Image Region Completion by Structure Reconstruction and Texture Synthesis

Najm Alotaibi

Department of Computer Science
Aberystwyth University

August
2009

This thesis is submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy of Aberystwyth University.

ABSTRACT

In this thesis, we present a new image completion method that automates the filling in of holes left by the removal of undesired areas in images so that the final output image is visually plausible. The reconstruction of the hole is based on the assumption that regions, particularly in natural images, tend to be spatially continuous and are only separated by the hole and must therefore be linked. Therefore, our approach is based on first creating image structure (regions boundaries) in the hole and then propagating texture from surrounding areas constrained by this structure. Structure reconstruction is performed in order to preserve the global structure of the image, by creating regions in the hole with well defined boundaries such that they match the surroundings.

The images are first segmented into homogeneous regions. The regions touching the hole are then relabelled based on their colour and spatial distances. Similar regions are then linked resulting in creating a new area in the hole that will be flood-filled and then synthesised to match the surrounding structure. This reconstructed image is then used for texture synthesis as a constraint.

Our texture synthesis method proposes two modifications to the generic texture synthesis method and this includes a parallel synthesis order and an iterative synthesis scheme. The parallel synthesis, in which a pixel being synthesised is independent of other pixels during any given iteration and not affected by other previously synthesised pixels, helps reducing the directional bias caused by sequential scanning orders such as the raster scan. The iterative synthesis scheme allows global randomness which will progressively converge towards fine detailed texture. This scheme ensures that the created texture has sufficient, but not excessive, randomness and does not have replications of entire patches. As a result, the method is able to convert gradually the input image into plausibly synthesised image and to remove visible boundary artifacts. The combination of the image structure and texture synthesis methods results in having an image completion method that is capable of dealing with images with large holes that are surrounded by different types of structure and texture areas.

ACKNOWLEDGEMENTS

I would like to thank a number of people who have helped me greatly during my PhD until the production of this thesis.
Firstly, my supervisor, Dr. Frédéric Labrosse, for everything I have learned from him, his warm encouragement, thoughtful guidance, and infinite kindness and patience over the last four years. I also want to thank Dr. Reyer Zwiggelaar and Dr. Mark Neal for the helpful discussions we had about my thesis.
I also want to thank my sponsor, the Saudi Institute of Diplomatic Studies for the financial support during my PhD study.
I am grateful to my parents for their continuous help and sincere prayers. Their advise has always been helpful. Also, great thanks to my family: my wife, Modi and my kids Lama, Feras and Bayan for their support and patience over the past four years.
Finally, I must thank my brother Saad and cousin Maziad for their general support.

Contents

List of Figures

List of Tables

Glossary

HBR	Hole bounded region: a region created by the segmentation of the image that touches the hole of the image but is outside the hole, p. 66.
HBR-BE	HBR boundary extremity: a point at the end of the boundary between the hole and the hole bounded region.
Hole	An area of the image left by the removal of pixels corresponding to unwanted features and that needs to be filled-in with structure and texture in a plausible, convincing manner.
Hole region	A region that is created in the hole after its structure and texture have been reconstructed.
RCPL	Region connections priority list: a list of hole bounded regions which will be connected according to the order, p. 68.

Chapter 1

Introduction

1.1 Image completion

Images sometimes contain regions that are flawed or have undesirable objects. Fixing these flaws correctly is an important goal in applications such as photo editing, wireless transmission of images (recovering lost blocks), and film post-production. The problem of image completion is defined as being the following: given an original image and a hole mask, the goal is to automatically fill in the area of the image corresponding to the mask such that the synthesised part is as indistinguishable as possible from the rest of the image, as well as being plausible. Typically, such completion propagates new texture from the surrounding areas into the hole by "copying" similar examples from such areas. This propagation should produce a realistic structure and texture inside the hole so that it matches the surroundings with no visual artifacts. The hole can be of any size and shape and surrounded by many different areas. Figure 1.1 shows an example of applying our image completion method to the CLEDWYN image. There are editing tools,

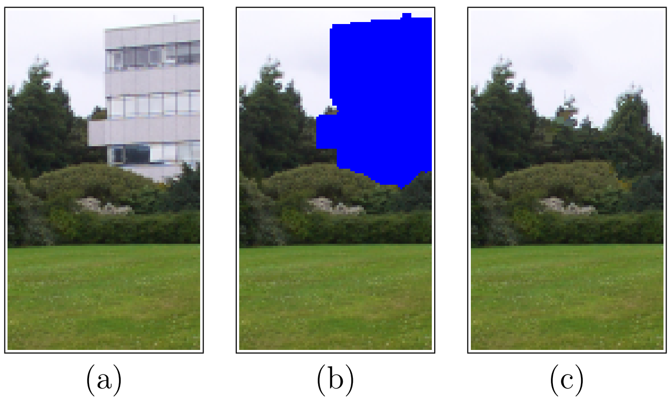


Figure 1.1: Example of applying our image completion to the CLEDWYN image: (a) original image, (b) original image with a hole masked, and (c) final result

such as clone brush strokes, that can be used to fill in the removed portion of the image. However, with such tools professional skills and manual user interaction are required. As a result, image completion deals with these limitations especially when the region to be removed is large and within many different textures.

During the last decade, there has been substantial work dedicated to deal with the problem of image completion and the current results are very convincing for many types of images. However, existing methods may produce visual artifacts such as hole boundaries, blurring, replicating large blocks of texture, or simply cannot preserve the global structure in the image, especially when the hole is large and surrounded by different types of textures. By the *structure of the image* we mean the spatial organisation of the different parts that constitute the image. These parts can typically correspond to areas of the image where the texture is homogeneous, according to some measure that will be discussed later. Methods used to solve this problem have used different approaches and strategies varying from completion using structure, completion using texture, and completion using both texture and structure. The methods of the first category fill in the hole based on propagating linear structure by generally using mathematical models such as partial differential equations (PDEs) and image diffusion. This approach is best suited to restoring and fixing small scale flaws such as scratches, stains and overlaid text, but is not suited for large holes surrounded by textured regions, as the process is local and can cause blurring artifacts. The second group of methods complete the hole using texture synthesis by creating texture that is copied or modelled from the surrounding examples of texture. This approach produces good results for a large variety of textures. However, it does not ensure the correctness of the global structure of the image as many of these methods are based on local sampling that doesn't take into account the global structure of the image. However, other methods of this approach that are based on global feature extraction can preserve the global image structure, but can result in insufficient representation of the texture structure. The third category of methods combines the two approaches by creating texture that is constrained by structure. These methods have been successful in completing holes in images, by propagating the structure in the hole first and then creating texture based on that structure constraint. The methods of producing structure and texture in the hole vary in their strategy of completion, and a review on these methods and other image completion methods is discussed in Section 2.4.

In this thesis, we present a new method that automates the filling in of a given hole left by the removal of an undesired area in an image so that the final output image is visually plausible. The method follows the approach of combining structure and texture to complete the hole of the image. The hole structure is propagated first in order to preserve the global structure of the image by creating regions in the hole with well defined boundaries so that they match the surrounding structure of the rest of the image. A modified texture synthesis method (based on the work proposed in [38]) is then used to create textures inside the hole that are constrained by the reconstructed structure, and the textures will be coherent globally, hence producing plausible results. Also, the texture synthesis method is used for all the stages of image completion which include the structure reconstruction and texture synthesis, which provides an integrated scheme to do the whole process. Also, it is important to distinguish between the structure of the image/hole and the structure of the texture. What we mean by the structure of the image/hole is a layout that define boundaries between regions in the hole while the structure of the texture defines various characteristics of the texture elements such as arrangement, size and shape.

It is important to point out that our method does not have high level knowledge of what the real hole structure should be and therefore some user interaction might in some cases be necessary to modify the existing structure so that it matches the structure of the surrounding areas.

The following sections describe the main contributions in this thesis followed by an overview of its chapters.

1.2 Research contributions

The main contributions in this thesis can be divided into two main parts: reconstructing image structure and synthesising texture.

- We propose an image structure reconstruction method that is based on the assumption that regions (particularly in natural images) tend to be spatially continuous and are separated by only the hole and must therefore be linked. These regions are homogeneous and are spatially continuous in the sense that their local statistics do not change with position. The originality of the method is that the reconstruction of the image structure is based on measuring the similarity and connectivity of these regions.

- Measuring the similarity is based on colour statistics and proximity of regions. Thus, the image is first segmented and the regions attached to the hole are then relabelled based on their colour and spatial distances (Section 5.3). In practice, colour similarity is computed by normalising colour histogram for each region that touches the hole and then calculating the Euclidean distance between the two histograms. The assumption that areas of the original image are only broken up into different regions because of the presence of the hole implies that regions should be linked not only based on their colour content but also on their spatial proximity. This is computed based on the Euclidean distance between points that link two corresponding regions across the hole.

- Connecting regions is done by joining similar regions together using straight lines, and then flood-filling the created enclosed regions (Section 5.4). These connections are governed by principles that do not violate other valid region information, and can produce acceptable structure. These newly created regions are then synthesised to match the surrounding structure (Section 5.6). This structure reconstruction of image regions in the hole incorporates the global structure of the images which results in a structure that is consistent with the surrounding structure and therefore will guide the texture synthesis to produce more realistic results.

- We propose a texture synthesis method based on the work proposed in [38]. Our method introduces modifications to the existing general texture synthesis methods, which works for many different textures with less parameter tweaking than these methods. The two main parts of our method are an iterative synthesis scheme and order independent parallel synthesis.

- The iterative synthesis scheme allows global randomness while preserving local precision of the synthesised texture. The selection of good matches is important for the quality of the results. This selection must ensure that the generated texture has sufficient variation without replicating entire patches. In order to achieve this, we iteratively fill in the hole (using the parallel method), first allowing a wide set of good matches, then reducing the size of the set (Section 4.5).

- The parallel synthesis, in which a pixel being synthesised is independent and not affected by other pre-synthesised pixels, helps reducing the directional bias caused by scanning orders such as the raster scan (Section 4.3), and avoids introducing boundary artifacts. In practice, a new image (temporary buffer) holds the values of the synthesised pixels. This temporary buffer is then copied back to the image being synthesised at the end of each iteration, when all the pixels have been processed. In order for the parallel syntheses to work, it requires an initial seed. The aim is to provide initial values in the hole that have a certain amount of randomness. However, the values must be plausible. Therefore, They are copied from nearby pixels that do not belong to the hole using a Gaussian distribution centred on the current pixel.

We combine both the structure reconstruction and texture synthesis methods by using the reconstructed structure image as a constraint for the texture synthesis. This ensures that the texture synthesis creates texture from relevant regions and also preserves the global structure of the image, thus, producing realistic textures, having consistent image structure.

1.3 Overview of the thesis

The rest of the thesis is structured as follows: in Chapter 2, a literature review on the work related to texture analysis, texture synthesis, and image completion is provided. Chapter 3 describes the methodology used in this work to evaluate the results and presents the images used in the experiments performed to evaluate the performance of the proposed method. Chapter 4 investigates the limitations of the generic texture synthesis approach and proposes our solution to these limitations. It also discusses the results of image completion using only texture synthesis. In Chapter 5, the structure reconstruction of the hole is analysed, and the results of the method are discussed. Chapter 6 describes the whole process of image completion by combining the structure reconstruction of the hole with the texture synthesis. Results of this combined method are also analysed and discussed. Finally, we conclude our work in Chapter 7 together with a discussion on possible future work.

Chapter 2

Literature Review

2.1 What is texture?

Texture is an important characteristic of an image that has been widely studied over the last three decades. Although human beings perceptually recognise a texture when they see one, the concept is difficult to define, and there is no general agreement of accepted definition as it can be formulated based on a particular application. However, in general, "texture" is usually referred to as visual elements that are spatially repeated deterministically or stochastically. Therefore, textures can be classified into two categories: structural (e.g. regular) textures and stochastic textures. Structural textures are formed based on the arrangement of image primitives (texture elements) and by placement rules defining the spatial distribution of the primitives. Such placement rules include scaling, rotation and reflection [27,50]. On the other hand, stochastic textures are based on probabilistic models such as Markov Random Field, and do not have clear identifiable primitives or regular placement rules. In reality, most real-world textures are mixtures of these two types. Figures 2.1 and 2.2 show examples of structural, statistical textures respectively.

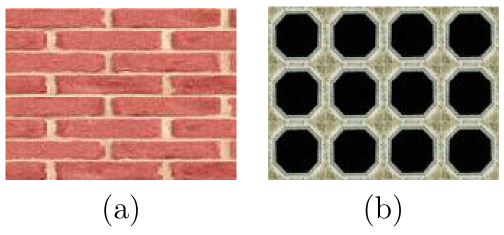


Figure 2.1: Examples of structural textures: (a) wall bricks and (b) stones, taken from [114].

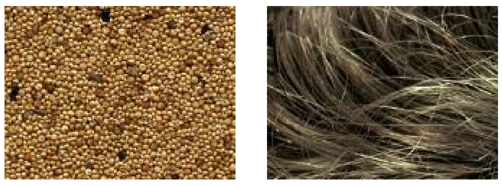


Figure 2.2: Examples of statistical textures: (a) birdseed and (b) hair, taken from [114].

2.2 Texture analysis

Many methods of analysing and characterising textures have been proposed and used for many years. They are categorised into four main groups: statistical, structural, model-based, and signal processing methods. These methods are discussed in the following sections.

2.2.1 Statistical methods

Statistical methods have been used to analyse the spatial distribution of grey values in an image by computing a set of statistical properties. The process starts by determining local features at each point in the image and then extracting a set of statistical properties from the distribution of the local features. The statistical methods can be a first-order (one pixel), second-order (two pixels), and higher-order (three pixels or more). There are many statistical methods of analysing texture such as intensity histogram features, co-occurrence matrix, autocorrelation function and edge frequency. The co-occurrence matrix is the most popular method for extracting features for texture analysis. Thus, it is chosen to be discussed in this section. The intensity histogram method is also explained as its statistics are used in Chapter 5. For more detail on the other statistical analysis tools, the interested reader is referred to [117].

Intensity histogram

The intensity histogram (H) of an image is a process of recording the number of occurrences of gray levels in the image, so it only contains first-order statistics. The probability density function of occurrence of grey level values is defined as:

$$P(i) = \frac{H(i)}{S}, i = 0, 1, 2...n - 1 \tag{2.1}$$

where $H(i)$ is the intensity histogram value at gray level i, S is the size of the image or its regions, and $P(i)$ is the probability of occurrence of gray level i.

Mean	$\mu = \sum_{i=0}^{n-1} iP(i)$
Variance	$\sigma^2 = \sum_{i=0}^{n-1} (i-\mu)^2 P(i)$
Energy	$E = \sum_{i=0}^{n-1} (P(i))^2$
Entropy	$H = -\sum_{i=0}^{n-1} P(i) \log_2(P(i))$

Table 2.1: Examples of statistical properties that can be extracted from intensity histogram of an image.

From the histogram, a number of useful statistical parameters can be extracted including mean, variance, energy and entropy. The mean describes the average intensity of the image while the variance measures the variation of intensity around the mean. The energy of a texture image describes its uniformity. Thus, the energy of an image is high when the image is homogenous. The entropy on the other hand, describes the randomness of the image. In general, when energy gets higher, entropy gets lower. Therefore, homogenous images have higher energy, but lower entropy. Table 2.1 shows examples of these statistics. The statistics of intensity histogram are used in Chapter 5 to extract statistical properties of image regions.

Co-occurrence matrix

Co-occurrence matrix is a widely used statistical method which is proposed by [53]. It is generated from the image by computing the statistics of pixel intensity. The use of the method is based on the concept that the same grey level values are repeated in the image. The size of the co-occurrence matrix (grey level matrix) is equal to the number of grey level values in the image, and computed for a given direction and a given distance as shown in the next example. Practically, an element $C(i, j)$ of the gray level co-occurrence matrix C counts the co-occurrence of pixels with gray values i and j at a distance d and a direction θ. To show an example of a gray level co-occurrence matrix (GLCM), consider a 4×4 image containing 3 different grey values:

$$\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 2 & 1 \\ 1 & 2 & 2 & 3 \\ 2 & 2 & 3 & 2 \end{bmatrix}$$

The 4×4 gray level co-occurrence matrix for this image for the east (1,0) spatial relationship, and a distance $d = 1$ is as follows:

$$C = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 2 & 2 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

The GLCM above is not symmetrical. If the GLCM is computed with symmetry, then only angles (θ) up to 180° need to be considered. Thus, texture calculations require symmetrical matrix in which the relationship i to j is indistinguishable for the relationship j to i, and directions on both sides (e.g. east and west) are considered. The above GLCM can be made symmetrical by adding it to its transpose as shown below:

$$C' = \begin{bmatrix} 4 & 1 & 1 & 0 \\ 1 & 2 & 2 & 0 \\ 1 & 2 & 4 & 3 \\ 0 & 0 & 3 & 0 \end{bmatrix}$$

The co-occurrence matrix is mainly used to derive second-order statistical texture properties. The most important used features are energy, entropy, inverse difference moment, inertia and correlation. These features are defined in Table 2.2.

where $C(i, j)$ is the (i, j) element in a co-occurrence matric C, μ_i is the the mean of the horizontal co-occurrence matrix, μ_j is the mean of the vertical co-occurrence matrix, σ_i is the the standard deviation of the horizontal co-occurrence matrix, σ_j is the standard deviation of the vertical co-occurrence matrix, and n is the number of grey levels.

When the high values of the co-occurrence matrix are near the main diagonal, the inverse difference moment's value is high. In contrast, the feature inertia has a high value when the high values of the matrix are far away from the main diagonal. Both inertia and inverse difference moment features can be used to measure the coarseness of an image. The last feature is the correlation which measures the correlation between the matrix elements. When the image correlation is high, the image seems to be more complex than when correlation is low.

Energy	$E = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} C(i,j)^2$
Entropy	$H = -\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} C(i,j) log C(i,j)$
Inverse difference moment	$D = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} \frac{C(i,j)}{1+(i-j)^2}$
Inertia	$T = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} (i-j)^2 C(i,j)$
Correlation	$R = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} \frac{(i-\mu_i)(j-\mu_j)C(i,j)}{\sigma_i \sigma_j}$

Table 2.2: Examples of statistical features that can be extracted from the co-occurrence matrix.

2.2.2 Structural methods

The structural methods of analysing textures are based on the fact that a texture consists of primitives with their geometrical properties. The structural methods of texture characterise texture as a set of primitives, which can produce (synthesise) texture by placing them according to specific placement rules. Texture primitives can refer to regions (blocks) in the image with uniform gray level values [117]. To analyse a texture using a structural method, two steps need to be considered: finding ways of deriving these primitives and identifying the placement rule.

Image primitives can be extracted by using different methods such as filtering the image with Laplacian of Gaussian masks at different image scales in order to extract "texels" (texture elements) [14]. Once the primitives of the texture have been identified, one can either compute the statistics of these primitives (e.g. intensity and shape) or identify the placement rule. The placement rule can be defined by a tree grammar as Fu described in his approach [45]. When applying the tree grammar rules, the texture can be viewed as a string in the language defined by the grammar whose terminal symbols are the texture primitives. This method can be used to analyse and synthesise texture.

2.2.3 Model-based methods

Model-based methods are based on capturing the underlying process that generates a texture. An image model can be used to describe, analyse and synthesise the texture, by estimating its parameters. One widely used model-based methods are Markov Random Fields models.

Markov Random Fields (MRFs)

MRF models are popular and widely used for modelling images. The models have the ability to capture the local spatial information in an image, and assume that the intensity at each pixel depends only on the intensities of the neighbouring pixels [117]. MRF models have also been used in many applications such as texture synthesis [31], image segmentation [29], and image restoration [48]. Moreover, the models have been used as a basic assumption for most texture synthesis techniques [3, 38, 49, 54, 125, 134].

2.2.4 Signal processing methods

Signal processing methods analyse the frequency content of the image. They are based on extracting certain features from filtered images using signal processing techniques, e.g. spatial domain filters, fourier domain filters, and Gabor filters. One of the most useful features, the spatial domain filters extract is measuring the image edge density per unit area. Measuring the edge density is usually done by using edge masks such as the Laplacian or Robert operators. For more detail on these filters, the interested reader is referred to [117].

2.3 Texture synthesis

Texture synthesis is the process of creating a new image that is different from the original, yet being perceptually similar to the original image. Texture synthesis is a better way to create textures compared with hand drawing and scanned images in the sense that images are more realistic and made at an arbitrary size. With hand drawing pictures, it is difficult to make them photo-realistic, and with scanned images, sizes are not adequate [125]. Therefore, texture synthesis can be used to generate large textures from small texture samples. Figure 2.3 shows an example of texture synthesis, which was taken from [125]. Texture synthesis has been an active area of research for the last decade. For example, in computer vision, computer graphics and image processing, there are many texture synthesis applications such as texture classification and segmentation, rendering, compression, wireless image transmission, image editing, etc.

Texture synthesis methods create a model of the texture and use that model to produce a new texture having the same properties as the original texture. The models can be categorised broadly into a feature matching (parametric) [15, 57, 97]

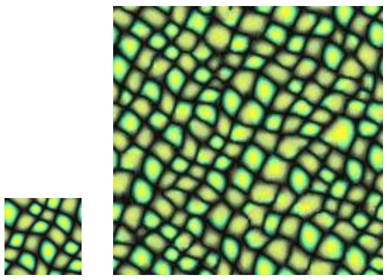


Figure 2.3: An example of texture synthesis from [125]: (a) original texture (b) synthesised texture

or non-parametric [3, 38, 125] approaches, and these models being derived from the original texture. In other methods, the model does not contain the original texture, but its sufficient statistics [49, 134]. The feature matching approach often use a set of features and synthesise texture based on matching the statistics of the features in the example texture. On the other hand, the non-parametric approach generates texture by local probability sampling.

In a general comparison between these two approaches, the feature matching approach can result in insufficient representation of the texture which can lead to failing to synthesise. Similarly, the non-parametric methods may not be able to capture the global representation of the texture. Also, non-parametric approaches have the advantage over the feature matching approaches that the computational cost is relatively lower due to avoiding explicit construction of statistical models.

2.3.1 Feature matching approach

Texture synthesis techniques based on this approach use global statical features to match between the original and new textures (in the context of pure texture synthesis) or between the undesired area of the image and the rest of the original image outside the hole. Thus, the new texture is generated according to statical models derived from the original texture after applying a bank of filters to the original texture.

Heeger and Bergen [57] have presented a method for synthesising an image that matches the appearance of a given texture. In practice, they synthesise textures by matching the histogram of filter outputs of an input texture and initial noise images. The noise is then modified to make it look like the input texture image. This was done by matching histograms of two texture images, using a combination of Laplacian pyramid and steerable pyramid. The synthesis technique succeeds on highly stochastic textures, but fails on structured ones. Figure 2.4 shows a successful texture synthesis example, which is presented in [57]. Figure 2.5 shows failure examples of their texture synthesis.

De Bonet [15] generates texture by matching the distributions of multi-resolution filter output in a way that preserves dependencies across multiple resolutions of the input texture. The method offered an improvement on the work presented in [57] by successfully synthesising a wide range of textures including structured ones. However, boundary artifacts can appear in the produced texture if the input texture is not tileable [125].

Portilla and Simoncelli [97, 108] synthesise texture by matching pairwise joint statistics on the wavelet coefficients using multi-scale image representation. Their method was able to successfully produce good results for both stochastic and structured images. However, it has difficulties in producing high frequency patterns with highly structured textures [38].

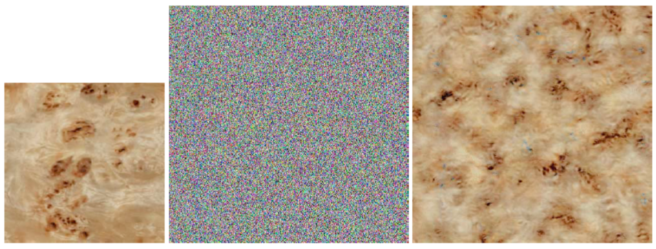


Figure 2.4: Successful synthesis example from [57]: (Left) sample texture image (Middle) random noise image (Right) synthesised texture

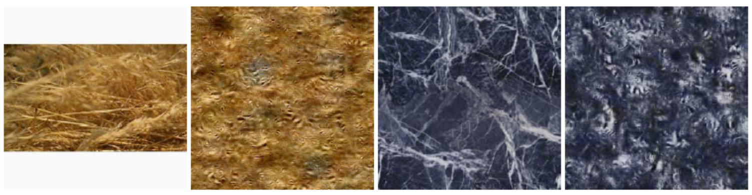


Figure 2.5: Failure examples from [57]: hay and marble texture images

2.3.2 Non-parametric approach

The non-parametric approaches synthesise a texture by computing conditional distribution of its pixels by using a neighbourhood matching strategy. The methods are motivated by a Markov Random Field model in the way that a value of a given pixel only depends on its neighbours. This means that the methods do not explicitly assume that the probability distribution of brightness values as MRF, but the MRF is only used as motivation. A number of nearest-neighbourhood texture synthesis methods have been recently presented in [3, 37, 38, 54, 58, 71, 78, 85, 125]. This approach of texture synthesis can be categorised based on the synthesis unit into pixel-based texture synthesis and patch-based texture synthesis. Pixel-based methods synthesise texture based on a pixel by pixel synthesis while patch-based methods use patches (blocks) of pixels to synthesise texture. The following sections analyse these methods.

2.3.2.1 Pixel-based texture synthesis

Efros and Leung [38] have developed a pioneering approach using probability sampling to generate texture. The method is very simple to implement and can synthesise different types of textures of any size and shape. In their work, the conditional probability distribution for a pixel is estimated by searching the sample image and finding the neighbourhood similarity to the pixel's neighbourhood. In practice, to synthesise a pixel P, the algorithm scans the whole sample image searching for all neighbourhoods that have a good match with P's neighbourhood. Best matches are computed by the normalised sum of squared differences metric (**SSD**). Only these within a distance less than a threshold ϵ are considered and included in the best matches list. Later, one is randomly selected and its centre pixel is then copied into P. One important parameter used is the size of the neighbourhood. If the size is too small, the local structure of the texture will not be captured properly. Contrary, if the size is too large, computational cost will be higher and there can be a risk of having verbatim copying of large areas of the texture.

Despite the fact that the method produces very good results for a wide range of textures, it reported a few failures. For example, when the texture contains too many different types of texture elements, the method gets into a wrong part of the search with no good matches and starts growing "garbage" as referred by the author to the unrealistic texture especially at the left corner of the bottom of the image [38]. Another failure is shown in Figure 2.6(b) when the method gets "stuck" at a particular area in the image and starts producing "verbatim" (exact) copying [38]. Also, the method uses a "causal" neighbourhood which only contains the preceding synthesised pixels of the current pixel being synthesised. This neighbourhood shape does not include other parts of the neighbourhood, and this can cause boundary artifacts [125].

Many variations of the method presented in [38] have been published that have

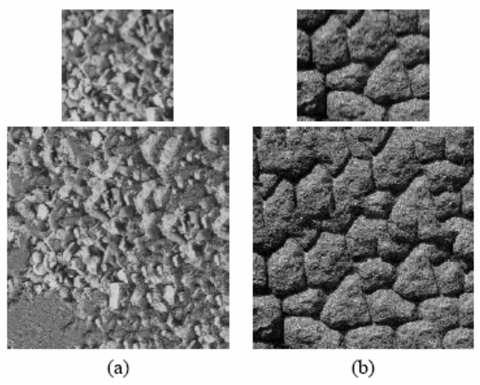


Figure 2.6: Failure examples from [38]: (a) the method gets into a wrong part of the search and starts growing “garbage”. This clearly appears at the left corner of the bottom of the image where the elements of the texture are not realistic, instead “garbage” was produced. (b) the method produces “verbatim” copying where exact blocks of the texture are copied from particular areas.

introduced improvements on the method [3, 37, 54, 78, 125]. Wei and Levoy [125] presented a multi-resolution synthesis in order to reduce the computational cost of the exhaustive search in [38] by representing large scale features in a lower resolution level by using a smaller neighbourhoods. The computational cost of applying a multi-resolution method is much lower than the single resolution method because at each level of the resolution pyramid, small neighbourhoods are used, which is faster to compute. The synthesised texture is generated from a coarse to fine scale in the image multi-resolution levels using a raster scan order. At each pixel, causal neighbourhoods on the current level and full neighbourhood (symmetric) in the next higher pyramid level are used. The search for the best neighbourhood match to the current pixel neighbourhood is performed as in [38] using the (**SSD**) metric. Also, the multi-resolution synthesis can help to reduce the artifacts caused by using the causal neighbourhood, by having all the neighbouring pixels available in the match. Also, the efficiency of the method is improved further by using tree-structured vector quantisation (TSVQ).

Ashikhmin [3] develops a modification method to [125] that produces better quality results for natural textures, and also that is faster than their method. He found that the method of [125] produces poor quality results on certain types of texture that contain various objects of irregular size and shapes, but “familiar” ones [3]. Figure 2.7) shows that the method of [125] has a tendency to introduce a blurring artifacts for particular types of textures, and this can be a result of using the TSVQ as reported in their paper. However, Ashikhmin [3] presented in his paper that the blurring effects can be the result of using the (**SSD**) metric and suggested to be replaced by perceptual metrics to improve the quality of the result, even though he acknowledged that such metrics are not completely reliable because of lack understanding of the human visual system.

Due to restrictions in the search domain, the method reduces the computational cost significantly. The method is based on the observation that given the positions in the original image of the neighbourhood of the pixel being synthesised, a list of candidates for that pixel can be computed by shifting these positions according to their displacement with that pixel. Although the method significantly speeds up the synthesis process, it also can reduce quality because the "small" list of candidate may lead to a shortage in good matches in the texture. The method is extended to give the user an interactive control over the appearance of the synthesised image by simply providing a target image with some general properties which show how the final result should look like, leaving the details to be filled by the synthesis algorithm.

Although Ashikhmin's method produces good results for natural textures, discontinuities can occur when synthesising highly structured textures. As a result, Hertzmann et.al. [58] proposed a method of combining Ashikhmin and Wei and levoy methods by using the coherent neighbour searching of Ashikhmin and the Euclidean distance in [125]. This combination of both perceptual similarity measures produces better results, especially with such textures.

Despite the fact that computational cost of texture synthesis has been reduced significantly, especially by Ashikhmin's coherent search, it is improved further in real time by Zelinka and Garland [135]. The method is very fast because there are no neighbourhood comparisons at the synthesis stage as they are already precomputed at the analysis stage and can be used for texture synthesis. They presented a method that synthesises texture based on data structure referred to by the authors as the "jump map". The jump map stores, for each pixel in the input texture, a set of k-nearest matching neighbours in a look up table. The table is then used to make random jumps at the texture synthesis stage, in which the random jumps decide from which neighbours to continue copying pixels.

2.3.2.2 Patch-based texture synthesis

Recent methods have used patches instead of single pixels as in [37, 71, 78, 81, 85, 86, 119, 131]. The patch-based methods are found to work well for a wide range of textures and their computational cost is lower than that of the pixel-based methods. The comparison between the pixel-based and patch-based methods is discussed in Section 2.3.3. Most of these methods are concentrated on introducing new techniques to handle patch overlap regions, thus avoiding boundary artifacts.

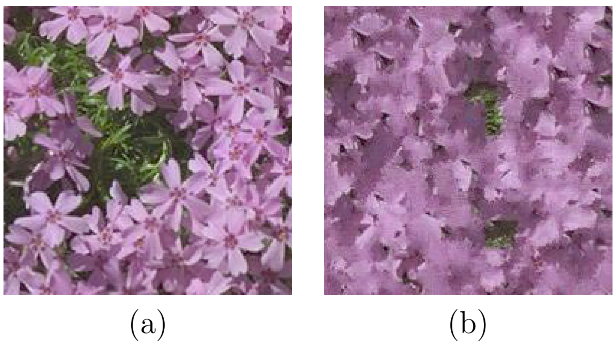


Figure 2.7: An example of applying Wei and Levoy's method on this particular texture, taken from [3]: (a) input texture, and (b) the method has a tendency to produce blurred results for some types of texture [3].

One of the first attempt to use patches for texture synthesis, instead of copying single pixels at a time, is the work of [131]. A set of patches are copied from the input texture and then randomly pasted onto the output texture. The boundaries between patches are treated by using a cross-edge filtering, which enables a smooth transition across these boundaries. However, the cross-edge smoothing can cause mismatch features and image blur.

Instead of blending overlap regions with a filter, Efros and Freeman [37] proposed a patch-based texture synthesis method that is based on merging patches that satisfy an overlap boundary region constraint. The synthesis process starts by placing a square patch, taken randomly from the sample image, in the upper left corner of the output image as a seed. A new patch is placed on the output image in a raster scan order overlapping the existing patch by some number of pixels. The new patch is chosen by searching in the sample image for a set of patches that satisfy some overlap constraints (the overlap region is on the left and top sides of the new patch) within some error tolerance.

Once these patches are selected, a random patch is picked up and placed on the output image. The method finds a good boundary between the newly chosen patch and the old patches by computing the minimum cost path from one end of the overlap region to the other. Once the boundary is chosen, it is used as the boundary of the new patch and placed on the output image. When placing a selected patch on the output image, the patch's pixels in the overlapped region have to be merged with the other old patches in the output image.

Irregular shapes of patches have been used in [78] to remove the seams on the overlapped regions. The method uses alpha blending across the overlapped regions to provide a smooth transition between the new and old existing patches. This however, can cause blurring along the edges of the patches.

A generalisation of the method in [37] to irregular shaped patches, was presented in Kwatra et al. [71]. The method uses a technique they call “graph cut” in which patches can be of any shape in order to determine the optimal patch. The shape (portion) of the patches are completely determined by computing the minimum cost of graph cut along the boundary of the new and old patches.

Nealen and Alexa [85] used adaptive patches and pixel-based re-synthesis to remove the visual seams in the overlapped regions. These patches are selected so that they minimise overlap error with the existing synthesised patches. The overlap error is computed for each pixel in the overlap regions and mismatched pixels above a certain threshold are re-synthesised per pixel. The error is computed using the Fourier transform iteratively. Although the method produces good results, especially for textures with anisotropic structure, the iterative process increases the computational cost. Because of this limitation, they proposed a new method [86] that is faster than the re-synthesis pixel-based synthesis motivated by the k-nearest neighbourhood search used in [116]. The k-nearest neighbourhood was used to increase the number of possible pixel locations in the input texture in which limited number of candidate patches tend to produce unsatisfactory results [86].

Unlike most patch-based texture synthesis methods, which uses only colour information when searching for the best patch, Wu and Yu introduced a structural similarity measure that is motivated by the fact that the Human Vision System (HVS) is most sensitive to edges, curves and other high frequency features [130]. The method combines the structural measures with colour information so that the visual boundary seams between adjacent pathes are minimised.

2.3.3 Pixel-based vs. patch-based methods

Although pixel-based methods are reported to perform well with many different textures, still there exist some limitations. For example, the main limitation with the pixel-based methods is the high computational cost. However, some improvements to speed up the process have been performed using a multi-resolution framework and tree-structured vector quantisation accelerator as in [125], restricted search area as in [3] or a jump map as in [136]. The other problem with the pixel-based synthesis is that it may not be able to preserve the global features of the texture properly, as shown in Figure 2.6(a). However, the methods are good at matching small features of the texture, and when a symmetric neighbourhood is used, the representation of global features is better. Another problem with these methods is the blurring artifacts that can be caused by the limitation of distance metrics in some textures that are not well suited for mimicking the HVS, as shown in Figure 2.7.

On the other hand, patch-based texture methods have lower computational cost than the pixel-based methods. They also tend to capture the underlying structure of the image better than the pixel-based methods, especially with structured textures. However, due to patch fitting problems, the methods can produce visual artifacts such as large self-similar areas and visible patch boundaries. The patch fitting problems can be caused by inaccurate similarity measures [130].

2.3.4 Filling Order

The order in which an image is synthesised is an important part of the synthesis process since it can highly affect the quality of the output image, as different synthesis orders can generate different textures. This is because pixels being synthesised always depend on previously synthesised pixels and this would result in "cyclic" dependencies among output texture pixels [126] causing directional bias in texture propagation.

During the synthesis process, pixels are considered in some order such as raster scan [125] or spiral from a centre seed in the case of unconstrained texture synthesis [38] or from the outside of the hole inwards as in the case of constrained texture synthesis [38,125]. The raster scan order (from top left corner of the image to the bottom right corner) can favour parts of the image at the expense of others and this can produce boundary artifacts. The spiral order on the other hand, has been used successfully to reduce the directional bias caused by the raster scan order, as shown in Figure 2.8. However, the problems with the spiral synthesis is that a boundary may remain visible at the "centre" of the spiral, particularly when a single pass is used.

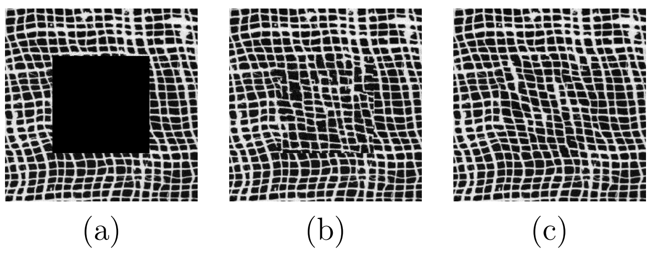

(a) (b) (c)

Figure 2.8: Synthesis order for constrained texture synthesis, from [125]. (a) original texture containing a region to be completed, (b) discontinuities caused by the raster scan synthesis order as well as the causal neighbourhood, and (c) a spiral order synthesis is used as well as the symmetric neighbourhood using two passes.

In general, any sequential order can produce boundary artifacts, maybe spread over the size of the neighbourhood in cases of in-homogeneous textures. Therefore, the best solution to these problems is to have order-independent texture synthesis [74, 126, 133]. The work of Wei and Levoy [126] presented a new approach to to the order of synthesis by using order-independent texture synthesis method. The method is simple as instead of synthesising pixels directly into the output image, they are stored in a separate image. The neighbourhood comparison are performed on the pixels of the separate image rather than the ones in the output image, and synthesise texture in a multi-dimensional paradigm.

We recently discovered that our parallel method is similar to the one proposed in [126] although it is used in a single resolution synthesis and has a different initialisation method. More on our parallel method is described in Chapter 4.

2.4 Image completion

2.4.1 Overview

Images often contain undesirable artifacts that need to be removed and later fixed. The main challenging requirements of image completion is that when removing the flawed region, the replacing texture of the region and its surroundings must be compatible and realistic. Also, the boundaries between the new and old regions must be invisible. These challenges have been investigated by a great deal of research that broadly can be classified as structure completion (image inpainting), texture completion, and combined texture and structure completion.

2.4.2 Image inpainting

Image inpainting fills in gaps or undesired small areas in images by smoothly propagating the surrounding structures inside the areas, considering their angle of arrival and possible curvature. Several structural propagation techniques have been investigated in the literature of image inpainting [7, 9, 12, 24, 39, 88, 113].

Image inpainting was first introduced by [9] and was based on replicating the way professional restoration artists work. Their idea of filling in the undesired objects is to smoothly propagate the surrounding information on the boundary of the area being removed inwards. This filling in can be achieved by iteratively propagating the image information along the direction of the isophote lines (lines with equal gray values) using partial differential equations (PDEs) and image diffusion. The propagation of image information includes the pixel values and the directions of the isophote, which is done by numerically solving the following partial differential equation:

$$\frac{\partial I}{\partial t} = \nabla^{\perp} I \cdot \nabla \Delta I, \tag{2.2}$$

where ∇, $\nabla^{\perp}$ and Δ represent the gradient (g_x, g_y), the perpendicular gradient and the Laplace operator respectively. This equation only applies inside the removed areas. The isophote direction is the direction of least change in gray values (normal to the gradient). A 2D Laplacian is used to smooth the propagation of the structure. This smoothness will continue along the isophote direction. Thus, few iterations of anisotropic diffusion [95] are performed after few iterations of inpainting to preserve sharpness in the structure of the removed area. However, sharpness in edges is not always preserved [88].

Inspired by the work of Bertalmio et al., Chan and Shen presented two methods for image inpainting: Total Variation (TV) [25] and Curvature-Driven Diffusion (CDD) [24]. The TV method is based on an Euler-Lagrange equation that uses the anisotropic diffusion in [95] inside the inpainting region based on the contrast of the isophotes. The method works with inpainting small regions and is best suited to noise removal. However, one of the drawback of this method is that it does not connect broken lines of objects, especially when the length of the inpainted object l is much larger than the width of the object w (far separated regions), as shown in Figure 2.9.

The CDD method extends the TV inpainting method to overcome the difficulties of connecting broken lines. This was achieved by considering geometrical information of the isophotes when defining the diffusion strength. The diffusion strength gets stronger where the isophotes contain a larger curvature. However, it becomes weak when the isophotes stretch out [24]. This allows to have a steady inpainting over large areas. Figure 2.10 shows a successful CDD inpainting of a broken bar.

A number of other mathematical models have been investigated in the literature of image inpainting [5–7, 12, 39, 113], and the interested reader is referred to a review paper published in [112].

It is obvious that the main focus of this approach is the propagation of linear structures, such as lines and curves, in the undesired area, and does not take into account texture. Therefore, it is best suited for restoring small undesired regions or fixing small scale flaws such as scratches, stains, and overlaid text. Another problem with this technique is that the diffusion process can cause some blur in the image especially when the filled area is large, as can be seen in Figure 2.11. Also, it does not rely on global features (it is local), [62].

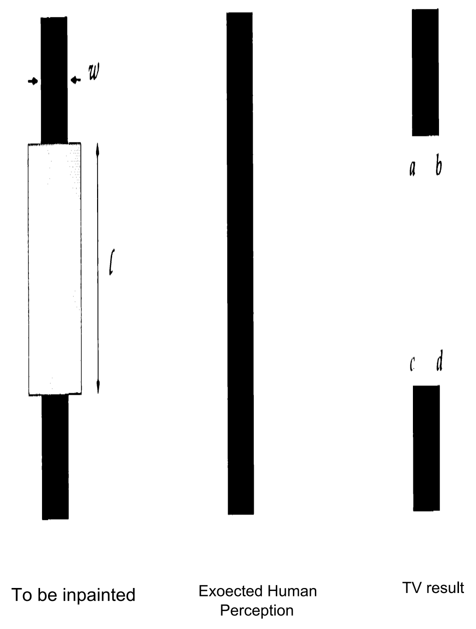


Figure 2.9: TV inpainting of a broken bar, from [24]. The l and w represent the length of the inpainted bar and the width of the bar respectively. The a, b, c and d show the parts where the disjoint bars should be connected (e.g a should be connected to c and b to d). However, the method clearly fails to reconstruct the structure of the bar.

2.4.3 Texture completion

Early work on image completion using only constrained texture synthesis has been presented in [17,35,38,60,61,69,79,125,129,138]. Efros and Leung directly apply their texture synthesis technique, described in Section 2.3.2.1, to fill in holes completely surrounded by homogenous texture. Similarly, Wei and Levoy use their texture synthesis technique to fill in unknown regions in images, but with some modifications on the original algorithm [125]. They synthesise texture in multi-resolution fashion, which includes using two passes to allow valid pixels in the search for neighbourhood match (Section 2.3.2.1). Also, a spiral order was used instead of the raster scan ordering (see Section 2.3.4). Most of texture completion methods performance vary depending on the followings: the filling order of the completion, neighbourhood searching strategy, the selection of the best match from the source image, and pasting it in the hole.

Inspired by the work in [38], Bornard et. al. [17] propose a method to generalise Efros and Leung's method to include complex textures in the context of image completion. In their work, the hole filling starts from the ones having the most valid neighbours (known pixels within the square neighbourhood window). For every pixel in the hole, the search for the best candidate matches is performed

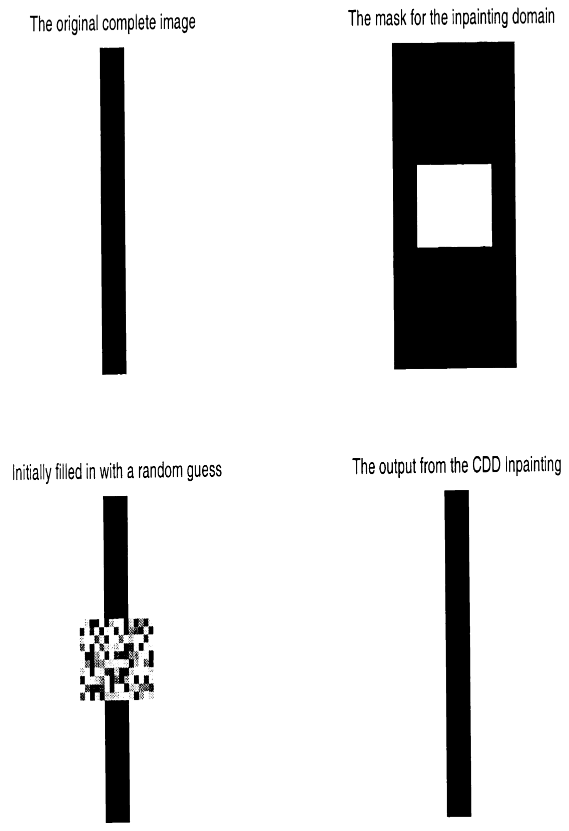

The original complete image
The mask for the inpainting domain
Initially filled in with a random guess
The output from the CDD Inpainting

Figure 2.10: CDD inpainting of a broken bar, captions are also taken from [24]. The TV inpainting method will produce two separate bars while the CDD inpainting successfully restored the whole bar, and this is what most human would perceive [24].

using a restricted adaptive domain search. This means that at a missing pixel, a sub image window of an initial size is centred on this pixel to be used as a search domain for the current pixel. The size of this sub image is increased until it reaches at least a minimum number of pixels outside the hole, in which the sub image is now ready to be searched. This domain search is later restricted by using the same coherence search introduced in [3]. This improvement is done to exclude irrelevant information in the sub image and to significantly reduce the computational cost even further. The method may have problems with using local restricted search as there might be a risk of not having enough examples of the texture, particularly when the hole is at the edge of the image. Also, the results of this work are only compared with the "image inpainting" method presented in [9], which gives an indication that the method may have difficulties in producing good result for texture image of large holes.

A more effective technique has been proposed by [35] to fill in large holes that

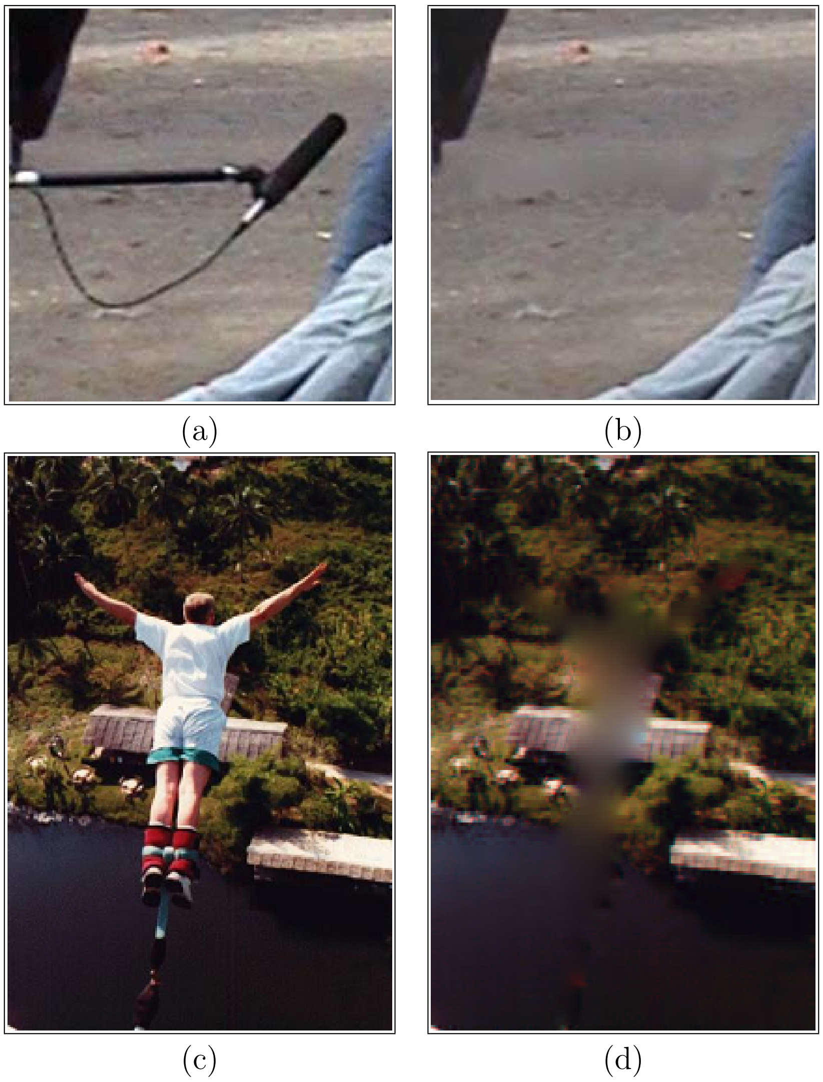


Figure 2.11: Image inpainting blurring problem, (a) and (b) taken from [62] and (c) and (d) from [30]. (a) and (c) original images with a microphone and a bungee-jumber to be removed, (b) and (d) the results of applying the image inpainting method, presented in [9], to (a) and (c) respectively.

are surrounded by multiple textures and structures. Their work used three main components to fill in the unknown regions: approximating the missing region, computing the filling order, searching for most similar source patch (outside the hole) and composting source and target patches. The algorithm runs these tasks on multiple levels of an image pyramid from coarse to fine where features from different resolutions are used. The smooth approximation of the unknown region is used to guide the search process, and is computed based on iterative filtering process. Once the approximation is done, the filling in process starts by identifying the traversal order for the unknown region based on a confidence map, which determines the position of the next target patch. As a result, the target pixels of high confidence will be filled before low confidence pixels. Therefore, pixels close to the border of the unknown region will be filled first. Once the target fragment is located, a search for the best source match is performed over all positions, five scales and eight orientations. The shape of the neighbourhood is circular and its size is adaptive dependant on feature at different scales. Once the best source fragment is found, it composites under the target fragment in which the pixels of high confidence in the source fragment seamlessly merges with the pixels of low confidence in the target fragment. Merging the source and target fragments is performed using a Laplacian pyramids and OVER operator. The OVER operator is used to composite two similar fragments, one over the other, to create seamless transition between them according to alpha values. Although the method produces very impressive results, it is complex and is computationally expensive.

Different filling orders and patch matching strategies have been proposed by [138]. First, the filling order of the unknown area is done by measuring the textureness in the target patches. The textureness is measured according to a spatial-range model. The model is a combination of distances between neighbourhood pixels and their mean value and spatial distances between the target pixel (at the centre of the target neighbourhood) and the rest of the neighbourhood pixels. Therefore, the filling in starts with the areas of less textureness (low frequency areas). After identifying the target patch, a source patch is selected by computing projective transformation parameters between the two patches before applying the SSD similarity distance. The reason of adjusting the two patches appearance before the comparison is that there are images with perspective deformation in which applying the SSD directly may result in inaccurate matching. Once the source patch is chosen, a graph cut algorithm is used where a minimum cut is applied in the overlapped region to separate the two patches in order to keep the boundaries of the patches least noticeable.

Lin [79] presented a patch-based image completion method based on restricting the search space to a region around the target patch. This is similar to [17], but the size of the searching region is a user defined parameter. He also used a different similarity metric that is not only based on computing gray level differences, but also includes the gradient differences. This metric integrates colour and anisotropic structure to find better matches for the patches of the hole. The filling order of the hole is based on shrinking the boundary of the hole based on identifying points on the boundary with strong edges where the filling in process starts from the two end points and shrinks to the middle. The method is fast and produces acceptable results. However, it can cause artifacts due to search restriction that limit the number of available examples of the source patches.

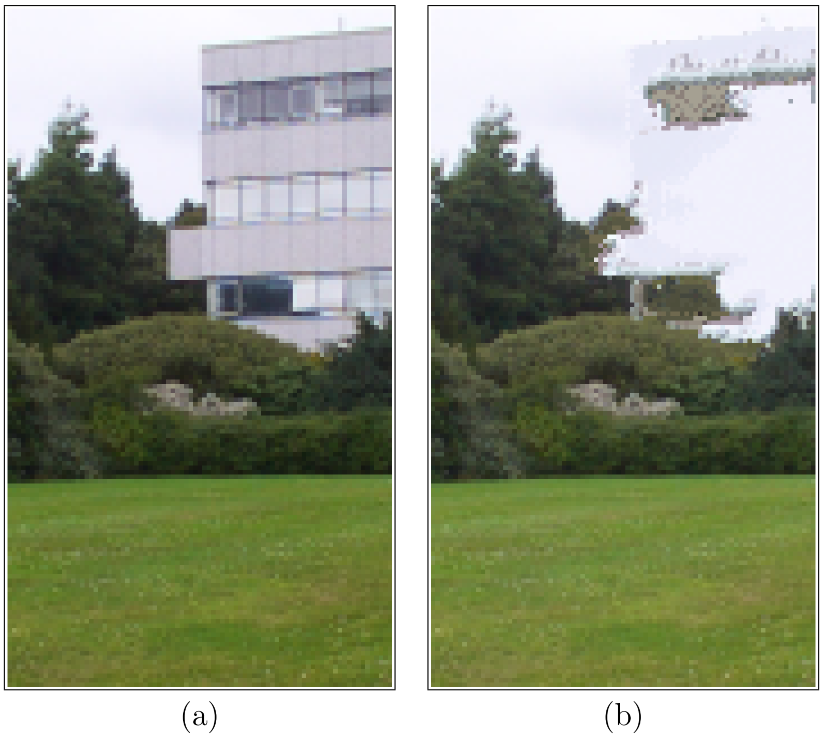


Figure 2.12: An example of a failure when applying the method in [38] to the image. The original image with a building to be removed (a), and (b) the result of applying the method in [38] which shows that different textures require different parameter in order to work properly. Also, it clearly shows that boundaries need to be created first. Note that these image are take from [72].

2.4.4 Structure and texture completion

The combination of both structure and texture propagation has been investigated in [11,30,51,64,72,101,110]. Most previously described methods of texture completion have their own limitations when filling in holes surrounded by many different textures. This includes the following: one is that many methods need a set of specially tuned parameters that work with one set of textures but not with others. Another problem is that the boundaries between the various synthesised textures must look realistic. Figure 2.12 shows an example of a failure when applying the method presented in [38] to this image.

This has been addressed in [72] where boundaries between textures were first propagated in the hole, thus creating regions containing only homogeneous texture, which were then filled in. Both synthesis (boundaries and texture) used a method based on that described in [38]. The differences were that possible matches of the current pixel were constrained to belong to the right boundary or region (using a segmentation of the image) and that a match was chosen using a Gaussian distribution to choose at random a match that is not too different from the best match.

In [11], the method combines image inpainting and texture synthesis approaches by decomposing an image into the sum of two functions: one for reconstructing the structure of the image and the other is for capturing the texture. Similarly, Criminisi et. al. [30] use both image inpainting and texture synthesis approaches to fill in holes in images. The image inpainting is used to propagate linear structures while the texture synthesis is used to propagate texture. The main idea is the designing of a hole filling order, which explicitly propagates linear structure and texture information. The filling order is based on the priority values that are given to patches that are centred on the boundary of the hole. The priority is calculated based on how much reliable information is around the pixel as well as the isophote at this point. For example, patches that are located on the continuation of strong edges will have high priority values, and thus the linear structure would be extended first. The priority of a patch $P(p)$ is defined as the product of two elements: a confidence term $C(p)$ and a data term $D(p)$. The confidence term describes the amount of reliable information surrounding the current pixel, which can be measured by computing the number of filled pixels in the patch. The data term describes the strength of the isophotes hitting the hole boundary. The data term is very important since it encourages linear structures to be synthesised first. Once the highest priority patch is found, a search for the most similar patch in the rest of the image is performed as in [38]. Once it is found, its pixels are then copied into the current patch. After the patch has been filled, the confidence value at P is updated, so as the filling proceeds, the confidence values decrease when coming close to the centre of the current region in the hole.

Rares et. al. [101] propose a new method for restoring missing areas in images and films. The method consists of three main components: image segmentation and feature extraction, hole structure reconstruction and texture synthesis. The segmentation of the image is performed to identify different textured regions around the hole. Once the segmentation is done, edges connected to the hole within a specified distance are extracted. A number of features used to identify edges with similar properties are computed. These features are the intensity and gradient angle histograms within the image segments. Therefore, each edge is characterised by these two features on the left and right side of the edge, shape fitting cost and sequentially. The sequentially factor is used to measure the degree to which the edge connections do not cross each other.

The second component is to construct the structure of the hole based on measuring the similarity of these edge features and selecting the best match. The output of this operation is a list of edge couples arranged in groups of couples and a list of spare edges. And edge couple consists of two edges from the same area while the spare edges are the ones for which there is no match. Once identifying all these groups of edges, all the edge couples are propagated in the hole according to a circle fitted to the couple while the spare edges are fitted as straight lines into the hole.

After the hole structure has been constructed, the texture is synthesised using a modified method presented in [17]. The modification is that the texture synthesis is constrained by a mask of the same segmented region inside the belt area. In this method, using a fixed belt size may influence negatively neighbourhoods and edges matching process. With the neighbourhoods matching process, it restricts the search to this belt area inside the current object region, and this may lead to ignoring some valuable information outside the belt area. The same problem may occur with edge matching if the edges do not have enough pixels to be compared with.

A different approach is presented in [110] as an interactive technique to image completion by propagating structure information in the hole using user specified curves. The curves are used to provide high level knowledge on what structure should be propagated and where texture can be obtained from. The method first synthesises the missing structure and texture along these curves in the unknown region using patches around the curves in the known region. In this work, structure propagation is considered as a global optimisation problem that is solved by dynamic programming or belief propagation. If only one curve is specified, dynamic programming is used while if multiple intersecting curves are specified, belief propagation is used to reduce the computational cost. After the user creates the curve, a set of anchor points are generated in the unknown region in which the centre of the synthesised patches are located on these points. Outside the unknown region the sample patches are centred within one to five pixels along the specified curve. After structure propagation, texture is then synthesised from the corresponding sub regions, which are partitioned by the creation of the user curves. The limitation of the method is that it lacks automation and requires user interaction for completing the unknown regions.

2.5 Conclusion

From the literature review, we can conclude the following:

- The generic texture synthesis methods are not well suited for dealing with images with large holes that are surrounded by different texture areas. Applying these methods directly to such images will cause problems related to replicating complete image areas, boundary artifacts or lack of preserving global image structure. Therefore, modifications to these methods has been introduced to handle these problems.

- The methods of image completion that are based on the image inpainting method presented in [9] are suited for propagating only linear structure and completing small holes. Also, problems of blurring can occur due to the diffusion process, which is local and does not consider global structures of the image.

- The methods that combine image structure and texture synthesis have produced good results for a large variety of textures. However, they do not ensure the global consistency image structure. Methods create image structure using image inpainitng which had its own problems as already discussed, or image segmentation, but connectivity of regions were based on local curvature that may not necessary represent the actual patterns of the surrounding structure. Other methods use interactive techniques to create the image structure, but such methods lack automation.

We propose to tackle these problems by developing a new method of image completion that will (1) explicitly re-create image structure in the hole that matches both the topology and geometry of the surroundings of the hole and (2) create texture in the hole that matches texture examples in the remaining of the hole. The new method will be such that the created pixel values merge seamlessly with the existing content of the image. To achieve this, we propose to modify existing texture synthesis methods so that they can work with different textures without requiring precise manual adjustment of parameters.

Chapter 3

Data evaluation and methodology

This chapter discusses the images used in the experiments and the reasons behind selecting them. This include expected difficulty in the shape, size or position of the hole or complexity of the surrounding texture and structure of the hole. The chapter also describes how the results could be and are evaluated, and the methodology used. Different methods of evaluating quality of images are examined including methods based on Human and formal metrics.

3.1 Images used in the experiments

Images used in our experiments have holes that are surrounded by different types of textures and structures. In practice, we used a large number of images chosen at random, and only a selection of the images (the interesting ones) are discussed and the appendix gives some more. The images have expected difficulty, and this includes size, shape, and position of the hole and complexity of the surroundings of the hole (structure and texture). Also, we decide to use competing technique's images to compare with our corresponding result images. The compared images were taken from their results. We considered these images as a benchmark against which to compare to our method, hence we used the images from the literature. Examples of theses images are shown in Figures 3.1–3.11. Images shown in Figures 3.1–3.5 are used in the method that uses only texture synthesis (Chapter 4) while images shown in Figures 3.6–3.10 are used in the method that combines texture synthesis and structure reconstruction (Chapter 6). The images shown in Figure 3.5 and Figure 3.4 are used in both methods. Most of the images used in the method of using only texture synthesis (Chapter 4) have holes that are surrounded by simple topology, while the images used in the combined method (Chapter 6) have large holes that are surrounded by complex topology.

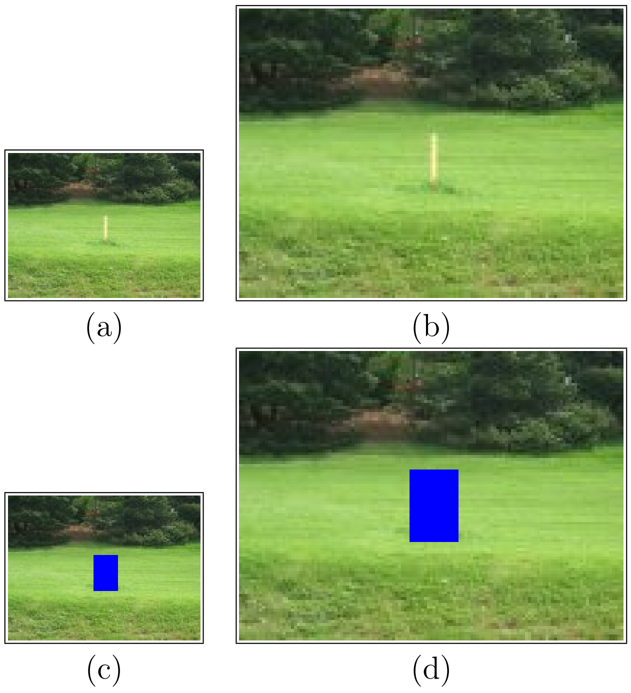


Figure 3.1: The GRASS image. Original image at normal size (a) and enlarged at twice the size (b), (c) original image with its hole at normal size and enlarged at twice the size (d).

Figure 3.1 shows an image (GRASS) which has a hole that is completely surrounded by homogeneous texture (GRASS). Therefore it is expected that the method of image completion will fill in he hole plausibly. In Figure 3.2, however, the surroundings of the hole in the WINDOW image is not completely homogeneous with areas that are locally more homogeneously grey compared to the rest of the brickwork that is usually more varied and of red-ish tones. The result is expected to have variations of both areas. Note that the size of the hole is small and its shape is regular which should make the filling in of the hole easier and more plausible.

The hole of the STREET image in Figure 3.3 spans three different textures corresponding to the bushes, road and grass. Such textures are distinct with well defined, simple boundaries between them. However, the texture and structure of the pavement region on the right of the hole do not match that on the left side of the hole. The completion of such an area would propagate both sides of the pavement into the hole. However this depends on how plausible the initialisation of the hole will be. Note that the hole is narrow which means that the completion of the hole should be easier than for wide holes.

The hole of the BUSH image is surrounded by two distinct areas: the regular brickwork of the wall at the top and the grass area at the bottom. A careful look at the grass shows that in fact it is made of two areas with differently coloured grass (grey at the top, green at the bottom). The topology of the areas around the

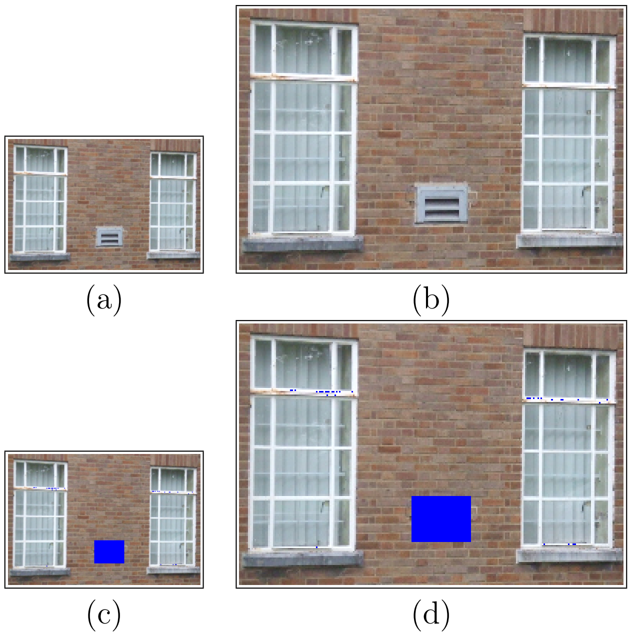


Figure 3.2: The WINDOW image. Original image at normal size (a) and enlarged at twice the size (b), (c) original image with its hole at normal size and enlarged at twice the size (d).

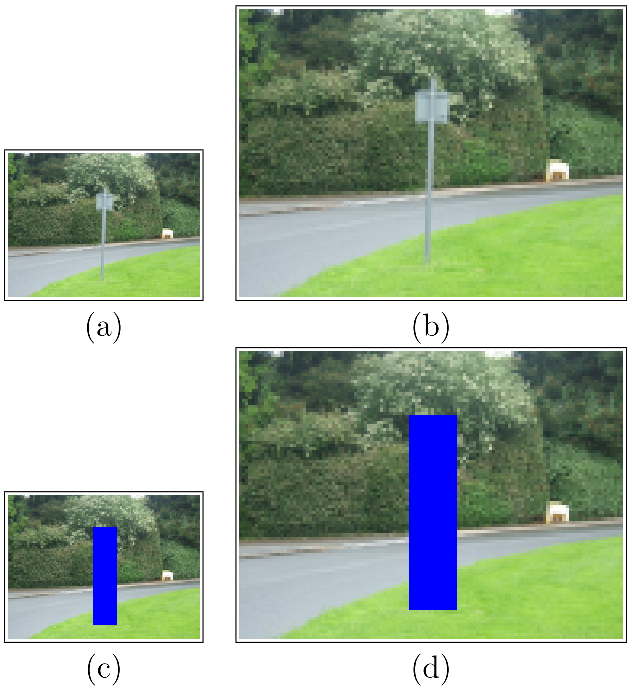


Figure 3.3: The STREET image. Original image at normal size (a) and enlarged at twice the size (b), (c) original image with its hole at normal size and enlarged at twice the size (d).

hole is similar to that of the STREET image. However, in the BUSH image the hole is wider. We therefore expect that the propagation of the grey grass area into the whole might be more difficult compared to the pavement area of the STREET image. Moreover, the green grass area being dominant around the hole, it is likely

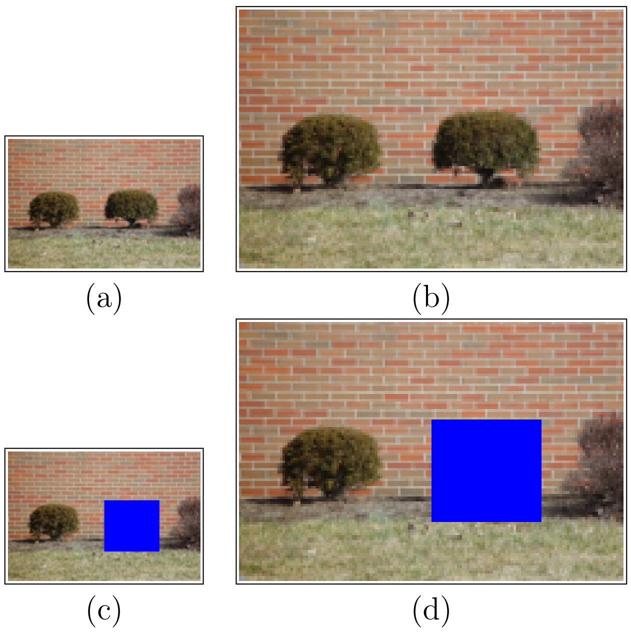


Figure 3.4: The BUSH image. Original image at normal size (a) and enlarged at twice the size (b), (c) original image with its hole at normal size and enlarged at twice the size (d).

that the grey area will not be properly preserved. Also, it is expected that a large neighbourhood size should be used to capture properly the underlining structure of the textures, particularly the brickwork of the wall.

Figure 3.5 shows an image (CLEDWYN) which contains a large hole that is positioned at the edge of the image and surrounded by differently textured areas. Because of these reasons, the completion of the hole is not likely to be simple as the boundaries between these areas are not clear. Therefore, explicit recreation of the boundaries of the hole will be required to create plausible structure that matches the surroundings. Without this explicit creation of the structure, the hole may not be realistic, and also the initial filling would initialise random pixels because of the lack of undefined boundaries between the areas surrounding the hole. This would directly affect the search for the neighbourhood matches.

In Figure 3.6, the hole of the DUSTBIN image covers four types of different textures including sky, bushes, trees and grass. The topology of the areas surrounding the hole looks obvious and therefore, it is expected that the boundaries of the areas inside the hole would be created properly. However, boundaries between the sky and trees areas are very specific and this needs to be recreated well to look plausible. Also, the texture of each area on the left side of the hole looks similar to that on the right side of the hole, therefore, it is likely that these textures will propagate plausibly inside the hole. This though depends on the availability of similar examples in these areas.

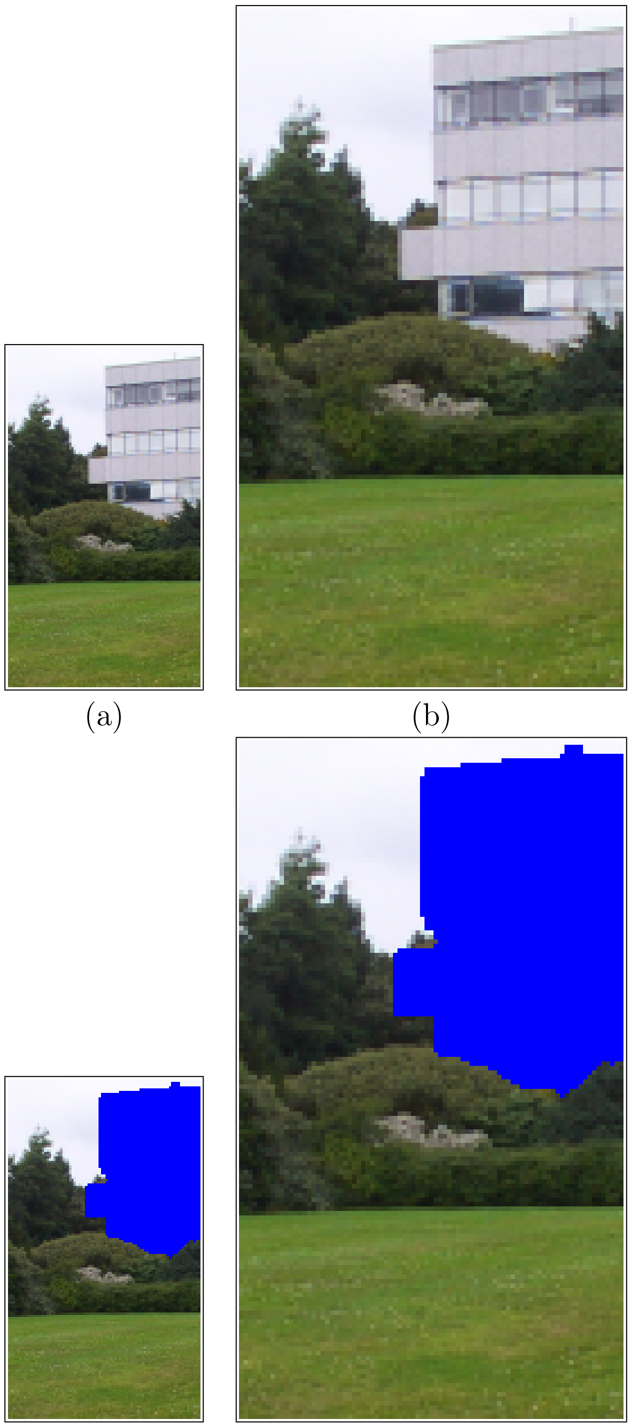


Figure 3.5: The CLEDWYN image. Original image at normal size (a) and enlarged at twice the size (b), (c) original image with its hole at normal size and enlarged at twice the size (d).

Figure 3.7 shows the hole of the PAVEMENT image spanning different textures corresponding to the pavement, kerb and road. In this image, the kerb area around the hole has two areas that have two slightly different levels of grey, one being dry and the other wet. It is expected that the kerb area with light grey would connect

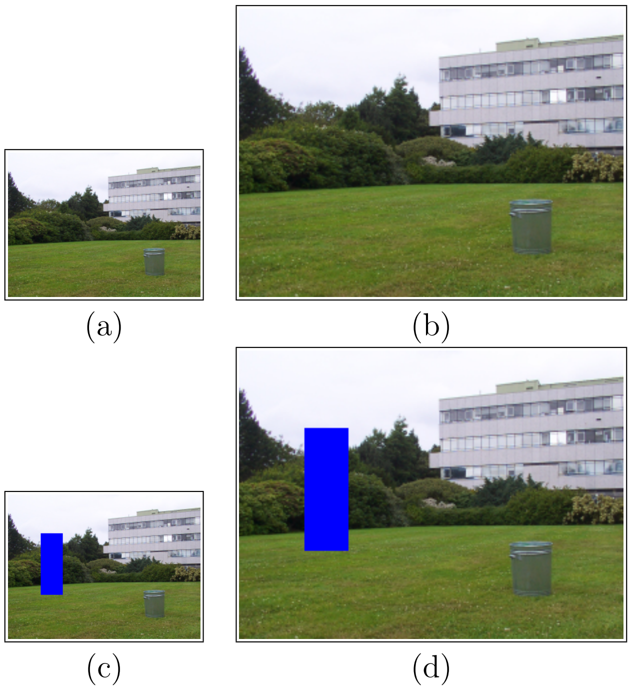


Figure 3.6: The DUSTBIN image. Original image at normal size (a) and enlarged at twice the size (b), (c) original image with its hole at normal size and enlarged at twice the size (d).

to that of dark grey area as the hole is not wide and there are sufficient examples of these areas around the hole. Also, the tiny part of the light gray kerb (on the left of the hole) may be connected to the same area on the right side of the hole, as the initialisation process is expected to have sufficient pixels from such area inside the hole. Textures are likely to be created plausibly in the hole because the surrounding topology is simple, the hole is narrow, and the textures are simple.

In Figure 3.8, a hole from the ROUNDABOUT image is surrounded by differently textured areas. What is interesting about this image is the completion of the area of the hole where the road, grass and roundabout meet. This is because the topology around this area of the hole is not obvious and therefore the completion will depend on the segmentation of the areas where the initialisation will be constrained to. Therefore, it is important to have a plausible initialisation of the area in order to create realistic textures corresponding to these surrounding areas.

In Figure 3.9, the hole of the BUNGEE-JUMBER image is large and covers about 40% of the image. Also, it is surrounded by different types of textures, a house, grounds, grass and river and such textures should be propagated well inside the hole. The structure of the textures should also be captured properly. For example, capturing the structure of the house would require increasing the neighbourhood size to be large enough, which would be prohibitively expensive.

In Figure 3.10, the hole in the TUBE image spans different textures, vertical and

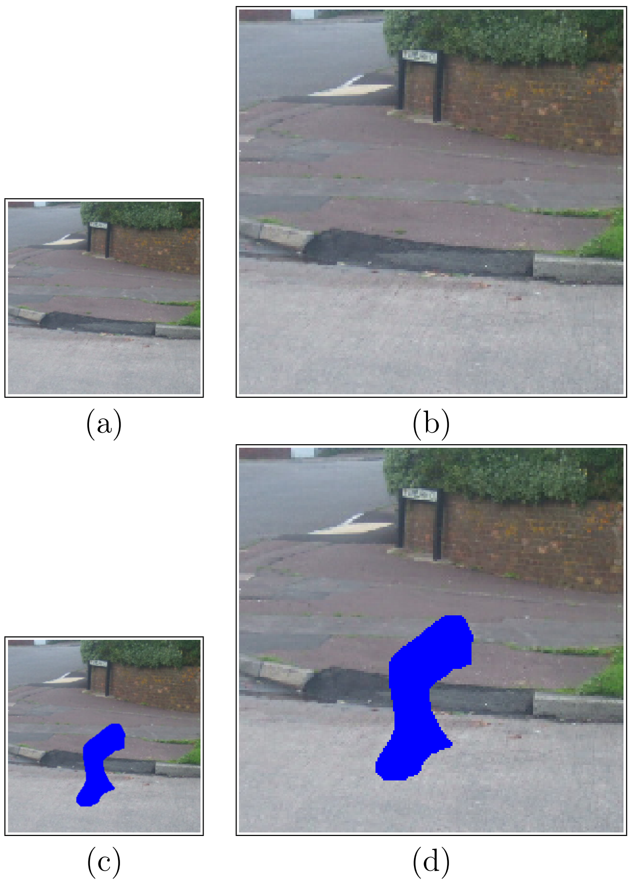


Figure 3.7: The PAVEMENT image. Original image at normal size (a) and enlarged at twice the size (b), (c) original image with its hole at normal size and enlarged at twice the size (d).

horizontal tubes, tree and grass. Most or the hole is expected to be easy to reconstruct because the hole is narrow and surrounded by an obvious topology. However, the part where the horizontal and vertical tubes meet is expected to be more difficult, if not impossible, because there are no examples of such a situation in the rest of the image. In such a case, only high level knowledge about the content of the image would guarantee a good reconstruction, knowledge that is not available to the method but that can be input by a user.

Figure 3.11 shows the ELEPHANT image which contains a large hole that is surrounded by different textures corresponding to the sand, river, shore, trees and mountain areas. Areas of the hole surrounded by the mountain, river, and sand are expected to have plausible filling as the the surrounding topology is not that complex and the area of the hole is not too wide. On the other hand, the area in the middle of the hole is expected to be difficult to complete as the area is wide and therefore will require good initialisation in order to reconstruct its structure and texture.

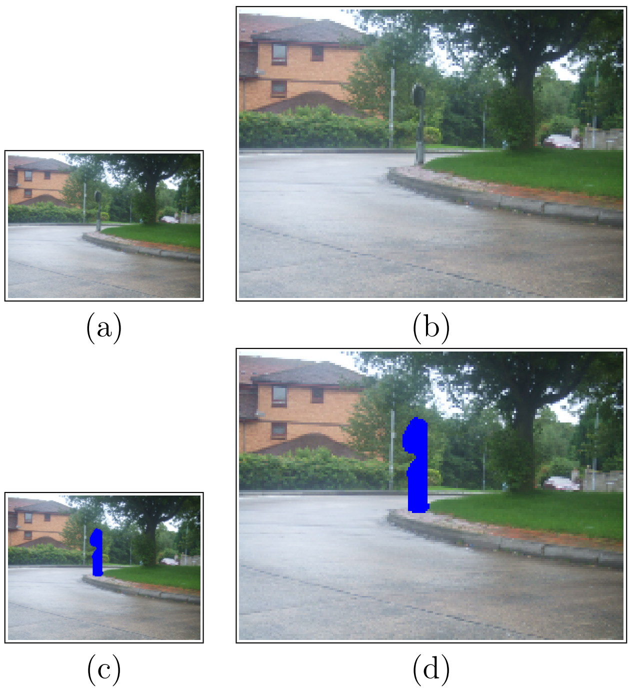


Figure 3.8: The ROUNDABOUT image. Original image at normal size (a) and enlarged at twice the size (b), (c) original image with its hole at normal size and enlarged at twice the size (d).

3.2 Evaluation of the results

3.2.1 Human evaluation

Evaluating image quality is important for many image processing applications, such as compression, acquisition, reproduction and restoration. The evaluation methods can be classified according to the availability of the reference image. Therefore the methods can be categorised into two main types: methods that have a reference image and methods that don't have one. Most methods fall into the first category and have an obvious limitation that the reference image may not be available to the evaluation algorithm. However, methods that predict image quality without a reference image have not been investigated fully, as they concentrated mostly on measuring the blocking artifacts [105].

A popular subjective method for evaluating image quality is to ask a number of observers to give their subjective assessment for the image with a ranking number between, e.g., 1 and 10, where 1 is the worst and 10 is the best. Observers often complain that they don't fully understand what the numbers represent. Showing them with the worst and best images would improve their responses. However, there are factors that can make the evaluation less efficient. For example the amount of variation in the responses of the observers would be obvious as the

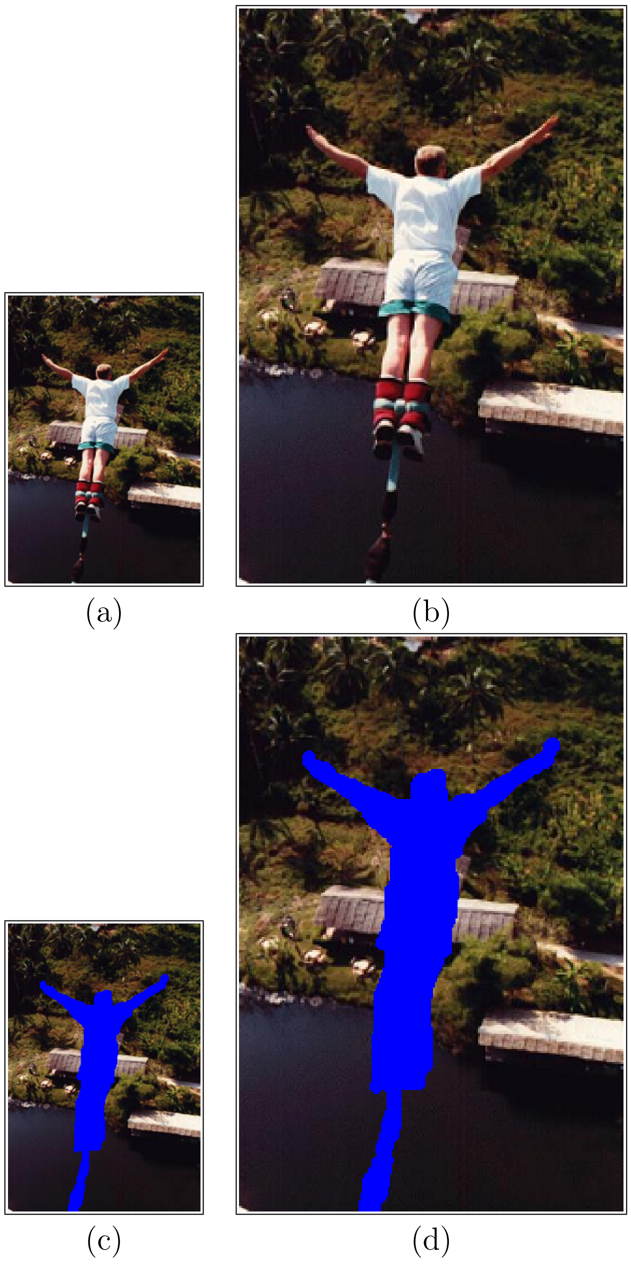


Figure 3.9: The BUNGEE-JUMBER image. Original image at normal size (a) and enlarged at twice the size (b), (c) original image with its hole at normal size and enlarged at twice the size (d).

ranking range is high and images that correspond to each ranking number may not be available. Yet, taking the average of the responses several times with different observers would reduce the variation in the data and the evaluation would be more reliable. This is because each observer may have different experience than others which would help to add more knowledge to the evaluation.

Another subjective method of evaluating image quality is to have a pairwise comparison between two images. Participants in this case would be shown a pair of images and asked to indicate which one of the two "looks visually better". This

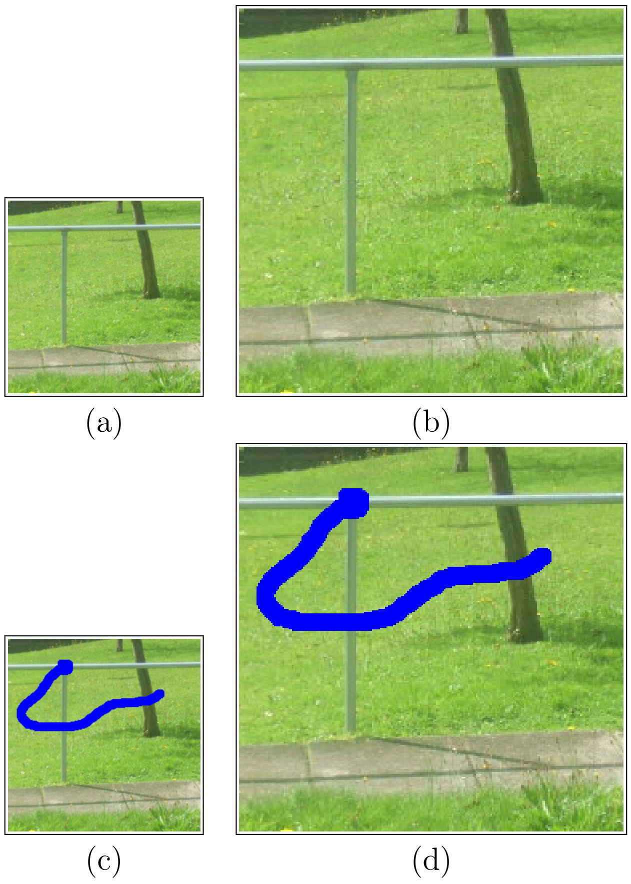


Figure 3.10: The TUBE image. Original image at normal size (a) and enlarged at twice the size (b), (c) original image with its hole at normal size and enlarged at twice the size (d).

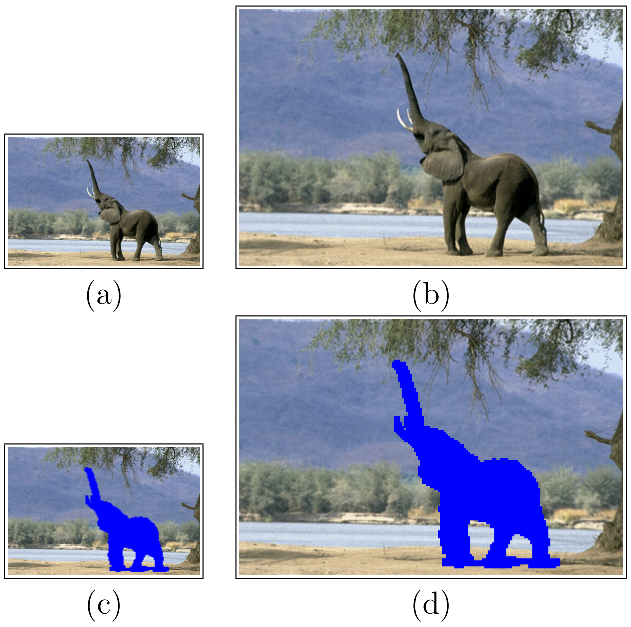


Figure 3.11: The ELEPHANT image. Original image at normal size (a) and enlarged at twice the size (b), (c) original image with its hole at normal size and enlarged at twice the size (d).

method of evaluation doesn't require prior assumptions about the criteria defining their judgments, but introduces factors that can impact judgment.

Although the aforementioned subjective evaluation approach is an important evaluation method to assess the quality of images, such evaluation is time consuming and expensive. Also, the results depend on various factors such as the observers background and motivation. However, such evaluation could be tried, and they might be productive, but they are for future work.

3.2.2 Formal quantitative evaluation

Quantifying visual image quality through subjective evaluation is seen to be more reliable as images are ultimately to be viewed by human beings. However, because of its aforementioned limitations, other evaluation methods can be used such as objective image quality metrics. In the past two decades, several metrics have been proposed for assessing image quality [32,40,87,137].

The most popular objective metrics are the mean square error (MSE) and signal to noise ratio (SNR). The reasons for their popularity are their simplicity to be computed. However, the methods do not generally correlate well with the perceived image quality [40,120] because human perception of artifacts and distortions is not accounted for in the metrics. However, counter examples of this have been published in [55]. As a result, there has been a great deal of interest in incorporating Human Vision System (HVS) characteristics into the image quality metrics. Thus, simple metrics such as the MSE have been improved by incorporating the HVS characteristics into their implementation [103]. Despite the fact that the HVS is a very complex system to fully understand, the incorporation of even simplified models into objective measures lead to a better correlation with human observers response [40].

There are two current approaches for the objective metrics in term of considering the HVS features: error sensitivity and structural similarity based frameworks. The error sensitivity approach aim is to quantify the error between the reference and distorted images. Most objective metrics follow this type of approach, where various HVS features are correlated in the quality assessment. The most commonly used features are luminance contrast sensitivity, frequency contrast sensitivity and masking effects. However, such methods have their own limitations. For example, it is difficult to deal with complex patterns such as in natural images, where most of the experiments conducted used simple patterns. Also, they are inefficient in terms of capturing structures in images [122]. Several methods for the error sensitivity approach have been proposed [89,103,105,121].

The other approach is based on measuring structural similarity [100, 120]. These methods assume that human visual perception is highly adapted for extracting structural information from an image. Therefore, the assessment of images is based on measuring the structural degradation between the reference and distorted images. Unlike the error metrics that are independent of the underlying signal structure, the methods of this approach contain measurements that are adapted to the structures of the reference image signal.

In the context of image completion, the image quality assessment of both approaches for the completed image may not be applicable for the following reasons. A reference image can be available. However, it is not necessary to look similar (in term of structure and texture) to the result image. For example, the original BUSH image has two bushes, the one on the right which is removed and replaced by the surrounding texture, and that one the left of the hole. Our method of completion fills in the hole of that image from the surrounding wall and grass areas rather than from the other bush area, which results in an image that is different from the reference image (grass and wall in the hole, instead of bushes). As a result, our method being random, the chances for the original (reference) image to match the synthesised one are rather small. So any metric that would compare two images should not just look at pixels (such as the MSE) but look at much higher level and subjective information such as "does it look plausible" or "does it look nice", which is very difficult to quantify, let alone evaluate.

3.2.3 Our evaluation approach

Because of the limitations explained in Sections 3.2.1 and 3.2.2, our evaluation of the results is based on individual subjective evaluation of the general quality and plausibility of the completion results. The following describes the evaluation approach.

Visual comparison

The visual comparison between the original and completed parts of given images is based on how plausible the completion is. The plausibility concept evaluates whether the completed part is integrated with the rest of the image. This means that the completed part should be within the context of the rest of the image. Also, it defines whether the structure and texture of the completed image matches the rest of the image. This implies that the completed image part should be indistinguishable from the rest of the image and without visible artifacts.

Pairwise comparison

Our results are compared to corresponding results from a variety of image completion methods. The comparison is done subjectively by perceptually evaluating the quality of the compared results. The images are also subjectively assessed according to their plausibility.

Chapter 4

Texture synthesis method for image completion

4.1 Introduction

The main goal of image completion is to automatically fill in the area of the image corresponding to an undesired area such that the synthesised part plausibly matches the surrounding areas. Typically, such completion propagates new texture from the surrounding areas into the hole by "copying" similar examples from such areas. This propagation should produce a realistic structure and texture inside the hole with no visual artifacts. Figure 4.1 shows an example of the texture synthesis completion.

Texture synthesis methods create a model of the texture and use that model to produce a new texture having the same properties as the original texture. The model can be statistical as in [15,57,97] or exemplar as in [3,38,125], and both cases being derived from the original texture. In other methods, the model does not contain the original texture, but its sufficient statistics [49,134]. All these methods assume that a single texture is used and that it is homogeneous. Therefore, it is expected that using such methods for completing holes in images will not work properly with images that contain many different textures.

Section 4.2 will briefly describe the generic method on which the currently most popular texture synthesis methods are based. Sections 4.3 to 4.6 will describe our modifications to this generic method such that it can be used in the context of image completion as defined above. Finally, Section 4.7 will present and discuss results of applying our method to a variety of images introduced in Chapter 3.

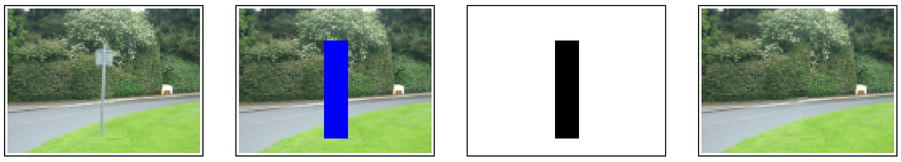


Figure 4.1: Example of the texture synthesis completion: (a) original image, (b) original image with the hole masked, (c) hole mask and (d) final result of the texture synthesis completion.

4.2 Generic method

Our texture synthesis method is based on the work proposed by [38], and it is a variation on a generic method presented here. In this generic method, pixels being synthesised are given a value by matching their neighbourhood to similar pixel configurations elsewhere in the image. In the context of image completion, possible matches are considered only outside the hole. The neighbourhood of the current pixel is compared with the neighbourhood of possible matches to build a list of good matches. From that list, one match is selected and its centre value is used as the new value for the current pixel.

The comparison is usually done using the normalised Sum of Squared Differences (SSD) between the two neighbourhoods and a good match is therefore one for which the distance to the current pixel's neighbourhood is low. "Low" is defined in various ways. In [38], this was set below a threshold defined as a proportion of the lowest distance. In [125], only the best match was kept while in [72], a Gaussian distribution was used to select at random a match close to the best match.

The problem with the use of fixed thresholds as in [38] is that the randomness of the created texture is dependant on the value of the threshold. Smooth textures require a lower threshold than rough textures. Therefore, if the threshold is too high, the texture will look too random. At the other extreme as in [125], selecting the best match (or very low threshold) can lead to replicating complete patches of texture [72]. Also, this fixed threshold used in [38] can create unrealistic boundaries between regions [72].

The improvement proposed in [72], namely the use of a Gaussian distribution to randomly choose a good match similar to the best match, constitutes a soft threshold rather than a fixed, hard threshold. This does help in dealing with many different textures in a single image, but does not guarantee that the best match is not often selected, and therefore that entire patches are not replicated.

Our approach uses an iterative synthesis scheme starting with a high value of

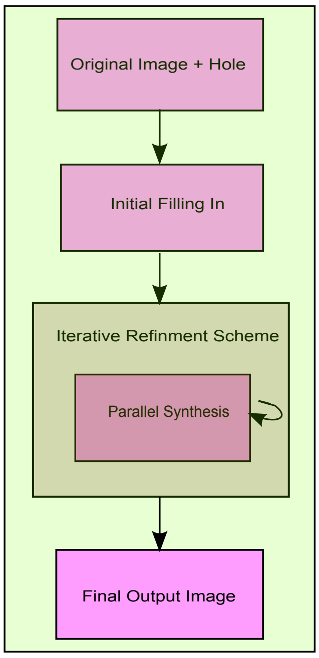


Figure 4.2: An outline for texture synthesis procedures.

the threshold which is then progressively lowered. This refinement scheme allows sufficient randomness and avoid replication of complete patches (see Section 4.5).

During the synthesis process, pixels are considered in some order such as raster scan [125] or spiral from a central seed in the case of unconstrained texture synthesis [38]. Problems with such synthesis orders can occur. For example, the raster scan order leads to directional bias in texture propagation which can propagate certain areas at the expense of others. Also, this synthesis order can produce boundary artifacts [125]. On the other hand, when using the spiral synthesis order, a boundary can remain visible at the "centre" of the spiral (see Section 2.3.4).

We propose in the following sections our solution to limitations of this generic method. The processes used in our solution are outlined in Figure 4.2 and explained in the following sections. A pseudocode for the texture synthesis method is described in Algorithm 1.

4.3 Parallel synthesis

The order in which an image is synthesised plays an important part as it can directly affect the quality of the output image, Section 2.3.4. Different synthesis orders produce different synthesised textures because pixels being synthesised always depend on previously synthesised pixels, causing "cyclic dependencies" among the texture pixels [126].

Algorithm 1 Pseudocode for the texture synthesis method

```
Initialise the hole as described in Algorithm 2.
DO
   For each pixel in the hole
     For each pixel outside the hole
       Neighbourhood template = get neighbourhood(current pixel)
       IF the Neighbourhood template have a number of pixels equal to
       or greater than the number of pixels in the current neighbourhood
         Compute Euclidean distance for the two neighbourhoods
         Save the location of the pixel and its neighbourhood distance in
         a NH list
       ENDIF
     ENDFOR
       Possible matches list = all pixels (i,j) in the NH list where Euclidean
       distance(i,j) <= Min(Euclidean distance) * randomness factor
       Best match = Pick up randomly a mach from the Possible matches list
       Pixel value = the value of the pixel at the centre of the Best match
       Copy the Pixel value to the current pixel in the hole
   ENDFOR
   Decrease the randomness factor by a value at each iteration
While ( the ratio of changed pixels of the current and previous iteration results
is greater than P)
```

A new order is proposed here to solve the above problems: all pixels are synthesised in parallel in an iterative scheme. At each iteration, the value of any given pixel is independent of the value of the other pixels and is therefore not affected by the order in which the pixels are considered. This approach has been mainly developed to reduce the directional bias and the propagation of some regions at the expense of others. The strength of this method is that hole pixels can be synthesised independently from the rest of the already synthesised pixels. Thus it is an order-independent synthesis.

In practice, a new image (temporary buffer) holds the values of the synthesised pixels. This temporary buffer is then copied back to the image being synthesised at the end of each iteration, when all the pixels have been processed. Figure 4.3 shows the parallel synthesis process.

In order for the parallel synthesis to work, it needs an initial seed. The idea of placing an initial seed for non-parametric texture synthesis was introduced in [38]. Also, previously published work [125] has also used initial filling in, but random uniformly distributed noise was used. This is discussed in Section 4.4.

During the parallel synthesis, the neighbourhood comparison is done using the Euclidean distance between the two neighbourhoods and a good match is therefore one for which the distance to the current pixel's neighbourhood is below a

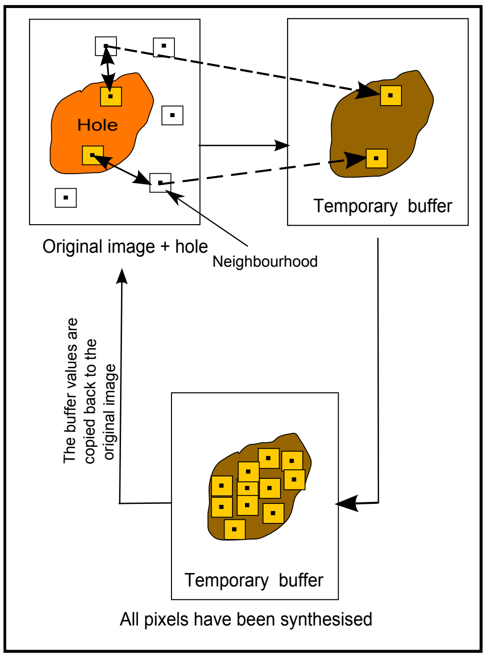

Hole
Temporary buffer
Original image + hole
Neighbourhood
The buffer values are copied back to the original image
Temporary buffer
All pixels have been synthesised

Figure 4.3: Parallel synthesis for hole pixels at each iteration.

threshold. One match is selected and its central pixel is copied to the current pixel in the buffer image.

This method is similar to the order-independent synthesis method presented in [126]. However, that method is based on multi-resolution synthesis and is used in the context of pure texture synthesis. Our method uses single resolution synthesis and is used in the context of image completion. For the order independent synthesis to work, it needs an initialisation of the texture. In [126], pixels are copied randomly from the lowest resolution of the image pyramid, and this initialisation completely determines the texture synthesis result as each synthesised pixel is only determined by the pixels at the lowest resolution [126]. Initialisation in our method is done using plausible values rather than random noise (Section 4.4).

4.4 Initial hole filling

Natural textures often have some degree of randomness and therefore capturing this characteristic is important for producing realistic textures. The aim here is to provide initial values in the hole that have a certain amount of randomness. However, the values must be plausible. They are therefore copied from nearby pixels of the image that do not belong to the hole, and in the same region of the current pixel being processed (when image segmentation constraint is used, see Chapter 6). This is done using a Gaussian distribution centred on the current pixel. The distribution allows us to obtain random positions close to the current pixel, depending on the value of the standard deviation of the distribution. It is first set to 1 pixel, implying that 99.7% of the positions will be 3 pixels away from the current pixel. If after a number of randomly selected positions (N = 1000), none fall outside of the hole, the standard deviation is increased (by one) and the process is repeated until one position is found outside the hole. In practice, we found that 1000 produced an acceptable compromise between accuracy and speed. Figure 4.4 shows the initial filling in for a given pixel. A pseudocode for the initial filling in is described in Algorithm 2.

Algorithm 2 Pseudocode for Initial hole filling in

```
rndgenr = new Random()
For each pixel in the hole
    xpos = current x position
    ypos = current y position
    WHILE (xpos and ypos inside the hole)
        IF $i \geq N$
            xstdev = xstdev +1
            ystdev = ystdev +1
        ENDIF
        xpos = (xpos + (xstdev * rndgenr.nextGaussian()))
        ypos = (ypos + (ystdev * rndgenr.nextGaussian()))
        i = i+1
    ENDWHILE
    Image(x,y) = Image(xpos , ypos)
ENDFOR
```

In our work, we use a symmetrical neighbourhood during the matching process, see Section 4.6, instead of using, say, an L-shaped neighbourhood as in [125], which was reported to cause boundary artifacts. Indeed, the symmetric neighbourhood contains more available, plausible, values compared to the L-shaped neighbourhood, and therefore provides a better selection mechanism. It is possible to use such neighbourhood in our case because the initial filling in provides plausible values for all the pixels in the hole. Note that in [125], initial filling in was also used but was made of uniformly distributed random noise (matching the colour statistical properties of the whole image), which was introducing uncertainty in the neighbourhood matching and was therefore only used to synthesise the first few rows of the images.

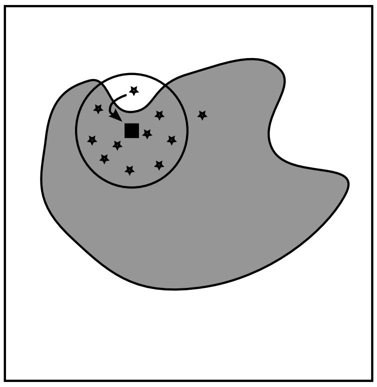


Figure 4.4: Initial filling in for a given pixel. Stars show tentative pixels for copy, while the black square shows the current pixel. The radius of the circle is three times the standard deviation.

4.5 Iterative synthesis scheme

Iterative synthesis has been used in previously published work with different purposes. In [125], Wei and Levoy used two passes as a modification to their original method for constrained texture synthesis. Ashikhmin [3] used several passes to improve on the quality of his user-controlled texture synthesis method. Also, in [126], multiple synthesis passes were used for generating texture. With patch based synthesis, Efros and Freeman presented a texture transfer method based on using an iterative synthesis process [37]. In this work, an iterative refinement scheme is also used as an essential part of the method.

The selection of good matches is important for the quality of the results (Section 4.2). This selection must ensure that the generated texture has sufficient, but not excessive, randomness and does not have replications of entire patches. To achieve this, we iteratively fill in the hole (using the parallel method described above for each iteration), first allowing a wide set of good matches, then reducing the size of the set. This refinement scheme allows global random initial texture structure that progressively converges towards good quality fine detailed texture. More precisely, the threshold T used in the matches selection is initially set to:

$$T = d \times r, \tag{4.1}$$

where d and r respectively represent the Euclidean distance for the best match and the randomness factor, and $r = 1.2$ at the first iteration. The threshold is then reduced by decreasing the value of r by 0.01 at each iteration. A list of matches which are below this threshold is considered and then one match is randomly selected and its centre pixel is used as the new synthesised value for the current pixel. Other values of r such as 1.3 and 1.1 have been used, but were ruled out as they produce too random texture or insufficient randomness in the texture. Also other values for the reduction of r such as 0.001 has been tried, but were ruled out as gradual convergence of the texture was not maintained. The retained values represent a good compromise.

The automatic termination of the iterative synthesis can be based on using the difference in the Euclidean distance between the images of the current and previous iteration results, as reported in [3, 107]. However, the Euclidean distance is not a good measure of visual quality as the behaviour of visual similarity between two synthesised textures can vary depending on the particular texture [3]. As a result, we are not interested on how much the difference is, but whether the pixels in the hole are different. Thus, the termination of the iterative process is based on computing the number of changed pixels in the hole compared with the previous iteration. The comparison is done using the complement of Kronecker delta function defined as:

$$\bar{\delta}(i,j) = \begin{cases} 0, & i = j \\ 1, & i \neq j \end{cases} \tag{4.2}$$

where i and j are the pixel values for the current and previous iteration results respectively. Note that i and j are on the same position in the hole. The total number of changed pixels is normalised by the total number n of pixels in the hole as follows:

$$P(t) = \frac{\sum_{i,j=0}^{n-1} \bar{\delta}(i,j)}{n}, \tag{4.3}$$

where $P(t)$ is the ratio of changed pixels for the current iteration t. If P is below a threshold, the iterative process is stopped. The threshold is set to 0.1 for all images synthesised as described in this chapter and Chapter 6. The segmented images synthesised as described in Chapter 5 require a lower threshold (set to 0.002) because of their nature (colour corresponding to labels rather than texture). In practice, we found that the thresholds of 0.1 and 0.002 produced acceptable results for varieties of segmented and textured images respectively.

Figures 4.5 and 4.9 show the convergence of the iterative synthesis using only texture synthesis, and both texture synthesis and structure reconstruction respectively. As shown in these figures, the number of changed pixels decreases rapidly until iteration 20 and then continues to reduce and rise within a small range of values where the change is not significant. Figures 4.6, and 4.10 show few result

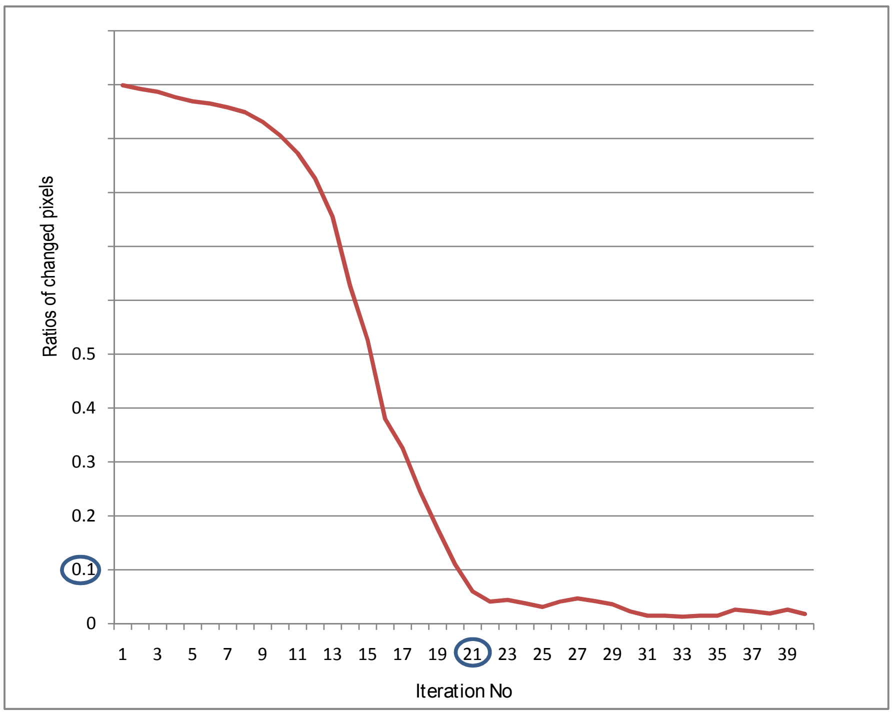


Figure 4.5: Convergence of the iterative synthesis for the BUSH image using only texture synthesis: the iterative synthesis process is stopped at iteration 21 where the ratio is lower than a threshold of 0.1

images before and after the iterative synthesis would have been stopped. Significant change occurred in the images before the iterative synthesis would have been stopped while only few number of pixels changed after the iterative process would have been terminated showing that this change is visually not significant.

Figure 4.7 shows the convergence of the iterative synthesis for structure reconstruction (Chapter 5). As shown in the figure, the number of changed pixels decreased significantly, but with fluctuations at iterations 13 and 17. From iteration 20 onward, the number of changed pixels is 0. Figure 4.8 shows a few result images after the iterative synthesis would have been stopped where only few number of pixels change, and this change is not significant.

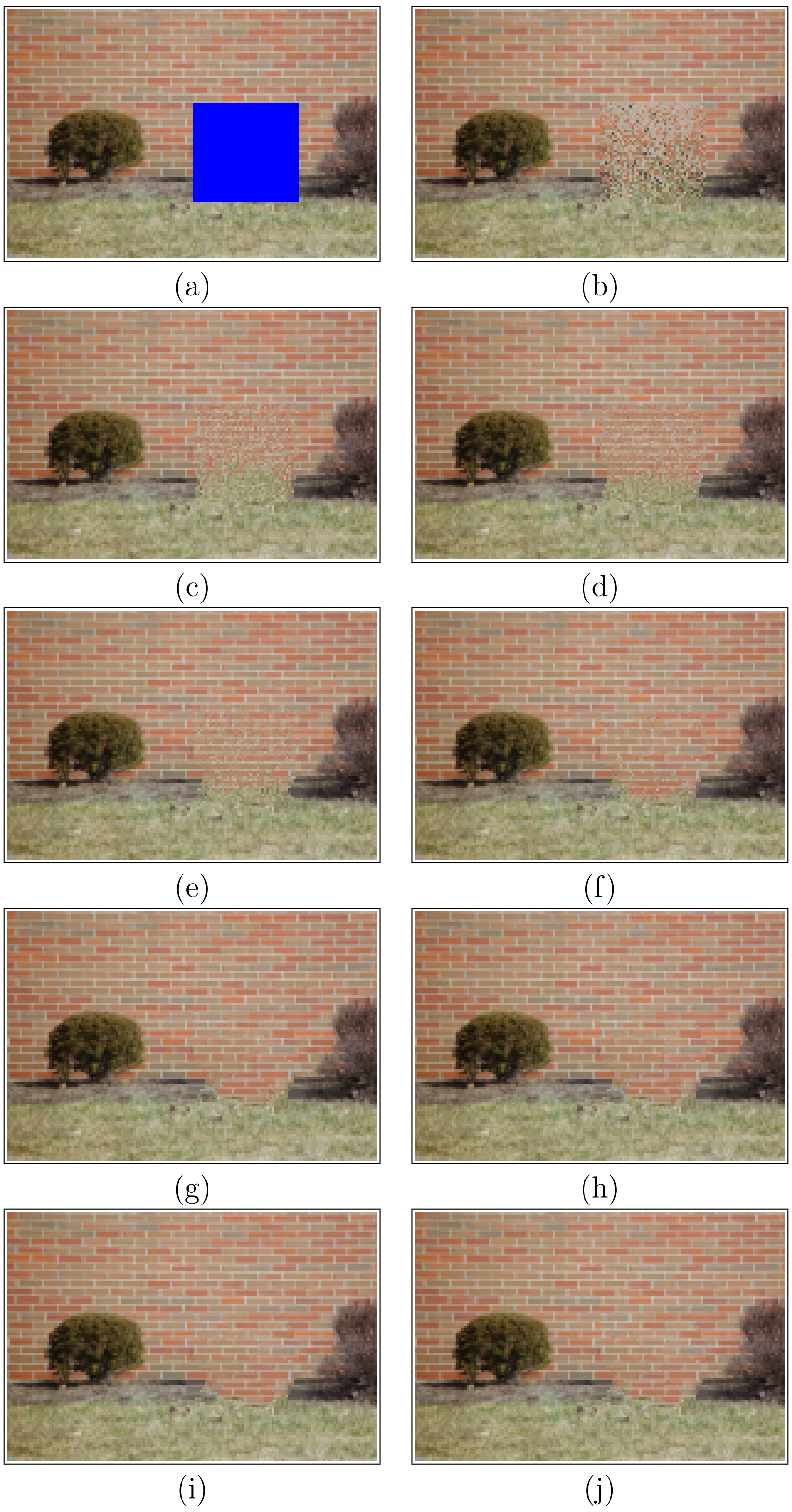


Figure 4.6: Results of the iterative synthesis before and after the stopping criterion using only texture synthesis: (a) the original image with the hole, (b) result of the initial filling in, (c) – (g) results for iterations 3, 7, 11, 15 and 19, (h) result for iteration 21 where the process would have been stopped and (i) and (j) results for iterations 22 and 25.

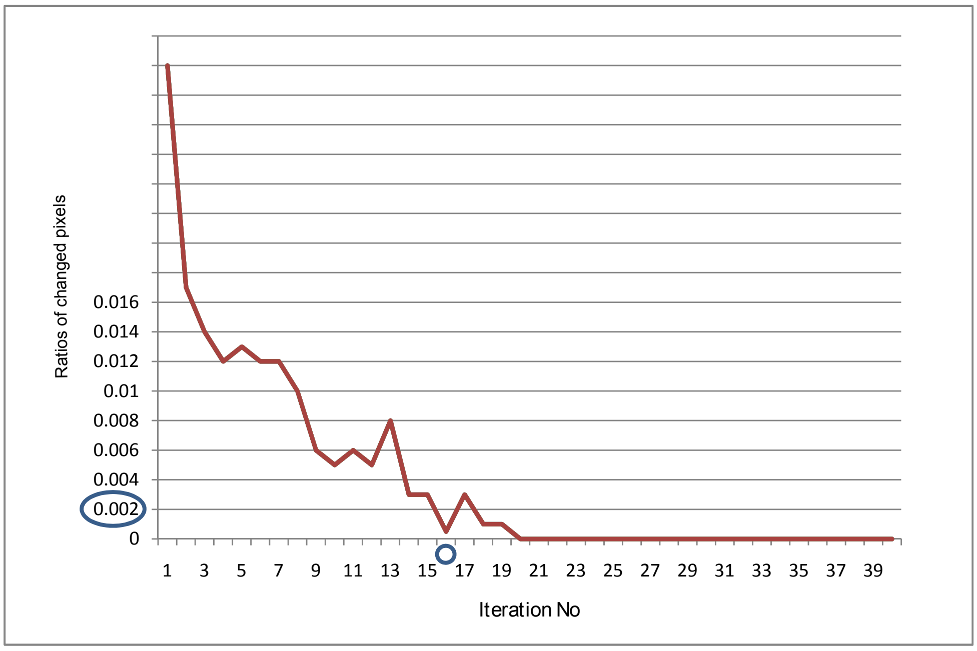


Figure 4.7: Convergence of iterative synthesis for the BUSH image for structure reconstruction: the iterative synthesis process is stopped at iteration 16 where the ratio is lower than a threshold of 0.002

4.6 Neighbourhood characteristics

4.6.1 Size and shape

The quality of the synthesised texture depends on the size and shape of the neighbourhood used in the search for possible matches. This size should be large enough to capture the underlying structure of the texture elements, otherwise the structure of the texture will be lost. At the same time, having a very large neighbourhood size may encourage too much region growing [3]. In our work, the neighbourhood size is set by the user and its default value is set to 11, unless otherwise specified.

The shape of the neighbourhood is also important as it can affect the result of the synthesis. In our method, the shape of the neighbourhood plays a different role from that in the original method used in [125]. In the original method of [125], it is essential to have a "causal" neighbourhood (one that contains only already synthesised pixels) or otherwise, the generated texture will be unrealistic. This is because a non-causal neighbourhood contains non-synthesised pixels (noise) which introduces uncertainty in the search for the best match. The method is later modified to ensure the use of a causal neighbourhood by using multi-resolution neighbourhood

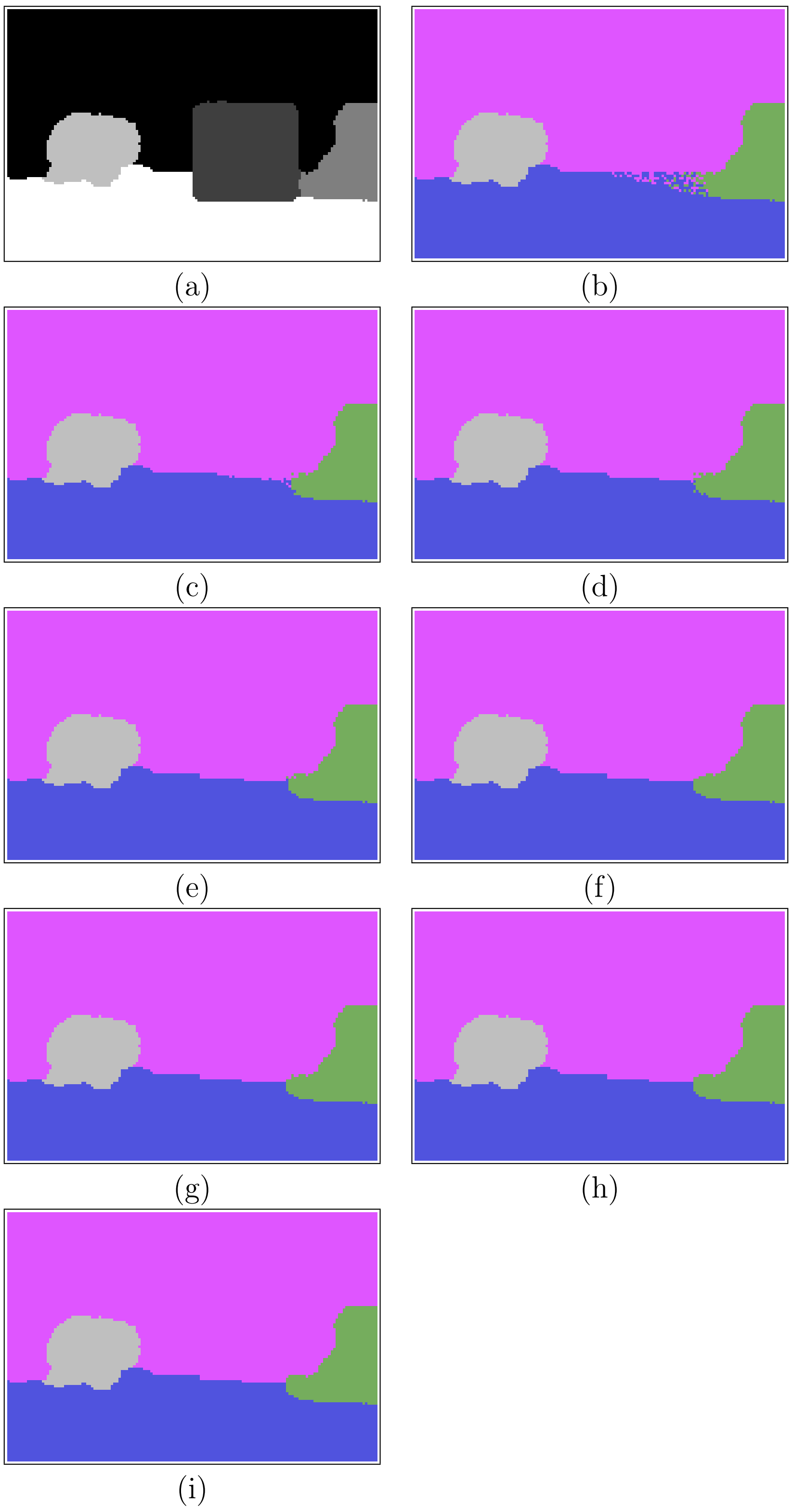


Figure 4.8: Results of the iterative synthesis before and after the stopping criterion using only texture synthesis: (a) the original segmented image with the hole, (b) result of the initial filling in, (c) – (f) results for iterations 3, 7, 11, and 15, (g) result for iteration 16 where the process would have been stopped and (h) and (i) results for iterations 17 and 20.

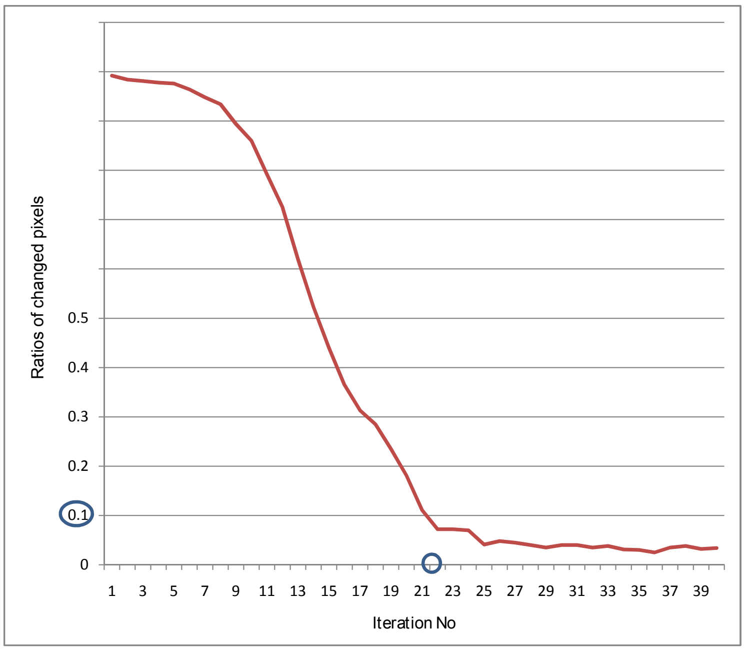


Figure 4.9: Convergence of iterative synthesis for the BUSH image for both texture synthesis and structure reconstruction: the iterative synthesis process is stopped at iteration 22 where the ratio is lower than a threshold of 0.1

in which texture synthesis is performed in a lower to higher resolution pyramid. In contrast, our method uses a non-causal (symmetric) neighbourhood without the need for the multi-resolution synthesis. This is done by the initialisation of the hole area with plausible values, which are included in the neighbourhood where the synthesis is done in parallel and iteratively.

4.6.2 Search area

In texture synthesis methods, identifying the search area for the neighbourhood match is important since it affects the quality and the speed of the synthesis process. Methods that use an exhaustive search (all pixels outside the hole are considered) are slow but ensure that relevant pixels are not excluded [38, 125]. Others restrict the search area using a fixed "belt" surrounding the hole area as in [101] or using a mask for each hole pixel and such mask area is determined by a fixed number of pixels outside the hole [17]. Such methods are reported to be faster. However relevant regions may be excluded in this type of search as it is difficult to decide if the search area is sufficient for producing realistic texture.

Our method is not designed for real time applications and therefore the focus of

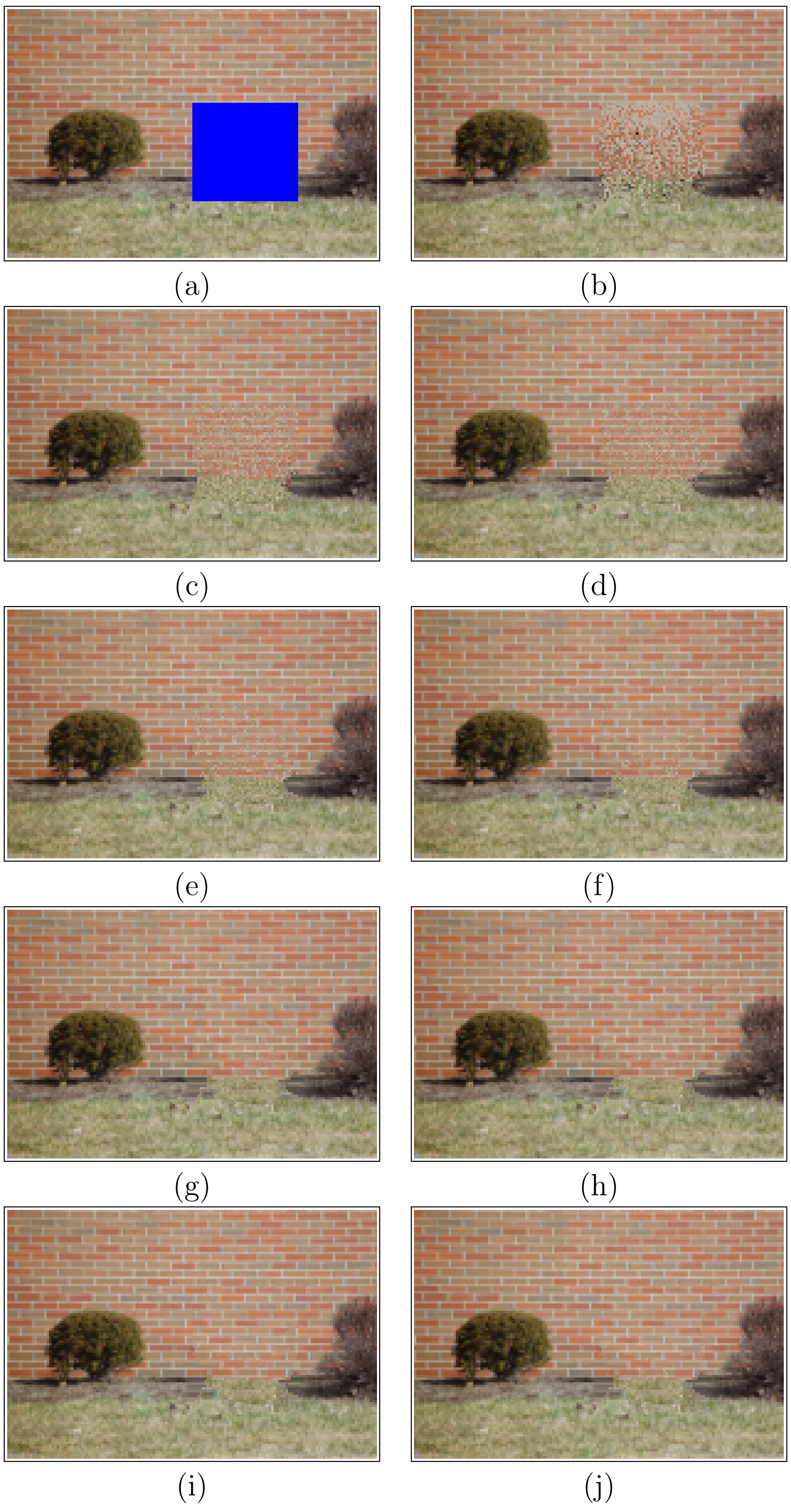


Figure 4.10: Results of the iterative synthesis before and after the stopping criterion using both structure reconstruction and texture synthesis: (a) the original image with the hole, (b) result of the initial filling in, (c) – (g) results for iterations 3, 7, 11, and 15, (h) result for iteration 22 where the process would have been stopped and (i) and (j) results for iterations 23 and 26.

this work is on the quality of the result rather than the speed. Thus, the search for source neighbourhood matches is done exhaustively over all pixels outside the hole. The search area is later improved by constraining it by the synthesised segmented image where only relevant regions are included, as discussed in Chapter 6.

Pixels near the image boundaries can affect the quality of the match in the sense that they only include partial neighbourhoods that may not be well matched with the target neighbourhood, and can cause boundary artifacts. A possible solution to this is to use circular indexing. For example, if $I(x, y)$ denotes a pixel at position (x, y) of the image I, then $I(x, y) == I(x \mod r, y \mod c)$ where r and c are the number of rows and columns respectively. However, using this edge handling method can include irrelevant pixels from different regions on the other side of the image. The other possibility is to use reflected indexing where each pixel outside the image can reflect itself back onto the image. However, this can cause uncertainty in the neighbourhood due to the reflected pixels. Also, another solution is to use only neighbourhoods that are completely inside the image, but this ignores pixels that are on the edge of the image, particularly when the neighbourhood size is large.

As a result, in order to keep as much information as possible, neighbourhoods matching the current neighbourhood are selected not only based on their Euclidean distance (Section 4.3) but also based on the number of pixels they have inside the image; only neighbourhoods that have a number of pixels equal to or greater than the number of pixels in the current neighbourhood are considered.

It is also important to note that only considering potentially matching neighbourhoods that are completely outside the hole can exclude pixels near the hole that could be good potential candidates for pixels in the hole. Therefore, neighbourhoods that have pixels in the hole but their central pixel outside of it are considered as potentially matching neighbourhoods.

Also, if the hole is touching the image border, some neighbourhoods in the hole may not be complete. Therefore, the method will only compare the pixels that are inside the image taking into account the aforementioned constraint. This gives less priority to pixels that are on the image border.

4.7 Results and discussion

The method of image completion using only texture synthesis has been applied to different types of textures and has produced acceptable results as shown in Figures 4.11–4.14. However, the method has its own limitations as shown in the

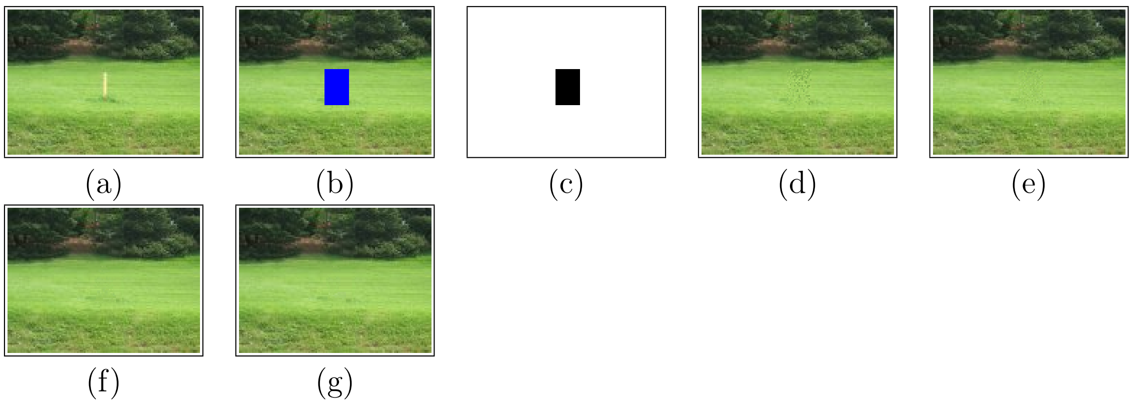


Figure 4.11: Texture synthesis result for the GRASS image: (a) original image, (b) original image with the hole masked, (c) hole mask, (d) initial filling in of the hole area, (e) first iteration of texture synthesis, (f) middle iteration of texture synthesis and (g) final synthesised result.

examples of Figures 4.16, 4.18.

Figure 4.11 shows the GRASS image where the hole is completely surrounded by homogeneous texture (grass). Intuitively, the method should easily handle holes that are completely contained in one homogeneous texture. In this case, the surrounding texture is indeed well propagated inside the hole.

In Figure 4.12 the hole of the WINDOW image is surrounded by structural texture which is not completely homogenous with areas that are locally more homogeneously grey compared to the rest of the brickwork that is usually more varied and of red-ish tones. The texture is propagated well in the hole with the locally dominant grey-ish texture from the top-right corner of the hole. On the other hand, the overall structure of the wall is plausibly reconstructed, yet the area in the middle does not appear to have a clear structure. This is due to the uncharacteristic feature of the wall around the hole. Figure 4.13 is a zoom in of the hole and parts of the surrounding areas of the original and final synthesised images.

In Figure 4.14, the hole of the STREET image spans three different textures (corresponding to the bushes, road and grass, which themselves in fact contain several distinct textures) with well defined, simple boundaries between them. As a result, the texture in the hole is well reconstructed and therefore looks natural. The method is able to propagate well the overall structure of the surrounding of the hole without using the explicit structure propagation discussed in Chapter 5. This due to the fact that the hole is horizontally narrow and not surrounded by complex regions, but instead by completely different homogenous regions, hence leading to more accurate initial filling and parallel syntheses. Figure 4.15 is a zoom in of the original and final synthesised images.

In Figure 4.16, the hole of the the BUSH image is closely surrounded by two distinct

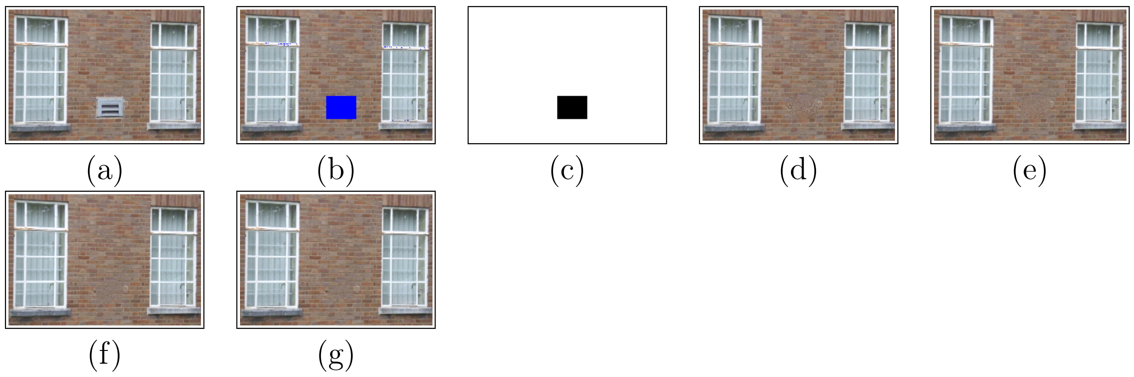


Figure 4.12: Texture synthesis result for the WINDOW image: (a) original image, (b) original image with the hole masked , (c) hole mask, (d) initial filling in of the hole area, (e) first iteration of texture synthesis, (f) middle iteration of texture synthesis, (g) final synthesised result.

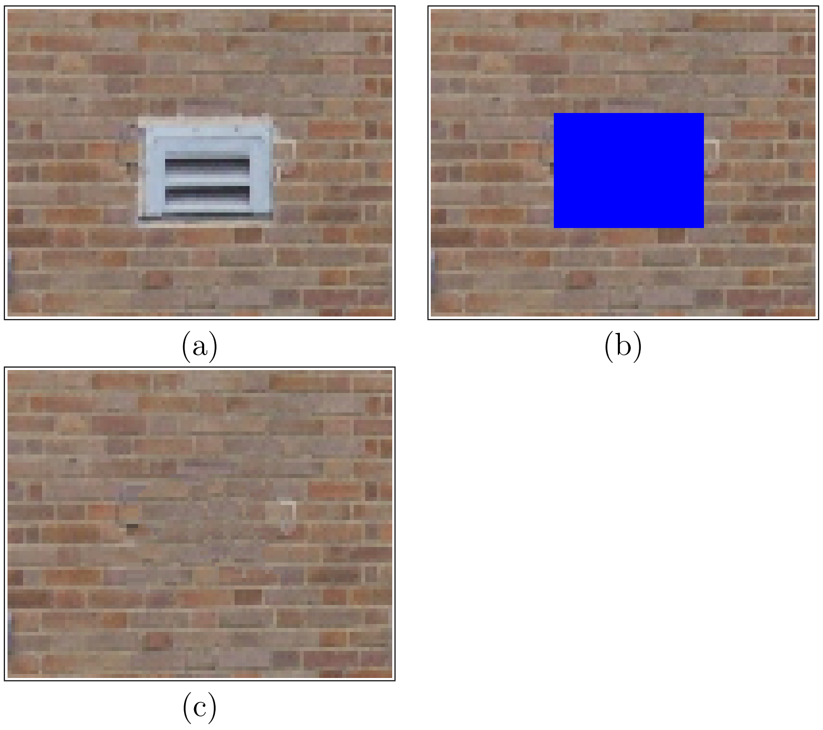


Figure 4.13: Zoom in of the hole and parts of the surrounding areas of the WINDOW image: (a) original image, (b) original image with the hole and (c) final synthesised image.

regions (wall and grass) where the structure of the wall in itself is reconstructed well (the vertical and horizontal alignments), but the wall texture is expanded over the grass area causing the structure of the image to look unrealistic. This is because this method is only designed to propagate texture information, and does not explicitly handle propagation of image structure. In this example, the texture synthesis alone was not able to create proper boundary between the wall and the "brown" grass regions because the initialisation of the hole does not contain enough pixels from the brown grass area at the bottom of the hole, but wall and "green"

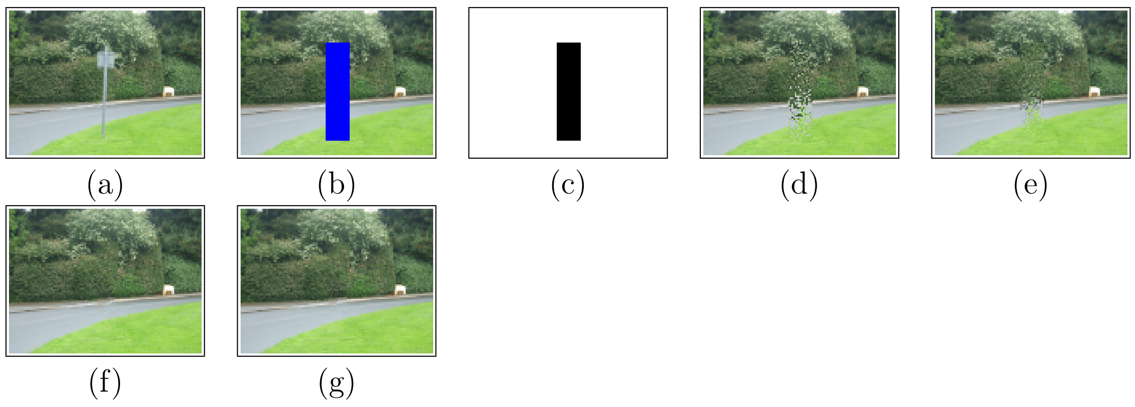


Figure 4.14: Texture synthesis result for the STREET image: (a) original image, (b) original image with the hole masked, (c) hole mask, (d) initial filling in of the hole area, (e) first iteration of texture synthesis, (f) middle iteration of texture synthesis and (g) final synthesised result.

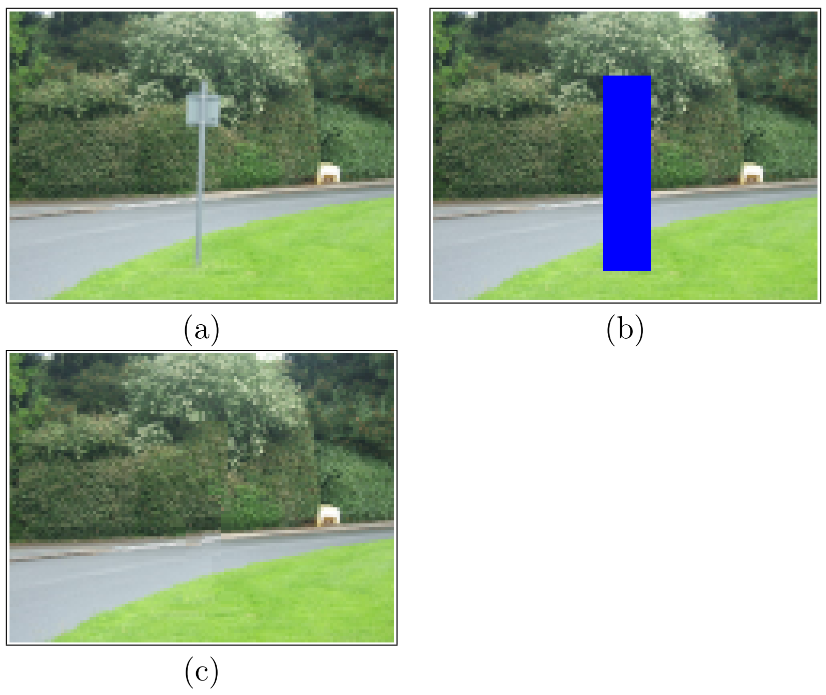


Figure 4.15: Zoom in of the original image, hole image and final synthesised image of the STREET image: (a) original image, (b) the original image with the hole and (c) final synthesised result.

grass pixels are dominantly copied.

As a result, the patches in this area of the hole mainly contain too many random wall and green grass pixels. This configuration of wall and green pixels in such patches do not exist in the examples outside the hole. Thus, the method only produces the closest matches to this configuration which is the wall pixels as they are more available in these patches than the green grass pixels. This problem is solved by constraining the initial filling and texture synthesis to the "correct" regions, see Chapter 6. Figure 4.17 is a zoom in of the original and final synthesised

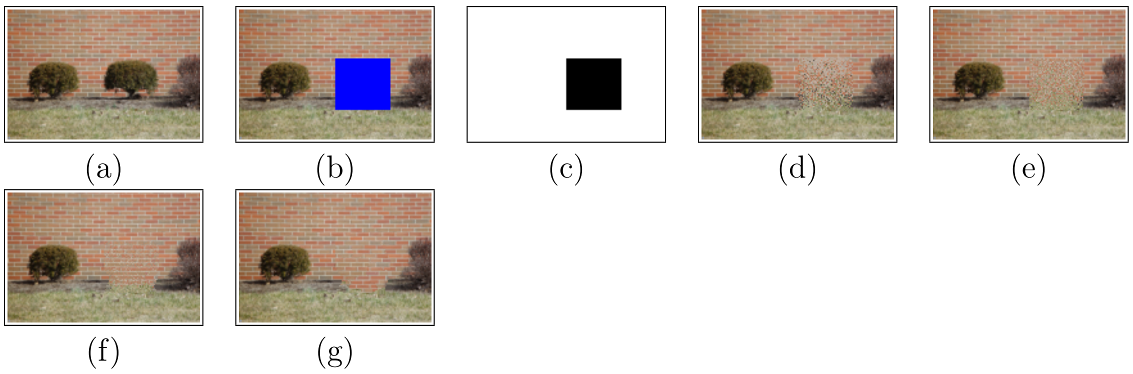


Figure 4.16: Texture synthesis result for for the BUSH image: (a) original image, (b) original image with the hole masked, (c) hole mask, (d) initial filling in of the hole area, (e) first iteration of texture synthesis, (f) middle iteration of texture synthesis and (g) final synthesised result.

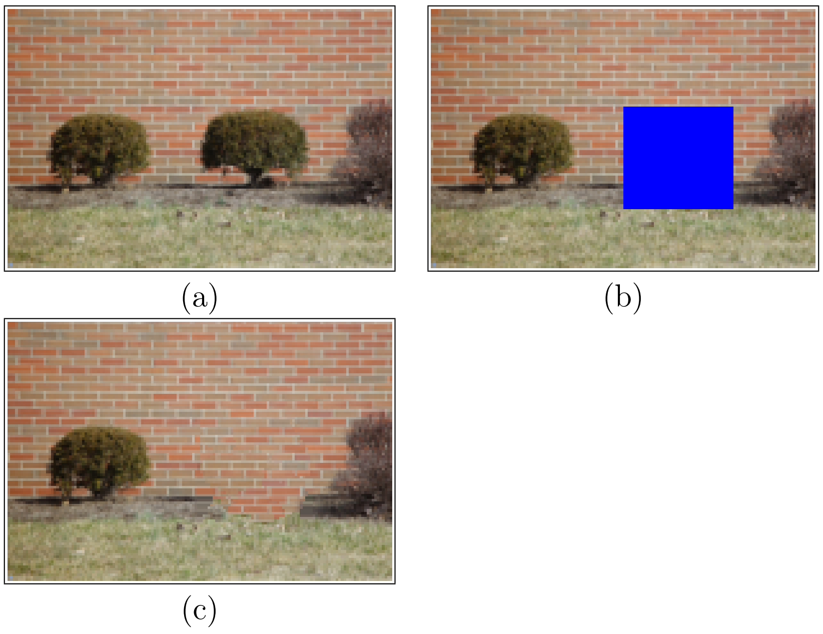


Figure 4.17: Zoom in of the original image, hole image and the final synthesised image of the BUSH image: (a) original image, (b) original image with the hole and (c) final synthesised result.

images.

In Figure 4.18, the hole of the CLEDWYN image is located at the edge of the image (top right) and surrounded by different textures, and therefore the filling in of the hole is not easy as the boundaries between these regions are not well defined. However, the method is reasonably able to create plausible, although repetitive, structures (region boundaries especially for the tree line). Yet the method fails to correctly create texture as seen in the image where unrealistic replication of complete grass patches into the area in the middle of the hole exists. This problem is due to a number of reasons. Because of the size of the hole, the initial filling in

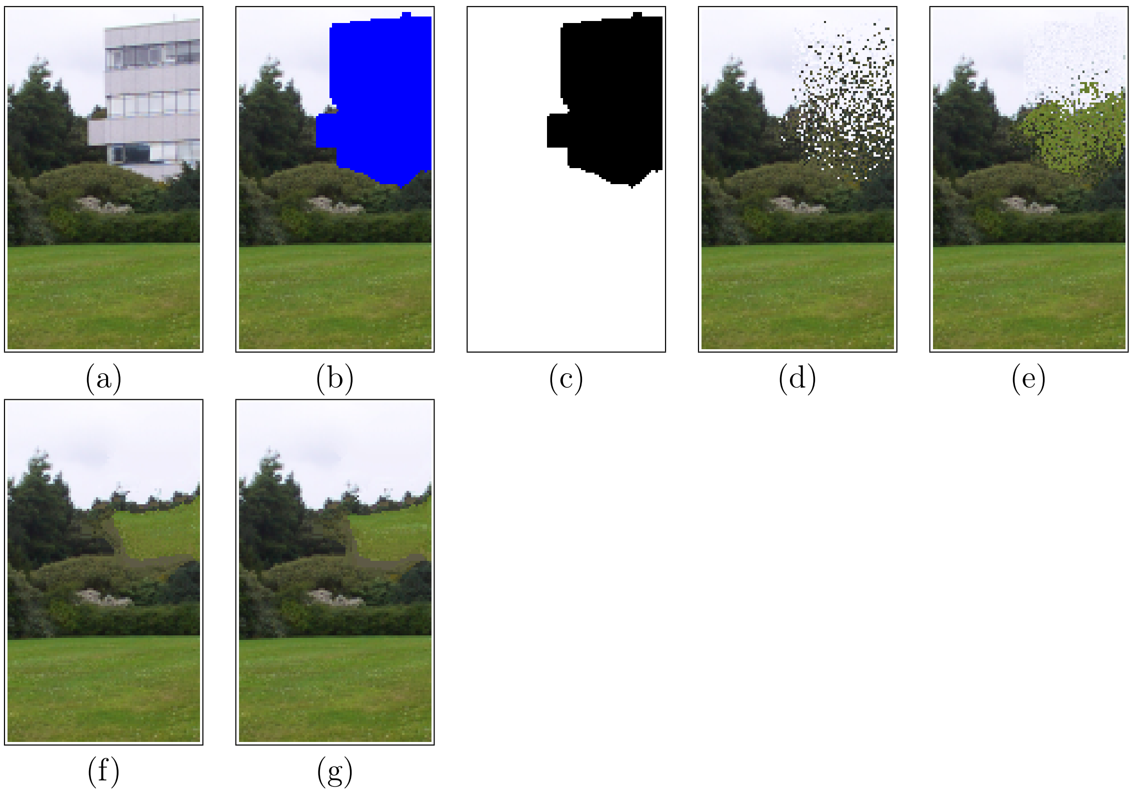


Figure 4.18: Texture synthesis result for the CLEDWYN image: (a) original image, (b) original image with the hole masked, (c) hole mask, (d) initial filling in of the hole area, (e) first iteration of texture synthesis, (f) middle iteration of texture synthesis and (g) final synthesised result.

of the hole is random especially at the middle of the hole (the size of the Gaussian had to be increased many times to obtained valid pixels). Therefore, the initial neighbourhoods of the pixels in the hole contain a random arrangement of dark green (coming from the bushes) and light grey (coming from the sky) pixels. In the RGB colour space, these two colours correspond to almost opposite corners of the RGB cube and the only other colour available is the green from the grass. As it happens, this green colour is roughly half-way between the dark green and light grey. Therefore a neighbourhood from the hole made of randomly positioned dark green and light grey pixels will match better with homogeneously green pixels than any other neighbourhood from the image. This resulted in the creation of grass in the hole.

This problem, as well as the one of the BUSH image, will be addressed in the following chapters by explicitly creating structure in the hole that will then be used to constrain the texture synthesis.

Chapter 5

Reconstruction of globally consistent image structure

5.1 Introduction

Pure texture synthesis is not often suited for completing holes which cover large areas of images that are surrounded by complex image structures. This is because the method depends on local neighbourhood search that does not consider the global structure of the image. Consequently, image completion using only texture synthesis with such images is expected to produce unrealistic results as shown in Figure 4.16 (g). Therefore, explicit reconstruction of the structure in the hole is necessary to deal with such images. This structure can then be used to create texture that will be coherent globally, hence producing plausible results.

The reconstruction of the hole structure is based on the assumption that regions (particularly in natural images) tend to be spatially continuous and are only separated by the hole and must therefore be linked. Regions are spatially continuous when the local statistics of the regions do not change with position. Thus, the images are first segmented and their regions are then relabelled based on their statistics. For example, if two regions with similar properties touch the hole, they should correspond to the same real region in the scene and should therefore be linked.

Our approach of explicit reconstruction of image structure is similar to recent work presented in [64, 101] in terms of using segmentation statistics to group similar regions and then connect them. However, the difference is in the way of measuring the similarity and connectivity of such regions. For example, in [64], the similarity of regions is based on computing their histograms and gradients, while in [101], the edges that separate different regions are selected and grouped based on their intensity and gradient angle histograms. The boundary curves (edges), are generated inside the hole using Tensor Voting in [64] while in [101], similar edges (couples) are joined inside the hole according to a circle fitted to the edge couple, and straight lines for edges (single edges) which do not have any matches.

The method of connecting curves inside the hole in [64] has the advantage of using Tensor Voting to smoothly link these curves. However, the method can have difficulties in deciding what constitutes a boundary between two regions [30]. In [101], the connectivity of region edges is based on the assumption of local curvature, being either circular or straight. Although this assumption works for various images, it may fail with cases where other shapes are needed for connection.

Our method of reconstructing hole structure measures similarity between regions attached to the hole based on histogram statistics and proximity of these regions (Section 5.3). The connection of regions is performed based on joining similar regions together using straight lines, and then flood-filling the created enclosed regions (Section 5.4). The structure of the regions in the hole is then modified to match the surrounding structures (Section 5.6).

More specifically, the method proposed for reconstructing the structure of the hole has three main components: image segmentation and relabelling, structure reconstruction (connections and filling), and segmentation synthesis. The segmentation and relabelling part deals with segmenting the original image and relabelling the segmented regions based on their statistics (see Sections 5.2 and 5.3 respectively). The structure reconstruction focuses on connecting “similar” relabelled regions using lines which results in creating enclosed region(s). After connecting these regions, they are flood-filled to form distinct regions (Section 5.4). The remaining regions in the hole that have no similarity with other regions are initialised using the same filling in process discussed in Section 4.4), but applied to segmented images (see Section 5.5). Once the regions in the hole are reconstructed, they need to be synthesised to match the surroundings of the hole (Section 5.6). Section 5.7 presents and discusses results of the structure reconstruction. Figure 5.1 shows a general outline of the structure reconstruction method.

5.2 Image segmentation

The segmentation of an image is an important part in identifying different regions in the image. In this work, we have used the segmentation method reported in [33] and called JSEG, which we found to be working well for a wide variety of colour-textured images. Also, it is reported in [64,101] that they used the method in their

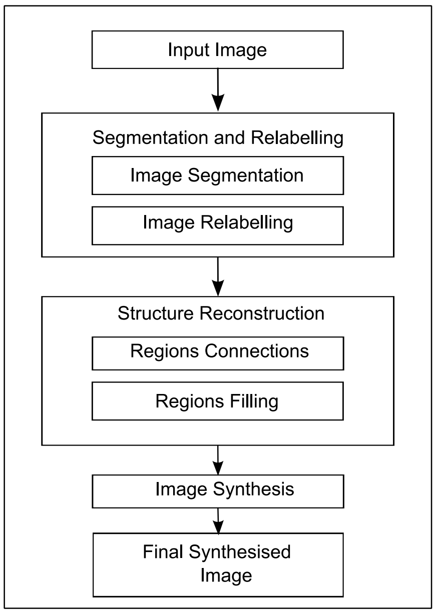


Figure 5.1: General outline of the structure reconstruction method

experiments, and even claimed JSEG to be one of the best available segmentation methods. Although the method in our experiments worked well for a wide range of images, it has failed to segment properly some reported cases. It is important to point out that image segmentation is not the subject of this work and therefore we only used this existing method.

The JSEG method works in two independent stages: colour quantisation and spatial segmentation. The colour quantisation stage quantises the image colour to different classes which are used to discriminate the image regions. The image pixels are then replaced by their corresponding class labels. As a result, the class labels form what is referred to as "class-map", a 2D vector of pixel positions in the image. The class-map is then used in the spatial segmentation stage to analyse the spatial distribution of the pixels rather than their corresponding similarity. Applying local measurements to the class-map introduces another concept called "J-images". This measures local homogeneities at different scales to find possible boundary locations. Based on the J-images, a region growing and merging method is then used to segment the image.

The JSEG method has three parameters which influence the segmentation results. The first parameter is a threshold for the process of colour quantisation, which computes the minimum distance between two quantised colours. This parameter can highly influence the segmentation results. For example, a small threshold can introduce a large number of quantised colours which will result in many regions being produced. On the other hand, the use of a small distance is justified to separate two neighbouring regions which have similar colours, yet are different.

According to the authors, a “good” parameter value is the one that can separate two regions with the minimum number of colours [33]. The second and third parameters are the number of scales and the threshold for region merging. In our experiment, the threshold value of the colour quantisation is set to 200, the number of scales is set to 1 and the merging threshold is set to 0.4, unless otherwise specified. Also, the segmentation method can be used to help a human operator select parts of the image he/she wants to remove, therefore creating the hole from the segmentation.

5.3 Relabelling of image regions

5.3.1 HBRs histogram matching

A hole bounded region (HBR) is a region created by the segmentation that is limited by the hole. It therefore has as part of its boundary a section of the boundary of the hole. The two extremities (HBR-BE) of this section are labelled in a consistent order (clockwise when turning around the hole) and are noted, unless otherwise specified, p_1^i and p_2^i respectively being the first and second extremity of the boundary between the hole and the hole bounded region i.

Based on the assumption that regions tend to be spatially continuous and are only broken up by the presence of the hole, the regions created by the segmentation may have to be relabelled to ensure that similar regions separated by the hole are indeed considered to be the same region. Similarity is in here measured using statistical properties of the colour of the regions. These similar regions will then be used to create structure in the hole by connecting them (Section 5.4). However, without high-level knowledge about the image, such relabelling can create false or too numerous possible matches that would be impossible to reconcile. Therefore, because it is more likely for nearby regions to match, connections between nearby regions will be preserved over connections between distant regions.

In our method, similar HBRs are identified based on their colour similarity and location. Colour similarity is computed based on colour histograms. This process works by computing the normalised colour histogram for each region that touches the hole and then calculating the Euclidean distance between the two histograms. Using the RGB colour space, the colour Euclidean distance between regions i and

j can be expressed as:

$$d(i,j) = \sqrt{\sum_{c=0}^{n} (H_i^R(c) - H_j^R(c))^2 + (H_i^G(c) - H_j^G(c))^2 + (H_i^B(c) - H_j^B(c))^2}, \tag{5.1}$$

where H^R, H^G and H^B are the histograms for the three colour components, $H(c)$ is the cth bin of histogram H, H_i and H_j are the colour histograms of regions i and j and n is total number of histogram bins (255 in our experiments). We assumed that R, G and B are independent, and computed the one-dimensional histograms of the colour component and this works with a variety of images because mainly the regions are taken from the same image. Thus, less variation between these regions compared to the image retrieval problem. Also, using the RGB colour space is not ideal but acceptable here because we only deal with one image and shadows are considered to be different regions (we don't want to propagate shadows where we don't want them).

In practice, we used the Euclidean distance to measure similarity between images (regions, patches and histograms). However, it is not a good measure for perceptual similarity between images as it does not take into account the relationship between the light source, the object and the viewer. However, because our method is designed to complete the hole from only the same image (same camera), it is less likely these factors will cause problems.

Because regions around the hole could be different, a general threshold on the distance is used to exclude regions that cannot be possible matches because of the substantial differences in their colour histograms. Therefore, the pair of regions having a distance below this general threshold ($G = 0.2$) is selected and included in a list of possible matches. A second threshold is then introduced and applied to this list as a way of narrowing down the selection of these good matches to choose better matches. This only includes regions that are below a set threshold as a proportion of the lowest distance (d_m), and this threshold is set to be $S \times d_m$, where $S = 1.2$. In practice, we found that $G = 0.2$ produced acceptable results for variety of textured images.

5.3.2 HBRs spatial proximity

Once the similarity between HBRs has been established based on their colour content, their spatial proximity must be evaluated to determine possible matches. Indeed, the assumption that areas of the original image are only broken up into different regions because of the presence of the hole implies that regions should be

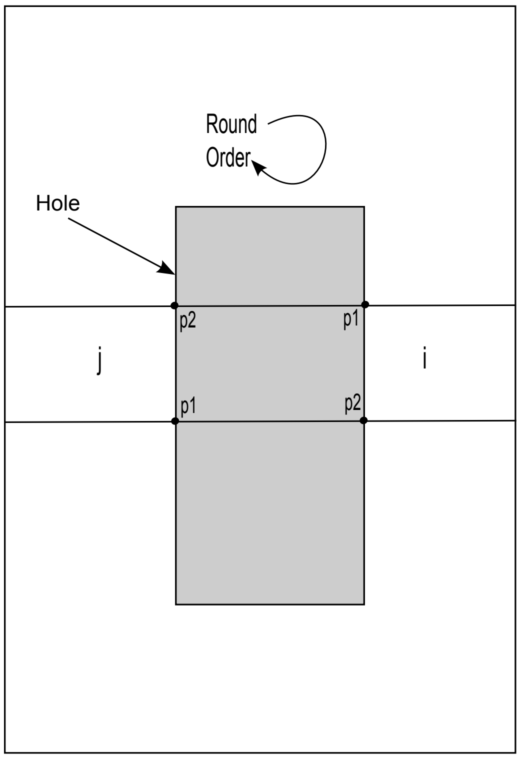


Figure 5.2: A typical configuration between two HBRs: Regions i and j are connected using two lines. The points are determined through the sequential order scan discussed in Section 5.4.

linked not only based on their colour content but also on their spatial proximity. The HBR pairs with the minimum spatial distances are considered and relabelled with similar labels for each pair and then saved in a region connections priority list (RCPL) for connection. The spatial distance between two HBRs i and j is expressed as:

$$d_{ij} = d(p_1^i, p_2^j) + d(p_2^i, p_1^j), \tag{5.2}$$

where $d(p, q)$ is the Euclidean distance between points p and q. The lines (p_1^i, p_2^j) and (p_2^i, p_1^j) will be considered later for linking the two corresponding regions across the hole (5.4). Figure 5.2 shows a typical configuration between two HBRs.

As the aim of finding similar HBRs is to link them across the hole, we must ensure that pairs of HBRs considered for linking would not break the existing structure (outside the hole). The lines considered to link the regions (Equation 5.2) are therefore checked for crossing of other regions before computing their spatial distances. If any crossing occurs, then the pair of HBRs is discarded (see Section 5.4.2). Figure 5.3 shows a general example of the region relabelling process for region R1 based on computing colour and spatial distances.

We now describe the creation of the connections and constraints used, followed by

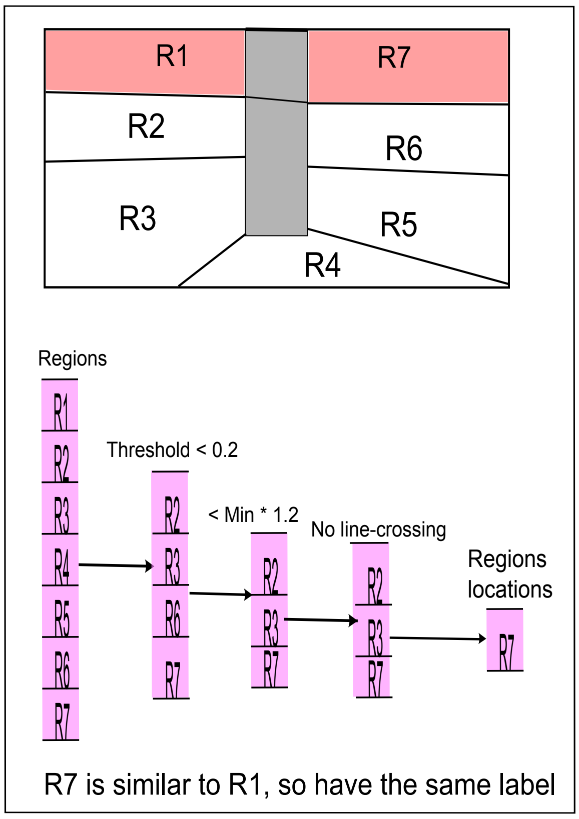


Figure 5.3: An example of showing the regions relabelling process. In this example, if we assume that region R1 is the current region processed for relabelling, then we compare the possible matches to R1 by comparing histogram differences, checking line-crossing and applying spatial proximity. In this example, we select region R7 to be the best match to Region R1

the flood-fill procedure.

5.4 Connecting HBRs

The structure reconstruction in the hole is essentially based on the idea that pairs of HBRs with similar properties are connected using lines and the created enclosed area are then flood-filled by the same label. Similarly, the HBRs that are not pairs are self-connected using straight lines and then flood-filled. The principles of establishing such connections and flood-filling are discussed in this section.

5.4.1 HBR connection principles

Connecting a pair of regions inside the hole is governed by principles that can create a connection that does not violate other valid region information, and can produce acceptable structure. The created structure is acceptable when the con-

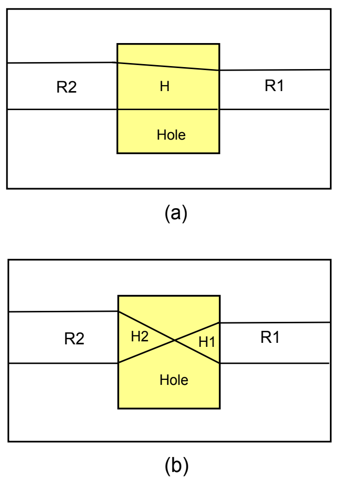


Figure 5.4: Example of compatible and non-compatible hole structure. In (a) the structure of the hole area looks compatible with the structure of the regions pair. However, in (b) the lines which connect regions cross each other and therefore produce two enclosed regions that are not compatible with the structure of the region pair.

nection produces an area in the hole in which its structure is compatible with the structure of the surrounding region pair, as can be seen in Figure 5.4(a). On the other hand, the created structure is not acceptable when it is not compatible with the structure of the regions pair. For example, when two connecting lines cross each other, it produces enclosed regions that are not compatible with the surrounding structure. Figure 5.4(b) shows an example of this crossing and the structure created. Such situation is easy to prevent by using a sequential scan order so that each HBR-BE is connected to the closest HBR-BE of the other region (Section 5.4.2.2).

One could argue that limiting the creation of connections between pairs of regions is too constraining and that tripartite, or even larger, groups should be considered. However, this can potentially lead to intractable situations that may be difficult, if not impossible, to solve. Instead, we iteratively create connections between pairs of HBRs, which, overall, provides us with a tractable way of creating larger groups of HBRs, see Section 5.4.5

Another case is to be considered: the case of HBRs that do not match any other HBR but can nevertheless lead to connections. We will call these self-connections. Figure 5.5 shows examples of pair- and self-connections.

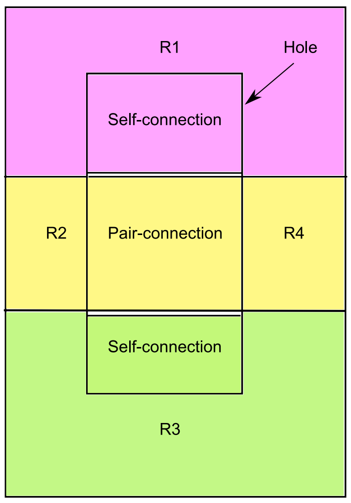


Figure 5.5: Examples of pair- and self-connections. In this example, we assume that regions R2 and R4 are similar, so they should be pair-connected. On the other hand, regions R1 and R3 do not match with other regions therefore, they should be self-connected.

5.4.2 Line-crossing detection and straight line creation

5.4.2.1 Line-crossing detection

As mentioned before, some connections cannot be kept as they would modify the structure of the image outside the whole. These connections, either pair- or self-connections, are detected by checking for intersections between the corresponding lines and any other region outside the hole other than the connected ones. If an intersection happens, the connection is discarded. Figure 5.6 shows an example of a pair-connection that violates that constraint and cannot therefore be kept.

It is worth pointing out here that a line-crossing that is completely inside the hole is allowed in order to preserve parts of the structure of the pair regions that are being painted over by higher priority regions. This depends on the priority of the regions being connected which is discussed later in Section 5.4.3. Figure 5.7 shows an example of allowing line-crossing inside the hole. In this figure, the lines which connect regions R1 and R6 cross with the lines that connect regions R3 and R8, and this crossing is allowed as long as the region connections are prioritised according to the RCPL.

5.4.2.2 HBRs linking points

Before explaining the straight line creation for linking HBR pairs, the points between the hole and the HBRs need to be identified first in order to make the

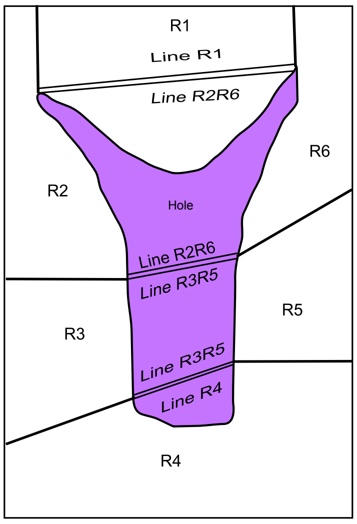


Figure 5.6: Example of line-crossing check. In this example, let us assume that the regions R2 and R6 on the one hand and R3 and R5 on the other hand are similar, according to the image relabelling process discussed in Section 5.3. The region pair (R3, R5) has no line-crossing and therefore can be connected. Also, regions R1 and R4 have no line crossing and hence can have self-connections. On the other hand, regions R2 and R6 cannot be connected because one of the lines R2R6 crosses region R1 and therefore the connection between region R2 and R6 cannot be made. Note that the double lines R1, R2R6, R4, R3R5 are shown for the pair and self connections, while lines R2R6 and R3R5 are shown for the pair connection only where each of these lines is connected to its corresponding region points.

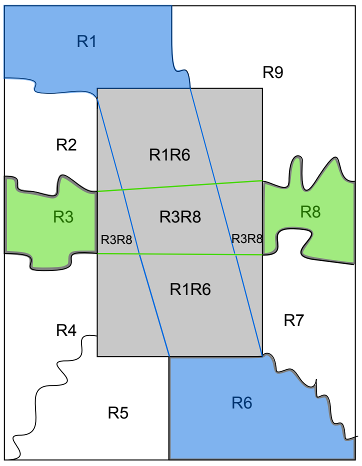


Figure 5.7: Example of allowing line-cross inside the hole and prioritising pair regions connections. In this example, we assume, according to the RCPL, that the region pair (R1, R6) is connected before the region pair (R3, R8). This means that this pair paints on the connected (R1, R6) region. However, parts of the region pair (R1, R6) are preserved.

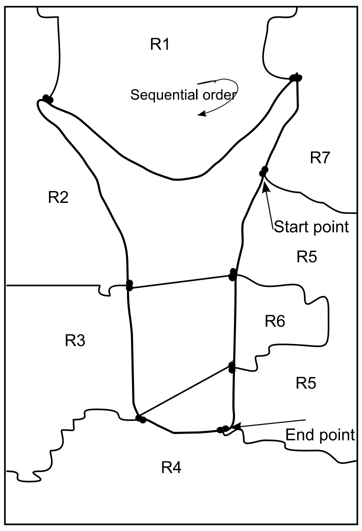


Figure 5.8: Example of identifying hole regions linking points. In this example all regions except R5 have two neighbouring regions and therefore have two sequential start and end points. Region R5, however, starts from region R7 (if we assume the sequential order starts at region R7 down to region R5, R6, etc.) and ends at region R4.

connection. Identifying the HBR-BEs is important since they determine the beginning and ending of the connection. Identifying extremities can vary depending on the nature of the region and its spread around the hole, as well as the shape of the hole. However, regions, particulary natural ones, often tend to have two points which connect them to different neighbouring regions as shown in Figure 5.8 where each region of $R1$, $R2$, $R3$, $R4$, $R6$ and $R7$ has two neighbouring regions and hence two linking points. However, there exist cases where regions can spread around the hole and have more than two neighbouring regions. In such cases, the points of such regions are determined based on the beginning and ending points of the region according to the sequential order. Figure 5.8 shows an example where region $R5$ is spread around region $R6$ and therefore has four connecting points with its neighbouring regions ($R4$, $R6$ and $R7$). As a result, region $R5$ linking points therefore should be the start and the end of the region as shown in Figure 5.8.

The determination of the HBR-BEs is done for each HBR according to a sequential scan order. This order starts from an initial hole boundary point (first located pixel in a raster scan order) and then scans the hole points in a clockwise or anti-clockwise order depending on the maximum number of hole pixels for each of the eight pixels centred around the current pixel. The scanning of the hole pixels scans on its way the hole regions pixels that are connected to the hole and then select only the region's boundary points that link the region to other HBRs.

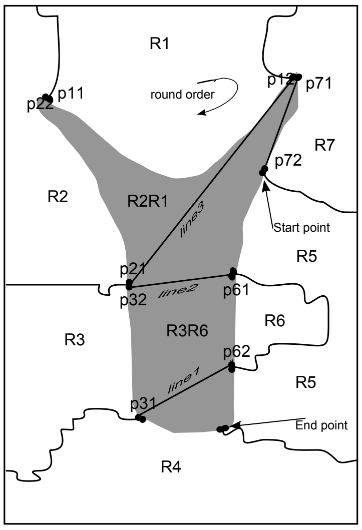


Figure 5.9: Example of connecting region pairs and self-regions using lines. In this example, the regions of the pair (R6, R3) are connected using lines where points are connected according to the closest point to them. For example, if we take the region pair (R6, R3), points p62 and p31 on the one hand, and p32 and p61 on the other hand are connected. The points of the region pair (R2, R1) are connected as point p21 with p12, but points p22 and p11 do not need a line to connect because they are adjacent points. For the self-region connection, region R7 is self-connected using lines between points p72 and p71, but did not create much structure in the hole due to fact that it does not surround much of that hole area.

5.4.2.3 Straight line creation

Once HBR-BEs are identified, connecting HBRs are done using a straight line connection. The line starts connecting the region points according to the sequential order. This means that a HBR-BE is connected to its closest HBR-BE of the other HBR, and this can often ensure that an enclosed region is created in the hole for flood-filling. However, if it happens that the regions of the pair are neighbours, or the self-regions are not surrounding the hole, the connection may not create much structure in the hole, as can be seen in Figure 5.9 (e.g. region R7). Figure 5.9 also shows an example of creating connecting region pairs and self-regions using lines.

Using straight lines may seem unhelpful especially when the surrounding structure of the hole does not necessarily have this particular structural pattern. However, these line structures are modified to match the surrounding regions structure outside the hole. This is done by synthesising the hole in order to match the structure of the surroundings, which is discussed in Section 5.6.

5.4.3 HBRs connection priorities

We previously established the RCPL of good possible matches for the HBRs based on colour statistics and spatial proximity. We then try to link these matches in an iterative process by creating connections in the hole, making sure that some constraints are respected (no crossing, regions belong to the same hole). Each closed region created is then flood-filled.

The main reason for prioritising HBR pair connections is the tendency for nearby regions to have clearer structures when connected than distant regions, and to prevent distant regions from crossing such connections and hence changing the structure of the existing reconstructed regions. This priority of connections enables nearby regions to have complete connections (the two lines connect the regions without any discontinuity) over distant regions. On the other hand, the pairs of regions with low priority (large distances) can have complete connections if there are no crossing over by the pairs of nearby regions, or partial connections (the two lines are disconnected by the existence of other regions, but create some structure in between) when they are being painted over, hence recovering at least parts of their structure.

In practice, once the HBR pairs have been identified and relabelled, they are stored in the RCPL list. The list is sorted in a descending order according to their corresponding spatial distances. This means that regions that are distant from each other are connected first while the nearby ones are connected next, allowing painting on the existing reconstructed regions that are distant from each other.

Figure 5.7 shows an example of prioritising these connections. In this example, we assume that each of the region pairs (R1, R6) and (R3 and R8) has similar statistics according to the relabelling process discussed in Section 5.3. Regions R1 and R6 are connected first because their corresponding spatial distance is larger than the distance between regions R3 and R8. Regions R3 and R8 are then connected and their lines paint on the R1R6 lines. This enables us not to loose the structure of regions R1 and R6 (caused by the connection of regions R3 and R8), and to include parts of the structural information of regions R1 and R6. Regions R3R8 and R1R6 will be flood-filled according to their corresponding label, which will be discussed in Section 5.4.4.

5.4.4 Hole regions flood-filling

Pair or self-connections and their corresponding connecting lines create an area in the hole that is completely surrounded by the same colour, that corresponding

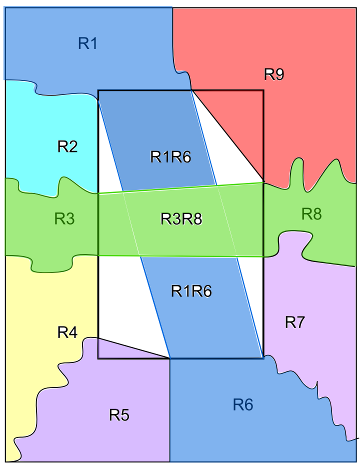


Figure 5.10: Example of hole regions flood-filling. In this example, the region pairs (R1, R6) and (R8, R3) are connected and flood-filled forming enclosed region(s) for each pair. Self regions R5 and R9 are also flood-filled while region R2, R4 and R7 have not been flood-filled due to the fact that there are no areas in the hole for flood-filling for these regions.

to the label of the region(s) that initiated the connection. This area is flood-filled with the same label to create a single new region covering a part of the hole and therefore structure. The flood-fill order follows the RCPL order where HBR pairs with higher spatial distance are flood-filled first.

The region pairs (R1, R6) and (R8, R3) in Figure 5.10 show the flood-filling of the region pairs. Once such region pairs are flood-filled, the self regions are flood-filled next in the same way as the region pairs. The reason for connecting and flood-filling pair regions before self regions is that pair regions connections can create more structure (structure of two regions) than self connections which only reconstruct single region's structure. In Figure 5.10, the region pairs (R1, R6) and (R8, R3) are flood-filled. Regions R5 and R9 are self-connected and flood-filled. Regions R2, R4 and R7 are self-connected, but there are no area in the hole for flood-filling.

5.4.5 Iterative connections and flood-filling of hole regions

As discussed in Section 5.3, limiting the connections of regions to pair connections may seem too constraining, however, using tripartite, or even larger groups may lead to a situation that could be very difficult to solve. Therefore, using iterative pair- and self- connections can be a possible solutions to this.

In practice, when computing the histogram matching for the HBRs, only best similar pairs are selected. However, there can still be good similar regions that are not being considered and could be good matches. Thus, the regions should be connected and flood-filled in an iterative connection and flood-filling process.

The process of this stage is similar to the main connections and flood-filling stage (first iteration), but differs in aspects explained here in more detail. First, for each new connected and flood-filled hole region, similar regions are selected from the already saved good match regions list when computing the colour histogram in the regions relabelling process discussed previously in Section 5.3, so there is no need for relabelling these regions again.

The second step is to exclude regions that are not touching the same hole as the target region. The reason for this is that the hole after the main connections and flood-filling may not be a a single connected hole. Thus, potential HBRs should belong to the same hole as the current hole region, as this emphasise the concept of spatial proximity of regions. Once this is tested, the new regions list is checked for line-crossing as done in the main regions connections. This produces a new list and from the list the spatial distances are computed and the region with minimum distance is selected to be the best match for the current hole region. It is exactly the same computation of spatial distances discussed previously in Section 5.3.

In addition, the principles of hole regions connections (Section 5.4) are applied to the new reconstructed hole, taking into consideration that HBR pairs belong to the same hole, and their HBR-BEs are the ones that are connected to the hole in a sequential order. Figure 5.11(a) shows an example of the iterative hole regions connections and flood-filling.

5.5 Remaining unfilled areas inside the hole

When connecting and flood-filling pairs and self regions, there can still be remaining areas in the hole that need to be filled in order to complete the entire structure of the hole. Because these remaining areas can be of any shape and surrounded by different regions, the topology of such areas can be complicated, and there can be different connection possibilities that might or might not be acceptable for flood-filling. Figure 5.12 shows an example of the difficulty of connecting such unfilled areas of the hole.

As a result, randomly initialising prior to synthesising the structure of the unfilled areas can be a solution to this problem as it propagates parts of these areas inside a hole. Thus, the same initial filling process described in Chapter 4 is used, but applied to segmented images. The process initialises the areas of the hole

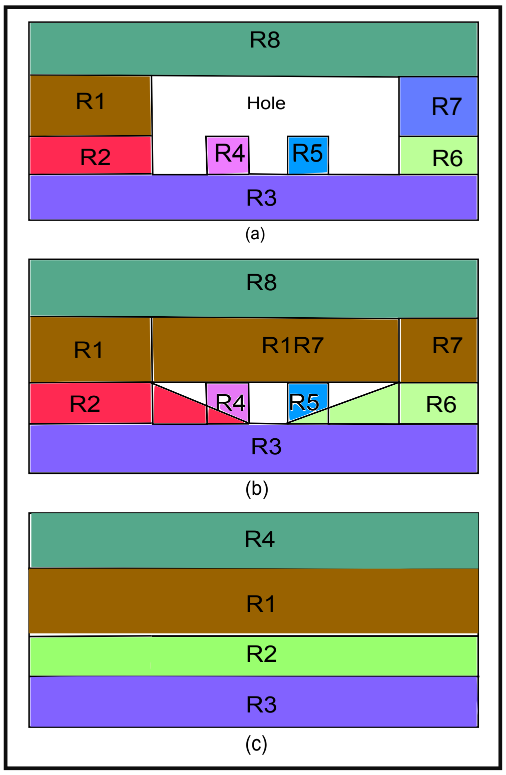


Figure 5.11: Example of Iterative hole regions connection and flood-filling. In this example, (a) shows a hole that is surrounded by eight regions, and we assume that regions R1 and R7 on the one hand and R2, R4, R5 and R6 on the other hand are similar according to the image relabelling process discussed in Section 5.3, (b) is the main connection (first iteration) where the pair (R2, R4) in (a) was assumed similar and thus relabelled with the same label, connected and flood-filled. The same procedure applies to the pair (R5, R6). Note that parts of regions R4 and R5 are missed during the connection process. This is because there are no HBR-BEs to connect the upper part of the regions since they are not attached to another region, (c),is the second iteration of connection and flood-filling where the regions of the pairs (R2, R4) and (R5, R6) in (b) were assumed similar according to the relabelling process, and therefore iteratively connected and flood-filled.

to provide initial values that have some degree of randomness, but are plausible. Regions inside the hole are then synthesised in order to coerce the structure of the regions to match the structure of the ones surrounding the hole. This process is discussed in Section 5.6.

It is also important to point out that as the main goal here is not to fully automate the connection process because this is not possible without high level knowledge about the scene, there can be problems related to regions mismatch in the relabelling process and therefore creating wrong connections. As a result, some user interaction may be needed to modify these connections. A simple user interaction could happen via a file that saves the structure of the hole. The file contains the

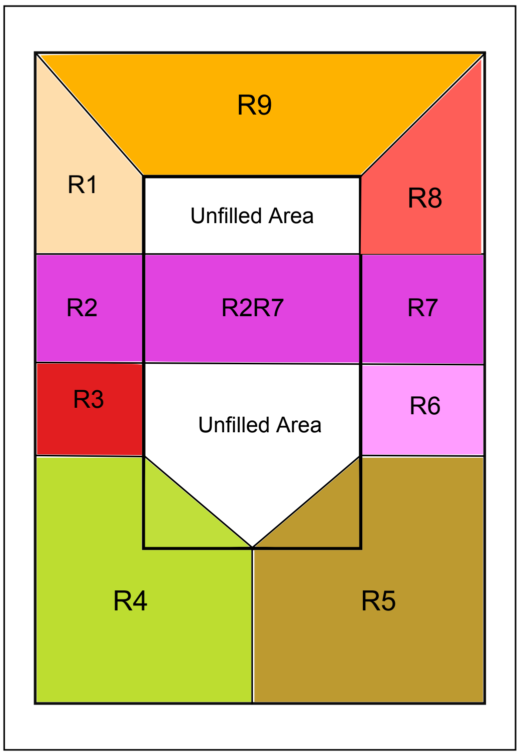


Figure 5.12: Example of complex unfilled areas of the hole.

regions being selected for connection, which the user can modify if necessary. The other possibility of user interaction would be to design a graphical user interface with which the user could modify current region connections by adding or deleting straight lines.

5.6 Segmentation synthesis

In previous sections we proposed a method to fill in the hole with a structure that matches as best as possible the topology of the surroundings of the hole. However, the structure was created using straight boundaries between adjacent regions. This is unlikely to match existing boundaries outside the hole. We therefore need to modify the shape of these boundaries to create a more plausible structure with boundaries that match the shape of, and seamlessly propagate, the boundaries outside the hole into the hole. This is performed using the same texture synthesis method as described in Chapter 4 but applied to label images, instead of colour images, using a different method to match neighbourhoods.

The synthesis starts by updating the hole pixels by searching for similar neighbourhoods in the rest of the image. Unlike the texture synthesis method described in Chapter 4, the neighbourhood matching comparison is not done using the Euclidean distance but, instead, using the complement of the Kronecker delta function defined as:

$$\bar{\delta}(i, j) = \begin{cases} 0, & i = j \\ 1, & i \neq j \end{cases} \tag{5.3}$$

The reason for choosing such a binary distance function is that what is compared here are labels, not colours. There is no point in considering how different labels are, only that they are different.

The synthesis process is done in parallel to update the pixels in the hole, without considering the values of the already synthesised pixels. As a result, the value of a pixel being synthesised will not be affected by previously synthesised ones. More on the parallel synthesis is discussed in Section 4.3.

Iteratively, we update the hole using the same iterative synthesis scheme used for texture synthesis, but applied here to segmented images. This refinement scheme allows global random initial structure that progressively converges towards good quality structure. More on iterative refinement scheme is explained in Section 4.5. When the image reaches its final convergence, it is then used as a constraint to propagate texture, which is discussed in Chapter 6.

When searching for the possible matches to the current target pixel's neighbourhood, it is important to take into consideration the size of the neighbourhood. This size should be large enough to capture the structure of the image (shape of the boundaries between regions), otherwise the structure of such images will be lost. In our work, the neighbourhood size was set to 11×11 pixels, unless otherwise specified. A pseudocode for the reconstruction of image structure in is described in Algorithm 3.

5.7 Results and discussion

Our method for reconstructing the structure of the hole has been applied to different images and produced good results as can be seen in Figures 5.13–5.31. Figure 5.13 shows the results of structure reconstruction for the CLEDWYN image. It contains a building that needs to be removed and replaced with proper structure and texture. Reconstructing the structure of the hole is not easy as the HBRs are not completely surrounding the hole because it is located on the edge of the image. Although the pair and self region connections have been able to reconstruct part of the hole, a large area of the hole is left to be filled in. This means that the

Algorithm 3 Pseudocode for the reconstruction of image structure

```
Segment the image
For each HBRs
    Compute R, G and B histograms independently
    Normalise histograms
    Compute histogram difference for each HBR pair
    Select all pairs that their distances are less or equal to G
    Select all pairs that their distances are less or equal to $Minmumdistance \times S$
    Check line crossing
    IF there is no line crosses other region
        Compute spatial distances
        Select a region with the minimum spatial distance
        Save the HBR pair in RCPL
        Remove the two regions from the HBRs list
        Relabel the HBR pair with the same label
    ENDIF
ENDFOR
Order the RCPL in a descending order according to their corresponding spatial distances
For each region pair in the RCPL
    Connect them
    Flood-fill them
ENDFOR
For the remaining HBRs
    Check line crossing
    IF there is no line crosses other region
        Connect self region
    ENDIF
ENDFOR
IF there is unfilled area in the hole
    Initialise the areas using the algorithm described in Algorithm 2
ENDFOR
Synthesise the reconstructed image as described in Algorithm 1, but applied to the segmented image
Produce the final reconstructed image and use it for the image completion method
```

structure reconstruction will not only depend on the self and pair connections but also, and to a large extent, on the initial filling-in and the synthesis process.

In practice, the original image is segmented and its unwanted building area (hole) is masked out, as is shown in Figure 5.13 (b) and (c) respectively. The hole is then reconstructed by relabelling, connecting and flood-filling the regions corresponding to the trees and the bushes in the middle, as shown in Figure 5.13 (d). Also, the sky and the small bush regions (on the bottom right of the hole as in (b)) are self-connected, forming a sky region starting from the middle to the top of the building, as shown in Figure 5.13 (d). In the second iteration of the hole reconstruction, the dark green and blue regions are judged similar, hence are connected and flood-filled, as shown in Figure 5.13 (e). This leaves a remaining hole area (undecided area) in the middle, shown in Figure 5.13 (f), to be filled by the initial filling process as shown in Figure 5.13 (g). This initialised area has random pixels from the light green region (sky) on top and the light blue region (trees and bush) on the bottom of the area, which makes plausible initial values for the synthesis process to use. This image in Figure 5.13 (g) is synthesised so that the hole structure matches the surroundings. The images in Figure 5.13 (h), (i) and (j) show the first, middle and final iterations respectively of the iterative parallel synthesis. Clearly, the structure of the hole in the final result is reconstructed well, producing acceptable boundaries in the hole, and propagating plausible structure between the sky and the tree-bush regions. Figure 5.14 shows a zoom in of the the CLEDWYN image results.

Figure 5.15 shows a structure reconstruction for the BUSH image where the removed area was the bush part of the image. The hole in this image is surrounded by three textured regions, one of them is a regular texture. The regions are different in terms of colour statistics and therefore, there is no similar regions pairs in this image. The reconstruction of this area is done using only self region connections. Therefore, self region connections are performed for the pink (wall) and purple (grass) regions, and then flood-filled, as shown in Figure 5.15 (d). The remaining unfilled area is initialised from the surrounding regions (pink, purple and green areas), as shown in Figure 5.15 (f).

In Figure 5.15 (i), the final hole structure is shown. Although the structure reconstruction of the hole is acceptable, the boundaries between the wall and grass regions were not propagated along the horizontal direction. This can be caused by inaccurate values of the initial hole filling process which selects more wall pixels than grass pixels as shown in the middle area between them in Figure 5.15 (f). This is due to the fact that at this hole area, the wall region is more available (outside the hole) in the area of the Gaussian distribution than the grass area.

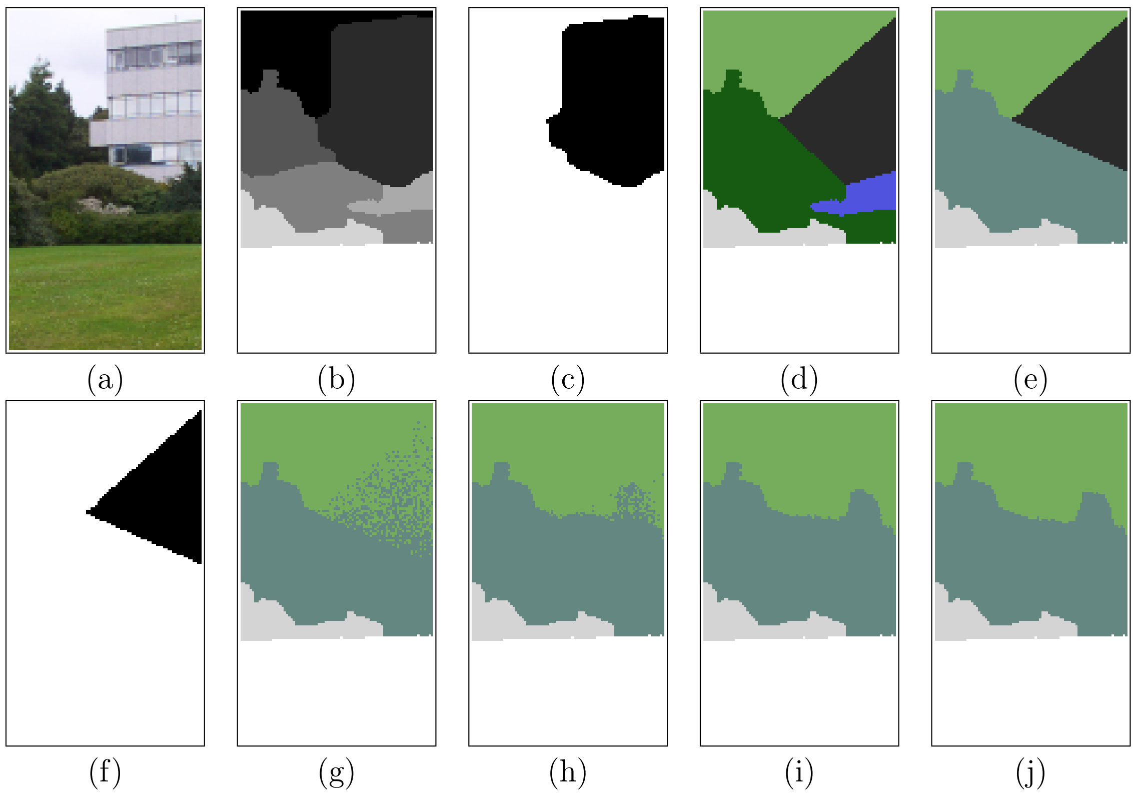


Figure 5.13: Hole structure reconstruction results for the CLEDWYN image: (a) original image, (b) segmented image, (c) hole, (d) and (e) first and second iteration of reconstructing the hole structure, (f) undecided area, (g) initial filling in of the undecided area, (h), (i) and (j) are first, middle and final iterations of synthesising the reconstructed segmented image.

As a result, the first synthesised image in Figure 5.15 (g) and the following iterations will be affected by these initial values when searching for good neighbourhood matches. The problem is also related to the fact that the segmentation method fails to segment the grass region into two regions (brown grass at the top and green grass at the bottom of the region). Figure 5.16 shows zoom in of the BUSH image results.

Therefore, if the brown grass had been identified in the segmentation, then the pair-connection of these regions would have been able to connect the regions properly and there would have been no need for inaccurate initial filling, as shown in Figure 5.17 (d). In Figure 5.17 (f), a clear reconstruction of the brown grass area is created where boundaries are well defined between this area and the wall on one side, and the brown grass and the green grass areas on the other side. This emphasises the fact that the segmentation is an important process of the method and that it needs to be carefully controlled so that the correct regions are created. Note here that the threshold of the colour quantisation for the segmentation of this image was set to 400 instead of 200 to exclude unnecessary regions. Also, the

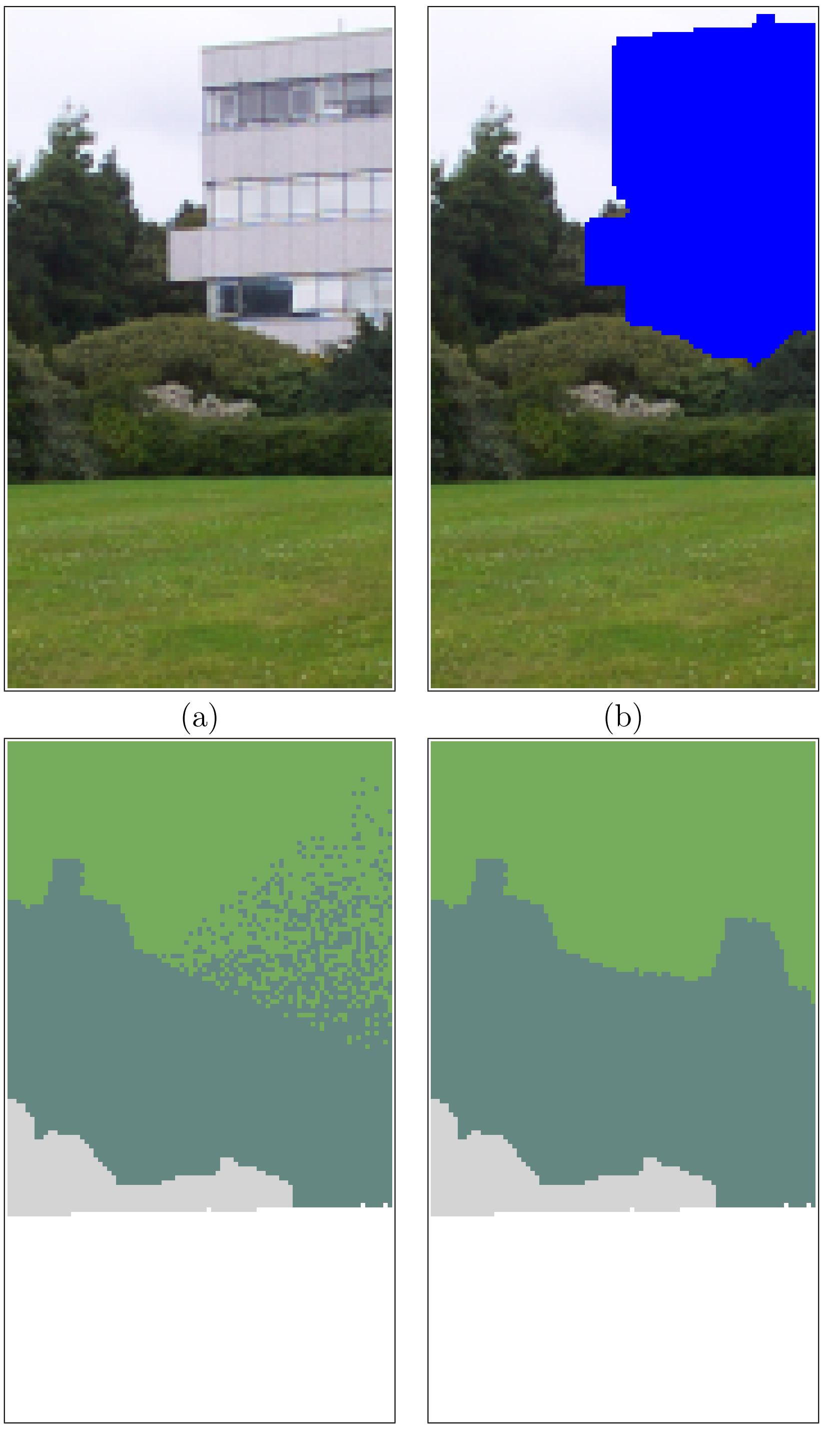

(a) (b) (c) (d)

Figure 5.14: Zoom in results for the CLEDWYN image: (a) original image, (b) original image with a hole, (c) initial filling in, and (d) final reconstructed image.

value of neighbourhood size was set to 15 to capture properly the structure of the wall. Figure 5.18 shows a zoom in of the the improved BUSH image results.

In Figure 5.19 (b), the hole of the DUSTBIN image is surrounded by different types of regions and their structures will be propagated inside the hole to create its structure. Figure 5.19 (d) shows how the structure of the outside regions are

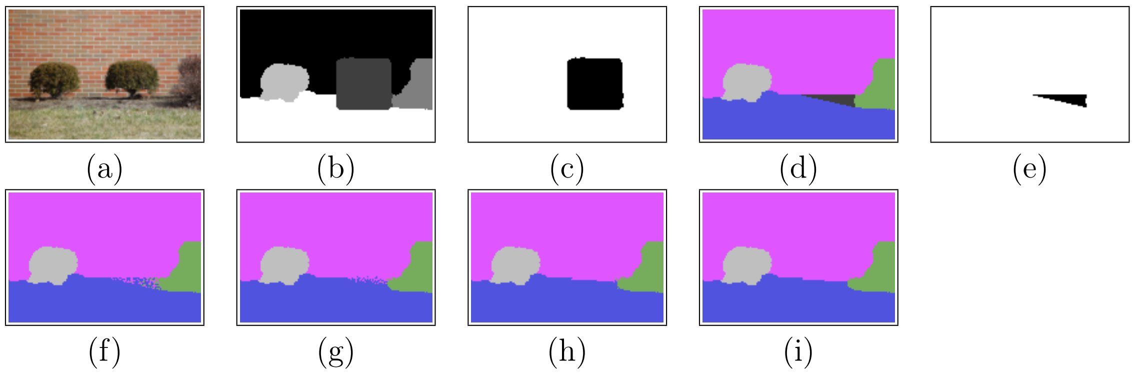


Figure 5.15: Hole structure reconstruction results for the BUSH image: (a) original image, (b) segmented image, (c) hole, (d) reconstructing the hole structure, (e) undecided area, (f) initial filling in of the undecided area, (g), (h) and (i) first, middle and final iterations of synthesising the reconstructed segmented image.

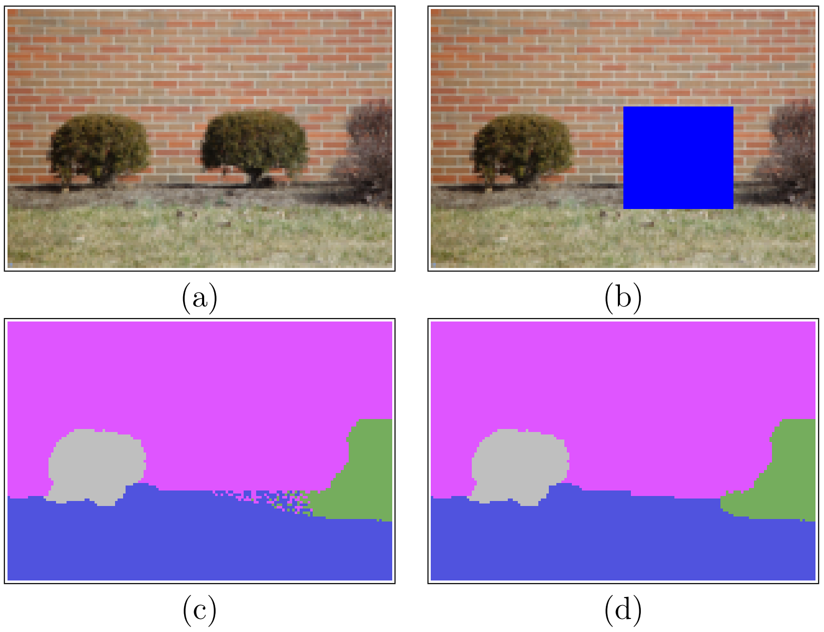


Figure 5.16: Zoom in results for the BUSH image: (a) original image, (b) original image with a hole, (c) initial filling in, and (d) final reconstructed image.

clearly reconstructed, particularly in the pink area where the curvy line patterns are propagated, as shown in Figure 5.19 (g). Also, the figure shows the result of connecting the region pairs in the middle of the hole (red and pink regions). The regions on the top (sky) and the bottom (grass) of the hole are self-connected. These pair and self regions are flood-filled forming a well structured region. In the synthesis process shown in Figure 5.19 (d), (e), (f) and (g), the structure of the pink region inside the hole has converged to match the surrounding structural patterns (curvy lines) of that regions. The structure of the blue (grass) region is also reconstructed well. Figure 5.20 shows a zoom in of the the DUSTBIN image results.

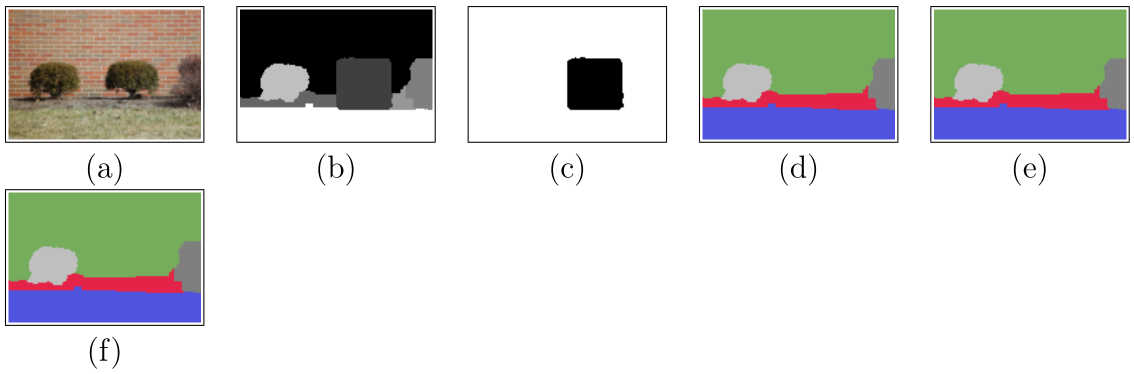


Figure 5.17: An example of improving the reconstruction of the hole by manually segmenting the regions that correspond to the areas of the brown grass of the BUSH image. (a) original image, (b) manual segmented image (dark and light gray regions), (c) hole, (d) reconstructing the hole structure, (e) and (f) first and final iterations of synthesising the reconstructed segmented image.

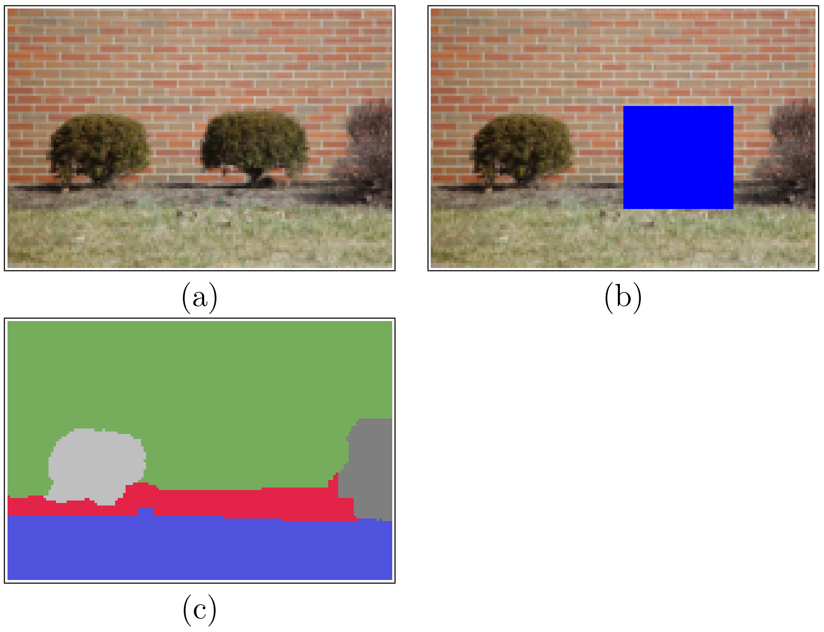


Figure 5.18: Zoom in results for the improved BUSH image: (a) original image, (b) original image with a hole, and (c) final reconstructed image.

On the other hand, the structure does not change a lot in the top area of the hole due to the fact that there are not enough patches outside the hole that include a green area in the top half of the patch and red in the bottom of it. Therefore, the line connection pattern did not converge to match the surrounding regions since such regions have insufficient examples of the same patch. However, when we change the hole size to allow more examples of the matching patches, the line structure converges to match the surrounding structures (curvy lines), as shown in Figure 5.21 (g). Figure 5.22 shows a zoom in of the improved DUSTBIN image results.

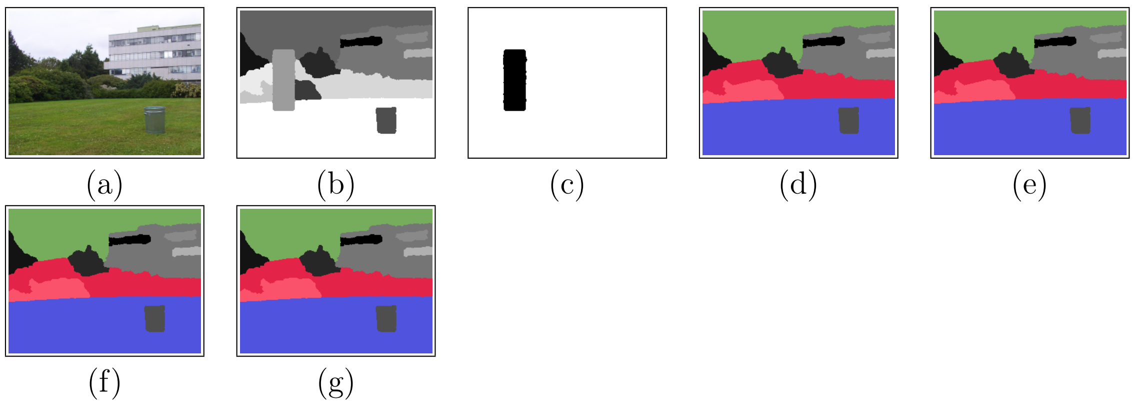


Figure 5.19: Hole structure reconstruction results for the DUSTBIN image: (a) original image, (b) segmented image, (c) hole, (d) reconstructing the hole structure, (e), (f) and (g) first, middle and final iterations of synthesising the reconstructed segmented image.

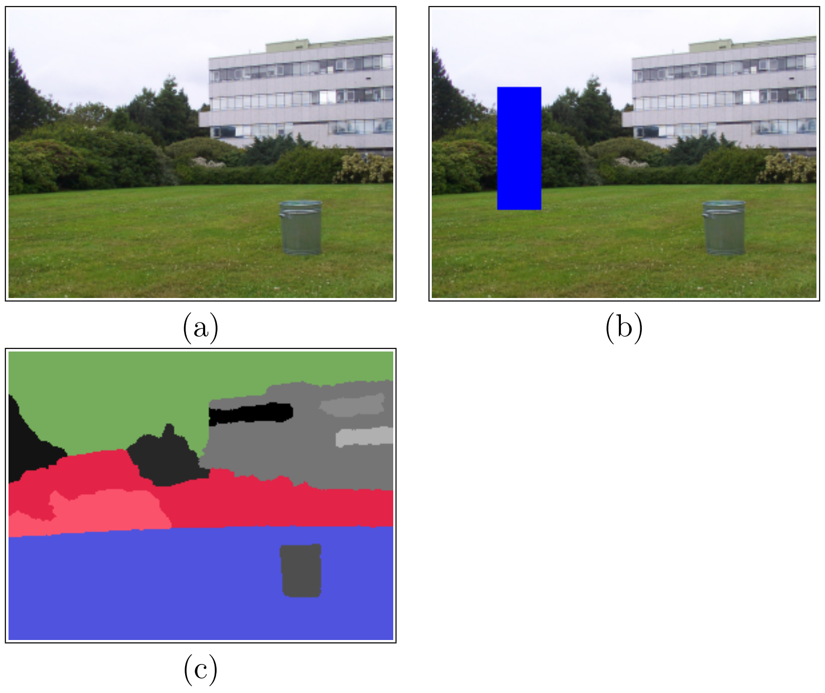


Figure 5.20: Zoom in results for the DUSTBIN image: (a) original image, (b) original image with a hole, and (c) final reconstructed image.

Figure 5.23 shows the results of structure reconstruction for the PAVEMENT image. The results demonstrate that initial filling-in and parallel synthesis are able to connect regions that are not identified as similar by the relabelling process. The reconstruction of the hole is done using only self-region connections, as shown in Figure 5.23 (d), as the regions of the pavement (pink) and the road (green) connect themselves and are then flood-filled. Although the pavement regions on the right and the left of the hole should have pair connections, the colour (of the original image) corresponding to the region on the right of the hole is fairly different. Thus, the relabelling process did not identify them as similar regions.

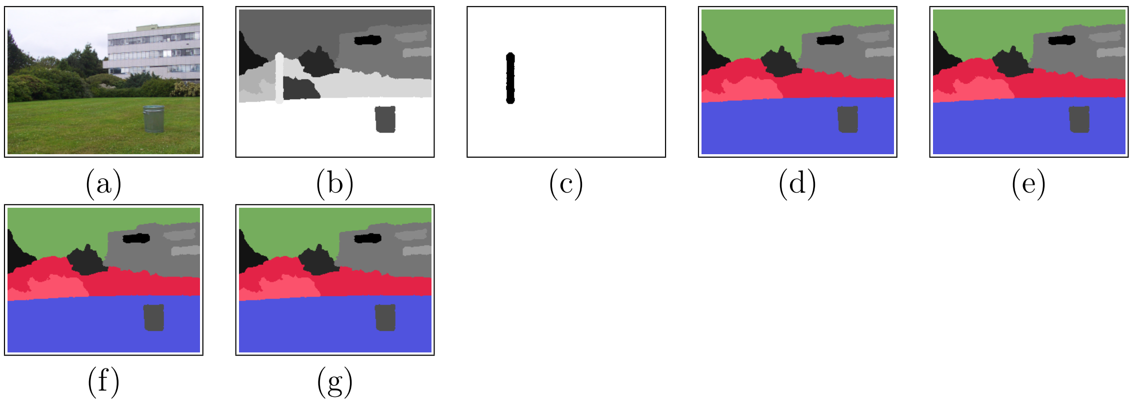


Figure 5.21: Changing the size of the hole for the DUSTBIN image to see the changes in the structure: (a) original image, (b) segmented image, (c) segmented image mask for the new hole, (d) reconstructing hole structure, (e), (f) and (g) first, middle and final iterations of synthesising the reconstructed segmented image.

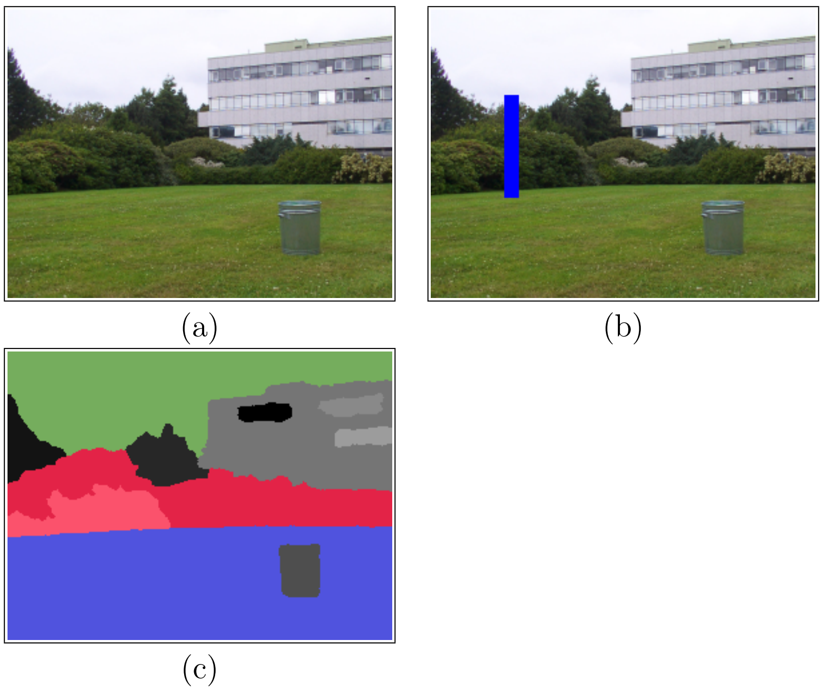


Figure 5.22: Zoom in results for the improved DUSTBIN image: (a) original image, (b) original image with a hole, and (c) final reconstructed image

After self-region connections, the remaining hole area is initially filled in as shown in Figure 5.23 (f). The hole is then synthesised starting from first iteration in Figure 5.23 (g) and ending at the final one in Figure 5.23 (i). Although the pavement regions are not connected at the beginning, the iterative parallel synthesis is able to connect such regions with the help of the initial filling in, as shown in Figure 5.23 (h) and (i). This is because the hole is horizontally narrower and there are no available patches of pink and green areas outside the hole. The final image in Figure 5.23 (i) shows proper and clear propagation of neighbouring HBR structures into the hole. Note that the value of neighbourhood size was set to 15

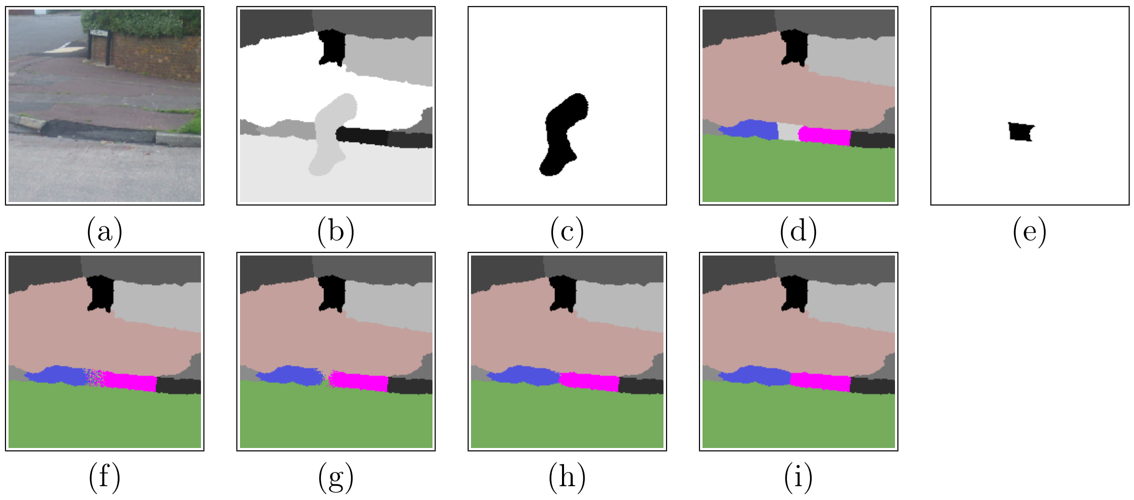


Figure 5.23: Hole structure reconstruction results for the PAVEMENT image: (a) original image, (b) segmented image, (c) hole, (d) reconstructing hole structure, (e) undecided area, (f) initial filling in of the undecided area, (g), (h) and (i) first, middle and final iterations of synthesising the reconstructed segmented image.

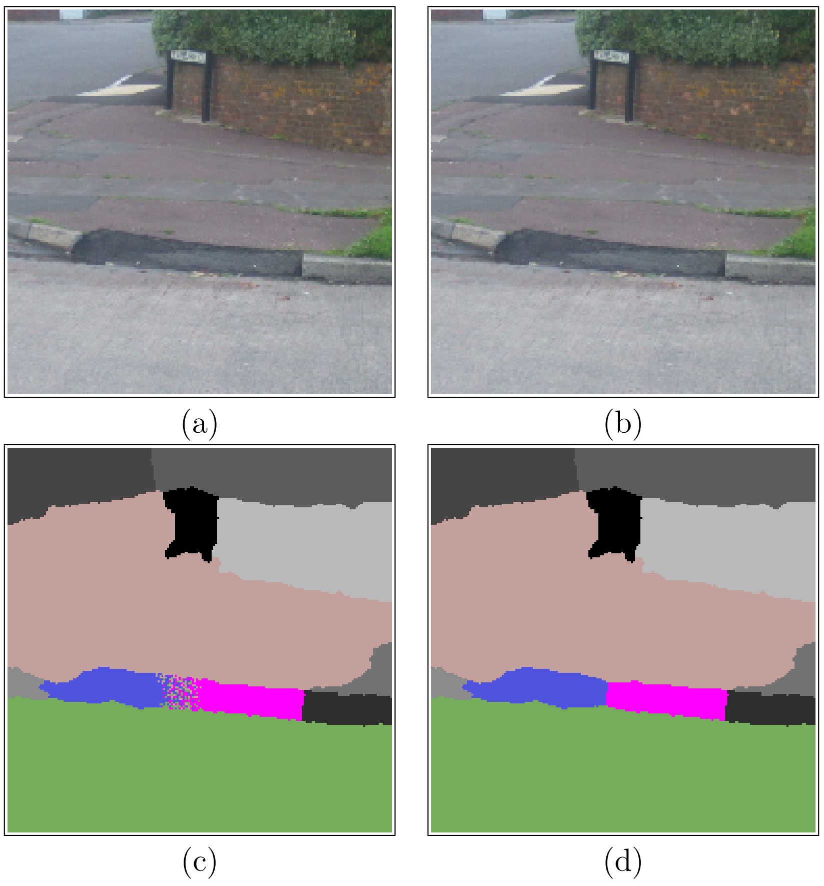


Figure 5.24: Zoom in results for the PAVEMENT image: (a) original image ,(b) original image with a hole, (c) initial filling in, and (d) final reconstructed image.

to capture properly the underlying structure of the pavement. Figure 5.24 shows a zoom in of the the PAVEMENT image results.

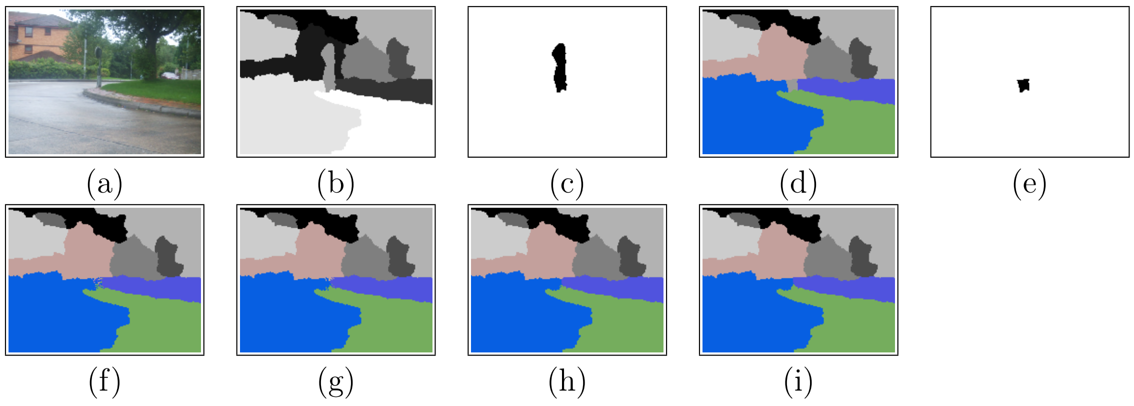


Figure 5.25: Hole structure reconstruction results for the ROUNDABOUT image: (a) original image, (b) segmented image, (c) hole, (d) reconstructing hole structure, (e) undecided area, (f) initial filling in of the undecided area, (g), (h) and (i) first, middle and final iterations of synthesising the reconstructed segmented image.

Figure 5.25 shows the results of structure reconstruction for the ROUNDABOUT image. It contains a sign that need to be removed and plausibly filled in by the surrounding region structures. Therefore the image is segmented and the hole is connected as shown in Figures 5.25 (b) and (d) respectively. Only self-region connection were applied here to the bush (pink) and roundabout(green) regions, creating a large structure of the hole and leaving at the same time a small area to be filled-in by the initial filling-in process. The initial filling-in was able to provide a good initialisation as more road and grass pixels are added than the bushes pixels, as shown in Figure 5.25 (f). This guides the iterative parallel synthesis to match this structure with the surroundings and this results in propagating regions structure plausibly in the hole, as shown in the final image in Figure 5.25 (i). This is clear in the construction of well defined boundaries between these regions. Note that the small road area on the right side of the hole was not identified by the segmentation as a distinct area, but was selected as part of the grass region on the right hand side of the hole. This does not affect the structure in this case, but may affect the quality of the match at the texture synthesis stage. Figure 5.26 shows zoom in of the the ROUNDABOUT image results.

In Figure 5.27 (a) the original image (BUNGEE-JUMPER) from [9] has a large area to be removed (the bungee-jumper), and then replaced by proper structure. The house structure in general was propagated well in the hole forming a one block house, as shown in Figure 5.27 (d). However the bottom edge of the house was not reconstructed properly during the synthesis process as possible matching patches have green area dominantly at the top of them and not enough light brown areas on the bottom of these patches. Also, there are not enough source patches that have straight lines structures, but angled patches along the edge of the hole between the

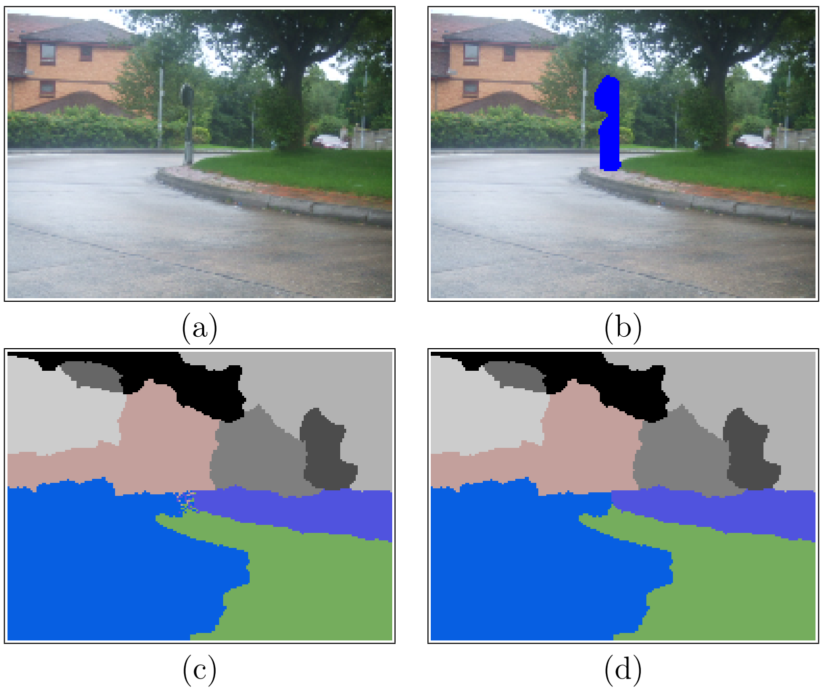


Figure 5.26: Zoom in results for the ROUNDABOUT image: (a) original image, (b) original image with a hole, (c) initial filling in, and (d) final reconstructed image.

green and light brown areas. All of these factors created the angled line between the green and brown areas, as shown in Figure 5.27 (g), (h) and (i). The other regions are reconstructed well forming region boundaries plausibly propagated inside the hole, as seen, e.g., in the reconstruction of boundaries between the blue and light brown regions. Note that the value of the neighbourhood size was set to 21 to properly capture the structure (of the texture) of the house which covers too many pixels for the normally used size of 11. Figure 5.28 shows a zoom in of the the BUNGEE-JUMPER image results.

Figure 5.29 shows the results of structure reconstruction for the TUBE image. Although the structure reconstruction of the most of the hole regions were clearly created (e.g. the tree, tubes, and grass), it failed to connect the vertical and horizontal tubes. This is because this kind of connection requires high level knowledge that our method doesn't have. During the connection process, similar regions of the horizontal tube on one hand and the vertical tube on the other hand were connected and flood-filled. The grass region on the left of the vertical tube is self-connected as shown in Figure 5.29 (d). The remaining unfilled regions are initialised where the lower grass region is completely surrounded by grass regions, hence producing suitable filling as shown in Figure 5.29 (f). However, this will not affect the synthesis process where the vertical tube will not be able to connect to the horizontal one since there are no examples of green and brown patches outside the hole. Therefore, due to the fact that our method does not have high level

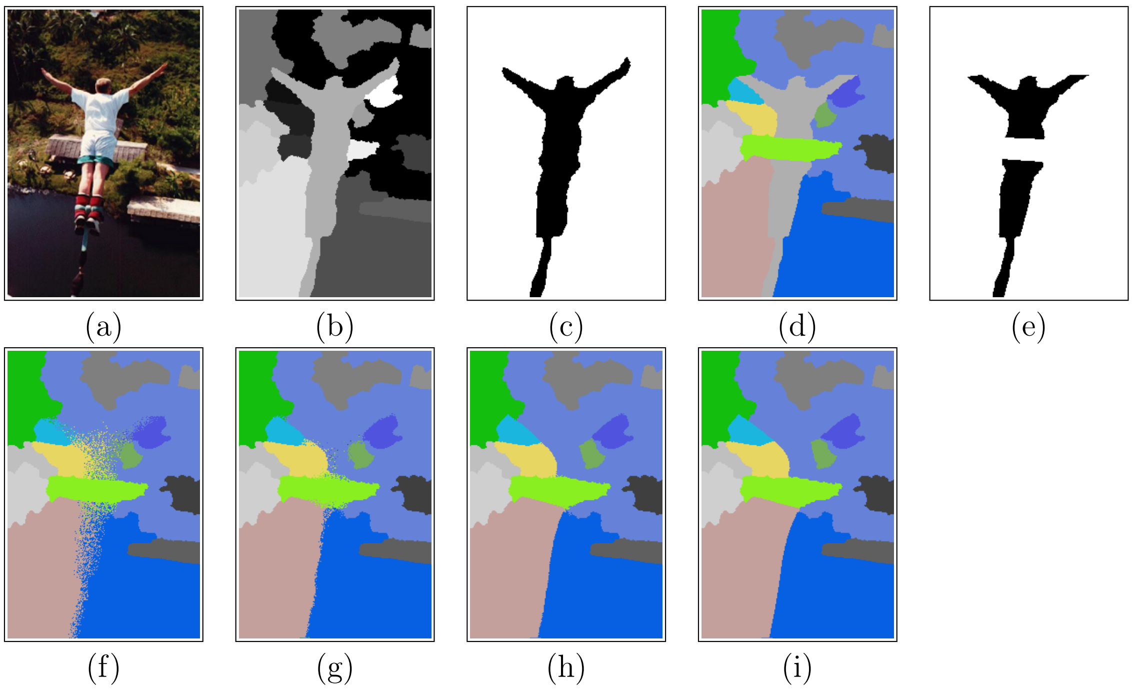


Figure 5.27: Hole structure reconstruction results for the BUNGEE-JUMBER image: (a) original image from [9], (b) segmented image, (c) hole, (d) reconstructing hole structure, (e) undecided area, (f) initial filling in of the undecided area, (g), (h) and (i) first, middle and final iterations of synthesising the reconstructed segmented image.

knowledge of what the real hole structure should be, some user interaction is necessary to edit or add connections. In this case, the user can connect the vertical tube with the horizontal ones and run the synthesis process to match the surrounding structures. Figure 5.29 (j) shows a manual user interaction which connects and flood-fills the vertical and horizontal tubes. Note that the user connection in this case is not performed before the synthesis process, but after, because of the same reason of unavailable examples. Thus, the synthesis after the manual connection would not be helpful in this case as it could affect the connected regions and replace them with undesirable structure. As a result, manual connection should be performed on the final image of the synthesis result. For this image, the threshold of the colour quantisation was set to 400 instead of 200 to exclude unnecessary regions. Note that the value of neighbourhood size was set to 15 to properly capture the structure of the tree. Figure 5.30 shows a zoom in of the the TUBE image results.

In Figure 5.31, the original image (ELEPHANT) has a large area to be removed (the elephant), and then replaced by proper structure. The structure of the hole is created, having well defined boundaries of the sand, river, shore, trees and mountain regions, as shown in Figure 5.31 (i). Although the tree regions on the right and left sides of the hole are connected plausibly, the boundary that

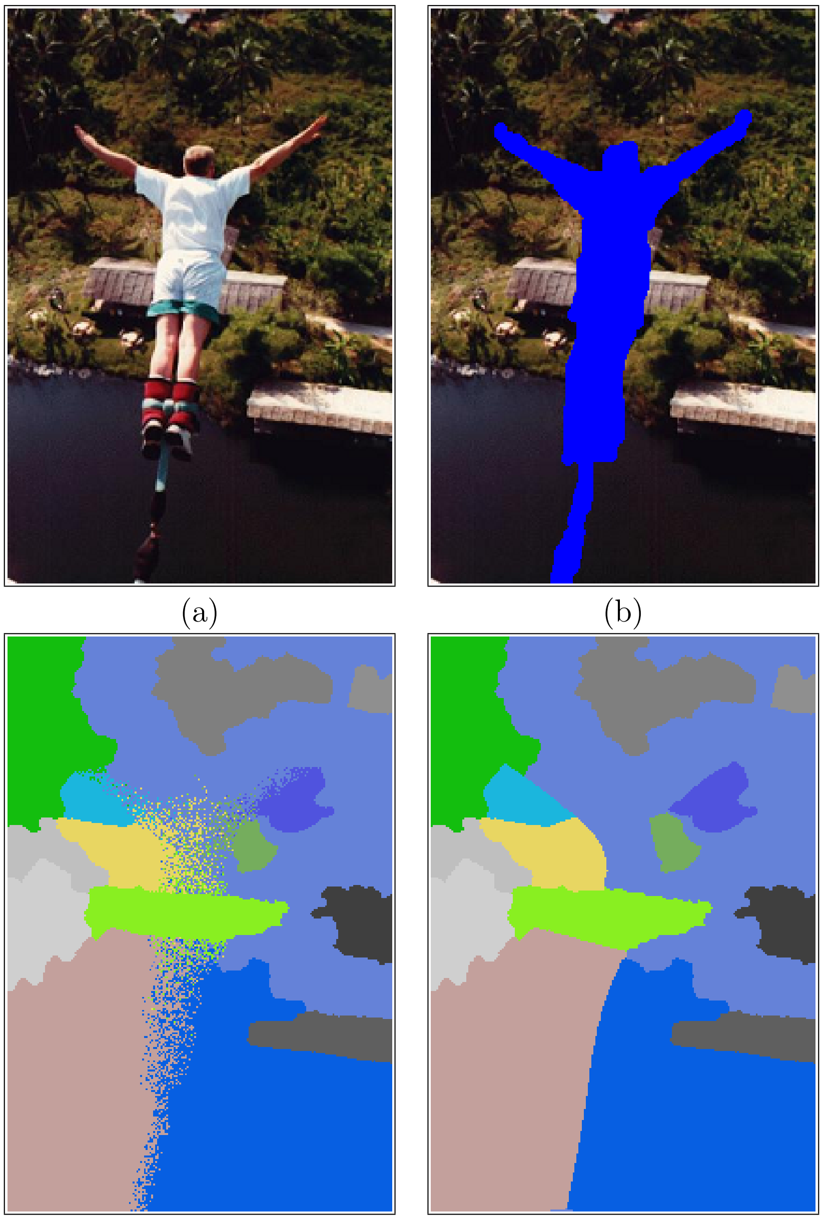

(a) (b) (c) (d)

Figure 5.28: Zoom in results for the BUNGEE-JUMBER image: (a) original image from [9], (b) original image with a hole, (c) initial filling in, and (d) final reconstructed image.

separate them from the mountain region is not created properly as its structure does not match well the surrounding structure along that area. This is because the connected line between the tree regions are angled slightly (due to region segmentation) and the best match to this configuration is taken from the angled small area of the tree on the right side of the hole. However, this could be fixed by increasing the neighbourhood size to capture the shape of the boundaries between such regions. Figure 5.32 shows a zoom in of the the ELEPHANT image results.

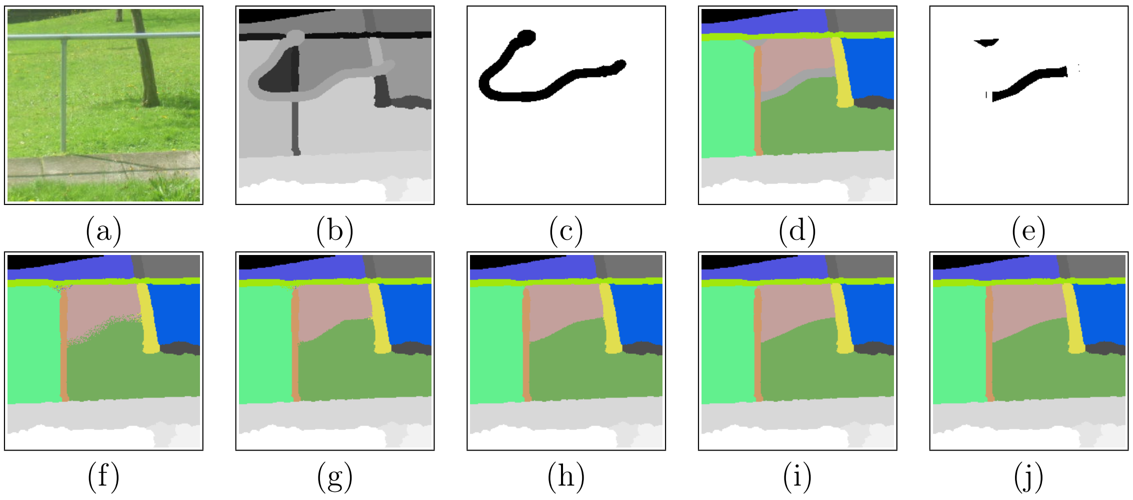


Figure 5.29: Hole structure reconstruction results for the TUBE image: (a) original, image (b) segmented image, (c) hole, (d) reconstructing hole structure, (e) undecided area, (f) initial filling in of the undecided area, (g), (h), and (i) first, middle and final iterations of synthesising the reconstructed segmented image, and (j) a manual user interaction connecting and flood-filling the region between the vertical and horizontal tube.

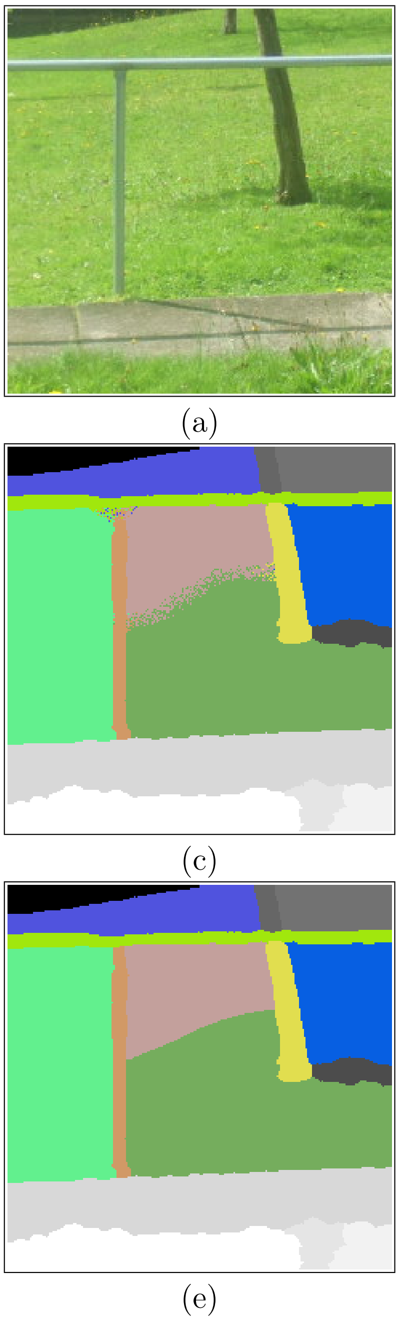

(a) (b) (c) (d) (e)

Figure 5.30: Zoom in results for the TUBE image: (a) original image, (b) original image with a hole (c) initial filling in, (d) final reconstructed image, and (e) a modified user image.

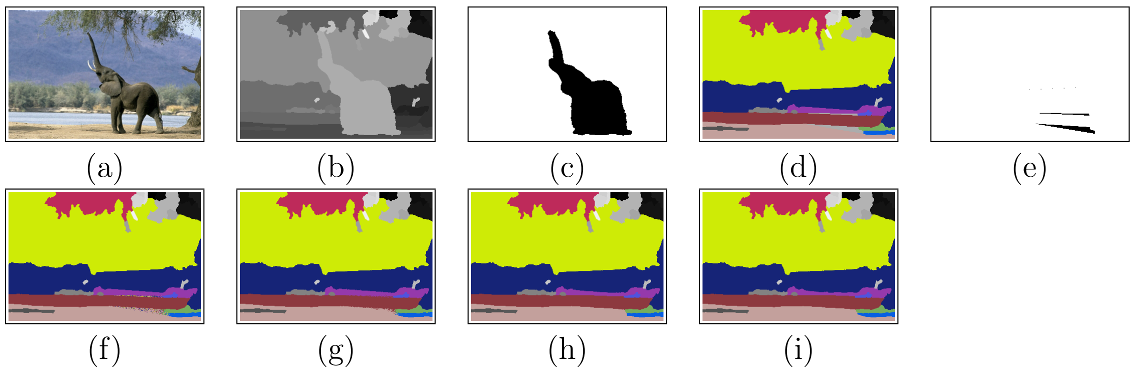


Figure 5.31: Hole structure reconstruction results for the ELEPHANT image, from [35]: (a) original, image (b) segmented image, (c) hole, (d) reconstructing hole structure, (e) undecided area, (f) initial filling in of the undecided area, (g), (h), and (i) first, middle and final iterations of synthesising the reconstructed segmented image.

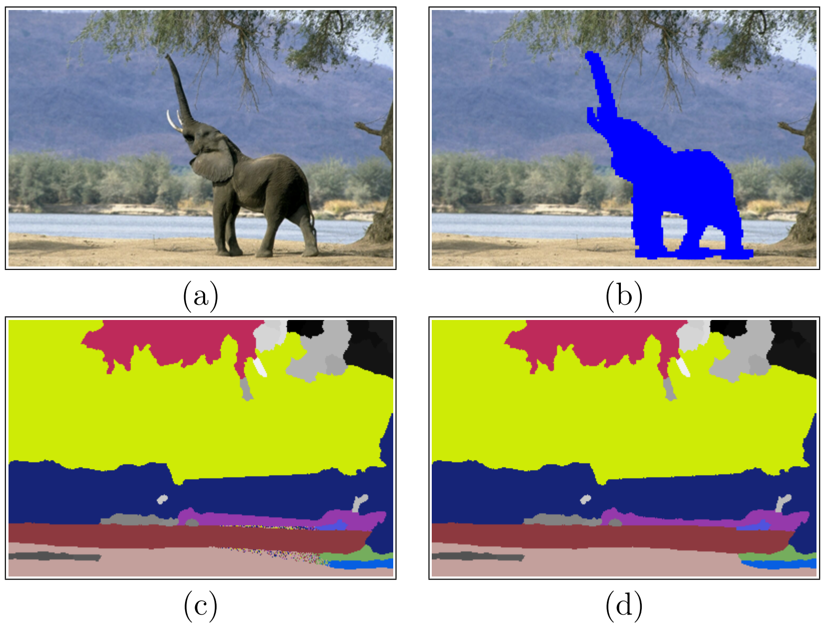


Figure 5.32: Zoom in results for the ELEPHANT image, from [35]: (a) original image, (b) original image with a hole, (c) initial filling in, (d) final reconstructed image.

Chapter 6

Image completion: combining both methods

6.1 Introduction

Texture synthesis methods are well suited for creating texture, but do not handle propagation of image structure, yet they can produce implicit structure for images with simple structure surrounding the hole as has been shown in Figure 4.14. However, the methods can fail with images that have large holes that are surrounded by complex image structures. As a result, explicit reconstruction of the structure of the hole is necessary in order to deal with such images to produce realistic results. The structure can then be used to create texture that will be coherent globally, hence producing plausible completion.

Our image completion approach is based on combining both explicit structure reconstruction (Chapter 5), and texture synthesis (Chapter 4). Figure 6.1 shows a general outline of the image completion process.

6.2 The completion method

Combining the methods of structure reconstruction and texture synthesis is important for producing plausible completion of holes in images. The structure reconstruction method creates the structure of the hole having regions with defined boundaries that matches the surrounding structure. One example of this reconstruction of structure is demonstrated in Figure 6.2 for the CLEDWYN image where the structure of the regions inside the hole match the surrounding regions, particularly for the tree region. Compared with the same image, but without

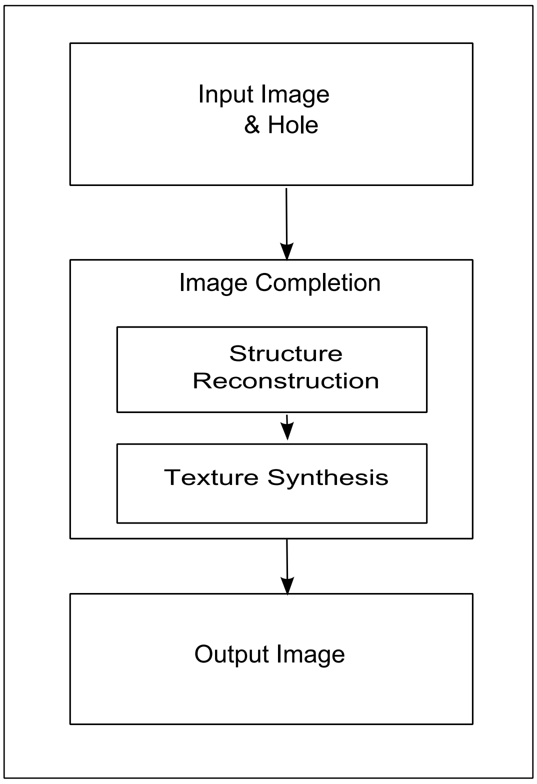


Figure 6.1: General outline of the image completion process

reconstructing the structure (only texture synthesis), the structure in the hole was not properly preserved, particularly the tree area, as shown in Figure 4.18. Also, without using the structure reconstruction of the hole, the completion can produce unrealistic filling as shown in Figure 4.16 where the region of the wall expanded into the grass area resulting in unrealistic completion. However, when the structure was reconstructed, it guided the texture synthesis to preserve the boundaries between the grass and wall regions, as shown in Figure 6.4.

The method of texture synthesis creates texture constrained by the reconstructed structure. This allows to synthesis textures from relevant areas of the image by searching for similar neighbourhoods that belong to the current region of the pixel being synthesised. Also, the initial filling in of the hole is constrained by the reconstructed structure. Each pixel being initialised will be taken from the relevant area according to that constraint. A pseudocode for the image completion method is described in Algorithm 4.

However, the processes of initialisation and texture synthesis will be highly influenced by the quality of the segmentation. The segmentation may fail to segment images by including irrelevant regions or excluding relevant ones, and this will affect the initialisation of the hole by copying pixels from irrelevant areas. This will affect the synthesis process in the sense that the neighbourhood match of a pixel

Algorithm 4 Pseudocode for the image completion method

```
Initialise the hole as described in Algorithm 2, but constrained to the regions
in the final reconstructed image described in Algorithm 3.
DO
   For each pixel in the hole
      For each pixel outside the hole and **inside the corresponding region of
      the final reconstructed image**
         Neighbourhood template = get neighbourhood (current pixel)
         IF the Neighbourhood template have a number of pixels equal to or
         greater than the number of pixels in the current neighbourhood
            Compute Euclidean distance for the two neighbourhoods
            Save the location of the pixel and its neighbourhood distance in a NH
            list
         ENDIF
      ENDFOR
      Possible matches list = all pixels (i,j) in the NH list where Euclidean
      distance(i,j) <= Min(Euclidean distance) * randomness factor
      Best match = Pick up randomly a mach from the Possible matches list
      Pixel value = the value of the pixel at the centre of the Best match
      Copy the Pixel value to the current pixel in the hole
   ENDFOR
While ( the ratio of changed pixels of the current and previous iteration results
is greater than P)
```

in the hole will not have plausible values, which can result in mismatches in the neighbourhood matching process. Another problem related to the quality of the segmentation is when the method produces too small regions. This can result in difficulty of reconstructing structure due to the possibility of mismatch in identifying similar regions and uncertainty in connecting them, hence creating a structure that might not be compatible with the surrounding (creating large structure that is different from the one of the small regions, especially when the hole is large). The texture synthesis will also be affected when using such small regions by not having enough variation and examples which can result in replication of complete patches.

6.3 Results and discussion

The results of our image completion method are based on integrating both structure reconstruction and texture synthesis. The method has been applied to different types of textured images, and has produced good results as can be seen in Figures 6.2–6.27. Other result images are shown in Appendix A.

Figure 6.2 shows the CLEDWYN image with a building to be removed and replaced with proper structure and texture. The structure of the hole was already created in the structure reconstruction stage, discussed in Section 5.4, producing acceptable boundaries between the regions of the hole, as shown in Figure 6.2 (c). This structure is used when synthesising texture as a constraint. This appears clearly in the initial filling-in of the hole and the synthesised images where boundaries between the sky and tree regions are well defined and match the surrounding structure, compared with Figure 4.18 (g) where the boundaries between the regions were not well preserved. In Figure 6.2 (d), the initial filling-in is able to initialise plausible values in the hole. However, the sky area has noise which is produced from parts of the tree area near the hole between the sky and tree regions. These tree parts are considered by the constrained structure reconstructed image as sky because it was not "correctly" identified by the segmentation as trees at the segmentation stage. The noise is then modified by the texture synthesis to form sky region as shown in Figures 6.2 (e), (f) and (g). The final synthesised image in Figure 6.2 (g) shows that the image structure is well propagated inside the hole where clear boundaries between the sky and tree regions are created. Also, the texture is properly synthesised capturing the structure of the surrounding texture elements and also producing branches of the tree inside the hole. Figure 6.3 shows a zoom in of the original image, the initial filling in and the final synthesised image.

In Figure 6.4, the BUSH image has a hole that is surrounded by regular texture (wall) around the upper half of the hole and different grass textures (green and brown) around the lower half of the hole. In Figure 6.4 (g), the structure of the hole is created with well defined boundaries between the wall and the grass areas. Also, the structure of the wall elements are captured and aligned well. Its texture is also well synthesised. However, the texture synthesis of the grass only propagates green grass which does not match the area of the surrounding brown grass. This is because the brown grass area is not identified in the segmentation as a separate region from the green grass region and therefore, the initial filling-in in the area mainly copies pixels from the green grass, as shown in Figure 6.4 (d). This is due to the fact that at this hole area, the green grass region is more available for copying pixels (outside the hole) in the area of the Gaussian distribution than the brown grass area. This affects the iterative synthesis results which produce green grass instead of brown grass in that area of the hole, as shown in Figures 6.4 (e), (f) and (g). However, this problem can be solved by using a different segmented image which identifies the brown grass as a separate region. Thus, the initial filling-in image using this new constrained image is able to copy pixels from the brown grass areas, as shown in Figure 6.6 (d). This would "coerce" the iterative

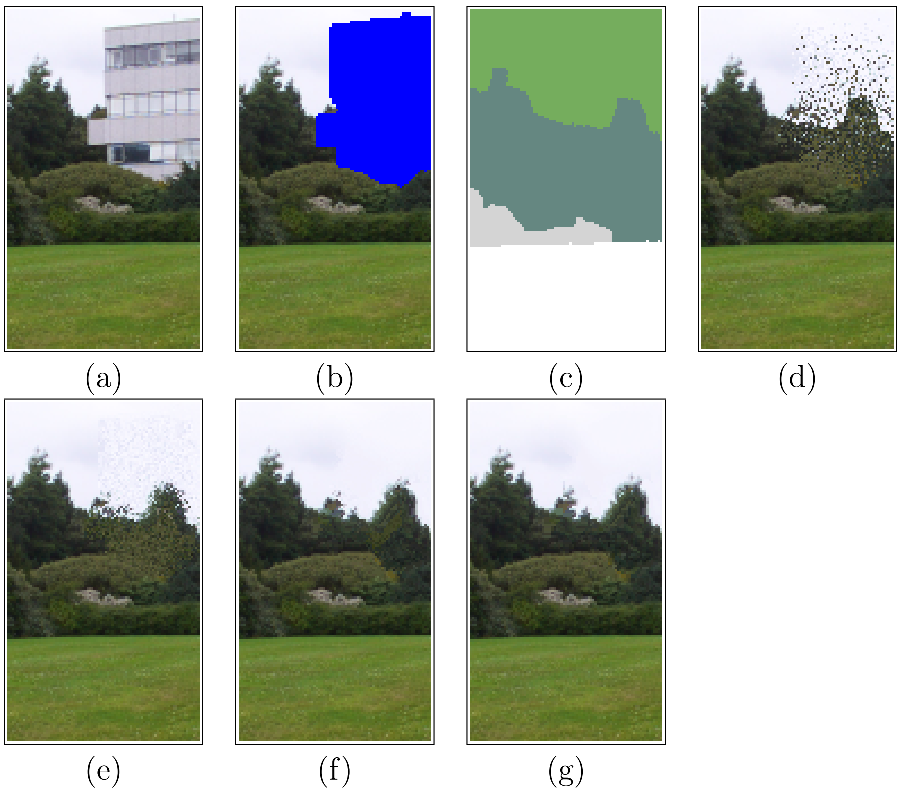
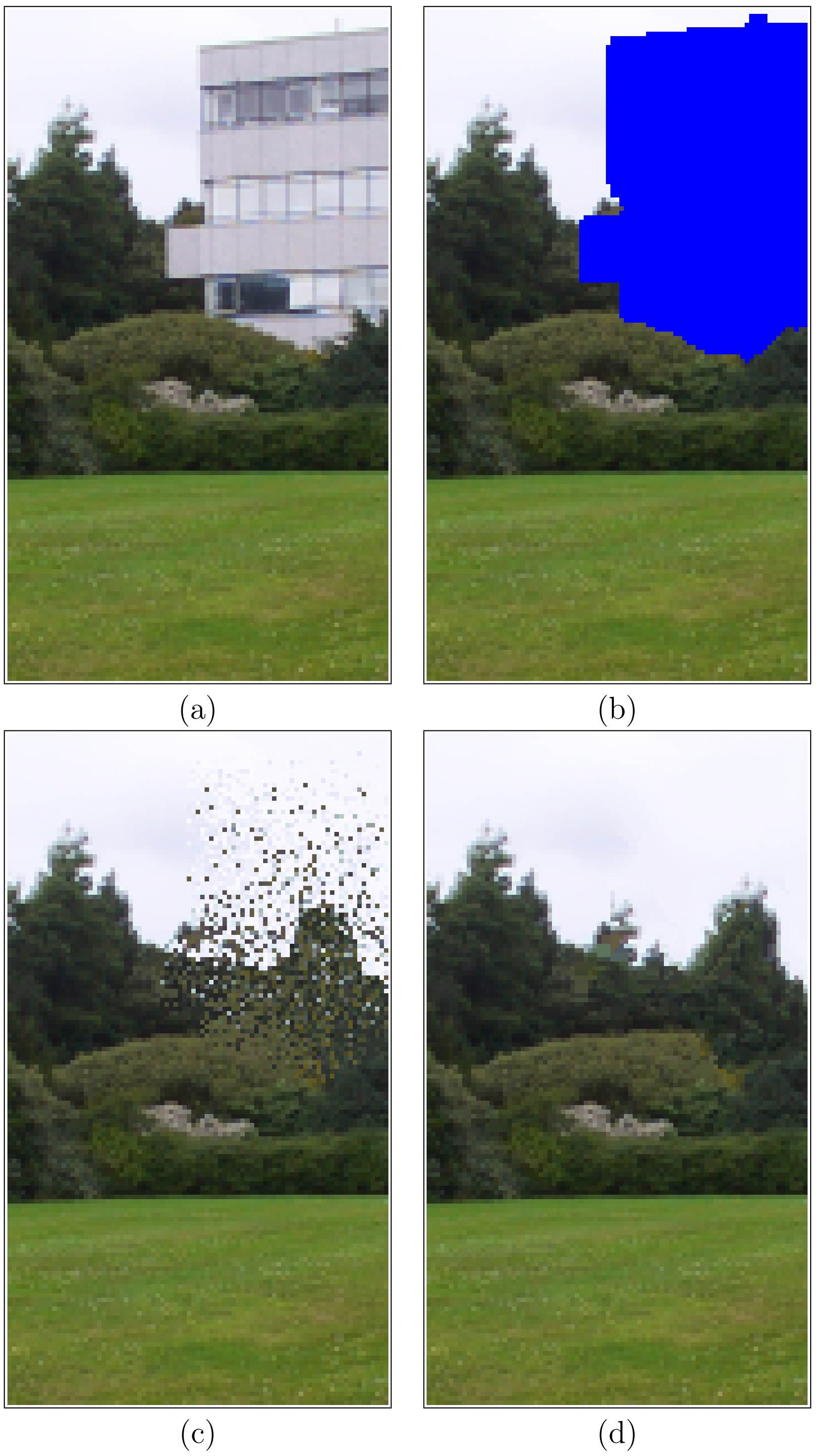


Figure 6.2: Image completion results for the CLEDWYN image. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, (d) initial filling in of the hole area, (e), (f) and (g) first, middle and final iterations of image completion.

texture synthesis to create plausible brown grass texture in this area of the hole, as shown in Figure 6.6 (g). Figures 6.5, 6.7 show zoom in of the original, the initial filling in, and the final synthesised images of the BUSH image and its improved result respectively.

In Figure 6.8, the hole of the DUSTBIN image is surrounded by differently textured areas such as smooth sky area around the upper part of the hole, and other varied textures (bushes and grass) on the middle and bottom of the hole. The structure of the hole is properly created having clear boundaries between the regions of the hole. Also, the texture of the hole is propagated satisfactorily inside the hole. However, implausible repetition of parts of the texture is noticed, see Figure 6.8 (g). For example, the area of the hole between the sky and bushes has “small” tree branches that are repeated along the horizontal line of that area. This is because there are no sufficient patches similar to the ones in the hole and therefore the matching often selects these patches, as shown in Figure 6.8 (g). However, when a narrowed hole is used to allow more patches along the area between the sky and the bushes, more patches are included in the search for the possible matches which resulted in having more variations of the patches, hence



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

Figure 6.3: Zoom in results for the CLEDWYN image. (a) original image, (b) original image with a hole, (c) initial filling in of the hole area, and (d) final result.

producing better results, as shown in Figure 6.10 (g). Figures 6.9, 6.11 show zoom in of the original image, the initial filling in and the final synthesised image of the bush and its improved result respectively.

In Figure 6.12, the hole spans different types of textured regions where one of

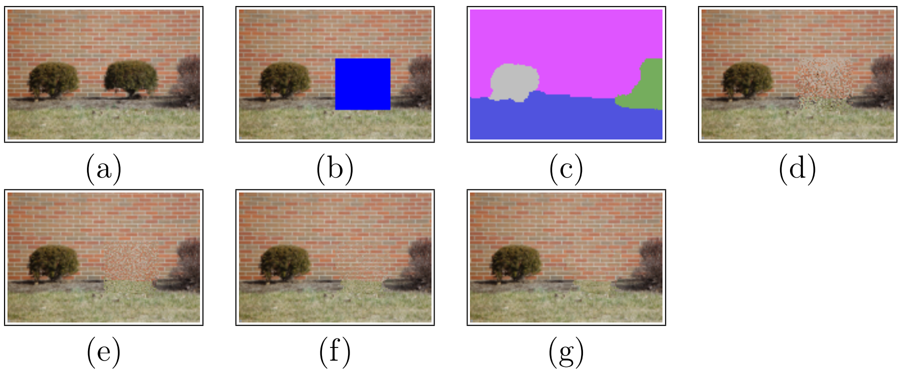


Figure 6.4: Image completion results for the BUSH image. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, (d) initial filling in of the hole area, (e), (f) and (g) first, middle and final iterations of image completion.

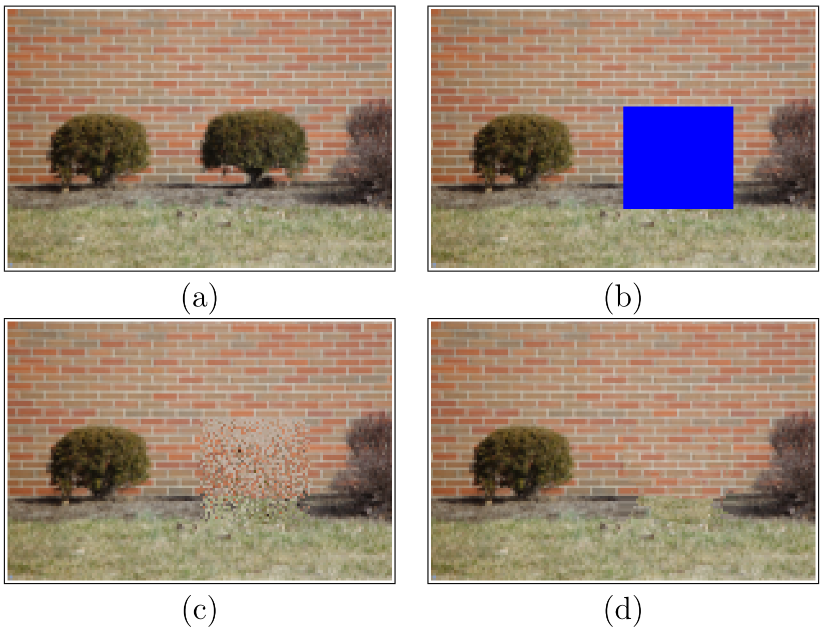


Figure 6.5: Zoom in results for the BUSH image. (a) original image, (b) original image with a hole, (c) initial filling in of the hole area, and (d) final result.

them is not identified in the segmentation process as a separate region (the pavement's light gray area) and the other (the kerb) is identified as two regions in the structure reconstruction process (where it should be one). Although the structure reconstruction process was not able to reconstruct the pavement area properly, the texture synthesis propagates texture well in this area.

The kerb region is identified in the segmentation as two separate regions for which the structure reconstruction method was not able to connect the two regions due to the different texture characteristics of these regions. However, it managed to create reasonable boundaries between itself and the pavement region on the one hand, and the road on the other hand. It also connects part of the kerb on the

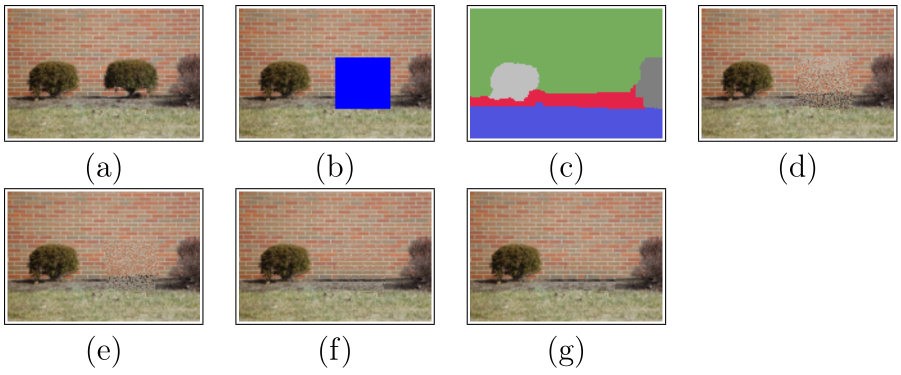


Figure 6.6: Improving the image completion result for the BUSH image by using a new constrained image. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, (d) initial filling in of the hole area, (e), (f) and (g) first, middle and final iterations of image completion.

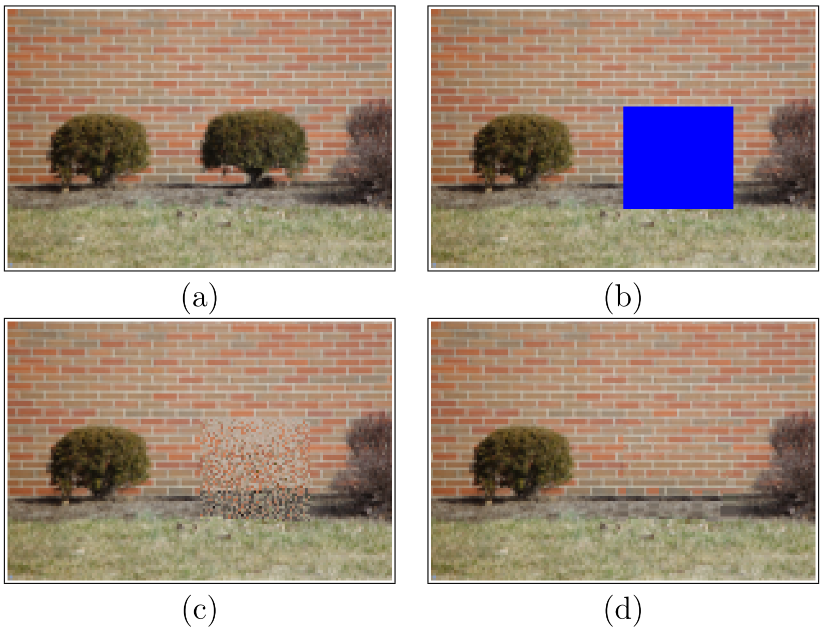


Figure 6.7: Zoom in results for the improved BUSH image. (a) original image, (b) original image with a hole, (c) initial filling in of the hole area, and (d) final result.

right of the hole with its other part on the left of the hole. The texture created in the kerb region is also synthesised well, having this area of the hole created with different kerb textures that meet in the middle of the hole, and this makes a plausible image completion. Figure 6.13 shows a zoom in of the original image, the initial filling in and the final synthesised image.

In Figure 6.14, the ROUNDABOUT image contains a speed limit sign to be removed and replaced by structure and texture from the surrounding regions. The structure of the hole is already reconstructed, having well defined boundaries between the regions of the hole, as discussed in Section 5.4, and shown in Figure 6.14 (c). The propagation of texture area of the roundabout, grass, and road are well created,

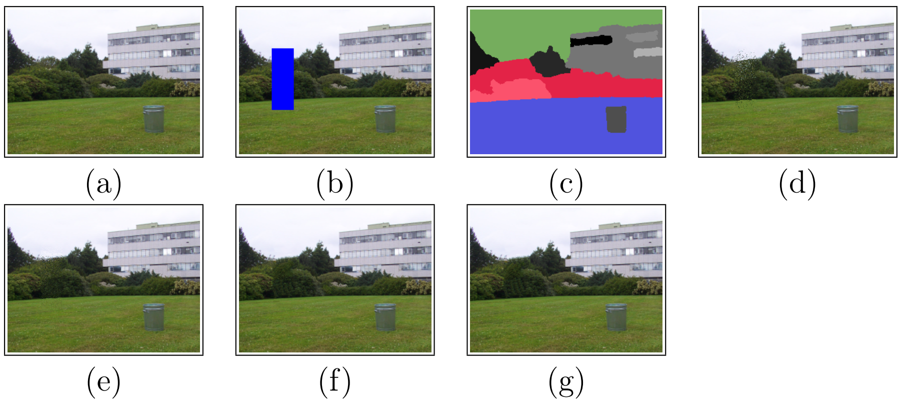


Figure 6.8: Image completion results for the DUSTBIN image. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, (d) initial filling in of the hole area, (e), (f) and (g) first, middle and final iterations of image completion.

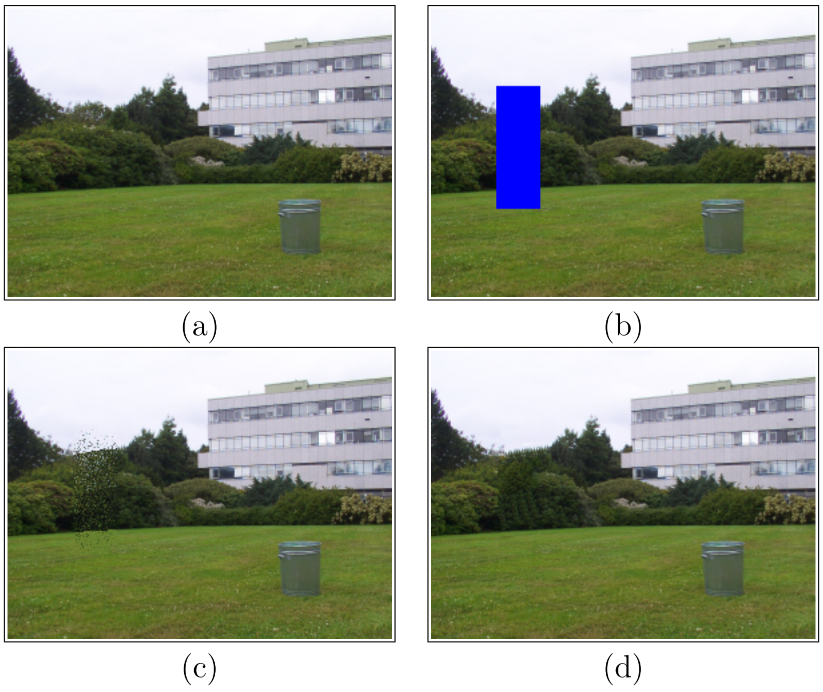


Figure 6.9: Zoom in results for the DUSTBIN image. (a) original image, (b) original image with a hole, (c) initial filling in of the hole area, and (d) final result.

as shown in Figure 6.14 (g). However, the reconstruction of the kerb on the right of the hole is slightly different from the left. This is because the structure of the kerb on the right of the hole is slightly different from the one on the left of the hole, as one is higher than the other. The initial filling in is plausible having pixels copied from the surrounding bushes and roundabout areas into their corresponding regions in the hole, as shown in Figure 6.14 (d). The initialisation of the road however, does not have enough road pixels from the region on the right of the middle of the hole, but this is fixed by the iterative texture synthesis process, as shown in Figure 6.14 (g). Also, the texture synthesis creates texture that look

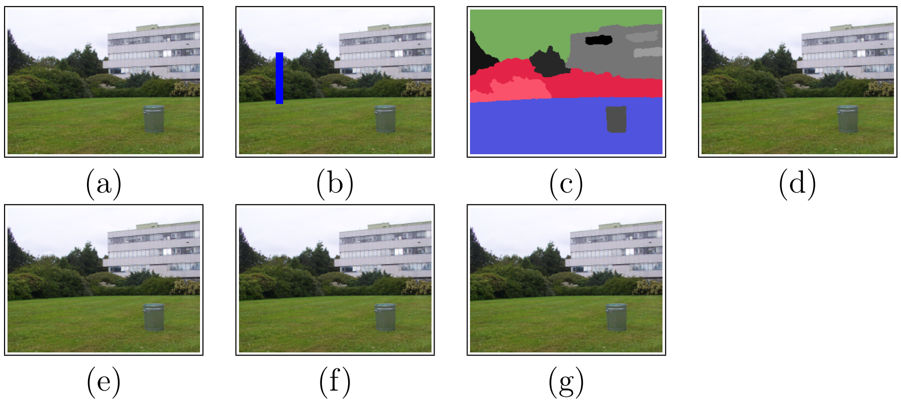


Figure 6.10: Narrowing the hole size for the DUSTBIN image. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, (d) initial filling in of the hole area, (e), (f) and (g) first, middle and final iterations of image completion.

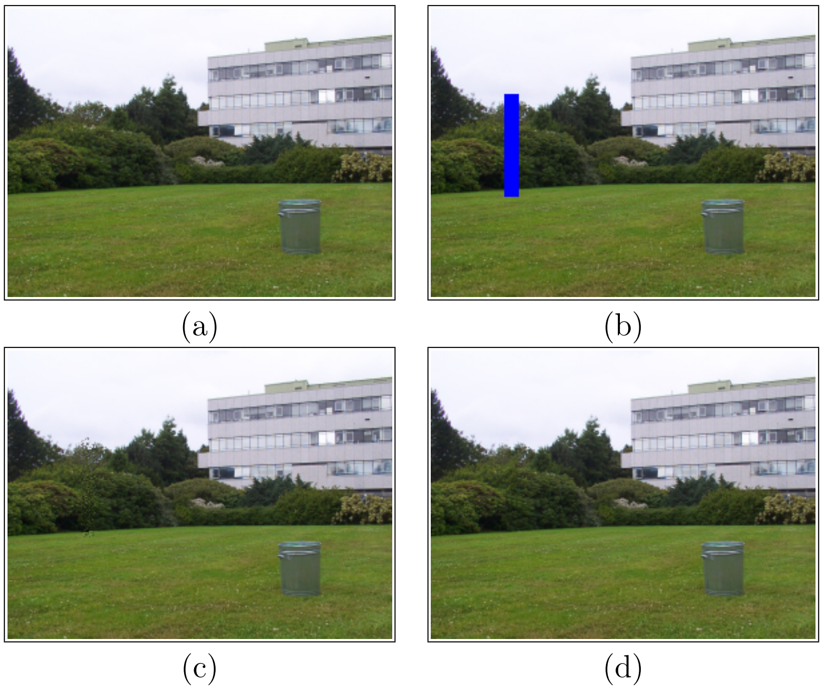


Figure 6.11: Zoom in results for the DUSTBIN image with narrowed hole. (a) original image, (b) original image with a hole, (c) initial filling in of the hole area, and (d) final result.

realistic in the hole, as show in in the Figure 6.14 (g). Figure 6.15 shows a zoom in of the original image, the initial filling in and the final synthesised image.

Figures 6.16 (a), (b) show a large area to be removed from the BUNGEE-JUMBER image and then replaced by proper structure and texture. The areas of the hole are properly filled in with the surrounding textures. However, the filling in of the area down the house with water does not match the surrounding texture. This filling in of the areas is because the shore area in the bottom of the house is not connected with the other shore (in the right side of the middle of the hole) in the structure

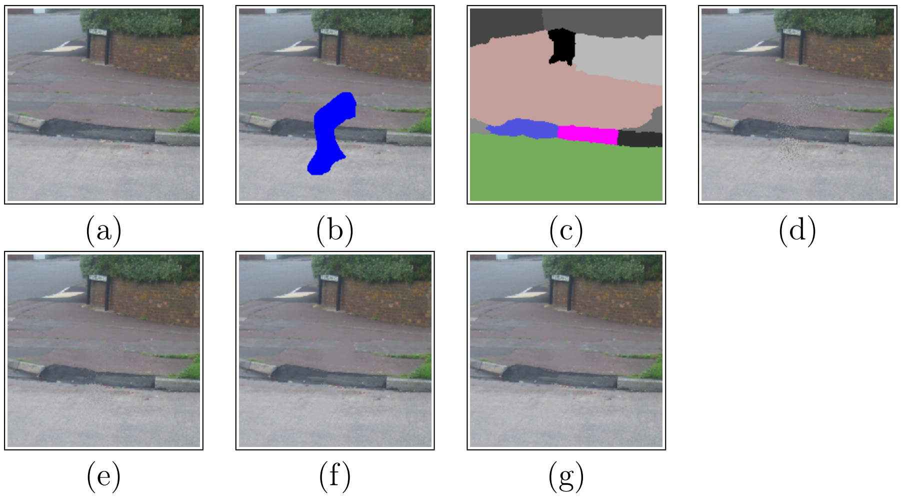


Figure 6.12: Image completion results for the PAVEMENT image. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, (d) initial filling in of the hole area, (e), (f) and (g) first, middle and final iterations of image completion.

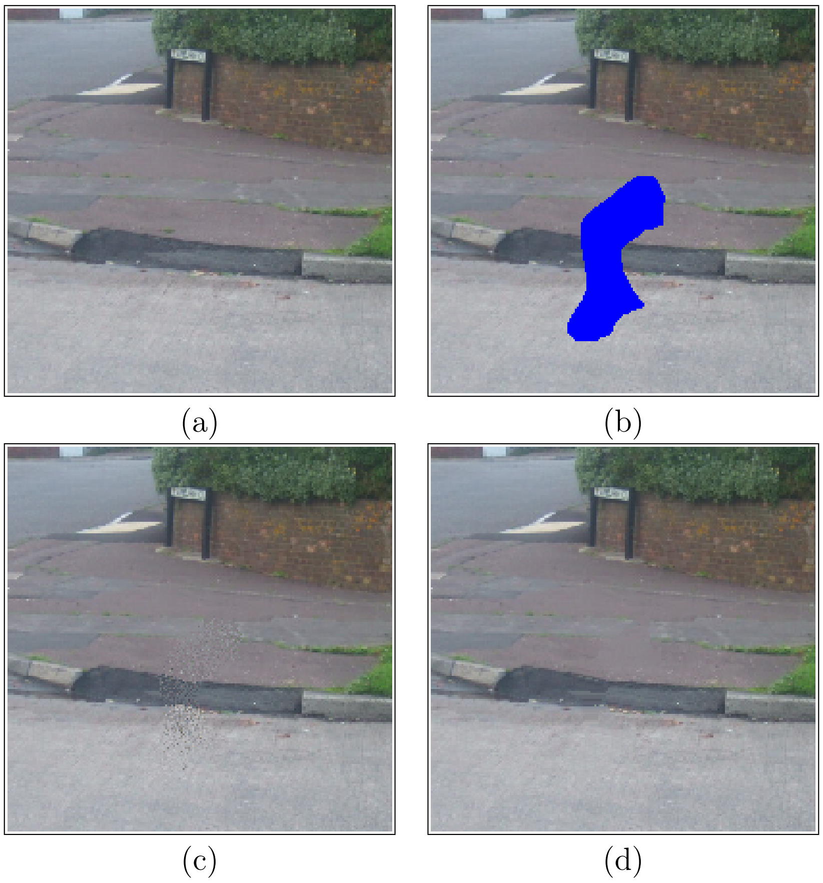


Figure 6.13: Zoom in results for the PAVEMENT image. (a) original image, (b) original image with a hole, (c) initial filling in of the hole area, and (d) final result.

reconstruction process because this region was not identified in the segmentation as a separate region, but part of the water region. Also, because the surrounding

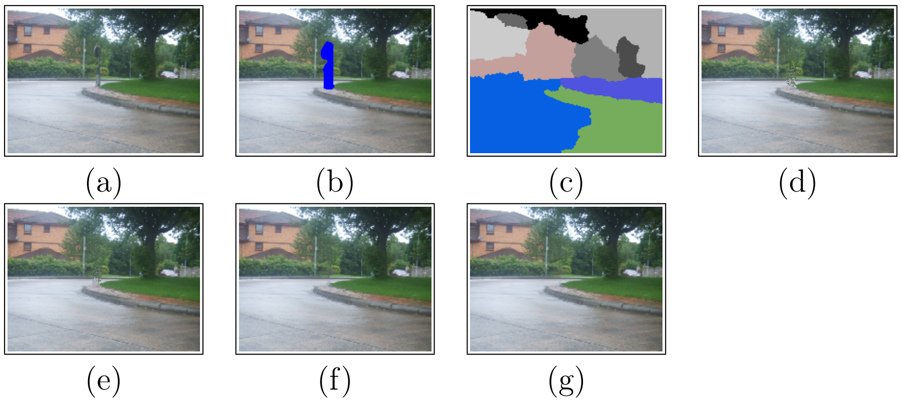


Figure 6.14: Image completion results for the ROUNDABOUT image. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, (d) initial filling in of the hole area, (e), (f) and (g) first, middle and final iterations of image completion.

area to that region is a shadow of the house which is dark area where the closest region in terms of colour is the water region. The region covers a larger area than the shore area, which results in more water pixels selected at the beginning of the texture synthesis process than the shore due to the randomisation effect at the beginning of the texture synthesis process. The texture of the regions of the hole is synthesised well, having clear structure of the house with proper alignments of its block elements. The other region textures are also created well where the ground region above the house matches the surrounding ground areas. The texture of the water is also propagated well inside the hole. Figure 6.17 shows a zoom in of the original image, the initial filling in and the final synthesised image.

Figure 6.18 shows comparisons with results of other methods. Figure 6.18 (b) shows the result of applying the image inpainting method presented in [9], which produces a great deal of blur artifacts. This is because of the use of diffusion process to propagate colour inside the hole. Also, a complete lack of texture is clearly shown in the result due to the diffusion process. In Figure 6.18 (c), the structure is well propagated into the hole, especially the shore area and the bottom of the house. However, the roof structure is not well preserved. Also, the texture is generally created plausibly, however, parts of the shore area (grass) is expanded towards the water area. Moreover, visual artifacts in the area of the bungee cord is obvious where texture was not propagated well in that area. These artifacts are marked using "circular" dots. Our result shown in Figure 6.18 (d) produces better results in term of creating plausible image structure, particularly the reconstruction of the roof of the house. however, the reconstruction of the shore structure was not preserved as the shore areas were not identified as separate regions in the segmentation stage, hence not connected. The texture of the regions

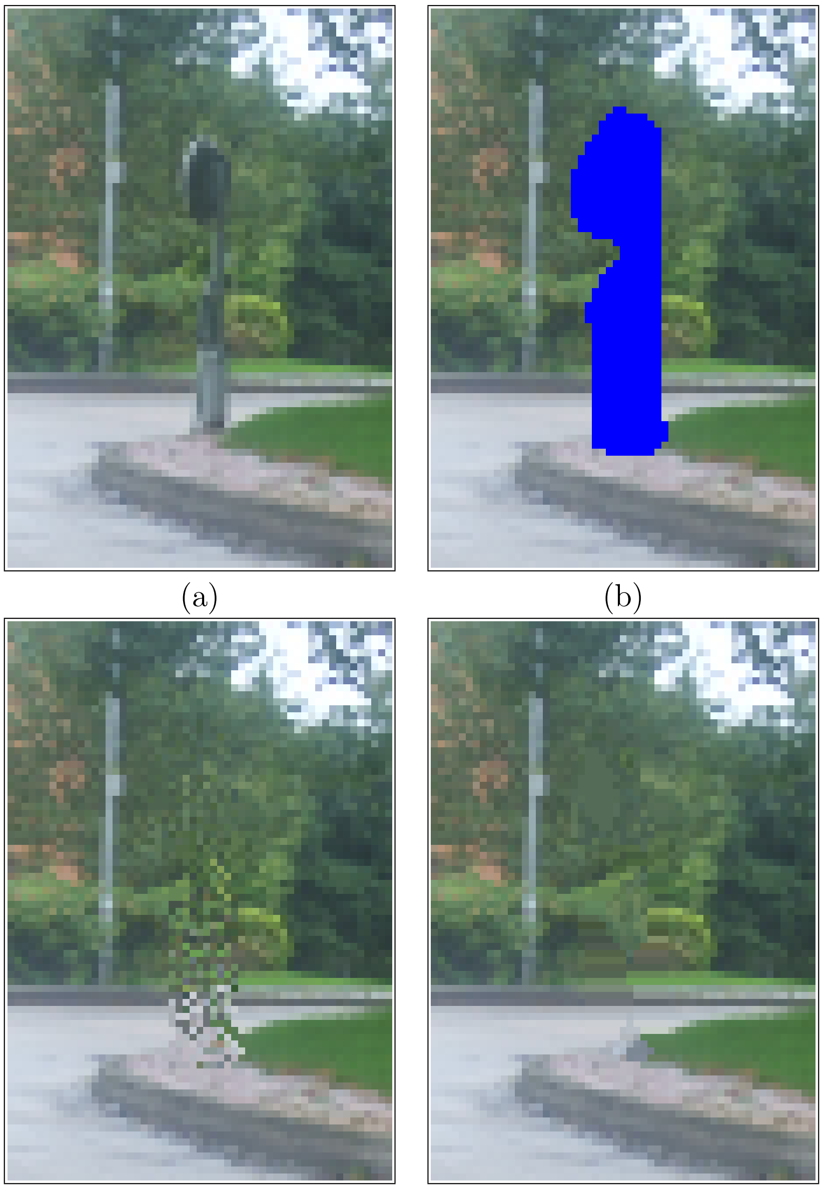

(a) (b) (c) (d)

Figure 6.15: Zoom in results of the area of the hole and parts of the surroundings for the ROUNDABOUT image. (a) original image, (b) original image with a hole, (c) initial filling in of the hole area, and (d) final result.

inside the hole is also propagated well, having clear structure, particularly the house region. Figure 6.19 shows zoom in of the comparison results.

In Figure 6.20 (g), the structure of the hole of the TUBE image is reconstructed well for the vertical and horizontal tubes, and the tree. These regions connected inside the hole with well defined boundaries. However, the horizontal tube is not connected to the vertical one as it was not connected in the reconstructed segmented image shown in Figure 6.20 (c). Therefore, some user interaction is needed to modify the reconstructed image, as shown in Figure 6.22 (c), (see Section 5.4 for more details). This new reconstructed image is used for the texture synthesis and

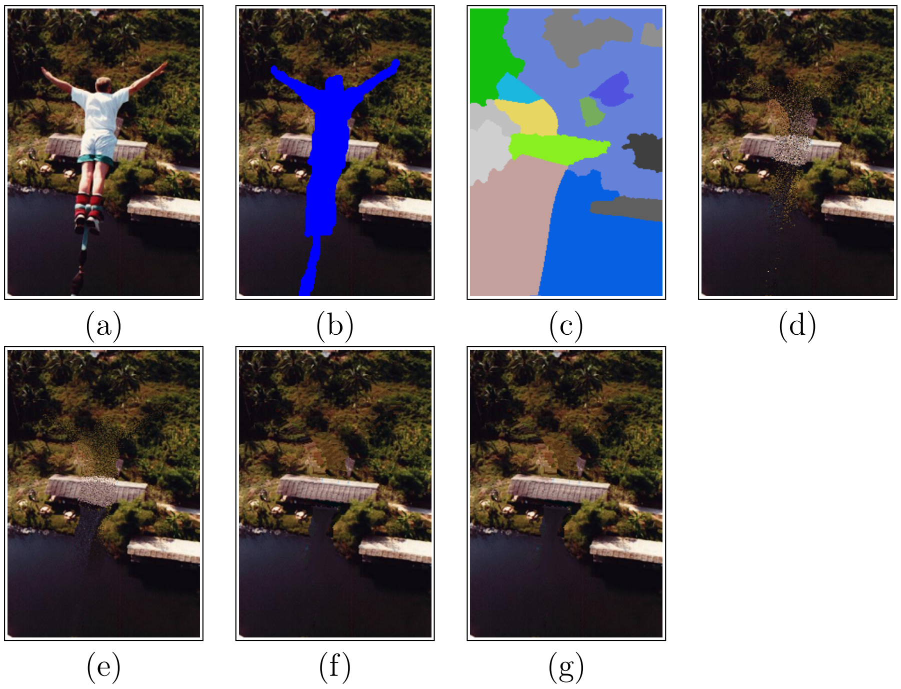


Figure 6.16: Image completion results for the BUNGEE-JUMBER image. (a) original image from [9], (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, (d) initial filling in of the hole area, (e), (f) and (g) first, middle and final iterations of image completion.

therefore the reconstruction of the structure between the vertical and horizontal tubes is created, as shown in Figure 6.22 (g), mainly filled from the horizontal tube area as defined by the structure reconstructed image.

Also, the texture of the hole is propagated well, having clear tubes and tree structures and textures that match the surrounding corresponding textures and structures. Note that the vertical tube, horizontal tube, grass and tree are homogeneous regions surrounding the hole which results in plausible initial filling in for the hole as shown in the initial filling-in of the areas in Figure 6.22 (d). Thus, this leads to plausible completion of the image as shown in Figure 6.22 (g) where each region of the hole has its structure and texture plausibly created. Figures 6.21, 6.23 show zoom in of the original image, the initial filling in and the final synthesised image for the TUBE image and its improved results respectively.

Figure 6.24 (a) and (b) shows a large area to be removed from the ELEPHANT image and then replaced by proper structure and texture. The structure of the hole generally looks plausible, having created defined region boundaries in the hole. The texture is also created well, particularly in the sand, river and mountain areas. The texture of the tree region is reasonably propagated inside the hole. However, the top middle area includes areas from the mountain, which is due to the “missed” structure of that region in the reconstructed structure. The texture of the shore

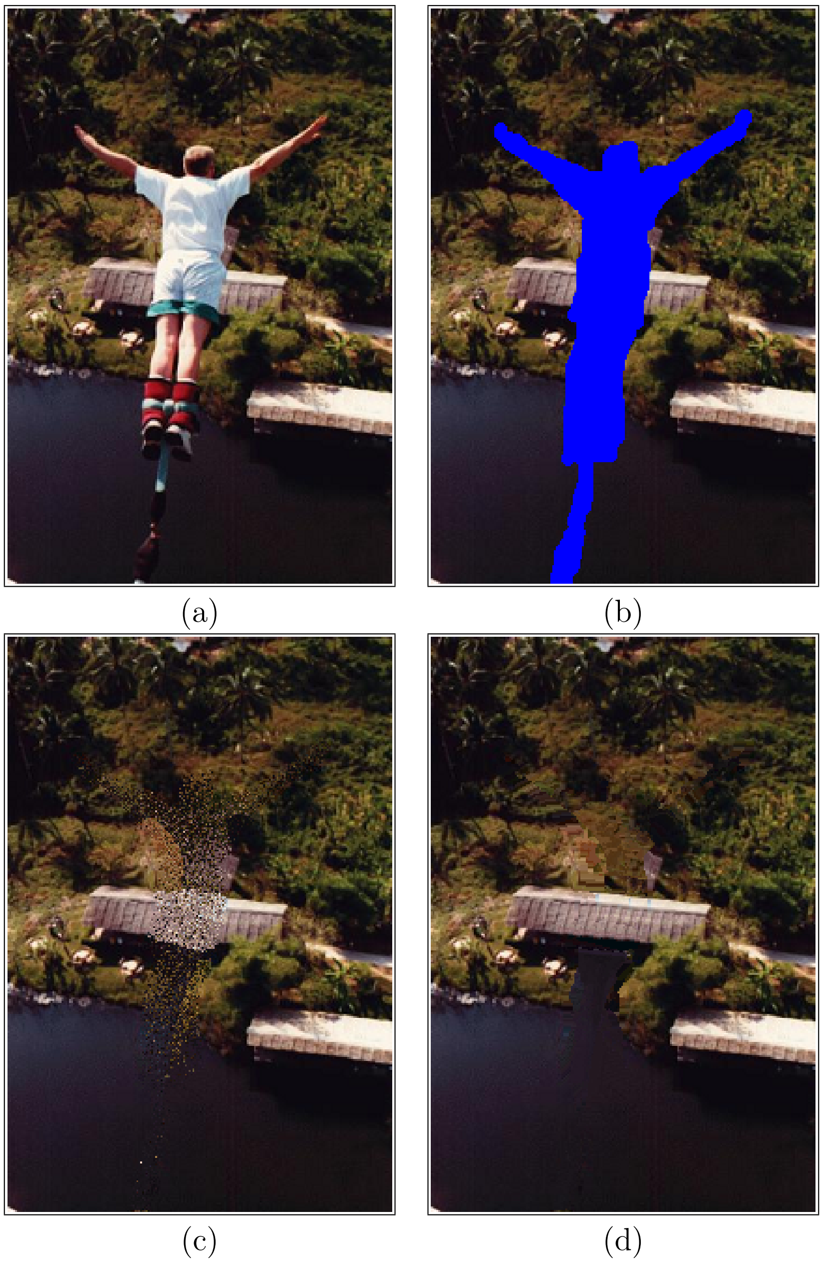


Figure 6.17: Zoom in results for the BUNGEE-JUMBER image. (a) original image from [9], (c) original image with the hole, (c) initial filling in of the hole area, and (d) final result

area is also reasonably propagated. However, repetition of some parts of the shore are noticed, specifically in the areas between the legs of the elephant. This is because there is no sufficient examples of the shore areas, and this is due to the existence of a small shore area outside the hole between the left legs of the elephant, where pixels are copied from. Figure 6.25 shows a zoom in of the original image, the initial filling in and the final synthesised image.

Figure 6.26 shows a comparison with the result of the method presented by Drori et al. in [35]. Our method shows comparable results to their method in term of

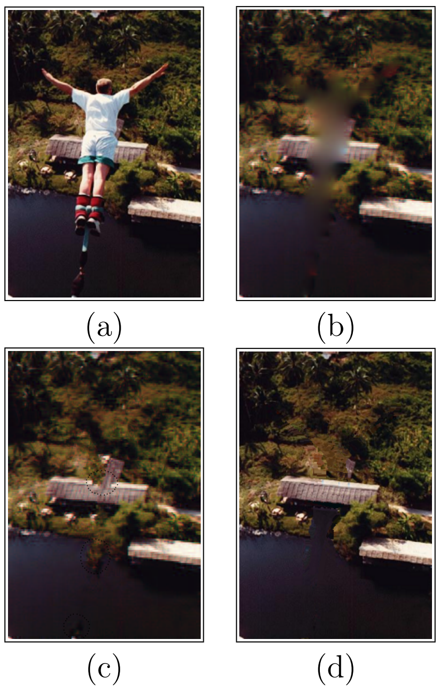
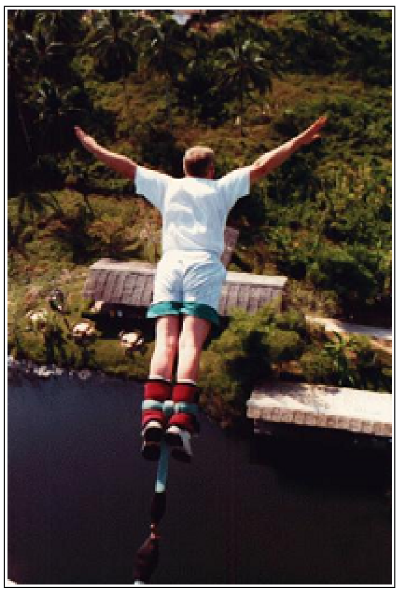


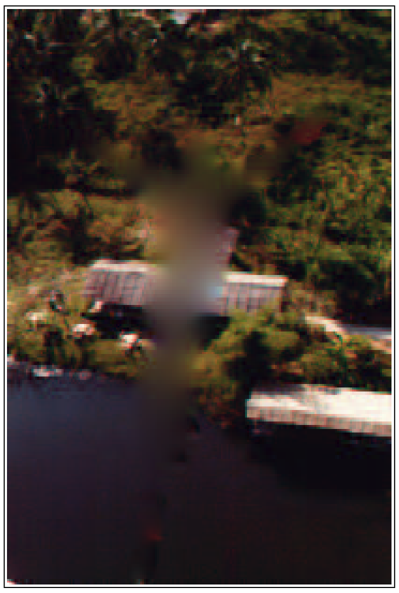
Figure 6.18: Comparing our image completion result for the BUNGEE-JUMBER image. (a) original image from [9], (b) the result of applying image inpainting in [9], (c) the result of applying the method in [30], and (d) our result.

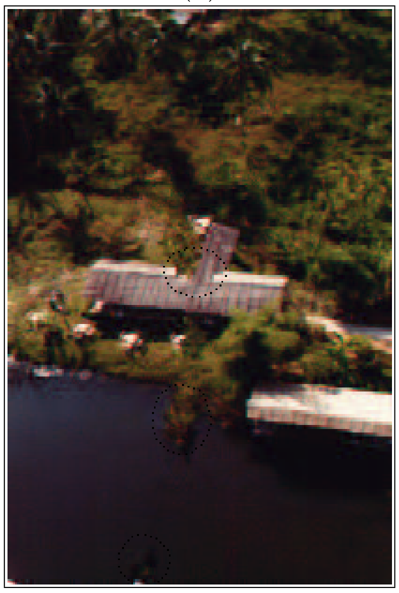
creating plausible structure of boundaries for regions in the hole. Also, the structure is well propagated for most regions, however, the method in [35] outperforms our method in creating proper texture of the tree in the middle of the hole. This is because their method is patch-based texture synthesis which can capture texture structure better than the pixel-based texture synthesis methods. Figure 6.27 shows zoom in of the comparison results.

6.4 Conclusion

In summary, the chapter combines the methods of structure reconstruction and texture synthesis in order to produce realistic results. The combination is performed by firstly reconstructing a structure (Chapter 5) and then using this structure as a constraint for the texture synthesis method (Chapter 4). This means that the texture synthesis method is guided by the reconstructed structure which will propagate texture from relevant regions and preserve the global structure of the image.


(a)


(b)


(c)

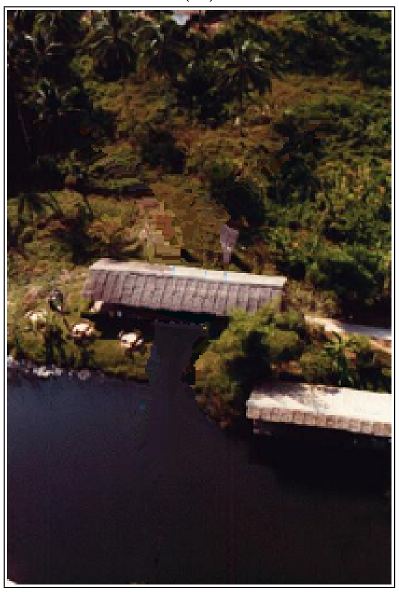

(d)

Figure 6.19: Zoom in results for comparing our image completion result to others for the BUNGEE-JUMBER image. (a) original image from [9], (b) the result of applying image inpainting in [9], (c) the result of applying the method in [30], and (d) our result.

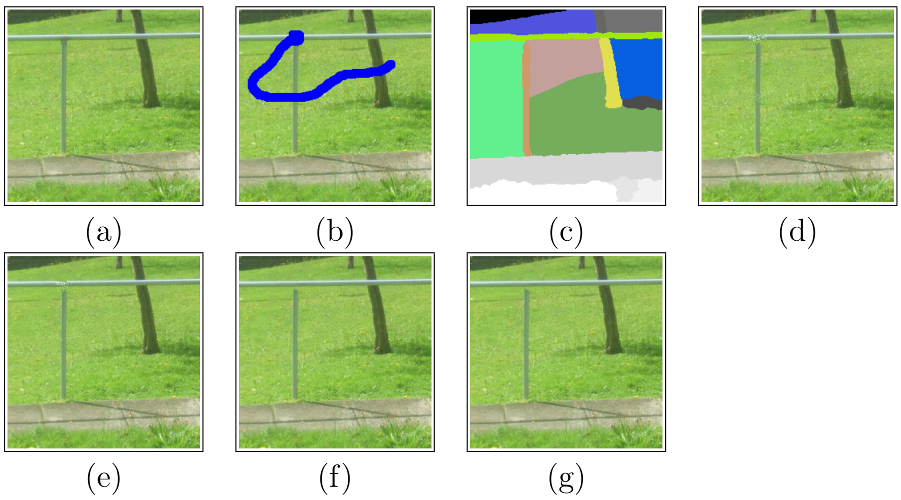


Figure 6.20: Image completion results for the TUBE image. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, (d) initial filling in of the hole area, (e), (f) and (g) first, middle and final iterations of image completion.

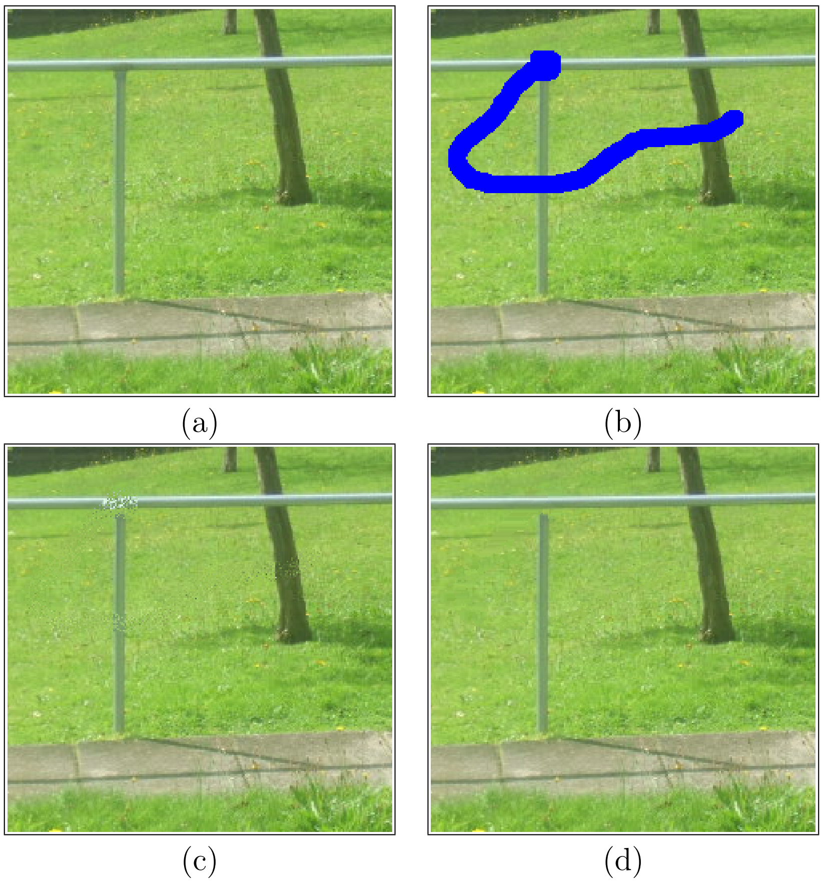


Figure 6.21: Zoom in results for the TUBE image. (a) original image, (b) original image with a hole, (c) initial filling in of the hole area, and (d) final result.

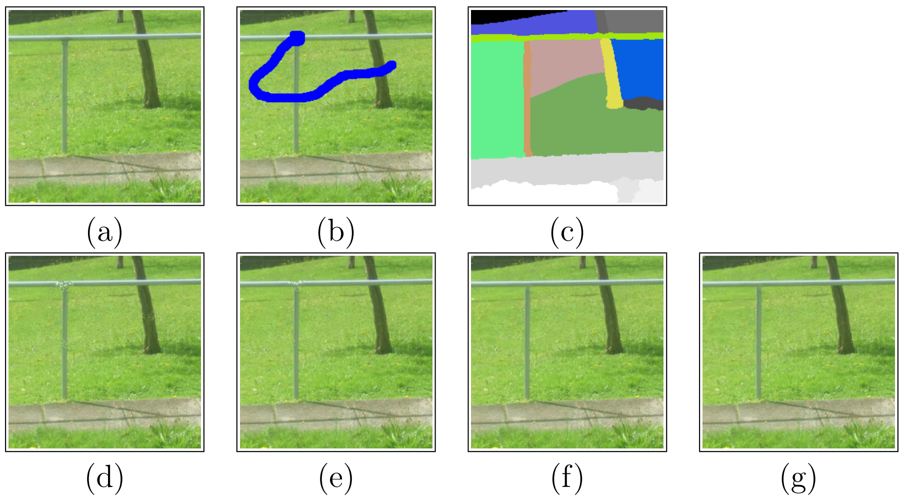


Figure 6.22: Image completion results for the improved results of the TUBE image. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process using a manual user interaction connecting and flood-filling the region between the vertical and horizontal tube, (d) initial filling in of the hole area, (e), (f) and (g) first, middle and final iterations of image completion.

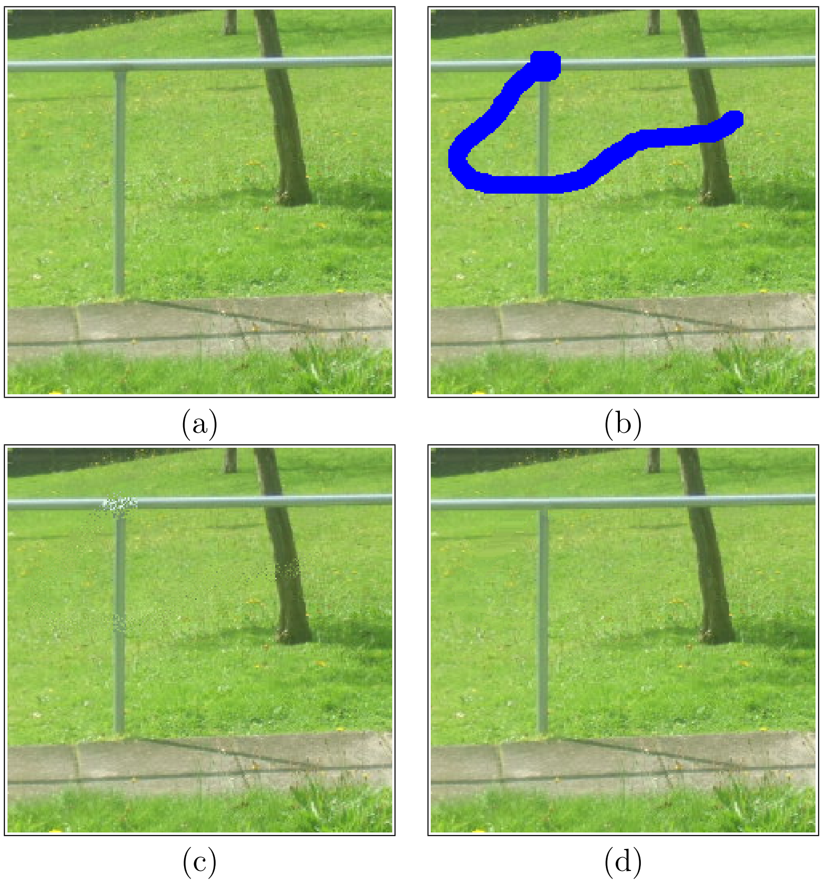


Figure 6.23: Zoom in for the improved results of the TUBE image. (a) original image, (b) original image with a hole, (c) initial filling in of the hole area, and (d) final result.

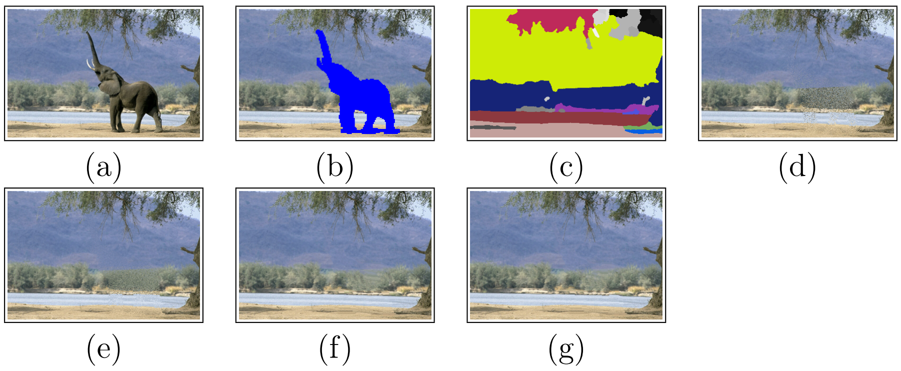


Figure 6.24: Image completion results for the ELEPHANT image. (a) original image from c, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, (d) initial filling in of the hole area, (e), (f) and (g) first, middle and final iterations of image completion.

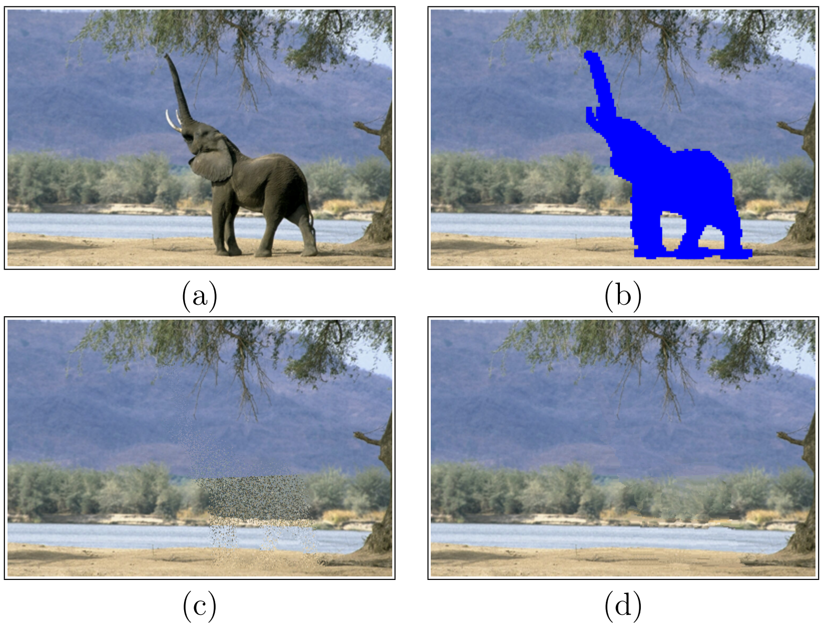


Figure 6.25: Zoom in results for the ELEPHANT image. a) original image, (b) original image with a hole, (c) initial filling in of the hole area, and (d) final result.

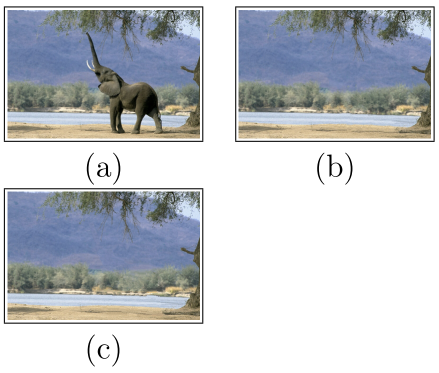
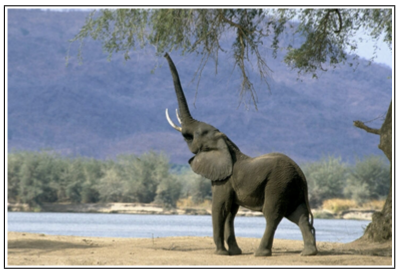
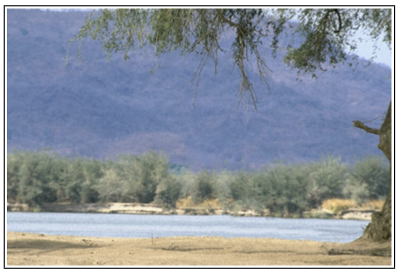


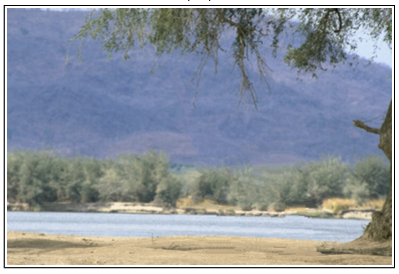
Figure 6.26: Comparing our image completion result for the ELEPHANT image: (a) original image from [35], (b) the result of applying the method in [35], and (c) our result.



(a)



(b)



(c)

Figure 6.27: Zoom in results for comparing our image completion result to others for the ELEPHANT image. (a) original image from [35], (b) the result of applying the method in [35], and (c) our result.

Chapter 7

Conclusion and future work

7.1 Conclusion

In this thesis, we presented a new method that automates the completion of a given hole left by the removal of an undesired area in an image. the aim was to automatically, without high-level knowledge about the content of the image, create a completion that is plausible, i.e. not showing any visible visual artifacts and such that the created structure and texture seamlessly fuses with the surrounding areas. The method first creates image structure that is then used as a constraint to guide the texture synthesis method.

The aim of reconstructing image structure in the hole is to create regions in the hole with well defined boundaries such that the structure of these regions is globally consistent with the structure of the surrounding areas. The method is based on the assumption that regions of images, particularly in natural images, tend to have spatial continuity that are only broken by the presence of the hole, and therefore must be linked. In order to link them, the image is segmented and its similar regions that are attached to the hole are relabelled with the same label and then connected and flood-filled. The similarity measure is based on histogram statistics and proximity of regions.

The method of texture synthesis creates texture inside the hole that is constrained by the reconstructed structure image, and this texture will be coherent globally and matches the surrounding texture. Two modifications to the generic method of texture synthesis have been introduced and this includes the parallel synthesis order and the iterative synthesis scheme. As the order of the synthesis can affect the quality of the output image, a parallel synthesis order is presented which synthesises each pixel in the hole independently from its previously synthesised pixels. This will ensure that pixels being synthesised are not dependant on previous pixels during any given iteration, and therefore will remove any directional bias to specific regions. Also, it allows us to have full symmetric neighbourhood (with the help of the initial hole filling) for the neighbourhood matching process. This symmetric neighbourhood contains more available, plausible, values compared to the L-shaped neighbourhood. This provides a better mechanism of selecting matches. The other modification is the iterative synthesis scheme which allows to have global randomness which will progressively converge towards fine detailed texture. This scheme ensures that the created texture has sufficient, but not excessive, randomness and does not have replications of entire patches. This is achieved by iteratively filling the hole (using the parallel method), allowing a wide set of good matches, and then reducing the size of the set at each iteration.

Our image completion method combines both structure reconstruction and texture synthesis methods by using the structure reconstructed image as a constraint for the texture synthesis method. This results in producing consistent image structure in the hole as well as creating texture from relevant regions, thus producing realistic completion.

Although our method produces good results for many variety of images (Chapter 6), it has its own limitations. First, the success of the method of reconstructing the structure of the hole is based on the quality of the segmentation. Although we used one of the best available segmentation methods, the method may fail to properly segment images (see Figure 6.4). This can result in losing connection of regions, connecting the "incorrect" regions, or simply producing small segmented regions that may cause incompatibility in the reconstruction of region structures, especially when the hole is large. This will also directly affect the texture synthesis quality as the texture synthesis method uses the reconstructed image as a constraint and this image does not sufficiently represent the proper image structure of the hole.

Another limitation is that our method does not have high level knowledge of what the real image structure and texture in the hole should be (e.g. the case in Figure 6.20) and therefore, some user interaction might be necessary in such cases to modify the existing structure such that it matches the surrounding structure.

Also, using lines when connecting similar regions in the structure reconstruction stage may not necessarily match the structure of the surrounding areas. Although these line structures are modified by synthesising the segmented image, it may not properly match the patterns of these structures, as it might be difficult to find good matches for the these line structures due to lack of similar examples in the rest of the image.

Finally, our method was not intended to be designed for real time completion. Improving the quality of the completed image was of high priority, possibly with some sacrifice in efficiency. Although, the method is slow due to pixel-based processing and iterative synthesis, it is inherently parallel and could therefore be implemented on a parallel machine.

7.2 Future work

In the future, we intend to investigate the followings:

- We plan to introduce an adaptive neighbourhood size that can properly capture texture elements. As the hole can be surrounded by many different types of textures, determining the "correct" neighbourhood size for each region separately will create a more representative texture of the specified region. This can be based on computing statistics of the texture regions such as homogeneity of these regions and the arrangements of their elements.
- We intend to investigate further the initial hole filling process to improve its accuracy by using texture orientation.
- We also plan to extend our approach of image relabelling to be based not only on colour and spatial proximity of regions, but also on the size and shape of these regions to ensure more accurate similarity measures between image regions. This will result in more accurate reconstruction of image structure which will improve the quality of the synthesised texture.
- Instead of using lines, when connecting similar regions in the hole, and matching that structure with the surroundings, it could be better to capture the structure of the region boundary (lines/curves) around the hole by approximating their pattern and then propagating that structure inside the hole. However, the approximation can be difficult in cases where the structure of the boundary of a region is completely different on both sides of the hole. Therefore, synthesising the approximated regions will create more plausible structure that matches the structure of the surrounding areas.
- We plan to incorporate the gradient when computing the Euclidean distance in order to improve the image similarity measure.

Bibliography

[1] Scott Acton, Dipti Mukherjee, Joebob Havlicek, and Alan Bovik. Oriented texture completion by AM-FM reaction-diffusion. *IEEE Transactions on Image Processing*, 10(6):885–896, 2001.

[2] Aseem Agarwala, Mira Dontcheva, Maneesh Agrawala, Steven Drucker, Alex Colburn, Brian Curless, David Salesin, and Michael Cohen. Interactive digital photomontage. *ACM Transactions on Graphics*, 23(3):294–302, 2004.

[3] Michael Ashikhmin. Synthesizing natural textures. In *Proceedings of the 2001 symposium on Interactive 3D graphics*, pages 217–226, New York, NY, USA, 2001. ACM Press.

[4] Bara'a Attea and Laylan Rashid. A gentic algorithm for texture synthesis and transfer. In *Proceedings of the 4th International Workshop on Texture analysis and synthesis*, pages 59–64, 2005.

[5] Coloma Ballester, Vicent Caselles, and Joan Verdera. A variational model for disocclusion. In *Proceedings of the International Conference on Image Processing (ICIP 03)*, pages 677–680, 2003.

[6] Coloma Ballester, Vicent Caselles, Joan Verdera, Marcelo Bertalmío, and G Sapiro. A variational model for filling-in gray level and color images. *Proceedings of the Eighth IEEE International Conference on Computer Vision (ICCV 01)*, 01:10, 2001.

[7] Celia Zorzo Barcelos, Marcos Batista, Adriana Martins, and Antonio Nogueira. Level lines continuation based digital inpainting. *Computer Graphics and Image Processing, XVII Brazilian Symposium on (SIBGRAPI 04)*, 00:50–57, 2004.

[8] William Barrett and Alan Cheney. Object-based image editing. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques (SIGGRAPH 02)*, pages 777–784, New York, NY, USA, 2002. ACM.

[9] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proceedings of the 27th annual conference on Computer Graphics and Interactive Techniques (SIGGRAPH 00)*, pages 417–424, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[10] Marcelo Bertalmío, Luminita Vese, Guillermo Sapiro, and Stanley Osher. Image filling-in in a decomposition space. In *Proceedings of the International Conference on Image processing (ICIP 03)*, pages 853–855, 2003.

[11] Marcelo Bertalmio, Luminita Vese, Guillermo Sapiro, and Stanley Osher. Simultaneous structure and texture image inpainting. In *Proceedings of the Conference on Computer Vision and Pattern Regonition*, volume 2, page 707. IEEE Computer Society, 2003.

[12] Marcelo Bertalmío, A Bertozzi, and Guillermo Sapiro. Navier-stokes, fluid dynamics, and image and video inpainting. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 01)*, 1:355, 2001.

[13] Didier Besset. *Object-Oriented Implementaion of Numerical Methods an Introduction with Java and Smalltalk.* Morgan Kaufmann Publishers, San Francisco, USA, 2001.

[14] Dorothea Blostein and Narendra Ahuja. Shape from texture: integrating texture-element extraction and surface estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12):1233–1251, Dec 1989.

[15] Jeremy De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques (SIGGRAPH 97)*, pages 361–368, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[16] Siddharth Borikar, K Biswas, and Sumanta Pattanaik. Fast algorithm for completion of images with natural scenes. Technical report, University of Central Florida, 2004.

[17] Raphaël Bornard, Emmanuelle Lecan, Louis Laborelli, and Jean-Hugues Chenot. Missing data correction in still images and image sequences. In *Proceedings of the tenth ACM international conference on Multimedia (MULTIMEDIA 02)*, pages 355–361, New York, NY, USA, 2002. ACM.

[18] Toby Breckon. *Completing unknown portions of 3D scenes by 3D visual propagation.* PhD thesis, School of Informatics, University of Edinburgh, 2006.

[19] Stephen Brooks, Marc Cardle, and Neil Dodgson. Enhanced texture editing using self-similarity. In *Proceedings of the conference on Vision, Video and Graphics*, Bath, UK, 2003. Eurographics Association.

[20] Stephen Brooks, Marc Cardle, and Neil Dodgson. Replicated texture editing. Technical report, Faculty of Computer Science, Dalhousie University, Halifax, Canada, 2005.

[21] Stephen Brooks and Neil Dodgson. Self-similarity based texture editing. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques (SIGGRAPH 02)*, pages 653–656, New York, NY, USA, 2002. ACM.

[22] R Cant and C Langensiepen. A multiscale method for automated inpainting. In *Proceedings of the 17th European Simulation Multiconference*, Nottingham Trent University, 2003.

[23] Tony Chan and Sung Kang. Error analysis for image inpainting. *Journal of Mathematical Imaging and Vision*, 26(1-2):85–103, 2006.

[24] Tony Chan and Jianhong Shen. Non-texture inpainting by curvature-driven diffusions (CDD). Technical report, Department of Mathematics, University of California, Los Angeles, September 2000.

[25] Tony Chan and Jianhong Shen. Mathematical models for local nontexture inpaintings. *Journal of Applied Mathematics (SIAM 02)*, 62(3):1019–1043, 2002.

[26] Tony Chan, Jianhong Shen, and Luminita Vese. Variational pde models in image processing. *Notices of the American Mathematical Society (AMS)*, 50(1), 2003.

[27] Chen Chaur-Chin and Chen Chien-Chang. Texture synthesis: A review and experiements. *Journal of Information and Science and Engineering*, 19(2):371–380, 2003.

[28] Qiang Chen, Yingxiang Zhang, and Yuncai Liu. Image inpainting with improved exemplar-based approach. In *Multimedia Content Analysis and Mining, International Workshop (MCAM 2007)*, pages 242–251, 2007.

[29] Fernand Cohen and David Cooper. Simple parallel hierarchical and relaxation algorithms for segmenting noncausal markovian random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(2):195–219, March 1987.

[30] Antonio Criminisi, Patrick Pérez, and Kentaro Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on Image Processing*, 13(9):1200–1212, 2004.

[31] George Cross and Anil Jain. Markov random field texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(1):25–39, January 1983.

[32] Scott Dally. The visible differences predictor: an algorithm for the assessment of image fidelity. In *Digital images and human vision*, pages 179–206, Cambridge, MA, USA, 1993. MIT Press.

[33] Yining Deng and B Manjunath. Unsupervised segmentation of color-texture regions in images and video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(8):800–810, 2001.

[34] Feng Dong and Xujiong Ye. Multiscaled texture synthesis using multisized pixel neighborhoods. *IEEE Computer Graphics and Applications*, 27(3):41–47, 2007.

[35] Iddo Drori, Daniel Cohen-Or, and Hezy Yeshurun. Fragment-based image completion. *ACM Transactions on Graphics*, 22(3):303–312, 2003.

[36] Nick Efford. *Digital Image Processing A practical Introduction using Java.* Pearson Education Limited, Essex, England, 2000.

[37] Alexei Efros and William Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques (SIGGRAPH 01)*, pages 341–346, New York, NY, USA, 2001. ACM Press.

[38] Alexei Efros and Thomas Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the International Conference on Computer Vision*, pages 1033–1038, Washington, DC, USA, 1999. IEEE Computer Society.

[39] Selim Esedoglu and Jianhong Shen. Digital inpainting based on the mumford-shah-euler image model. *European Journal of Applied Mathematics*, 13(04):353–370, 2002.

[40] Ahmeed Eskicioglu and Paul Fisher. Image quality measures and their performance. *IEEE Transactions on Communications*, 43(12):2959–2965, Dec 1995.

[41] Chih-Wei Fang and James Lien. Fast image replacement using multi-resolution approach. In *Proceedings of the 7th Asian Conference on Computer Vision (ACCV 06)*, pages 509–520, 2006.

[42] Hui Fang and John Hart. Textureshop: texture synthesis as a photograph editing tool. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 04)*, pages 354–359, New York, NY, USA, 2004. ACM.

[43] FreeFoto.com. Accessed 10/06/09.

[44] Haoying Fu, Hongyuan Zha, and Jesse Barlow. Efficient block noise removal based on nonlinear manifolds. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV 05)*, pages 549–556, Washington, DC, USA, 2005. IEEE Computer Society.

[45] King Sun Fu. *Syntactic Pattern Recognition and Applications.* Englewood Cliffs; London: Prentice-Hall, 1982.

[46] Claire Gallagher and Anil Kokaram. Wavelet based texture synthesis. In *The Irish Machine Vision and Image Processing (IMVIP 04)*, Dublin, Ireland, 2004.

[47] Claire Gallagher and Anil Kokaram. Nonparametric wavelet based texture synthesis. In *IEEE International Conference on Image Processing (ICIP 05)*, volume 2, pages 462–465, Genova, Italy, September 2005.

[48] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, Nov. 1984.

[49] G Gimel'farb. Texture modeling by multiple pairwise pixel interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(11):1110–1114, 1996.

[50] L Gool, P Dewaele, and A Oosterlinck. Texture analysis anno 1983. *Computer Vision, Graphics, and Image Processing (CVGIP 85)*, 29(3):336–357, March 1985.

[51] Harald Grossauer. A combined pde and texture synthesis approach to inpainting. In *8th European Conference on Computer Vision (ECCV 04)*, pages 214–224, 2004.

[52] Harald Grossauer. *Completion of Images with Missing Data Regions.* PhD thesis, Leopold-Franzens-Universität Innsbruck, 2005.

[53] Robert Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786–804, May 1979.

[54] Paul Harrison. A non-hierarchical procedure for re-synthesis of complex textures. In *Proceedings of the Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG 01)*, pages 190–197, 2001.

[55] Richard Harvey, Stephen King, Richard Aldridge, and J Bangham. Comparing image resamplers via a model of the human vision system. In *Proceedings of the British Machine Vision Conference*, volume 1, pages 162–172, 1997.

[56] James Hays and Alexei Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics (SIGGRAPH 07)*, 26(3), 2007.

[57] David Heeger and James Bergen. Pyramid-based texture analysis/synthesis. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques (SIGGRAPH 95)*, pages 229–238, New York, NY, USA, 1995. ACM Press.

[58] Aaron Hertzmann, Charles Jacobs, Nuria Oliver, Brian Curless, and David Salesin. Image analogies. In Eugene Fiume, editor, *Proceedings of the annual conference on Computer Graphics and Interactive Techniques (SIGGRAPH 01)*, pages 327–340. ACM Press / ACM SIGGRAPH, 2001.

[59] Anil Hirani and Takashi Totsuka. Combining frequency and spatial domain information for fast interactive image noise removal. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques (SIGGRAPH 96)*, pages 269–276, New York, NY, USA, 1996. ACM.

[60] Zhang Hongying, Peng Qicong, and Wu Yadong. Image completion algorithm based on texture synthesis. *Journal of Systems Engineering and Electronics*, 18(2):385–391, 2007.

[61] Homan Igehy and Lucas Pereira. Image replacement through texture synthesis. In *Proceedings of the 1997 International Conference on Image Processing (ICIP 97)*, page 186, Washington, DC, USA, 1997. IEEE Computer Society.

[62] Siddharth Jain. Image inpainting and texture synthesis: Two methods for hole filling in images. Technical report, Berkeley University of California, 2003.

[63] Jiaya Jia, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. Drag-and-drop pasting. *ACM Transactions on Graphics*, 25(3):631–637, 2006.

[64] Jiaya Jia and Chi-Keung Tang. Image repairing: Robust image synthesis by adaptive nd tensor voting. *Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 03)*, 01:643, 2003.

[65] Jiaya Jia and Chi-Keung Tang. Inference of segmented color and texture description by tensor voting. *IEEE Transactions on Pattern Analalysis and Machine Intelligence*, 26(6):771–786, 2004.

[66] M Johnson, G Brostow, J Shotton, O Arandjelovic, V Kwatra, and R Cipolla. Semantic photo synthesis. *Computer Graphics Forum*, 25(3):407–414, September 2006.

[67] K Karu, A Jain, and R Bolle. Is there any texture in the image? In *Proceedings of the 13th International Conference on Pattern Recognition (ICPR 96)*, page 770, Washington, DC, USA, 1996. IEEE Computer Society.

[68] Erum Khan, Erik Reinhard, Roland Fleming, and Heinrich Bülthoff. Image-based material editing. *ACM Transactions on Graphics*, 25(3):654–663, 2006.

[69] Anil Kokaram. Parametric texture synthesis for filling holes in pictures. In *IEEE International Conference on Image Processing 2002*, September 2002.

[70] Vivek Kwatra, Irfan Essa, Aaron Bobick, and Nipun Kwatra. Texture optimization for example-based synthesis. *ACM Transactions on Graphics*, 24(3):795–802, 2005.

[71] Vivek Kwatra, Arno Schödl, Irfan Essa, Greg Turk, and Aaron Bobick. Graphcut textures: image and video synthesis using graph cuts. *ACM Transactions on Graphics*, 22(3):277–286, 2003.

[72] Frédéric Labrosse. On the editing of images: Selecting, cutting and filling-in. In *Proceedings of Vision, Video, and Graphics (VVG 03)*, pages 71–78, 2003.

[73] Laurent Lefebvre and Pierre Poulin. Analysis and synthesis of structural textures. In *Proceedings of the Graphics Interface 2000 Conference*, pages

77–86, Montréal, Québec, Canada, 2000. Canadian Human-Computer Communications Society.

[74] Sylvain Lefebvre and Hugues Hoppe. Parallel controllable texture synthesis. *ACM Transactions on Graphics*, 24(3):777–786, 2005.

[75] M Lettner, P Kammerer, and R Sablatnig. Texture analysis of painted strokes. In *28th Workshop of the Austrian Association for Pattern Recognition*, pages 269–276, Hagenberg, Austria, 2004.

[76] Anat Levin, Assaf Zomet, and Yair Weiss. Learning how to inpaint from global image statistics. In *Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 03)*, page 305, Washington, DC, USA, 2003. IEEE Computer Society.

[77] John-Peter Lewis. Texture synthesis for digital painting. *ACM SIGGRAPH Computer Graphics*, 18(3):245–252, 1984.

[78] Lin Liang, Ce Liu, Ying-Qing Xu, Baining Guo, and Heung-Yeung Shum. Real-time texture synthesis by patch-based sampling. *ACM Transactions on Graphics*, 20(3):127–150, 2001.

[79] Yi Lin. Fast image completion. In Heung-Yeung Shum Andrew Tescher Shipeng Li, Fernando Pereira, editor, *Proceedings of Visual Communications and Image Processing*, volume 5960, 2005.

[80] Yanxi Liu, Wen-Chieh Lin, and James Hays. Near-regular texture analysis and manipulation. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH 04)*, pages 368–376, New York, NY, USA, 2004. ACM.

[81] Jeremy Long and David Mould. Improved image quilting. In *Proceedings of Graphics Interface (GI 07)*, pages 257–264, New York, NY, USA, 2007. ACM.

[82] Simon Masnou. Disocclusion: a variational approach using level lines. *IEEE Transactions on Image Processing*, 11(2):68–76, 2002.

[83] Andrzej Materka and Michal Strzelecki. Texture analysis methods - a review. Technical report, Technical University of Lodz, Institute of Electronics, Brussels, 1998.

[84] Tim Morris. *Computer Vision and Image Processing.* Palgrave Macmillan, 2004.

[85] Andrew Nealen and Marc Alexa. Hybrid texture synthesis. In *Proceedings of the 14th Eurographics workshop on Rendering (EGRW 03)*, pages 97–105, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[86] Andrew Nealen and Marc Alexa. Fast and high quality overlap repair for patch-based texture synthesis. *Computer Graphics International (CGI 04)*, 00:582–585, 2004.

[87] Laszlo Neumann, Kresimir Matkovic, and Werner Purgathofer. Perception based color image difference. Technical Report TR-186-2-97-21, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria, December 1997.

[88] Manuel Oliveira, Brian Bowen, Richard McKenna, and Yu-Sung Chang. Fast digital image inpainting. In *Proceedings of the IASTED International Conference on Visualization, Imaging and Image Processing (VIIP 01)*, pages 261–266, Marbella, Spain, 2001.

[89] Thrasyvoulos Pappas and Robert Safranek. Perceptual criteria for image quality evaluation. In *Handbook of Image and Video Processing*, pages 669–684. Academic Press, 2000.

[90] Patricio Parada and Javier Ruiz.del.Solar. Texture synthesis using image pyramids and self-organizing maps. In *Proceedings of the 11th International Conference on Image Analysis and Processing (ICIAP 01)*, page 244, Washington, DC, USA, 2001. IEEE Computer Society.

[91] Mark Pauly, Niloy Mitra, Joachim Giesen, Markus Gross, and Leonidas Guibas. Example-based 3d scan completion. In *Proceedings of the third Eurographics symposium on Geometry processing (SGP 05)*, page 23, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.

[92] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. *ACM Transactions on Graphics*, 22(3):313–318, 2003.

[93] Patrick Pérez, Michel Gangnet, and Andrew Blake. Patchworks: Example-based region tiling for image editing. Technical report, Microsoft Research, 2004.

[94] Ken Perlin. An image synthesizer. *Proceedings of the 12th annual conference on Computer graphics and interactive techniques (SIGGRAPH 85)*, 19(3):287–296, 1985.

[95] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.

[96] M Pietikäinen and T Ojala. Nonparametric texture analysis with simple spatial operators. In *Proceedings of the 5th International Conference on Quality Control by Artificial Vision*, Canada, 1999.

[97] Javier Portilla and Eero Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40(1):49–70, 2000.

[98] Gonzalez Rafael and Woods Richard. *Digital Image processing.* Tom Robbins, 2 edition, 2001.

[99] Shantanu Rane, Guillermo Sapiro, and Marcelo Bertalmío. Structure and texture filling-in of missing image blocks in wireless transmission and compression applications. *IEEE Transactions on Image Processing*, 12(3):296–303, 2003.

[100] D Rao and L Reddy. Image quality assessment based on perceptual structural similarity. In *Pattern Recognition and Machine Intelligence*, pages 87–94, 2007.

[101] A Rares, M Reinders, and J Biemond. Constrained texture restoration. *Journal on Applied Signal Processing (EURASIP 05)*, 2005(1):2758–2771, 2005.

[102] Todd Reed and J Buf. A review of recent texture segmentation and feature extraction techniques. *Image Understanding (CVGIP 93)*, 57(3):359–372, 1993.

[103] Amine Samet, M Ayed, Nouri Masmoudi, and Lazhar Khriji. New perceptual image quality assessment metric. *Asian Journal of Information Technology*, 4(11):996–1000, April 2005.

[104] Andrei Sharf, Marc Alexa, and Daniel Cohen-Or. Context-based surface completion. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2004)*, pages 878–887, New York, NY, USA, 2004. ACM.

[105] Hamid Sheikh, Alan Bovik, and Lawrence Cormack. No-reference quality assessment using natural scene statistics: Jpeg2000. *IEEE Transactions on Image Processing*, 14(11):1918–1927, November 2005.

[106] Timothy Shih and Rong chi Chang. Super-resolution inpainting. *Journal of Zhejiang University Science*, 6A(6):487–491, 2005.

[107] Seunghyup Shin, Tomoyuki Nishita, and Sung Yong Shin. On pixel-based texture synthesis by non-parametric sampling. *Computers & Graphics*, 30(5):767–778, oct 2006.

[108] Eero Simoncelli and Javier Portilla. Texture characterization via joint statistics of wavelet coefficient magnitudes. In *Proceedings of Fifth International Conference on Image Processing*, pages 4–7. IEEE Computer Society, 1998.

[109] Maneesha Singh and Sameer Singh. Spatial texture analysis: A comparative study. *16th International Conference on Pattern Recognition (ICPR 02)*, 01:10676, 2002.

[110] Jian Sun, Lu Yuan, Jiaya Jia, and Heung-Yeung Shum. Image completion with structure propagation. *ACM Transactions on Graphics*, 24(3):861–868, 2005.

[111] Francesca Taponecco. User-defined texture synthesis. In *Proceedings of the 12th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG 04)*, pages 251–258, 2004.

[112] Zinovi Tauber, Ze-Nian Li, and Mark Drew. Review and preview: Disocclusion by inpainting for image-based rendering. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 37(4):527–540, 2007.

[113] Alexandru Telea. An image inpainting technique based on the fast marching method. *Journal of Graphics Tools*, 9(1):23–34, 2004.

[114] Texturesforrest. Texture Library[`http://textures.forrest.cz/`]. Accessed 22/05/06.

[115] Huang Ting, Shifeng Chen, Jianzhuang Liu, and Xiaoou Tang. Image inpainting by global structure and texture propagation. In *Proceedings of the 15th international conference on Multimedia (MULTIMEDIA 07)*, pages 517–520, New York, NY, USA, 2007. ACM.

[116] Xin Tong, Jingdan Zhang, Ligang Liu, Xi Wang, Baining Guo, and Heung-Yeung Shum. Synthesis of bidirectional texture functions on arbitrary surfaces. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques (SIGGRAPH 02)*, pages 665–672, New York, NY, USA, 2002. ACM.

[117] Mihran Tuceryan and Anil Jain. Texture analysis. In C.H. Chen, L.F. Pau, and P.S.P. Wang, editors, *The Handbook of Pattern Recognition and Computer Vision*, chapter 2.1, pages 207–248. World Scientific Publishing Co, second edition, 1998.

[118] Joan Verdera, Vicent Caselles, Marcelo Bertalmío, and Guillermo Sapiro. Inpainting surface holes. In *IEEE International Conference on Image Processing (ICIP 03)*, pages 903–906, 2003.

[119] Bin Wang, Jun-Hai Yong, and Jia-Guang Sun. Real-time texture synthesis with patch jump maps. *Lecture notes in Computer Science*, (3314):1155–1160, 2004.

[120] Z Wang, C Bovik, R Sheikh, and P Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004.

[121] Zhou Wang, Alan Bovik, and Ligang Lu. Why is image quality assessment so difficult? In *Proceesings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 3313–3316, 2002.

[122] Zhou Wang, Hamid Sheikh, and Alan Bovik. *Objective Video Quality Assessment*, chapter Chapter 41 in The Handbook of Video Databases: Design and Applications, pages 1041–1078. CRC Press, 2003.

[123] Li-Yi Wei. Deterministic texture analysis and synthesis using tree structure vector quantization. In *Proceedings of the XII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 99)*, pages 207–214, Washington, DC, USA, 1999. IEEE Computer Society.

[124] Li-Yi Wei. *Texture Synthesis By Fixed Neighborhood Searching*. PhD thesis, Stanford University, 2001.

[125] Li-Yi Wei and Marc Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th annual conference on Computer Graphics and Interactive Techniques (SIGGRAPH 00)*, pages 479–488, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[126] Li-Yi Wei and Marc Levoy. Order-independent texture synthesis. Technical report, Stanford University Computer Science Department, 2002.

[127] Tomihisa Welsh, Michael Ashikhmin, and Klaus Mueller. Transferring color to greyscale images. *ACM Transactions on Graphics*, 21(3):277–280, 2002.

[128] Yonatan Wexler, Eli Shechtman, and Michal Irani. Space-time completion of video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):463–476, 2007.

[129] Marta Wilczkowiak, Gabriel Brostow, Ben Tordoff, and Roberto Cipolla. Hole filling through photomontage. In *Proceedings of the British Machine Vision Conference, Oxford, United Kingdom*, pages 492–501, July 2005.

[130] Qing Wu and Yizhou Yu. Feature matching and deformation for texture synthesis. *ACM Transactions on Graphics*, 23(3):364–367, 2004.

[131] Ying-Qing Xu, Baining Guo, and Harry Shum. Chaos mosaic: Fast and memory efficient texture synthesis. Technical report, Microsoft Research, 2000.

[132] Hitoshi Yamauchi, Jörg Haber, and Hans-Peter Seidel. Image restoration using multiresolution texture synthesis and image inpainting. *Computer Graphics International (CGI 03)*, 00:120, 2003.

[133] Alexey Zalesny, Vittorio Ferrari, Geert Caenen, and Luc Gool. Parallel composite texture synthesis. In *Texture 2002 Workshop in conjunction with ECCV 2002*, pages 151–155, 2002.

[134] Alexey Zalesny and Luc Gool. A compact model for viewpoint dependent texture synthesis. In *Revised Papers from Second European Workshop on 3D Structure from Multiple Images of Large-Scale Environments (SMILE 00)*, pages 124–143, London, UK, 2001. Springer-Verlag.

[135] Steve Zelinka and Michael Garland. Towards real-time texture synthesis with the jump map. In *Proceedings of the 13th Eurographics workshop on Rendering (EGRW 02)*, pages 99–104, Aire-la-Ville, Switzerland, 2002. Eurographics Association.

[136] Steve Zelinka and Michael Garland. Jump map-based interactive texture synthesis. *ACM Transactions on Graphics*, 23(4):930–962, 2004.

[137] Guangtao Zhai, Wenjun Zhang, Xiaokang Yang, and Yi Xu. Image quality assessment metrics based on multi-scale edge presentation. *Signal Processing Systems Design and Implementation*, pages 331–336, November 2005.

[138] Yunjun Zhang, Jiangjian Xiao, and Mubarak Shah. Region completion in a single image. In *Proceedings of European Association for Computer Graphics (EUROGRAPHICS 04)*, volume 23, 2004.

Appendix A

Additional results

The method of image completion has also been applied to additional textured images, taken from [43]. In general, the images show acceptable hole completion. However, there are failures with some of the images. Note that small neighbourhoods (11×11) were used which can cause problems related to insufficient representation of the texture and the structure of the image. Figures A.1–A.5. show these results.

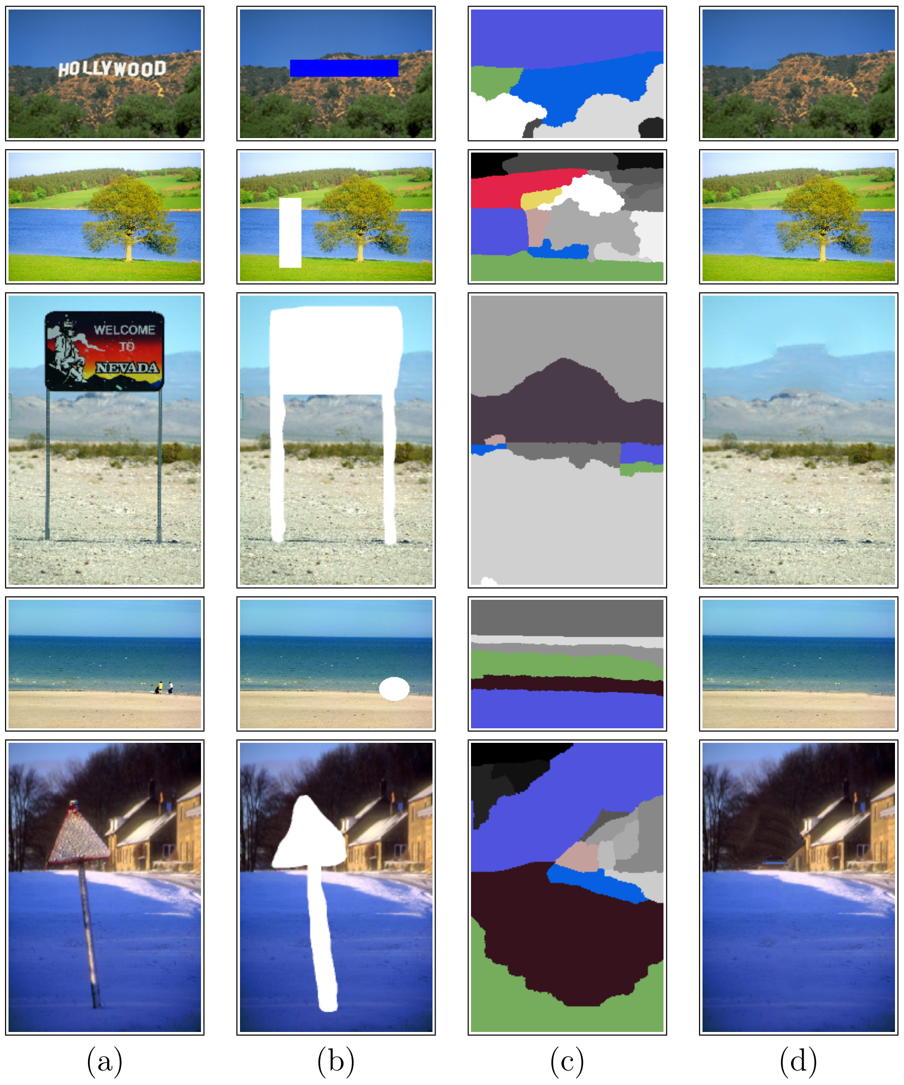


Figure A.1: Image completion results. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, and (d) final result of image completion.

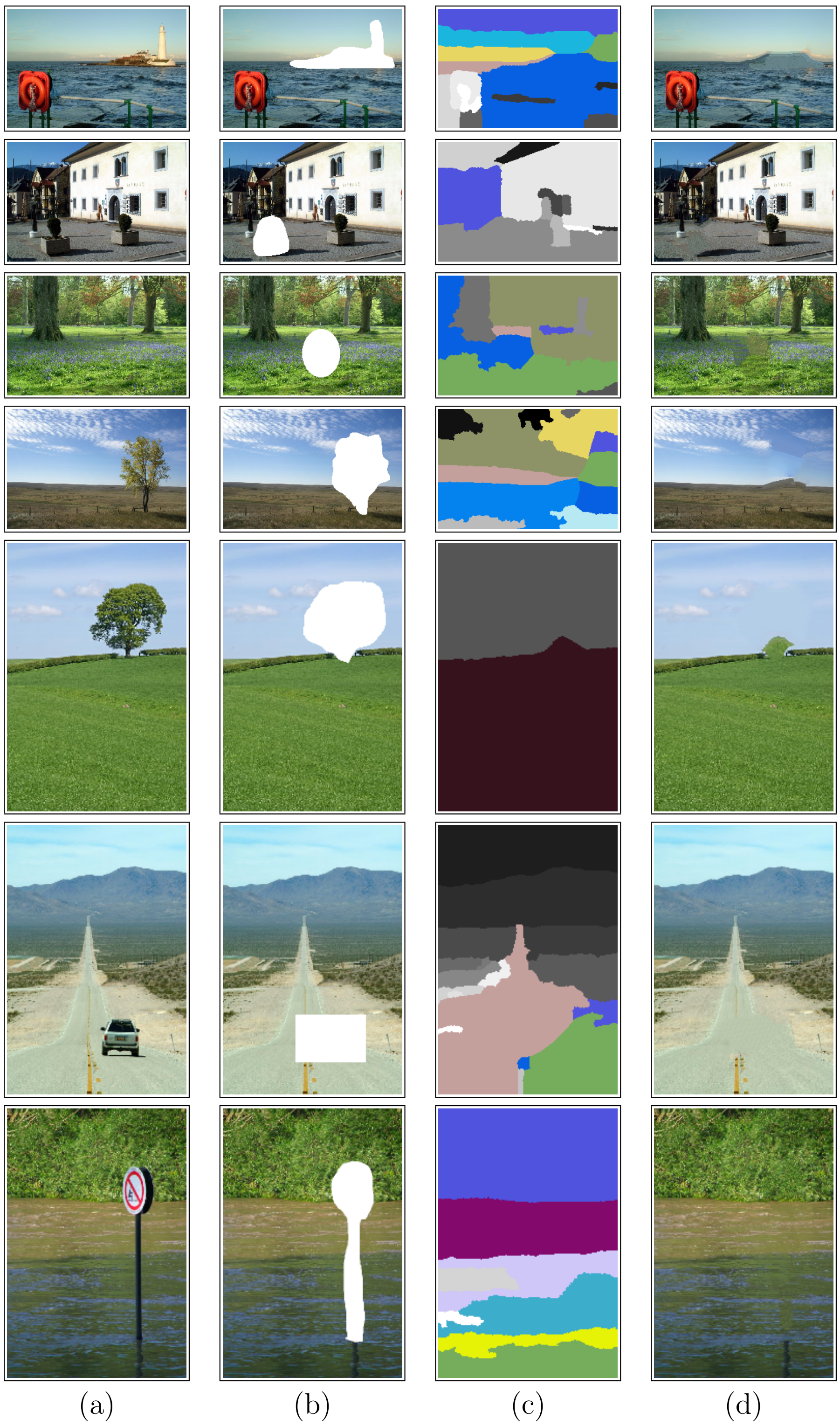


Figure A.2: Image completion results. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, and (d) final result of image completion.

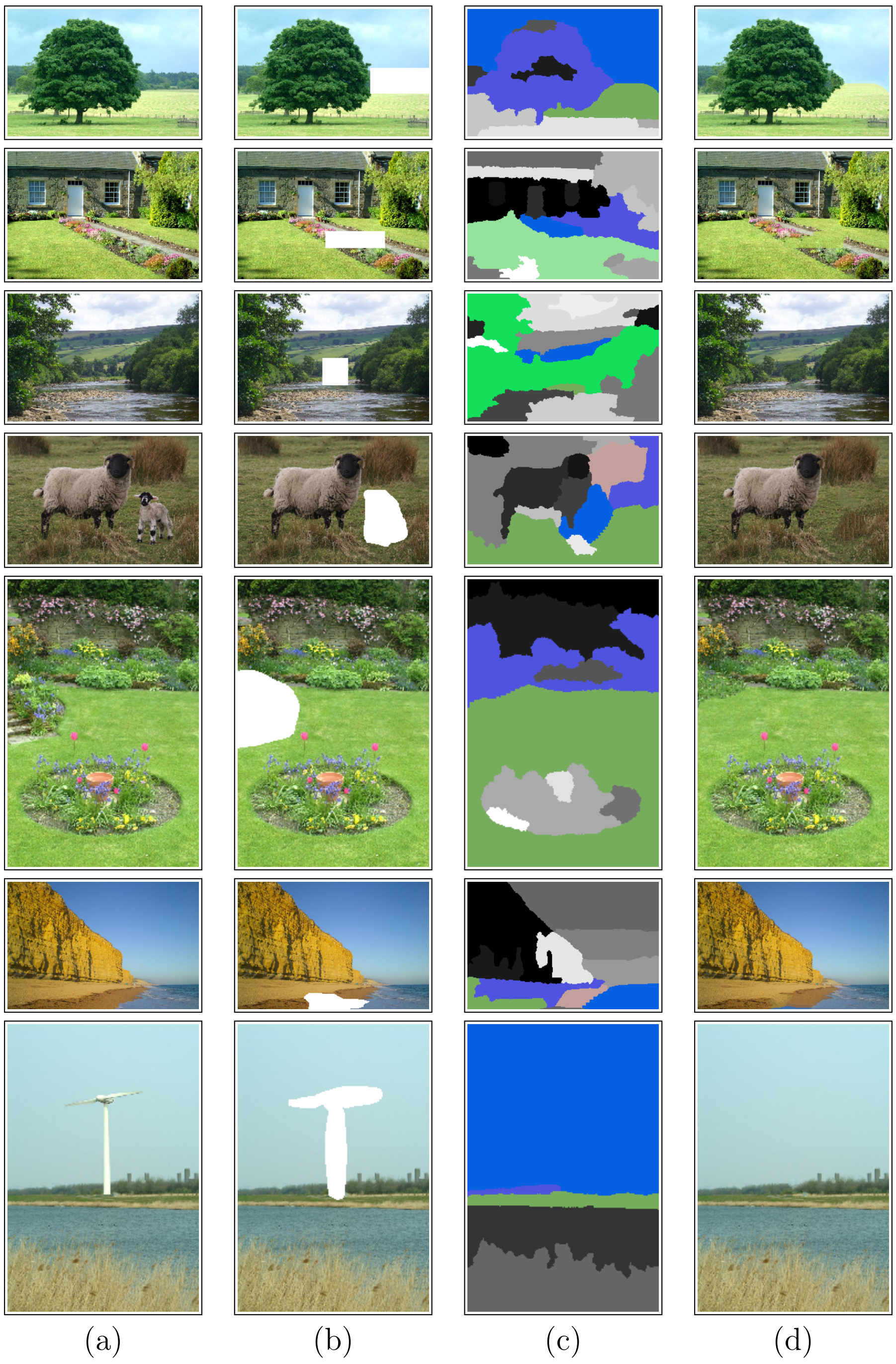


Figure A.3: Image completion results. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, and (d) final result of image completion.

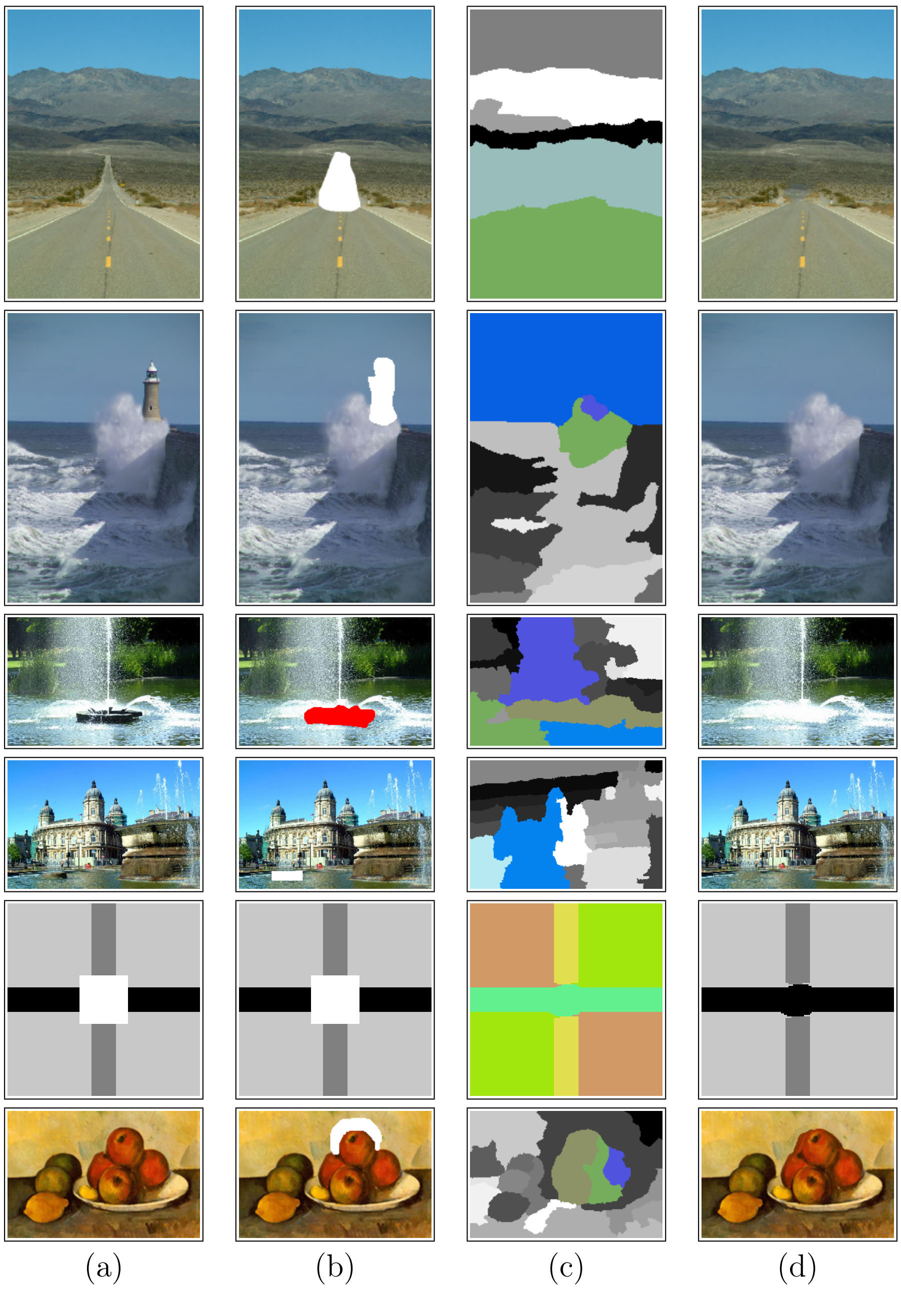


Figure A.4: Image completion results. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, and (d) final result of image completion.

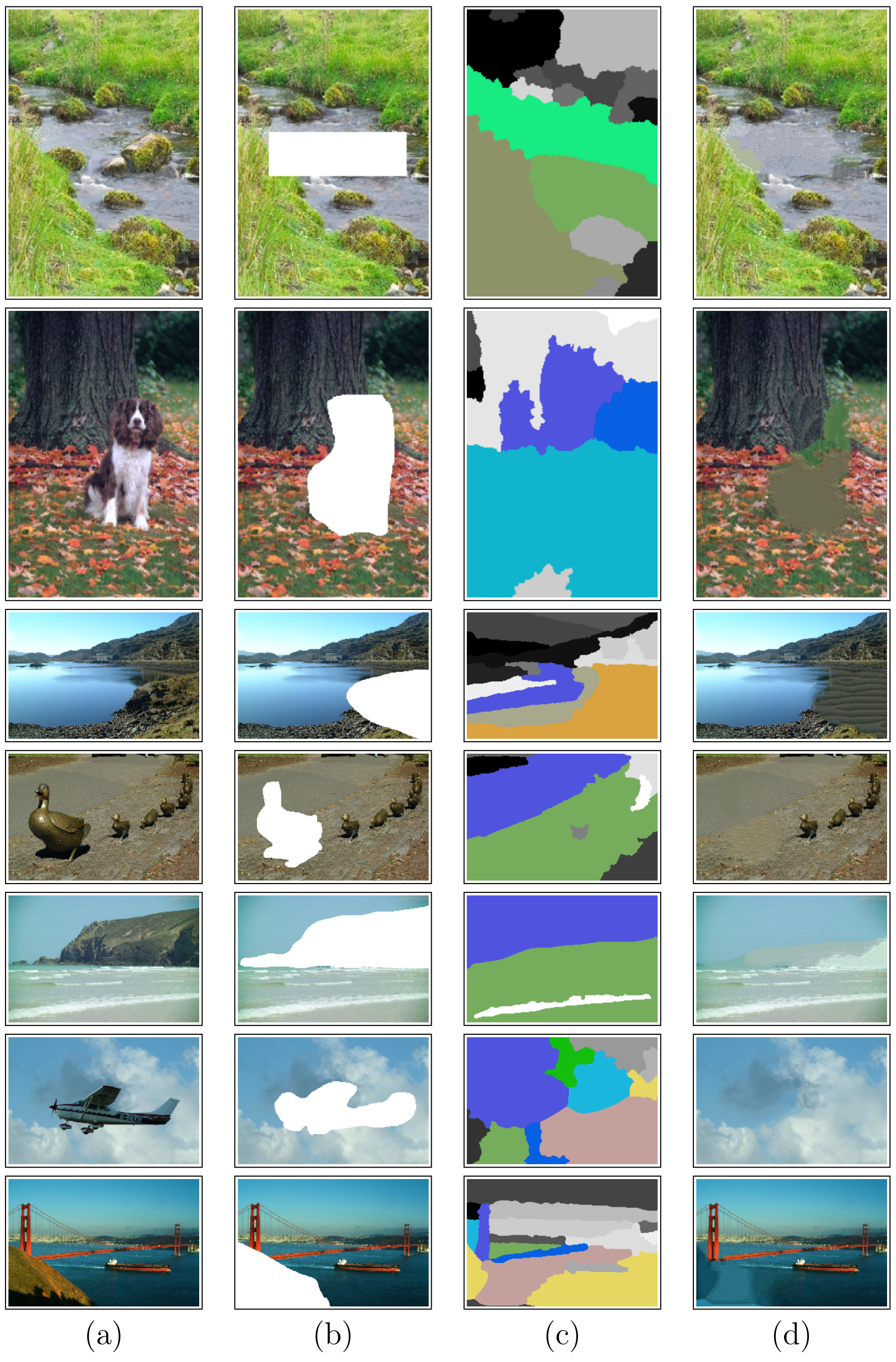


Figure A.5: Image completion results. (a) original image, (b) original image with a hole masked, (c) final image of the hole structure reconstruction process, and (d) final result of image completion.