MAJOR PROJECT

On

Inpainting - Techniques and Applications

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE AWARD OF DEGREE

Of

MASTER OF ENGINEERING

Submitted by:

SUPRIYA CHHABRA
University Roll No 13904

Under the Guidance of:

Dr. S.K SAXENA

DEPARTMENT OF COMPUTER ENGINEERING
DELHI COLLEGE OF ENGINEERING
(Now Delhi Technical University)
BAWANA ROAD, DELHI-110042
DELHI UNIVERSITY

Certificate

This is to certify that the major project entitled **"Inpainting - Techniques and Applications"** is the work of **Supriya Chhabra (**Univ. Roll No. 13904), a student of Delhi College of Engineering. This work was completed under my direct supervision and guidance and forms a part of Master of Engineering (Computer Technology & Applications) course and curriculum.

(Dr. S. K. SAXENA)
Project Guide
Department of Computer Engineering
Delhi College of Engineering
(Now Delhi Technological University)

Acknowledgement

It gives me a great pleasure to express my profound gratitude to my project guide **Dr. S. K. SAXENA,** Senior Faculty, Department of Computer Engineering, Delhi College of Engineering, for his valuable and inspiring guidance throughout the progress of this project.

At the same time, I would like to extend my heart felt thanks to **Dr.(Mrs.) Daya Gupta,** Head of the Department, Department of Computer Engineering, Delhi College of Engineering, for keeping the spirits high and clearing the visions to work on the project.

Supriya Chhabra
Roll No. 13904
(21/ME/CTA/PT/09)

ABSTRACT

The project entitled “INPAINTING - TECHNIQUES AND APPLICATIONS” deals with the study of inpainting, need of inpainting and different techniques to perform inpainting. Image inpainting is the process of seamlessly filling in holes in an image so as to preserve its overall continuity. Applications of this technique include the restoration of old photographs and damaged film; removal of superimposed text like dates, subtitles, or publicity; and the removal of entire objects from the image like microphones or wires in special effects. Digital inpainting is a relatively young research area in computer graphics, a large variety of techniques have been proposed since the concept was first introduced. In this paper, we analyze different digital inpainting algorithms for still images. The simultaneous propagation of texture and structure information achieved. The texture image repaired by the exemplar –based method; for the structure image, the Laplacian operator is used to enhance the structure information. The Laplacian image is inpainted by the exemplar-based algorithm and the Poisson equation based reconstruction is applied thereafter. In 8 pixel neighborhood method, central pixel value is identified by investigating surrounded 8 neighborhood pixel properties like color variation, repetition, intensity and direction. Finally, in edge-based inpainting technique, original image analyzed at encoder side so that some blocks removed during encoding. At decoder side, the image is restored by edgee-based inpainting and texture synthesis. A patch-based image compression framework is introduced which shows the inpainting method can be integrated into a patch-based image compression framework.

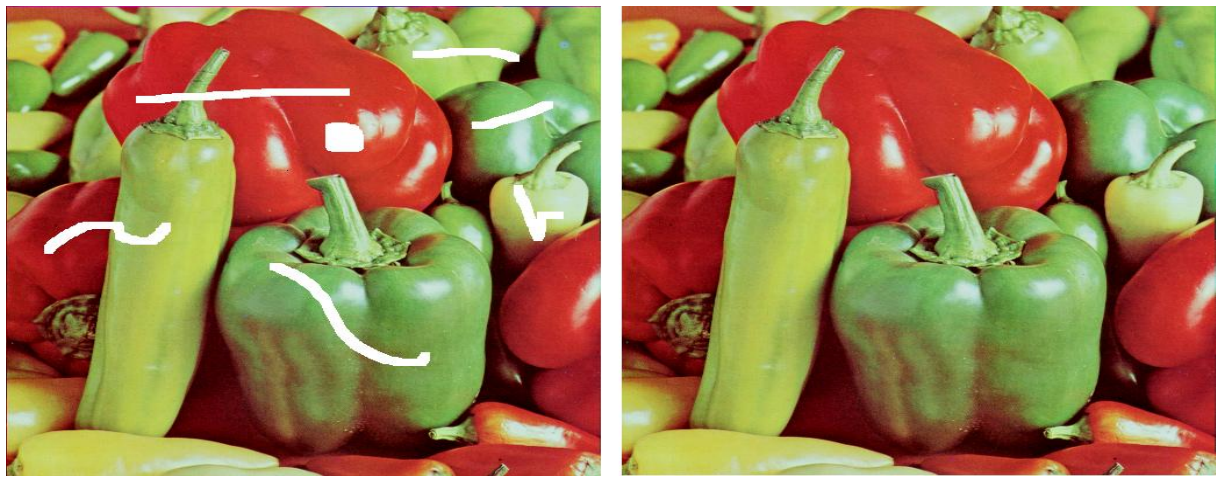
FIGURES INDEX

TABLE OF CONTENTS

Chapter 1

Introduction

In real world, many people need a system to recover the damaged photographs, artwork, designs, drawings etc. Damage may be due to various reasons like scratches, overlaid text or graphics, scaled image etc. Traditionally, inpainting has been done by professional artists. However, we could not expect the accuracy and quality if it was done by human and time-consuming process. Image inpainting is an important element in image restoration study. It makes use of the information not lost of the image to fill the lost or damaged part according to certain rules, so that after the inpainting, the images are close to mathematical point of view, it is to repair image in the regions of blank area in accordance with the information around them. Digital repair technology was introduced earliest by Bertalmio [1]. After that, it is widely used in image processing, visual analysis and film industries. At present, the image inpainting technology is a hotspot in computer vision and computer graphics, and has an important value in heritage preservation, film and television special effects production, removing redundant objects.



(a) (b)

Figure 1.1: Example of inpainting : (a) Original input image with destortion; (b) Results of applying a texture synthesis inpainting algorithm.

1.1 Image Inpainting

Image and painting restoration includes numerous different methods and restoration techniques such as surface cleaning, varnish removal, removal of previous fillings, repairing of rips and tears etc. Image inpainting is important for paintings; need some form of retouching or fixing due to accidental damage such as scratches and stains or fading due to age. Image inpainting is tedious and time-consuming job requiring careful attention to recreating the information to accurately restore the original image. In digital world we can define Inpainting as follows:

Image inpainting is a process of filling in missing data in a designated region of the visual input.[2]

Image inpainting refers to the process of filling-in missing data in a designated region of the visual input (Figure 1.1). The object of the process is to reconstruct missing parts or damaged image in such a way that the inpainted region cannot be detected by a causal observer. Applications range from the reconstruction of missing blocks introduced by packet loss during wireless transmission, reversing of impairments, such as cracks, scratches, and dirt, in scanned photographs and digitized artwork images, to removal/introduction of image objects such as logos, stamped dates, text, persons, and special effects on the scene. Typically, after the user selects the region to be restored, the inpainting algorithm automatically repairs the damaged area by means of image interpolation.To recover the color, structural and textural content in a large damaged area, inpainted (output) pixels are calculated using the available data from the surrounding undamaged areas. The required input can be automatically determined by the inpainting technique or supplied by the user. Since different inpainting techniques focus on pure texture or pure structure restoration, both the quality and cost of the inpainting process differ significantly. For example, exemplar-based techniques effectively generate new texture by sampling and copying color values from an undamaged source. Such an approach produces good results in replicating consistent texture seen in artificially generated imagery, but fails when it comes to reconstruct missing parts in photographs of natural scenes. This is due to the fact that most image areas consist of both texture and structure. Boundaries between image regions constitute structural (edge) information which is a complex, nonlinear phenomenon produced by blending together different textures. Therefore, it is not surprising that the state-of-the-art inpainting methods attempt to simultaneously perform texture and structure filling in.

In Bertalmio et al. [1] the underlying method is introduced that image conservation specialists use when restoring damaged artwork. General method they adhere to is as follows:

1.The global picture determines what is to be inpainted. The hole should be filled with data that is a natural continuation of the content found outside of it.

2.The structure in the area of the hole is continued into the missing region by the extension of the contour lines that arrive at the hole’s border.

3.The resulting regions are filled with colour based on the information found at the hole’s edge.

4.The final detail is added to the various sections of the hole.

5.The structure in the area of the hole is continued into the missing region by the extension of the contour lines that arrive at the hole’s border.

6.The resulting regions are filled with colour based on the information found at the hole's edge.

7.The final detail is added to the various sections of the hole.

1.2 Project Objectives

As stated earlier, the goal of inpainting is to try and fill a hole in an image with some meaningful data, based on the information present in the rest of the image. The idea is to recreate what there was originally, but this is not possible without some prior knowledge about the image. Art restoration works based on a similar idea, the conservators have impression of original image so they know what is required to fill in the holes. But in case of digital images, we only have the image we are working on and thus if we are filling in a hole that encompasses an entire object, it is impossible to replace that entire item based only on the information present. The aim of the algorithms presented is to fill in the holes in such a way that there is continuity and the defect isn't detectable and the result is visually pleasing. A large number and variety of algorithms exist for inpainting. The main purpose of this project is to look at a various methods available and there efficiency. Four different algorithms are compared and patch-based image compression is studied to compress the image using inpainting.

1.3 Report Structure

The remainder of the report is structured as chapter 2 covers the history of the inpainting field, chapter 3 introduces the four inpainting algorithms that are to be compared. It gives a comprehensive overview of the various approaches, chapter 4 describes the applications of inpainting and briefly define the framework for compression of image using inpainting and chater 5 shows the results fro the implemented algorithms and chapter 6 concludes the report, presenting some improvements for the chosen algorithms as well as some future work.

Chapter 2

Inpainting Techniques

Image completion is discussed in a lot of papers and different approaches are used for this. This chapter describes important methods of image comlpetion and some examples are shown.

2.1 PDE Approach

Bertalmio et al.[1] introduce the term image inpainting to computer science. The algorithm they presented was based on the idea of extending both geometric and photometric information that hits the border of the occluded area into the area itself. This is done through the continuation of these so called ‘isophote’ lines, which try to capture the direction of minimal change.In the algorithm the region that has to be inpainted will be filled-in by information of the region surrounding the hole. The curves of equal intensity (isophotes) arriving at the boundary are propagated inwards. Criminisi et al.[3] propose an algorithm inspired by the algorithm by Bertalmio et al.[1] and texture synthesis [4]. In contrast to the paper of Bertalmio et al. the unknown region is inpainted patch by patch. The sequence of which patch should be inpainted is based on the isophote information and the amount of known information surrounding the patch. In the algorithm of Oliveira et al.[5], the unknown region of the image is convolved with a Gaussian kernel. To prevent edges to be blurred the user manually specifies barriers for the diffusion. This algorithm is much faster than the other algorithms but it can only be used with very small unknown regions and is less accurate. An example of this algorithm is shown in figure

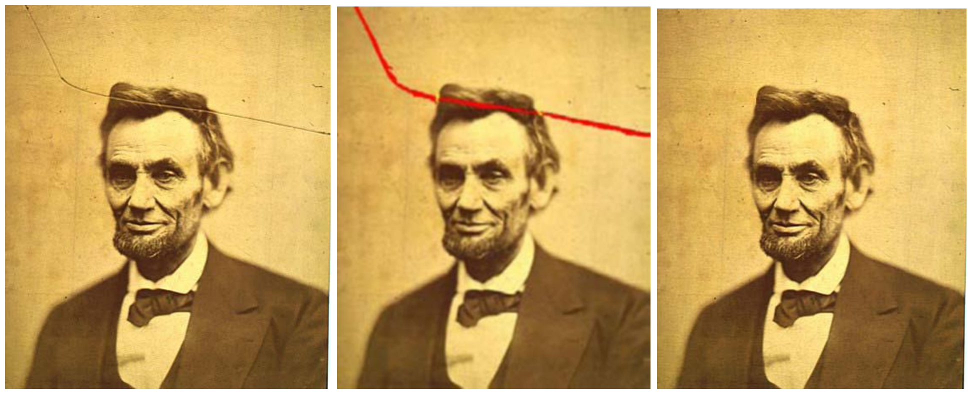


Figure 2.1: Olivera method [5], input, mask and result, note the diffusion barriers near the boundaries of the hair of Abraham Lincoln.

In the algorithm of Perez et al.[6] gradient domain reconstruction is used in image editing applications. Poisson equation is exploited to find out the value for missing pixel that locally matches the gradients while obeying the exact matching conditions at the contour. The author of [7] proposed a new inpainting algorithm based on propagating an image smoothness estimator along the image gradient. Similar to the algorithm of Bertalmio et al.[1], the image smoothness is estimated as the weighted average of the known image neighborhood of the pixel to inpaint. The fast marching method (FMM) is used to create a distance function to the initial boundary. The pixels of the unknown region are inpainted in the order of the distance to the boundary, proceeding from the smallest to the largest. This method is fast but creates blurry effects with larger unknown regions. An example of this algorithm is shown in Figure 2.2.

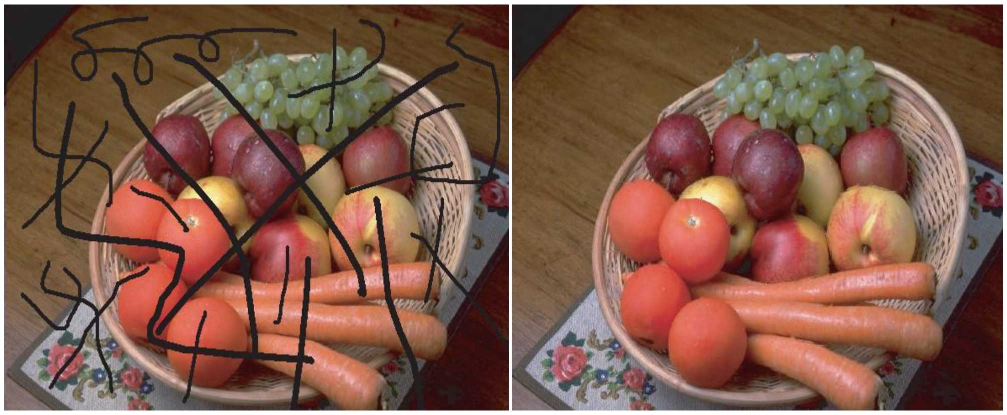


Figure 2.2: Telea method [7], input plus mask and result.

2.2 Texture Synthesis

A large number of algorithms have been proposed that are based around the concept of synthesizing 'artificial' texture based on a source sample. These algorithms have generally been designed to create an arbitrary sized texture based on a limited source sample. In this section we will look at two different types of synthesis techniques.

2.2.1 Pixel Based Synthesis

A handful of texture synthesis methods exist which are based on replicating a single pixel at a time. Efros et al.[4], method described uses the source texture as a starting point for the synthesis process creating new artificial texture by expanding it at its borders. This expansion is done pixel-by-pixel through the use of a template block (of a predetermined size) based on the pixel data that we already have and then searching the source area for possible matches. Once a match has been selected, the pixel at the center of the template is copied to its destination. The process continues until all pixels have been filled in. This method is extremely versatile in that it can reproduce both ‘stochastic’ and ‘deterministic’ textures, but its success depends on the size of the template window. A problem with using this approach is that it is extremely slow at synthesising the new texture in that the entire source sample area needs to be checked for each new pixel created. This can be improved upon through the use of the methods proposed in the next two sections, although, as we shall see, these improvements occasionally exhibit problems of their own.

2.2.2 Multiresolutional Approaches

Most of the work done in this area of texture synthesis is based on the method presented by Heeger and Bergen in "Pyramid-Based Texture Analysis-Synthesis" [9]. The approach is based on creating multiple intermediate images called image pyramids representing the texture at different roughness scales. A random noise image is transformed into a texture through the use of image pyramids and histogram matching. The noisy texture image’s histogram is matched to the original image’s histogram for each pyramid level. Finally, the noisy pyramid is collapsed to create a preliminary version of the texture. This process is performed for a number of iterations giving us the resulting synthetic texture. This process is relatively simple and provides good results for stochastic textures, but the matching of histograms fails to capture more structured textures. More work has done to improve the failure to reproduce deterministic textures by this algorithm. In addition to this, when synthesising textures larger than the original, the source texture sample is just tiled to fill the extra space.

2.2.3 Block Replicating Methods

One of the main characteristics of the previously mentioned texture synthesis approaches is that they work by synthesising only a single pixel at a time. Although this method works relatively well, the problem with this is that it is rather slow and even wasteful in that a pixels value is usually predetermined based on the local neighbourhood. The methods presented in this section try to remedy these problems by replicating blocks of pixels to generate the new texture data. In Efros and Freeman [10], set out to try and fix some of the shortcomings of the standard pixel by pixel texture synthesis methods. They devised a way of 'stitching' together blocks of pixels with one or more other blocks using a process they call 'image quilting' [10]. The method works by finding a 'minimum error boundary cut' between the existing texture pixels and the new block to be placed. This essentially means that the new block overlaps the existing data so as to minimize the error in the overlapping region. This approach produces some excellent results for all types of textures and is considerably faster than many of the single pixel texture synthesis method described earlier. However there is a drawback to using this method as an inpainting method in that as the texture fills the hole, towards the center it will become progressively harder to find a suitable block. This is because the error between the new block and the existing data will tend to be higher. As a result, noticeable artifacts may become present towards the center of this area. The multiresolutional approach algorithm determines a fill order for that level based on a

confidence value for each of the fragments in the occluded area. For each target fragment, it then searches for a match using an enhanced version of the method proposed by Efros and Leung in [4]. A different block copying approach was proposed by Criminisi et al. in [3]. It describe a technique similar to that in [4], but emphasize on the importance of a fill ordering. To do this, they use patch priorities to determine the fill order which are based on a confidence term (which favours patches surrounded by more data) and a data term (which tries to continue structure into the image). This essentially allows the structure around the hole to be propagated as long as a certain amount of information around the edges is present. It is less prone to creating artifacts due to overshooting the edges as is a common problem when using texture synthesis algorithms for inpainting purposes. The only drawbacks of this algorithm is that it can only handle linear edges and the patch copying phase needs an accurate source for the patches it is using. Texture synthesis is the process of algorithmically constructing a large digital image from a small digital sample image by taking advantage of its structural content. Texture synthesis can be also used to fill in unknown regions in images, the algorithm of Criminisi et al.[3] is partially based on these algorithms. An example of the algorithm by Efros et al.[4] is shown in Figure 2.3

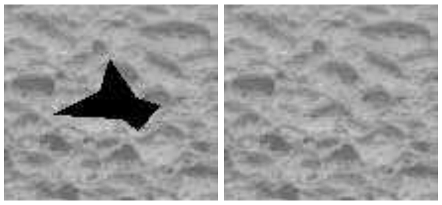


Figure 2.3: Texture Syntheses by Efros et al.[4], input plus mask and result.

2.3 Guided method

Dong Liu et al.[8] introduces a new direction in image inpainting. Egdes that are suppose to be continued in the removed region are found and their texture synthesis help in filling the hole with appropriate value from the known region surrounding the hole. This algorithm first fills in the unknown information around the support lines by dynamic programming or belief propagation, depending on the structure of the support lines. After that the rest of the image is filled in by texture propagation. The result of this algorithm is good. An example of this algorithm is shown in Figure 2.4.

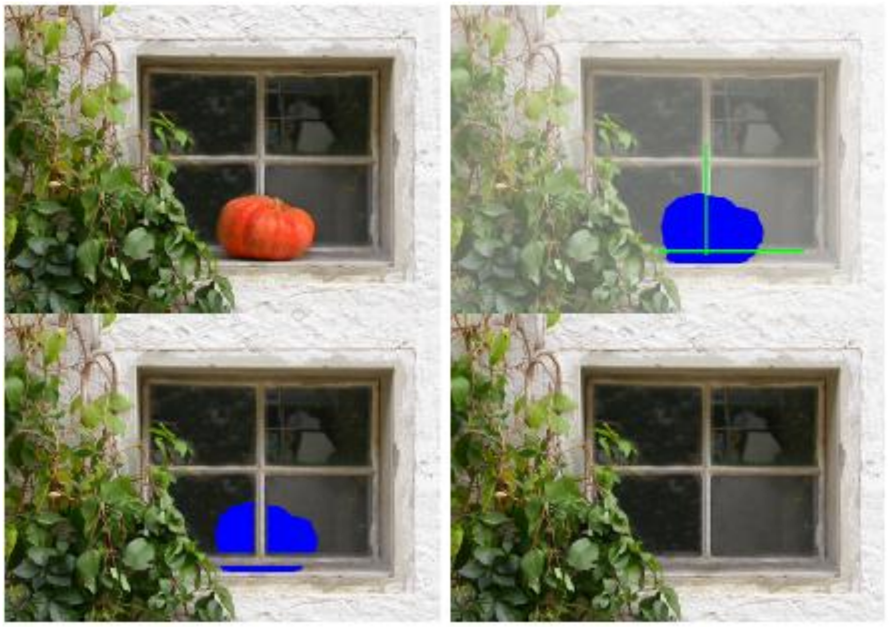


Figure 2.4: Dong Liu method [8], input, mask, edge detection, inpainted image

2.4 Other Algorithms

Inpainting methods are divided into different catagories. Although some of the methods could have been placed into multiple categories there was some specific feature that help in categorize in particular category. There are a number of algorithms which don't use a single distinctive approach. These algorithms tries to find out whether a missing block contains texture or structure by looking at the areas around it. Once classified, the algorithm then uses Efros and Leungs texture synthesis method for filling in texture blocks, and Bertalmio et al's image inpainting technique for structure blocks. The advantages of using such a method is that it gets around the problems faced by using just a single inpainting method, in that the diffusion based methods don’t work well with texture and the texture synthesis algorithms don’t reconstruct edges very well. The only problem however is when a block might contain both structure and texture. In these cases, the algorithm prioritizes the structure reconstruction thus leaving out texture information where it obviously belongs. Motivated by the problems presented above, Bertalmio et al. set out to create an inpainting method that could recreate both texture and structure simultaneously. In [11] the approach first decompose the input image into a texture and a structure layer using a bounded variation approach. Then Efros and Leungs method is used for the texture layer and Bertalmio et al’s inpainting method for the structural reconstruction. These two layers are then used to create the resulting image. Its success however depends heavily on the initial decomposition phase. The only apparent problem with using this method is that the resulting edges aren’t very sharp due to the diffusion steps present in the structure reconstruction algorithm, however this approach is an extremely good step towards creating a fully automated and accurate inpainting algorithm, which is capable of mimicking the method used by professional restorators.

Chapter 3 Overview of Algorithms

In this section, four different inpainting algorithms will be explored that were implemented for this project. First of all introduction of what the inpainting algorithms are to achieve is explained along with some common terminology that will be used throughout the section. Then their will be discussion on some methods of detecting the area to fill in. After that there will be detailed explanation of the four algorithms.

- Object removal by exemplar based inpainting method
- Poison Equation method
- 8-pixel neighborhood fast sweeping method.
- Edge-Based inpainting method

3.1 Inpainting Problem

The goal of the inpainting procedure is to fill in a missing area with some meaningful data. It is completely infeasible without some prior knowledge of what was there an inpainting algorithm would have to replace a hole with data that is a continuation of that which is found outside of the holes. The process is aimed at producing visually pleasing results as opposed to an exact reproduction of what was originally there. At the moment, there are only a few assumptions that are made on the input image. The first assumption is that this area has been marked by a uniform colour (for example through the use of a simple program such as MSPaint or the GIMP). The region to be filled in needs to be marked based on the user’s subjective judgement. In addition to this, the area must be at least three pixels wide. This width restriction is important for the missing area detection process, which will be explained shortly. Apart from these constraints, the missing area can be of any size and topology. With these restrictions, the only input the algorithms need is the image containing a marked occluded area. This image is then fed into the first stage, which involves detecting the uniformly marked region. This can be done by a number of different methods. Once we have a mask representing the occluded area, both the image and the mask are passed to the four respective algorithms. Each of the algorithms then performs their inpainting steps, outputting the fully inpainted image as the result. There is a set of symbols that represent various parts of the inpainting process (see figure 3.1).

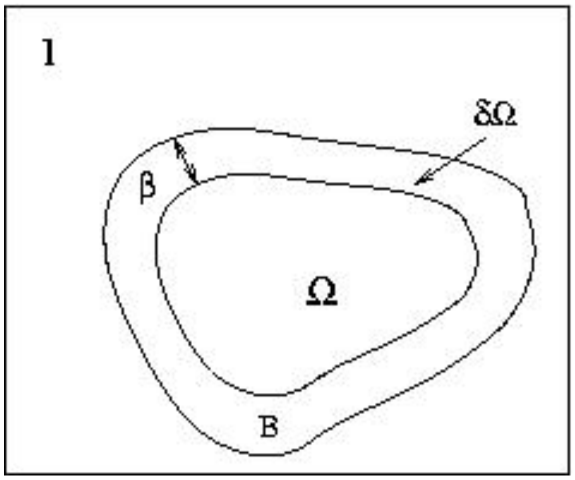


Figure 3.1: Inpainting terminology.

The hole in the image will be denoted as Ω, with the border to this area being δΩ. In addition to this, the band around the hole, which will be used for obtaining information to propagate into the hole will be denoted as B. Each of the algorithms may have some additional more specific parameters and terminology, but these will be introduced and discussed in their respective sections.

3.2 Missing Area Detection

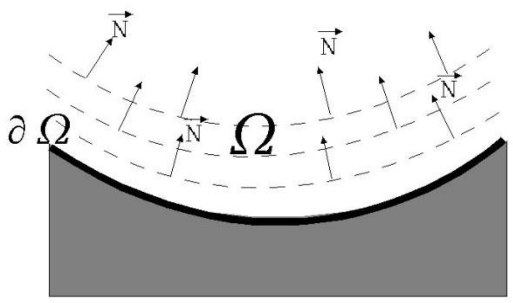


Figure 3.2[1]: Direction of the isophote lines .

The area, which is to be inpainted must have some sort of representation so that the algorithm distinguish between the source and the target region. This step could be considered as pre-processing step to the algorithms described here. It is neccessary to have a good representation of the area to be filled in as it directly influence the inpainting process. There are a number of different approaches for this purpose. One of the most common methods used is to supply a second image alongside the input image containing an overlay mask specifying the area to be inpainted. This method is one that has been adopted in a number of previous works [1][3]. Although this method is quite robust it allows the user to specify a region of any size/shape, the problem is that we need to create a separate file in which the mask covers the regions that need to be inpainted. This process can prove to be rather difficult in that it is easy to mark either too much of the image or too little. The second method is to detect the edges. The edge detector (canny) is applied to the entire image and then search for circular edges that contained an area of uniform variance. The detected edges didn't perfectly border the occluded area, the resulted edges can be frequently off by one pixel. May result in either too many or too few pixels being marked for inpainting. The method adopted is based on calculating variance values for each pixel and then searching for areas of zero variance. This involved first applying a 3x3 mask to each pixel to calculate the mean value. Then it is used for calculating its variance by obtaining the sum of the difference between that pixel's colour value and the mean value obtained. Once the variance values where calculated, the detection of the occluded area followed naturally in that all that was required was to search for regions that had zero variance. The problem with this however is that the region detected is one pixel too small in each direction. This is because the variance of the outermost pixels in the occluded area won't be equal to zero, due to the mean calculation using values other than those found in the uniformly marked area. Thus, the resulting area needs to be expanded by one pixel in each direction. The advantage of this technique is that the only input to the program is the original image with the region to be filled marked with a uniform colour. A minor problem with this method is that it can only detect regions that are larger than 3x3 pixels. This is due to the 3x3 area required for the mean calculations. If the missing area is smaller than the mask size, we won't be able to detect any pixels that have zero variance.

3.3 Object Removal by Exemplar-Based Inpainting

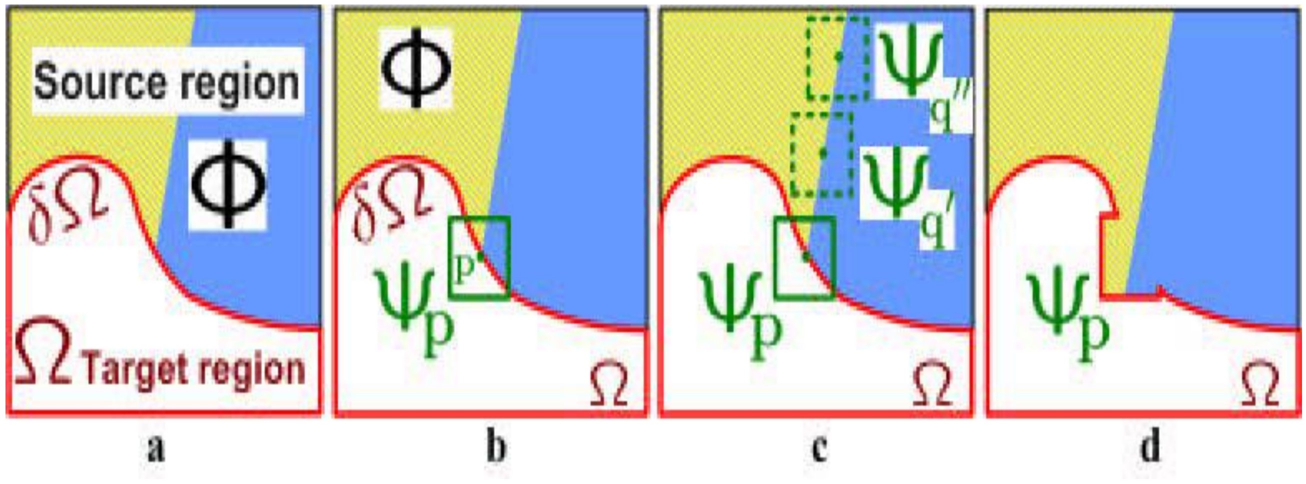


Figure 3.3[12]: Structure propagation by exemplar-based texture synthesis.(a) Original image, with the target region Ω, its contour δΩ and the source region Φ clearly marked. (b) We want to synthesize the area delimited by the patch Ψp centred on the point p ∈δΩ. (c) The most likely candidate matches for Ψp lie along the boundary between the two textures in the source region, e.g., Ψq' and Ψq'' . (d) The best matching patch in the candidates set has been copied into the position occupied by Ψp, thus achieving partial filling of Ω. The target region Ω has, now, shrank and its front has assumed a different shape.

The algorithm is based on an isophote-driven image sampling process. The exemplar-based approaches perform well for two-dimensional textures and for propagating extended linear image structures[12]. A target region, Ω, is selected to be removed and filled. The source region, Φ, may be defined as the entire image minus the target region (Φ = I−Ω). The size of the template window Ψ must be specified. Each pixel maintains a colour value if in Φ region and empty if in Ω region and a confidence value, which reflects our confidence in the pixel value, and which is set once a pixel has been filled. Patches along the fill front are also given a temporary priority value to determines the order in which they are filled. The algorithm iterates the following three steps until all pixels are filled:

3.3.1 Computing patch priorities

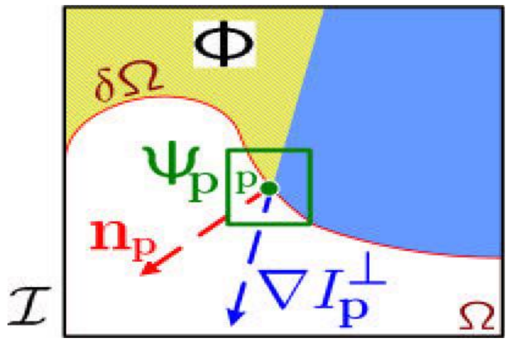


Figure 3.4[12]: Notation diagram. Given the patch Ψp, np is the normal to the contour δΩ of the target region Ω and $\nabla I_p^\perp$ is the isophote (direction and intensity) at point p. The entire image is denoted with I.

The priority computation is partial toward those patches which are on the continuation of strong edges and which are surrounded by high-confidence pixels. Given a patch Ψp centred at the point p for some $p \in \delta\Omega$ (fig. 3.3), its priority $P(p)$ is defined as the product of two terms:

$$P(p) = C(p)D(p). \qquad (1)$$

$C(p)$ the confidence term and $D(p)$ the data term are defined as follows[12]:

$$C(p) = \frac{\sum_{q \in \psi_p \cap \bar{\Omega}} C(q)}{|\psi_p|}, D(p) = \frac{|\nabla I_p^{\perp} . n_p|}{\alpha} \qquad (2)$$

$C(p)$ approximately enforces the desirable concentric fill order.Those patches which have more of their pixels already filled are filled first. The data term $D(p)$ defines function of the strength of isophotes on the front δΩ at each iteration. This term boosts the priority of a patch. This factor tends to synthesized linear structures first, therefore propagated into the target region.

3.3.2 Propagating texture and structure information

The patch $\psi_{\hat{p}}$ with highest priority is found. We then fill it with data extracted from the source region Φ. Search in the source region for that patch which is most similar to $\psi_{\hat{p}}$.

$$\psi_{\hat{q}} = \arg\min_{\psi_q \in \Phi} d(\psi_{\hat{p}}, \psi_q) \qquad (3)$$

3.3.3 Updating confidence values

The *C(p)* is updated in the area enclosed by $\psi_{\hat{p}}$ after the patch $\psi_{\hat{p}}$ has been filled with new pixel values, by:

$$C(q) = C(\hat{p}) \quad \forall q \in \psi_{\hat{p}} \cap \Omega. \tag{4}$$

3.4 Poisson-based image inpainting

3.4.1 Image decomposition

Image I_0 is decomposed into texture and structure image. For an image texture information is assumed to be noise as compared to structure information. A classical variational denoising algorithm i.e. total variation (TV) minimizing process [13] is used for decomposition of the image. This algorithm yields sharp edges in the output image *I* while maintaining the fidelity to the original noisy input image I_0. The energy function with scalar fidelity controller λ is defined as[13]:

$$E = \int_{\Omega} \left(|\nabla I| + \frac{1}{2}\lambda(I - I_0)\right) dxdy \tag{5}$$

According to Euler- Euler-Lagrange equation,

$$\nabla.\left(\frac{\nabla I}{|\nabla I|}\right) + \lambda(I_0 - I) = 0 \tag{6}$$

The solution can be achieved by the gradient descent method, which means we solve

$$I^{(n+1)} = I^{(n)} + \nabla T.\lambda\left(I_0 - I^{(n)}\right) + \nabla T.\frac{I_{xx}I_y^2 + I_{yy}I_x^2 - 2I_xI_yI_{xy}}{(I_x^2 + I_y^2)^{3/2}} \tag{7}$$

$$I_x = \frac{\partial I^{(n)}}{\partial x}, I_x = \frac{\partial I^{(n)}}{\partial y}$$

$$I_{xx} = \frac{\partial^2 I^{(n)}}{\partial x^2}, \ I_{yy} = \frac{\partial^2 I^{(n)}}{\partial y^2}, I_{xy} = \frac{\partial^2 I^{(n)}}{\partial x \partial y}$$

where $I^{(n)}$ is the result of the nth iteration, and ΔT is the step size. The structure image is extracted by increasing the number of iteration by assigning λ to a small value. Texture image can be extracted from the residual image.

3.4.2 Method of Poisson-Based Structure image inpainting

The exemplar-based inpainting method performs well on texture information and is able to handle large holes. But, due to the direct patch duplication, the algorithm tends to produce block effect in processing of structure information degrading the visual effect of repaired images. To improve the performance of structure inpainting, the exemplar-based inpainting approach is combinedwith the Poisson equation. The theoretical foundation is that the Poisson equation is able to reconstruct a scalar function from a guidance field and a boundary condition[5]. The Poisson equation can also be used as a least-squares minimization method, so block effect introduced by block duplication can be removed via reconstruction. First apply the Laplacian operator to the structure image and find the Laplacian field. In the Laplacian field edges are enhanced and backgrounds are almost completely removed. It provides a more accurate structure inpainting result when employing the exemplar-based method. Then the structure image is reconstructed by the Poisson equation taking inpainted Laplacian field as the guidance field.

3.5 Inpainting algorithm based on the 8-neighborhood fast sweeping method

Take a small neighborhood $B_\varepsilon(p)$ of size ε of the known image around p, for ε small enough, we consider a first order approximation $I_q(p)$ of the image in point p, given the image I (q) and gradient $\nabla I(q)$ values of point q [14]:

$$I_q(p) = I(q) + \nabla I(q)(p - q) \qquad (8)$$

Next, we inpaint point p as a function of all points q in $B_\varepsilon(p)$ by summing the estimates of all points q, weighted by a normalized weighting function ω(p,q) .

$$I(p) = \frac{\sum_{q \in B_\varepsilon(p)} \omega(p,q)[I(q)(p-q)]}{\sum_{q \in B_\varepsilon(p)} \omega(p,q)} \tag{9}$$

The weighting function ω(p,q) , is designed such that the inpainting of p propagates the gray value as well as the sharp details of the image over $B_\varepsilon(p)$. To inpaint the whole Ω, we iteratively apply Eqn.(9) to all the discrete pixels of ∂Ω, in increasing distance from $\partial\Omega_j$'s initial position $\partial\Omega_j$, and advance the boundary inside Ω until the whole region has been inpainted. Implementing the above requires a method that propagates ∂Ω into Ω by advancing the pixels of ∂Ω in order of their distance ∂Ωto the initial boundary $\partial\Omega_j$.

(1) Initialize: Set $u_{ij}=\infty$ for (i, j)∈Γ .For each γ ∈Γ set u(γ) = 0 . For z ∈Γ not in Ω_d compute the exact solution at the vertices of the grid cell in which z lies.

(2) For each sweep direction in {(x+, y+), (y-, x-),(x+, y-), (x, y+)} iterate through each grid pointaccording each of the sweep directions or according to the fast marching heap sort.

(3) At each grid E with index (i, j) If all the neighbors are infinity, skip. If there is at least one non-infinity neighbor in both the x- and y- direction, Let P and Q are the smallest one in each direction. Inpaint (p,q).

3.6 Edge-based inpainting

In edge-based inpainting some blocks are removed from the image. The edges are extracted from the original image and with some structural and textural information related to these removed blocks are used to inpaint the removed blocks. At the time of image restoration edge-based inpainting and texture synthesis make full utilization of the transmitted edges and naturally restore the removed blocks. In [8] inpainting and synthesis methods are integrated with normal image encoder / decoder such as JPEG codec shown in Figure 3.5.

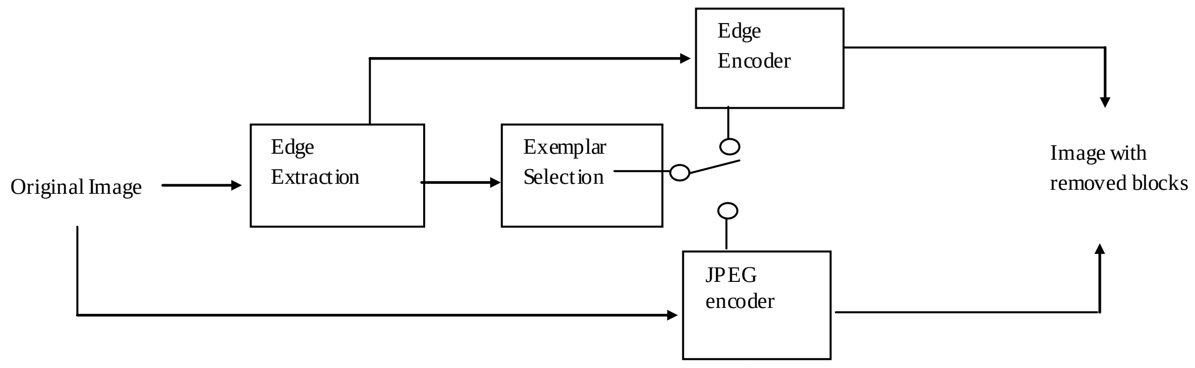


Figure 3.5: Region removal and Edge Extraction from image

Edge extraction is performed on the original image. Then, according to exemplar selection, some blocks are removed and the others are encoded. The coded blocks are called *exemplars*[3] which helps in inpainting and synthesis. For the removed blocks, corresponding edges are encoded and transmitted. At decoder side, edge-based inpainting and texture synthesis are performed successively, both of which utilize edges and exemplars to restore the removed blocks as shown in Figure 3.6.

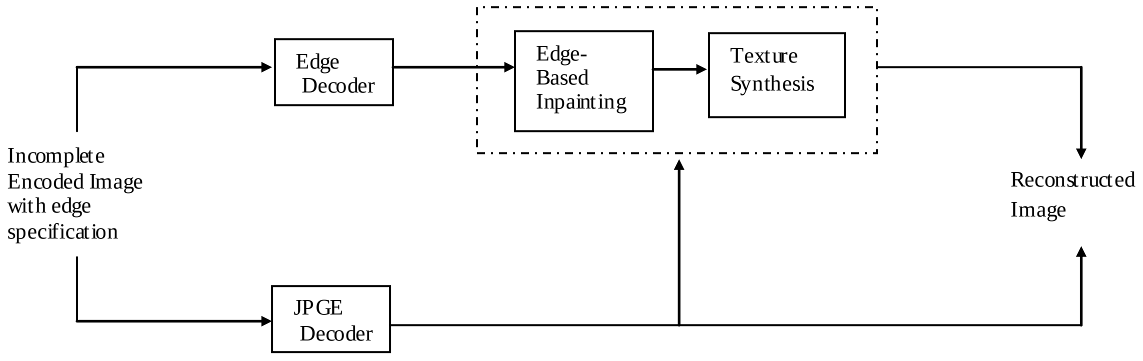


Figure 3.6: Procedure to reconstruct the image using Inpainting

Edges are regarded as the assistant information for inpainting or synthesis for the following reasons:

1. Edges are crucial to the restoration of salient structures in images.
2. Edges are concise and easy to describe in the compressed form.
3. Edge information is a low-level image feature, can easily be extracted as compared to some semantic image features, so the implementation of edges enables a fully automatic system.

3.6.1 Edge Extraction and Exemplar Selection

Edges are extracted from the image by a edge detection method. Then an edge thinning process is conducted to make the edge one pixel wide. The exemplar selection is conducted at non-overlapped 8×8 block level, each classified into *structural* or *textural*. Classification is done as if a block contains more than one-quarter pixels which are located within a short distance (e.g. 5-pixel) from the edges, it will be regarded as structural, otherwise textural. These two kinds of blocks are processed independently.

Stuctural exemplar selection is done on the basis of necessary and additional blocks in order to improve the visual quality of reconstructed image. White blocks in figure 3.7(a) are necessary as end or conjunctions of edges contain the transition between

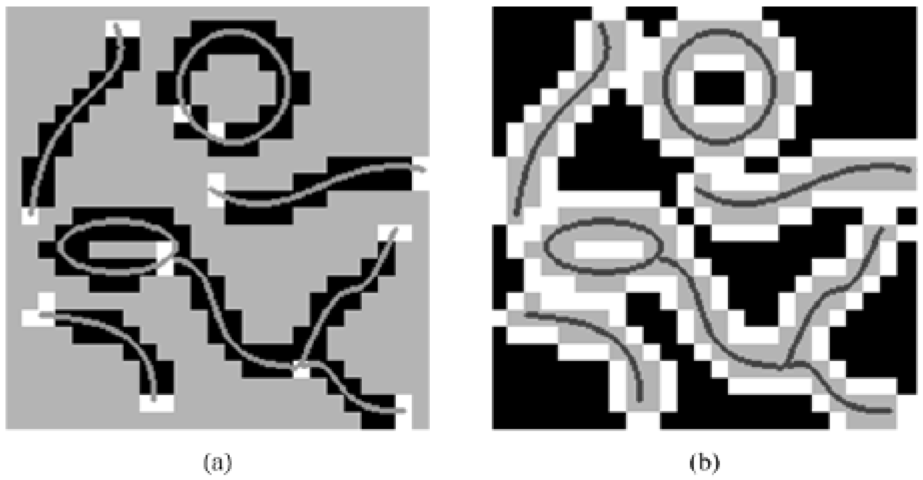


Figure 3.7[8]: Illustration of exemplar selection. Thick curves denote edges.

image partition and cannot be restored. For circular edges, two necessary blocks are identified in its inner and outer regions, respectively. Additional structural blocks are selected to represent local variations. Block containing obvious variation have the priority to be preserved. Each black block in Figure 3.7(a), denoted as *Bi*, is assigned a variation parameter *Vi* defined as

$$V_i = \omega_{\mathbf{1}} Var(B_i) + \omega_2 \sum_{B_j = \mu_{\mathbf{4}} \in (B_i)} \left| E(B_i) - E\left(B_j\right) \right| \qquad (10)$$

where ω_1 and ω_2 are weighting factors, and $\mu_4()$ indicates four neighbors of a certain block. The functions $Var()$ and $E()$ are the variance and the mean of pixel values, respectively. Note that in one block, the different partitions separated by edges are independent in calculating the variance and the mean; then the resulting parameters of different partitions are summed up to get the total variation parameter of the block. Given an input threshold, the blocks with higher variation parameters will be selected as exemplars. Similarly, textural exemplars are also selected in two steps. If a textural block is next to a structural one white blocks in figure 3.7(b), it is considered as necessary. Such blocks clearly separate textural blocks from structural ones and facilitate texture synthesis at the decoder. Additional blocks are also selected according to their variation parameters which can be calculated by (10). The textural blocks with higher variation parameters will be preserved as exemplars.

3.6.2 Image Restoration

Image restoration method fully utilize the transmitted edges. Edge-based inpainting (EBI) is performed to restore the pixel values on the edges and neighboring the edges, and texture synthesis is utilized for the restoration of textural regions. For each edge, EBI is completed in two steps. First, a linear interpolation is adopted to generate the unknown pixels on the edge from the known ones on the same edge. Second, the neighborhood of an edge, known as *influencing region*, is progressively filled-in by pixel generation. Edges represent the structural information that consists of the discontinuities in images. But the textural information, referred to as kinds of regularities in statistics, geometric shapes, etc., also exists in the neighborhoods of edges. These two types of information are simultaneously restored by the pixel generation method. For each unknown pixel in the influencing region, known as the target pixel, two candidate pixels are found. One is denoted as structural candidate (S-candidate), which lies within the influencing region; the other, textural candidate (T-candidate), locates within the neighborhood of the target pixel. S-candidate is chosen to minimize the following weighted sum of squared difference (SSD):

$$D_s = \sum_i (\left|d\left(x_s^i\right) - d\left(x_t^i\right)\right| + 1) \times \left|f\left(x_s^i\right) - f\left(x_t^i\right)\right|^2 \qquad (11)$$

where $\boldsymbol{x_s^i}$ is the *i*th pixel in the neighborhood of the S-candidate. $\boldsymbol{x_t^i}$ is the *i*th pixel in the neighborhood of the target pixel. *d*() means the distance from each pixel to the edge and *f*() is the reconstructed image. By minimizing (11), we can find the S-candidate whose distance from the edge is similar to that of the target pixel. Differently, ordinary SSD is considered as the criterion to choose T-candidate:

$$D_T = \sum_i \left|f\left(x_T^i\right) - f\left(x_t^i\right)\right|^2 \qquad (12)$$

$\boldsymbol{x_T^i}$ represents the *i*th pixel in the neighborhood of the T-candidate. At last, one of the two candidates will be chosen to fill-in the target pixel. The choice is made by comparing $\boldsymbol{D_T}$ and $\boldsymbol{D'_S}$ which are defined in (12) and (13) respectively:

$$D'_s = (d/d_0 + pd/pd_0) \times \sum_i \left| f\left(x_s^i\right) - f\left(x_t^i\right) \right|^2 \qquad (13)$$

where $\boldsymbol{d_0}$and $\boldsymbol{pd_0}$ are constants. $\boldsymbol{d = d(x_0)}$ stands for the distance from the target pixel to the edge, and $\boldsymbol{pd}$ indicates the distance from the target pixel to the S-candidate. If $\boldsymbol{D_T}$is smaller than $\boldsymbol{D'_s}$, T-candidate is selected to fill-in the target pixel, otherwise S-candidate is selected. After EBI, a patch-wise texture synthesis algorithmis adopted to restore the remaining regions, where patches are blocks with fixed size. The patch furthest from the edges will be generated first, so as to prevent the interference of edges in texture synthesis. For a target patch which is partially unknown, a known patch is searched out from its neighborhood in terms of the least SSD between two patches. This process is repeated until no unknown pixel exists.

Chapter 4
Inpainting Applications

Inpainting is an artistic synonym for *image interpolation*, and has been circulated among museum restoration artists for a long time. In photography and cinema, is used for film restoration; to reverse the deterioration (e.g., cracks in photographs or scratches and dust spots in film). It is also used for removing red-eye, the stamped date from photographs and removing objects to creative effect. This technique can be used to replace the lost blocks in the coding and transmission of images, for example, in a streaming video. It can also be used to remove logos in videos. There are many objectives and applications of this technique.

4.1 Removal of Defects/Scratches in image

As mentioned above inpainting has the main application in artistic world. Any artifacts that are deteriorated due to natural process or some miss happening are restored by artists using the colors in the surrounding, symmetry in the areas outside and inside the hole. This thing can be done by knowing how the image would look before the deterioration and is very tedious and lenthy process for manual restoration. This thing can be done by use of inpainting algorithms. These algorithms make the process automatic and fast. As shown in Figure 2.1, inpainting method is used to reverse the effect of hair line defect. Like this images which are distorted accidently or for fun or unknowingly by a child can be restored by scanning and applinf the suitable algorithm for inpainting. Figure 4.1 shows the scratched image restoration using inpainting.

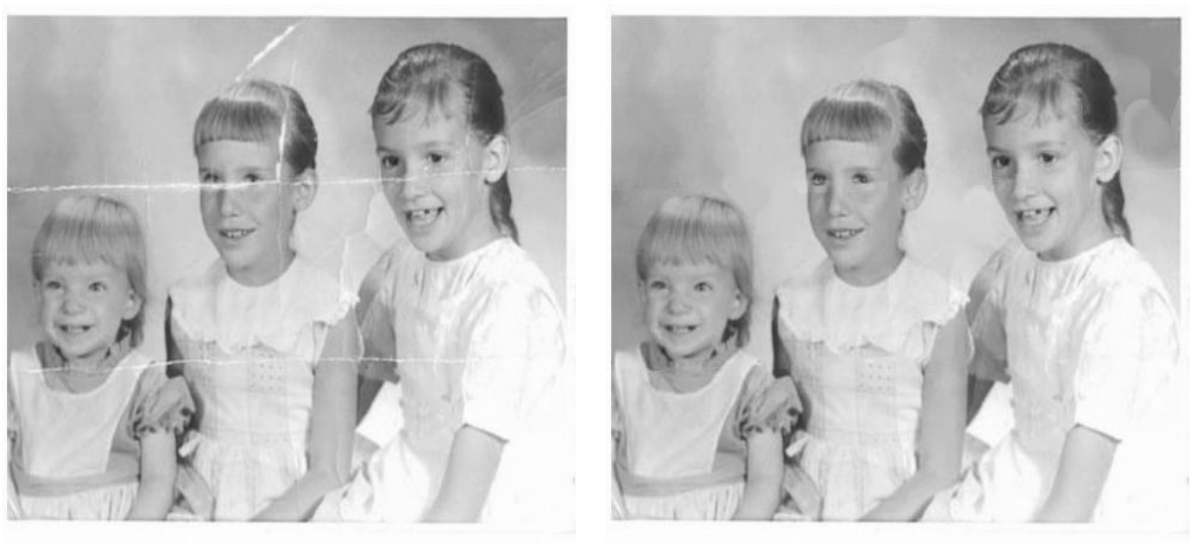
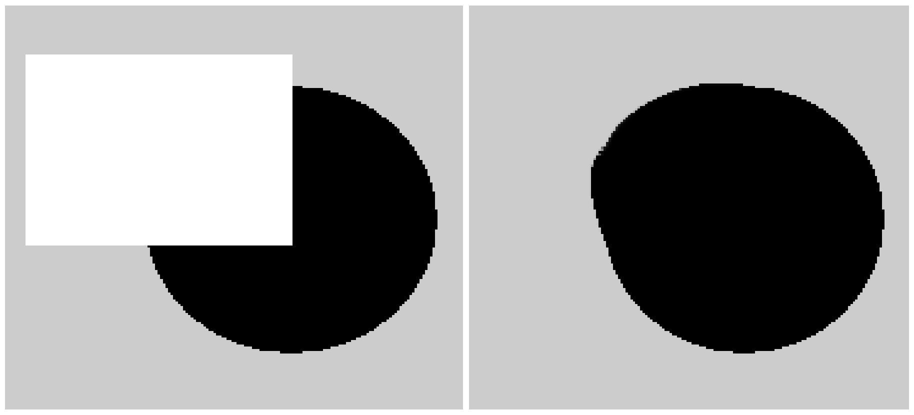

(a) (b)

Figure 4.1:Scrached image restoration (a) Scratches on an old pictures of three sisters (b) Scratches are filled in by inpainting procedure.

4.2 Fill in Missing Parts of images

Missing parts in an image can be filled in using inpainting methods. The missing parts are filled in by the background of the image or the surrounding objects if are incomplete can be completed to fill the region. The choice is done on the basis of algorithm used for the inpainting. If algorithm used analysis the pixels surrounded the missing pixel it will fill the pixel with greater probability and minimum displacement from the target pixel. The algorithm analyzing the blocks find out the best possible examplar[3] for the missing area block. The algorithm's choice is done on the basis of size of missing area. Small area can be filled by any efficient inpainting algorithm. For bigger region algorithms in [3][5][9] etc.can be used. Figure 4.2 depicts the missing area filling using inpainting.



(a) (b)

Figure 4.2: missin circular region in image(a) White portion in image is the missing region (b) filed region using inpainting procedure.

4.3 Deleting Unwanted Objects from image

Inpainting can be used to remove unwanted objects from the image. The unwanted element is first selected and then a mask is created for the image without that object and an inpainting algorithm is applied which fill in the surrounded region into the masked area. This procedure can create the special effects if used the image as frames of a video. This features can be used to create frames in the animation videos. Objects can be removed by [3] [12] [14] using exemplars. The removal of objects may need to fill in some textue or structure or curves and egdes in it then [8] [10] [11] approaches can be used. Figure 4.3 shows the object removal example by inpainting.

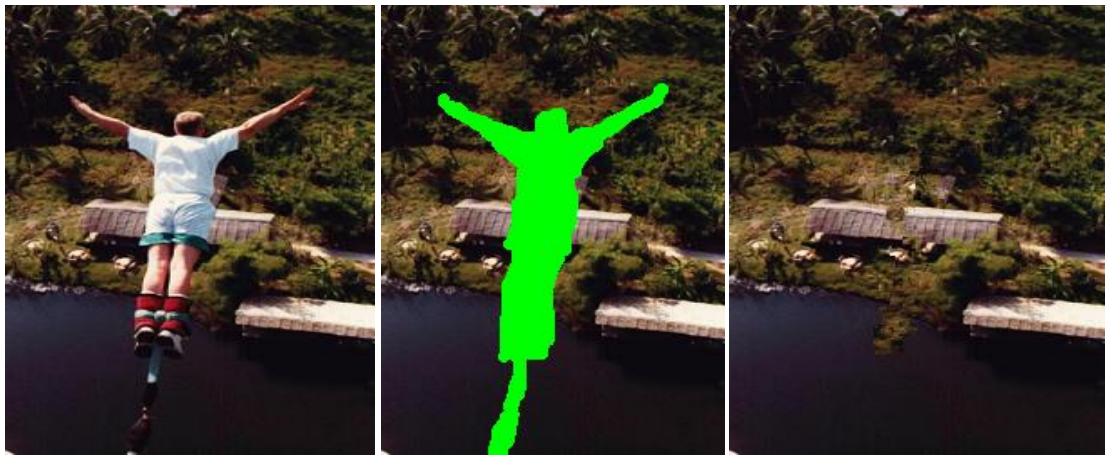


Figure 4.3:oject removal by inpainting (a) original image, (b) mask defining the region to be removed and the remaining region, (c) filling in the region to be removed by inpainting

4.4 Inpainting in Medical Field

In medical world inpainting plays a very important role. Specially in *Dermoscopy Images.* These images are related to skin cancer. The images are analysed by the physician to find out the cancer on the skin and recovery. The Artifacts in dermoscopy images are air bubbles and hair on skin of patient. These artifacts in images interfere with computer-aided diagnosis. Before reaching to a decision it is necessary to remove these unwanted things from the images. [15] proposed the algorithm "Feature-preserving artifact removal (FAR)" for this pupose with Detection: Explicit curve modeling detection and Exemplar-based inpainting for removal of the unwanted objects.

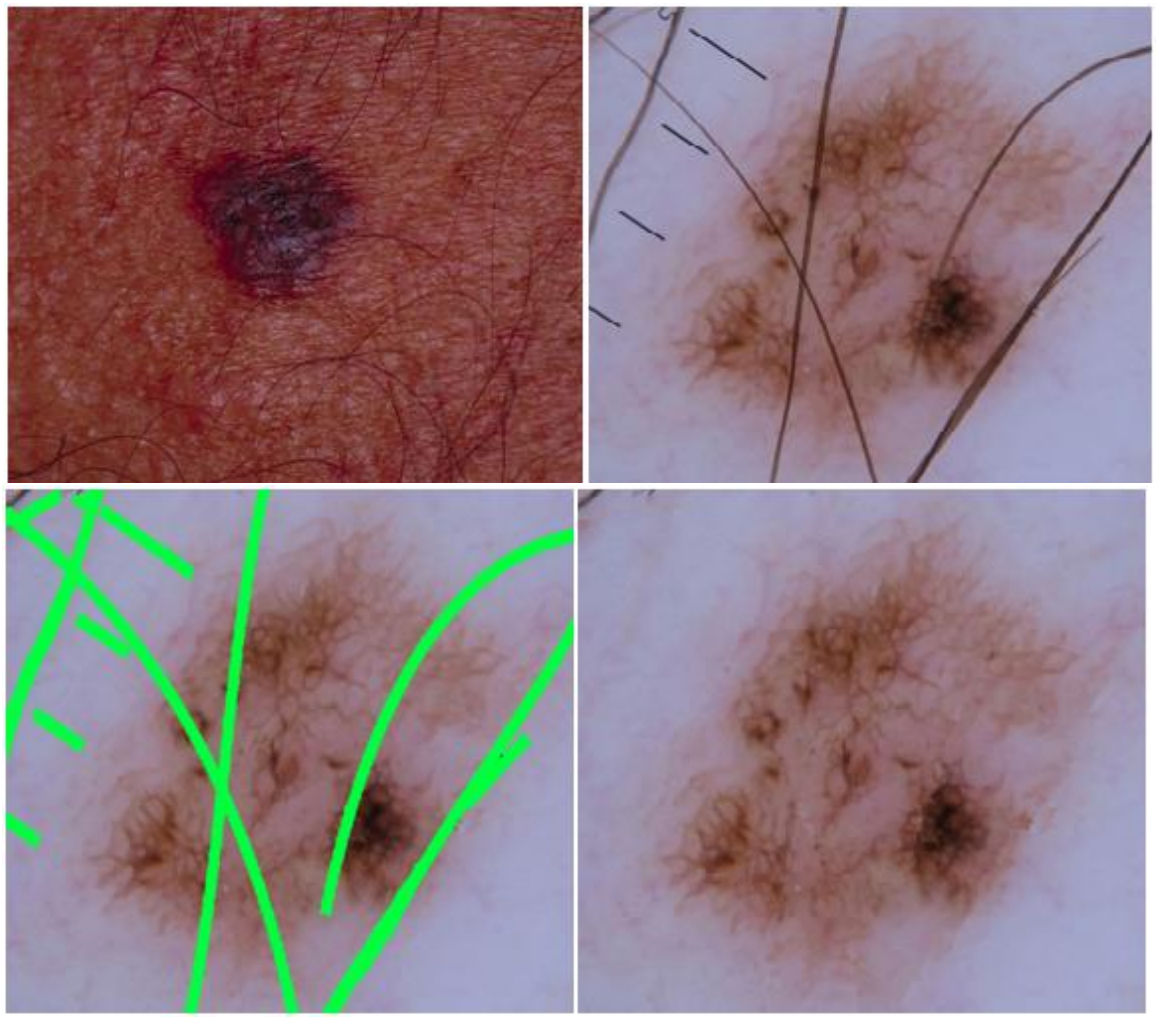


Figure 4.4 Dermoscopic Inpainting clinical view(top left) dermascopic view(right) mask(bottom left) inpainted image

4.5 Removing Text from image

Inpainting can be used to remove time stamp, date or some text written on the image. First of all we have create a mask for the text to remove and then after creating the mask the image with text and the image with mask are compared to find out the region to remove and the best patch or block to replace the text in the image. Figure 4.5 shows the effect.

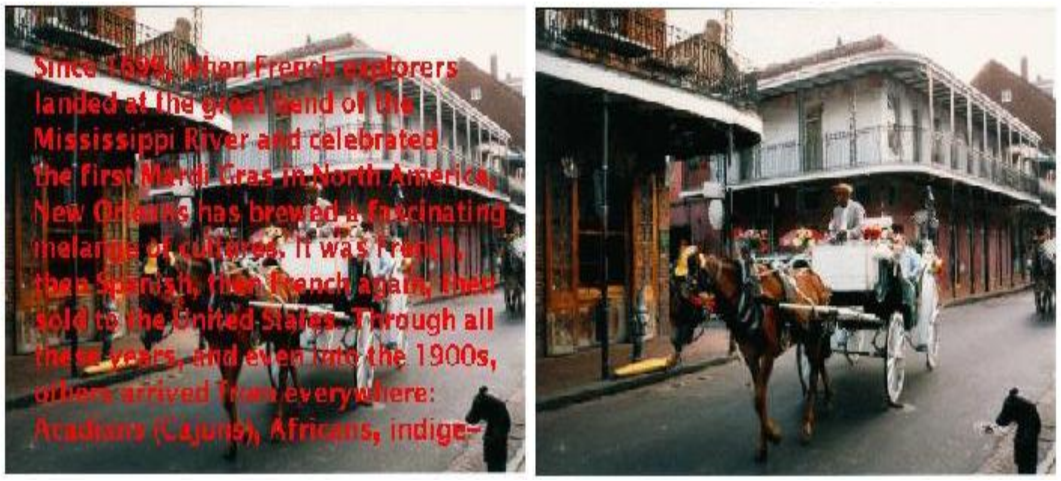

Since 1699, when French explorers
landed at the great bend of the
Mississippi River and celebrated
the first Mardi Gras in North America,
New Orleans has brewed a fascinating
melange of cultures. It was French,
then Spanish, then French again, then
sold to the United States. Through all
these years, and even into the 1900s,
others arrived from everywhere:
Acadians (Cajuns), Africans, indige-

Figure 4.5 Text removal from an image

4.5 Image compression

Image compression is the field in which motive is to minimize the image size in such a way that it take less space, less time to download and upload without any visible defect in it due to data loss during compression. Inpainting can be used for image compression. At encoder side image blocks of fixed size are analysed for the normal encoding and inpainting cost. Blocks having smaller inpainting cost are removed and some assistance information is placed for decoder to assist decoder in inpaint that block with most eligible surrounding block. At decoder most blocks are removed has the compression is high. Computation cost for this process can be high at encoder side but result is good. For decoder inpainting cost small as the assistance information is transmitted by the encoder to help in inpainting. The cost of computation can be lowered by using wavelet transform which is under research. The idea behind this procedure is that there are many recurrent patterns in an image, thus some image patches can be well inferred from the others[16].

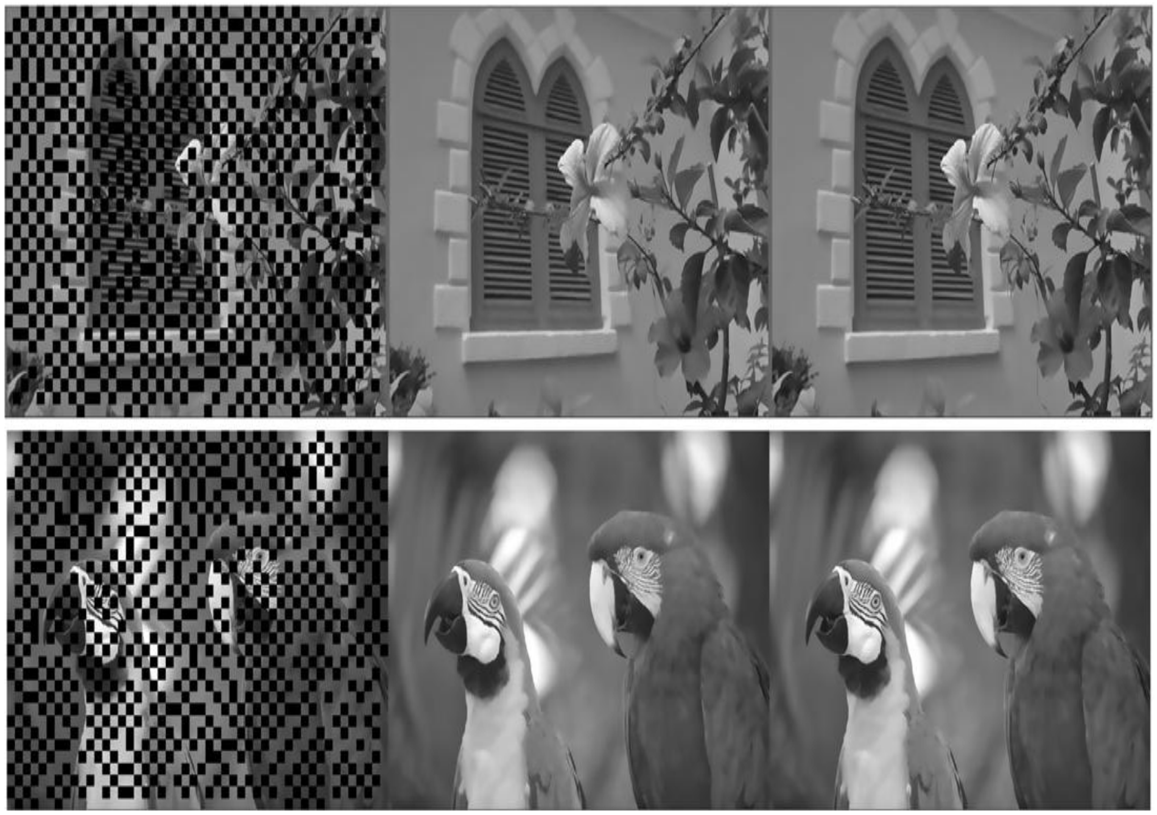


Figure 4.5 [16] image compression. First part shows the compressed image with black removed blocks rest of the blocks are encoded using image encoding techniques , second image shows the decoding with inpainting and third shows the image with normal encoding technique

Chapter 5

Results

Various inpainting algorithms are studied for this project. The requirement of the quality of image needed and the time to get that inpainted image help to choose the algorithm best suited. The fast methods can produce blurry results other methods can produce quality images but time can be large. The images used to show the applications of inpainting are lena, bungee, parrot, three sisters, circle etc. Images include the original image, mask for the image and the inpainting result. Results shows the remarkable outputs. Depending upon the size of images time taken is less or more. Standard file formats used are png, jpg , tiff and bmp. Some algorithms are implemented for gray scale images and some on RGB due to time constaint and to be specific in certain format and color requirements. Gray scale images take less than 15 seconds to be processed and colored may take time in minutes. Results show out put for object removal, scratch removal and new object creation. The papers implemented produce the results on different images as follows:

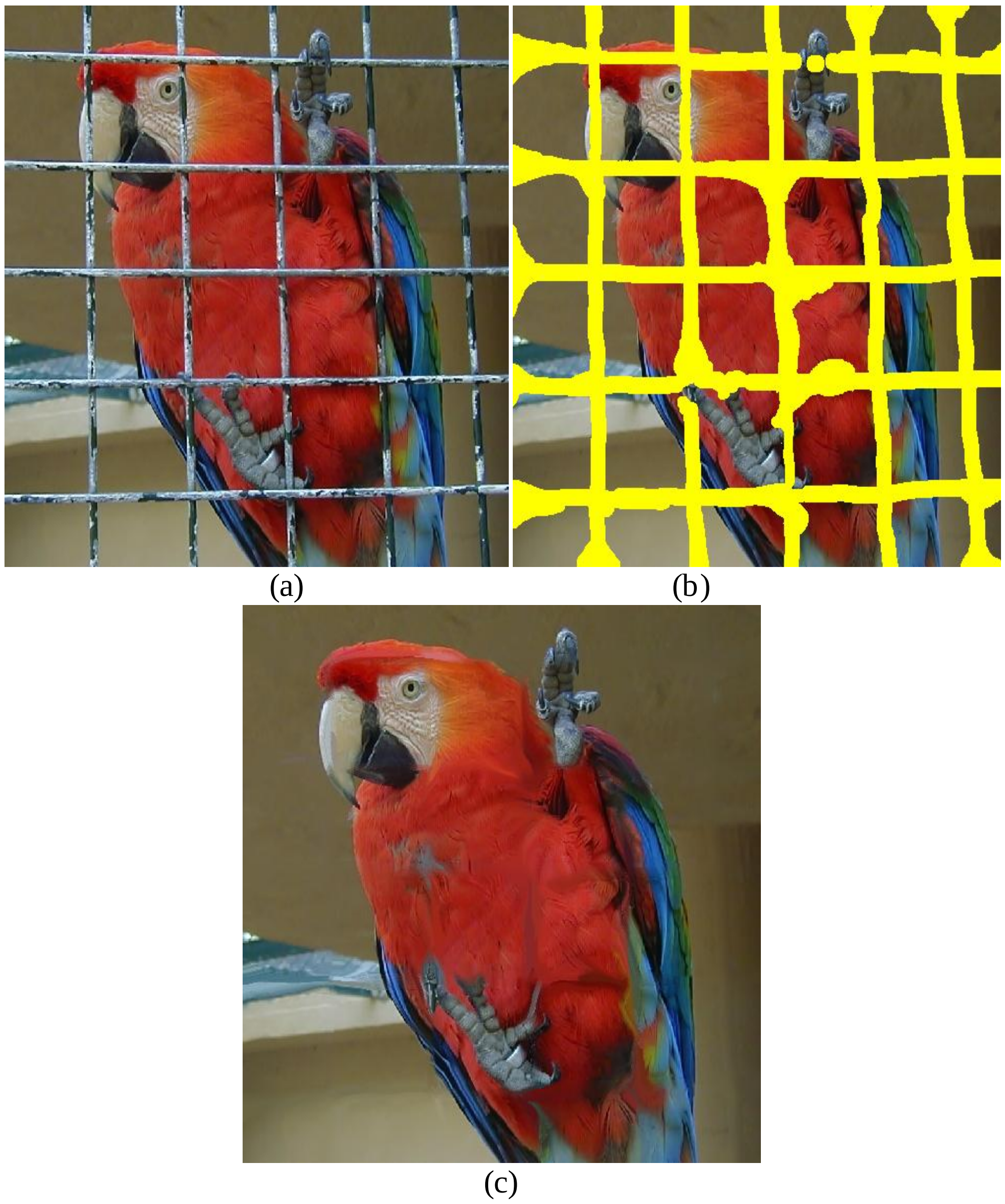
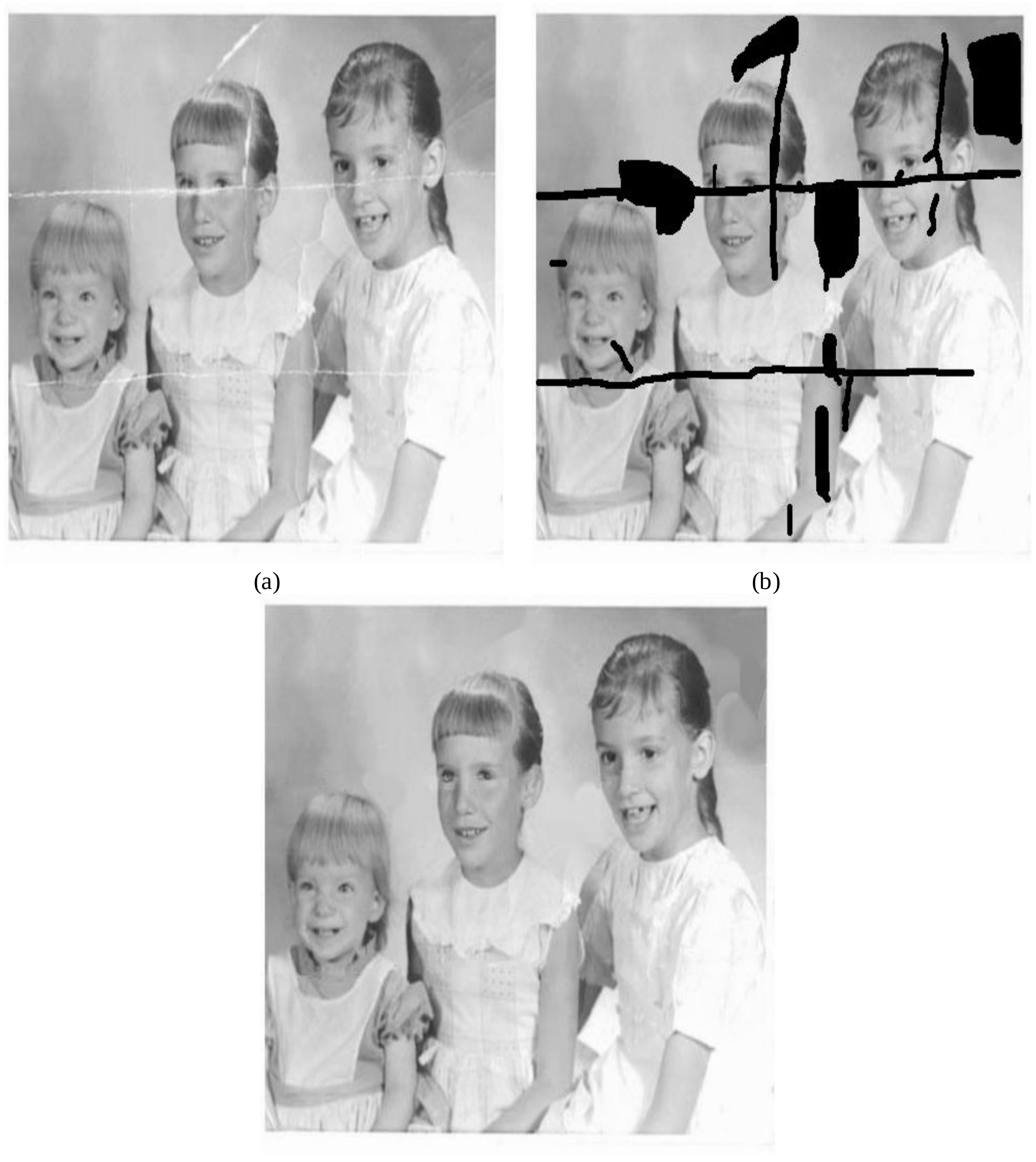
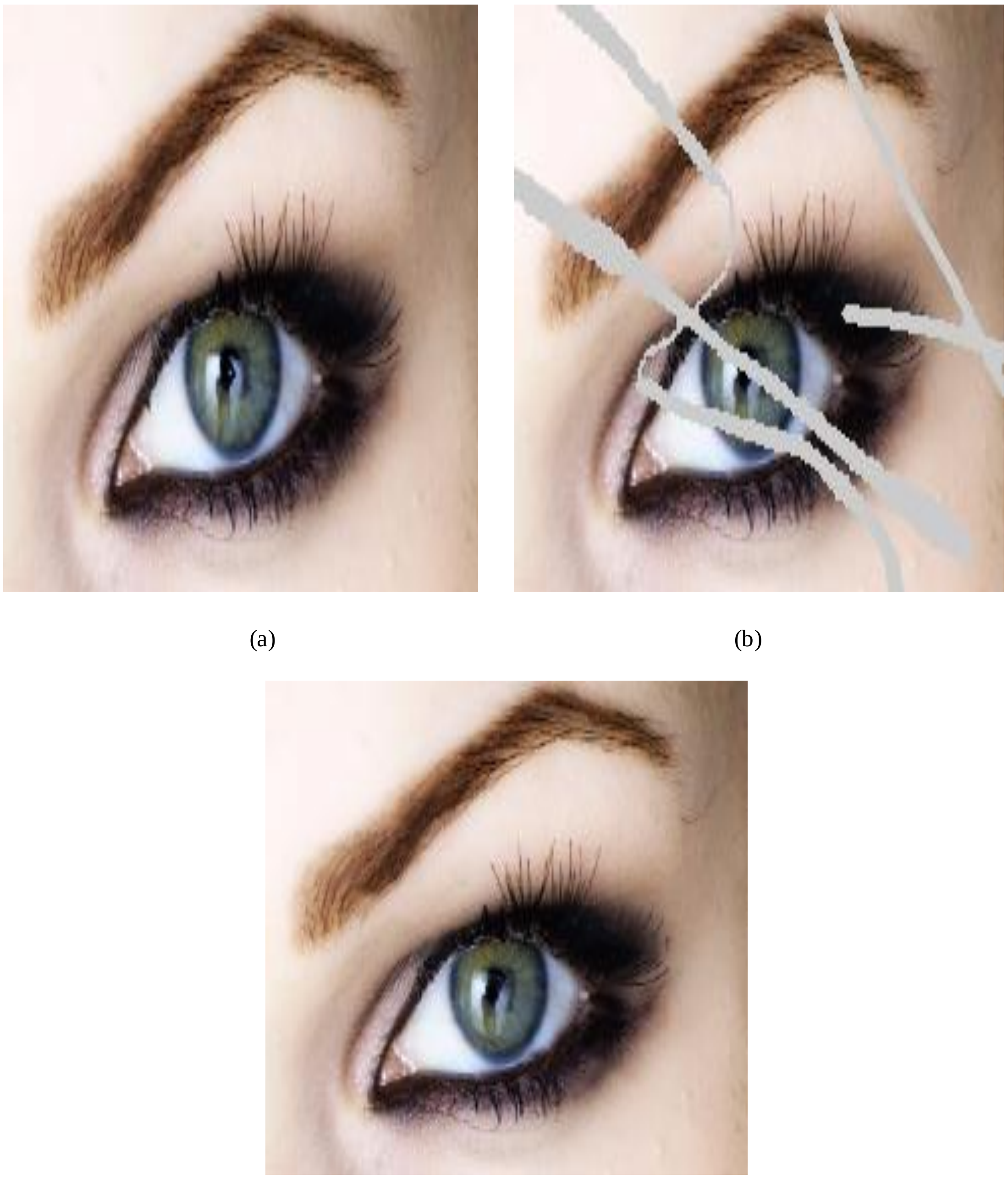


Figure 5.1 the original parrot image(a), and the yellow mask for the image to remove the fence from it(b) and image after appling Inpainting algorithm(c)


(a) (b)

(c)

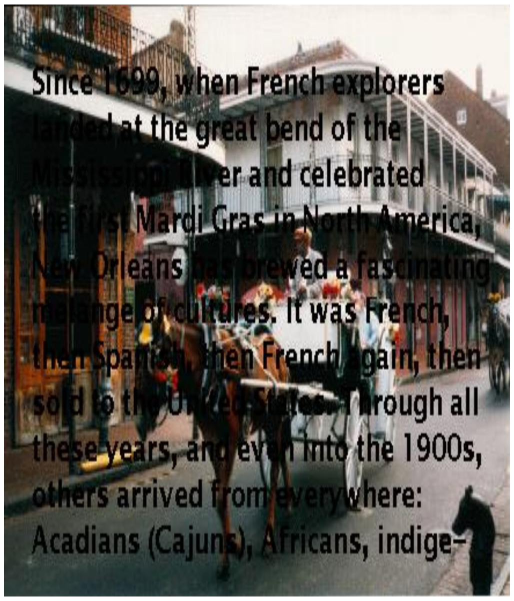
Figure 5.2 (a) original imageof three girls with scratches, (b) black mask to be inpainted region, (c) Inpainted image



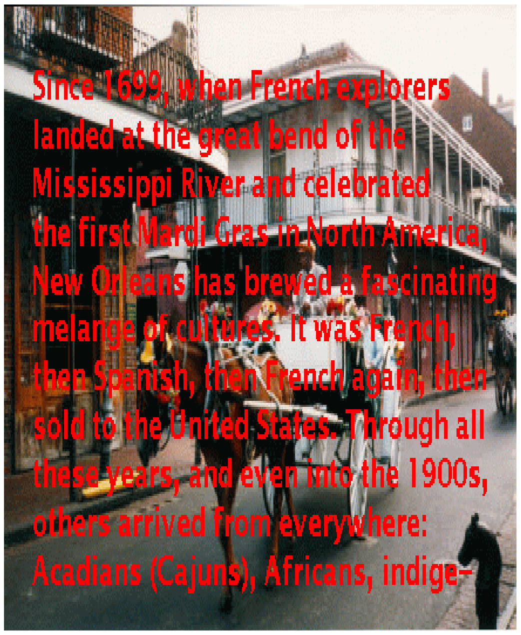
(a) (b)

(c)

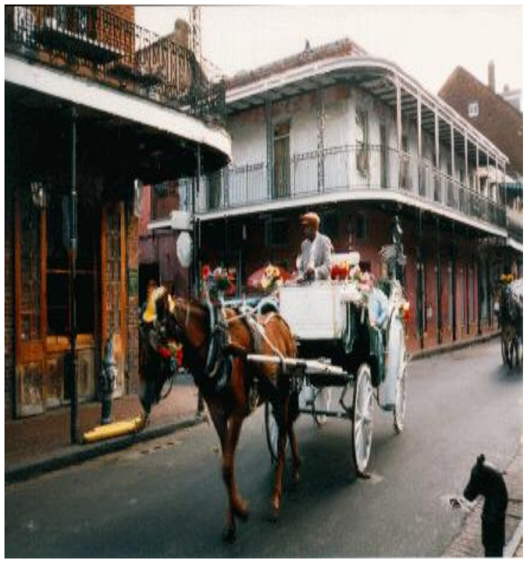
Figure 5.2 (a) original image of eye with scratches, (b) gray mask mask, (c) Inpainted image


Since 1699, when French explorers
landed at the great bend of the
Mississippi River and celebrated
the first Mardi Gras in North America,
New Orleans has brewed a fascinating
melange of cultures. It was French,
then Spanish, then French again, then
sold to the United States. Through all
these years, and even into the 1900s,
others arrived from everywhere:
Acadians (Cajuns), Africans, indige-

(a)

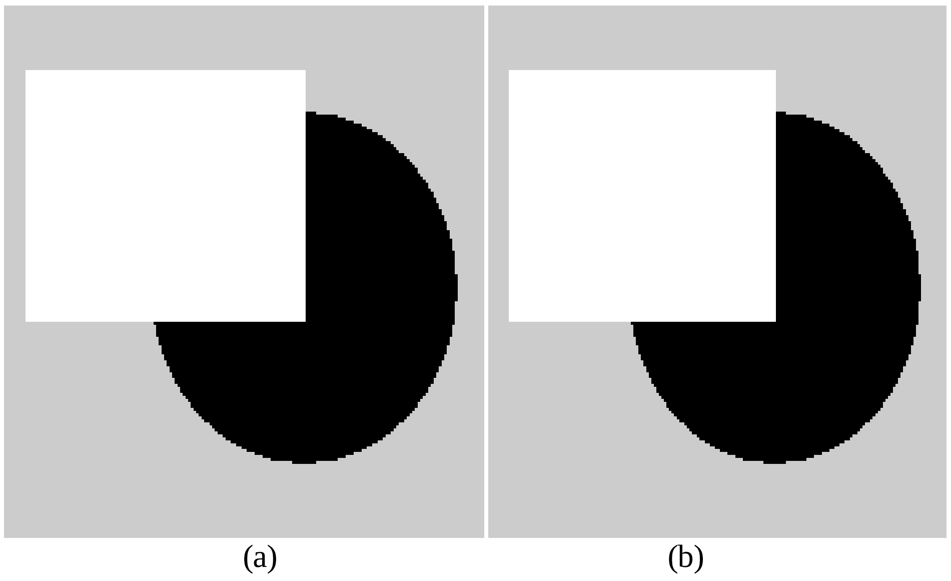

Since 1699, when French explorers
landed at the great bend of the
Mississippi River and celebrated
the first Mardi Gras in North America,
New Orleans has brewed a fascinating
melange of cultures. It was French,
then Spanish, then French again, then
sold to the United States. Through all
these years, and even into the 1900s,
others arrived from everywhere:
Acadians (Cajuns), Africans, indige-

(b)



(c)

Figure 5.4 (a)text on image newOrleans, (b) mask with red color area to be inpainted and (c) inpainted image



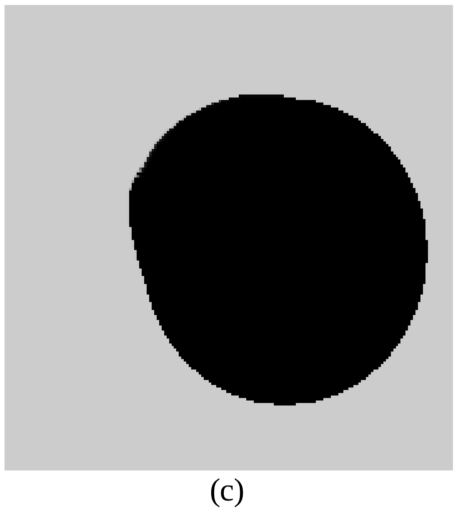


Figure 5.5 Circle image with missing area(a), mask defining to be filled region white region (b), inpainted image with filled occluded area

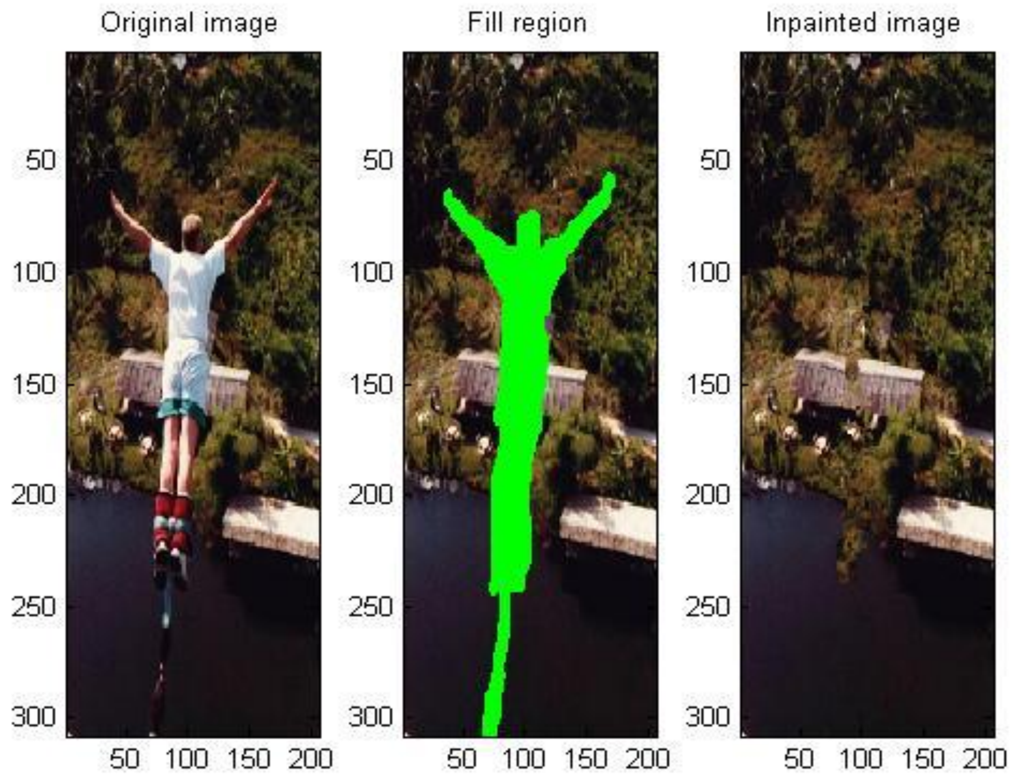


Figure 5.6 result of exemplar-based object removal

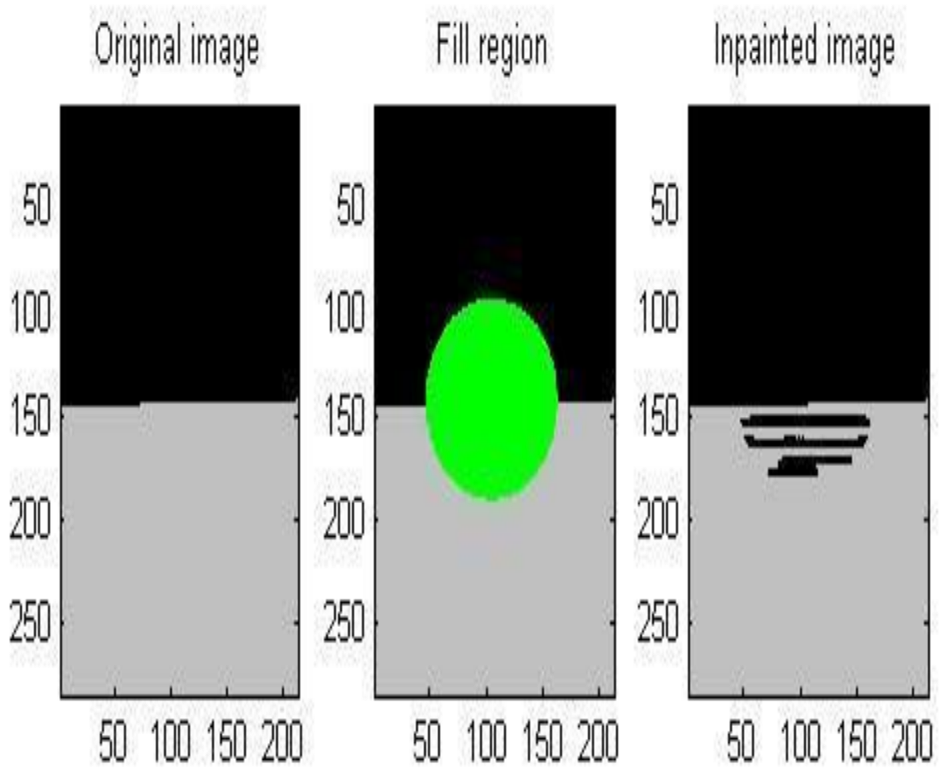


Figure 5.6 fill the arbitrary region in an image using exemplar algorithm

Chapter 6

Conclusions

Inpainting technique has various applications. There is a need for inpainting technique that automatically detects the area of defect in the image like scratches. The automatic area detection will reduce the human intervension in the process and make the process faster. All the algorithms presented detect either textural or structural information in the image. Algorithms are present which can process both the information but by dividing the image in separate partion defining blocks independent to each other. If synthesis of both textural or structural blocks in an image are done simultaneously inpainting can be fast and more accurate.

The future scope of the project is to find out the automatic mask generation for the images. Inpainting can be used for image compression but require to produce exact result at decoder side. The assistance information for the removed blocks should be provided to decoder for inpainting purpose helping the exact block to be inpainted by the sources available. The challenge is to find out the assistance information. Wavelet transform can be used for inpanting and image compression using inpaintinf techniques.

References

[1] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in Siggraph 2000, Computer Graphics Proceedings (K. Akeley, ed.), pp. 417-424, ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

[2] http://encyclopedia.jrank.org/articles/pages/6762/Image-Inpainting.html

[3] A. Criminisi, P. Perez, and K. Toyama, “Region filling and object removal by exemplar-based image inpainting," IEEE Transactions on Image Processing, vol. 13, pp. 1200-1212, 2004.

[4] A. A. Efros and T. K. Leung, “Texture synthesis by non-parametric sampling," in Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2, ICCV '99, (Washington, DC, USA),pp. 1033{, IEEE Computer Society, 1999.

[5] M. M. Oliveira, B. Bowen, R. Mckenna, and Y. sung Chang, “Fast digital image inpainting," in Proceedings of the International Conference on Visualization, Imaging and Image Processing (VIIP 2001), pp. 261-266, ACTA Press, 2001.

[6] P. P_erez, M. Gangnet, and A. Blake, “Poisson image editing," in ACM SIGGRAPH 2003 Papers, SIGGRAPH '03, (New York, NY, USA),pp. 313-318, ACM, 2003.

[7] A. Telea, “An image inpainting technique based on the fast marching method," Journal of Graphics, GPU, and Game Tools, vol. 9, no. 1, pp. 23-34, 2004.

[8] Dong Liu, Xiaoyan Sun, Feng Wu, ICME 2007, “Edge-Based Inpainting And Texture Synthesis For Image Compression”

[9] D. J. Heeger and J. R. Bergen, "Pyramid-based texture analysis/synthesis," in SIGGRAPH, pg. 229-238, 1995.

[10] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in SIGGRAPH 2001, Computer Graphics Proceedings (E. Fiume, ed.), pp. 341-346, ACM Press / ACM SIGGRAPH, 2001.

[11] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, "Simultaneous structure and texture image inpainting," IEEE Transactions on Image Processing, vol. 12, pp. 882-889, aug 2003.

[12] Criminisi, P. Pérez, and K. Toyama. "Object removal by exemplar-based inpainting." in Proc. Conf. Computer Vision and Pattern Recognition, Madison, WI, June 2003

[13] Xiaowei Shao, Zhengkai Liu, Houqiang Li. Department of Electronic Engineering and Information Science, University of Science and Technology:"An Image Inpainting Approach Based on the Poisson Equation"

[14] Xiaowei Shao, Zhengkai Liu, Houqiang Li. Department of Electronic Engineering and Information Science, University of Science and Technology: "An Image Inpainting Approach Based on thePoisson Equation"

[15] Howard Zhou, Mei Chen, Richard Gass, James M. Rehg, Laura Ferris, Jonhan Ho, Laura Drogowski "Feature-preserving Artifact Removal from Dermoscopy Images"

[16] Dong Liu, Xiaoyan Sun, Feng Wu, J. Vis. Commun. Image R. 23 (2012) 100–113 - "Inpainting with image patches for compression"