SCENE COMPLETION USING TOP-1 SIMILAR IMAGE

BY

ROMANA TALAT

01-243141-008

SUPERVISED BY

DR. MUHAMMAD MUZAMMAL

Session-2014-16

A Report submitted to the Department of Computer Science

Bahria University, Islamabad

in partial fulfilment of the requirement for the degree of MS(CS)

CERTIFICATE

We accept the work contained in this report as a confirmation to the required standard for the partial fulfilment of the degree of MS (CS).

Head of Department

Supervisor

Internal Examiner

External Examiner

DEDICATION

To My Parents

DECLARATION OF AUTHORSHIP

I hereby certify that I am sole author of this thesis and no part of this thesis has been published. To the best of my knowledge, my thesis does not disobey anyone’s copyrights and the ideas, techniques and other material from the work of other people included in my thesis are fully acknowledge in accordance with standard referencing practice.

I declare that this is true copy of my thesis, including final revisions approved by panel and this thesis has not been submitted for higher degree to any other universities.

ACKNOWLEDGEMENTS

First of all, I am highly indebted to my supervisor Dr. Muhammad Muzammal for his support, encouragement and guidance at every step that enabled me to develop an understanding of the subject. It is really fortunate that I got such an opportunity to learn ABC of research which will definitely go a long way in my career. I would always remember the lengthy and useful discussions we have had about the thesis. I have gained a lot from the constructive suggestions made by Dr. Muzammal, who taught me the art of presenting my work. I will always be thankful to him for his enthusiastic guidance and support which made this dissertation possible.

I am equally indebted to Bahria University Islamabad campus to encourage me to pursue my research work. In particular, I would like to thank Dr. Imran Siddiqi for many useful suggestions in this work. I appreciate the cooperation and support from all my friends, colleagues and teacher especially Prof. Dr. Tasneem Shah.

I dedicate this thesis to my family. I have achieved this because of my Mother's prayer and my Father's support. My sister Ms. Nazia Talat has always supported and encouraged me and is always an inspiration.

Above all, I pay my reverence to the Almighty ALLAH.

Romana Talat

ABSTRACT

Scene Completion is an interesting Image Processing problem that has recently been studied in the context of data, i.e. by using large repositories of data. The idea is to extract the most suitable image from a data repository that could later be used in constructing the missing part of the input image. One of the requirements for such a data intensive approach is that the completion has to be completed without human intervention. This is rather challenging as it may not be clear that what could be the most suitable image in the data repository for the completion purpose. Thus, whilst it is important that the repository should be of reasonable size so that a matching image could be found, it is even more important that we are able to retrieve that top-1 image that could be the best candidate for scene completion in an automated fashion. This is the focus of this work, i.e. we propose a methodology for finding the top-1 image in a data repository that could be the best candidate for scene completion. Then we give an algorithm for scene completion that works as follows. In the first step, matching images with the query image are retrieved that could be the candidates for the completion. We consider three features for the purpose namely Gist, Texture and Colour. Gist feature represents the shape and structure of the image. Texture helps when segments from different images are being combined together. Similarly, the Colour feature helps in finding and harmonising the colour difference between the input image and the candidate completion image. Next, we propose a ranking scheme that retrieves top-1 most suitable candidate image. The ranking scheme is able to work with any number of features computed and also satisfies the value-Invariance property.

The experimental results show that we are able to find a matching image that is used to complete the input image seamlessly. The approach is data-driven and there is no need of labelling by user. Unlike other techniques, our approach utilizes data from matching image to fill the missing region. We use a number of techniques to improve the completion method, i.e. we use Poisson blending to combine the images, we use graph cut techniques for the purpose of removing blending artefacts. Although, the completion process is automated, we also allow users to select an image from top-10 matches in order to have a completion that is semantically valid. Our experimental results show the usefulness of the work.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

PDE	Partial Differential Equations
FFS	Fast Fractal Stack
SIFT	Scale Invariant Feature Transform
CBC	Collection –Based Completion
DISC	Data Intensive Scene Completion
SFTA	Segmented –based Fractal Texture Analysis
TTBD	Two-Threshold Binary Decomposition
SSD	Sum of Square Difference
RGB	Red Green and Blue

Chapter 1

Introduction

CHAPTER 1. INTRODUCTION

Scene Completion is an interesting image processing problem that arises in a variety of contexts. Sometimes, we want to erase something from old snapshot i.e. ex-family friend from family photo or another undesirable object. In some cases, there is some missing data in an image. In summary, Scene completion is the task of filling missing parts of the image in such a way that the modifications are not recognizable from naked eye.

1.1. Scene Completion

Scene Completion problem has been greatly considered in literature. Scene completion schemes can broadly be divided into three categories, namely (1) PDE-based methods [1] (2) Exemplar–based methods [2] and (3) Data intensive methods [3]. These methods can be defined as follows:

1. **PDE-based Methods** in which missing region of the image is filled by diffusion process. In general, colour information is propagated inward into the completion part with different variations to fill the hole in the image.
2. **Exemplar-based Completion Methods** rely on the observed part of an incomplete image and use regions in the image that are best suitable for completion.
3. **Collection-based Completion Methods** use a large collection of images to find semantically valid matching images and then use segments from the best match to complete the missing region.

Note that (1) and (2) above, i.e. PDE-based and Exemplar-Based completion methods work by filling missing regions using known parts of the same image. These methods don’t perform well in a variety of situations, e.g. when missing regions of the image are not bounded by a textual region; or when it is not possible to fill the missing region of the image with the data of the source image, say for example when the entire roof of a house is masked out.

Hays et al. [3] proposed Collection-Based scene Completion (CBC). The idea is to fill the missing part of the image with an image that is extracted from a large image repository. CBC works by filling the missing region of the image by using matching content from other semantically valid images rather than the source image. Consequently, an entirely novel object and texture can be inserted in the missing part.

1.2. Research Problem

In this work, we investigate collection-based scene completion. We first give some limitations in CBC.

1. Completed images often suffer from texture mismatch, i.e. visibility of low-level artefacts in the completed image.
2. Another issue is the selection of the best completing image from a large collection of images.
3. Scene completion algorithms do not recognize objects during completion of scene.

We define the following research problems.

(1) Is it possible to obtain the same results as state-of the- art in literature?

(2) Is it possible to address the texture mismatch issue?

(3) Is it possible to have a ranking scheme that can automatically select the best completion image?

We now give our contributions.

1.3. Our Contributions

We address the issues pointed above ((2) – (3) above).

1) We propose a ranking scheme that finds the top-1 image from a large image repository; top-1 image is the best completion image based on the computed features ((3) above). The ranking scheme we propose, computes a set of features which help in finding the matching images which are semantically similar to the query image. The ranking scheme

selects the top-1 image using a value-invariance scheme rather than actual match values. Our results show the significance of the proposed scheme.

2) We use the top-1 image to find the finest region in the completing image that is most similar to the area around the missing region in the input image. Thus, we are able to address the texture mismatch issue ((2) above).
3) Further, in order to merge the input and the completing images seamlessly, we use techniques like Poisson blending. In order to avoid blending vestige, we use state-of-the-art graph cut segmentation scheme that finds the outline for Poisson blending that has minimal image gradient magnitude. Our results show the usefulness of the blending method.

1.4. Scene Completion Problem

We now give an outline of the Scene completion steps.

1.4.1. **Matching Image Retrieval**

In order to successfully complete missing region, we find the matching images that are semantically valid for the query image as say for example, we don’t want to complete a building top with a tree. We find images that are semantically valid for the query image.

We have used Gist descriptor [4] for the purpose which performs well for scene recognition that are semantically valid. We also compute the Texture feature [5] and the colour difference to retrieve images that are textually similar and have the same colour theme with the query image.

1.4.2. **Context Matching**

After finding semantically similar images, we find the best patch from the matching image. We have used traditional template matching to align the matching images with the space around the missing region of image. It is not convenient to search the whole image for finding best patch, so we define the local context to be all pixels within 85 radius pixels (after extensive experimentation) around the missing region. This area is compared against the matching image at different scales and all valid translation using Sum of Square difference (SSD). We consider translation and scale for which context is fully hold in the identical image. In addition to pixel-wise positioning we also figure out the texture similarity using median filter in order to measure the compatibility of source image and fill-in region.

1.4.3. **Image composition**

We have stitched matching image into the incomplete image as its best induction using Poisson blending. In order to avoid visible seams, we use graph cut [6] technique to find the borderline for Poisson blending that has minimal gradient magnitude. When graph cut technique and Poisson blending is performed in sequence it makes sense to improve the seams in such a way that it lessen the visible seams left behind.

1.5. Thesis Organization

This thesis is organized as follows. Chapter 2 focuses on history of scene completion. Chapter 3 is a discussion of the methodology adopted to solve the problem. Implementation of all these techniques and methods is discussed in Chapter 4. A comprehensive analysis and our results are presented in chapter 5. Chapter 6 concludes this thesis.

Chapter 2

Scene Completion

CHAPTER 2. SCENE COMPLETION

In this chapter, we discuss concepts related to scene completion. First, we define the scene completion problem and then examine the scene completion techniques proposed in literature. After that, we discuss data intensive approach for scene completion and give a detailed outline of the steps for completing missing part in an image.

2.1. Scene Completion

Scene completion is an interesting research problem. Given an input image I and a supporting image set S, where S can either be (1) the image itself i.e. $S \leftarrow I$ or (2) a collection of images which may contain images $S^{'} \subseteq S$ that can be used for scene completion. The objective is to find the best match in $S^{'}$ that can be used for completing the image I such that completion is visually plausible.

Formally, we define the Scene Completion problem as follows.

Definition 2.1 (Scene Completion): *Given an input image I, a target region Ω to be filled, the source region ω may be defined as entire image minus target region* $(\omega = I - \Omega)$ *that is used to fill the target region Ω.*

We now define Data Intensive Scene Completion.

Definition 2.2 (Data Intensive Scene Completion): *Given an input image I, the image set S where S is the collection of images, find the images $S^{'} \subseteq S$ where $S^{'}$ is a set of candidate images for completion; and then select the best match M in $S^{'}$ that is used for completing image I.*

Many approaches have been proposed in literature for the problem which can be broadly classified into three categories namely (1) PDE-based methods [1] (2) Exemplar–based methods [2] and (3) Data intensive methods. An outline of these methods is given below.

1. **PDE-based Method** fills the missing region in the image by a diffusion process.
2. **Exemplar-based Completion** methods rely on the observed part of an incomplete image, use region in the image that is suitable for completion.
3. **Collection-based Completion** methods use a large collection of images to find semantically valid matching image and then use segments from the best match to complete the missing region.

We now discuss these in detail.

2.1.1. **PDE-Based Completion**

PDE-based completion method is used for filling missing part of an image by diffusion process. It works by propagating the data from borderline of the hole to the interior part of the region. Diffusion process is imitated by partial differential equations.

Partial differential equations inspired from equation in fluid dynamics have been used for image completion. Bertalmio et al. [1] proposed an algorithm that is used for digital in-painting of still images. When the user selects the region to be restored, the algorithm automatically fills in the missing parts by propagating the information from boundary of selected region. An example of such a completion is shown in Figure 2.1.

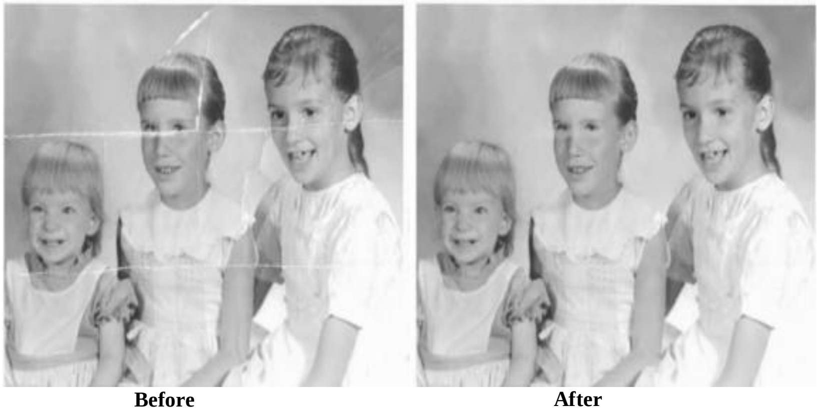


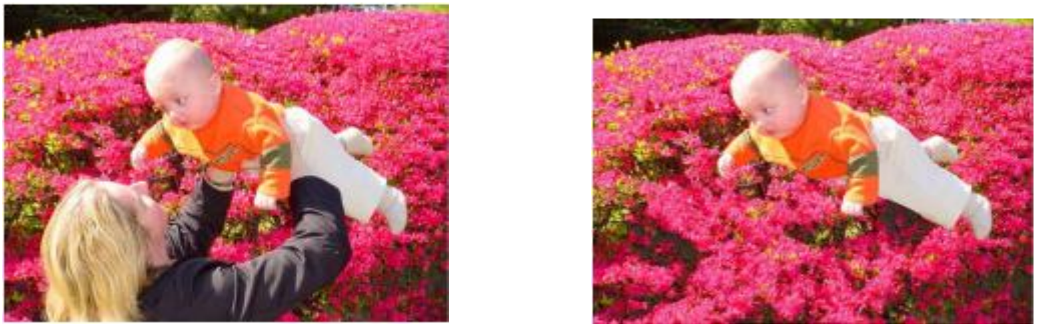
Figure 2.1: PDE-based image completion [1]

Discussion: One of the main problems with these methods is reproduction of large texture regions as PDE-based completion methods are designed for filling small or narrow holes in an image.

2.1.2. **Exemplar-Based Completion**

Exemplar Based completion methods are based only on observed part of an incomplete image. In literature, an exemplar-based texture synthesis [2] technique has been proposed that is used to complete an image with source image. The idea is to first determine the filling order of the missing region in which image would be completed. For that purpose each pixel maintains some confidence value, which together with isophotes information is used to determine their fill priority.

Exemplar based completion is done using non-parametric model [7]. A non-parametric model for texture synthesis is used that grows texture pixel by pixel, outward from initial seed. Given a sample texture image, a new image is synthesized one pixel at a time. In addition, scene completion can be accomplished by adjusting and cloning large image patches whose macrostructure is compatible with the hole [8]. A hybrid approach for completion has also been proposed that works by decomposing the image into two images (1) a structure image and (2) a texture image. Reconstruction of the image is performed separately by using structure in-painting and exemplar based texture synthesis [9]. An example of exemplar–based completion is shown in Figure 2.2.



Before

After

Figure 2.2: Exemplar-based completion [2]

Discussion: These methods are well suited for the case when there is small hole to be filled. But there are some cases for which exemplar-based methods do not perform well for example when there is a large missing region in the image. Further, if the total area is masked out it cannot be filled with existing image.

2.1.3. **Collection Based Completion**

Collection-Based Completion is based on filling missing part by finding semantically valid matching images that can be used for extracting optimal patch. Collection-based methods find semantically valid matching images that are used to fill the missing area by finding matching patches from matched images [3].

We now discuss the steps in collection-based scene completion.

2.1.3.1. Finding Matching Images

Finding a matching image is the first step in collection-based scene completion. Many approaches have been proposed in literature for finding matching images which are semantically valid [4]. Matching images is found by outline based feature (chain code string feature) for content based image retrieval [10]. Local information and spatial relationship can also be used for recognizing of indoor scenes [11]. In addition, BlobWorld [12] representation of image can be used for retrieving relevant images. BlobWorld actually provides the transformation of raw pixel data to smallest set of image region which are coherent in texture and colour.

Texture plays a very important role in image analysis and image understanding. One of the widely used approaches to characterize the texture is counting the presence of grey-level at different displacement and angle. Different statistics such as entropy, contrast and energy can be calculated using Grey Level Co-occurrence Matrix (GLCM) in order to obtain texture feature [13].

For texture analysis, Filter bank-based methods [14] are used for image classification and segmentation. Gabor filter [15][16] is an example of filter bank-based method for its variance respect to rotation, scale and displacement. In addition Gabor filter is used for texture extraction by calculating the mean and variance of Gabor filtered image [17].

Recently, Segmented-based fractal analysis algorithm [5] was proposed for texture feature extraction. This can be done by finding binary images from which we can compute fractal dimensions. This algorithm overcomes the limitations of fast fractal stack(FFS)[18].FFS is used to calculated the fractal dimensions from binary images that can be obtained using binary stack decomposition algorithm[19] that does not include information such as grey level distribution for selecting threshold.

Content based image retrieval is done by calculating Gist [4] feature that finds the matching images that are semantically valid from collection of images. Gist feature actually bypasses the segmentation and represent the structure of whole image. Furthermore Scale Invariant Feature Transform (SIFT) descriptor [20] is used for content-based image retrieval. Moreover, side information i.e. location of body is also utilized for improving image retrieval accuracy.

Once the matching images have been discovered then graph-cut techniques are used to find the pixels in “completing” image and the “completed” image that will be joined to obtain the final image. We now discuss graph-cut techniques for scene completion.

2.1.3.2. Graph-Cut Applications

Graph- cut is an important technique and has been used in domains like computer vision and image processing. Graph cut is used for image segmentation [21], image restoration [22], image and video synthesis [23].

In literature, method of segmentation using graph-cut has been proposed. The idea is to separate a pixel from a special point outside the image using cut of minimum cost. The minimum cut creates groups around pixel which are disjoint or nested each other which actually give natural segmentation of image[21]. Moreover, graph cut can be used for segmentation of multi-dimensional images. For that purpose, user only needs to specify which pixels should belong to background and foreground. Optimal segmentation can be obtained using graph cut [24].

A new method [25] is proposed that actually calculates the visual correspondence with occlusion using graph cut. The method performed well both at detecting occlusion and computing disparities. There are many application of graph-cut algorithm such as it is used to regain the object surface using silhouette and colour information [26].After the graph cut, we have boundary of minimum gradient magnitude now blending is needed to composite the image without visible seams.

2.1.3.3. Image Blending

After stitching the patch into incomplete image, image blending is required to remove the visible seams. Image blending is another important subject in computer vision and image processing. This can be done by using generic interpolation machinery [27] based on solving Poisson equation for seamless image editing. This method allows to seamlessly access of opaque and transparent source image region into destination region. Multi-resolution spline technique [28] that is used to combine two or more images in order to create image mosaic. For this purpose, image is decomposed into band-pass filter component images. These component images are assembled into corresponding band pass mosaic and these are joint using weighted average within a transition zone which is actually proportion to size of wavelength. In this way, a spline is matched to feature and if coarse features are found near the border of image then these are blended gradually without producing the blurring artifacts.

Image blending can be done using multi-blending that uses vector of blending weights, one for class of each feature instead of single transparent value. Multi-blending preserves most of feature of both palette and background window [29].

2.2. Collection-Based Completion [3]

In this work we investigate the Collection-based completion method proposed by Hays et al.[3]. The steps in CBC are as follows:

1. Data Repository
2. Features calculation
3. Context Matching
4. Image composition using graph-cut and Poisson blending

Now, we discuss all these steps in detail.

2.2.1. **Creating Data Repository**

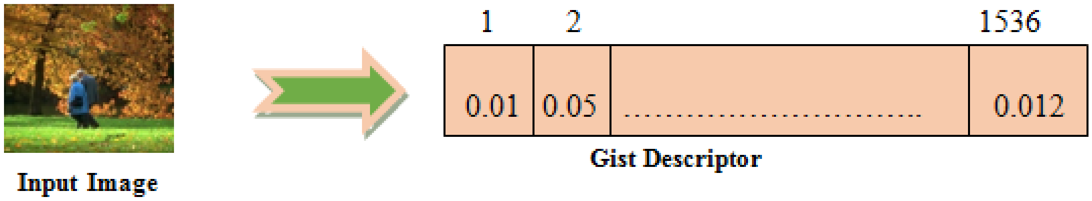
Data Repository has fundamental importance. Because it is data intensive approach so creating data repository is very important step in CBC.There are different methods of creating data repository, such as using cloud set up used by Hays et al.[3]. The focus of Hays was on quantity of data instead of quality. Hays et al. used 2.3 million images and retrieves top 200 images for image completion. Our focus is on quality of data; the challenge is to find the best image from sample set. We take a sample set of 300 images and retrieve the ranked best images from sample set for scene completion.

2.2.2. **Features**

In order to successfully fill the missing region, we need matching images which are not only (1) seamless (2) and proper textured but are also semantically valid. In order to find the matching images from the huge database Hays et al.[3] computed the Gist feature and colour feature between input image and database image. Gist describes the shape, structure of images at different scales and orientation. According to Gist feature, two images are semantically valid if they have a smaller distance between them.

2.2.2.1. Gist Descriptor

The Gist descriptor was initially suggested in [4]. The idea is to establish the low dimensional representation of the scene; the dimensions naturalness, roughness, openness, expansion and ruggedness can be used for the representing structure of scene. Gist descriptor emphasis on shape of scene itself, structure of scenes.



2.2.2.2. Colour Difference

There are many colour space representation schemes for which most commonly used is RGB because mostly images are available in RGB. The purpose of colour spaces is to care the process of describing colours between people or machine. Hays et al.[3] used Lab Colour space for calculating similarity between query image and matched images. Lab colour space has three dimensions where dimension L is lightness and $a \; and \; b$ dimension are for colour dimension. Lab colour space include all perceivable colour, which means its gamut exceed those of RGB. The purpose of lab colour space is to create a space which can be calculated using simple formulas of XYZ, but it is more perceptually uniform as compared to XYZ colour space.

The main attribute of lab colour space is; it is device independent, device independent colour space is one where same colour would produce under the same specification.

2.2.3. **Context Matching**

After having semantically valid scenes, the next step is to find best region from matching image that can fill the hole of input image in such a way that no one can detect modification.

It is more convenient to search the matching pixels that are close to boundary of hole rather than for whole image. All pixels near hole's boundary are important as compared to other pixels so Hays et al. defined local context to be all pixels 80 radius around missing region's boundary. For context matching, First matching image is resized to input image at different scale then local context is compared against matching image to find the matching patch.

2.2.4. **Image composition using graph cut and Poisson blending**

The idea for image composition is that instead of copying and pasting matched patch into hole, stitch it together with original image so that seam between matching patch and original image is less noticeable. This can be done by finding boundary having minimum gradient magnitude using graph cut. Hence visible seams can be removed using Poisson blending and graph cut.

Hays et al. synthesize each matching scene into the incomplete image at its best induction using a form of graph cut seam finding [23] and Poisson blending [27].

We now give some of the limitations of CBC.

Discussion: There are some limitations of collection based completion.

1. There is failure case in which low-level artefacts are visible i.e. texture mismatch;
2. Another reason of failure is insufficiency of good matching images.
3. Existing algorithm for scene completion does not recognize the objects during completion of scene.

We have tried to solve the first two problems (1) texture mismatch, we compute texture feature [5] of query image and image in data repository then calculate L2 distance in order to retrieve best image that are texturally similar to query image (2) Lack of good matching image: for this particular case pointed in Hays's work, we have good enough matching images that are semantically valid to query image.

2.3. Image Retrieval Relevance

As discussed in Section 2.2 , Collection –based completion is based on finding a suitable patch of an image from a large collection of images that can be used for completing the missing portion of the image.

The matching images are calculated based on two features, namely Gist and colour, and resultant match is maximum match based on the features considered for the purpose. However, such a matching may not be desirable in situations where

1. More than two features are calculated.
2. There is significant difference in the matching values for different features.

For example, for images X and Y if the matches reported are (2.8, 0.9) and (0.2, 3.6) the image selected by Hays et al. would be image Y as the total match score for Y is higher than X. however, it appears as if the match scores for image X were more convincing as both the features for image X had relatively high score; and for image Y, one feature had a lower value while the other feature had relatively higher value. This means that the higher match score for one particular feature should not dominate the lower value scores by the features especially when more than two features are considered. This property is referred to as value-invariance [30].

The value invariance property can be stated as:

Definition 2.3(Value Invariance): *Let D denote a relation which include score value $v_1 \leq v_2 < \cdots$ let $s_i^{'}$ be any set of score values satisfying $v_1^{'} \leq v_2^{'} < \cdots$, and define $D_1^{'}$ to be D with all scores v_i replace with $v_i^{'}$. The value invariance property requires that $R_k(D) = R_k(D^{'})$ for any k.*

In this work, we proposed a unified ranking scheme that satisfies

1. value–invariance property and
2. is able to handle any number of features.

The idea is to have an image rank for each feature and compute the final rank based on majority scores. The details are discussed in section 3.1.4.

2.4. Summary

In this chapter, we have discussed scene completion schemes proposed in literature. We have described in detail the steps and issues in a data intensive scene completion. Furthermore, we have also discussed value-invariance property and we argue that the proposed ranking scheme obeys value-Invariance property.

Chapter 3

Data Intensive Scene Completion

CHAPTER 3. DATA INTENSIVE SCENE COMPLETION

This chapter is about Data Intensive Scene Completion (DISC). An overview of DISC has been presented in Section 2.2. In this chapter; we present the details of DISC. We divide the DISC process into three steps, namely (1) Matching Image Retrieval (2) Image Synthesis and (3) Image Harmonization. An outline of the steps is given below:

1) Matching Image Retrieval, i.e. finding the images that could be best candidates for completing the input image.
2) Image Synthesis i.e. using graph cut techniques to complete the input image using the best matching portion of the best matched image.
3) Image Harmonization i.e. using techniques like blending to harmonise the image by removing any visible seam.

We now discuss these steps in detail.

3.1. Matching Image Retrieval

The first step in Data Intensive Scene Completion is to be able to find images which are not only (1) seamless (2) have a proper texture but also are (3) semantically valid[4]**.** Nowadays, digital photography, cheap storage and lightning network speed made it possible to have large data repositories. Broadly speaking image retrieval can be classified into two classes (1) text-based image retrieval (2) content-based image retrieval. Whilst text-based image retrieval depends on image metadata (user-tags); Content-based image retrieval is based on the content of the image. Content of images refers to visual information such as colour, texture, shape and structure of the image.

In order to find the matching images from the data repository we have considered three features [4][5] namely (1) Gist (2) Texture and (3) Colour Difference. Gist feature have gained a lot of attention in context of scene recognition. It describes the shape and structure of the images at different scales and orientation. Two images are semantically valid if they have smaller distance between them. Texture is very important when segments from different images are being combined together. Similarly, the Colour feature helps in finding and then harmonising the colour difference between the input image and the completion image. We now discuss these features in detail.

3.1.1. **Gist Descriptor**

From scene perception studies, it is shown that witness can notice real world images at a single glance [31]. During this formation, the witness's visual system forms spatial representation of real world scene that is suitable for understanding the meaning of scene. This representation actually specifies the Gist of the scene.

Gist consists of all level of information from low- level (e.g. colour) to intermediate (e.g. texture, shape) to high level (i.e. semantic information), so Gist can be defined as conceptually and perceptually. Conceptual Gist refers to semantic information that is understood while seeing the scene and then disappeared from the view. In other words, we can say that conceptual Gist is a verbal description of an image that was actually perceived. Perceptual Gist can be defined as the structural representation of natural scene that is built during perception.

Perceptual content of Gist contains the image properties, e.g. spatial frequency, colour and texture. These properties actually provide structural representation of real-world images. Using these properties, e.g. filtering out edges of images, one can determine the structure of the scene that is helpful for scene identification. Two images having same structural representation may belong to same category. Oliva and Schyns [32] have studied the role of spatial frequency scale for rapid scene identification. They distinguish the information from blobs and edges by presenting hybrid stimuli (combining low spatial frequency of an image and high spatial frequency of an image). Figure 3.1 shows hybrid stimuli where low spatial scale shows oriented blobs and high spatial scale conveys information related to surface and contours.

Gist descriptor has received a lot of attention in the context of scene recognition. Gist descriptor was initially proposed in [4]. The idea is to develop the low dimensional representation of scene using dimensions like naturalness, roughness, openness, expansion and ruggedness that can be used for representing the structure of scene.

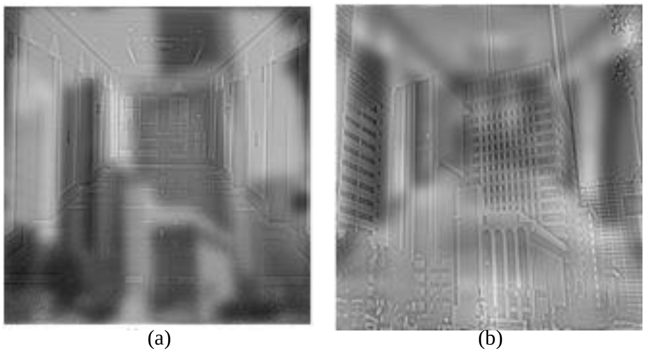

(a) (b)

Figure 3.1: (a): A hybrid image representing of a hall scene in high spatial frequency and urban scene in low spatial frequency. (b): The hybrid image showing a urban scene in high spatial frequency and hall scene in low spatial frequency (image taken from [32])

We computed the colour difference in LAB colour space [33] between database image and query images. We now discuss in details the Colour difference in LAB colour space.

3.1.2. **Colour Difference**

Colour is another important feature for image retrieval. Humans are not so be effected by small variation in colour. There are different colour space representation schemes of which most commonly used is RGB because mostly images are available in RGB. RGB is non-uniform colour space that can be represented by human eye perception. The aim of colour spaces is to aid the process of describing colours between people, machine or programs.

L*a*b colour space was defined by the CIE (International Commission on Illumination) and is colour opponent space with dimension L for lightness and *a and b* for colour dimension. L*a*b colour space includes all recognizable colours, which means its gamut (*subset of colours)* exceeds those of RGB colour model. The purpose of L*a*b colour space is to define a space which can be calculated using simple formulas of XYZ, but it is more intuitively uniform as compared to XYZ colour space. Perceptually uniform means that when choosing a colour space, we want to have a colour space that is (1) easy to compute and (2) exhibits perceptual uniformity which means that a change Δ_i in the colour value has an impact Δ_j on the visual appearance of the image such that Δ_i is comparable to Δ_j. The Lab colour space is used in many applications, e.g. in Adobe Photoshop, when graphic for print has to convert from RGB to CMYK.

We have used L*a*b Colour space [33] for calculating similarity between query image and matched images. One of the advantages with lab colour space is that it is device independent. Device independent colour space is one where a set of parameters will produce the same colour regardless of the underlying hardware.

L*a*b Colour Space comprises of three coordinates namely (1) L which represents lightness of colour (L* =0 indicates black, L*=100 represent diffuse white) (2) colour position between red/magenta and green (a*, negative values indicate green and positive values indicate magenta/red) and (3) colour position between blue and yellow (b*, negative values indicate blue and positive value indicate yellow). The coordinate values ranges from 0 to 100.

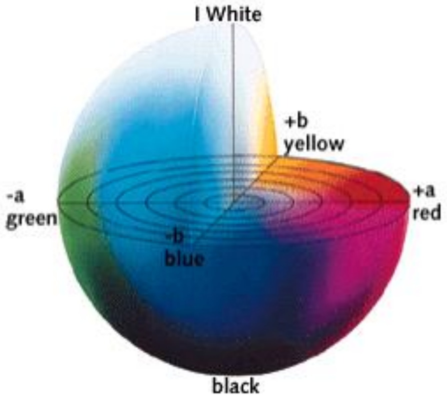


Figure 3.2: Lab colour space Model

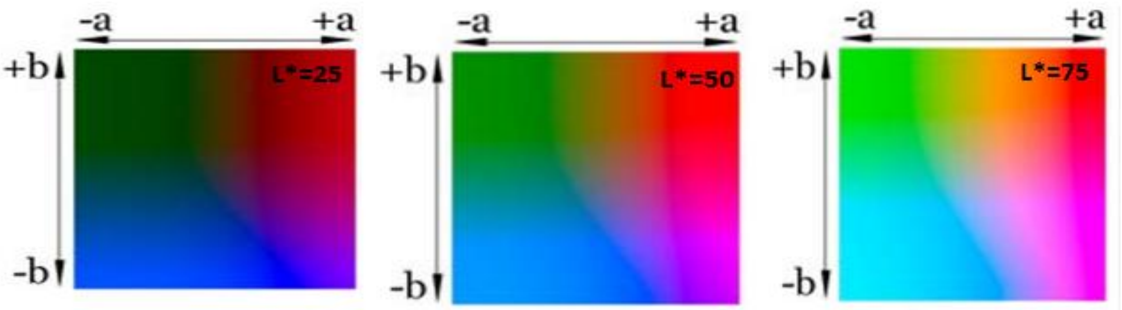
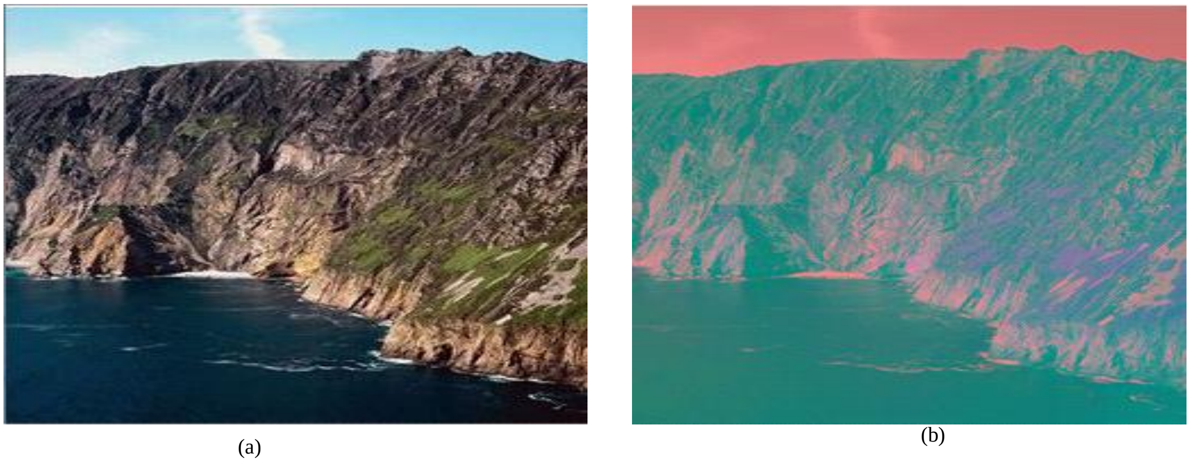


Figure 3.3: The CIE 1976 (L* a* b*) colour Space

The Lab colour space model is a three dimensional model, i.e. it can only be represented in three dimension space. Colour space conversion is just the transformation of colour from one basis, e.g. RGB, to another, e.g. Lab Colour Space. An example of such a transformation is shown in Figure 3.4 which is just a representation of colour from one basis into another.



(a) (b)

Figure 3.4: (a) RGB image (b) Image in Lab Colour space

We now discuss the texture feature in detail.

3.1.3. **Texture Feature**

Texture plays a very important role in image analysis and understanding specifically for remote sensing and medical imaging. In addition, texture information is used for image segmentation by classifying pixels based on surrounded texture information.

We have computed SFTA texture feature proposed in [5]. The SFTA algorithm is divided into two sections. First, we crumble the grayscale images into binary images. That can be done using two thresholds binary decomposition (TTBD). In two threshold binary decomposition, we have grayscale image and return a set of binary images. For binary image, some threshold values are required that can be obtained using multi-level Otsu algorithm. .The Otsu algorithm finds thresholds that actually minimize the intra-class variance. This algorithm is employed recursively to each region of image until desired number of thresholds n is achieved where n is user defined parameter.

The next step of TTBD is to decompose the input grayscale image I into a set of binary images. This can be done by using pairs of threshold from set of threshold and applying two-threshold segmentation as Eq. 1,

$$I_b(x,y) = \begin{cases} 1 \; if \; t_r < I(x,y) < t_p \\ \quad 0 \; otherwise \end{cases} \tag{1}$$

Where t_r and t_p denote lower and upper threshold values respectively. The main assets of TTBD is that the set of binary images obtained is a superset of all the binary images that were retrieved by applying one threshold segmentation using threshold that figure out with multi-level Otsu algorithm [34].

3.1.3.1. Otsu Algorithm

The purpose of Otsu algorithm is to cluster-based segmentation. The idea of this algorithm is that image consists of two classes named as background and foreground then it computes the optimal thresholds that separates the classes so that their combined variance (intra-class variance) is lesser so that equivalently their inter-class variance is maximal.

An outline of Otsu algorithm is given in Algorithm 3.1.

Algorithm 3.1: Otsu Algorithm

```
1. Compute probabilities and histogram of each level
2. Setup initial $\omega_i(0) and\ \mu_i(0)$ where $\omega_1(t) = \sum_{i=1}^{t} p_i$
                                  and  $\mu_1 = \sum_{i=1}^{t} ip_i/\omega_1(t)$
3. Step through all possible threshold m = 1……maximum Intensity
    1. Update $\omega_i\ and\ \mu_i$
    2. Compute $\sigma_n^2(m)$
4. Output desired threshold, i.e. maximum$\{\sigma_n^2(m)\}$
```

3.1.3.2. SFTA Feature Extraction Algorithm

After applying two thresholds binary decomposition, next step is to construct SFTA feature vector that consists of binary image's size, mean gray level and boundaries' fractal dimensions. The fractal measurements are used to define the complexity of objects and structures segmented in input image. The boundary region of binary image is represented as border image as follows:

$$\Delta(x, y) = \begin{cases} 1\ if\ \exists(x^{'}, y^{'}) \in N_8[(x, y)] \\ I_b(x^{'}, y^{'}) = 0\ \ and \\ I_b(x, y) = 1 \\ 0\ otherwise \end{cases} \tag{2}$$

where $N_8[(x, y)]$ are 8-connected neighbor pixels of (x, y). The value of $\Delta(x, y)$ will be 1 if the pixel at position (x, y) in the corresponding binary image position $I_b(x, y)$ is 1 and the value of at least one neighbor pixel is 0; the value of $\Delta(x, y)$ will be 0 otherwise. The fractal dimensions are calculated from each border pixel using box counting algorithm.

The gray-level and size provide additional information that is extracted from binary images. Hence, the dimension of SFTA feature vector corresponds to number of binary images obtained using TTBD process multiplied by 3. The details of the box counting algorithm are given in Section 4.1.2.

3.1.4. **Top-1 Image**

In recent years, large image repositories have been created in many areas, i.e. in academic, commercial and medical domains. In general, images are manually tagged with some additional descriptors that allow retrieving images. However, the method of manually tagging images is rather time consuming.

One image retrieval method that actually addresses these issues is content based image retrieval. Content-based image retrieval is a process of finding matching images of query image from large collection of data. These are performed based on features namely colour, texture, shape. There is a problem in content based image retrieval. For different queries, different types of features have different significance. Now the issue is how to derive the weighted scheme to balance the relative importance of all features and there is no universal formula for all queries.

As already mentioned that in this work, we consider three features namely Texture, Gist and colour and compute the feature vector of each image. After intensive experiments, we observed significance of feature and assigned weights to Gist, Texture and colour as 0.5, 0.2 and 0.3 respectively. For a given input image, we retrieve the matching images based on minimum distance. We have used L2 distance for the purpose as a similarity measure to find the distance between the query image and images in the data repository (Eq. 3).

$$\mathrm{ActualDifference} = \mathrm{sort}(\sum_{\mathrm{i}=1}^{\mathrm{n}}(\mathrm{GW} * \mathrm{GistDiff(i)}\,, \mathrm{TW} * \mathrm{TextureDiff(i)}, \mathrm{CW} * \mathrm{ColorDiff(i)}) \tag{3}$$

where

$\mathrm{GW} : \mathrm{Gist\ weight} = 0.5$

$\mathrm{TW}: \mathrm{Texture\,weight} = 0.2$

$\mathrm{CW}: \mathrm{colour\ weight} = 0.3$

The drawback of this scheme is that it is biased towards the actual values. There are situations where the distance between images based on one particular feature is very high and that distance dominates the results from other features. To address this problem, we have proposed a unified ranking scheme that is able to handle any number of features and satisfies the value invariance property (defined in Section 2.3).

The idea of this ranking scheme is to have an image rank combined with weights for each feature and compute final rank based on majority scores. We have shown ranking scheme in Table 3.1. Where GR, CR and TR are individual' ranks of images based on Gist, Colour and Texture respectively. After calculating individuals rank then we compute final rank of the image based on the values and the rank score. The difference in results has been highlighted in Table 3.1.

	Gist			Colour			Texture			Unified Rank
	GR	W1=0.5	A1=GR*W1	CR	W2=0.3	A2=CR*W2	TR	W3=0.2	A3=TR*W3	Sum(A1,A2,A3)
Image1	2	0.5	1	1	0.3	0.3	3	0.2	0.6	1.9 R2
Image2	1	0.5	0.5	3	0.3	0.9	2	0.2	0.4	1.8 R1
Image3	3	0.5	1.5	2	0.3	0.6	1	0.2	0.2	2.3 R3

Table 3.1 : Ranking Scheme

3.2. Context Matching

After having semantically valid matching *s* images, the next step is to find the best region from matching image that is used to fill the hole of input image in such a way that the changes are not visible with naked eye.

It is more convenient to search the matching pixels that are close to boundary of hole rather than searching the whole image. Note that pixels near to holes' boundary are more important as compared to other pixels. We define local context to be all pixels 85 radius as in literature around missing region's boundary as shown in Figure 3.5.

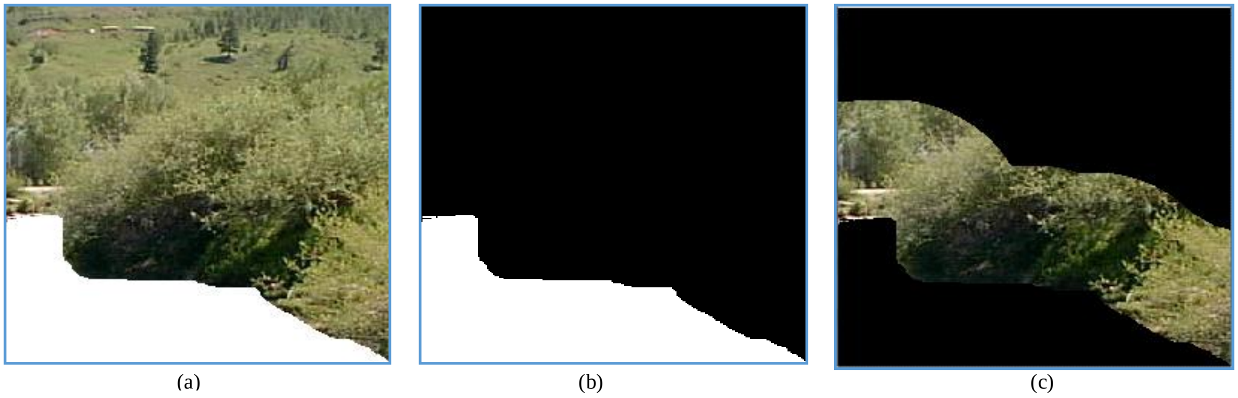


Figure 3.5: a) Input Image b) Mask Image c) Local context around hole

For context matching, first matching image is resized to input image at different scales. This context is compared against matching images across all translations and scales [.081, 0.90, 1] as in literature using Eq.4.

$$\text{ContextSSD} = \sum_{i=1}^{3} (\text{InputImage} - \text{MatchedImage})^2 \tag{4}$$

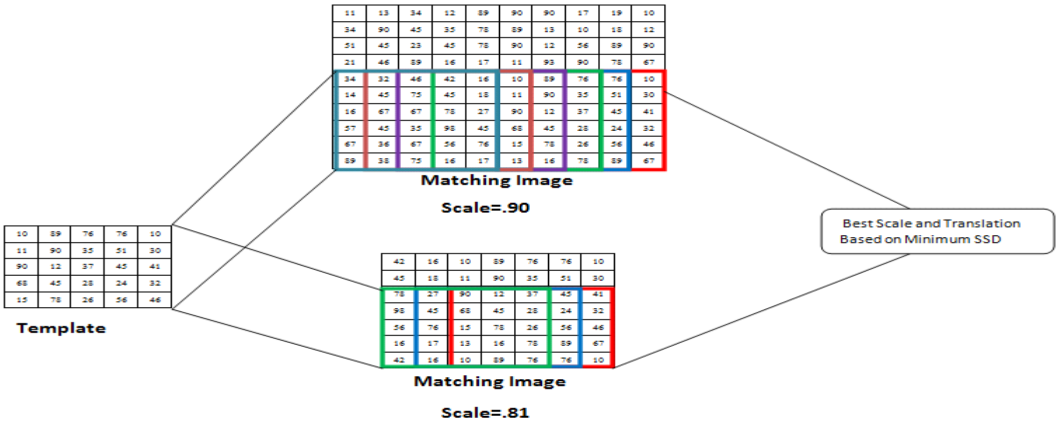
where $\mathrm{i} =$ Colour Channel.

The translation and scale for which context is contained in the matching images are considered as best placement based on minimum SSD error. In addition to find SSD error, we have also computed texture similarity in order to measure the affinity between source image and region that is being filled under local context.

A texture descriptor is computed using median filter of image gradient at each pixel. Median filtering is basically a non-linear operation in image processing that is used to reduce the “salt and pepper” noise. Median filtering is most appropriate when aim is to remove noise and preserve edges as well. The basic idea behind the median filter is to consider each pixel along with its neighbour pixels and replace the pixel value with the median of those values. The median is calculated by sorting all surrounding pixel values in an ascending order and replacing pixel value being considered with the middle pixel value.

Median filtering is a kind of smoothing technique like Gaussian filtering. All smoothing techniques are impressive at removing noise in smooth regions but it is very important to preserve the edges because edges are very important for the visual importance of an image.

Median filtering performs well for removing small to medium level of noise; on the other hand its performance deteriorates when noise levels are high. After computing texture descriptor using median filter, both texture descriptors are compared using sum of square difference (SSD).



11	13	34	12	89	90	90	17	19	10
34	90	45	35	78	89	13	10	18	12
51	45	23	45	78	90	12	56	89	90
21	46	89	16	17	11	93	90	78	67
34	32	46	42	16	10	89	76	76	10
14	45	75	45	18	11	90	35	51	30
16	67	67	78	27	90	12	37	45	41
57	45	35	98	45	68	45	28	24	32
67	36	67	56	76	15	78	26	56	46
89	38	75	16	17	13	16	78	89	67

10	89	76	76	10
11	90	35	51	30
90	12	37	45	41
68	45	28	24	32
15	78	26	56	46

42	16	10	89	76	76	10
45	18	11	90	35	51	30
78	27	90	12	37	45	41
98	45	68	45	28	24	32
56	76	15	78	26	56	46
16	17	13	16	78	89	67
42	16	10	89	76	76	10

Figure 3.6: Template matching at different scale and Translation

3.3. Finding Best patch using Graph Cut

Once the completing patch has been obtained, we composite matching scene into incomplete image. The input mask covers the area of the incomplete image that user specified but edges of the mask may not be a good option for stitching two images when gradient of both the images are rather different from each other. This can be reduced by refining the input mask. We allow the mask to leave from its original path and find its best place so that subsequent blending looks more persuasive.

Instead of copying and pasting matched patch into hole, we stitch it together with original image such that the seam between matching patch and the original image is less noticeable. This can be done by finding maximum flow/minimum cut via graph cut. Graph-cut schemes have been used in domains like image segmentation [35][36][21][37], image restoration[22] and stereo[38][39][40] [25],Texture synthesis [23], object recognition[41], shape reconstruction[42], etc.

For solving the graph cut problem, each pixel in the overlapping region represents node of the graph, all node are connected to its four neighbours. We assign weights to all edges of the given pixel using Eq. 5**.**

$$\mathrm{SM}(\mathrm{s},\mathrm{t},\mathrm{A},\mathrm{B}) = \|\mathrm{A}(\mathrm{s}) - \mathrm{B}(\mathrm{s})\| + \|\mathrm{A}(\mathrm{t}) - \mathrm{B}(\mathrm{t})\| \tag{5}$$

Where

A= Matching Patch

B= Old Patch

s= current pixel , t= neighbour pixel

In order to find maximum flow inside graph, we need two terminal sources and sink as shown in Figure 4.4. All pixels under mask are connected to sink and similarly all pixels at the maximum border of local context are connected to source terminal. Thus, flow must go through the border of the input mask. For calculating maximum flow/ minimum cut, we have used C++ library implementing Max-Flow Algorithm [6] **.**

For using this library we need two matrices (1) Adjacency matrix that represents edge weight of all neighbours using Eq. 5 and (2) Terminal matrix that represents the source and sink edge weights. All pixels under mask are connected with sink with infinite weights and all pixels at maximum border of local context are connected with source terminal with infinite weight; this means that these pixels must come from the matching patch and original patch respectively. All other pixels under overlap region are designated as source or sink by the max-flow Algorithm. These two matrices are used for max-flow algorithm that specifies the pixels coming from the matched or the original patch.

3.4. Poisson Blending for removal of visible seam

The basic purpose of blending is to seamlessly blend a source image into a target image. The simple method is to copy pixels from source image and paste into target image but it will produce visible seam because human visual system is more sensitive to gradient rather than absolute intensities.

3.5. Summary

We have discussed steps of Data intensive scene completion (DISC) in detail. We have discussed the importance of image retrieval and the features that are computed for finding similar images. We have also discussed the image composition step in detail. Smoothing techniques are also discussed to conclude this chapter.

Chapter 4

Implementation

CHAPTER 4. IMPLEMENTATION

In this chapter, we discuss implementation details of the steps in DISC. We first talk about scene matching and discuss the feature computation details. Then, we discuss context matching, i.e. how we are able to select a patch from the completion image for the input image. In the end, we discuss Graph-cut implementation details which are needed for the completion purposes.

4.1. Scene Matching

As mentioned already that in order to find the matching images from a large repository, we have calculated three features namely Gist, Texture and Colour. Gist describes the shape, structure of images at different scale and orientation. Texture is very important when segments from different images are being combined together. Similarly, the Colour feature helps in finding and then harmonising the colour difference between the input image and the completion image.

We now discuss the implementation details for these features.

4.1.1. **Gist Descriptor**

Gist descriptor focuses on shape and structure of scene itself. The implementation details are as follows. We first pre-process the image by resizing the image to $256 \text{ x } 256$ pixels. Next, we convolve the input image with 32 Gabor filter at 4 scales and 8 orientations, producing 32 feature map of same size as of input image.

In the next step, we divide each map into 16 regions (4x4 grids) and average the feature value of each region then concatenates all 16 region values of all 32 feature map resulting in $16 * 32 * 3 = 1536$ Gist descriptor.

We have calculated Gist descriptor for all images in the repository and compared Gist of query image with the Gist of all images using L2 Distance according to Eq. 6,

$$\mathrm{GistSSD} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \big(\mathrm{ImageGist(i)} - \mathrm{QueryGist(i)}\big)^2 \qquad (6)$$

where

ImageGist is the Gist descriptor of database images,

QueryGist is the Gist Descriptor of Query Image and

n = size of the Gist descriptor.

4.1.2. **Texture Feature**

To compute the texture feature we have used Segment-based Fractal Texture Analysis (SFTA) method. The SFTA algorithm consists of two steps.

1. Two-Threshold Binary Decomposition
2. SFTA extraction Algorithm

4.1.2.1. Two-Threshold Binary Decomposition

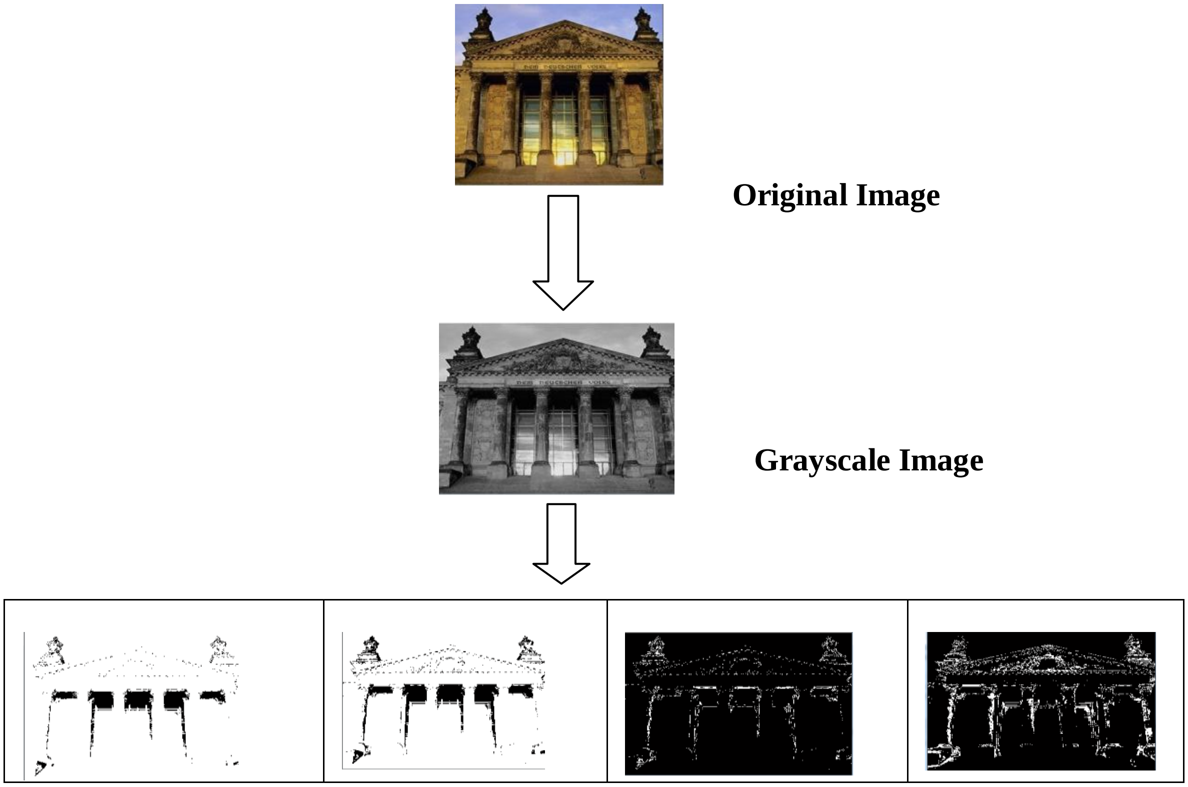
Two-threshold binary decomposition takes grayscale image $Im(x, y)$ as input and outputs set of binary images. The first step of TTBD is computing set of threshold T. This can be done using Otsu algorithm [34] (discussed in Section 3.1.3.1.). Otsu method is used for image thresholding. An image is 2D grayscale and consists of N pixel and L gray levels. We have extended multi-thresholding of an image. We have k-1 thresholds that actually divide the image into K classes, $C_1 = [1 \dots . t], C_2 = [t + 1 \dots \dots \dots t_2] , \dots \dots \dots \dots \dots \dots \dots . , C_M = [C_{M-1} \dots \dots \dots . . L]$. The optimal thresholds $\{t_1^*, t_2^* \dots \dots . . t_M^*]$ are chosen by maximizing the σ_B^2 as follows:

$$\{\mathrm{t}_1^*, \mathrm{t}_2^* \ldots \ldots .. \mathrm{t}_\mathrm{M}^*\} = \mathrm{Arg} \max_{1<t\ldots\ldots<M-1} \{\sigma_\mathrm{B}^2(\mathrm{t}_1, \mathrm{t}_2, \mathrm{t}_3 \ldots \ldots \ldots \mathrm{t}_\mathrm{M}) \qquad (7)$$

The next step of TTBD is to decompose the grayscale image into a set of binary images. This can be done by using pairs of threshold from set of threshold *T*. The criteria of segmentation can be defined as follows:

$$\begin{cases} 1 & \text{if } \mathrm{t}_\mathrm{l} < Im(\mathrm{x}, \mathrm{y}) \leq \mathrm{t}_\mathrm{u} \\ & 0 \text{ otherwise} \end{cases} \qquad (8)$$

Where t_l and t_u are lower and upper thresholds. The set of binary images are obtained by using two thresholds segmentation using all pairs of contiguous threshold values. The reason for using pairs of thresholds is to compute the binary images to segment the objects that are not possible using regular segmentation. The result of TTBD is shown in the Figure 4.1.



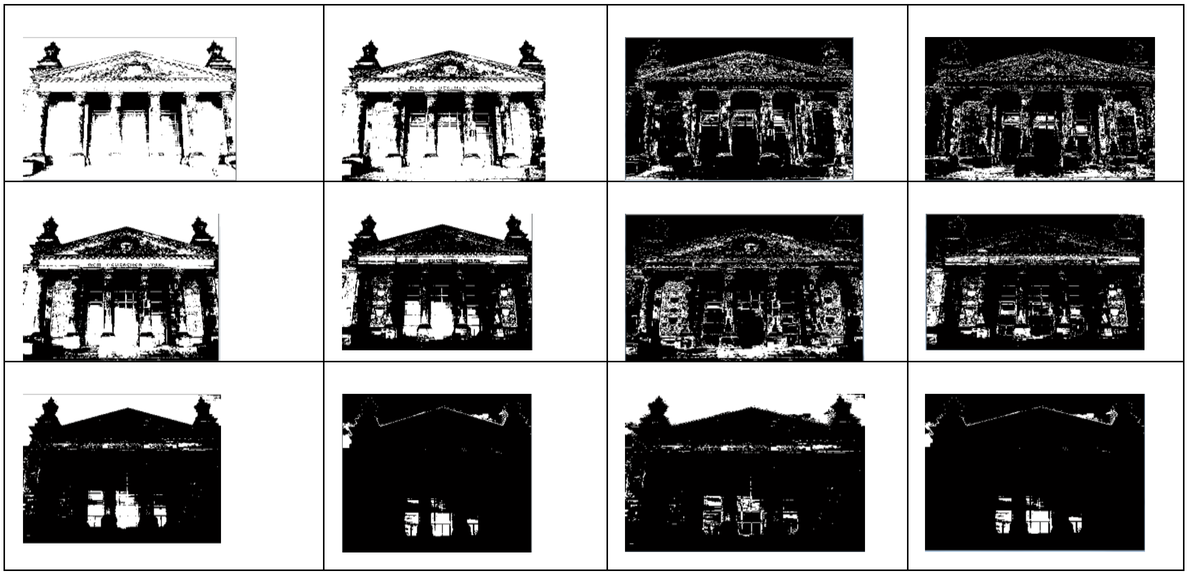


Figure 4.1.Decomposition of input image using TTBD algorithm

4.1.2.2. SFTA Extraction Algorithm

After applying TTBD on input image, SFTA feature vector is constructed that uses binary image's size, mean gray level and boundaries' fractal dimensions. The fractal measurements are used to find the complexity of the object and structures segmented in the input image. For calculating boundaries of region, input image $Im(x,y)$ are represented as border image and $\Delta(x,y)$ is calculated using Eq. 2 (Section 3.1.3.2). Thus, the border is one pixel wide. The border image is shown in Figure 4.2.

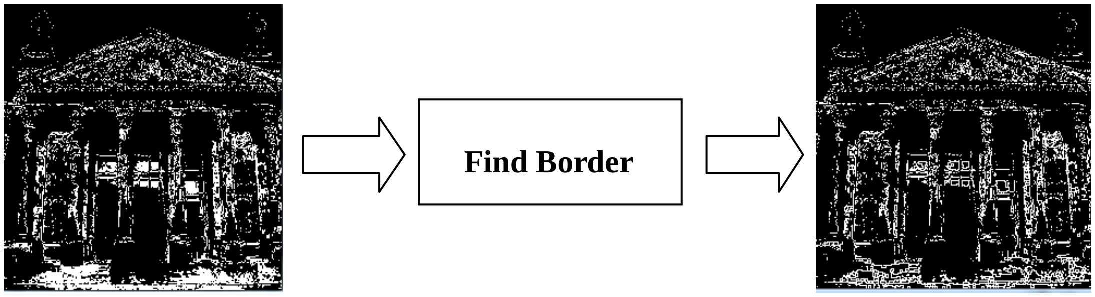


Figure 4.2: Border Image

After calculating the border image, fractal dimensions are calculated using box counting algorithm. The box counting algorithm works as follows:

1. Pad the image with background pixel so that the dimensions would be power of *2*.
2. Set the box size e to the size of image.
3. Compute $N(e)$, which represents number of boxes of size e which contains at least one pixel object.
4. If $e > 1$ then $e = e/2$ and repeat step 3.
5. Compute points $\log\big(N(e)\big)\, X \log(\frac{1}{e})$ and use least square method to fit the line to these points.
6. The returned fractal dimension D would be slope of line.

The mean gray level and pixel count can enhance the information extracted from binary images. Hence the SFTA feature vector dimensionality corresponds to three times the number of binary images.

4.1.3. **Colour Difference**

We have used Lab Colour space for calculating similarity between query image and matched images. We have calculated colour difference between database image and query image in lab colour space using the Eq.9.

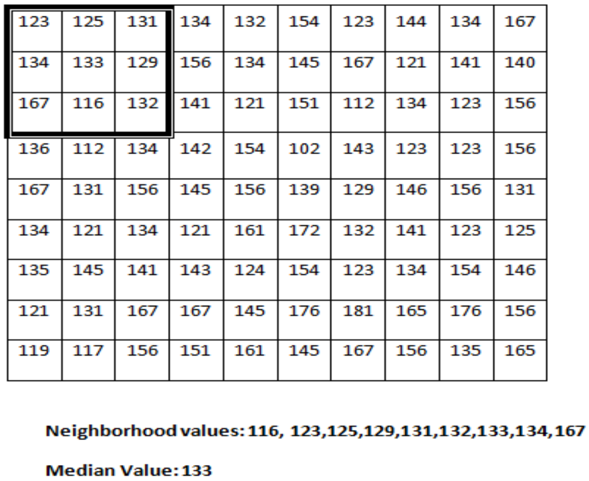
$$\text{ColorDiff} = \sqrt{}(\ \Delta L^2 + \Delta a^2 + \Delta b^2\ ) \tag{9}$$

4.2. Context Matching

In order to find the best patch from matching image, we consider all pixels near holes' boundary as more important as compared to other pixels. We define local context to be all pixels 85 radius around missing region's boundary as shown in Figure 3.5. First matching image is resized to input image at different scale. This context is compared using Eq. 4 against matching images across all translation and scales [.081, 0.90, and 1] as best placement based on minimum SSD error. In addition to SSD error, we have also computed texture similarity in order to measure the affinity between source image and the region that is being filled under local context.

4.2.1. **Texture Similarity**

We compute texture descriptor using median filter. Median filter works like mean filter, it considers all the pixels in the image and checks its surrounding pixels and determines whether it is representative of its surrounding pixel or not. It replaces the value with median of those values. The median of all values is calculated by first sorting the values in numeric order then replacing the pixel being considered with middle pixel value. The example is shown in Figure 4.3



123	125	131	134	132	154	123	144	134	167
134	133	129	156	134	145	167	121	141	140
167	116	132	141	121	151	112	134	123	156
136	112	134	142	154	102	143	123	123	156
167	131	156	145	156	139	129	146	156	131
134	121	134	121	161	172	132	141	123	125
135	145	141	143	124	154	123	134	154	146
121	131	167	167	145	176	181	165	176	156
119	117	156	151	161	145	167	156	135	165

Neighborhood values: 116, 123,125,129,131,132,133,134,167

Median Value: 133

Figure 4.3: Median Filter

After computing the texture descriptor using median filter, both texture descriptors are compared using Sum of square difference.

4.3. Graph-Cut Implementation

We discuss the graph cut in detail.

4.3.1. **Background on Graph**

A graph G= {V, E} consists of set of nodes V and set of edges E. Each node represents a single pixel. For graph cut solutions, two additional nodes source and sink (termed collectively as terminals) are added to the graph as shown in Figure 4.4. Terminals correspond to set of labels that can be assigned to each pixel that is being considered. George et al. were the first to propose the max-flow/min-cut algorithms. In flow graphs, there are two types of edges, N-links and T-links. N-links connects pairs of pixel. Cost of N-links refers to a penalty for discontinuity between pixels. T-links are used to connect pixels with terminal nodes. Cost of T-links refers to a penalty for assigning a specific label to the pixel. In the next section we will describe Min-cut/Max-flow problem briefly.

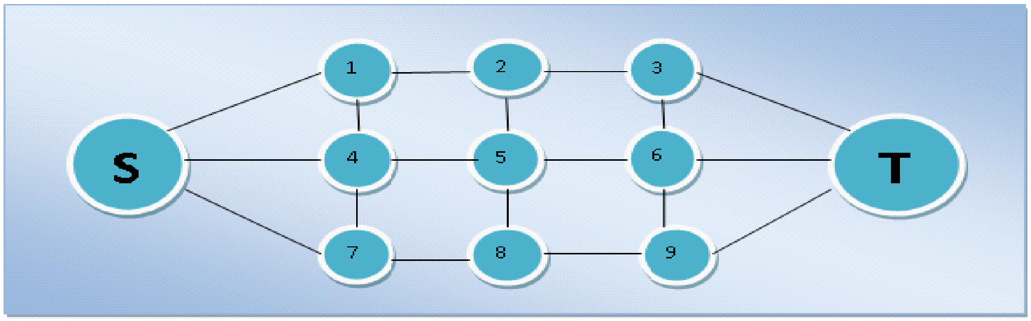


Figure 4.4 : Example of graph with source and sink

4.3.2. **Minimum Cut/Max Flow**

A $s-t$ cut *CT* with two terminals is partition of graph into subset S_1 and S_2 in such a way that s belongs to S_1 and *t* belongs to S_2. Figure 4.5 shows example of a simple graph cut which means that nodes 1, 2, 4 and 7 belong to subset S_1 while nodes 3, 5, 6, 8 and 9 belong to subset S_2. The cost of cut *CT* can be defined as Sum of cost of all the boundary edges. Maximum Flow problem can be solved by finding maximum flow from source *S* to sink *T*. This is done using Ford-Fulkerson algorithm. Note that each subset S_1 and S_2 contain only one terminal which means that a cut corresponds to assigning a label to a pixel.

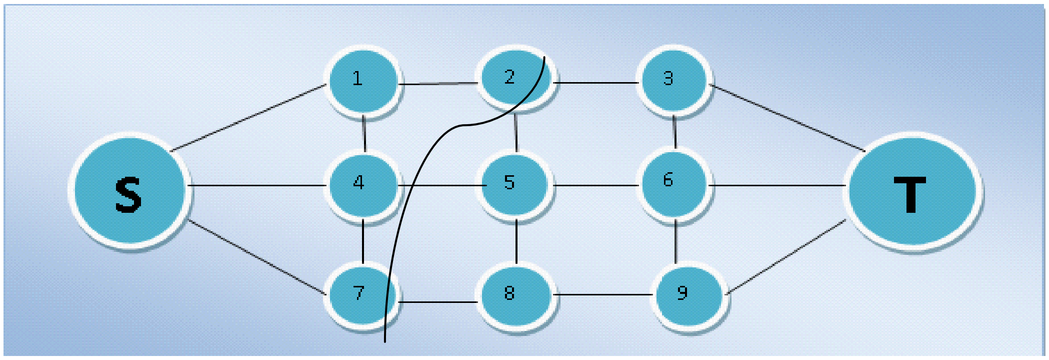


Figure 4.5: Example of cut on graph

4.3.3. **Minimum Cut Procedure**

Figure 4.6 shows basic terminology. There are two non-overlapping search trees S_1 and S_2 connected with parent/root at source S and sink T respectively. In trees S_1 and S_2 all edges from its parent to children are non-saturated. The nodes that are not connected to any terminal node are called free nodes.

The nodes in tree S_1 and S_2 can be passive and active labelled as A and P respectively. An example of search Tree S_1 (Green nodes) and S_2 (red nodes) after growth Stage when path is found from S to T is shown in Figure 4.6. Free nodes are represented by black nodes. The basic idea is that active nodes are allowed to grow the tree by getting new children from the free nodes. Active nodes are those which are on the border while passive nodes are internal nodes that cannot grow the tree because these nodes are being blocked by other nodes. An augmented path, a flow path, is found when an active node in one of the trees S_1 or S_2 finds a node that belongs to other tree. The algorithm has three steps which are discussed below.

1) **Growth Stage**: In this stage search tree S_1 and S_2 grow by finding new children from set of free nodes until one node detects a node that belongs to the other search tree, thus yielding an s-*t* path. The active nodes search for non-saturated adjacent nodes and get new children from set of free nodes. Free nodes are represented as blank nodes in Figure 4.6. These new children become active node. Next, all the neighbors of a given active node are explored and the active nodes then become passive. Growth stage is terminated

when active node finds a node that belongs to other tree and thus, we get an *s-t* path.

2) **Augmentation Stage:** In this stage, a discovered path during growth stage is augmented that actually breaks search tree into a forest. Augmentation stage augments the path by pushing maximum flow so that some edges in the path become saturated. Some of the nodes in this stage become "Stray" as edges linking stray nodes to their parents become saturated (having maximum flow).
3) **Adoption Stage:** In this stage tree S_1 and S_2 are restored. The main purpose of this stage is to restore the single-tree structure of tree S_1 and S_2 with roots S and T. Adoption stage finds parent of "Stray" nodes that should be from same tree S_1 or S_2 and connected with non-saturated edge, if adoption stage fails to find valid parent of "Stray" node then this node is removed and becomes free node This stage terminates when there is no "Stray" node and trees S_1 and S_2 are restored.

4.3.4. **Minimum Cut Algorithm**

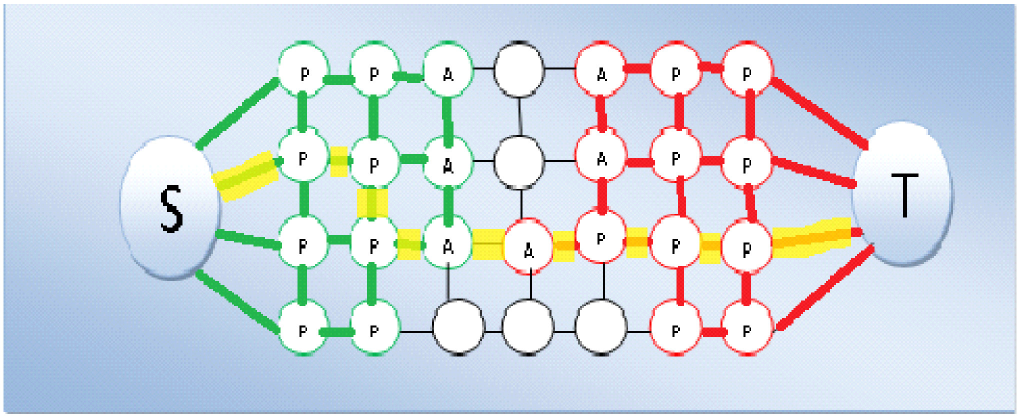


Figure 4.6: Example of search Tree S1 (Green nodes) and S2 (red nodes) after growth Stage when path is found from S to T .Free nodes are represented by black nodes. Active nodes are labeled by A and passive nodes are labeled by P.

An overview of the Minimum cut algorithm is given in Algorithm 4.1.

Algorithm 4.1: The General Structure of Algorithm

```
Initialize:    S1= {S}    S2= {T}   A={S, T}   Stray=ϕ
While True
Grow S1 or S2 to find Augmenting Path “P” from S to T
If P =ϕ terminates
Augment on PATH
Restore Stray
End while
```

It is helpful to store content of search tree S_1 and S_2 using a flag that actually describes the relationship of each node q as,

$$\text{flag}(\text{q}) = \begin{cases} \text{S1 if q} \in \text{S1} \\ \text{S2 if q } \in \text{S2} \\ \emptyset \text{ if q } \in \text{Freeode} \end{cases}$$

If node q belongs to any tree then its content would be stored as $\text{Parent}(\text{q})$. It is important to mention that the roots of the search trees say source and sink. Note that all these all these edges should be non-saturated for node q to be a valid parent for its child r.

4.3.4.1. Growth Stage Algorithm

At this stage, Active nodes get new children from free nodes. The Growth stage algorithm is shown in Algorithm 4.2.

Algorithm 4.2:Growth Stage Implementation

```
While $A \neq \emptyset$
          Select an Active node $q \in A$
          For every neighbor $r$ such that $\text{Res\_Cap}(\text{q} \to \text{r}) > 0$
                 if $\text{flag}(\text{r}) = \emptyset$ Then add r to tree as an Active node
                        $\text{flag}(\text{r}) := \text{flag}(\text{q})$
                        $\text{Parent}(\text{r}) \coloneqq \text{q}$
                        $\text{A} \coloneqq \text{A} \cup \{\text{r}\}$
                 If $\text{flag}(\text{r}) \neq \emptyset$ and $\text{flag}(\text{r}) \neq \text{flag}(\text{q})$
```

```
                Return P= $\text{path}_{S\to T}$
        End for
        Remove q from A
    End while
    Return P $= \emptyset$
```

4.3.4.2. Augmentation Stage Algorithm

During growth stage, an *s-t* path has been found. The augmentation stage takes the path as an input and augments that path by pushing maximum flow from S to T. Details of Augmentation stage are given in Algorithm 4.3.

Algorithm 4.3: Augmentation Stage Implementation

```
Find the bottleneck capacity C on path P
Update the residual graph $G_f$ by pushing flow f through path P
For each edge $(q, r)$ in P that becomes saturated
    If $\text{flag}(q) = \text{flag}(r) = \text{S1 then set}$
        $\text{Parent}(r) := \emptyset$ and
            $\text{Stray} \coloneqq \text{Stray} \cup \{r\}$
    If $\text{flag}(q) = \text{flag}(r) = \text{S2 then set}$
            $\text{Parent}(q) \coloneqq \emptyset$ and
            $\text{Stray} \coloneqq \text{Stray} \cup \{q\}$
End   For
```

4.3.4.3. Adaptation Stage Algorithm

In this stage, all stray nodes are processed. Each node *q* being processed finds a new parent within the same tree. If node *q* finds a valid parent that node will remain in the same tree but with different parent and if it does not find a valid parent then this node will be removed from the list of stray nodes and becomes a free node and the child nodes become stray nodes. The steps in the implementation of adoption stage are shown in Algorithm 4.4.

Algorithm 4.4: Adaptation Stage Implementation

```
While  stray $\neq \emptyset$
    Select a Stray node q $\in$ Stray and remove it from stray
    Process q
End while
```

The operation "Process q" contains following steps. First, find new valid parent of q from all its neighbours. A parent *r* is a valid parent if $\mathrm{flag(q)} = \mathrm{flag(r)}, \mathrm{Res\_Cap(r \rightarrow q)} > 0$ and origin of *r* should be source or sink because during adoption stage a node may come from Stray in the search tree S_1 and S_2. If node q catches the legal parent then we assign as $\mathrm{parent(q) = r}$ and status of node q remain same but if it does not find valid parent then node q is treated as a free node. Consequently, following operations are performed.

1) Scan all neighbour of node q such that $\mathrm{flag(r) = flag(q)}$
    i. if $\mathrm{Res\_Cap} > 0$ $add\ r\ to\ the\ active\ set\ A$
    ii. if $\mathrm{Parent(r) = q}$ add r to the set of Stray and set $\mathrm{Parent(r)} := \emptyset$
2) $\mathrm{flag(q)} \coloneqq \emptyset$, $\mathrm{A} \coloneqq \mathrm{A} - \{\mathrm{q}\}$

We have used the above algorithm for image composition that actually decides which pixel will come from original image or matching image that helps us for image completion in such a way that the user does not detect any modification in the image with naked eye. We have shown graph cut mask in Figure 3.5.

We have mask image in which black corresponds to 0 and white colour corresponds to 1. All pixels with label 0 should come from original image and the pixels with label 1 come from the matching image.

4.4. Poisson Equation for Removal of Visible Seam

To solve Poisson blending, some terminologies may help to understand as shown in Figure 4.7 where v represents gradient of region in an image as a vector while g corresponds to source region we need to paste. The notation f represents unknown function in domain Ω and f^* corresponds to a known function that exists in domain S. The Ω corresponds to the area where source image g is placed on domain S where S corresponds to target background. The $\partial\Omega$ represents the boundary between the source and the target image. The simplest interpolate f of f^* over Ω is membrane interpolate which can be defined as minimization problem as follow.

$$\min \iint_{\Omega} |\nabla \mathrm{f}|^2 \text{ with } \mathrm{f}|\,\partial\Omega = \mathrm{f}^*|\,\partial\Omega \qquad (10)$$

where $\nabla = \partial x + \partial y$ is the gradient operator. The minimiser must satisfy the Euler-Langrage equation as:

$$\Delta \mathrm{f} = 0 \text{ over } \Omega \text{ with } \mathrm{f}|\,\partial\Omega = \mathrm{f}^*|\,\partial\Omega \qquad (11)$$

$\Delta . = {}^{\partial^2}/_{\partial x^2} + {}^{\partial\,2}/_{\partial y^2}$ is Laplacian operator. For image editing this method produces blurred interpolates. This problem can be solved by introducing hard constraints in the form of guidance vector field. The guidance field is a vector field used in minimization problem as Eq.10. The solution of this problem is the Poisson equation with Dirichlet boundary conditions.

$$\min \iint_{\Omega} |\nabla \mathrm{f} - \mathrm{v}|^2 \text{ with } \mathrm{f}|\,\partial\Omega = \mathrm{f}^*|\,\partial\Omega \qquad (12)$$

where

$div\mathrm{v} = {}^{\partial u}/_{\partial x} + {}^{\partial v}/_{\partial y}$ is the divergence property of vector field $\mathrm{v} = (u, v)$.

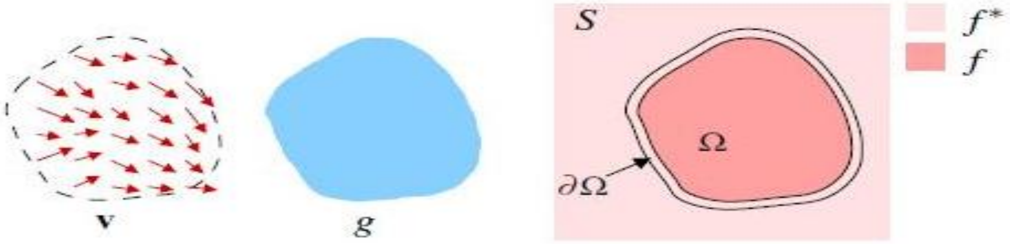


Figure 4.7: Guided Interpolation Notations (image Taken from[27])

4.4.1. **Discrete solution and implementation with sparse matrix**

To solve Poisson blending algorithm, we need to construct system of equations. Given target image A and source image B and a new image C (where C should be improved version of image B that blends in with image A better). First the boundary constraints, the pixels on boundary of C should be same pixel of A on that boundary so that we can match those pixels outside mask and blend them. Mathematically,

$$C_{(x,y)} = A_{(x,y)} \forall (x, y) \in \partial B \tag{13}$$

So we already know the solution of pixel at the boundary, we need to solve the value of interior pixel of C. The gradient of interior pixel of C should be equal to gradient of interior pixel of B. The gradient of image at a given point can be calculated as sum of difference between pixels and its entire neighbour. Mathematically it can be written Eq.14.

$$\left|\nabla B_{(x,y)}\right| = 4B(x, y) - B(x - 1, y) - B(x + 1, y) - B(x, y - 1) - B(x, y + 1) \tag{14}$$

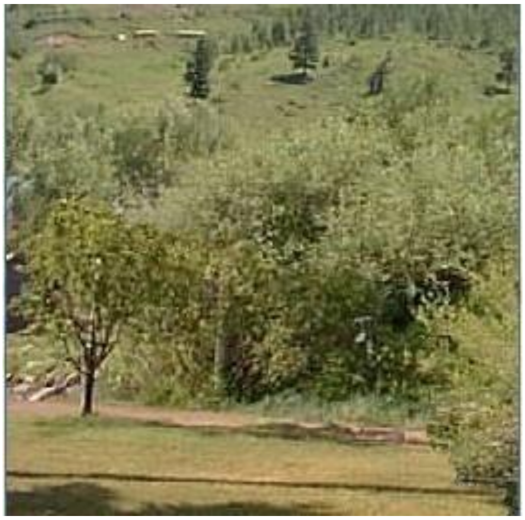
If one of the neighbour belong to a boundary then its value would be fixed, on the other hand if the one of the neighbour is outside of mask region it must be excluded. This is summarized for all cases for every pixel in C using following Eq.15.

$$
\begin{aligned}
|N|H(x,y) - \sum_{(dx,dy)+(x,y)\in\Omega} C(x+dx,y+dy) \\
- \sum_{(dx,dy)+(x,y)\in\partial\Omega} A(x+dx,y+dy) \\
= \sum_{(dx,dy)+(x,y)\in\Omega\cup\partial\Omega} (B(x=dx,y+dy) - B(x,y)
\end{aligned}
\tag{15}
$$

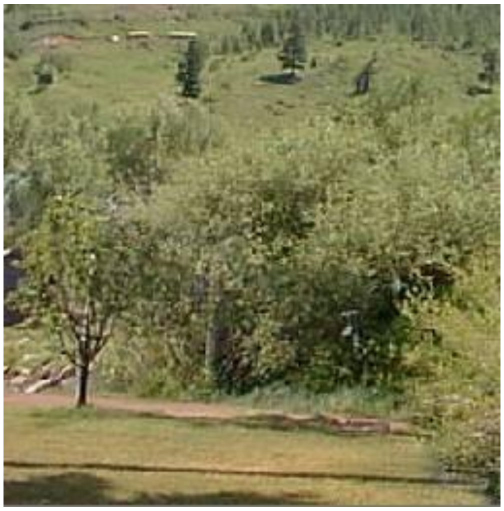
We need to solve this equation for every pixel in C, so this can be done using system of equation as

$$Ax = b \tag{16}$$

where *A* is a sparse matrix NxN where *N* is the number of pixel to be copied, *b* is the guiding gradient plus the sum of all non-masked neighbour pixels in target image A. The non-masked neighbour pixel defines value of pixels at the boundary of mask which are blended across the mask area. If the guiding gradient is zero, we are just solving the Laplace equation and the values at the boundary are blended smoothly. We can find values of x which are values of pixels inside the target image by solving Eq.16.



Before Blending



After Blending

4.5. Summary

We have discussed the implementation techniques for scene completion. We have also discussed the algorithms in detail that are used to find the maximum flow for solving graph cut problem.

Chapter 5

Evaluation

CHAPTER 5. EVALUATION AND ANALYSIS

In this chapter, we give the evaluation of this work. We first discuss the system setup. Then, we discuss the dataset that is used for finding matching images for scene completion. Finally we discuss our experimental results and evaluation.

5.1. System setup

Our system is implemented using Matlab, on a Dell Machine with Intel® Pentium® 2.10GHz processor.

5.2. Libraries

We have used maxflow-v3.01 library that computes the max-flow/min-cut on graph that is compiled on Visual studio 2010. It implements the Boykov-kolmogorov algorithm, which tends to be fast in computer vision problems.

5.3. Datasets

We evaluated our approach on publically available dataset [43]. We have constructed our image collection from [43] that contain different categories i.e. forest, opencountry, street, insidecity, coast and tall building. Our data repository contains 300 images for each category. In addition, we have also included results of Hays in our dataset in order to validate the results from Hays et al.

5.1. Evaluation

5.1.1. **Image Preparation**

We have created input image by removing unwanted objects or portion from an image

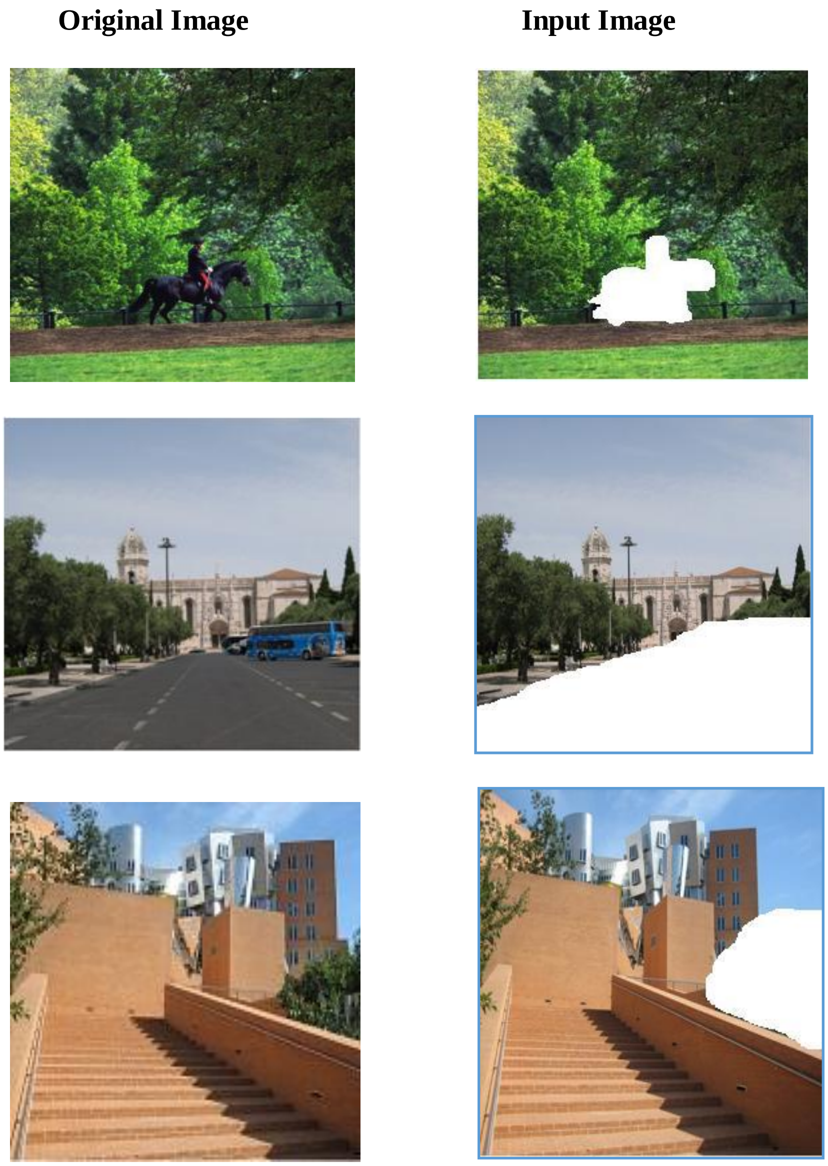


Figure 5.1: Create Input Images

5.1.2. **Top-1 Image**

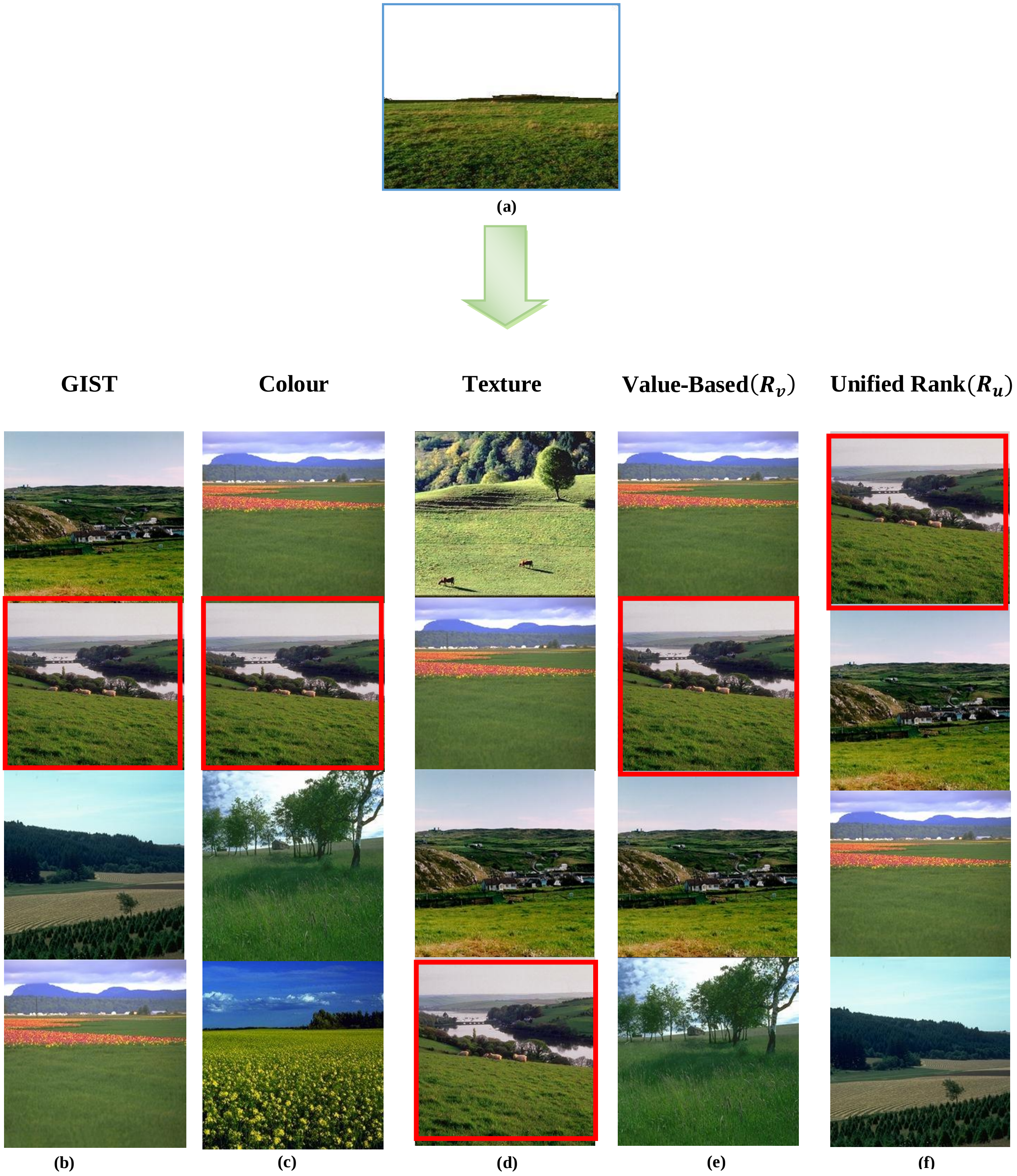


Figure 5.2: (a) Input Image (b) Image Retrieval based on Gist (c) Image Retrieval based on colour (d) Image Retrieval based on Texture (e) Value-based Image Retrieval (f) Unified Ranked based Image Retrieval

We evaluated our proposed ranking scheme on a sample set. First, we show image retrieval based on single feature Gist, Texture and colour as shown in Figure 5.2. After intensive experiments based on single feature, we assigned weights to each feature and retrieve images based on combined weighted feature called value-based(R_v) image retrieval. The above result shows that this retrieval approach is biased toward the actual value. In this particular case, Gist and colour feature dominate over texture feature. In order to evaluate our proposed unified ranking scheme that satisfies value-invariance property and is able to handle the more than two features. We have an image rank for each feature and compute the final rank based on majority scores called unified Rank(R_u) image retrieval.

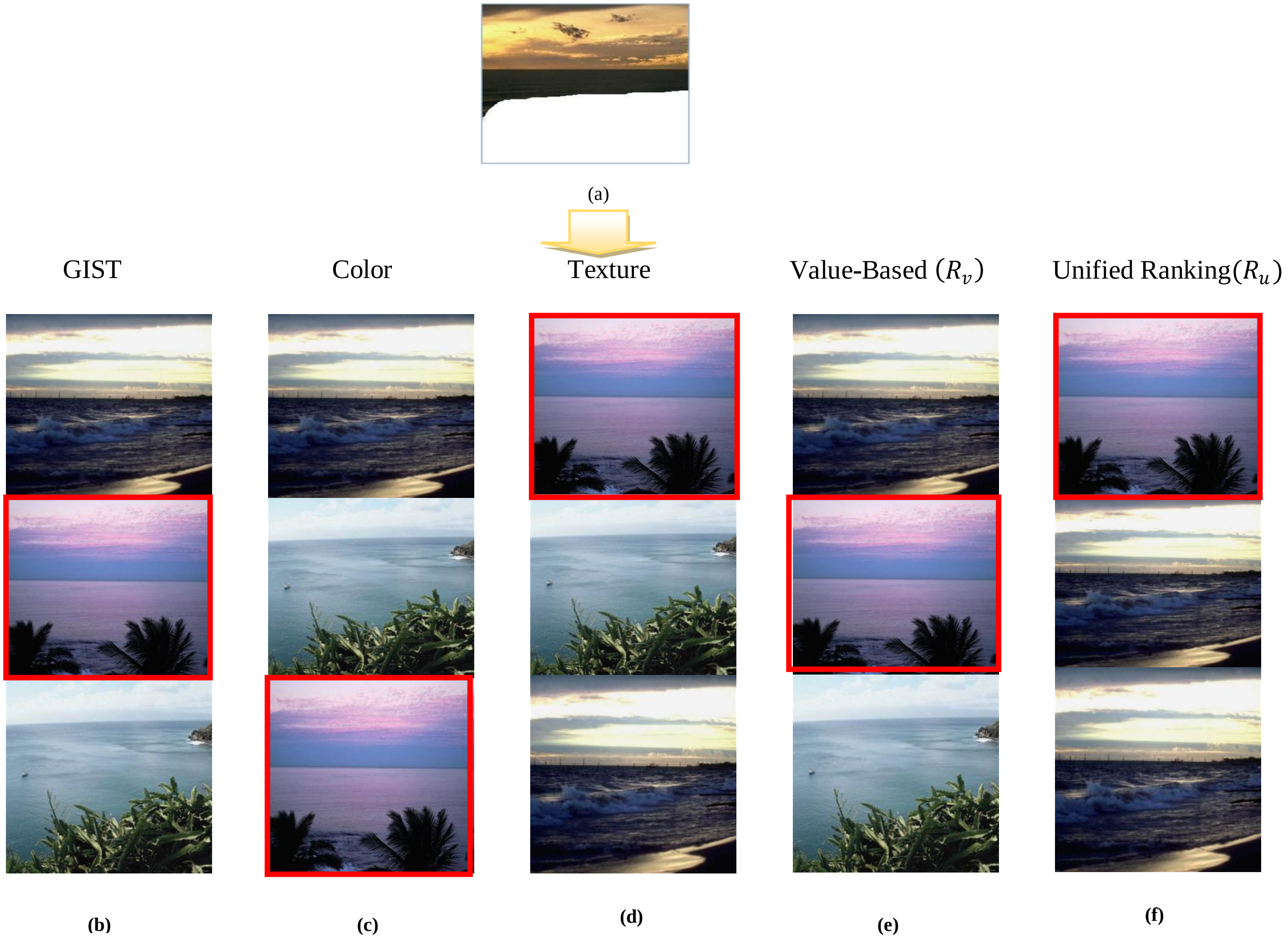


Figure 5.3: (a) Input Image (b) Image Retrieval based on Gist (c) Image Retrieval based on colour (d) Image Retrieval based on Texture (e) Value-based Image Retrieval (f) Unified Ranked based Image Retrieval

5.1.3. **Scene Completion**

5.1.3.1. Validation

In this work, we validate the results of Hays et al. using same matching image and the results are shown in Figure 5.4

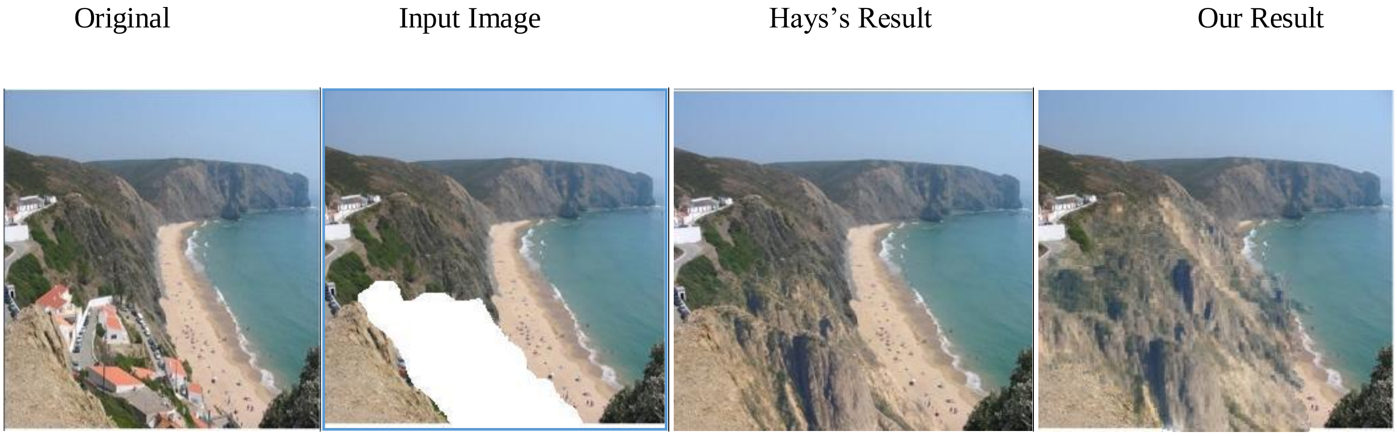
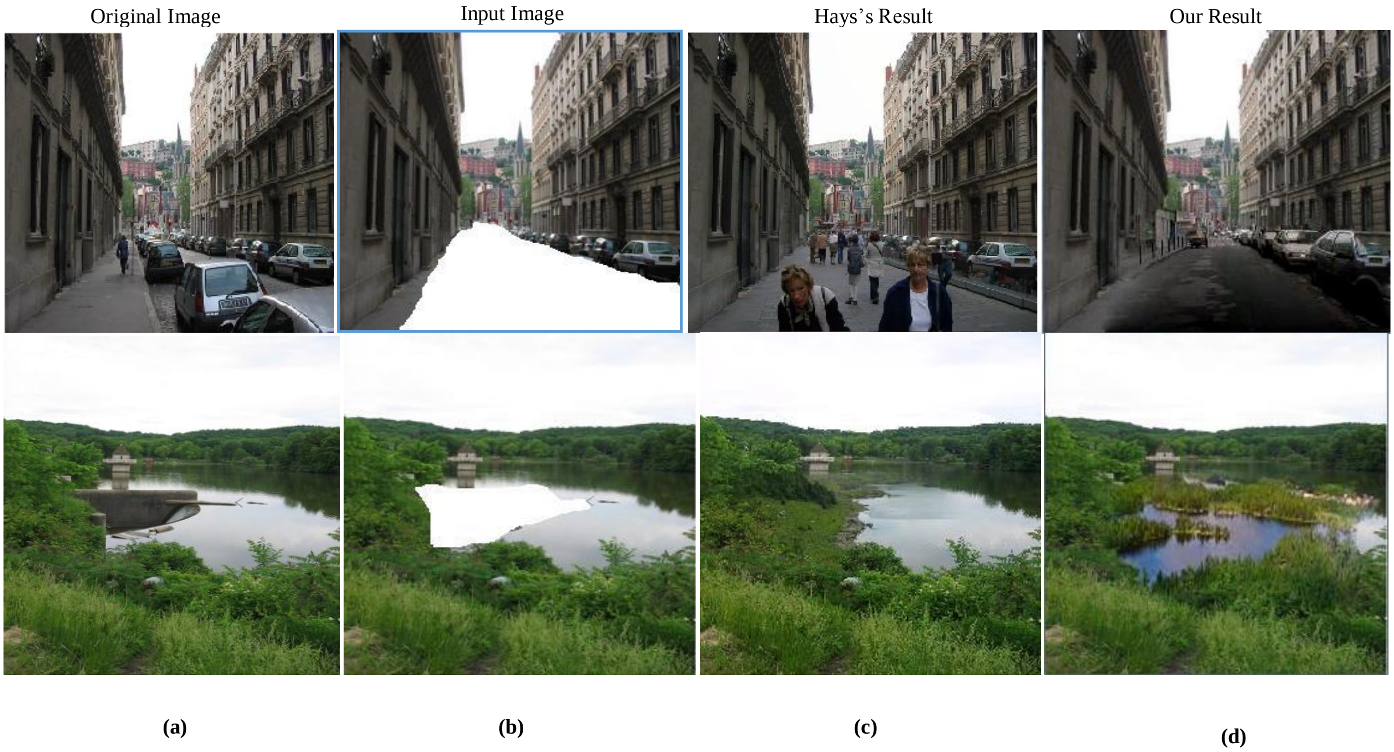


Figure 5.4: Validation of results using same matching Images.

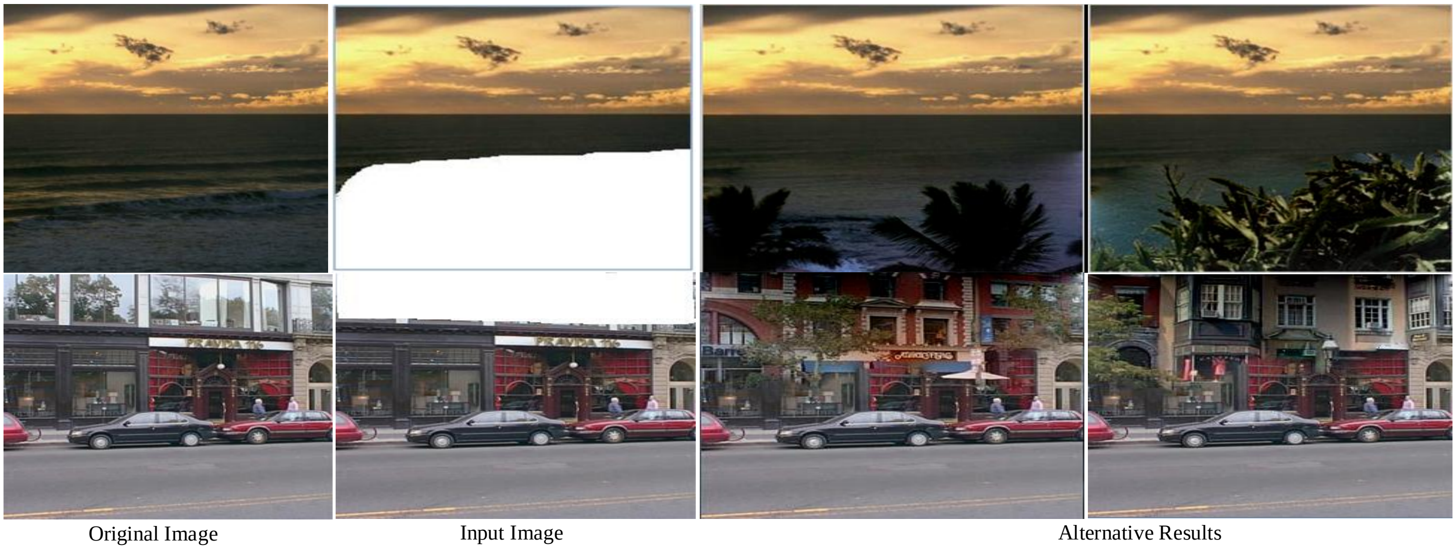
In addition, we have also tried to validate the results of Hays et al. using different matching image retrieved from our data repository and validate the previous results. In Figure 5.5, we have shown the actual result of Hays et al. and then our result.


Original Image Input Image Hays's Result Our Result

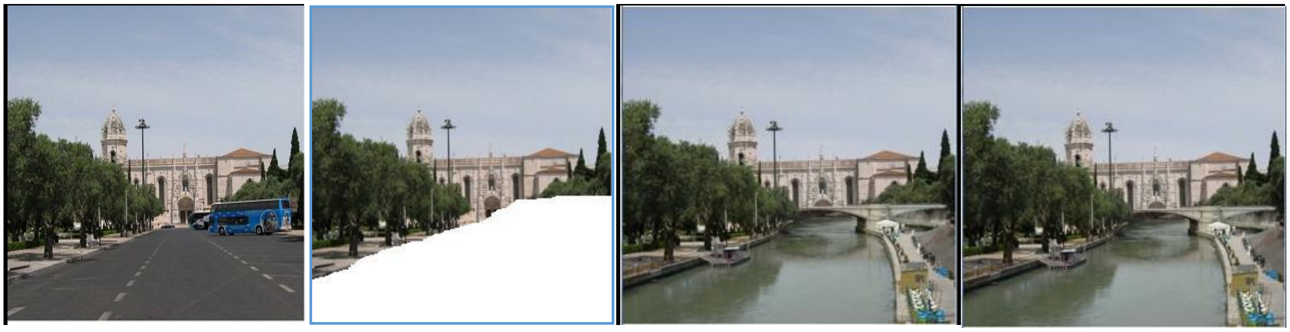
(a) (b) (c) (d)

Figure 5.5: (a) Original Image (b) Input Image (c) Hays 'Result (d) Our Result

We have shown some alternative results using different matching images against single query image.


Original Image Input Image Alternative Results

Data intensive scene completion performs well if we have multiple photographs of same physical scene, in order to verify our approach we have used the results of Hays et al. for completing the query image. The result is shown in Figure 5.6.

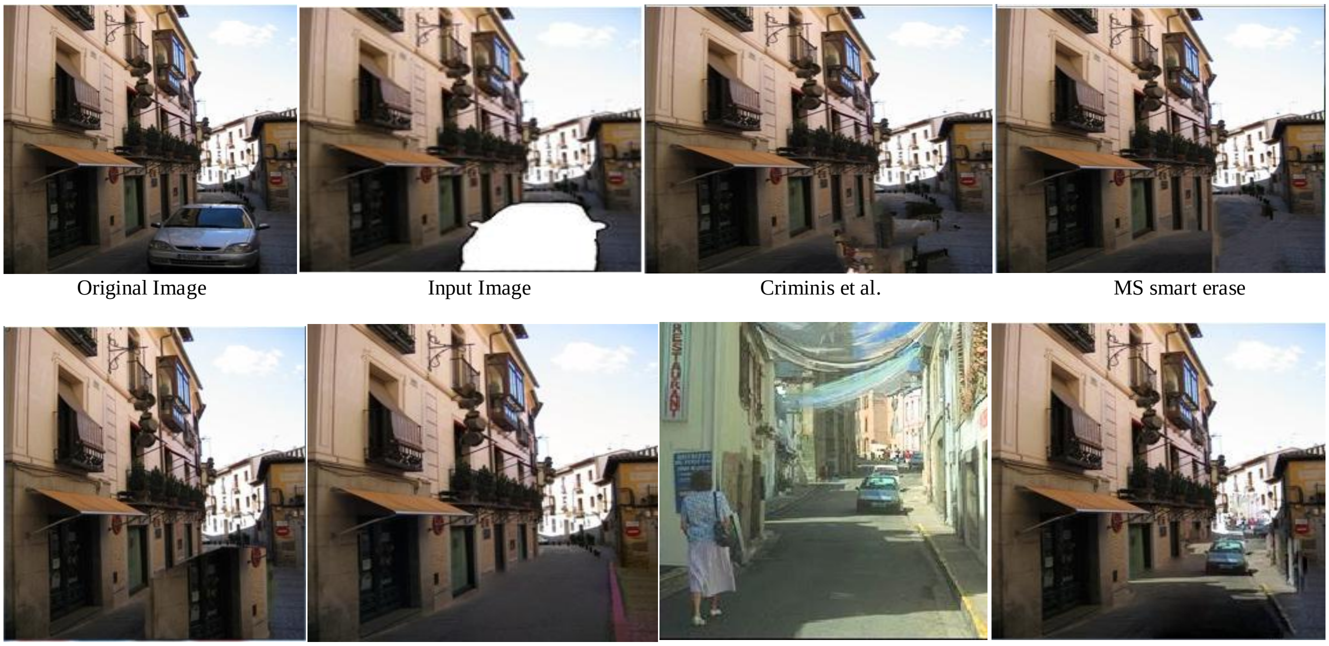


(a) (b) (c) (d)

Figure 5.6: (a) Original Image (b) Input Image (c) Matching Image (d) Output

5.1.3.2. Improvements

We have also compared our result with the recent techniques of scene completion in literature and we have more convincing results as compared to other techniques as shown in Figure 5.7



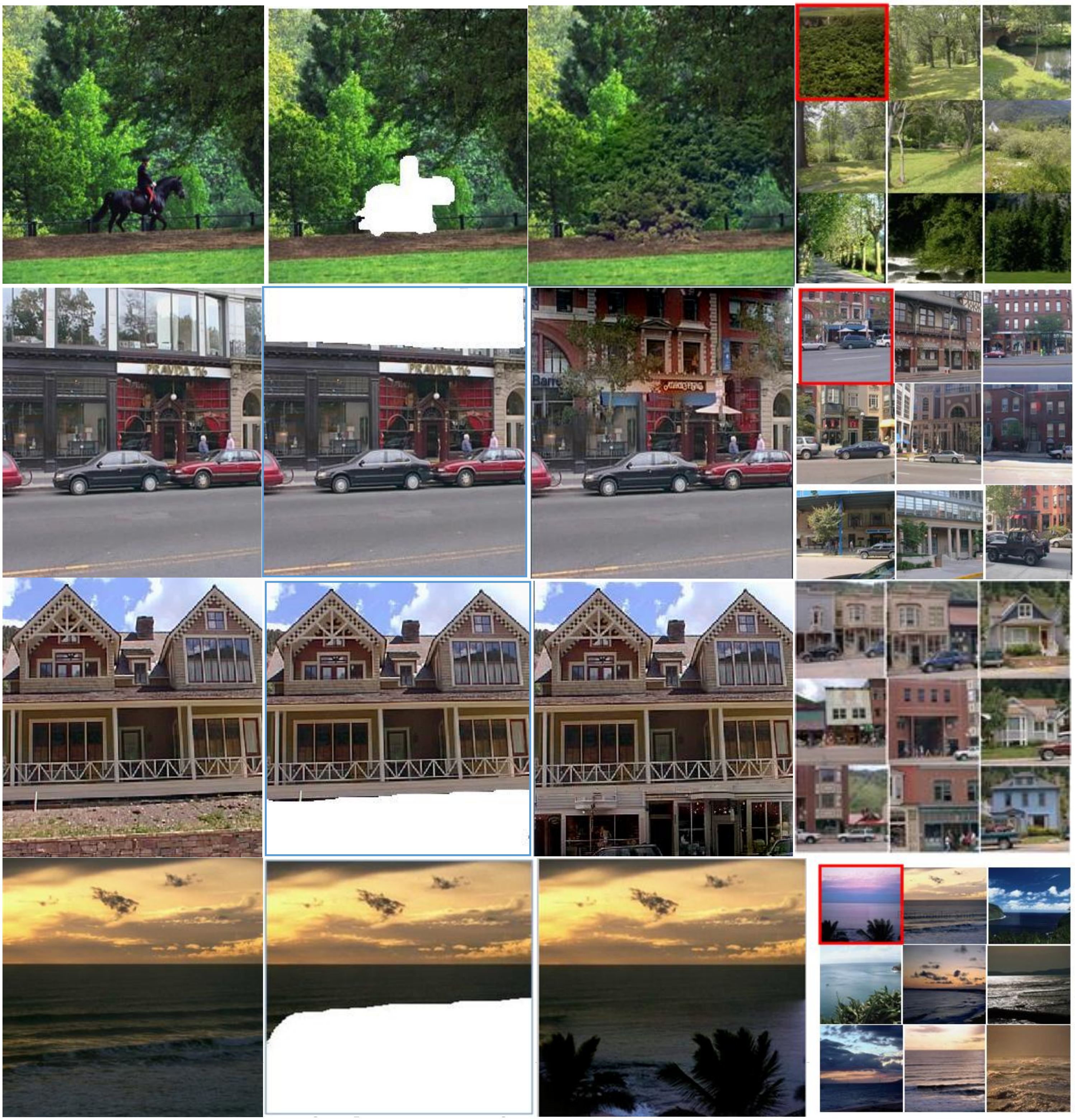
Original Image Input Image Criminis et al. MS smart erase

Wilczkowiak et al. Hays et al. Matching Image Our Result

Figure 5.7: Results from various image completion algorithms

5.1.3.3. Further Results

Furthermore, we have shown our results in Figure 5.8.



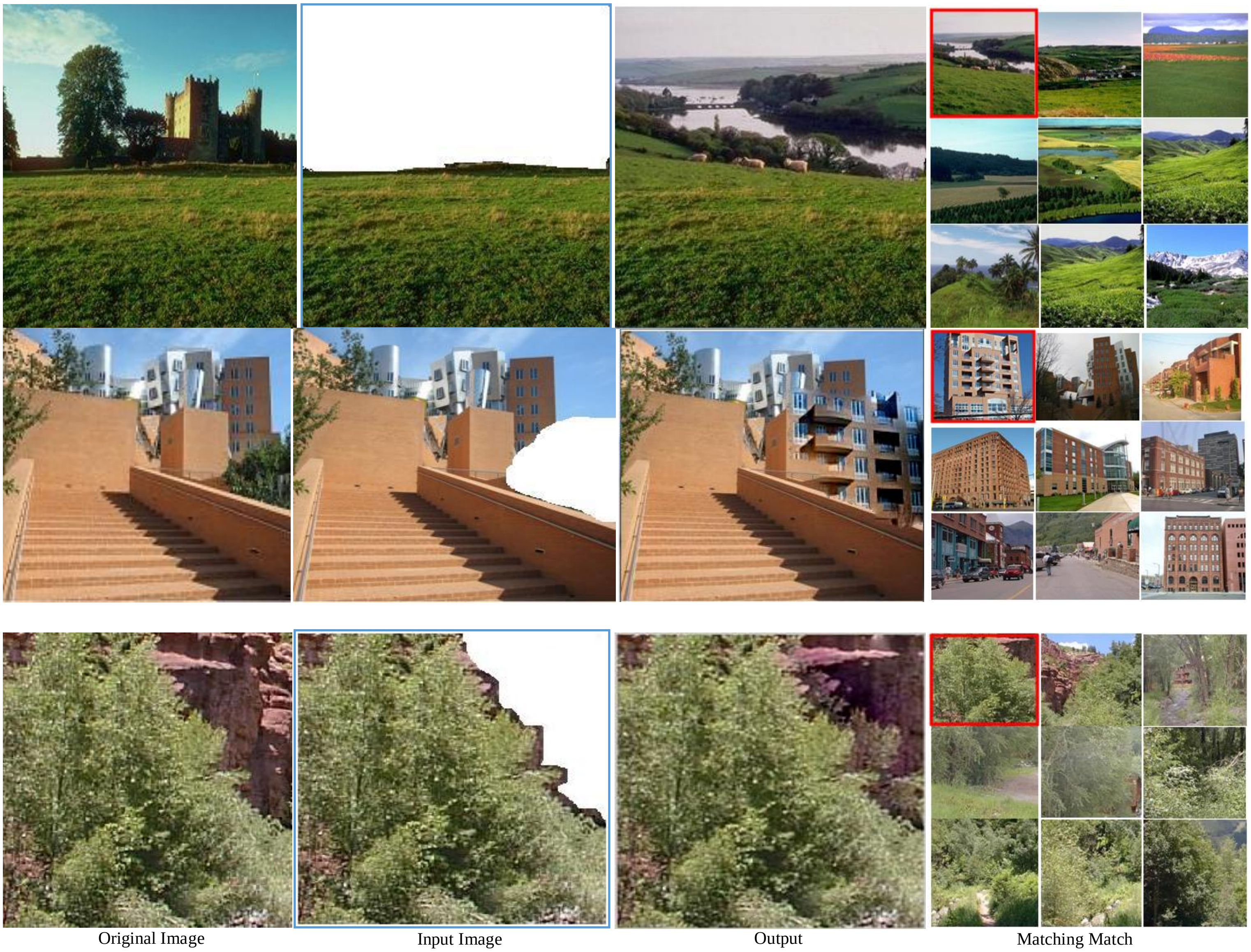


Figure 5.8: (a) Original Image (b) Input Image (c) Output (d) Matching Image

5.2. Conclusion

In this chapter, we have evaluated our proposed ranking scheme and validated the results of Hays et al. For evaluating our approach, we have included the results of Hays et al. in our data repository. Data Intensive Scene completion performs well if we have multiple photograph of same physical scene. Furthermore, for some of the un-seen situations we have shown results which are better than the results from literature.

Chapter 6

Conclusion and Future work

CHAPTER 6. CONCLUSIONS AND FUTURE WORK

We have proposed a scene completion scheme named Data Intensive scene completion (DISC). In this work, we have validated the results from Hays et al. Further, we have been able to address the research problems presented in Section 1.2. We observe that tasks such as image retrieval are rather complicated because of large variation in everyday photographs. It is very difficult to capture a large variation in images. Hence, selection of good features is also a tricky task. For image retrieval, taking clues from literature we have used Gist, Texture and colour feature. After intensive experimentation, we observed the significance of each feature and assigned weights to each feature. It appears as if Gist is a more significant feature as compared to others because Gist is able to extract images that are semantically valid to query image. Traditional retrieval approaches use minimum distance measure that is biased towards distance value. Our proposed unified ranking scheme selects suitable top-1 matching candidate that is semantically valid to query image and produces good results.

6.1. Conclusions

In this work, we have proposed a Unified ranking scheme that is used to find the top-1 best matching candidate. In addition, we have computed texture feature for retrieving the image that are texturally similar to query image in order to overcome the texture mismatch problem. We have used graph cut techniques for image composition and compare result obtained by using Ford Fulkerson Algorithm. In summary,

(1) We have validated the results from Hays et al.

(2) We have addressed the texture mismatch issue as was identified in Hays et al. work.

(3) We have proposed a unified ranking scheme that can automatically select the best completion image from an image repository.

6.2. Future Work

In future, we plan to extend our work by using our proposed Unified ranking scheme on large cloud based repository. A limitation of the proposed algorithm is the inability to recognize the objects in scene completion. In future, we plan to investigate object recognition techniques to solve this problem. A comprehensive survey and study on the usefulness of features that could be used for image retrieval is also an interesting future work. For example, TinyImage feature can be an interesting feature to look at.

REFERENCES

[1] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester, "Image inpainting," in *27th Annual Conference on Computer Graphics and Interactive Techniques*, 2000, pp. 417-424.

[2] Antonio Criminisi, Patrick, and Kentaro Toyama, "Region filling and object removal by exemplar-based image inpainting," *Image Processing, IEEE Transactions on*, vol. 13, no. 9, pp. 1200-1212, 2004.

[3] James Hays and Alexei Efros, "Scene completion using millions of photographs," *ACM Transactions on Graphics* , vol. 26, no. 3, p. 4, 2007.

[4] Aude Oliva and Antonio Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope.," *International journal of computer vision*, vol. 42, pp. 145-175, 2001.

[5] Alceu Ferraz and Humpire-Mamani, Gabriel and Traina, Agma Juci Machado Costa, "An efficient algorithm for fractal analysis of textures," in *25th SIBGRAPI Conference on Graphics, Patterns and Images.*, Ouro Preto, Brazil, 2012, pp. 39-46.

[6] Yuri Boykov and Vladimir 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.

[7] Alexei Efros, Thomas Leung, and others, "Texture synthesis by non-parametric sampling," in *Seventh IEEE International Conference on Computer Vision.*, Kerkyra, Greece, 1999, pp. 1033-1038.

[8] Marta Wilczkowiak, Gabriel J Brostow, Ben Tordoff, and Roberto Cipolla, "Hole filling through photomontage," in *British Machine Vision Conference 2005*, 2005.

[9] K Sangeetha, P Sengottuvelan, and E Balamurugan, "Combined structure and texture image

inpainting algorithm for natural scene image completion," *Journal of Information Engineering and Applications*, vol. 1, pp. 7-12, 2011.

[10] Babu M Mehtre, Mohan S Kankanhalli, and Wing Foon Lee, "Shape measures for content based image retrieval: a comparison," *Information Processing \& Management*, vol. 33, pp. 319-337, 1997.

[11] Elham Seifossadat, Niloofar Gheissari, and Ali Fanian, "Indoor Scene Recognition Using Local Semantic Concepts," *An International Journal on Advances in Computer Science,* vol. 4, pp. 29-35, 2015.

[12] Serge and Carson, Chad and Greenspan, Hayit and Malik, Jitendra Belongie, "Color-and texture-based image segmentation using EM and its application to content-based image retrieval," in *Sixth International Conference on Computer Vision.*, Bombay, India, 1998, pp. 675--682.

[13] Robert Haralick, Karthikeyan Shanmugam, and Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man and Cybernetics*, pp. 610-621, 1973.

[14] Manik Varma and Andrew Zisserman, "A statistical approach to material classification using image patch exemplars," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 2032-2047, 2009.

[15] Bangalore S Manjunath and Wei-Ying Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions onPattern Analysis and Machine Intelligence*, vol. 18, pp. 837-842, 1996.

[16] Xuewen Wang, Xiaoqing Ding, and Changsong Liu, "Gabor filters-based feature extraction for character recognition," *Pattern Recognition*, vol. 38, pp. 369--379, 2005.

[17] Dengsheng Zhang, Aylwin Wong, Maria Indrawan, and Guojun Lu, "Content-based image retrieval using Gabor texture features," in *IEEE Pacific-Rim Conference on Multimedia, University of Sydney, Australia*, 2000.

[18] Alceu Ferraz Costa, Joe Tekli, and Agma Juci Machado Traina, "Fast fractal stack: fractal analysis of computed tomography scans of the lung," in *International ACM workshop on Medical multimedia Analysis and Retrieval*, Scottsdale, USA, 2011, pp. 13-18.

[19] Yan Qiu Chen, Mark S Nixon, and David W Thomas, "Statistical geometrical features for texture classification," *Pattern Recognition*, vol. 28, pp. 537-552, 1995.

[20] Anil K Jain, Jung-Eun Lee, Rong Jin, and Nicholas Gregg, "Content-based image retrieval: An application to tattoo images," in *16th IEEE International Conference on Image Processing (ICIP).*, Cairo, Egypt, 2009, pp. 2745-2748.

[21] Olga Veksler, "Image segmentation by nested cuts," in *IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, SC, USA, 2000, pp. 339-344.

[22] DM Greig, BT Porteous, and Allan H Seheult, "Exact maximum a posteriori estimation for binary images," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 271-279, 1989.

[23] Vivek Kwatra, Arno Sch, Irfan Essa, Greg Turk, and Aaron Bobick, "Graphcut textures: image and video synthesis using graph cuts," in *ACM Transactions on Graphics (ToG)*, vol. 22, 2003, pp. 277-286.

[24] Yuri Y Boykov and Marie-Pierre Jolly, "Interactive graph cuts for optimal boundary \& region segmentation of objects in ND images," in *Eighth IEEE International Conference on Computer Vision (ICCV)*, Vancouver, BC, Canada, 2001, pp. 105-112.

[25] Vladimir Kolmogorov and Ramin Zabih, "Computing visual correspondence with occlusions using graph cuts," in *Eighth IEEE International Conference on Computer Vision (ICCV)*, Vancouver, BC, Canada , 2001, pp. 508--515.

[26] Son Tran and Larry Davis, "3D surface reconstruction using graph cuts with surface," in *European Conference on Computer Vision*, 2006, pp. 219-231.

[27] Patrick, Michel Gangnet, and Andrew Blake, "Poisson image editing," in *ACM Transactions*

on Graphics (TOG), vol. 22, 2003, pp. 313-318.

[28] Peter J Burt and Edward H Adelson, "A multiresolution spline with application to image mosaics," *ACM Transactions on Graphics (TOG)*, vol. 2, pp. 217-236, 1983.

[29] Baudisch Patrick and Gutwin Carl, "Multiblending: displaying overlapping windows simultaneously without the drawbacks of alpha blending," in *SIGCHI conference on Human Factors in Computing Systems*, Vienna, Austria, 2004, pp. 367-374.

[30] Graham Cormode, Feifei Li, and Ke Yi, "Semantics of ranking queries for probabilistic data and expected ranks," in *IEEE 25th International Conference on Data Engineering.*, 2009, pp. 305-316.

[31] Aude Oliva, "Gist of the scene," *Neurobiology of Attention*, vol. 696, no. 64, pp. 251-258, 2005.

[32] Philippe G Schyns and Aude Oliva, "From blobs to boundary edges: Evidence for time-and spatial-scale-dependent scene recognition," *Psychological science*, vol. 5, no. 4, pp. 195-200, 1994.

[33] Vivek Singh Rathore, Messala Sudhir Kumar, and Ashwini Verma, "Colour based image segmentation using L* a* b* colour space based on genetic algorithm," *IJETAE*, vol. 2, no. 6, pp. 156-162, 2012.

[34] Ping-Sung Liao, Tse-Sheng Chen, and Pau-Choo Chung, "A fast algorithm for multilevel thresholding," *J. Inf. Sci. Eng.*, vol. 17, pp. 713-727, 2001.

[35] Yuri Boykov and Gareth Funka-Lea, "Optimal object extraction via constrained graph-cuts," *International Journal of Computer Vision (IJCV)*, 2004.

[36] Hiroshi Ishikawa and Davi Geiger, "Segmentation by grouping junctions," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition.*, 1998, pp. 125-131.

[37] Junmo Kim et al., "Incorporating spatial priors into an information theoretic approach for

MRI data analysis," in *Medical Image Computing and Computer-Assisted Intervention--MICCAI* , Pittsburgh, PA, USA, 2000, pp. 62-71.

[38] bastien Roy and Ingemar J Cox, "A maximum-flow formulation of the n-camera stereo correspondence problem," in *Sixth International Conference on Computer Vision*, Bombay, India, 1998, pp. 492-499.

[39] Yuri Boykov, Olga Veksler, and Ramin Zabih, "Markov random fields with efficient approximations," in *IEEE computer society conference on Computer vision and pattern recognition.*, Santa Barbara, CA, USA , 1998, pp. 648-655.

[40] Vladimir and Zabih, Ramin Kolmogorov, "Multi-camera scene reconstruction via graph cuts," in *Computer Vision—ECCV 2002*.: Springer, 2002, pp. 82--96.

[41] Yuri Boykov and Daniel P Huttenlocher, "A new bayesian framework for object recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, Fort Collins, CO, USA, 1999.

[42] Dan Snow, Paul Viola, and Ramin Zabih, "Exact voxel occupancy with graph cuts," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2000, pp. 345-352.

[43] (2015, September) Vision. [Online]. http://vision.csd.uwo.ca/code/

[44] Nobuyuki Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, pp. 23--27, 1975.

[45] Aude Oliva and Phillipe G Schyns, "Coarse blobs or fine edges? Evidence that information diagnosticity changes the perception of complex visual stimuli," *Cognitive psychology*, vol. 34, pp. 72-107, 1997.