Strongly Transparent) Input large	Troll (Strongly Transparent) Input user	Doll (Strongly Transparent) Input small	Doll (Strongly Transparent) Input large	Doll (Strongly Transparent) Input user	Donkey (Medium Transparent) Input small	Donkey (Medium Transparent) Input large	Donkey (Medium Transparent) Input user	Elephant (Medium Transparent) Input small	Elephant (Medium Transparent) Input large	Elephant (Medium Transparent) Input user	Plant (Little Transparent) Input small	Plant (Little Transparent) Input large	Plant (Little Transparent) Input user	Pineapple (Little Transparent) Input small	Pineapple (Little Transparent) Input large	Pineapple (Little Transparent) Input user	Plastic bag (Highly Transparent) Input small	Plastic bag (Highly Transparent) Input large	Plastic bag (Highly Transparent) Input user	Net (Highly Transparent) Input small	Net (Highly Transparent) Input large	Net (Highly Transparent) Input user
Random Walk Matting	**5.2**	3.9	7.8	4	0.8 2	0.8 5	0.8 3	**0** 1	**0** 1	**0** 1	0.2 21	0.2 18	0.2 23	**0** 1	**0** 1	**0** 1	0.1 3	0.4 34	0.1 1	**0** 1	**0** 1	**0** 1	**0.2** 1	**0.2** 1	**0.6** 1	**1.2** 1	**1.2** 1	**1.2** 1
Transfusive Weights	**7.8**	4.8	6	13	0.8 3	0.8 6	0.8 4	0 2	0 2	0.1 2	0.2 22	0.2 19	0.2 24	0 3	0	0	0.1 2	0.1 9	0.7 25	0 2	0 2	0 2	0.2 2	0.2 2	0.7 2	1.2 2	1.2 2	3.4 32
CCM	**9.6**	10.4	9.8	8.6	**0.7** 1	0.8 4	0.7 2	0.4 26	0.6 22	0.6 30	0.2 3	0.2 6	0.2 4	0 10	0 3	0 8	0.1 13	0.1 7	0.4 11	0.1 5	0.3 8	0.1 7	15.4 21	12 22	0.7 3	1.4 4	1.5 6	1.4 4
LNSP Matting	**9.9**	12.3	9.6	7.9	0.8 5	**0.7** 1	0.8 6	0.2 6	0.3 5	0.2 5	0.2 5	0.2 3	0.2 2	0 14	0 13	0 17	0.1 8	0.1 3	0.1 3	0.2 10	0.5 18	0.1 8	25.9 33	12.3 24	1.1 10	2.1 17	1.8 10	1.9 12
Closed-Form Matting	**10.5**	10.5	9.4	11.5	0.8 4	0.8 9	**0.7** 1	0.2 5	0.3 7	0.3 10	0.2 10	0.2 5	0.2 15	0 18	0 12	0 19	0.1 14	0.2 15	0.4 14	0.2 22	0.4 16	0.2 17	0.5 3	1.7 4	1 9	1.6 8	1.6 7	1.5 7
Trajectory	**10.9**	13.3	10.6	8.9	0.8 6	0.7 2	0.8 7	0.2 7	0.3 6	0.2 6	0.2 6	0.2 4	0.2 3	0 15	0 14	0 18	0.1 9	0.1 4	0.1 4	0.2 11	0.5 19	0.1 9	25.9 34	12.3 25	1.1 11	2.1 18	1.8 11	1.9 13
GCHPC Matting	**12.3**	14.3	13.9	8.8	0.8 7	0.8 13	0.8 8	0.1 4	0.2 4	0.2 4	0.2 27	0.2 17	0.2 20	0 5	0 4	0 5	0.1 23	0.1 10	0.1 5	0.2 15	0.2 6	0.1 4	15.7 22	25.8 44	1.4 16	1.6 11	1.8 13	1.6 8
Large Kernel Matting	**14.5**	16.6	12.6	14.4	1 29	0.9 17	0.9 23	0.2 10	0.4 12	0.3 9	0.2 12	0.2 15	0.2 21	0 22	0 10	0 13	0.1 18	0.2 17	0.5 16	0.2 24	0.4 15	0.2 14	3.3 6	1.6 3	0.8 5	1.8 12	1.8 12	2 14
anonymous_submission (modified version)	**15.3**	15.3	14.4	16.3	1 27	1 25	1 32	0.2 11	0.3 9	0.3 12	0.2 23	0.2 20	0.2 22	0 11	0 7	0 4	**0.1** 1	**0.1** 1	0.1 2	0.1 3	0.1 3	0.1 6	20.4 27	12.4 26	3.1 30	2.1 19	2.4 24	2.6 22
SVR Matting	**15.5**	17.4	13.4	15.8	1.2 35	1 22	1.1 34	0.3 15	0.5 15	0.4 16	0.3 33	0.2 26	0.2 29	0 8	0 8	0 24	0.1 11	0.2 18	0.2 7	0.1 6	0.1 4	0.1 3	20.3 26	4.8 10	1 8	1.5 5	1.3 4	1.4 5
Improved color matting	**16**	18.1	13.6	16.1	0.8 9	0.8 10	0.8 15	0.3 18	0.6 16	0.5 21	0.2 16	0.2 11	0.2 13	0 19	0 6	0 14	0.1 21	0.2 14	0.6 22	0.1 4	0.1 5	0.1 5	29.5 37	15.6 33	1.8 19	2.3 21	1.9 14	2.4 20
LocalSamplingAndKnnClassification	**17.5**	20.5	16.6	15.3	0.8 11	0.8 8	0.8 12	0.2 8	0.3 8	0.3 7	0.2 15	0.2 12	0.2 14	0 31	0 18	0 12	0.1 27	0.2 23	0.7 26	0.2 23	0.4 14	0.3 21	32.9 42	22 42	2 24	1.6 7	1.7 8	1.5 6
Local Spline Regression (LSR)	**17.8**	22.6	17.4	13.4	0.9 18	0.7 3	0.8 5	0.2 9	0.3 10	0.3 14	0.2 26	0.2 21	0.2 17	0 25	0 19	0 21	0.1 22	0.2 22	0.4 12	0.2 25	0.3 10	0.2 16	39.2 46	32.8 45	1.2 12	1.6 10	1.7 9	1.6 10
Cell-based matting Laplacian	**18.6**	20.5	16	19.4	0.9 17	0.8 11	0.8 17	0.4 22	1.3 35	0.5 24	0.2 17	0.2 13	0.2 16	0 39	0.1 27	0.1 40	0.1 17	0.1 11	0.6 19	0.4 36	0.3 9	0.3 23	11.5 13	8.5 19	1.3 14	1.2 3	1.3 3	1.2 2
Anonymous SP_Lett_Subm	**18.8**	18.3	21.4	16.8	0.9 20	1.1 29	0.9 25	0.3 16	0.4 13	0.3 13	0.3 36	0.3 29	0.2 31	0 4	0.1 23	0 6	0.1 25	0.2 20	0.4 13	0.2 13	0.8 29	0.2 11	13.7 17	3.7 7	1.8 20	2 15	2.2 21	2.1 15

Figure 5.4: Ranking of the matting algorithms according to the connectivity metric on the alpha matting benchmark [3], on the 18^{th} of *April 2016*. The presented technique appears under the name 'GCHPC' matting. It tops the method presented in Chapter 4, which appears under the name 'Anonymous SP˙Lett˙Subm', and other SoA sampling-based techniques, with respect to this metric.

Sum of Absolute Differences	overall rank	avg. small rank	avg. large rank	avg. user rank	Troll (Strongly Transparent) Input small	Troll large	Troll user	Doll (Strongly Transparent) Input small	Doll large	Doll user	Donkey (Medium Transparent) Input small	Donkey large	Donkey user	Elephant (Medium Transparent) Input small	Elephant large	Elephant user	Plant (Little Transparent) Input small	Plant large	Plant user	Pineapple (Little Transparent) Input small	Pineapple large	Pineapple user	Plastic bag (Highly Transparent) Input small	Plastic bag large	Plastic bag user	Net (Highly Transparent) Input small	Net large	Net user
anonymous_submission (modified version)	**3**	4.8	1.3	3.1	12 8	14.1 2	14.5 4	5.3 1	6.4 1	6.8 3	3.9 2	4.5 2	3.4 3	1.6 10	2.5 1	2.2 3	6 9	6.9 1	9.1 1	4 3	6 1	5.3 2	19.9 3	19.2 1	19.1 7	19.4 2	20 1	21.2 2
Cluster-based Sampling Matting	**7.4**	6.5	9.4	6.4	**9.6** 1	14.5 3	13.3 2	5.7 8	7.8 6	7.2 4	4.6 11	5.8 18	3.5 4	1.5 5	4.6 17	2.7 9	6.2 10	8.3 9	11.9 12	4.3 7	9.4 14	6.2 6	22 9	21.9 6	20.6 13	**19** 1	21.5 2	**21** 1
CSC Matting	**10.5**	14.4	6.4	10.6	13.6 22	15.6 4	14.5 3	6.2 12	7.5 4	8.1 16	4.6 15	4.8 4	4.2 19	1.8 15	2.7 2	2.5 7	5.5 3	7.3 2	9.7 2	4.6 9	7.6 5	6.9 13	23.7 13	23 12	21 14	26.3 26	27.2 18	25.2 11
LNSP Matting	**10.8**	7.3	10.5	14.8	12.2 9	22.5 30	19.5 35	5.6 3	8.1 7	8.8 23	4.6 12	5.9 20	3.6 6	1.5 6	3.5 10	3.1 19	6.2 11	8.1 6	10.7 7	4 1	7.1 2	6.4 8	21.5 7	20.8 3	**16.3** 1	22.5 9	24.4 6	27.8 19
Anonymous_TIP_submission	**11.4**	11.8	12	10.5	12.6 15	20.5 22	14.8 8	5.7 7	7.3 2	6.4 2	4.5 9	5.3 10	3.7 8	1.4 2	3.3 7	2.3 5	6.3 14	7.9 3	11.1 9	4.2 4	8.3 9	6.4 7	28.7 31	31.3 32	27.1 27	23.6 12	25.1 11	27.3 18
Trajectory	**11.8**	8.3	11.5	15.8	12.2 10	22.5 31	19.5 36	5.6 4	8.1 8	8.8 24	4.6 13	5.9 21	3.6 7	1.5 7	3.5 11	3.1 20	6.2 12	8.1 7	10.7 8	4 2	7.1 3	6.4 9	21.5 8	20.8 4	16.3 2	22.5 10	24.4 7	27.8 20
KL-Divergence Based Sparse Sampling	**12.1**	11.6	11	13.8	11.6 6	17.5 10	14.7 5	5.6 5	8.5 12	8 12	4.9 28	5.3 8	3.7 9	1.5 8	3.5 9	**2.1** 1	5.8 5	8.3 8	14.1 24	5.6 20	9.3 13	8 20	24.6 17	27.7 25	28.9 31	20.7 4	22.7 3	23.9 8
Comprehensive sampling	**13.9**	11.6	14.1	16	11.2 5	18.5 13	14.8 6	6.5 18	9.5 22	8.9 26	4.5 8	4.9 5	4.1 16	1.7 12	3.1 5	2.3 6	5.4 2	9.8 16	13.4 19	5.5 18	11.5 24	7.4 19	23.9 15	22 8	22.8 15	23.8 15	28 20	28.1 21
Iterative Transductive Matting	**14.2**	15.8	13.5	13.3	13.1 19	17.2 8	15.6 15	5.7 6	8.6 14	7.8 10	5.1 31	5.5 12	3.9 10	1.9 16	5.8 28	2.6 8	6.6 16	8.5 10	13.8 23	5.4 15	10 17	7.4 17	25.5 20	24 15	23.8 19	20.1 3	22.7 4	22.7 4
Comprehensive Weighted Color and Texture	**14.8**	15.4	15.3	13.6	14.6 27	16 5	15.7 17	6.8 24	10 27	7.9 11	4.3 6	5 6	4.1 14	1.7 11	3.5 12	2.2 2	**5.4** 1	9.9 18	12.8 17	4.3 6	7.4 4	**5.2** 1	28.3 30	28.1 26	25.4 23	24 18	30.2 24	28.7 24
SVR Matting	**14.8**	17.6	14.4	12.3	18.7 40	30.7 42	19.1 32	6.8 25	7.7 5	7.6 8	4.7 21	5 7	3.4 2	1.9 17	4.7 19	2.9 15	5.8 4	8.7 11	10.5 4	4.3 5	8 7	5.6 4	21.2 5	22.1 9	17.1 4	25.6 24	26.1 15	30.6 29
Sparse coded matting	**15.2**	18.5	15.8	11.4	13.7 23	25.8 38	14.8 9	6.4 15	8.2 10	**6.2** 1	4.7 16	5.4 11	4 12	1.8 14	3.1 4	2.3 4	5.9 7	8 5	10.6 5	4.5 8	8 6	5.5 3	30.3 33	33.1 35	29.2 32	27.7 32	27.2 17	29 25
LocalSamplingAndKnnClassification	**16.7**	19.1	14.1	16.8	12.6 14	16 6	**12.4** 1	5.8 9	8.1 9	8 13	4.5 10	5.5 13	4.1 13	2.2 24	5.1 22	3.4 28	8.1 29	10.5 20	15.6 30	7.3 30	12.3 29	9.4 27	24.1 16	21.8 5	19.7 9	24.7 21	24.8 9	25.9 13
Weighted Color and Texture Matting	**17.1**	14.9	19	17.5	13.1 20	17.8 11	15.8 18	6.5 17	9.4 20	8.6 20	4.2 5	4.7 3	3.9 11	1.7 13	6 29	2.7 10	6.4 15	11 25	16.3 31	4.8 10	9.1 12	6.5 10	23.7 12	24.8 17	23.2 17	26.5 27	40.2 35	28.5 23
CCM	**17.2**	20	17	14.6	13.8 25	20.8 23	16.9 26	6.4 16	8.9 16	8.2 17	4.7 17	5.9 22	3.6 5	2.5 28	4.3 15	3 16	7 19	9 12	10.6 6	4.9 11	8.1 8	5.7 5	25.6 21	27.5 24	24.5 20	25.4 23	26.4 16	28.2 22
Shared Matting	**17.9**	16.6	21.1	15.9	10.8 2	20.5 20	15 11	7.8 33	11.6 33	8.1 14	4.2 4	5.3 9	4.2 17	2.1 19	5.8 27	2.9 14	5.9 8	9.2 13	11.4 11	5 12	8.8 10	6.8 12	34.9 38	34.9 36	34.3 36	23.9 17	28.4 21	25.7 12
LNCLM matting	**18**	20.4	17.6	16	10.9 3	**11.2** 1	16.7 24	6.9 27	8.9 17	7.2 5	5.6 37	7 37	4.1 15	2.5 30	5.1 23	3.5 30	7.5 23	10 19	12 13	5.5 19	11.3 21	7.3 16	25.2 19	23 13	19.9 11	21.2 5	25 10	26.4 14
Anonymous SP_Lett_Subm	**18.3**	17.4	17	20.6	12 7	17.2 7	14.9 10	7.1 28	9.7 26	9.8 32	4.9 27	5.8 19	4.6 30	1.5 3	3.2 6	3.4 27	7.9 28	10.6 21	13.7 22	6.6 28	11.5 25	8.7 26	19.1 2	22 7	19.2 8	23.8 16	30.8 25	25.1 10
GCHPC Matting	**18.7**	19.9	13.1	23.1	13.8 24	17.4 9	15.7 16	6.7 22	9.3 19	10.5 35	4.8 23	5.7 16	4.2 18	**1.4** 1	2.7 3	3.6 31	9.9 39	10.8 23	15.3 29	8.1 33	10 16	11.4 36	21.3 6	23.7 14	19 5	23.2 11	24.1 5	26.5 15

Figure 5.5: Ranking of the matting algorithms according to the SAD metric on the alpha matting benchmark [3], on the *18th* of *April 2016*. The presented technique (GCHPC matting) achieved the 1st position in the first trimap of the image 'Elephant'.

Mean Squared Error	overall rank	avg. small rank	avg. large rank	avg. user rank	Troll (Strongly Transparent) Input small	Troll (Strongly Transparent) Input large	Troll (Strongly Transparent) Input user	Doll (Strongly Transparent) Input small	Doll (Strongly Transparent) Input large	Doll (Strongly Transparent) Input user	Donkey (Medium Transparent) Input small	Donkey (Medium Transparent) Input large	Donkey (Medium Transparent) Input user	Elephant (Medium Transparent) Input small	Elephant (Medium Transparent) Input large	Elephant (Medium Transparent) Input user	Plant (Little Transparent) Input small	Plant (Little Transparent) Input large	Plant (Little Transparent) Input user	Pineapple (Little Transparent) Input small	Pineapple (Little Transparent) Input large	Pineapple (Little Transparent) Input user	Plastic bag (Highly Transparent) Input small	Plastic bag (Highly Transparent) Input large	Plastic bag (Highly Transparent) Input user	Net (Highly Transparent) Input small	Net (Highly Transparent) Input large	Net (Highly Transparent) Input user
anonymous_submission (modified version)	**3.8**	5	1.6	4.6	0.4 7	0.5 2	0.7 6	**0.2** 1	**0.3** 1	0.4 4	0.2 3	0.3 2	0.2 6	0.1 8	0.1 1	**0.1** 1	0.4 11	0.4 1	0.8 6	0.2 4	0.4 3	0.3 3	1.3 4	1.2 2	1 6	0.7 2	0.7 1	0.9 5
LNSP Matting	**9.1**	6.3	8.6	12.4	0.5 11	1.9 32	1.2 33	0.2 2	0.4 3	0.5 15	0.3 8	0.4 14	0.2 3	**0** 1	0.1 4	0.2 10	0.4 12	0.5 3	0.8 4	**0.2** 1	**0.3** 1	0.4 7	1.4 7	1.2 3	**0.8** 1	1 8	1.1 9	1.5 26
Cluster-based Sampling Matting	**9.8**	8.8	10.5	10	**0.3** 1	0.6 3	0.5 2	0.3 10	0.4 8	0.5 10	0.3 15	0.4 22	0.2 10	0.1 9	0.2 16	0.2 20	0.5 15	0.5 11	1 14	0.3 9	0.7 15	0.5 11	1.5 10	1.4 7	1.2 12	**0.7** 1	0.8 2	**0.8** 1
Trajectory	**10.1**	7.3	9.6	13.4	0.5 12	1.9 33	1.2 34	0.2 3	0.4 4	0.5 16	0.3 9	0.4 15	0.2 4	0 2	0.1 5	0.2 11	0.4 13	0.5 4	0.8 5	0.2 2	0.3 2	0.4 8	1.4 8	1.2 4	0.8 2	1 9	1.1 10	1.5 27
KL-Divergence Based Sparse Sampling	**12.1**	11.8	10.6	14	0.4 5	0.9 9	0.7 4	0.3 8	0.5 13	0.5 14	0.3 26	0.4 8	0.3 16	0.1 7	0.2 12	0.1 2	0.4 8	0.4 2	1.2 21	0.4 17	0.6 13	0.6 19	1.7 19	2 25	2.1 32	0.8 4	0.8 3	0.9 4
CCM	**12.6**	15.6	12.6	9.5	0.5 16	1.2 17	0.8 19	0.3 13	0.5 17	0.5 11	0.3 5	0.4 10	**0.2** 1	0.1 20	0.2 6	0.1 4	0.5 22	0.6 13	0.7 3	0.3 8	0.4 5	0.3 2	1.7 18	1.8 20	1.5 21	1.2 23	1.2 13	1.3 15
Anonymous_TIP_submission	**13.1**	13.8	13.3	12.4	0.5 18	1.5 23	0.8 9	0.3 7	0.3 2	0.3 2	0.3 19	0.4 18	0.3 14	0 4	0.2 7	0.1 7	0.5 16	0.5 6	0.9 9	0.2 5	0.6 11	0.5 10	2.2 30	2.4 32	1.9 30	1 11	1.1 7	1.3 18
Comprehensive sampling	**13.6**	12.5	13.5	14.8	0.4 6	1.2 19	0.8 10	0.3 17	0.6 21	0.6 27	0.3 21	0.3 4	0.3 20	0.1 12	0.2 8	0.1 5	**0.2** 1	0.5 10	0.9 11	0.3 14	0.9 22	0.5 12	1.6 14	1.5 8	1.5 17	1 15	1.2 16	1.3 16
SVR Matting	**13.9**	18	12.3	11.4	1.1 37	2.7 41	1.2 32	0.4 32	0.4 6	0.4 7	0.3 22	0.3 5	0.2 2	0.1 11	0.2 11	0.2 16	0.3 4	0.5 9	0.7 2	0.2 6	0.4 6	0.3 4	1.4 6	1.4 5	0.9 4	1.3 26	1.2 15	1.5 24
Comprehensive Weighted Color and Texture	**15**	14.9	16	14.3	0.7 28	0.8 7	0.8 20	0.4 25	0.7 29	0.5 13	0.3 6	0.3 3	0.2 11	0.1 10	0.2 14	0.1 3	0.3 2	0.7 18	1 17	0.2 3	0.4 4	**0.3** 1	2.2 31	2.1 28	1.7 26	1 14	1.7 25	1.4 23
LocalSamplingAndKnnClassification	**15.7**	17.6	13.9	15.6	0.5 13	0.7 4	**0.5** 1	0.3 4	0.4 7	0.5 18	0.3 7	0.3 6	0.2 9	0.1 15	0.2 17	0.2 15	0.9 33	1 29	1.7 31	0.6 30	1 26	0.9 29	1.8 20	1.5 10	1.2 10	1.1 19	1.2 12	1.2 12
CSC Matting	**16.1**	20	9.8	18.6	0.6 23	0.8 5	0.7 7	0.3 16	0.4 5	0.7 28	0.4 32	0.4 12	0.4 36	0.1 28	0.2 13	0.2 28	0.4 6	0.5 5	0.8 7	0.3 13	0.5 7	0.6 18	1.7 17	1.5 11	1.3 14	1.2 25	1.3 20	1.2 11
Anonymous SP_Lett_Subm	**17**	17.4	14.1	19.5	0.4 3	0.9 8	0.7 5	0.4 29	0.6 22	0.7 30	0.3 28	0.4 11	0.3 27	0 6	0.1 3	0.3 32	0.7 30	0.7 20	1.1 20	0.6 29	0.8 19	0.8 26	1.2 2	1.5 9	1.1 9	1 12	1.3 21	1 7
Weighted Color and Texture Matting	**17.5**	16.3	18.9	17.5	0.5 17	0.9 11	0.8 8	0.4 21	0.6 23	0.6 24	0.3 4	**0.3** 1	0.2 7	0.1 23	0.5 34	0.2 24	0.5 17	0.8 22	1.5 28	0.3 11	0.5 9	0.5 9	1.6 13	1.6 15	1.5 19	1.2 24	2.6 36	1.4 21
Sparse coded matting	**17.7**	20	18.9	14.3	0.6 26	2.7 40	0.8 18	0.3 15	0.4 11	**0.3** 1	0.3 29	0.4 24	0.3 23	0.1 19	0.2 10	0.1 6	0.4 5	0.5 7	0.9 8	0.2 7	0.5 8	0.3 5	2.3 32	2.6 34	2.1 31	1.3 27	1.3 17	1.4 22
Global Sampling Matting	**18.3**	13.6	22.4	18.8	0.4 4	2.3 36	0.9 22	0.3 11	0.6 20	0.6 25	0.3 12	0.4 28	0.3 12	0.1 16	0.3 25	0.2 19	0.5 18	0.6 12	1 15	0.4 19	1 25	0.7 23	1.9 22	2 27	1.7 25	0.9 7	1 6	1.1 9
LNCLM matting	**18.9**	21.3	18.4	17.1	0.4 2	**0.4** 1	0.9 23	0.4 24	0.5 16	0.4 3	0.4 39	0.6 38	0.3 25	0.1 34	0.3 23	0.2 27	0.6 28	0.8 21	1.1 18	0.4 16	0.9 24	0.5 13	1.8 21	1.6 16	1.1 8	0.8 6	1.1 8	1.3 20
Iterative Transductive Matting	**19.1**	20.3	18	19	0.6 21	0.9 12	0.8 17	0.3 9	0.7 26	0.6 23	0.4 34	0.4 17	0.3 19	0.1 24	0.5 36	0.2 22	0.6 24	0.6 16	1.4 27	0.4 24	0.8 20	0.7 22	1.9 23	1.6 13	1.5 20	0.8 3	0.9 4	0.9 2
Shared Matting	**19.6**	18.5	22.4	17.9	0.5 8	1.6 25	0.9 21	0.5 34	0.9 35	0.5 19	0.3 10	0.4 9	0.3 22	0.1 25	0.4 29	0.2 13	0.4 9	0.6 15	0.9 10	0.4 15	0.6 12	0.5 14	2.9 37	2.6 36	2.7 36	1 10	1.3 18	1 8
KNN Matting	**20.1**	22.8	20.4	17.3	0.8 31	1 14	0.8 15	0.4 23	0.5 19	0.5 9	0.4 33	0.5 32	0.3 34	0.1 29	0.3 26	0.2 29	0.7 31	0.9 28	0.9 13	0.3 12	0.5 10	0.4 6	**1.1** 1	**1.1** 1	0.9 3	1.2 22	2.3 33	1.6 29
GCHPC Matting	**20.2**	23	13.4	24.3	0.6 20	0.8 6	0.8 12	0.4 28	0.6 24	0.8 38	0.3 27	0.4 13	0.3 24	0 5	0.1 2	0.3 34	1.3 40	0.9 27	1.3 25	0.9 39	0.7 16	1.3 36	1.5 9	1.6 14	1.2 11	1 16	0.9 5	1.2 14

Figure 5.6: Ranking of the matting algorithms according to the MSE metric on the alpha matting benchmark [3], on the 18^{th} of *April 2016*.

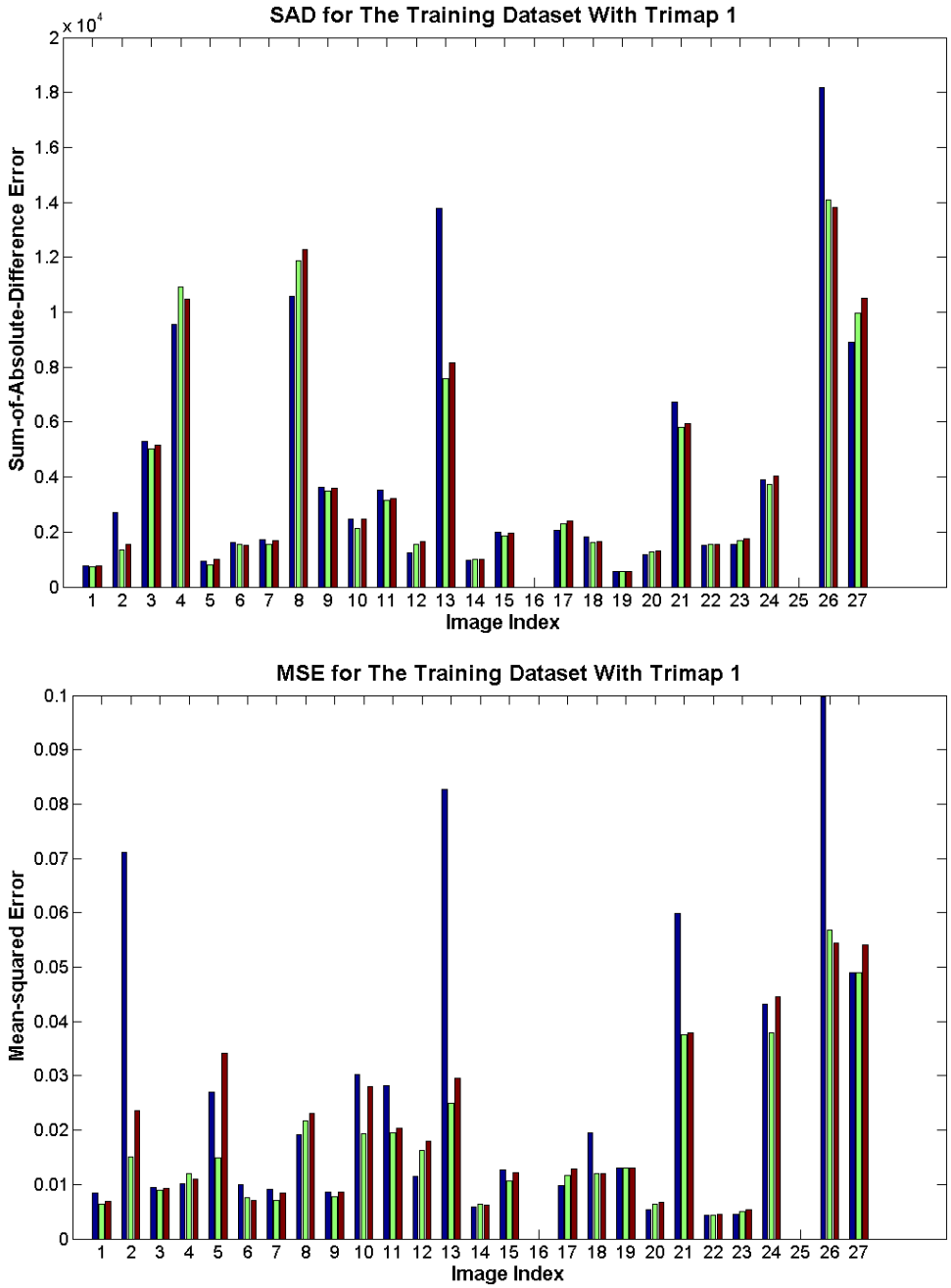


Figure 5.7: Objective comparison of the performance of GHC matting with the performance of the approaches discussed in Chapter 4, on the training dataset of [3]. The blue, green and red bars represent GHC, my method with the extended cost function and my method with the chromatic distortion cost function respectively. GHC matting achieved the least MSE and SAD in 10 images out of 25 images. Results for images number 16 and 25 were omitted for the clarity of presentation.

Chapter 6

A New Formulation for Automatic Trimap Generation Using Laws of Perceptual Grouping

In this chapter, I address the second kind of problems that arise while applying image matting techniques amid a novel view synthesis pipeline. In Chapter 4 and Chapter 5, I addressed the first class of problems, which is caused by the inefficiency of one or more stages in the pipeline of sampling-based alpha map computation. While the latter does exist with matting, regardless the context, the former kind of challenges is specific to the combination of matting and novel view synthesis problems, i.e., it may not be significant if matting is done in the context of single image editing operations.

Let us recall that the matting problem is usually relaxed using some kind of user interactions (scribbles or trimaps). To benefit from alpha maps in rendering visually appealing novel views, those user interactions have to be acquired automatically, for every rendered view, in a fast and a computationally-efficient manner. Throughout the rest of this Chapter, I will refer to this problem as 'automatic trimap generation'. It is worth mentioning that the only matting technique in the literature that featured realtime operation [7] had assumed the availability of dense trimaps for every frame, which is not a realistic assumption. In video matting, off-line operation is often acceptable. However, two of its three main challenges lie in: Allowing the the user to specify the foreground object spatio-temporally, and how to propagate such interaction through the whole sequence

in a temporally coherent way [9, 92].

On first thought, the problem of automatic trimap generation, for video matting and view synthesis applications, may appear as a trivial problem. Due to the enormous body of research on object detection and tracking, the problem of trimap generation is often, erroneously, over-simplified to an instance of object detection (on the first frame) and tracking (on the subsequent frames). In fact, it requires more contemplation for the following reasons:

- The object proposals, *that are usually acquired as bounding boxes*, do not solve either the matting problem, or the trimap generation problem. The bounding box does not align with the boundaries of the detected object, and thus we still need to identify, within the bounding box, which pixels are foreground and which pixels are really mixed. This requirement becomes more critical if we want to minimize the pixels that are labelled as mixed, to cut back the complexity of the matte computation.

- A lot of research is still ongoing to enhance the tracking of objects, even those objects with crisp boundaries. It is thus a long shot to rely on the stable tracking of fuzzy structures that appear often in the matting applications. Hence, performance of object detection and tracking for trimap generation is questionable, especially for the cases where the fuzzy regions occupy most of the scene, which is encountered in some examples in all the matting datasets available to date [3, 93].

- Sometimes, the idea of manually providing a few inputs on key frames is used; while this is still tedious, yet bearable for video matting, it is inapplicable in IBR systems.

I specify the characteristics of a perfect trimap generator as: automatic, realtime, aligns with object boundaries and, most importantly, detects mixed pixels only. My attempt to develop an automatic trimap generator satisfies the first and the third requirements only. I conclude that Chapter by discussing a framework for extending the proposed approach to address the rest of the characteristics of a perfect trimap generator. In the following section, I present a brief literature review before explaining my proposed approach.

6.1 Literature Review

This research overlaps with several areas in the literature, among which is automatic object segmentation and the use of Gestalt laws for scene analysis. For unsupervised object segmentation, Gestalt laws had been adopted before [94], and a few saliency-based techniques [95, 96, 97] have been proposed to deal with that challenge as well. I argue that automatic trimap generation is a more general problem than unsupervised object segmentation. The standard matting datasets [3, 93] show that the amount of fuzziness at object boundaries and the types of transparencies that can be encountered in the matting context imply more challenges for object(s) cut-out compared to the crisp boundaries that are prevalent in the datasets used to evaluate object segmentation methods. To the best of my knowledge, none of the saliency-based object segmentation techniques in the literature reported their performance on any of the standard matting datasets [3, 93].

The authors of [98] adopted laws of visual attention (similarity and proximity) to generate alpha maps automatically. While my proposed method fuses more cues (concavity, symmetry and proximity) to automate trimap generation, their algorithm provides the method in [70] with shift traces of focus-of-attention as input scribbles. From the video matting literature, I highlight three recently-proposed and state-of-the-art techniques [99, 100, 14]. In [99], the authors proposed an interactive system for object cut-out from video sequences. The performance is dependent on the amount of user assistance and, in some cases, frame-by-frame control points should be supplied by the user. The method in [14] adopts the non-local principle. In addition to being more efficient than [100] (which is also non-local-principle-based), it often suffices to provide one trimap on the first frame of a video sequence. However, the performance deteriorates gracefully with motion blur.

Automatic trimap generation had also been investigated by other researchers. In reverse chronological order, the algorithm in [101] uses a feature map calculated from the RGB image, morphological dilation and the region growing algorithm to generate a trimap. A learning-based approach was suggested in [102] in which GMMs for the FG and the BG (estimated static) are initially learnt. Afterwards, a per-pixel classification is computed to generate a trimap for the scene at hand. In [103], a range sensor was used to acquire depth information from which the trimaps were generated.

My automatic trimap generator is based on spectral matting [70]. As mentioned earlier, none of the saliency-based automatic object cut-out techniques were tested on matting datasets. Hence, I consider [70] the only matting technique in the literature that supports unsupervised operation (no user interactions required). Although it offers automatic matte computation, it suffers from critical limitations, among which is the unreliable results in the absence of user interactions. To state this more clearly: spectral matting is a framework which makes the automatic alpha map computation possible, procedure-wise; however, it often produces erroneous results in the absence of user interactions. In that sense, my proposed technique can also be seen as an enhancement over spectral matting. Before proceeding to the discussion of the proposed trimap generator, and due to its pivotal role in my research, I devote the next section to give a review of spectral matting.

6.2 Review of Spectral Matting

In [70], Levin *et al.* proposed a matting technique that is inspired by spectral clustering. In spectral image segmentation, the smallest eigenvectors of the image's graph Laplacian (where the pixels of the image represent the sites of the graph) are used to automatically extract the hard segments in an image. Similarly, the smallest eigenvectors of the matting Laplacian defined in [75] were shown to span, usually quite well, the matting components of an image. Those fuzzy components can thus be obtained by linear transformations of the smallest eigenvectors, and then act as building blocks for a meaningful complete foreground matte. An example of the matting components of an image can be found in Fig. 6.1. Those components were calculated using the publicly available spectral matting code [104]. As can be seen in the figure, each of these components is an image of the same size as the original image, and the pixel values in these components range from 0 to 1. The components surrounded by red boxes are the components that when added (their pixel values are added), will result in the complete alpha map which is shown in the bottom right corner of Fig. 6.1. Cast as a labelling problem, we need to label the components surrounded by red boxes with the label 1, and the rest of the components with the label 0, so that we can get the shown complete alpha map. This alpha map is

still not the ground truth, but very close to it. That closeness will depend on the quality of the calculated components. Taking the head of Donald Duck as an example, it can be seen in the figure that the thin structure representing the end of its hat is missed in the corresponding component.

Figure 6.1: Matting components of the 6^{th} training image in [3]. The first two images in the first row are the original and the ground truth respectively. The image in the bottom right corner is the best grouping which is the result of adding the components with red boxes. I calculated ten components for each image; only nine matting components are depicted in the figure though, for a clearer presentation of the figure.

The algorithm's pipeline is depicted in Fig. 6.2 and it proceeds as follows: given an input image with N pixels, it calculates the $N \times N$ sparse Laplacian matrix L and its K eigenvectors corresponding to the K smallest eigenvalues (please recall my discussion on graph Laplacian matrices in section 2.4). K matting components are then obtained from linear transformation of the K eigenvectors by minimizing a non-convex energy function that respects the sparsity in the resulting components and their linear-convexity at each

pixel [70]. This energy function is given by

$$\sum_{i,k} |\alpha_i^k|^\gamma + |1 - \alpha_i^k|^\gamma \text{ , where } \alpha^k = Ey^k \tag{6.1a}$$

$$\text{subject to } \sum_k \alpha_i^k = 1, \tag{6.1b}$$

where i is an index for the image pixels, k is an index for the matting components, $E = [e^1, \cdots, e^K]$ is an $N \times K$ matrix of Eigen vectors and y^k is a set of K linear combination vectors. Newton's method is used to minimize that energy function and the process is initialized by applying an unsupervised k-means clustering on the smallest eigenvectors.

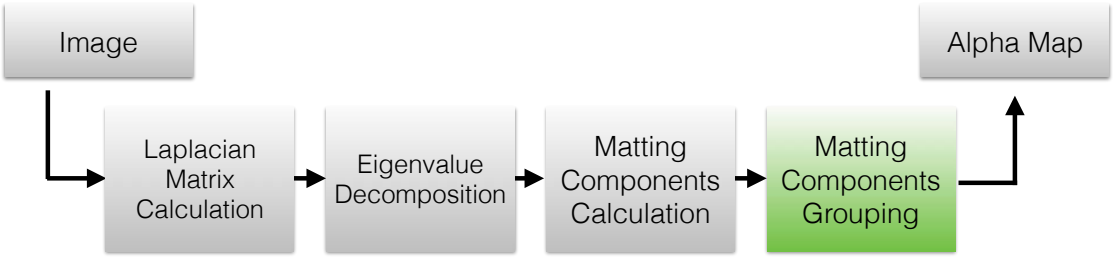


Figure 6.2: The block diagram of the un-supervised spectral matting algorithm. The stage where my proposed trimap generator comes into play is highlighted in green.

Starting with K matting components, like those shown in Fig. 6.1, the last step in the algorithm aims at grouping those components, to extract a complete matte. As mentioned earlier, only some of these components should be high-flagged (assigned label 1), then added to get the alpha map, while the rest of the components should be given a 0 label. Since K matting components have been computed, the authors of [70] test the 2^K grouping hypotheses in which each of the K matting components is either high or low (active or inactive). Hence, the term 'grouping' can refer to two things; it is an alpha map that is acquired by adding $k \leq K$ matting components, and equivalently it is the binary vector that corresponds to that grouping. In [70], the best grouping is the one that minimizes the quadratic cost given by

$$J(\alpha) = \alpha^T \times L \times \alpha, \tag{6.2}$$

where L is the matting Laplacian and α is the vector of N alpha values for the pixels in the image (the alpha map in vector form). To minimize Eqn. 6.2, the correlations between

the K matting components are precomputed via L and saved in a $K \times K$ matrix Φ where $\Phi(k,l) = \alpha^{k^T} \times L \times \alpha^l$. Then, the score of a hypothesis is calculated as

$$J(\alpha) = b^T \times \Phi \times b, \tag{6.3}$$

where b is a K dimensional binary vector indicating whether a component is active or not. Such a cost is biased towards nominating groupings with small number of fuzzy pixels and the authors of [70] resorted to balanced cuts, putting a constraint on the size of the FG object, which impacts negatively the generality of their grouping approach. This is where my contribution lies, a new objective function for efficient grouping of matting components.

Contrary to [70], my proposed cost function does not make any presumptions about the FG object's size; it is designed to favour the grouping hypotheses that adhere to the Gestalt laws instead. My objective function probes the symmetry, the concavity and the connectedness of a grouping, and gives it a Gestalt score. Formally, I pick the grouping that satisfies the following equation:

$$\underset{\mathcal{G}}{\operatorname{argmax}}\ G_s(\mathcal{G})\ ;\ \mathcal{G} \coloneqq \sum_k C_p^k\,, \tag{6.4}$$

where $G_s(\cdot)$ refers to the Gestalt score of the argument; it reflects the argument's concavity, symmetry and connectedness, and $\mathcal{G}$ is one of the 2^K grouping hypotheses. A single component in a grouping takes the symbol C_p, thus a particular grouping can be expressed as a summation of k components (C_ps).

In the following sections, I will give more details on the data terms of the suggested objective function, with which I quantify the adherence to the Gestalt laws. Analyzing the contour of a grouping under consideration plays a pivotal role in the formulation of all the proposed data terms. Contours are usually obtained by calculating an edge map, before connecting the edges together to construct the contours. If I fed the grouping itself to an edge detector, it is often the case that I get a large number of disconnected edges (a noisy edge map), especially in highly fuzzy regions of the image. To overcome this problem, two strategies can be adopted:

1. Thresholding the grouping before detecting its edges: This is the technique used

to obtain the results that will be shown in this document. After thresholding, if the contour is disconnected, I checked the distance between the ends of the disconnection; if it is less than a threshold distance, I connected them with straight line segment (short segments should not bias the Gestalt score), otherwise the hypothesis is ruled out as a disconnected grouping.

2. Another possible strategy is to calculate the matting components of the cartoon component of the image under consideration, instead of calculating the matting components of the original image. Consequently, most of the fuzzy (textural) regions will be left in the texture component, and a smooth contour can be obtained for subsequent processing.

6.3 Estimating the symmetry of a grouping

According to Gestalt laws, we tend to perceive objects as being symmetric figural entities. Even though, explicit symmetry is an over-simplification of the scenes encountered in matting tasks, the spatial extent of a BG in an image/scene is usually more than that of the FG which creates an instance of a non-trivial symmetry. This is also closely related to the FG object being observed as a closed surface [105]. I designed a cost function that is expected to have low values (leading to a low cost) in case of a symmetric grouping. This function is given by Eqn. 6.5.

To single out an enclosed grouping, I adopt a notion that is related to symmetry which is the centre of mass (CoM). The CoM was incorporated twice, once as a prior and once as a constraint. The prior, which is realized by the first term in Eqn. 6.5, favours groupings which minimize the absolute horizontal distance between the CoM and the apsis of the alpha matte (the FG pixel with the largest vertical coordinate in the image plane). The CoM-based constraint is realized by ruling out all the groupings in which the CoM lies outside the FG. This constraint is backed by the observation that objects like a horseshoe, for example, are not common in matting scenes. However, it could be reformulated as a prior based on the context. For the standard matting datasets, on which the performance of the proposed method was tested, I did not find this reformulation of remarkable significance.

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

Figure 6.3: An erroneous grouping (a) and its canny edge map (b) featuring its W as a green rectangle, and the best map (c) with its edge map (d). The figure illustrates the significance of the second and the third terms in Eqn. 6.5.

Inspired by [105], the second and third terms of Eqn. 6.5 capitalize on the closed-boundary characteristic of an object to infer symmetry. In the case of a symmetric grouping, the bounding box 'W' of the grouping's edge map should enclose most of the figural object and most of it should be occupied by that object as well. The significance of that rule is illustrated in Fig. 6.3. The symmetry cost is thus calculated as

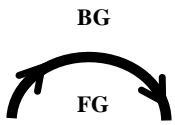
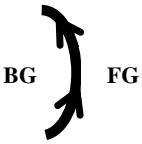
$$J_{sym}(\alpha) = \left(\frac{D_H}{I_W}\right) + \left(\frac{|F_G \setminus W|}{N_{F_G}}\right) + \left(1 - \min\left\{\mathcal{T}, \frac{|F_G \cap W|}{|W|}\right\}\right), \tag{6.5}$$

where D_H is the absolute horizontal distance between the CoM and the matte's apsis, I_W is the width of the image, $|F_G \cap W|$ is the number of foreground pixels in W, $|W|$ is the area of W, $|F_G \setminus W|$ is the number of foreground pixels outside W, N_{F_G} is the total number of foreground pixels in the grouping and $\mathcal{T}$ is a constant ($\mathcal{T} = 0.5$ in my experiments). The reason for using a threshold and how I set it is explained as follows: As mentioned earlier, I want to favour the groupings in which the bounding box is *mostly* occupied by the foreground object; once the ceiling of half the area of the bounding box is reached, the third term *should not* favour one grouping over the other. Otherwise, I would favour the high-flagging of extra components that fill more area of the bounding box without adding to the symmetry of the alpha map.

6.4 Estimating the concavity of a grouping

Motivated by the study in [40], I have incorporated the concavity cue to penalize (and eventually rule out) erroneous groupings where an area bounded by a concave arc (or contour) is assigned to a background. To cut back the algorithm complexity, in contrast

Table 6.1: Two entries from the dictionary used for concavity/convexity calculation

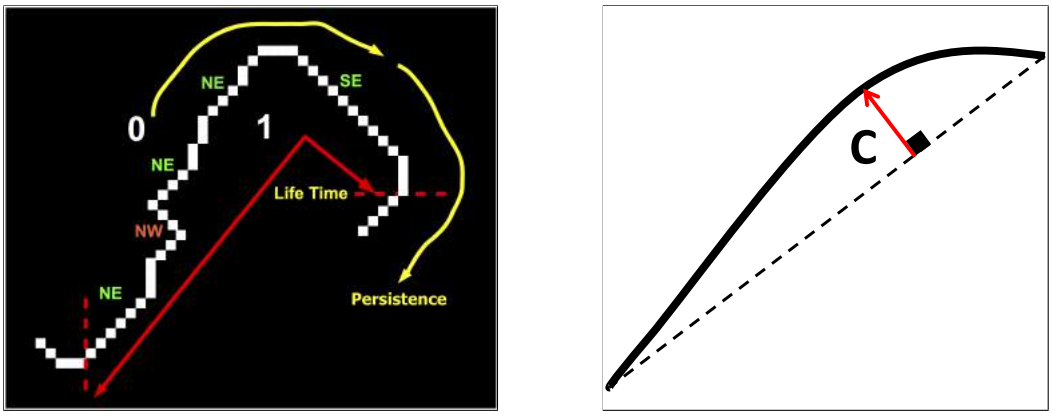
Direction Change		Indication
North-east To South-east		**Concavity**
North-east To North-west		**Convexity**

to [40], I didn't fit splines to contours. Instead, I analyzed the contours of the groupings themselves (like those contours shown in Fig. 6.3 (b) and (d)) and used a simple dictionary of rules to decide if a certain pair of direction changes indicates concavity or convexity. Two examples of the entries of that dictionary are shown in Table. 6.1

After obtaining the contour of the grouping under consideration, using any of the two methods indicated at the the end of section 6.2, I have divided the contour into sections, each of which corresponds to a pair of direction change, like those indicated in the previous table. A section in the contour is thus a set of pixels which represents a part of the contour, and features a pair of direction changes. The concavity score of a contour is formulated as

$$\mathcal{CAV} = \sum_{S_j^{cv}} C \times L^f \times (1 + \lambda \times P), \tag{6.6}$$

where C is the curvature of a contour section, L^f is the lifetime of a contour section, P is a constant which contributes to the concavity(or convexity) score if a curve persists on a certain status (either concavity or convexity), and lastly λ is a binary indicator which takes the value of 1 or 0 if persistence exists or does not exist respectively. Figure 6.4(a) depicts a typical example of a concave section in a contour, and the different variables in Eqn. 6.6 are shown on it. The overall concavity score of a particular contour, $\mathcal{CAV}$, is then the summation of the concavity scores of the set of concave curve sections $\{S_j^{cv}\}$ in the contour. Fig. 6.4(b) depicts a section of a contour (a set of pixels representing a part of the contour), and it illustrates how I quantified the curvature of a contour section; this



(a) (b)

Figure 6.4: (a) An illustration of a concave section of a contour with variables in Eqn. 6.6 shown on it, and (b) depicts a contour section and shows how its curvature is measured. Please see text for more details.

curvature is formulated as follows:

$$C = \sup_{a \in A} \inf_{b \in B} \mathrm{D}(a, b), \tag{6.7}$$

where 'sup' is the supremum, 'inf' is the infimum, A is the set of all the contour pixels of the curve whose curvature is to be quantified, B is the set of all points on the chord subtending to A, and $D(a, b)$ is the Euclidean distance from a to b. Particularly, the curvature is computed as the largest minimum distance from the points on the contour to the chord subtending to it. If the contour under consideration is a perfect circular arc, its maximum curvature is given by its sagitta. An equation similar to Eqn. 6.6 is used to calculate the convexity score, where the summation over S_j^{cv} is replaced with the summation over S_j^{cx}, the set of convex curve sections in the contour.

Having the overall concavity and convexity scores (of the contour sections) calculated, the concavity cost of a particular grouping is then computed as

$$J_{cav}(\alpha) = \frac{2 \times \mathcal{CVX}}{\mathcal{CVX} + \mathcal{CAV}} \ , \tag{6.8}$$

where $\mathcal{CVX}$ and $\mathcal{CAV}$ are the convexity and the concavity scores respectively.

6.5 Estimating the connectedness (proximity) of a grouping

I assumed that the foreground object is spatially connected, which is true for all trimap-based matting [10]. Dealing with multiple foreground objects has not been addressed in the current implementation, as will be discussed at the end of this Chapter. Consequently, the graph whose nodes are the matting components has to be connected, which is satisfied if every pair of components are connected. This can be verified from the components adjacency matrix using the Dulmage-Mendelsohn decomposition [49]. For the adjacency matrix A with entries $a_{i,j}$:

$$a_{i,j} = 1 \iff \exists\, p \in c_i^r : D_{min}(p, c_j^r) \leq \mathcal{D} \tag{6.9}$$

where c_i^r and c_j^r are the two sets of the contour pixels of components i and j respectively, $D_{min}(p, c_j^r)$ is the minimum Euclidean distance between pixel p and the set of contour pixels c_j^r, and $\mathcal{D}$ is a constant ($\mathcal{D} = 4$ in our experiments). We used the connectivity as a constraint and defined J_{conn} as

$$J_{conn}(\alpha) = \begin{cases} 1 & \text{if } T(A_{DM}) = 1 \\ \infty & \text{otherwise,} \end{cases} \tag{6.10}$$

where A_{DM} is the Dulmage-Mendelsohn decomposition of the adjacency matrix A, and $T(A_{DM})$ is an indicator function which returns '1' if the grouping 'α' is connected and returns '0' otherwise. The final cost function is then formulated as the product of the costs corresponding to the three cues discussed so far, and is given by

$$J(\alpha) = J_{cav}(\alpha) \times J_{sym}(\alpha) \times J_{conn}(\alpha), \tag{6.11}$$

where $J_{cav}(\alpha)$ is given by Eqn. 6.8, $J_{sym}(\alpha)$ is given by Eqn. 6.5 and $J_{conn}(\alpha)$ is given by Eqn. 6.10. Results of the proposed automatic trimap generator are shown in Fig. 6.5.

To show the significance of every term in the objective function, I tried to disable two of them at a time and keep the thrid, then check the nominated groupings. The results are shown in Fig. 6.6. Leaving any term out of the objective function would

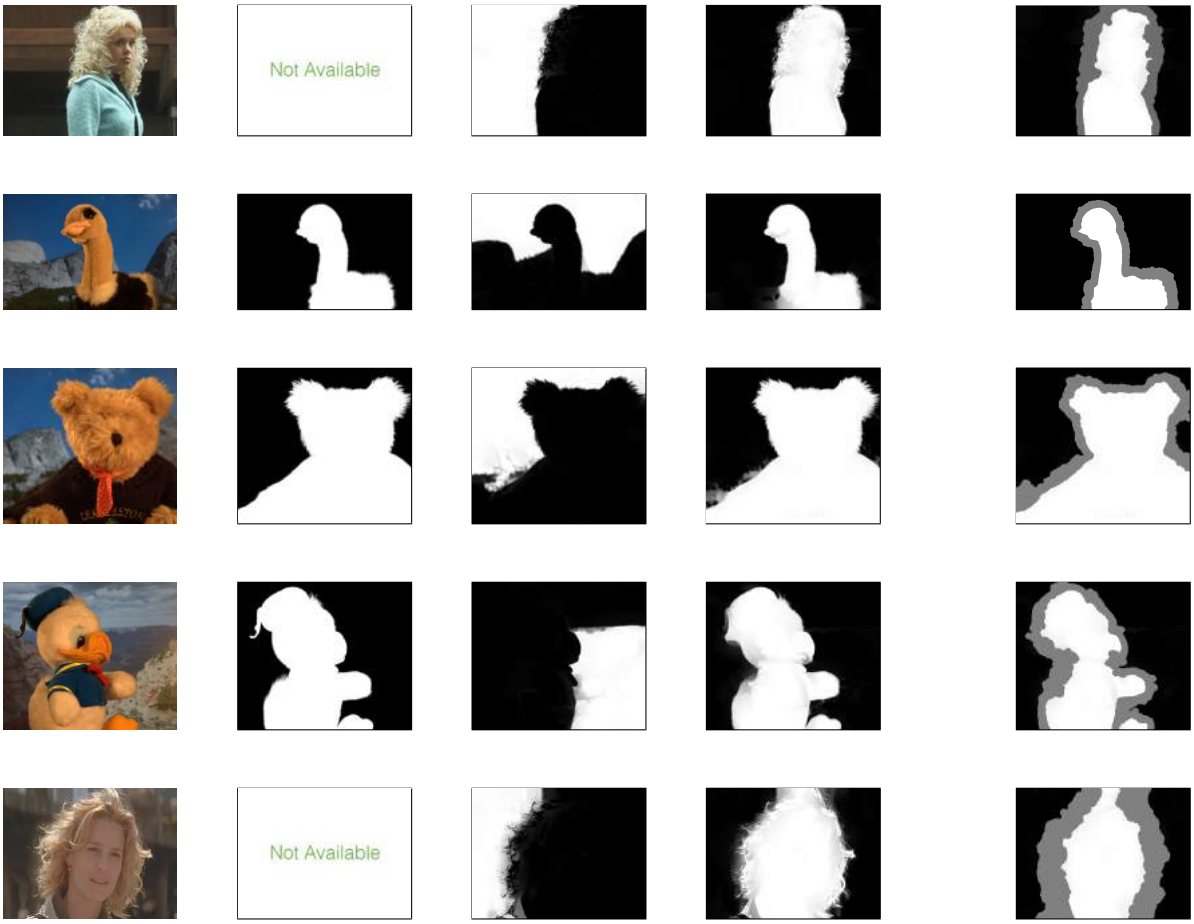


Figure 6.5: The results of the unsupervised spectral matting (column 3), the alpha mattes produced automatically by our algorithm (column 4) and the trimaps generated from them (column 5). Column 1 shows the original images and column 2 shows the ground truth. The results in column 3 were calculated using RGB affinities.

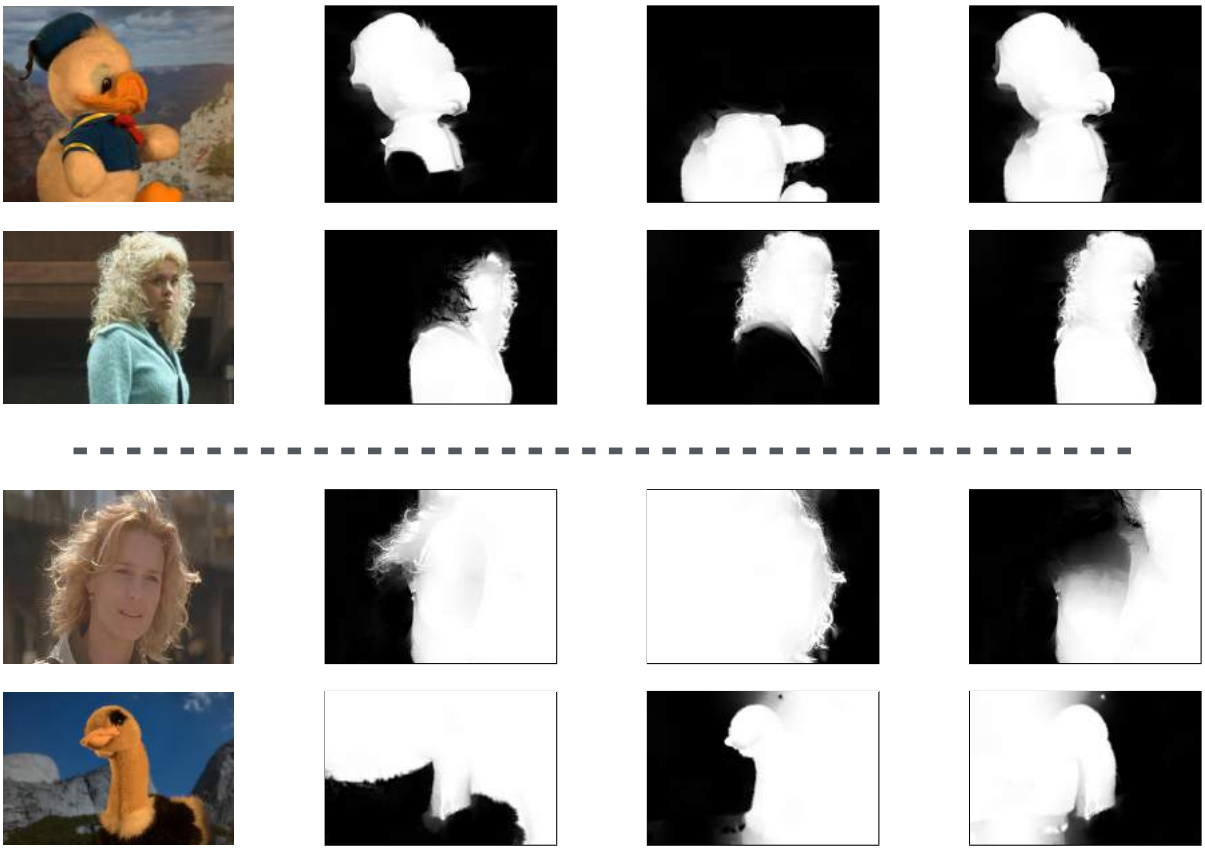


Figure 6.6: Examples for erroneous groupings calculated by adopting a single cue in the objective function, instead of using three cues for inferring the correct matte. The first column is the original image. The upper two rows show instances of highly-symmetric, yet erroneous, groupings. The lower two rows show overall concave groupings with bad symmetry score.

result in erroneous groupings, which emphasizes the effectiveness of the proposed objective function and aligns with the holistic role that the Gestalt laws play in the process of visual perception.

6.6 A probabilistic view

The problem of picking the best grouping can be seen as a set of optimum decisions, each of which determines whether a component C_p in that grouping will be set to high or low. A label l_c that is assigned to a matting component can thus be modelled as a binary random variable, and the labelling of a single component in a grouping $\mathcal{G}$ can be formulated in a Bayesian fashion as follows:

$$P(l_c/\mathcal{I}) \propto P(\mathcal{I}/l_c) \cdot P(l_c), \tag{6.12}$$

where $P(l_c/\mathcal{I})$ is the posterior probability that a certain matting component C_p will take the label l_c ($l_c = 1$ means the component is active), given the original image $\mathcal{I}$. The first term on the right hand side, $P(\mathcal{I}/l_c)$, is the observed data term which can be formulated as the likelihood that the component C_p contains a FG object or mixed pixels with fractional alpha values ($0 < \alpha \leq 1$). A possible formulation of this term will be pointed out in the future research directions in Chapter 8.

The second term in this formulation, $P(l_c)$, encodes the prior assumptions about the final matte. Particularly, given the labels of the rest of the components (all neighbours of C_p), this term will consider the label $l_c = 1$ highly probable if the overall grouping is more concave, symmetric and connected than if $l_c = 0$. For the formulation of an energy function, this prior would be represented by a higher order term that involves interactions between the labels of all the available components. The best grouping is thus the one which maximizes the equation given by:

$$\underset{j}{\operatorname{argmax}} \sum_{k=1}^{K} P^j(\mathcal{I}/l_c^k) \cdot P^j(l_c^k);\ j \coloneqq \{l_c^k\} \tag{6.13}$$

where $P^j(\mathcal{I}/l_c^k)$ is the likelihood of component C_p to have label l_c^k in grouping 'j'; here, k is an index for the matting components and j is an index for the possible 2^K grouping hypotheses. $P^j(l_c^k)$ is the probability of component C_p to have label l_c^k in grouping j given the Gestalt score of the grouping.

As mentioned earlier, finding the optimal grouping guarantees an alpha map that is very close to the ground truth, up to the accuracy of the calculated components. It is often the case that even the optimal grouping is not an accutrate alpha map. This means that the accuracy of the calculated components do not let them constitute an accurate alpha map with satisfactory quality, even though the optimal grouping is very close to the ground truth **compared to** the output of Spectral Matting without my proposed grouping algorithm. My technique makes it possible to single out the best grouping, dilate the boundaries, and feed it to a more robust matting technique as a trimap. Although this trimap has been acquired automatically, resorting to the final step of the morphological dilation is not desirable, since it may augment the number of pixels with unknown alpha values unnecessarily. Moreover, even though I did not adopt

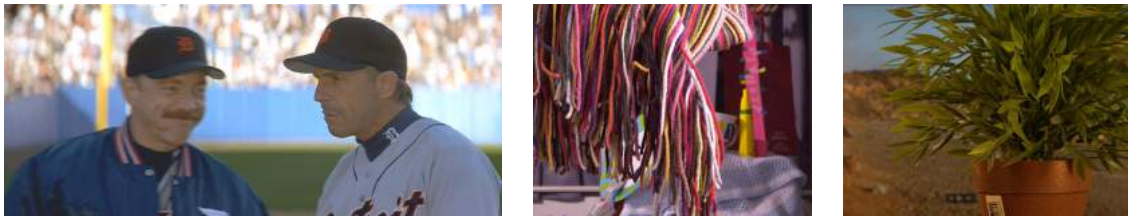


Figure 6.7: A few examples for cases that the proposed method is not ready to handle, cases of failure.

computationally demanding stages like spline fitting for instance, the proposed approach is still far from realtime operation. It is an automatic trimap generator, it aligns with the foreground object boundaries, rather than generating bounding boxes; however, a lot of room for more development is still available. Most importantly, augmenting the ability of picking a Gestalt-adhering grouping with the ability to detect mixed pixels is highly favourable; those two requirements were addressed by the presented probabilistic framework. Last but not least, the proposed algorithm is not ready to deal with multiple foreground objects, or cases where the foreground object is not concave, or cases where the foreground object contains concave parts at varying scales. Thus, I consider those cases as cases of failure, and I give a few examples of them in Fig. 6.7. The first image is from [106], while the last two images are from the standard image matting dataset [3]. Dealing with multiple foreground objects though (as in the first image) is a point of future research.

In this chapter, I proposed a new formulation for generating trimaps automatically using the Gestalt laws of visual perception. Using the matting components computed from the Eigen decomposition of the matting Laplacian, a new cost function that incorporates concavity, symmetry and proximity was proposed to generate the trimap by computing the best combination (grouping) of matting components. In the light of the characteristics of a perfect trimap generator, which were mentioned in the beginning of the chapter, the merits and the drawbacks of the proposed method were discussed. Finally, future research directions will be highlighted in Chapter 8.

Chapter 7

Image Completion Using Image Skimming and Near-globally-optimal Shift Maps

As a recall to what have been mentioned in Chapter 3, the dis-occlusion management (scene completion) is the most general setup of the hole filling problem. This is due to the fact that we have temporal information and we have depth maps, in addition to the multiple views. Accordingly, at the heart of every dis-occlusion management technique, a single-image hole filling approach plays a pivotal role. I found that there is still room for improvement even for single-image hole filling. Therefore, I start off by proposing a new method for single image completion that lends itself well to manage dis-occlusions in the context of novel view synthesis. A high-level sketch for the framework that adopts my proposed technique in the novel view synthesis scenario will then be given in section A.7.

I propose an algorithm for image completion that follows the same two-stage framework of the techniques discussed in Chapter 3; it starts by constructing a BoSP and then uses those patches to fill the hole. Akin to [26] and [28], my technique seeks a near-globally-optimal shift map, while simultaneously addressing their inherent limitations. The optimality here refers to the process of minimizing (or maximizing) an energy function for the completion, while 'global' describes the nature of the technique used for doing the optimization, which finds a local minimum within a known factor from the global minimum. Please check section A.8. Recall that for robust image completion, it is

favourable to have:

- A small number of shifts, from which an optimal shift is nominated for every hole pixel. The confinement of the number of shifts is crucial for the quality of completion as well as the computational complexity of the optimization stage of the algorithm.
- A set of shifts that are sufficient to fill the hole, i.e., calculated as a function of the hole size; as the hole gets larger, they cope accordingly. Otherwise, if they are collected based on the known part in the image, for instance, they may be smaller than required to bring a known pixel to inside the hole.

Any arbitrary shift in the image plane is comprised of a slope and a translation (shift) along that slope. Hence, a possible starting point for the hole filling procedure is to nominate dominant slopes, rather than shifts, then let the size of the hole decide the magnitude of the shifts required to bring pixels from the non-hole region to inside the hole region. The development of the presented method counted on another key observation that is: No matter how many pixels an image contains, a very small number of patches (square or rectangular windows) is required to describe a linear texture, for instance, in that image. Consequently, this research suggests another formulation for the image completion problem. My formulation uses some 'representative' square regions or patches in the image plane to calculate a few dominant slopes. The shifts required to fill the hole are then decided based on the spatial location (in the image plane) of the intersection between the hole region and the bag of patches when they are slid along the calculated dominant slopes.

The bag of representative patches will be called *the image skim*, and the term *skim* will be used as a verb and as a noun, throughout the rest of this thesis. The criteria for including a patch in an image skim, as well as the reason for the preference of using the term *skim* over the more-frequently-used term *summary* will be discussed in the following section. The pipeline of the suggested hole filling procedure is comprised of the three following steps: skim the image, nominate a few dominant slopes from that skim, slide the skim along the nominated slopes to *paint the hole*. The rest of this Chapter covers the details of the aforementioned three steps.

7.1 Constructing the Bag of Significant Patches Using Image Skimming

I seek a comprehensive image skim $\mathcal{S}$, which will represent my BoSP. Throughout the following discussion, a 'patch' is a square region in the image's spatial domain, $\mathbb{S}$ is the set of all patches in the image such that $\mathcal{S} \subset \mathbb{S}$, a 'patch's feature neighbour' is another patch that is very similar to its RGB pattern, i.e., the Euclidean distance between their RGB patterns is low, and a 'near feature neighbour' is a neighbour that is spatially close to the patch under consideration. Details on how close two patches should be to be considered spatially close will be given later on amid the discussion of the presented method. As for what constitutes the feature neighbours of a particular patch: An approximate nearest-neighbour (ANN) field is calculated for every patch based on the Euclidean distance between its RGB pattern and those of the rest of patches in the image. Thus, the K feature neighbours of a particular patch are the most similar K patches to it according to their RGB patterns.

Inspired by [107], I developed the skimming process so that it fulfills two requirements in $\mathcal{S}$, which are:

Similarity: A patch $\mathcal{P}$ is included in $\mathcal{S}$ if it has many similar patches in the image $\mathcal{I}$.

Orthogonality: A patch $\mathcal{P}$ is included in $\mathcal{S}$ if none of its feature neighbours is already included in $\mathcal{S}$.

The terms 'summarization' and 'skimming', either for images or video sequences [108], have been used interchangeably in the literature. However, I argue that their goals are different. Summarization from the perspective of [109] is a condensation process of the image's content. On the other hand, in skimming, we may neglect some of the image's content if it does not satisfy the aforementioned similarity requirement. The above two requirements can be formulated in a Bayesian fashion. To include a patch $\mathcal{P}$ in $\mathcal{S}$, I estimate the posterior as follows:

$$P(l_m/\mathcal{I}) \propto P(\mathcal{I}/l_m) \cdot P(l_m), \tag{7.1}$$

where $P(l_m/\mathcal{I})$ is the the posterior probability that a certain patch $\mathcal{P}_m$ will take the binary label l_m ($l_m = 1$ means the patch is included in $\mathcal{S}$), $P(\mathcal{I}/l_m)$ is the observed data

(likelihood) term and $P(l_m)$ represents our prior assumptions about the skimmed image. The prior captures orthogonality and enforces a smoothness constraint on the skimming binary field by taking into consideration the labels of the patches in the neighbourhood $\mathcal{N}$ of $\mathcal{P}_m$ while assigning the label l_m. Throughout the rest of this Chapter, when a patch is said to be 'high-flagged', it means that it has been assigned the label **1**, i.e., it is included in $\mathcal{S}$. Conventional wisdom says that we shouldn't assume the smoothness of the skim field, since the skimming of an image seems to be in contradiction with 'high-flagging' two neighbouring patches. I argue that, while this is true for smooth regions, it is not the case for regions with edges, for instance. Thus, I assumed the smoothness of the skim, and then let the inclusion of a patch in $\mathcal{S}$ abide to the orthogonality assumption, as will be clarified below. Inferring the MAP estimate can be done using several techniques. I have chosen two of them to present in this document, an iterative heuristic and a graph-based approach. A third method, which I decided not to detail here, has also been explored. It involved the formulation of a closed-form objective function for the skimming process, which can then be minimized using any quadratic integer programming method (the Matlab® function '**ga**' can be used).

7.1.1 A graph-based approach to construct $\mathcal{S}$

This approach starts by calculating an ANN field for the image patches in the known (non-hole) region. Using the calculated field, I generate some proposal patches and feed them to a graph labelling problem whose solution is the skim $\mathcal{S}$. Those patches get a priority to be included in $\mathcal{S}$ but they do not have to. To generate such proposals, I divided the known region in the image into non-overlapping rectangular windows, and high-flag K mode patches in each window. The set of mode patches $\mathcal{M}_p$, which are the proposals, are defined as the ones which appeared most often in the ANN field. A few examples of these proposals are shown on the first row of Fig. 7.3. Afterwards, I constructed the graphical model with nodes representing the patches in the image; this is shown in Fig. 7.1(a). The eventual goal is to obtain an optimal binary map where each node is assigned either the label **1** to be in the skim or **0** to be excluded from the skim. Towards this goal, I

formulated an energy function given by

$$E(L^b) = \sum_{\mathcal{P}_i \in \mathbb{S}} E_d(L^b(\mathcal{P}_i)) \quad + \sum_{(\mathcal{P}_i, \mathcal{P}_j) \mid \mathcal{P}_j \subset \mathbb{S}, \mathcal{P}_j \in \mathcal{N}_i} E_s(L^b(\mathcal{P}_i), L^b(\mathcal{P}_j)), \tag{7.2}$$

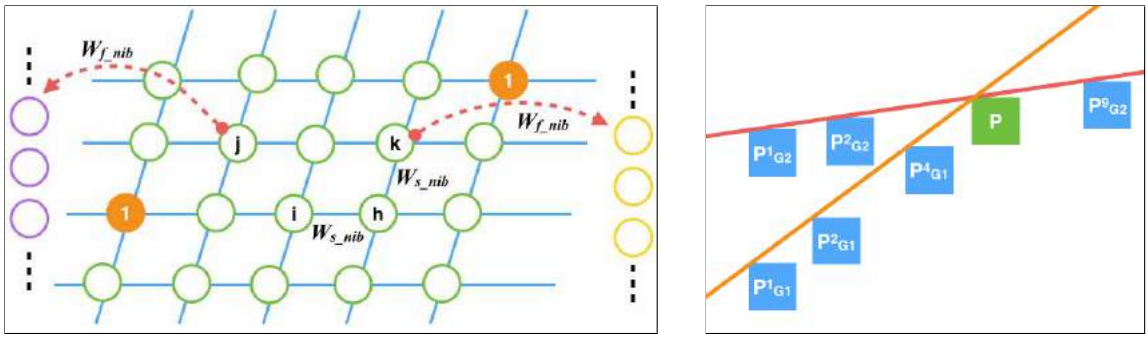
where $\mathcal{P}_i$ is a patch in the image $\mathcal{I}$, $L^b(\mathcal{P}_i)$ is the binary label assigned to $\mathcal{P}_i$ and $\mathcal{N}_i$ is the neighbourhood of $\mathcal{P}_i$. This neighbourhood is comprised of the spatial local neighbours $\mathcal{N}_{s_i}$ of the patch $\mathcal{P}_i$, in addition to $\mathcal{N}_{f_i}$ and $\mathcal{N}_{f_i}^2$ – the non-local feature neighbours of the patch $\mathcal{P}_i$ and the feature neighbours of its feature neighbours respectively, obtained from the ANN field. Since each node is actually a patch, the 'spatial local neighbours' refer to the patches with upper-left corners at a spatial distance that is at most patch-size-away from the patch's upper-left corner. I excluded from $\mathcal{N}_i$ any feature neighbour that is farther than D pixels away from the patch under consideration, and kept this D as a free parameter; the higher it is, the smaller the skim size. A pair of patches are considered connected if and only if they are in each others neighbourhoods. The neighbours-of-neighbours can be excluded from a patch's neighbourhood, but I found that this ***ANN field braiding*** is useful especially if the ANN field is raw, i.e., the approximation of the NN field makes it far from the exact due to the small number of neighbours and the small number of comparisons [87]. The data term $E_d(L^b(\mathcal{P}_i))$ is a constraint that takes the following values:

$$E_d(L^b(\mathcal{P}_i)) = \begin{cases} C_1 & \text{if } L^b(\mathcal{P}_i) = 0 \wedge \mathcal{P}_i \in \mathcal{M}_p \\ 0 & \text{otherwise,} \end{cases} \tag{7.3}$$

where C_1 is a high cost (a positive integer). The smoothness term $E_s(L^b(\mathcal{P}_i), L^b(\mathcal{P}_j))$ is a function of the graph edge weights that are established among the connected patches. These weights are set to the following values:

$$W(\mathcal{P}_i, \mathcal{P}_j) = \begin{cases} -1 & \text{if } \mathcal{P}_j \in \mathcal{N}_{s_i} \wedge \mathcal{P}_j \notin \{\mathcal{N}_{f_i} \cup \mathcal{N}_{f_i}^2\} \\ 1 & \text{otherwise,} \end{cases} \tag{7.4}$$

which read as follows: The weight will be set to -1 if and only if the patch $\mathcal{P}_j$ is a spatial neighbour of the patch $\mathcal{P}_i$ and if the patch $\mathcal{P}_j$ is not among the feature neighbours of $\mathcal{P}_i$ and is not among the feature neighbours of the feature neighbours of $\mathcal{P}_i$; otherwise, the



(a) (b)

Figure 7.1: (a)The proposed graphical model. In this graph, each site represents a patch, each of which has spatial local neighbours, in addition to feature (RGB pattern) non-local neighbours. Spatial neighbours are connected with the weight W_{s-nib} while the feature neighbours are connected with the weight W_{f-nib}. Both types of weights are set according to Eqn. 7.4; they are given different symbols for the sake of clarity. Building the graph is tailored to encourage spatially-connected skimmings and to reject the inclusion of close feature neighbours in $\mathcal{S}$. The sites labelled '1' are proposed to the algorithm as an input; they get a high privilege to be in $\mathcal{S}$ but they do not have to be included. (b) An illustration of the patch-slope nomination step. In this step, for every patch $\mathcal{P}$ in the skim, depicted by the green patch, and for every slope in the refined $\{G\}$, a line is extended from the upper-left corner of the patch. Two examples of these lines are the red and the orange lines. The mean Euclidean distances between the RGB pattern of $\mathcal{P}$ and those of the patches whose upper-left corners lying on each of those lines are then measured. The slope G^* that gives the minimum mean Euclidean distance is nominated for $\mathcal{P}$.

weight will be set to 1. The smoothness term $E_s(L^b(\mathcal{P}_i), L^b(\mathcal{P}_j))$ is then formulated as

$$E_s(L^b(\mathcal{P}_i), L^b(\mathcal{P}_j)) = \begin{cases} C_2 \times W(\mathcal{P}_i, \mathcal{P}_j) & \text{if } L^b(\mathcal{P}_i) = 0 \wedge L^b(\mathcal{P}_j) = 0 \\ C_3 \times W(\mathcal{P}_i, \mathcal{P}_j) \times (1 - |L^b(\mathcal{P}_i) - L^b(\mathcal{P}_j)|) & \text{otherwise,} \end{cases} \tag{7.5}$$

where $C_3 > C_2 > C_1$. Table 7.1 enumerates the different cases encountered by the smoothness cost term and the corresponding cost assigned in each case.

In Fig. 7.1(a), spatial neighbours are connected with the weight W_{s-nib} while the feature neighbours are connected with the weight W_{f-nib}. Both types of weights are set according to Eqn. 7.4; they are given different symbols merely for the sake of clarity. According to the formulated smoothness term, our energy is inherently non sub-modular. For binary problems, sub-modularity holds if and only if

$$V_{\mathcal{P}_i,\mathcal{P}_j}(0,0) + V_{\mathcal{P}_i,\mathcal{P}_j}(1,1) \leq V_{\mathcal{P}_i,\mathcal{P}_j}(0,1) + V_{\mathcal{P}_i,\mathcal{P}_j}(1,0). \tag{7.6}$$

To minimize Eqn. 7.2, I have tried the QPBO [57] to infer the optimal binary map using the publicly available code of Ref. [4][1]. An inherent limitation in the QPBO, however, is its partial optimality, i.e., it guarantees an optimal labelling but it does not guarantee a complete labelling, which may leave important patches unlabelled, and consequently not included in $\mathcal{S}$. Hence, I used the LSA method [56] using the publicly available code in [91].

Table 7.1: Different cases encountered by the smoothness cost term, in Eqn. 7.5, and the corresponding cost assigned in each case.

Case	**Cost**
Spatial neighbours and look alike and both labelled zero	$1 \times C_2 = C_2$
Spatial neighbours and look alike and labelled differently	$1 \times C_3 \times 0 = 0$
Spatial neighbours and look alike and labelled one	$1 \times C_3 = C_3$
Spatial neighbours and do NOT look alike and both labelled zero	$-1 \times C_2 = -C_2$
Spatial neighbours and do NOT look alike and labelled differently	$-1 \times C_3 \times 0 = 0$
Spatial neighbours and do NOT look alike and labelled one	$-1 \times C_3 = -C_3$

7.1.2 An iterative heuristic to construct $\mathcal{S}$

Similar to the graph-based skimming method, this approach starts by calculating an ANN field for the image patches in the non-hole region, followed by dividing the known region in the image to non-overlapping rectangular windows, and high-flagging K mode patches in each window. While the similarity condition is satisfied by the mode patches in $\mathcal{S}$, the orthogonality condition is satisfied by the following step which enforces the smoothness of the binary skim field. I loop over the set of non-mode patches $\mathcal{Q}$ and grant a patch the membership in $\mathcal{S}$ if and only if none of its near feature neighbours has been already included. So far, the smoothness assumption has not been enforced yet. It is enforced when the second case takes place, i.e., when a near feature neighbour has been already included in the skim. In this case, a final check is made to see if it is spatially nearer to the rest of $\mathcal{S}$ than the non-mode patch under consideration. If the latter is nearer, I include it in the skim and remove its near feature neighbour. Although this procedure is iterated in a greedy manner, I found that it produces more spatially connected skims. To

[1] http://www.robots.ox.ac.uk/%7Eojw/software.htm

show this, I chose a subset of the images[2] that were used to test the performance of the algorithm. The size of this subset is 15 images. For each of them, I calculated the image skim using the aforementioned heuristic with and without the smoothness enforcement step. For every calculated skim, I looped over the chosen patches and I calculated the distance of every patch to its nearest patch among the skim. The distance is defined as the spatial Euclidean distance between their upper-right corners. I then calculated the mean distance and called it the *mean inter-patch distance*. If the smoothness enforcement step results in more spatially connected skims, their mean inter-patch distance should be smaller. Fig. 7.2 shows a comparison between the mean inter-patch distance of the image skims computed with and without the smoothness enforcement step. It shows that the former is consistently smaller than the latter. A few example skims can be found in Fig. 7.3.

Last but not least, I consider any feature neighbour that is less than D pixels away (from the patch under consideration) to be a near feature neighbour. Accordingly, there are three free parameters in this procedure, which were set empirically: the number of neighbours in the calculation of the ANN field, the size of the rectangular windows used to nominate the mode patches, and the value of D. The size of the skim is proportional to the size of the final 'bag of shifts' that will be used to paint the hole, as will be seen in the following sections. Hence, while setting the aforementioned free parameters, the size of that 'bag of shifts' was taken in consideration simultaneously with the visual quality of completions. The pseudocode of the heuristic is shown in Algorithm 2.

7.2 The Hole Filling Step

7.2.1 Nominating a slope for every patch in $\mathcal{S}$

My eventual goal is to slide each patch in the calculated skim $\mathcal{S}$ along particular slopes to fill the hole. Towards the goal of identifying those particular slopes, I started by collecting a raw set of slopes which is calculated from the skim itself. Specifically, for every patch

[2]The images used in my experiments are a part of the Microsoft MSRA Salient Object Database and the Microsoft Research Cambridge Object Recognition Image Database, which can be found respectively at:
http://research.microsoft.com/en-us/um/people/jiansun/SalientObject/salient_object.htm
http://research.microsoft.com/en-us/downloads/b94de342-60dc-45d0-830b-9f6eff91b301/

Algorithm 2 A greedy heuristic to construct an image skim $\mathcal{S}$

1: **procedure** GetSkim(*Image, Patch Size, Number Of Neighbours, Window Size, Maximum Distance*)
2: Calculate the ANN field
3: Divide the known region to non-overlapping windows of size *Window Size*
4: Determine the mode patches in every window
5: NonModePatches = Number of non-mode patches
6: **while** NonModePatches **do**
7: Get the feature neighbours of the current non-mode patch, not farther than *Maximum Distance*
8: **if** No near feature neighbour already exists in $\mathcal{S}$ **then**
9: Add the current non-mode patch to $\mathcal{S}$
10: Remove the current non-mode patch from $\mathcal{Q}$
11: NonModePatches = NonModePatches - 1
12: **else**
13: Identify the neighbour of the current non-mode patch in $\mathcal{S}$
14: Check which of them is nearer to the rest of $\mathcal{S}$
15: **if** The current non-mode patch is nearer **then**
16: Remove the near feature neighbour patch from $\mathcal{S}$
17: Remove the current non-mode patch patch from $\mathcal{Q}$
18: Add the current non-mode patch to $\mathcal{S}$
19: NonModePatches = NonModePatches - 1
20: **end if**
21: **end if**
22: **end while**
23: **end procedure**

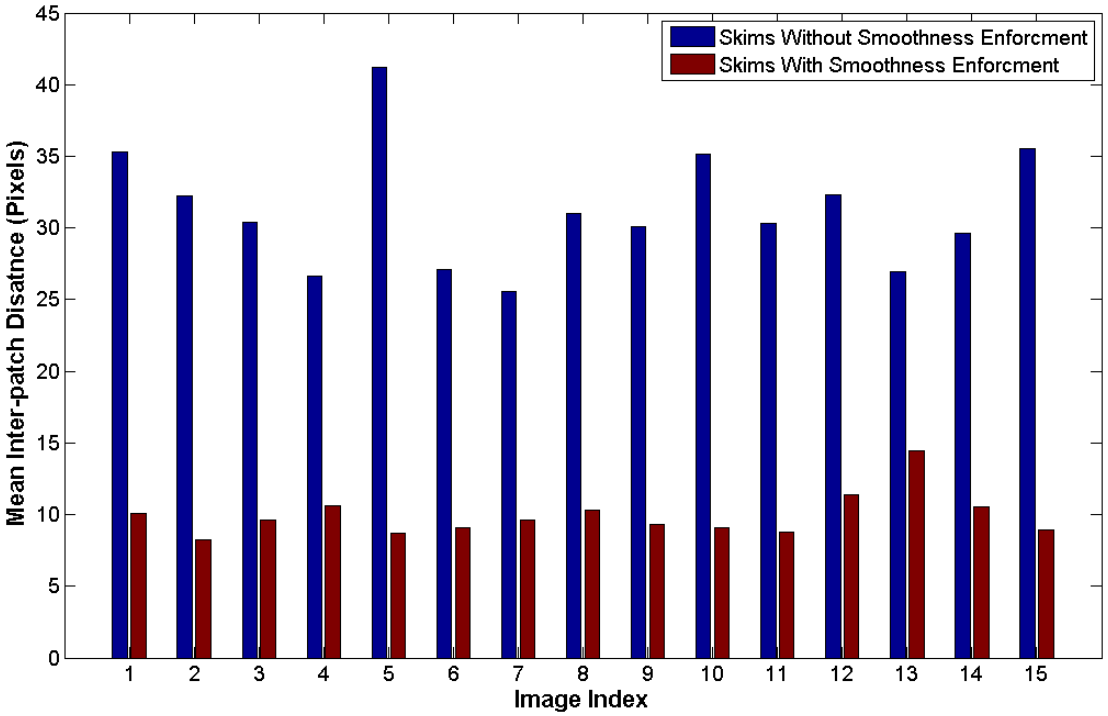


Figure 7.2: Comparing the spatial connectivity of the image skims computed with and without the smoothness enforcement step.

$\mathcal{P}$ in $\mathcal{S}$, I determined its most similar member in the skim. The initial set of slopes $\{G\}$ is then defined by the gradients of the lines joining the upper-left corners of every pair of similar patches. This step involves the patches in the skim only, and every patch adds a slope to the set of slopes $\{G\}$. I then refined that set through the following two steps. First, I discarded the slopes that correspond to lines which do not pass through the hole region. Second, I quantized the slopes and retained those that fall in the three most-frequent bins only.

Afterwards, for every patch $\mathcal{P}$ in the skim and for every slope in the refined $\{G\}$, a line is extended from the upper-left corner of the patch. The mean Euclidean distance between the RGB pattern of $\mathcal{P}$ and those of the patches lying on that line is then measured and the slope G^* that gives the minimum mean Euclidean distance is nominated for $\mathcal{P}$. This step is called the 'patch-slope nomination step' through the rest of this document. Its output can be described as a matrix – the patch-slope nomination matrix, which stores the corners of the patches and the best slope to slide each of them along. An illustration

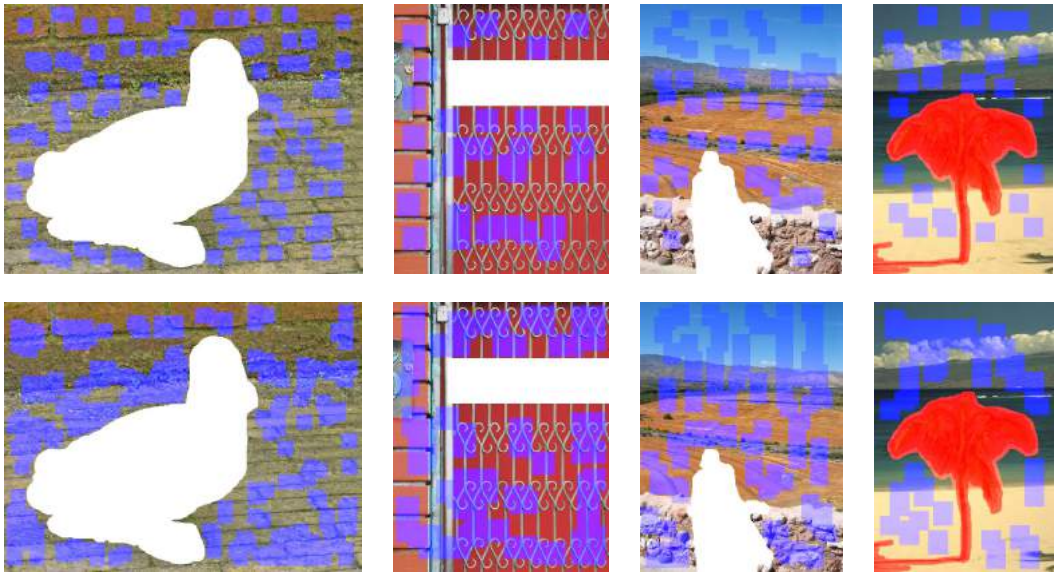


Figure 7.3: A few examples of image skims calculated using the iterative heuristic approach discussed in sub-section 7.1. The patches included in the skim are blue-masked in the images. The first row and the second row show the skims calculated with and without the smoothness enforcement step respectively. The images in the second row shows that the blue masked patches are more connected compared to the skims in the first row. The figure should be seen in color.

for this step is shown in Fig. 7.1(b) and it can be formally written as:

$$G^* := \underset{G_i}{\operatorname{argmin}} \frac{1}{N} \sum_{j=1}^{N} \| \mathcal{P} - \mathcal{P}^j_{G_i} \|^2 \quad \forall\, i = 1, 2, ..., N, \tag{7.7}$$

where $\mathcal{P}^j_{G_i}$ is the j^{th} image patch with upper-left corner lying on the line defined by the slope G_i and the upper-left corner of $\mathcal{P}$. The set of all G^*s will be denoted $\mathcal{G}$. If the skim is quite representative, the slopes in $\mathcal{G}$ should give an information about the dominant filling directions (or filling patterns) in the image, which well-guide the following step in the pipeline – the hole painting step. An example of such filling patterns is shown in Fig. 7.4 as dotted cyan lines.

While calculating $\mathcal{G}$, I did not consider the skim patches that are farther than 200 pixels from the hole. This complies with the proper setting of the boundary conditions as will be seen in the discussion on graph construction. Also, for a particular patch, prior to calculating the mean Euclidean distance for every slope, I disregarded the distances that are more than 25 times the minimum distance in that vector. The output of this step is the patch-slope nomination matrix, which stores the set $\mathcal{G}$ and the corresponding patch

corners; actually, the latter is a subset of the corners in $\mathcal{S}$. I also stored the minimum mean Euclidean distance that resulted in every patch-slope nomination, and the entries whose distances are larger than 25 times *the minimum of the minimum mean Euclidean distances* are omitted from the patch-slope nomination matrix.

The patch-slope nomination matrix can be further refined using the boundary conditions of the hole. I did this step using the following procedure. When every patch in the patch-slope nomination matrix is slid along its corresponding slope in $\mathcal{G}$, it brings non-hole pixel values to a subset of the set of the hole's boundary pixels. I calculated the nearest non-hole pixel (bounding pixel) to every boundary pixel and determined which patch in the patch-slope nomination matrix brings to every boundary pixel a known pixel that is most similar (according to the RGB feature) to its nearest bounding pixel. If a patch in the patch-slope nomination matrix did not succeed to bring any 'good non-hole pixels' to any boundary pixel, their entry is omitted from the patch-slope nomination matrix. In addition, if two corners in the matrix are less than $\sqrt{2} \times$ (Patch Size/2) apart, I retain only one of them in the matrix. Lastly, I limited my final bag of slopes so that it is comprised of the two most-frequent slopes in that refined patch-slope nomination matrix.

7.2.2 Painting the hole

In this step, every corner in the patch-slope nomination matrix (which is a subset of the corners in $\mathcal{S}$) is slid along each slope in the final bag of slopes. Whenever the upper-left corner of $\mathcal{P}$ traverses a pixel in the hole, the shift between them is calculated. If the sliding of a certain patch along a particular slope does not intersect the hole, it is discarded. The calculated shifts are then binned to buckets of size equal to the patch size used in the calculation of the ANN field. With the aggregated set of shifts, I sought an optimal shiftmap by solving a graph labelling problem, in a way similar to the previous techniques in the literature [26, 28, 39] using multi-label graph cuts [90, 55, 52, 91]. My energy function is given by

$$E(L) = \sum_{p_i \in \Omega} E_d(L(p_i)) \quad + \sum_{(p_i,p_j)\,\forall\, p_i \in \Omega, p_j \in \Omega} E_s(L(p_i), L(p_j)), \tag{7.8}$$

where Ω is the hole region in the image, p_i is a pixel in Ω and $L(p_i)$ is the shift assigned to p_i. The data term $E_d(L(p_i))$ is a constraint which assigns a zero cost to a shift (label) that moves the hole pixel p_i to a known pixel location, and assigns an ∞ cost to a shift if it moves the hole pixel to another hole pixel or to outside the image lattice. The smoothness cost $E_s(L(p_i), L(p_j))$ for the two arbitrary shifts s_a and s_b, between the neighbouring pixels p_i and p_j (4-connected neighbourhood) is given by:

$$E_s(p_i, p_j, s_a, s_b) = \|I(p_i + s_a) - I(p_i + s_b)\| + \|I(p_j + s_a) - I(p_j + s_b)\|, \qquad (7.9)$$

where $I(\cdot)$ is the color of the image at the specified attribute and $\|\cdot\|$ is the Euclidean norm. If s_a and s_b are not equal, p_i and p_j will be moved by different offsets and the smoothness cost will increase *unless* the shifting results in coherent seams [26], i.e., similar RGB values between the shifted pixels.

For setting the boundary conditions, I adopted the following procedure during graph construction. Similar to [26], I included the bounding pixels among the sites in the graph labelling problem. The bounding pixels are those pixels that are one-pixel away from the boundary pixels; the former are non-hole pixels while the latter are hole pixels. For those pixels in particular, a zero-shift is introduced in addition to the aggregated bag of shifts. The data term then serves to constrain that zero-shift only for the bounding pixels, so that it becomes invalid for any other site. The data term also serves to hamper the accessibility of the bounding pixels to any other shift except that zero-shift. For the boundary pixels, each of them is allowed only one shift, that is the shift that brings to it a non-hole pixel that is most similar (according to the RGB feature) to its nearest bounding pixel.

In addition to producing a near-globally-optimal solution, the proposed method avoids some critical limitations in the state-of-the-art techniques. First, as opposed to [28], it considers a limited number of possible shifts for completing the hole; this does not only impact the computational efficiency positively, but also contributes to the accuracy of completions [26]. Second, on the contrary to [26], I avoid the dependence on calculating the dominant offsets from the known region, since the success of this approach depends heavily on the size of the known region compared to the hole. The proposed method uses image skims, which are used at a later stage to figure out the dominant slopes, along

which the completion would be most plausible.

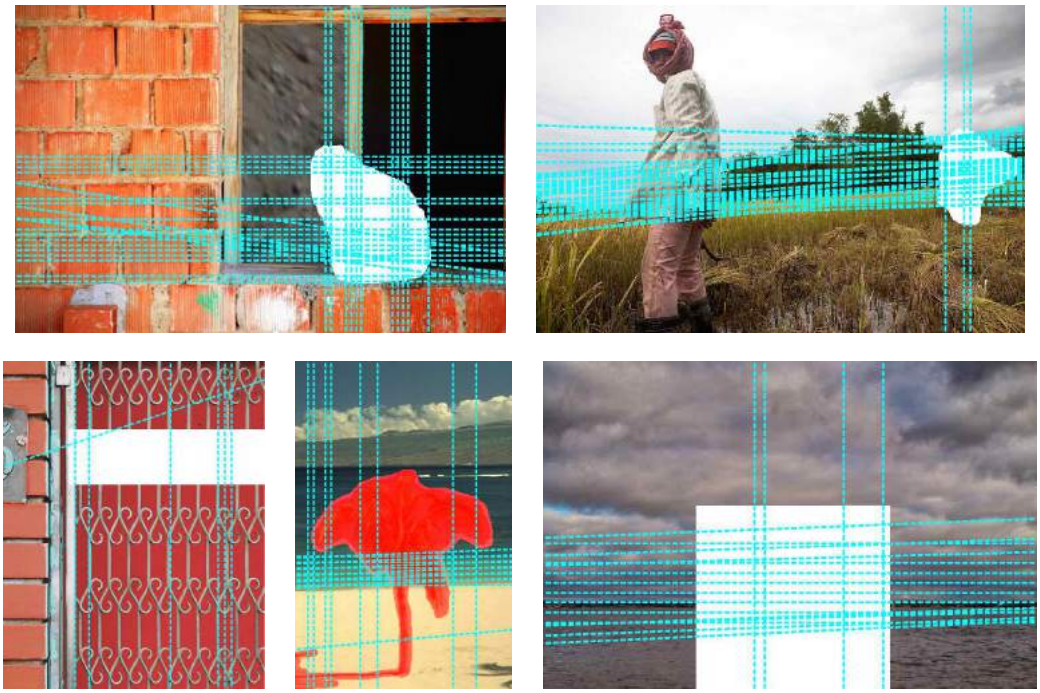


Figure 7.4: Dominant slopes (filling patterns) extracted from the calculated image skims, shown in cyan.

I compared my results with three of the most recent and the most relevant techniques in the literature, namely, hole-filling using statistics of patch offsets [26], content-aware filling [23, 24, 25] and shiftmap image editing [28]. All the results reported in this document for the aforementioned techniques were either cropped from [110][3] or acquired from [111]. I have used VLFeat [87] to calculate the ANN field of quarterly-overlapping patches that tile the whole image plane. The patch size was fixed to 32×32 pixels throughout the experiments and I considered five neighbours for every patch. While the technique in [26] considers a constant number of shifts (60 shifts) to fill the hole, the number of shifts considered by my technique is varying; it mainly depends on the skim size, the position of the skim patches and the size of the hole. Some images result in a number of shifts that is much less than 60, other cases required the consideration of more shifts. The most computationally-demanding step in my algorithm is the patch-slope nomination. While the skimming process takes around one second, the slope nomination step results in slower

[3]Those results were re-printed in accordance with the IEEE rules regulating the re-usage of copyrighted material by individuals working on theses. A printed copy of the permission grant is attached in section A.9. Copyright ©2014, IEEE.

completions compared to the timings reported in [26].

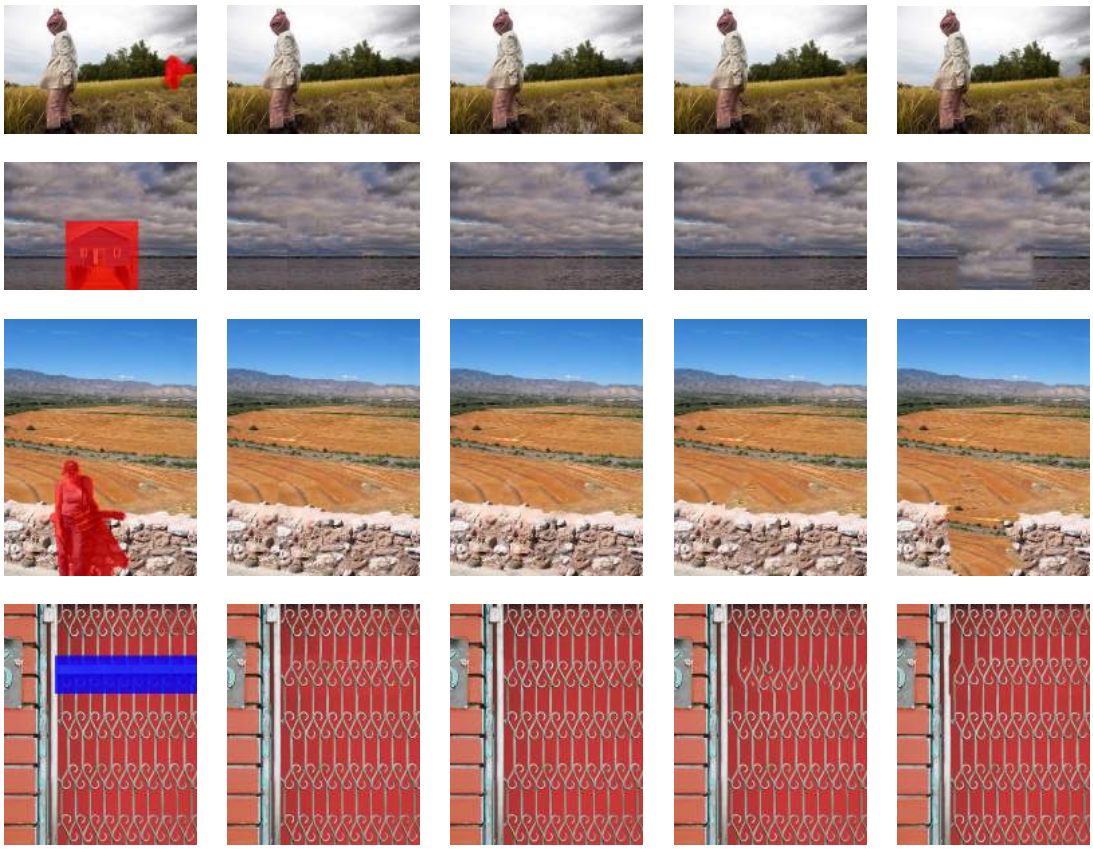
In Fig. 7.5 and Fig. 7.6, it can be seen that the proposed method and [26] are the only methods that consistently yield plausible completions. Nevertheless, the dominant patch offsets can repeat efficiently the image patterns, but it fails if the repetition 'period', determined from the known region, is less than the size of the hole. This limitation can result in failing the completion even for trivial cases as the one shown on the fourth row in Fig. 7.6. Since the shifts in my proposed method are obtained based on the nominated slopes and the hole size, the proposed method is able to complete the image successfully. It is worth mentioning that the method presented in [26] performs two more steps in addition to the computation of an optimal shift map, which are the misalignment removal (this involves solving another graph labelling problem) and the Poisson image blending for the around-hole seam removal. I postponed the implementation of those two steps for future work, in addition to other aspects that will be stated in the following paragraph. Figure 7.7 and Fig. 7.8 show more results for the proposed algorithm. Lastly, Fig. 7.9 shows some cases of failure.

There are some parameters in the current implementation of my technique that need more analysis, in terms of their effect on the computational complexity and the quality of completions. For example, the user is prompted to choose a bounding box around the hole, from which the algorithm computes the skim. This bounding box can be trivially set to be the whole non-hole region. The effect of changing the size of that bounding box on the performance needs further analysis. The patch size, the adopted feature vector for the pixels, the parameter D that was mentioned at the end of sub-section 6.1.2, and the weight of the smoothness term in the graph labelling problem are other parameters that may affect the performance of the presented technique, and thus require more analysis. Last but not least, in the current implementation, I limit the final bag of slopes to have a size of two (the two most-dominant ones); however, it could be better to determine the size of the final bag of slopes adaptively, based on the number of dominant structures in the image. Particularly, I am interested in exploring the possibility of augmenting my method with the real-time edge-maps of [85] to adaptively discard/retain slopes.

Scene completion is the eventual goal of the research presented in this thesis. Moreover, the author of this document believes that ideas are open-source, while their for-

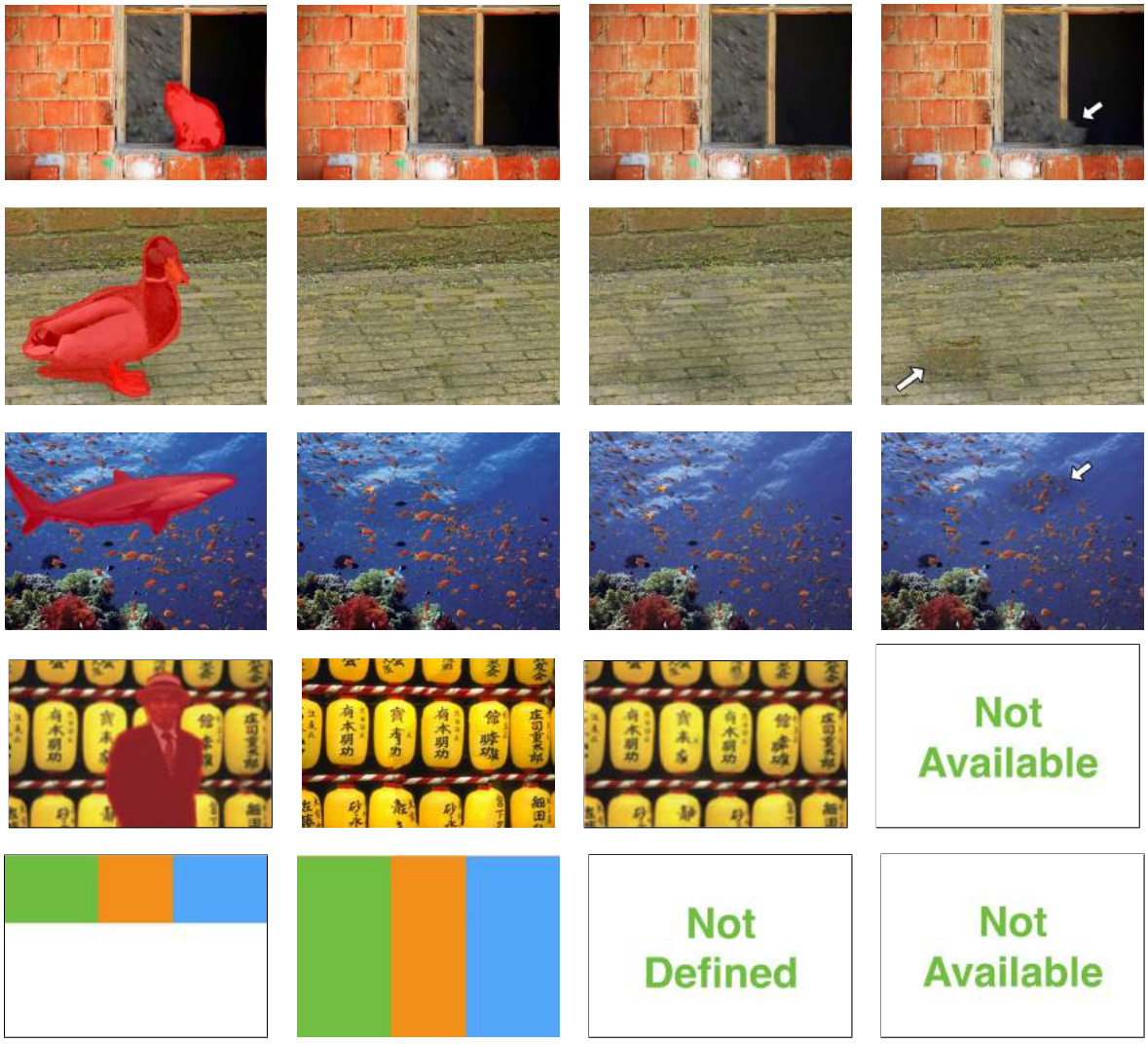
mulation and their implementation are not. Hence, and for the sake of completeness of presentation, I dedicate section A.7 for discussing a possible formulation for extending the suggested single-image completion method to the problem of depth-guided dis-occlusion management.

In this chapter, I presented a new method for filling holes in single images. The proposed pipeline started by skimming the image through nominating a few representative patches in it. Towards this goal, a heuristic and a graph-based method for image skimming were proposed. Dominant slopes in the input image are then calculated using the constructed image skim, before sliding the skim patches along these slopes. The latter step results in a bag of shifts, from which a near-globally-optimal shift map is computed using multi-label graph cuts. While suffering from high time complexity, the performance of the proposed method was shown to be on par with the SoA techniques, while overcoming two of their critical drawbacks.



(p) Image (q) Ours (r) Results of [26] (s) Results of [25] (t) Results of [28]

Figure 7.5: Results of the proposed algorithm (second column) compared to Ref. [26] (third column), content-aware filling[23, 24, 25] (fourth column) and shiftmap image editing[28] (fifth column). The input images with the red-masked and the blue-masked holes (first column) and the results of the latter three techniques were cropped from Ref. [110] or acquired from Ref. [111].



(q) Image (r) Ours (s) Results of [26] (t) Content-aware filling

Figure 7.6: More results from the proposed algorithm compared to [26] and content-aware filling [23, 24, 25]. The results of the two aforementioned techniques were cropped from [110] or acquired from [111]. The fourth row is not an illustration, it is a real result. While the proposed technique succeeded to complete the hole by nominating good slopes, the method of [26] will lack the proper shifts to complete the image. We do not have an access to content-aware fill, so its result for that image is not available.

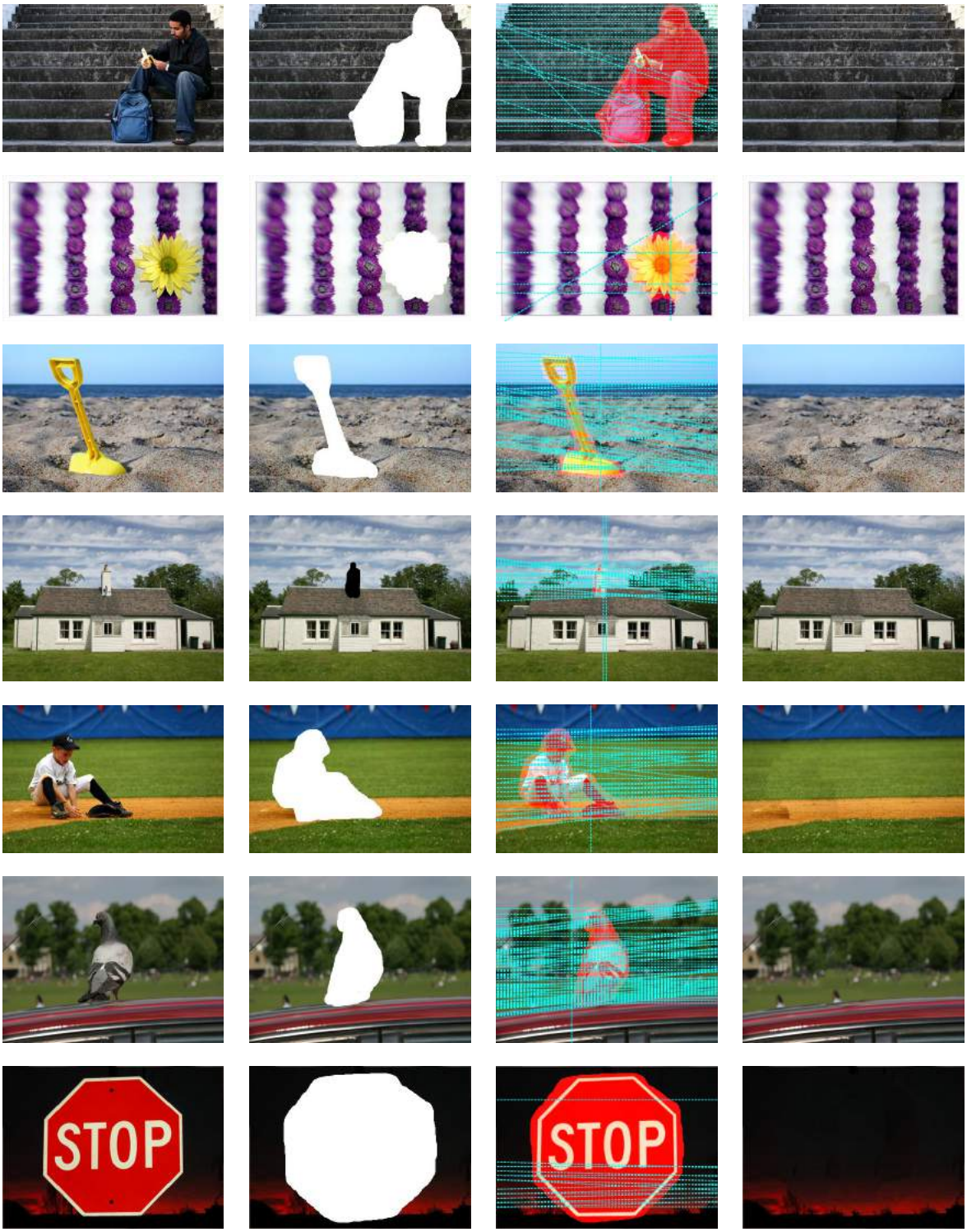


Figure 7.7: More results for the proposed image completion technique. The first, second, third and fourth columns show the original images, masked images, filling patterns and completed images respectively. Please see text for details about the databases from which the images were acquired.

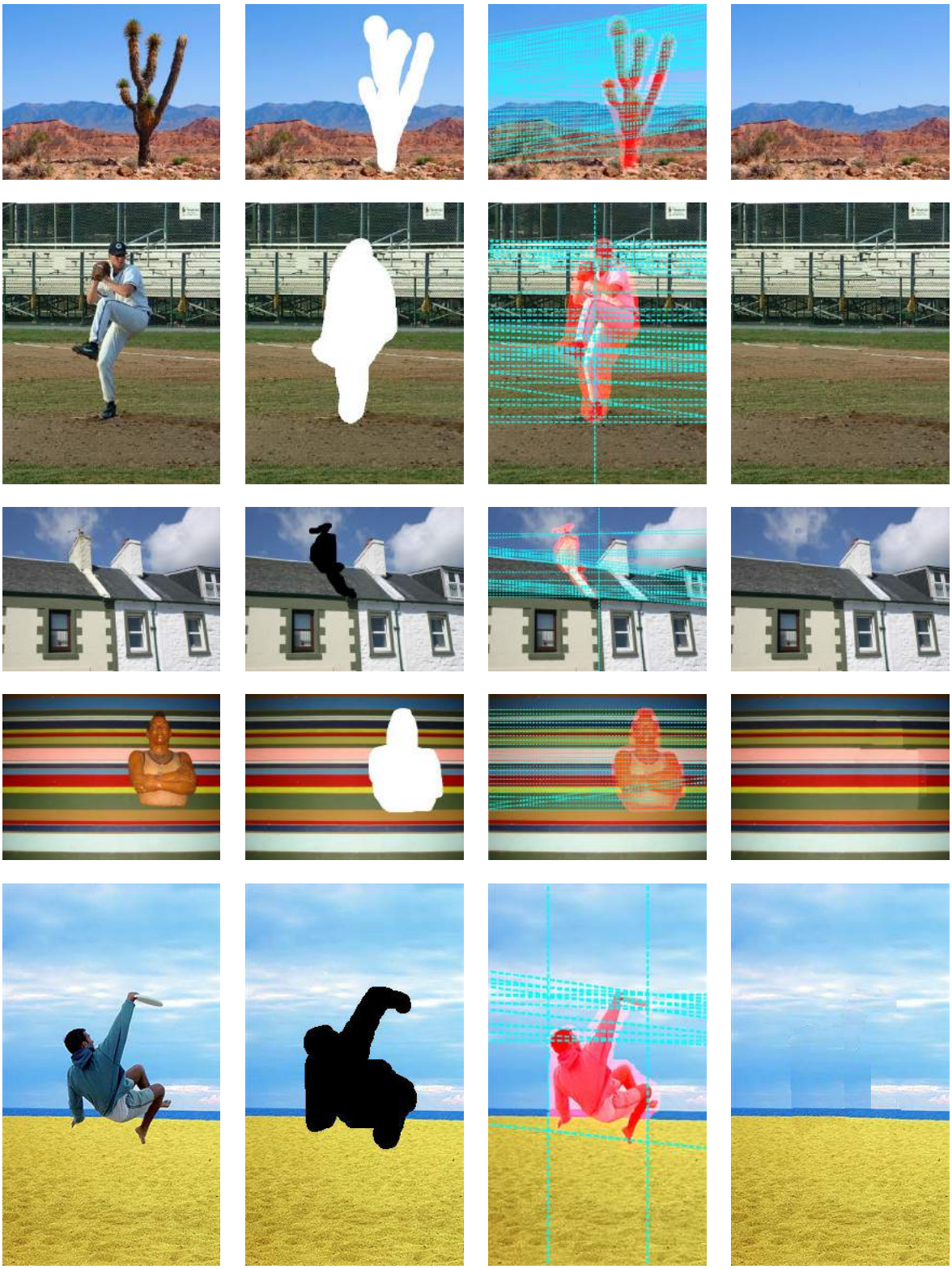


Figure 7.8: More results for the proposed image completion technique. The first, second, third and fourth columns show the original images, masked images, filling patterns and completed images respectively. Please see text for details about the databases from which the images were acquired.

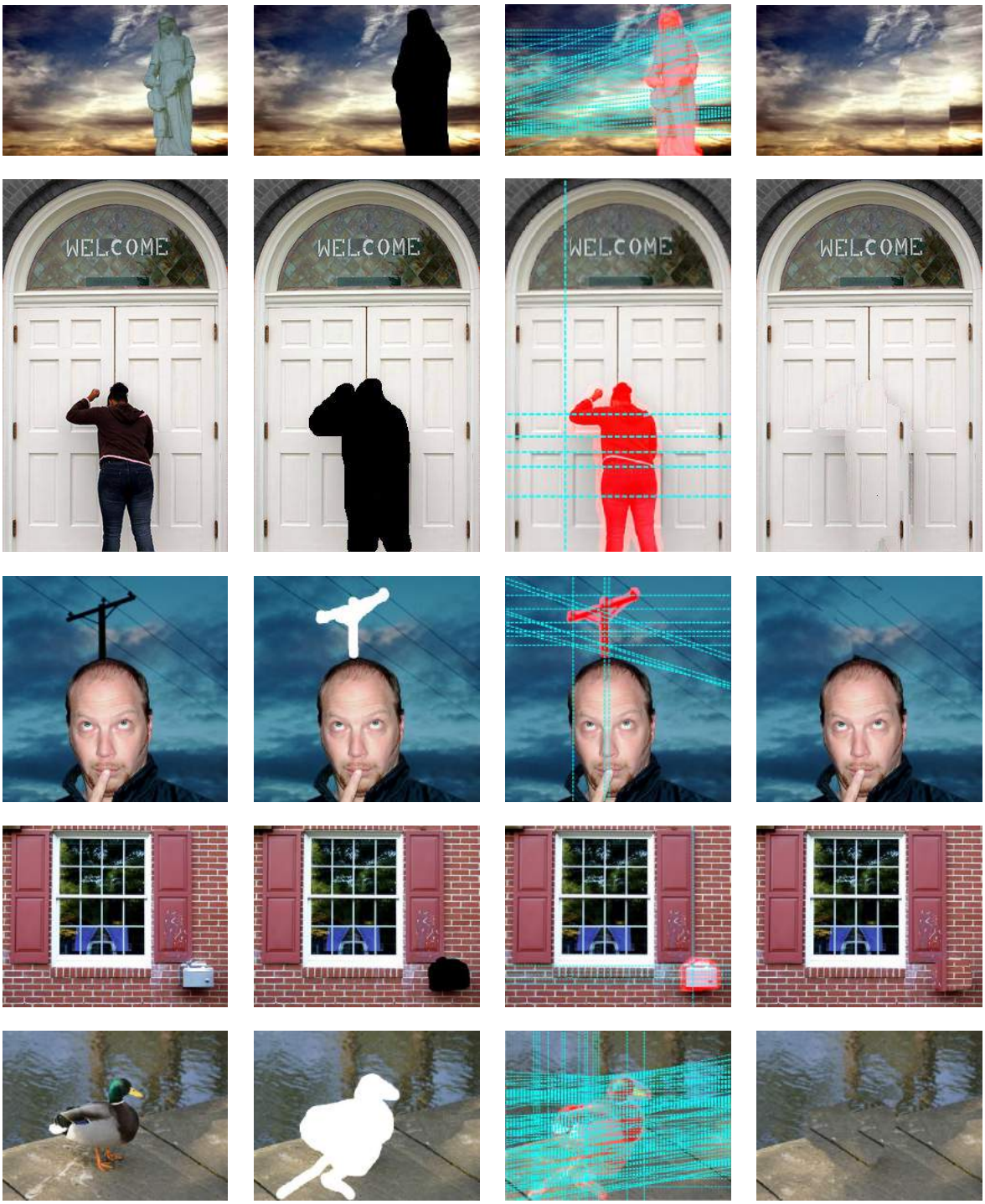


Figure 7.9: A few failure cases. The first, second, third and fourth columns show the original images, masked images, filling patterns and completed images respectively. Please see text for details about the databases from which the images were acquired.

Chapter 8

Conclusions

This thesis is inspired by the significant impact that image-based rendering systems have on a wide range of applications. It conceives image-based rendering as a mother problem in computer vision, with several constituent sub-problems, and it addresses two of those problems, namely, image compositing and image completion. After pinpointing a number of open problems in the literature, the thesis adopts a framework that is based on graph theory, and utilizes techniques from the literature of image decomposition, machine learning, mathematical optimization and perceptual grouping to propose new solutions for these challenges.

8.1 Thesis Summary

- In Chapter 1, the thesis' problem is defined. I give an overview on the spectrum of different techniques for image-based rendering systems, and I highlight view-interpolation-based novel view synthesis as an IBR technique of specific concern to the scope of the thesis. This is followed by a detailed discussion on the relation of NVS, as a mother problem, and other constituent problems including image compositing and completion.

- Chapter 2 features the mathematical modelling of the problems addressed in the thesis. It also presents and follows a map of concepts that collectively represent the theoretical background of the thesis. Image cues, general graph theory concepts and matrix representations, graph cuts as well as manifold learning and label propagation

methods are discussed. I also mention briefly a part of the literature of image-analysis-related problems that have adopted the aforementioned concepts. Finally, I state how and where each of those concepts are used in my research.

- In Chapter 3, a comprehensive review of the related literature is presented. It starts by reviewing the recent advances in sampling-based matting and what makes it more suitable for image-based rendering applications, compared to other matting techniques. The open problems in the literature are then enumerated, particularly, the challenges that are tackled by this thesis' research contributions. The rest of the Chapter is dedicated to a review on recent hole filling and dis-occlusion management techniques, and how they fall short with respect to the open problems in the literature and the requirements of image-based rendering systems as well.

- Chapters 4, 5, 6 and 7 discuss the contributions of the thesis. I start by discussing the proposed research on the matting problem. Chapter 4 presents new strategies for trimap sampling, Chapter 5 presents a new pipeline for sampling-based matting, involving new methods for sample gathering and pair assessment, while Chapter 6 highlights a new framework for generating trimaps automatically. Lastly, Chapter 7 is dedicated to explain a novel single image completion technique. The content of this Chapter is complemented by a section in the appendix that explains how the proposed technique can be extended to complete scenes and panoramas using texture+depth+temporal information.

8.2 Thesis Contributions

This thesis addresses several drawbacks in the existing techniques of image compositing and image completion. I sum up the contributions of the thesis as follows:

- For natural image matting, I point out two problems that impede its incorporation in IBR systems, and I address both of them in my research. The first problem is the robust sampling of trimaps. I discuss the pipeline of the most recent techniques in the literature, and I highlight their strengths and drawbacks which show that the development of more efficient sampling strategies is still an open problem.

With regards to the formulation of a new sampling strategy, I present the motivation behind my proposed concept before proceeding to my first attempt on realizing sequential pair selection matting. The contribution of that initial attempt is conceptual, since the results are not on-par with today's literature of sampling-based matting. Nevertheless, I explained how it avoids many pitfalls of the current sampling-based techniques. Most importantly, it does not suffer from the color ambiguity problem since it relies on determining a good half-pair and then complements it with a good complement half-pair. Towards this goal, I quantify the overlap between the color distributions of the unknown regions and their neighbouring known regions to decide whether the FG/BG samples should be acquired locally or non-locally.

The proposed method was evaluated according to an online benchmark in January 2014, and attained at that time a reasonable position among the rest of the techniques. For a few trimaps of particular images, it attained the first position in the benchmark according to a particular metric (out of the four adopted metrics). Nevertheless, it suffered from critical drawbacks such as the large size of the shortlisted pair pool and the non-robust feature vector used. This leads to performance deterioration if the BG and the FG distributions do overlap. Hence, I proposed another sampling technique that adopts the same sequential pair selection framework while addressing the drawbacks of my first attempt.

The second variant I have proposed for trimap sampling using sequential pair selection adopts a more robust color-texture feature vector (robustness is against overlap between color distributions), that is constructed by a computationally efficient cartoon-texture decomposition. The proposed method also exploits the redundancy among the known and the unknown neighbouring pixels. This is done by applying the local linearity principle on the SLIC pixels of the image under consideration. My proposed technique is thus the only method that makes use of that redundancy in the literature of sampling-based matting, which drastically impacts the time complexity of the matte computation. It also determines the best half-pair using a more reliable approach than the one used in my first realization of the sequential pair selection. The best half-pair is determined by solving a binary graph transduction problem.

The process of graph transduction propagates the certainty that a particular region is a FG(BG), or similar to a FG(BG), to its neighbouring unknown regions in the trimap. On the 23rd of May 2015, my proposed algorithm attained state-of-the-art results and it was on-par with the top performers in the online matting benchmark.

I also extend the classical chromatic cost function that is used in the first version of the algorithm, to encourage the sparsity of trimaps as well as the FG/BG half-pairs that come from well-separated color distributions. The performance of the graph-transduction-based sequential pair selection, together with the extended cost function (for pair assessment) is consistently better than the first version of the algorithm that uses the classical chromatic distortion. This is verified on the training dataset and is acknowledged by the online matting benchmark.

I conclude my contributions to sampling-based matting by proposing a third variant of the sequential pair selection strategy; it is a novel pipeline that involves new methods for sample gathering and pair assessment. I incorporate accurate and computationally efficient feature maps to guide the process of sample gathering. Another variant of propagation-based half-pair computation is adopted, and the pair assessment step was formulated as a graph labelling problem. While casting that problem, half-pair constraints are embedded as smoothness constraints in an energy function minimized by multi-label graph cuts. The proposed pipeline was evaluated on the *18th of April 2016* on the matting benchmark. As shown by an objective assessment of their performance on the whole training dataset of the benchmark, the proposed pipeline has a number of merits over the other matting techniques presented in the thesis.

The third variant of the sequential pair selection strategy has been shown to have some merits with regards to subjective quality, as compared to the second variant. Nevertheless, the overall performance of the second variant, as indicated by the objective evaluation on the training and testing datasets of the matting benchmark, was better than the third variant. Hence, the second variant is recommended over the other proposed variants of sequential pair selection.

- Another matting-related problem that is addressed by the presented research is the

automatic trimap generation. While this problem may not be tangible in matting-for-editing applications, it is pivotal in NVS. I give the reasons for considering this topic as an open problem, and I show that it has not been adequately addressed in the literature. To clearly identify my goal, I list the characteristics of an ideal trimap generator before I present the details of my proposed technique.

My proposed trimap generator has two characteristics of a perfect generator. First, it is completely automatic, so no user interaction is required at any stage of the pipeline to acquire a trimap. Second, it does not rely on bounding boxes to produce the trimap; instead, it aligns with the fuzzy object boundaries, which is an advantage in regards to the reduction of the number of alpha values to be computed in the alpha map. My method is not the first in the literature to incorporate the laws of visual grouping for segmenting a foreground object from a background. Nevertheless, the algorithm improves an automatic matting method, spectral matting, by developing a novel objective function that employs the Gestalt laws of perceptual grouping for the efficient grouping of matting components. This objective function favours the alpha maps that satisfy the Gestalt laws of visual perception; particularly, I propose to assess the concavity, symmetry and connectedness of a grouping. I also propose a probabilistic formulation of the problem that incorporates the proposed method in a framework that takes into consideration the other requirements of an ideal trimap generator. On the other hand, the algorithm still suffers from a few drawbacks, which may trigger the prospective research directions in the next section.

- Dis-occlusion management is the second topic addressed by my research; I am concerned with one of its principal constituent challenges – single image completion. My technique for single image completion suggests a solution for two critical drawbacks in the state-of-the-art methods. I formulate a novel framework that involves the development of a heuristic and a graphical model for image skimming; for the latter, the skim is constructed by solving a binary optimization problem. Using statistics on the constructed skims, I infer the slopes along which the completions would be most plausible subjectively, before I slide the skims along those slopes. I then collect a bag of shifts and seek a near-globally-optimal shift map using multi-label graph cuts. I show that my technique is on par with three other techniques

developed by distinguished research labs. It gives subjectively plausible results, and is generic enough to deal with some challenges that other techniques failed to handle. On the other hand, the proposed method still suffers from high computational time complexity; this becomes critical if the proposed method is expected to be incorporated in IBR systems, and is a point of future enhancement.

In summary, the specific contributions of the thesis are as follows:

1. A new trimap sampling algorithm that avoids the color ambiguity problem. The ambiguity was eliminated by following a sequential fashion in nominating good samples from the foreground and the background for every unknown pixel/super-pixel. It achieved overall reasonable results in the standard matting benchmark on the date of submission. This work was published in [11][1].

2. A new trimap sampling method that overcomes the disadvantages of the aforementioned technique while retaining its merits. It adopts a more robust feature vector to represent pixel values and presents a graphical framework for realizing sequential pair-selection; it nominates a known super-pixel (as a suitable half-pair) for every unknown super-pixel by solving a graph transduction problem. It achieved state-of-the-art results on the date of submission[2].

3. A new pipeline for sampling-based alpha matting. It uses feature maps to guide the process of sample gathering, then it solves a graph labelling problem for pair assessment. The proposed pipeline is another variant of the methods that realize sequential pair selection, and it encodes half-pair constraints as smoothness constraints in the graph problem solved by multi-label graph cuts.

4. A framework for automatic trimap generation that adopts laws of visual perception for the grouping of matting components. It facilitates the incorporation of matting in the pipeline of IBR systems. This work was published in [34][3].

5. A single-image completion method that ameliorates two critical disadvantages in the existing techniques. Its performance is not a function of the hole size, and the

[1] http://www.site.uottawa.ca/%7Eaalka046/spie2014matting/index-spie14.html
[2] http://www.site.uottawa.ca/%7Eaalka046/TspsMatting/index.html
[3] http://www.site.uottawa.ca/%7Eaalka046/spie2015matting/index-spie15matting.html

procedure for nominating plausible filling patterns results in a limited number of shifts, which contributes to the quality of the results while taking the complexity of the optimization into consideration. Through its pipeline, the technique also features two approaches for constructing a skim of an image which can play a role in a variety of image/video analysis applications. This work was published in [35][4].

8.3 Future Research Directions

Throughout the presentation of thesis, I have pointed out several drawbacks that still need to be dealt with. A few suggested future directions include:

- For some 'conservative' trimaps, whether generated manually or using morphological dilations, one of the true half-pairs for some unknown pixels could be missing from the known regions. In other words, every telling foreground (or background) pixel, about a particular unknown pixel under consideration, is labelled unknown as well. This happens due to the usage of a thick grey brush or over-dilating the trimap. To calculate an accurate alpha value in such a case, the formulation of a better composition model is required. This model should allow an unknown pixel under consideration to express itself if it did not find a pair that yields a 'low-enough' chromatic distortion. The black ribbon of the 'plastic bag' image of the benchmark testing dataset is an example of such a case. A technique that capitalizes on *synthesizing* good samples is currently being explored.

- The formulation of an objective function that is more efficient than the chromatic distortion is also a possible future direction. Metric learning is a concept that has not been adopted before in the matting research, and has potential for best-pair nomination.

- The pipeline proposed in Chapter 5 requires improvements to achieve better time complexity. I am currently exploring the construction of graphs for only a part of the unknown pixels within a patch, rather than all of them. The alpha values of the excluded pixels can then be constructed from the alpha values of the pixels included

[4]http://www.site.uottawa.ca/%7Eaalka046/spie2015completion/index-spie15completion.html

in solving the graph. In other words, I am trying to reduce the number of graph sites, while retaining the quality of the computed alpha maps.

- Another topic of future research is an automatic trimap generator that builds upon the presented probabilistic approach in Chapter 6. That approach integrates the Gestalt laws with other low-level features to nominate the most probable components grouping. Particularly, I propose to quantify the certainty about the existence of mixed pixels in the matting components before grouping those 'mixed components' using laws of visual perception. To pin down directly the mixed pixels in an image, different sharpness [112] and defocus metrics are currently being explored. These metrics may be used in conjunction with structured decision forests which have recently been proven efficient in a closely-related application, realtime edge detection [85]. While the goal of the authors of [85] was to label each pixel with a binary label, whether it is an edge pixel or not, my goal is to label each pixel whether it is mixed or not.

- The parameters involved in the single image completion algorithm of Chapter 7 require further analysis with regards to their effect on the speed and the quality of completions.

- A scene completion algorithm that seeks a near-globally-optimal shift map, and that is temporally and stereoscopically consistent, is another point of future research. In section A.7, I present a basic framework for the extension of the proposed single image completion technique so that it can be adopted to complete a multi-view scene. This involves the formulation of a new energy function that benefits from temporal and depth information, in addition to the spatial information, to manage dis-occlusions in warped views.

Appendices

Appendix A

A.1 A Discussion on The Alpha Matting Benchmark

The matting online benchmark was first introduced in the research work published in [113]. The motivation behind it is to introduce a quantitate benchmark, similar to the well-known benchmarks that are available for other low-level vision problems. These benchmarks represent an online repository with which the researchers working in a particular research area can track the recent advances in that area, compare the performance of the SoA techniques, in addition to evaluating their own algorithms. It has augmented the high-quality dataset presented in [114] to construct a set of images that feature natural scenes, with a variety of focus settings, highly-textured backgrounds, color ambiguities, hard and soft boundaries, different boundary topologies, in addition to other types of transparencies, e.g., translucent objects. Even though its size is much smaller than the datasets available for other computer vision tasks, like the object segmentation dataset for example, it has been accepted by the matting research community as a standard dataset.

In addition to creating a standard and a challenging dataset, the benchmark also provides perceptually-motivated metrics for the quality of the computed alpha maps. These error functions were meant to complement the classical pixel-wise error functions (e.g., MSE and SAD) that do not always reflect the quality of the alpha maps as humans perceive it. A user study was carried out to validate the compliance of the developed perceptually-motivated metrics with human perception.

The data was captured in a restricted studio environment. Eight images, out of a total of thirty-five images, featured an object shot against a natural scene that that was built by the benchmark developers. In the other twenty-seven images in the dataset, the

foreground object was shot against a screen displaying a natural scene. To obtain high-quality ground truth alpha mattes, the foreground objects were captured against a screen displaying a single-color background, and each object was shot against four different colors (black, red, green and blue). After capturing the foreground objects against those single-color backgrounds, the objects were removed and the backgrounds were photographed. The ground truth alpha mattes were then calculated using the triangulation technique which will be discussed in the next section. The developers of the benchmark also provided information in [113] about the specifications of the camera used in the experiment in addition to other precautions taken to avoid camera shakes during capture; such details were considered beyond the scope of this thesis.

The user input in the benchmark was simulated by three types of trimaps. The first and the second types are generated automatically by the morphological dilation of the unknown regions in the ground truth alpha mattes by 22 and 44 pixels respectively. The third type of trimaps represents a more natural user input and it was created for the testing images only in the dataset, i.e., type 3 trimaps is not available for the training images. To generate that type of trimaps, an experienced user was provided a paint tool with three brushes (foreground, background, unknown) and a flood filling functionality; the user was asked to generate the trimap within two minutes (for each image).

The matting algorithms proposed in [115, 70, 116, 117, 37, 118] were run over the thirty-five images in the dataset, and the MSE rates were calculated for all of them. The dataset was then categorized into four categories according to the amount of transparencies in their ground truth alpha mattes. The two most challenging images from every category were then selected to construct the set of eight testing images. An image 'x' is considered more challenging than another image 'y' if the average error from all the aforementioned algorithms on 'x' is larger than 'y' and if the quality of the results are more diverging on 'x' than 'y'.

In their pursuit to perceptually-motivated error functions, the developers of the benchmark focused on two specific error categories which seemed to lower the visual quality of image composites considerably. The first category is the connectivity errors which appear in the alpha map as disconnected foreground objects. The second category is the gradient errors which result from either over-smoothing discontinuities or introducing discontinu-

ities erroneously in the alpha map; in both cases, the gradient of the original image is not similar to that of the alpha map.

In order to make sure that the designed metrics for gradient and connectivity errors comply with human perception, psychological experiments were carried out on two sets of image compositions; each set suffers solely from either gradient errors or connectivity errors. To construct such sets, a variety of matting algorithms were run on the input images in the dataset, and crops from the computed alpha mattes were chosen to represent each of these error categories. The participants were then asked to rank those crops. More specific details on the number of participants, their age and gender, and the study procedure can be found in [113].

The metric used for quantifying the gradient error is given by:

$$G_e = \sum_i (\nabla\alpha_i - \nabla\alpha_i^*)^q \tag{A.1}$$

where G_e is the gradient error, $\nabla\alpha_i$ and $\nabla\alpha_i^*$ are the normalized gradients of the computed alpha matte and the ground truth at pixel i respectively. The normalized gradients are computed using a first-order Gaussian derivative filters with variance σ. The metric used for quantifying the connectivity error is given by:

$$G_c = \sum_i (\varphi(\alpha_i, \Omega) - \varphi(\alpha_i^*, \Omega))^p \tag{A.2a}$$

$$\varphi(\alpha_i, \Omega) = 1 - (\delta(d_i \geq \theta) \cdot d_i) \tag{A.2b}$$

where G_c is the connectivity error, φ measures the connectivity of the pixel i to the source region Ω, d_i is the difference between the alpha value of pixel i and the maximum threshold level l_i at which pixel i is 4-connected to Ω. The source region Ω is defined by the largest connected region where the alpha matte and its ground truth are completely opaque. The pixel i is fully connected if $\varphi = 1$, i.e., $\alpha_i = l_i$. The purpose of δ is to neglect any variations in d_i that are smaller than θ.

The parameters σ, q, θ and p are chosen to maximize the correlation between the aforementioned metrics and the 'average ranking' of the participants. This section of the appendix is meant to discuss briefly the benchmark details that are most pertinent to

the scope of the thesis. It explained: the structure of the dataset, how the ground truth maps were obtained, how the trimaps were generated, and finally what the perceptually motivated error metrics are. A comprehensive discussion can be found in [113].

A.2 A Discussion on Calculating Ground-truth Alpha Maps Using Triangulation

In section 2.1, the mathematical modelling of natural image matting was presented and it was shown that the problem is incompletely specified, and thus infinite number of solutions exist unless more information is provided. The authors of [2] presented three special cases where a solution of the matting problem can be found. These cases add some constraints on the general problem to prune the space of possible solutions. One of these special cases is when the un-composited foreground color $C_o \coloneqq \{R_o, G_o, B_o\}$ is known against two different shades of the backing color $C_k \coloneqq \{R_k, G_k, B_k\}$, which makes the problem solvable by triangulation.

The two shades will be given the symbols B_{K_1} and B_{K_2} respectively. Without loss of generalization, these colors can be taken to be the blue and the black; thus $B_{K_1} = cB_k$ and $B_{K_2} = dB_k$, where $d = 0$ and $d < c \leq 1$. The special case of knowing C_o against the two shades of B_k can be written formally as:

$$C_{f_1} \coloneqq \begin{bmatrix} R_o & G_o & \alpha \times B_o + (1-\alpha) \times B_{k_1} \end{bmatrix} \tag{A.3}$$

$$C_{f_2} \coloneqq \begin{bmatrix} R_o & G_o & \alpha \times B_o + (1-\alpha) \times B_{k_2} \end{bmatrix}. \tag{A.4}$$

By combining the expressions of the blue channel of the first foreground B_{f_1} and the second foreground B_{f_2}, the value of the alpha can be calculated as:

$$\alpha = 1 - \frac{B_{f_1} - B_{f_2}}{B_{k_1} - B_{k_2}}, \tag{A.5}$$

where the denominator can not be equal to zero because B_{k_1} and B_{k_2} are different. The solution of the matting problem is then completed by computing the components of the

un-composited foreground color C_o as:

$$\left[R_o = R_{f_1} = R_{f_2} \qquad G_o = G_{f_1} = G_{f_2} \qquad B_o = \frac{B_{f_2}B_{k_1} - B_{f_1}B_{k_2}}{B_{k_1} - B_{k_2}}\right]. \tag{A.6}$$

The aforementioned algorithmic procedure (triangulation) was used to compute the ground truth alpha maps provided by the online matting benchmark. Recently, the ground truth foreground colors (C_o) of the benchmark's training dataset were made available on the benchmark's webpage as well.

A.3 Results of The Transductive Sequential Pair Selection Matting on The Testing Dataset of The Matting Benchmark

Figure A.1 shows the results of the matting technique discussed in sub-section 4.3 on the testing dataset of [3].

A.4 Results of The Transductive Sequential Pair Selection Matting With The Extended Cost Function on The Testing Dataset of The Matting Benchmark

Figure A.2 shows the results of the matting technique discussed in sub-section 4.3 on the testing dataset of [3]. In the previous section, the alpha maps were calculated by assessing the FG/BG pairs using the chromatic distortion only. This section shows the results obtained by adopting an objective function that involves the chromatic distortion terms, and in addition promotes sparsity in the alpha maps and encourages the selection of FG/BG pairs where the FG half-pair and the BG half-pair come from well-separated color distributions.

A.5 Results of The Sequential Pair Selection Matting on The Testing Dataset of The Matting Benchmark

Figure A.3 shows the results of the matting technique discussed in sub-section 4.2 on the testing dataset of [3].

A.6 Results of GHC Matting on The Testing Dataset of The Matting Benchmark

Figure A.4 shows the results of the matting technique discussed in Chapter 5 on the testing dataset of [3].

A.7 Graph-based depth-guided scene completion

This part of my research is still at its infancy, even though its main building block (single image completion) has been already implemented. As a quick recall, the holes we get in single image completion are usually due to image editing operations, such as object removal and/or object re-arrangement; on the other hand, the holes we are dealing with in scene completion are mainly due to dis-occlusion. My proposed technique for single image completion lends itself well to that more general context of scene completion. As will be shown shortly, the same energy function I have used, to calculate an optimal shiftmap, can be extended to involve the additional terms that guarantee stereoscopic and temporal consistency. ***This extension is expected to be the first stereo-temporal near-globally-optimal scene completion method in the literature***. In the following few lines, I will present a high-level abstract of that extension.

My prospective method is expected to follow the classical pipeline that starts by filling the warped depth map, associated with the warped frame, and then use it to accomplish the texture synthesis step in the color frame. This depth filling step may also be done simultaneously with the color synthesis; actually, this may be more favourable according to my framework, since both of them are completion steps that seek the same

shiftmap. By checking the variety of datasets available in the literature, I could figure out some characteristics of the problem of scene completion which, if properly exploited, can result in a multi-fold advantage of my method over the state-of-the-art. For example, it usually suffices to limit the known region so that, instead of the whole frame, only the background region that is less than a particular distance from the hole is considered. This is expected to result in considerable speedups in the skimming and the multi-label graph cuts steps. This becomes very tangible if compared with iterative and greedy approaches. Moreover, since the known region is not the whole frame, the number of dominant slopes are not expected to be large; consequently, the most computationally-demanding step of patch-slope nomination can be discarded, with negligible compromise in the quality of completions. My energy function, which is similar to Eqn. 7.8, has a data term that is expressed as

$$E_D = \sum_{p_i \in \Omega} E_d(L(p_i)) + \sum_{p_i \in \Omega} E_d(D(p_i)), \tag{A.7}$$

where the first term penalizes the invalid shifts while the second term penalizes the shifts that violate the weak consistency check; this check was mentioned for the first time in Chapter 3 and is given by:

$$|I_L(x,y) - I_R(x - D_l(x,y), y)| < \epsilon \tag{A.8a}$$

$$|I_R(x,y) - I_L(x + D_R(x,y), y)| < \epsilon. \tag{A.8b}$$

The smoothness term in the energy function can be expressed as:

$$\begin{aligned} E_s = & \sum_{(p_i,p_j)\,\forall\, p_i \in \Omega, p_j \in \Omega} E_s(L(p_i), L(p_j)) + E_s(D(p_i), D(p_j)) \\ & + \sum_{(p_i, p_i^{V_n})\,\forall\, p_i \in \Omega, p_i^{V_n} \in V_n} E_s(L(p_i), p_i^{V_n}) + \sum_{(p_i, p_i^{t+1})\,\forall\, p_i \in \Omega, p_i^{t+1} \in f^{t+1}} E_s(L(p_i), p_i^{t+1}), \end{aligned} \tag{A.9}$$

where the first and the second terms ensure coherent seams in the warped frame and the warped depth map, while $p_i^{V_n}$ is the corresponding value of p_i in the view V_n and this term is meant to ensure stereoscopic consistency. Finally, p_i^{t+1} is the corresponding value of p_i in the succeeding frame f^{t+1} and this term is meant to ensure temporal consistency.

A.8 A Brief Presentation on The Convergence and Optimality Properties of Graph Cuts

For the convenience of the reader, I present in this section a brief discussion on the convergence properties of graph cuts, the multi-label optimization tool adopted in my research. It is worth mentioning that I was concerned with graph cuts merely as a tool, and the thesis does not contribute to the efficiency of graph cuts, rather, it benefits from it. The specific interest in graph cuts stems from the following two reasons. First, the graph cuts-based algorithms are known to be fast, and in many vision problems, their running time is lower than what can be expected from their computational complexity [64]. The second reason is the adherence of my problems to the type of energies that graph cuts can minimize, e.g., pixels are assigned one label from a discrete set of labels, and the neighbouring pixels are expected to favour similar labels which complies with the sub-modularity condition that should be satisfied by the pair-wise terms, (please check Chapter 2). Since my research does not contribute to that optimization tool, and did not involve convergence analysis of it, the following discussion will be comprised of a summary on the convergence and optimality properties of graph cuts as presented in [90].

The authors of [90] proposed two algorithms, based on graph cuts, that approximately minimize energies for which the computation of a global minimum is an NP-hard problem. Those energies are of a type that could surface in a variety of contexts, one of which is the Bayesian labelling of Markov random fields. The form of such energies was given in equation 2.15, and is reproduced here for the readers convenience as follows:

$$E(f) = \sum_{p \in P} D_p(f_p) + \sum_{(p,q) \in \mathcal{N}} V_{pq}(f_p, f_q), \tag{A.10}$$

where $D_p(f_p)$ is the cost of assigning the label f_p to pixel p, $\mathcal{N}$ is the set of interacting pairs of pixels and $V_{pq}(f_p, f_q)$ is the cost of assigning the labels f_p and f_q to the interacting (neighbouring) pixels p and q in a certain neighbourhood system (4 or 8-connected). The set of labels assigned to all the pixels constitutes what is called a *labelling*, which is given the symbol f, e.g., the optimal labelling that results in reaching the global minimum of E is referred to as f^*.

Even though the two proposed algorithms in [90] compute a local minimum for $E(f)$,

their superiority, when compared to the predecessor methods in the literature, stems from the type and the size of 'moves' they allow. The term 'move' in the context of discrete labelling problems is used to indicate how far two labellings are, e.g., a labelling f_1 is said to be near to another labelling f_2 if it lies within a single move of it. While the majority of local optimization problems allowed standard moves, where the label of just one pixel is allowed to change at a time, the algorithms proposed in [90] find a local minimum for E with respect to two very large moves, namely the $\alpha - \beta$-swap move and the α-expansion move, where the labels of arbitrarily large number of pixels are allowed to change at a time. These large moves help avoiding the problems of local minima and the low quality solutions associated with standard moves.

The $\alpha - \beta$-swap move can handle a wider class of smoothness penalties V_{pq} than the α-expansion move (Vs should be a metric distance function for expansion moves). However, it does not have the guaranteed optimality properties of the local minimum found when α-expansion moves are allowed. Hence, the rest of my presentation, for the exact definition of the move, as well as the optimality properties will be referring to that of the α-expansion moves. In the following paragraph, I explain what this type of move means.

There is a one-to-one correspondence between any labelling f and a partition $\mathbf{P}$ of image pixels, where $\mathbf{P} = \{\mathcal{P}_l | l \in \mathcal{L}\}$, $\mathcal{L}$ is the set of discrete labels, and $\mathcal{P}_l = \{p \in \mathcal{P} | f_p = l\}$ is the subset of image pixels which were assigned the label l. To make an α-expansion move from a partition $\mathbf{P}^1$ to another partition $\mathbf{P}^2$, which is equivalent to move from a labelling f^1 to the labelling f^2, these conditions should be satisfoed: $\mathbf{P}^1_\alpha \subset \mathbf{P}^2_\alpha$ and $\mathbf{P}^2_l \subset \mathbf{P}^1_l$, for any label $l \neq \alpha$. Stated in other words, during an expansion move, any set of pixels is allowed to alter their labels to α. The pseudocode of the α-expansion algorithm is summarized as follows:

In the terminology of [90], the single execution of the steps 5 through 8 is called an *iteration*, and the steps from 3 through 11 is called a *cycle*. The above procedure shows that the algorithm terminates after the first 'unsuccessful cycle', a cycle in which no further improvement can be achieved, where a single cycle takes $\mathcal{L}$ iterations. The α-expansion algorithm is guaranteed to terminate [90] in $O(|\mathcal{P}|)$ cycles, if the data and the smoothness penalties in Eqn. A.10 are independent of the image size, which is a valid assumption in

Algorithm 3

```
1: procedure GetLabelling(Image, Labels)
2:     Start with an arbitrary labelling f
3:     Set success := 0
4:     for every α ∈ L do
5:         Find f̂ = argmin E(f') among f' within one α-expansion of f
6:         if E(f̂) < E(f) then
7:             set  f := f̂  and sucess := 1
8:         end if
9:         if success = 1 then
10:            goto 3
11:        end if
12:    end for
13:    Return f
14: end procedure
```

practice. Nevertheless, the reported experiments showed that convergence happens within a few cycles, with the first iteration leading to the largest part (99% in some cases) of the improvement, starting from an arbitrary labelling f. The locally optimal labellings generated with respect to α-expansion moves were also proved to lie within a known factor of the global minimum, where this factor is the ratio of the maximum value of the smoothness cost to the minimum value of it.

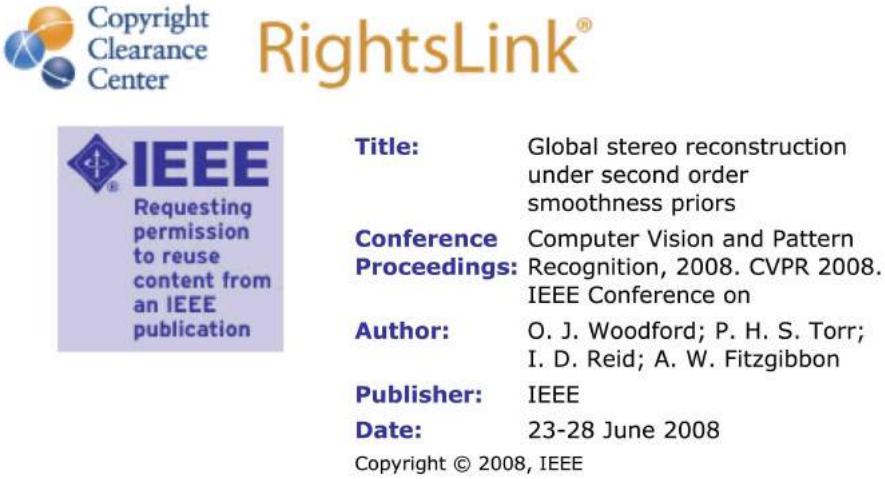
A.9 Permission Grant of IEEE Copyrighted Material

Figure A.1: Results of the transductive sequential pair selection matting algorithm on the testing dataset of [3]. The 2^{nd}, 3^{rd} and 4^{th} columns correspond to the 1^{st}, 2^{nd} and 3^{rd} trimaps respectively.

Figure A.2: Results of the transductive sequential pair selection matting algorithm with the extended cost function on the testing dataset of [3]. The 2^{nd}, 3^{rd} and 4^{th} columns correspond to the 1^{st}, 2^{nd} and 3^{rd} trimaps respectively.

Figure A.3: Results of the sequential pair selection matting algorithm on the testing dataset of [3]. The 2nd, 3rd and 4th columns correspond to the 1st, 2nd and 3rd trimaps respectively.

Figure A.4: Results of the GHC matting algorithm on the testing dataset of [3]. The 2^{nd}, 3^{rd} and 4^{th} columns correspond to the 1^{st}, 2^{nd} and 3^{rd} trimaps respectively.



Copyright Clearance Center RightsLink®

IEEE Requesting permission to reuse content from an IEEE publication

Title:	Global stereo reconstruction under second order smoothness priors
Conference Proceedings:	Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on
Author:	O. J. Woodford; P. H. S. Torr; I. D. Reid; A. W. Fitzgibbon
Publisher:	IEEE
Date:	23-28 June 2008

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Figure A.5: Permission Grant of IEEE Copyrighted Material.

Bibliography

[1] M. Solh and G. AlRegib, "Hierarchical hole-filling for depth-based view synthesis in FTV and 3D video," *IEEE Journal of Selected Topics in Signal Processing*, vol. 6, no. 5, pp. 495 – 504, 2012.

[2] A. R. Smith and J. F. Blinn, "Blue screen matting," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '96. ACM, 1996, pp. 259–268.

[3] Alpha matting online benchmark. [Online]. Available: http://alphamatting.com

[4] O. Woodford, P. Torr, I. Reid, and A. Fitzgibbon, "Global stereo reconstruction under second-order smoothness priors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 2115 – 2128, 2009.

[5] L. Wang, T. Xia, Y. Guo, L. Liu, and J. Wang, "Confidence-driven image co-matting," *Computers and Graphics*, vol. 38, pp. 131 – 139, 2014.

[6] B. Morse, J. Howard, S. Cohen, and B. Price, "Patchmatch-based content completion of stereo image pairs," in *Second International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, 2012.

[7] E. S. L. Gastal and M. M. Oliveira, "Shared sampling for real-time alpha matting," *Computer Graphics Forum*, vol. 29, no. 2, pp. 575–584, 2010.

[8] K. He, C. Rhemann, C. Rother, X. Tang, and J. Sun, "A global sampling method for alpha matting," in *CVPR*, 2011.

[9] J. Wang and M. Cohen, "Image and video matting: A survey," *Foundations and Trends in Computer Graphics and Vision*, vol. 3, no. 2, pp. 97 – 175, 2007.

[10] C. Rhemann, C. Rother, and M. Gelautz, "Improving color modeling for alpha matting," in *BMVC*, 2008.

[11] A. Al-Kabbany and E. Dubois, "Improved global-sampling matting using sequential pair-selection strategy," in *Visual Information Processing and Communication V*, 2014.

[12] E. Shahrian and D. Rajan, "Weighted color and texture sample selection for image matting," in *CVPR*, 2012.

[13] E. Varnousfaderani and D. Rajan, "Weighted color and texture sample selection for image matting," *IEEE Transactions on Image Processing*, vol. 22, no. 11, pp. 4260 – 4270, 2013.

[14] D. Li, Q. Chen, and C.-K. Tang, "Motion-aware KNN Laplacian for video matting," in *ICCV*, 2013.

[15] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *the 27th Annual Conference on Computer Graphics and Interactive Techniques*, 2000.

[16] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera, "Filling-in by joint interpolation of vector fields and gray levels," *IEEE Transactions on Image Processing*, vol. 10, no. 8, pp. 1200–1211, 2001.

[17] A. Levin, A. Zomet, and Y. Weiss, "Learning how to inpaint from global image statistics," in *ICCV*, 2003.

[18] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, "Simultaneous structure and texture image inpainting," in *CVPR*, 2003.

[19] S. Roth and M. Black, "Fields of experts: a framework for learning image priors," in *CVPR*, 2005.

[20] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling," in *ICCV*, 1999.

[21] A. Criminisi, P. Perez, and K. Toyama, "Object removal by exemplar-based inpainting," in *CVPR*, 2003.

[22] J. Jia and C.-K. Tang, "Image repairing: robust image synthesis by adaptive ND tensor voting," in *CVPR*, 2003.

[23] Y. Wexler, E. Shechtman, and M. Irani, "Space-time completion of video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 463 – 476, 2007.

[24] ——, "Space-time video completion," in *CVPR*, 2004.

[25] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "PatchMatch: A randomized correspondence algorithm for structural image editing," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 28, no. 3, 2009.

[26] K. He and J. Sun, "Statistics of patch offsets for image completion," in *ECCV*, 2012.

[27] N. Komodakis, "Image completion using global optimization," in *CVPR*, 2006.

[28] Y. Pritch, E. Kav-Venaki, and S. Peleg, "Shift-map image editing," in *ICCV*, 2009.

[29] N. Komodakis and G. Tziritas, "Image completion using efficient belief propagation via priority scheduling and dynamic pruning," *IEEE Transactions on Image Processing*, vol. 16, no. 11, pp. 2649 – 2661, 2007.

[30] A. Al-Kabbany and E. Dubois. Image completion using image skimming. [Online]. Available: http://www.site.uottawa.ca/%7Eaalka046/spie2015completion/index-spie15completion.html

[31] L. Zhang, W. J. Tam, and D. Wang, "Stereoscopic image generation based on depth images," in *ICIP*, 2004.

[32] L. Wang, H. Jin, R. Yang, and M. Gong, "Stereoscopic inpainting: Joint color and depth completion from stereo images," in *CVPR*, 2008.

[33] T.-J. Mu, J.-H. Wang, S.-P. Du, and S.-M. Hu, "Stereoscopic image completion and depth recovery," *The Visual Computer*, vol. 30, no. 6 - 8, pp. 833 – 843, 2014.

[34] A. Al-Kabbany and E. Dubois, "A novel framework for automatic trimap generation using the gestalt laws of grouping," in *Visual Information Processing and Communication VI*, 2015.

[35] ——, "Image completion using image skimming," in *Visual Information Processing and Communication VI*, 2015.

[36] J. Wang and M. Cohen, "Optimized color sampling for robust matting," in *CVPR*, 2007.

[37] Y.-Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski, "A Bayesian approach to digital matting," in *CVPR*, 2001.

[38] I. Ahn and C. Kim, "A novel depth-based virtual view synthesis method for free viewpoint video," *IEEE Transactions on Broadcasting*, vol. 59, no. 4, pp. 614–626, 2013.

[39] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen, "Interactive digital photomontage," in *ACM SIGGRAPH*, 2004.

[40] Y. Lu, W. Zhang, H. Lu, and X. Xue, "Salient object detection using concavity context," in *ICCV*, 2011.

[41] L. Gorelick, O. Veksler, Y. Boykov, and C. Nieuwenhuis, "Convexity shape prior for segmentation," in *ECCV*, 2014.

[42] W. Metzger, *Laws of Seeing*. MIT Press, August 2009.

[43] W. Ellis and K. Koffka, *A Source Book of Gestalt Psychology*. Gestalt Journal Pr., 1983.

[44] A. Buades, T. Le, J.-M. Morel, and L. Vese. Fast cartoon+texture decompsotion project webpage. [Online]. Available: http://www.ipol.im/pub/art/2011/blmv_ct/

[45] Y. Meyer, *Oscillating Patterns in Image Processing and Nonlinear Evolution Equations: The Fifteenth Dean Jacqueline B. Lewis Memorial Lectures*. Boston, MA, USA: American Mathematical Society, 2001.

[46] A. Buades, T. Le, J.-M. Morel, and L. Vese, "Cartoon+Texture Image Decomposition," *Image Processing On Line*, vol. 1, 2011.

[47] S. Osher, A. Sole, and L. Vese, "Image decomposition and restoration using total variation minimization and the Hˆ-1 norm," *Simul*, vol. 1, pp. 349–370, 2002.

[48] J. J. Shen, "Piecewise H 1 + H 0 + H 1 images and the Mumford-Shah-Sobolev model for segmented image decomposition," *APPL. MATH. RES. EXP*, vol. 4, pp. 143–167, 2005.

[49] A. L. Dulmage and N. S. Mendelsohn, "Coverings of bipartite graphs," *Canadian Journal of Mathematics*, vol. 10, pp. 517–534, 1958.

[50] Y. Boykov and O. Veksler, "Graph cuts in vision and graphics: Theories and applications," in *Handbook of Mathematical Models in Computer Vision*, N. Paragios, Y. Chen, and O. Faugeras, Eds. Springer, 2006.

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

[52] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 1124 – 1137, 2004.

[53] V. Kolmogorov. Vladimir kolmogorov's webpage. [Online]. Available: http://pub.ist.ac.at/~vnk/

[54] D. M. Greig, B. T. Porteous, and A. H. Seheult, "Exact maximum a posteriori estimation for binary images," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 51, pp. 271–279, 1989.

[55] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?" *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 147 – 159, 2004.

[56] L. Gorelick, Y. Boykov, O. Veksler, I. B. Ayed, and A. Delong, "Submodularization for binary pairwise energies," in *CVPR*, 2014.

[57] V. Kolmogorov and C. Rother, "Minimizing nonsubmodular functions with graph cuts-a review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 7, pp. 1274 – 1279, 2007.

[58] L. van der Maaten, E. O. Postma, and H. J. van den Herik, "Dimensionality reduction: A comparative review," 2008.

[59] Z. Farbman, R. Fattal, and D. Lischinski, "Diffusion maps for edge-aware image editing," *ACM Trans. Graph.*, vol. 29, no. 6, pp. 145:1 – 145:10, 2010.

[60] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *SCIENCE*, vol. 290, pp. 2323–2326, 2000.

[61] Y. Freund and R. E. Schapire, "A short introduction to boosting," 1999.

[62] A. Gammerman, V. Vovk, and V. Vapnik, "Learning by transduction," *CoRR*, vol. abs/1301.7375, 2013. [Online]. Available: http://arxiv.org/abs/1301.7375

[63] M. Belkin and P. Niyogi, "Semi-supervised learning on Riemannian manifolds," *Machine Learning*, vol. 56, no. 1-3, pp. 209 – 239, 2004.

[64] O. Duchenne, J.-Y. Audibert, R. Keriven, J. Ponce, and F. Segonne, "Segmentation by transduction," in *CVPR*, 2008.

[65] M. Hein, J.-Y. Audibert, and U. von Luxburg, "From graphs to manifolds-weak and strong pointwise consistency of graph laplacians," in *the 18th Annual Conference on Learning Theory*, 2005.

[66] J. Wang, "Image matting with transductive inference," in *Computer Vision/Computer Graphics Collaboration Techniques*, 2011, vol. 6930, pp. 239–250.

[67] D. Coppi, S. Calderara, and R. Cucchiara, "Appearance tracking by transduction in surveillance scenarios," in *Advanced Video and Signal-Based Surveillance (AVSS), 2011 8th IEEE International Conference on*, vol. 1, Aug 2011, pp. 142–147.

[68] R. Li and Y. Saad, "GPU-accelerated preconditioned iterative linear solvers," *The Journal of Supercomputing*, vol. 63, no. 2, pp. 443 – 466, 2013.

[69] J. M. Bahi, R. Couturier, and L. Z. Khodja, "Parallel GMRES implementation for solving sparse linear systems on GPU clusters," in *in the 19th High Performance Computing Symposia*, 2011.

[70] A. Levin, A. Rav-Acha, and D. Lischinski, "Spectral matting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, pp. 1699 – 1712, 2008.

[71] Q. Chen, D. Li, and C.-K. Tang, "KNN matting," in *CVPR*, 2012.

[72] X. Chen, D. Zou, S. Zhou, Q. Zhao, and P. Tan, "Image matting with local and nonlocal smooth priors," in *CVPR*, 2013.

[73] E. Shahrian, D. Rajan, B. Price, and S. Cohen, "Improving image matting using comprehensive sampling sets," in *CVPR*, 2013.

[74] E. Shahrian, B. Price, S. Cohen, and D. Rajan, "Temporally coherent and spatially accurate video matting," *Computer Graphics Forum*, vol. 33, pp. 381 – 390, 2014.

[75] A. Levin, D. Lischinski, and Y. Weiss, "A closed-form solution to natural image matting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 228–242, 2008.

[76] K. He, J. Sun, and X. Tang, "Guided image filtering," in *ECCV*, 2010, pp. 1–14.

[77] A benchmark for change detection. [Online]. Available: http://changedetection.net/

[78] L. Zhang and W. J. Tam, "Stereoscopic image generation based on depth images for 3D TV," *IEEE Transactions on Broadcasting*, vol. 51, no. 2, pp. 191 – 199, 2005.

[79] I. Daribo and H. Saito, "A novel inpainting-based layered depth video for 3D TV," *IEEE Transactions on Broadcasting*, vol. 57, no. 2, pp. 533–541, 2011.

[80] J. Gautier, O. Le Meur, and C. Guillemot, "Depth-based image completion for view synthesis," in *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, 2011.

[81] S. Choi, B. Ham, and K. Sohn, "Space-time hole filling with random walks in view extrapolation for 3D video," *IEEE Transactions on Image Processing*, vol. 22, pp. 2429 – 2441, 2013.

[82] L. Grady, "Random walks for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 11, pp. 1768 – 1783, 2006.

[83] M. Gong and Y.-H. Yang, "Fast stereo matching using reliability-based dynamic programming and consistency constraints," in *ICCV*, 2003.

[84] S. Luo, Y. Sun, I. Shen, B. Chen, and Y. Chuang, "Geometrically consistent stereoscopic image editing using patch-based synthesis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 21, no. 1, pp. 56–67, 2015.

[85] P. Dollár and C. L. Zitnick, "Structured forests for fast edge detection," in *ICCV*, 2013.

[86] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Su, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274 – 2282, 2012.

[87] VLFeat library. [Online]. Available: http://www.vlfeat.org/index.html

[88] D. Comaniciu, P. Meer, and S. Member, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)*, vol. 24, pp. 603–619, 2002.

[89] Q. Chen, D. Li, and C.-K. Tang, "KNN matting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 9, pp. 2175 – 2188, 2013.

[90] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 1222 – 1239, 2001.

[91] O. Veksler and Y. Boykov. The code page of the computer vision group at the University of Western Ontario. [Online]. Available: http://vision.csd.uwo.ca/code/

[92] X. Bai, J. Wang, and D. Simons, "Towards temporally-coherent video matting," in *Proceedings of the 5th International Conference on Computer Vision/Computer Graphics Collaboration Techniques*, vol. 1. Springer-Verlag, 2011, pp. 63–74.

[93] Video matting benchmark. [Online]. Available: http://videomatting.com/

[94] J. Carreira and C. Sminchisescu, "Constrained parametric min-cuts for automatic object segmentation," *CVPR*, 2010.

[95] C.-C. Kao, J.-H. Lai, and S.-Y. Chien, "Automatic object segmentation with salient color model," in *2011 IEEE International Conference on Multimedia and Expo*, July 2011, pp. 1–6.

[96] X. Bai and W. Wang, "Saliency-svm: An automatic approach for image segmentation," *Neurocomputing*, vol. 136, pp. 243 – 255, 2014.

[97] J. Kim, J. Park, and K. Park, "Unamt: Unsupervised adaptive matting tool for large-scale object collections," in *ACM SIGGRAPH 2015 Posters*, ser. SIGGRAPH '15, vol. 56. ACM, 2015, pp. 1–1.

[98] W. Sun, S. Luo, and L. Wu, "A biologically-inspired automatic matting method based on visual attention," in *7th International Symposium on Neural Networks*, L. Zhang, B.-L. Lu, and J. Kwok, Eds., vol. 2. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 170–177.

[99] X. Bai, J. Wang, D. Simons, and G. Sapiro, "Video snapcut: Robust video object cutout using localized classifiers," in *ACM SIGGRAPH*, 2009.

[100] I. Choi, M. Lee, and Y.-W. Tai, "Video matting using multi-frame nonlocal matting laplacian," in *ECCV*, 2012.

[101] S. Singh, A. Jalal, and C. Bhatanagar, "Automatic trimap and alpha-matte generation for digital image matting," in *Sixth International Conference on Contemporary Computing*, 2013.

[102] K. Lee, "Learning-based trimap generation for video matting," Master's thesis, University of California, San Diego, 2010.

[103] O. Wang, J. Finger, Q. Yang, J. Davis, and R. Yang, "Automatic natural video matting with depth," in *15th Pacific Conference on Computer Graphics and Applications*, 2007.

[104] A. Levin, R. Alex, and D. Lischinski. Spectral matting project page. [Online]. Available: http://www.vision.huji.ac.il/SpectralMatting/

[105] B. Alexe, T. Deselares, and V. Ferrari, "What is an object?" in *CVPR*, 2010.

[106] Y. Chuang, A. Agarwala, B. Curless, D. Salesin, and R. Szeliski. Video matting of complex scenes. [Online]. Available: http://grail.cs.washington.edu/projects/digital-matting/video-matting/

[107] I. Simon, N. Snavely, and S. M. Seitz, "Scene summarization for online image collections," in *ICCV*, 2007.

[108] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool, "Creating summaries from user videos," in *ECCV*, 2014.

[109] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani, "Summarizing visual data using bidirectional similarity," in *CVPR*, 2008.

[110] K. He and J. Sun, "Image completion approaches using the statistics of similar patches," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, pp. 2423 – 2435, 2014.

[111] ——. Statistics of patch offsets for image completion-project webpage. [Online]. Available: http://research.microsoft.com/en-us/um/people/kahe/eccv12/

[112] R. Ferzli and L. Karam, "A no-reference objective image sharpness metric based on the notion of just noticeable blur (jnb)," *IEEE Transactions on Image Processing*, vol. 18, no. 4, pp. 717–728, 2009.

[113] C. Rhemann, C. Rother, J. Wang, M. Gelautz, P. Kohli, and P. Rott, "A perceptually motivated online benchmark for image matting," in *CVPR*, 2009.

[114] C. Rhemann, C. Rother, A. Rav-Acha, and T. Sharp, "High resolution matting via interactive trimap segmentation," in *CVPR*, 2008.

[115] J. Wang and M. Cohen, "Optimized color sampling for robust matting," in *CVPR*, 2007.

[116] Y. Guan, W. Chen, X. Liang, and Q. Peng, "Easy matting - a stroke based approach for continuous image matting," *Computer Graphics Forum*, vol. 25, pp. 567–576, 2006.

[117] J. Sun, J. Jia, C.-K. Tang, and H.-Y. Shum, "Poisson matting," in *ACM SIGGRAPH 2004 Papers*, ser. SIGGRAPH '04, vol. 1, 2004, pp. 315–321.

[118] L. Grady, T. Schiwietz, S. Aharon, and R. Westermann, "Random walks for interactive alpha-matting," in *VIIP 2005*, 2005.