Project Report

Image Inpainting

05 Dec, 2011

Aadish Gupta
2009001
Sanchit Garg
2009041

Introduction

Image inpainting is the technique of reconstructing missing or damaged portions of images such that a user who has not seen the original image is not able to point out the changes. It has a wide range of applications and can be used in restoration of damaged photographs, films and paintings and also helps in removing superimposed text, stamps, publicity (logos) and other objects from images and films. It also helps in producing special effects in films.

In this project we have tried to restore still images using information from neighboring pixels. We have implemented two different algorithms for this. We also tried automatic text detection in images so that superimposed text could be removed without any user intervention.

Background study

Image inpainting is a technique as old as art itself. Traditionally image inpainting was done manually by skilled artists but given its wide range of applications many different algorithms have been proposed for it in the past few years. Now it is a standard feature in many image processing libraries and tools like Photoshop. Some of the techniques proposed are based on partial differential equations[1], combining global frequency and spatial information, convolution with filters[2], Gaussian processes[3], disocclusion algorithms[1], etc. Techniques for texture synthesis have also been proposed for digital inpainting. Most of the algorithms require user intervention as the portion to be inpainted needs to be specified to the algorithm by creating a mask for the image.

Methodology

In this project we have implemented two different algorithms:

a) Fast digital inpainting algorithm using 3x3 filters [2]
b) Image inpainting based on fast marching method [4]

We then did automatic text detection in images so that the user need not specify the mask for removing superimposed text.

Fast Digital inpainting algorithm [2]

In this algorithm the image to be inpainted is iteratively convolved with a 3x3 filter. The user needs to create a mask for the portion which needs to be restored. Using this mask the area to be inpainted is initialized first and then repeatedly convolved with a filter. After applying the filter iterative for hundred times very good results were observed. The filter used was a weighted average Gaussian kernel that considers only the neighborhood pixels and has zero weight at the center.

It was observed that after applying this filter some blurring was produced in areas with high contrast edges. For removing this blurring effect diffusion barriers need to be created by the user at these areas. These diffusion barriers are two pixels in width and act as boundaries for the diffusion process inside the area to be inpainted and help in reconnecting the edges.
We implemented this algorithm differently for object removal and text removal. In object removal the user specifies the mask by selecting the area to be inpainted using Ms Paint which is called by the program itself. After obtaining this image from the user we created a mask by finding the differences between this image and the image to be inpainted. Selecting the area to be inpainted is the most time consuming step in this whole inpainting process.

In case of text removal the text was automatically detected and then the filter was applied only on the area which contained the text. This process required no user intervention and hence no time was spent on creating the mask.

Image inpainting based on fast marching method[4]

Fast Marching Method Technique

Fast marching method is used to propagate the boundary of the selected region inside the selected region (area to be inpainted) in the order of the distance of the points from the initial points. This method is continued until the complete area to be inpainted is not filled. By using this method it is ensured that the points which are near to the known pixels are filled first and then this method is continued iteratively to cover all pixel points. This method gives us the point on which we need to apply our algorithm.

FMM Initialization

At the beginning all the points outside the selected region and on the boundary are initialized to 0 and the points inside the region are set to a large value.

Flags are assigned to each point on the basis whether they are on the boundary or outside or inside the boundary of the selected region.

We need to store the points lying on the boundary in a data structure which returns the value of the pixel with minimum distance from the boundary. In our implementation we have used a combination of three linked lists to perform this task.

FMM Algorithm

We take the first point from our data structure and find the flag for this point. For all the horizontal and diagonal neighbors of this point we find their minimum distance and pass these points to the inpainting algorithm. Then all the points are added to the boundary list and fetched according to their distance.

To find the distance we take the neighbors of the point and calculate their distance if they are in the known region and thus find the minimum distance from all known neighbors.

Inpainting Algorithm

Any Inpainting algorithm can be applied with this methodology as it just passes the point which needs to be inpainted and makes the inpainting algorithm independent unit.

Here we use the small neighborhood of the point to determine its value. First we find the gradient of the whole image. We have used sobel operator for this purpose. We find the sobel gradient in both horizontal (x) and vertical (y) direction and then combine both of them. The Y/X value of the sobel operator gives us the direction which is used in calculating the weight which is assigned to each neighbor while adding their value to the unknown pixel and this is how inpainting is done.

Text detection

We tried two different techniques for detecting the location of text in images.

Using morphological operations[5]:

We used a circle as the structuring element of size 3 units. The structuring element was found by hit and trial method. We used various structuring elements and chose the one which gave the best results. We performed the closing operation on the image and subtracted it from the original image. Then we used a thresholding function on this image to separate the text from the background image. This technique works effectively when the intensity of the superimposed text is quite higher than that of the background. Then we used a smoothing filter to remove unwanted background information from the obtained image and finally dilated the image so that the disjoint text pixels could be connected.

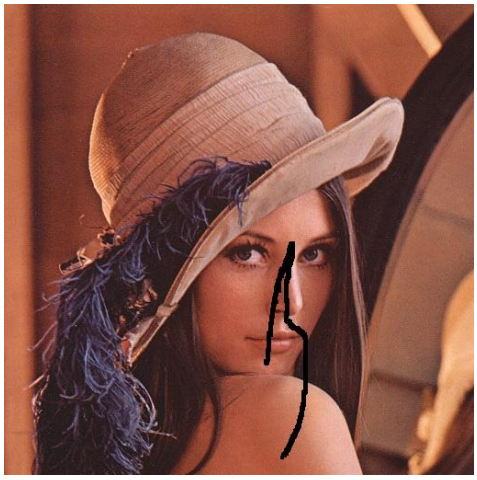
Using wavelets [6]:

We tried another technique to extract the text by finding the Haar wavelet transform of the image but were unable to implement the whole technique. In this process the three color components of the image are converted into the intensity component and then the image is passed though a weighted median filter to remove the noise. Then the 2-D Haar wavelet transform of this image

is found and the image is decomposed to four sub-images. Then sobel edge detector is used to extract the strong text edges from each sub-image. Using these text images an edge map is formed and morphological dilation operation is applied. Then the non-text region is removed by using various techniques to finally get the text. In this project we could not implement the whole algorithm due to time limitations and were limited to performing the wavelet transform only. So we could not use this approach in determining the location of text.

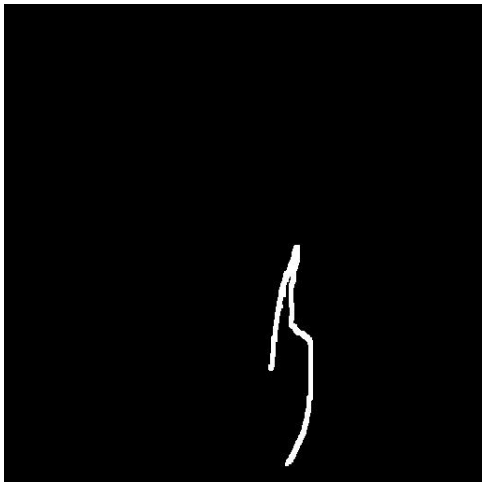
Results

Using Fast Digital Image Inpainting (Object removal)

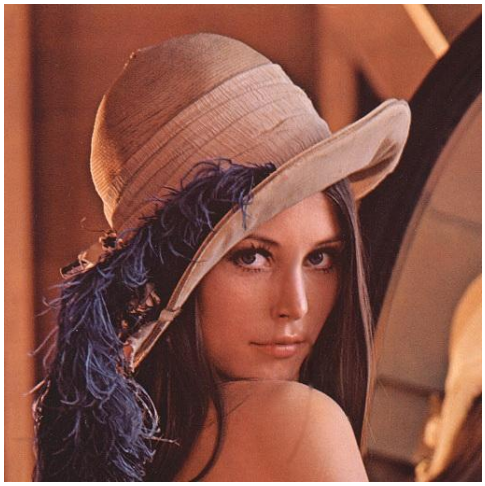


Damaged Image

Region selected by user



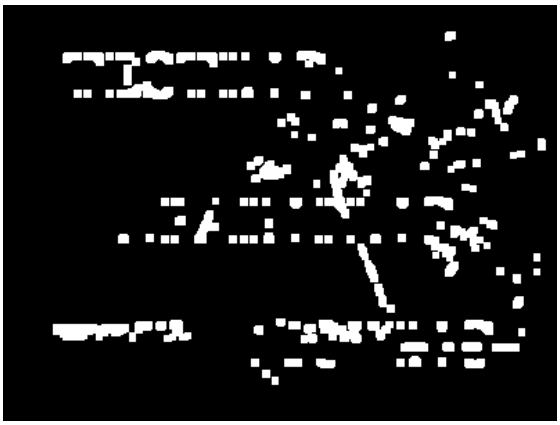
Mask created by program



Final restored image

Text Removal

Image with superimposed text



Mask that was created automatically

Final Result

Image inpainting based on fast marching method

Input image

Restored Image

Conclusions and Future Work

We were able to implement the fast digital image inpainting algorithm completely. We were getting results similar to those given in the paper. We also tried automatic text detection and were able to detect the super imposed text partially and restore the original image thus reducing user interaction.

We implemented fast marching inpainting technique partially and it was working for very small objects or thin lines.

We would like to continue our work on automatic text detection. We would further like to extend it to removing other superimposed lines and logos etc. This could be done if there is high intensity variation in the background and the superimposed information. This would help in minimizing user interaction there by reducing total time taken to inpaint and would also lead to better results as it is very difficult to create the exact masks manually. We would also like to work on inpainting using Gaussian processes[3] as it is a highly effective technique and can be used to remove large objects from the image with very high quality results. Texture synthesis can also be combined with various inpainting algorithms to give high quality results[7].

References

1. Image Inpainting, Bertalmio, Sapiro, Caselles, Ballester
2. Fast Digital Image Inpainting, Manuel M. Oliveira, Brian Bowen, Richard McKenna, Yu-Sung Chang
3. Image Inpainting with Gaussian Processes, Alfredo Kalaitzis
4. An Image Inpainting Technique Based on the Fast Marching Method, Alexandru Telea
5. Text Detection from Natural Scene Images: Towards a System for Visually Impaired Persons, Nobuo Ezaki, Marius Bulacu, Lambert Schomaker
6. Text segmentation from images with textured and colored background, Roshanak Farhoodi and Shohreh Kasaei
7. Simultaneous Structure and Texture Image Inpainting, M. Bertalmio, L. Vese, G. Sapiro, and S. Osher