ISBN 978-93-5156-328-0
International Conference of Advance Research and Innovation (ICARI-2014)

Image Inpainting using Exemplar based, DCT and FMM Algorithm

Shivani Gaikar, Neha Khairnar [*], Nikita Rane, M. J. Surti
Department of Computer Engineering, SIEM, Nasik, India

Article Info

Article history:
Received 2 January 2014
Received in revised form
10 January 2014
Accepted 20 January 2014
Available online 1 February 2014

Keywords

Exemplar Based
Discrete Cosine Transform
Fast Marching Method

Abstract

Inpainting is the art of restoring lost parts of an image and reconstructing them based on the background information. This has to be done in an undetectable way. The main aim of our project is to remove the unwanted objects or unwanted data from the original image but this change is not noticeable by the user. Hence this will be done by using the three algorithms namely: Exampler Based Algorithm, DCT Based Algorithm, and Fast Marching Method. Digital Image Inpainting tries to imitate this process and perform the Inpainting automatically. Details that are hidden completely by the object to be removed cannot be recovered by any mathematical method. Therefore the objective for Image Inpainting is not to recover the original image, but to create some image that has a close resemblance with the original image. Such software has several uses. One use is in restoring photographs. Another use of Image Inpainting is in creating special effects by removing unwanted things from the image. Unwanted things may range from microphones, ropes, some unwanted person and logos, stamped dates and text etc. in the image. These parts can then be reconstructed using Image Inpainting. Natural images and photographs sometimes may contain stains or undesired objects covering significant portions of the images. The filling-in of missing information is very important in image processing, with applications including image coding and wireless image transmission (e.g.: recovering lost blocks) and image restoration (e.g.: scratch removal). Object removal from images is an image manipulation technique. The process of removing objects from images starts with mask out the undesired object, making the area where the object previously occupies a gap. Then the gap will be filled using graphical techniques. Among the graphical techniques that are used to fill the gap after object removal, two most commonly used are: Image Inpainting and texture synthesis. Exemplar based techniques and DCT based algorithm, which cheaply and effectively generate new texture by sampling and copying color values from the source. Exemplar based and DCT based algorithms is used for removing large objects from digital photographs and replacing them with visually possible backgrounds. The Fast Marching Method (FMM) is used to remove all scratches within the image.

1. Introduction

The aim of the project is to develop a tool to inpaint selected regions from an image. Inpainting is the art of restoring lost parts of an image and reconstructing them based on the background information. This has to be done in an undetectable way. The term inpainting is derived from the ancient art of restoring image by professional image restorers in museums etc. Digital Image Inpainting tries to imitate this process and perform the inpainting automatically. Details that are hidden/ occluded

Corresponding Author,
E-mail address: neha.khairnar07@gmail.com

completely by the object to be removed cannot be recovered by any mathematical method. Therefore the objective for image inpainting is not to recover the original image, but to create some image that has a close resemblance with the original image. Such software has several uses. One use is in restoring photographs. In fact, the term inpainting has been derived from the art of restoring deteriorating photographs and paintings by professional restorers in museums etc. Ages ago, people were already preserving their visual works carefully. With age, photographs get damage and scratched. Users can then use the software to remove the cracks from the photographs. Another use of image inpainting is in creating special effects by removing unwanted things from the image. Unwanted things may range from microphones, ropes, some unwanted person and logos, stamped dates and text etc. in the image. During the transmission of images over a network, there may be some parts of an image that are missing. These parts can then be reconstructed using image inpainting. There have been a few researches on how to use image inpainting for super-resolution and zooming of images Natural images and photographs sometimes may contain stains or undesired objects covering significant portions of the images. Region completion is a method to fill in such significant portions of an image by using the information from the remaining area of the image. The filling-in of missing information is very important in image processing, with applications including image coding and wireless image transmission (e.g.: recovering lost blocks) and image restoration (e.g.: scratch removal). Object removal from images is an image manipulation technique. The purpose of region completion varies from remove-undesired object to improve the quality of the image. The process of removing objects from images starts with mask out the undesired object, making the area where the object previously occupies a gap. Then the gap will be filled using graphical techniques. Among the graphical techniques that are used to fill the gap after object removal, two most commonly used are: image inpainting and texture synthesis. Image inpainting refers to the process of changing an image so that the change is not noticeable by an observer. It is usually applied to the task of restoring damaged images and is considered an art for the careful artist's eye it takes to get all the details right. The need to inpaint the image in an unobtrusive way extended naturally from paintings to photography and _lm. Texture synthesis approach is used to remove unwanted objects within the image. Texture synthesis is a way to fill image regions with pure textures repetitive 2-dimensional texture patterns with moderate stochasticity. Exemplar based techniques and DCT based algorithm, which cheaply and effectively generate new texture by sampling and copying color values from the source. Exemplar based and DCT based algorithms is used for removing large objects from digital photographs and replacing them with visually plausible backgrounds. The Fast Marching Method (FMM) is used to remove all scratches within the image. FMM algorithm is used for removing all scratches from digital photographs and replacing them with visually plausible backgrounds.

2. Literature Survey

2.1 Image Inpainting by Kriging Interpolation Technique

Kriging is a geostatistical interpolation method that takes into account both the distance and the degree of variation between known points when predicting values in unknown locations. Kriging is aiming to estimate unknown values at specific points in space by using data values from its surrounding regions Kriging predictions are treated as weighted linear combinations of the known locations. According to Kriging technique, the closer the input, the more positively correlated predictions. Now, let's bring the previously mentioned thoughts into digital image processing. According to, the pixels within the same k*k block are highly correlated, therefore; the application of Kriging inside the k*k block will yields high positively correlated predictions. Kriging gives weights for each point inside k*k block in accordance to its distance from the unknown value actually, these predictions treated as weighted linear combinations of the known values. The contamination may be thin scratch, thick scratch, text, bad areas generated by aging, or even unwanted objects that may be eliminated from the original image. These contaminated areas will be marked according to its corresponding mask. After that, the k*k block will be dispatched to Kriging interpolation technique to predict the contaminated areas using the accurate prediction feature of Kriging Image.

2.2 Inpainting Using Cloning Algorithm

In recovery from the damaged image cloning algorithms are used, in which the user specifies the co-ordinates for the area known as source domain. Damaged area is target domain. To interpolate the source domain with target domain seamlessly. Cloning algorithms are used. Steps of algorithms:

1. Identify the co-ordinate of target domain.
2. Specify the approximate co-ordinates of source domain which is to be used for recovery.
3. Mask source domain and select the region of interest.
4. Apply cloning algorithms namely poisson editing, Multiscale transformation method, Shepard's method to get the seamless inpainted image.

2.3 Hybrid Inpainting

Hybrid inpainting technique is also called as Image Completion. It is used for filling large target (missing) regions. And also preserves both structure and texture in a visually plausible manner. The hybrid approaches combine both texture synthesis and PDE based Inpainting for completing the holes. The main idea behind these approaches is that it decomposed the image into two separate parts, Structure region and texture regions. The corresponding decomposed regions are filled by edge propagating algorithms and texture synthesis techniques. These algorithms are computationally intensive unless the fill region is small One important direction we believe is more natural to the inpainting process is by structure completion through segmentation. This technique uses a two-step approach: the first stage is structure completion followed by texture synthesis. In the structure completion stage, segmentation, using the algorithm of, is performed based on the insouciant geometry, color and texture information on the input and then the partitioning boundaries are extrapolated to generate a complete segmentation for the input using tensor voting. The second step consists of synthesizing texture and color information in each segment, again using tensor voting.

2.4 Image Inpainting using TV based algorithm

Image inpainting algorithm based on TV model is essentially a weighted-average algorithm. The smaller the difference between target-pixel and neighborhood pixels is, the greater weight is, on the contrast, the greater the difference is, the smaller the weight is. Iteration is in fact a process of anisotropic information diffusion. Experiments show that there are more non damaged pixels around damaged pixel ,then the speed of the diffusion is faster, which means the converged speed is faster and inpainting is also faster. Thus, this paper try to divide the damaged region of image into a number of layers, and then doing the inpainting layer by layer. However, there is still a problem that there are some pixels whose non -damaged neighborhood-pixels is less in each layer. Therefore, it will slow down the diffusion speed of information and reduce the reliability. In order to solve this problem, a threshold TV was set, whose purpose is to filter these pixels that are placed in the next layer of the current layer. At that moment, there will be more available information around the pixel, so information diffusion will be faster and stronger. However, the traditional image inpainting method based on TV model iteratively inpaint damaged pixels line by line, and it did not treat damaged region as a whole, which will inevitably result in the available information around damaged region has not been maximized. In particular, when damaged region is relatively narrow and horizontal, the problem that information diffusion is slow is more permanent. Therefore, image inpainting requires more iteration.

3. System architecture

Architectural design involves identifying the software components, decoupling and decomposing then into processing modules and conceptual data structures and specifying the interconnections among the components. Software architecture alludes to the overall structure of the software and the ways in which that structure provide conceptual integrity for a system. In its simplest form, architecture is the hierarchical structure of the program components (modules), the manner in which these components interact, and the structure of the data that are used by the components. The primary objective of architectural design is to develop a modular program structure and represent the control relationships between modules. In addition, architectural design

melds program structure and data structure, defining interfaces that enable data to follow throughout the program. The Architectural design of this work is presented in the following specification.
Image to be filled is selected by the user. Find the edge by

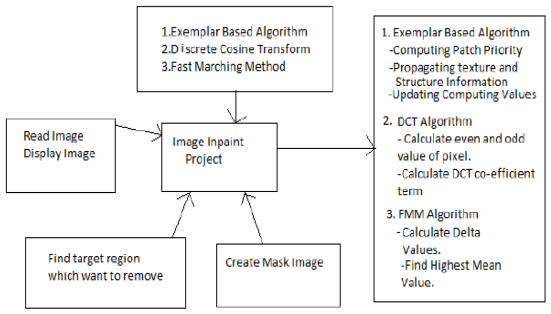
1.Exemplar Based Algorithm
2.D iscrete Cosine Transform
3.Fast Marching Method
Read Image
Display Image
Image Inpaint Project
Find target region which want to remove
Create Mask Image
1. Exemplar Based Algorithm
-Computing Patch Priority
-Propagating texture and Structure Information
-Updating Computing Values
2. DCT Algorithm
- Calculate even and odd value of pixel.
-Calculate DCT co-efficient term
3. FMM Algorithm
- Calculate Delta Values.
-Find Highest Mean Value.

Fig: 1. System Architecture

means of canny algorithm. Find the target region to be filled. Process each pixel and find the patch that is to be filled by exemplar inpainting method. Find the Confidential term and Data term. Data term is found by Gradient Vector and Orthogonal Vector. Find the quadrant coefficient term and fill the region by appropriate color.

4. Exemplar Based Algorithm

Exemplar-based approaches perform well for two-dimensional textures. But, in addition, that exemplar-based texture synthesis is sufficient for propagating extended linear image structures. This algorithm is proposed for removing large objects from digital images. exemplar-based texture synthesis contains the essential process required to replicate both texture and structure; the success of structure propagation, however, is highly dependent on the order in which the filling proceeds. As effective as these techniques are in replicating consistent texture, they have difficulty filling holes in photographs of real-world scenes, which often consist of linear structures and composite textures – multiple textures interacting spatially. The main problem is that boundaries between image regions are a complex product of mutual influences between different textures. In contrast to the two dimensional nature of pure textures, these boundaries form what might be considered more one-dimensional, or linear, image structures. Their drawback is that the diffusion process introduces some blur, which is noticeable when the algorithm is applied to fill larger regions. The algorithm presented here combines the strengths of both approaches. As with inpainting, we pay special attention to linear structures. But, linear structures abutting the target region only influence the fill order of what is at core an exemplar-based texture synthesis algorithm. The result is an algorithm that has the efficiency and qualitative performance of exemplar-based texture synthesis, but which also respects the image constraints imposed by surrounding linear structures. Our algorithm builds on very recent research along similar lines. The work in decomposes the original image into two components; one of which is processed by inpainting and the other by texture synthesis. The output image is the sum of the two processed components.

This approach still remains limited to the removal of small image gaps, however, as the diffusion process continues to blur the filled region. The automatic switching between “pure texture-” and “pure structure-mode” described in is also avoided. One of the first attempts to use exemplar-based synthesis specifically for object removal was by Harrison. There, the order in which a pixel in the target region is filled was dictated by the level of “text redness” of the pixel’s neighborhood1. Although the intuition is sound, strong linear structures were often overruled by nearby noise, minimizing the value of the extra computation.

A related technique drove the fill order by the local shape of the target region, but did not seek to explicitly propagate linear structure. Finally, Zalesny et al. describe an interesting algorithm for the parallel synthesis of composite textures. They devise a special-purpose solution for the interface between two textures. In this paper we show that, in fact, only one mechanism is sufficient for the synthesis of both pure and composite textures.

Exemplar-based synthesis suffices

The core of our algorithm is an isophote-driven image sampling process. It is well-understood that exemplar based approaches perform well for two-dimensional textures. But, we note in addition that exemplar based texture synthesis is sufficient for propagating extended linear image structures, as well.

A separate synthesis mechanism is not required for handling isophotes. For ease of comparison, we adopt notation similar to that used in the inpainting literature. The region to be filled, i.e., the target region is indicated by Ω, and its contour is denoted ∂Ω. The contour evolves inward as the algorithm progresses, and so we also refer to it as the "fill front". The source region ϕ, which remains fixed throughout the algorithm, provides samples used in the filling process. We now focus on a single iteration of the algorithm to show how structure and texture are adequately handled by exemplar-based synthesis. Suppose that the square template Ψp\epsilon Ω centred at the point p, is to be filled. The best-match sample from the source region comes from the patch (Ψ^)_qϕ, which is most similar to those parts that are already filled in Ψp. In the example in fig. 2b, we see that if Ψp lies on the continuation of an image edge, the most likely best matches will lie along the same (or a similarly coloured) edge. All that is required to propagate the isophote inwards is a simple transfer of the pattern from the best-match source patch. Notice that isophote orientation is automatically preserved. In the figure, despite the fact that the original edge is not orthogonal to the target contour ∂Ω, the propagated structure has maintained the same orientation as in the source region.

We now proceed with the details of our algorithm. First, a user selects a target region, Ω to be removed and filled. The source region,ϕ may be defined as the entire image minus the target region (ϕ=I-Ω), as a dilated band around the target region, or it may be manually specified by the user. Next, as with all exemplar-based texture synthesis , the size of the template window ϕ must be specified. We provide a default window size of 9χ9 pixels, but in practice require the user to set it to be slightly larger than the largest distinguishable texture element, or "texel", in the source region. Once these parameters are determined, the remainder of the region-filling process is completely automatic. In our algorithm, each pixel maintains a colour value (or "empty", if the pixel is unfilled) and a confidence value, which reflects our confidence in the pixel value, and which is frozen once a pixel has been filled. During the course of the algorithm, patches along the fill front are also given a temporary priority value, which determines the order in which they are filled. Then, our algorithm iterates the following three steps until all pixels have been filled:

1. Computing patch priorities

In this algorithm performs the synthesis task through a best-first filling strategy that depends entirely on the priority values that are assigned to each patch on the fill front. The priority computation is biased toward those patches which: (i) are on the continuation of strong edges and (ii) are surrounded by high-confidence pixels. Given a patch ΨP centered at the point p for some p ϵ dΩ . we define its priority P(p) as the product of two terms:

P(p) = C(p).D(p)

C(p) is the confidence term and D(p) is the data term, and they are defined as follows:

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

Where |ΨP | is the area of ΨP, α is a normalization factor (e.g., α = 255 for a typical grey-level image), np is a unit vector orthogonal to the front dΩ in the point p and $\perp$ denotes the orthogonal operator. The priority P(p) is computed for every border patch, with distinct patches for each pixel on the boundary of the target region. During initialization, the function C(p) is set to I-Ω The confidence term C(p) may be thought of as a measure of the amount of reliable information surrounding the pixel p. The intention is to fill first those patches which have more of their pixels already filled, with additional preference given to pixels that were filled early on (or that were never part of the target region). At a coarse level, the term C(p) of (1) approximately enforces the desirable concentric fill order. As filling proceeds, pixels in the outer layers of the target region will tend to be characterized by greater confidence values, and therefore be filled earlier; pixels in the centre of the target region will have lesser confidence values. The data term D(p) is a function of the strength of isophotes hitting the front dΩ at each iteration. This term boosts the priority of a patch that an isophote" flows" into. This factor is of fundamental importance in this algorithm because it encourages linear structures to be synthesized first, and, therefore propagated securely into the target region. Broken lines tend to connect, thus realizing the "Connectivity Principle" of vision psychology.

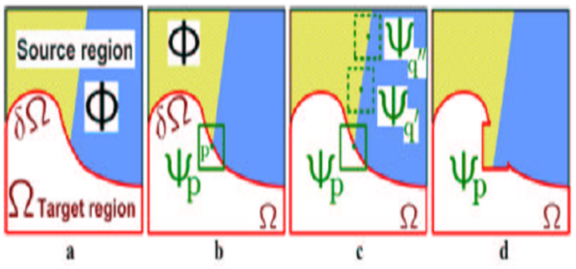

Fig: 2. Structure propagation by exemplar-based texture synthesis.

1. Original image, with the *target region* Ω, its contour ∂Ω and the *source region* ϕ clearly marked.
2. We want to synthesize the area delimited by the patch Ψp centred on the point p *p€∂*Ω.
3. The most likely candidate matches for Ψp lie along the boundary between the two textures in the source region, *e.g.*, Ψq' and Ψq''.
4. 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. See text for details.

2. Propagating texture and structure information

Once all priorities on the fill front have been computed, the patch ΨP with highest priority is found. We then fill it with data extracted from the source region Φ. In traditional inpainting techniques, pixel-value information is propagated via diffusion. As noted previously, diffusion necessarily leads to image smoothing, which results in blurry fill-in, especially of large regions. On the contrary, we propagate image texture by direct sampling of the source region. We search in the source region for that patch which is most similar to formally,

ψ_(q)ˆ=arg (min)⊤(φ_q\ε\ϕ) d(φ_pˆ,φ_q) ϕ

Where the distance d(ψa,ψb) between two generic patches and ψa and ψb is simply defined as the sum of squared differences (SSD) of the already filled pixels in the two patches Having found the source exemplar ψq, the value of each pixel-to-be-filled, p' ,|p'єψp ∩ Ω is copied from its corresponding position inside ψˆq. This suffices to achieve the propagation of both structure and texture information from the source Φ to the target region Ω, one patch at a time (cf., fig. 3.10d). In fact, we note that any further manipulation of the pixel values (e.g., adding noise, smoothing etc.) that does not explicitly depend upon statistics of the source region, is more likely to degrade visual similarity between the filled region and the source region, than to improve it.

3. Updating confidence values:

After the patch ψˆp has been filled with new pixel values, the confidence C(p) is updated in the area delimited by ψˆp as follows:

C(P)=C(Pˆ) ∀_(P)\ε φ(P ∩) ̂ψ

This simple update rule allows us to measure the relative confidence of patches on the fill front, without image-specific parameters. As filling proceeds, confidence values decay, indicating that we are less sure of the color values of pixels near the centre of the target region. A pseudo-code description of the algorithmic steps is shown in table I. The superscript t indicates the current iteration.

Example:

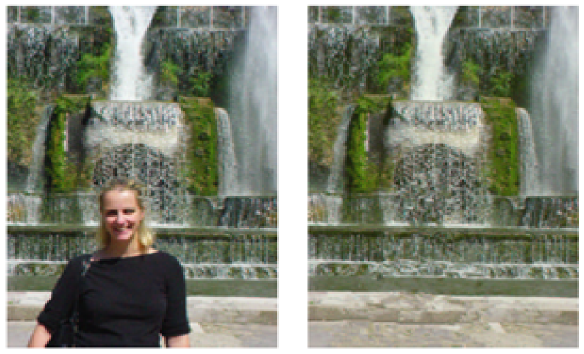

Fig: 3. Example of exemplar based algorithm

5. DCT (Discrete cosine transform) Algorithm

We propose an image inpainting optimization model whose objective function is a smoothed l^1 norm of the weighted no decimated discrete cosine transform (DCT) coefficients of the underlying image. By identifying the objective function of the proposed model as a sum of a differentiable term and a no differentiable term, we present a basic algorithm inspired by Beck and Teboulle's recent work on the model. Based on this basic algorithm, we propose an automatic way to determine the weights involved in the model and update them in each iteration. The DCT as an orthogonal transform is used in various applications. We view the rows of a DCT matrix as the filters associated with a multiresolution analysis. Nondecimated wavelet transforms with these filters are explored in order to analyze the images to be inpainted. Our numerical experiments verify that under the proposed framework, the filters from a DCT matrix demonstrate promise for the task of image inpainting.

Generally cartesian co-ordinate method is used to calculate frequency of color but in DCT polor co-ordinate system is used to calculate the frequency of color. The color patch is made up and compare with another pixels. The coefficient values of color pixels are calculated. For even co-ordinates the even coefficient values formula is used and for odd co-ordinates the odd coefficient value formula is used. Coefficient values can be calculated with FFT (Fast Fourier Transfer) Method.The object removal process is done more efficiently. It gives the maximum efficient output image as compare to Examplar based and FMM(Fast Marching Method). The time complexity of DCT algorith is (n^3)^2. we propose an inpainting method which could be viewed as a sort of combination of a variational approach and a sparse representation based approach. The proposed inpainting method is developed based upon an optimization model whose objective functional is a smoothed l^1 norm of the coefficients of the underlying image under a given redundant system.
The redundant system is generated from the discrete cosine transform (DCT) matrix of second type. More precisely, we identify the rows of a given DCT matrix as the filters associated with a multiresolution analysis which is referred to as the DCT-Haar wavelet system in the following discussion. The non-decimated transform using the DCT-Haar wavelet filters yields a redundant system that is used in our model. We show that this redundant system incorporated with the proposed model performs particularly well for inpainting images with incomplete information. We propose an adaptive inpainting algorithm to solve the proposed model. We also propose a way to automatically tune the parameters appearing in our algorithm. Our inpainting model is formulated as an optimization problem in which the variational objective functional has a regularization term formed by a sparse representation of the underlying image. We point out that the proposed optimization model is connected with the reweighted l^1 minimization model, but with several distinct and promising properties. We begin with introducing some notation used in the following. Let forg the original image be defined on the domain Ω = {1, 2 . . . n} and a nonempty proper subset D of Ω be given. The observed image g is modeled as
Where h[k] with k \ε

D could represent any types of degradations to the original image including impulsive noise and texts superposed on matrix, denoted by PD, whose k-th diagonal entry is 1 if k \epsilon _\D and 0 if k \epsilon D. The goal of image inpainting is to seek an image f such that PD f = PDg while f can truthfully retain original information of f_org. f_org. Associated with the sets Ω and D, we define an n × n diagonal

Let us consider an image given below We will consider patch A and patch B .Patch is the patch from the whole original image and patch B is the patch from the object which we want to remove. Now our aim is to replace the patch B with the patch A. The every patch is in the form of 9*9 matrixes. To remove the object from the given picture, the even value and the odd value of the patch is calculated. The even value and the odd value is calculated using following formula;

Where k is the value given by the user which may vary and N is the total size of matrix (row*column) Even values/coefficients are calculated by using the term ,And the odd values / coefficients are calculated using the term , By using the above formula for each patch , 9*9 matrix is made i.e. patch A will have is own matrix and patch B will have its own matrix .Now for each of the matrix i.e. A and B , DCT coefficient is calculated. The DCT Coefficient matrix is represented by Aˈ and Bˈ. After finding the coefficient matrix i.e. Aˈand Bˈ matrix, we will have to find the difference between the above two matrix i.e. Aˈ- Bˈ. If the difference between the matrixes is minimum, then the patch is replaced by the other patch, in our example patch B will be replaced by the patch A.

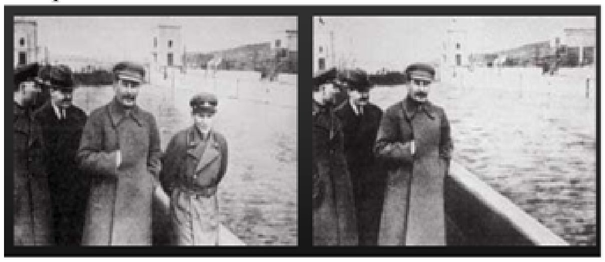
Fig: 4. Image Inpainting using DCT

6. Fast Marching Method

Digital inpainting, the technique of reconstructing small damaged portions of an image, has received considerable attention in recent years. Digital inpainting serves a wide range of applications, such as removing text and logos from still images or videos, reconstructing scans of deteriorated images by removing scratches or stains, or creating artistic effects. Most inpainting methods work as follows. First, the image regions to be inpainted are selected, usually manually. Next, color information is propagated inward from the region a boundary, i.e., the known image information is used to fill in the missing areas. In order to produce a perceptually plausible reconstruction, an inpainting technique should attempt to continue the isophotes (lines of equal gray value) as smoothly as possible inside the reconstructed region. In other words, the missing region should be inpainted so that the inpainted gray value and gradient extrapolate the gray value and gradient outside this region. Several inpainting methods are based on the above ideas. In the image smoothness information, estimated by the image Laplacian, is propagated along the isophotes directions, estimated by the image gradient rotated 90 degrees. The Total Variational (TV) model uses an Euler-Lagrange equation coupled with anisotropic diffusion to maintain the isophotes' directions. The Curvature-Driven Diffusion (CCD) model enhances the TV method to drive diffusion along the isophotes' directions and thus allows inpainting of thicker regions. All above methods essentially solve a Partial Differential Equation (PDE) that describes the color propagation inside the missing region, subject to various heuristics that attempt to preserve the isophotes' directions. Preserving isophotes is, however desirable, never perfectly attained in practice. The main problem is that both isophote estimation and information propagation are subject to numerical diffusion. Diffusion is desirable as it stabilizes the PDEs to be solved, but leads inevitably to a cetain amount of blurring of the inpainted area. A second type of methods repeatedly convolves a simple 3 × 3 filter over the missing regions to diffuse known image information to the missing pixels. However impressive, the above methods have several drawbacks that preclude their use in practice. The PDE-based methods require implementing nontrivial iterative numerical methods and techniques, such as anisotropic diffusion and multiresolution schemes. Little or no information is given on practical implementation details such as various thresholds or discretization methods, although some steps are mentioned as numerically unstable. Moreover, such methods are quite slow, e.g., a few minutes for the relatively small inpainting region shown in Figure 1. In contrast, the convolution-based method described in is fast and simple to implement. However, this method has no provisions for preserving the isophotes' directions. High-gradient image areas must be selected manually before inpainting and treated separately so as not to be blurred. We propose a new inpainting algorithm based on propagating an image smoothness estimator along the image gradient, similar to . We estimate the image smoothness as a weighted average over a known image neighborhood of the pixel to inpaint. We treat the missing regions as level sets and use the fast marching method (FMM) described in to propagate the image information. Our approach has several advantages:

• it is very simple to implement (the complete pseudocode is given here);

• it is considerably faster than other inpainting methods–processing an 800×600 image (Figure 1) takes under three seconds on a 800 MHz PC;

• it produces very similar results as compared to the other methods it can easily be customized to use different local inpainting strategies.

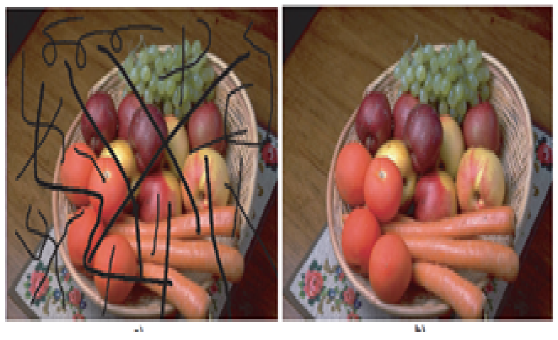

Fig: 5. Image inpainting using FMM

References

[1] Alexandru Telea, Eindhoven University of technology, “An Image Inpainting Technique based on Fast Marching Method “, 2003
[2] Yan-Ran, Lixin Shen and Bruce W. Suter, Senior Member, “Adaptive inpainting Algorithm Based on DCT”, IEEE 22(2), February 2013
[3] A.Criminisi, P.Perez, K.Toyama,”Object Removal by Exemplar Based In Painting”, IEEE 2003
[4] R. Monneare, “Introduction to Fast Marching Method”, June 4 2011
[5] Richard Woods & Rafel Gonzalez, Handbook of Digital image processing, 3
[6] B.Chanda, D. mujumdar, Handbook of Digital image processing and Analysis