Image Inpainting

Final Project Evaluation

Team Name - DIP Project
Team Number - 24
Team Members - Shreya Gupta(201531123)
Naila Fatima(201530154)
Naga Sunethra Vysyaraju (201530072)

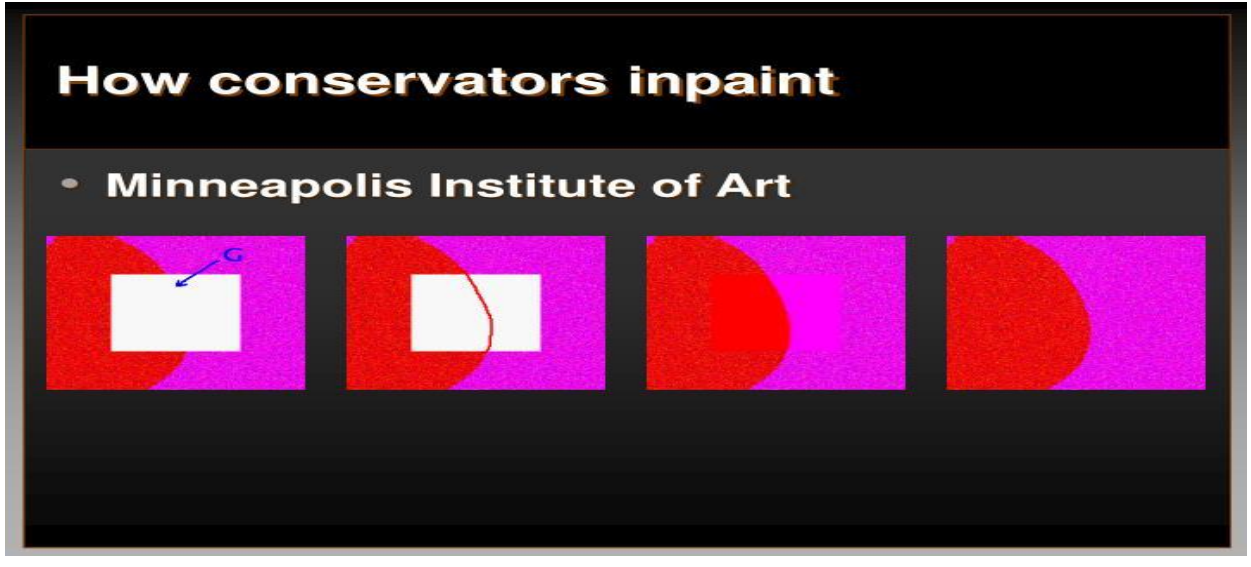
What is image inpainting?

Image inpainting is a method to reconstruct missing or damaged portions of images. This is used in order to fill in the cracks and scratches in an image as well as remove elements or text. This technique is usually used by museum curators in order to repair damaged paintings.

Image inpainting is also used to some extent in image compression (as certain formats like JPG often lose blocks of information which can be recovered by inpainting) as well as transmission of images.

The curator’s approach to inpainting (Most basic)

1) Given the region to inpaint, the boundaries are first constructed within the region.
2) The colors are then filled in the inpainted region.
3) Texture is added.



Our Approach (1)

We will be using the algorithm provided in the paper *Fast Digital Image Inpainting* by *Oliviera* and *Bowen*.

The algorithm defines a region R to be inpainted which will be user-defined as a mask for that particular region will be provided as input. The region R in the image will be initialized to 0 (removing its color information). The boundary dR of region R will be of size one pixel and this will be used to propagate the information inside R. The region R will be convolved with a diffusion kernel (we will be using a Gaussian kernel) for multiple iterations.

The number of iterations used can be fixed (the authors of the paper kept it at 100 and we did the same) or we can provide a threshold such that the inpainting process will be stopped if the change in intensities for the image is less than the threshold provided.

Our Approach (2)

We can notice that we will be using an isotropic process to inpaint the region R. However, this will lead to blurring across high contrast edges. In order to eliminate this, we can use either anisotropic diffusion or diffusion barriers (which are a user input) which will act as boundaries for the diffusion process inside the region R. We can see that this will give the same result as using anisotropic diffusion along edges but without the complexity. As the diffusion process reaches a diffusion barrier, the pixel has its colour set and the process is stopped. By adding the diffusion barriers, the user stops the diffusion process from mixing information on both sides. This paper recommends us to use diffusion barriers of size 2 pixels.

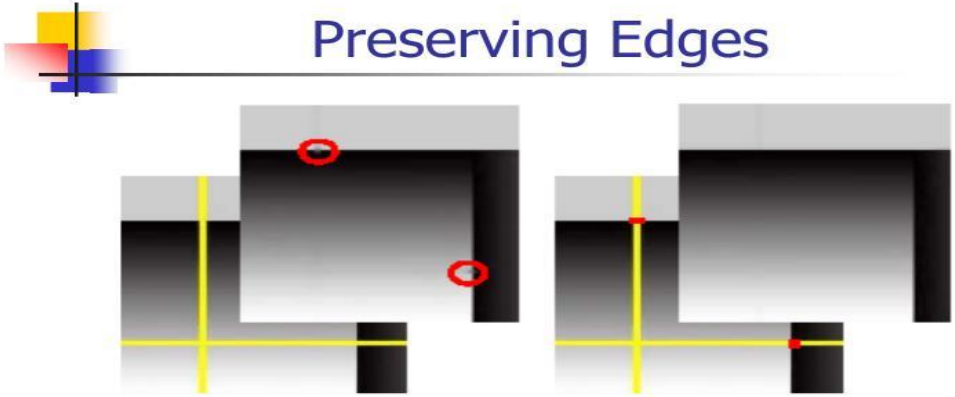
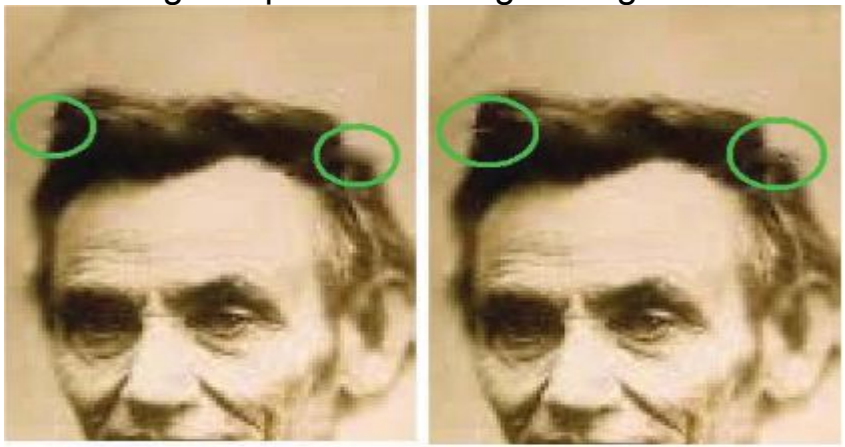


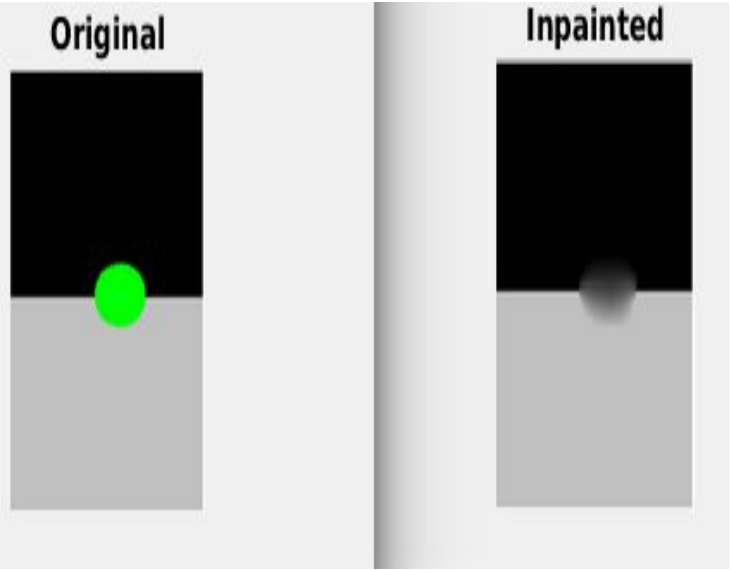
Fig. 3. The crossing lines define the inpainting domain (back left). Result of the isotropic diffusion introduces some blurring along high contrast edges (top left). User-added diffusion barriers (back right). Result produced with diffusion barriers (front right).

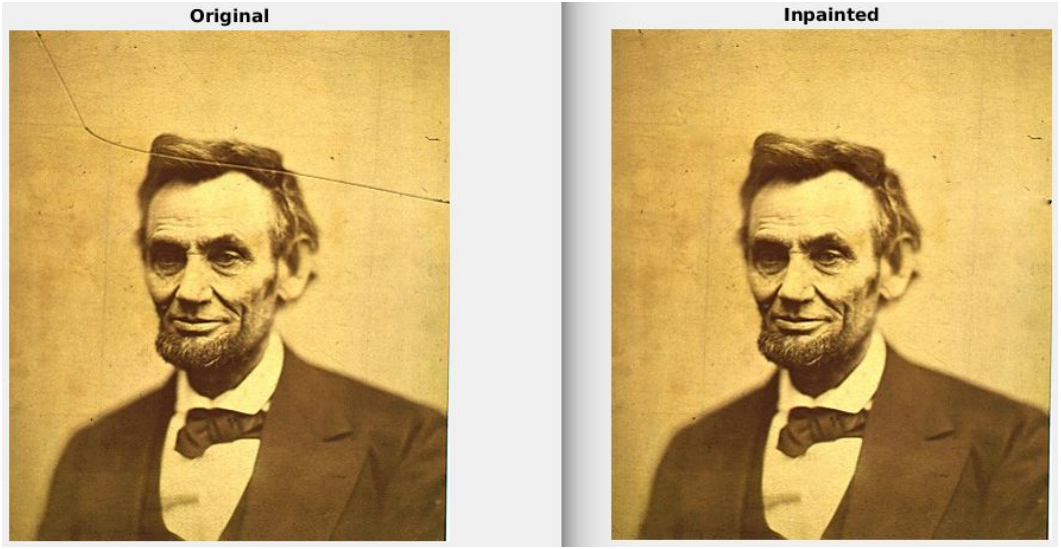
Diffusion Barriers

Diffusion barriers are effective at preventing blurred regions in resulting images due to intersections between the region to be inpainted and high-contrast edges. In the paper we are using, size of diffusion barriers is given to be equal to 2 pixels. A two-pixel wide barrier is not sufficient for our program since there would still be information spilling over edges and thus causing blurring. Having a white,linear diffusion barrier five pixels wide is ideal because the program associates a pixel value of one, or white, with a part of the image to be preserved and since we are taking value from five pixels so the edges will not be blurred. We can observe that left image has more blurring compared to the right image which is using diffusion barrier of 5 pixels.



Results Obtained (without diffusion barriers) (1)





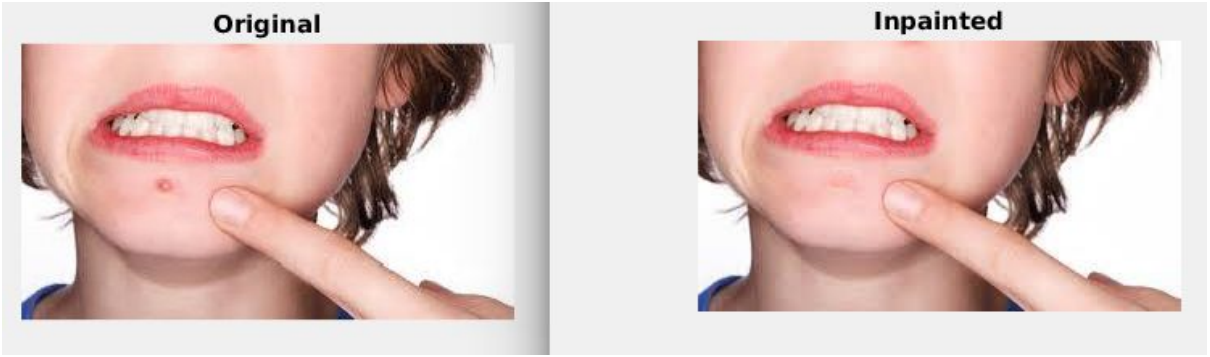
Results (without diffusion barriers) ...(2)

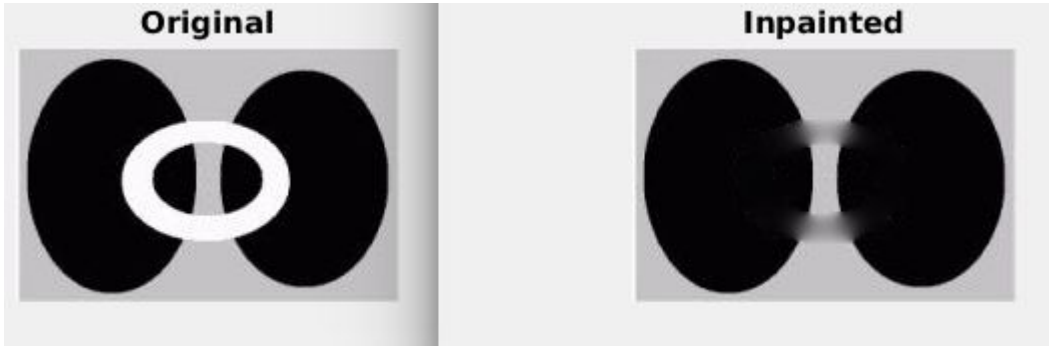
Results obtained (without diffusion barriers) …(3)

Mask used for Previous Image -

Results obtained (without diffusion barriers) …(4)

Results obtained (without diffusion barriers) …(5)



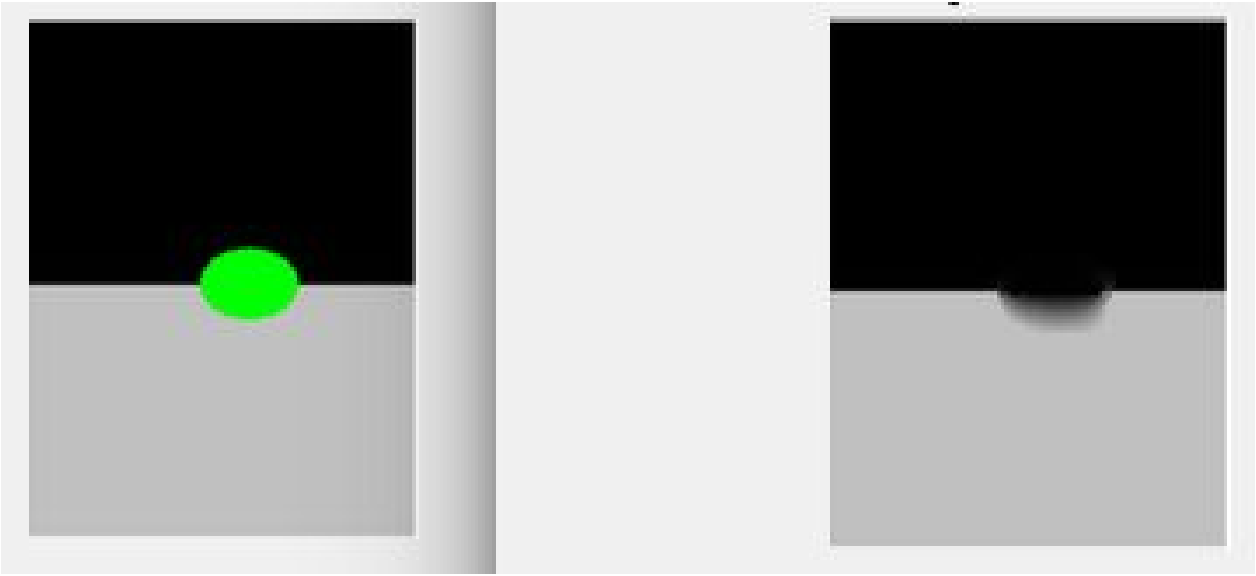


Results Obtained without Diffusion Barriers (6)

Results Obtained without Diffusion Barriers (7)

Results Obtained with diffusion barriers (1)

Notice the sharp edge between the black and gray portions which was not present in the absence of diffusion barriers (slide 7).

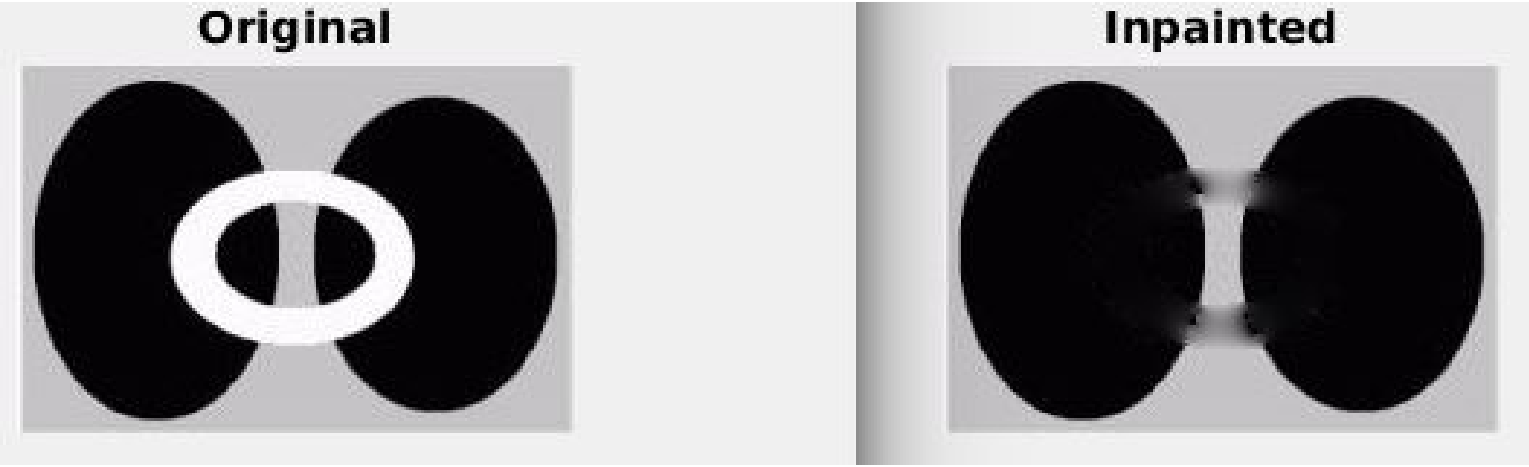


Results obtained with diffusion barriers (2)

Edge present at the top most right corner since we have only used one diffusion point over there.

Results obtained with diffusion barriers (3)

Note that we can observe that the edges of the black circles have been reconstructed.



Model 2: PDE Model (1)

The equation that we will be using in order to fill in the region R is:

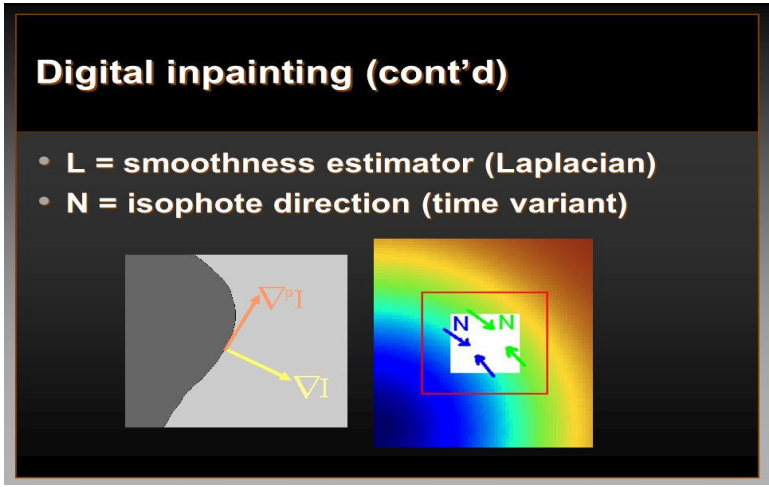
$$dI(i,j)/dt = dL(i,j) \cdot N(i,j) \quad \textit{Equation 1}$$

where L is the information we want to propagate into R, N is the direction in which we want to propagate the information and dL is the change in L.

We can easily note that dL.N (dot product) will be the change in L in the direction of N. Ideally, we want this dot product to be 0 as this then means that L is constant in the direction of propagation. If L is not constant, this leads to a jump in intensities which we may be able to perceive. Since we want the dot product of dL and N to be zero, this is equivalent to saying that we want dL to be perpendicular to N.

Model 2: PDE Model (2)

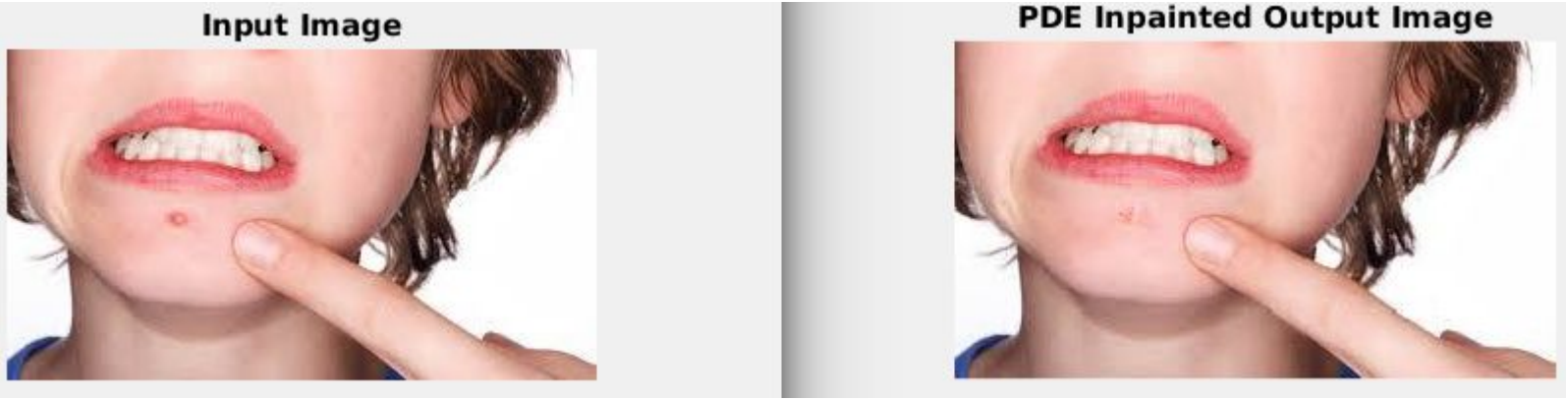
We want to achieve steady state such that there is no change. We do this by making the image change in time according to *Equation 1* in previous slide. We use L to be a smoothness estimator and use the Laplacian (second partial derivative, Ixx + Iyy) of the image. N will give us the isophote direction. Isophote is a curve on an illuminated surface that connects points of equal brightness.



Note that the edge direction is perpendicular to the gradient of edge. Also, N will be perpendicular to the gradient and therefore in the direction of the edge.

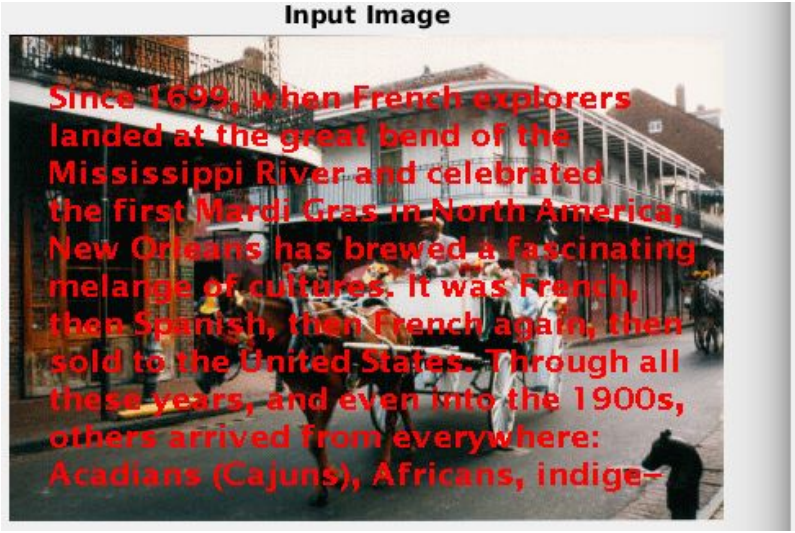
We cannot use the first derivative as L since then L will be dI (gradient) and N will be perpendicular to the gradient so in this case L and N will *always* be perpendicular. Note that a color image must be broken into RGB channels for the method to work.

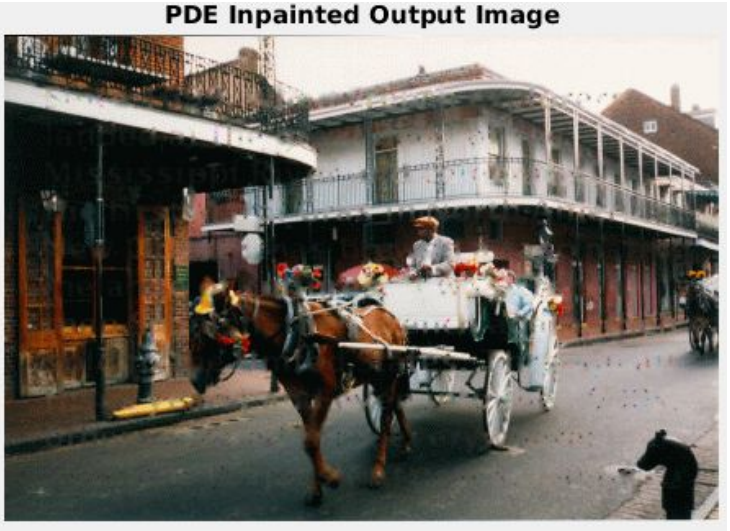
Results Obtained with PDE Algorithm (1)



Results Obtained with PDE Algorithm (2)

Results Obtained with PDE Algorithm (3)

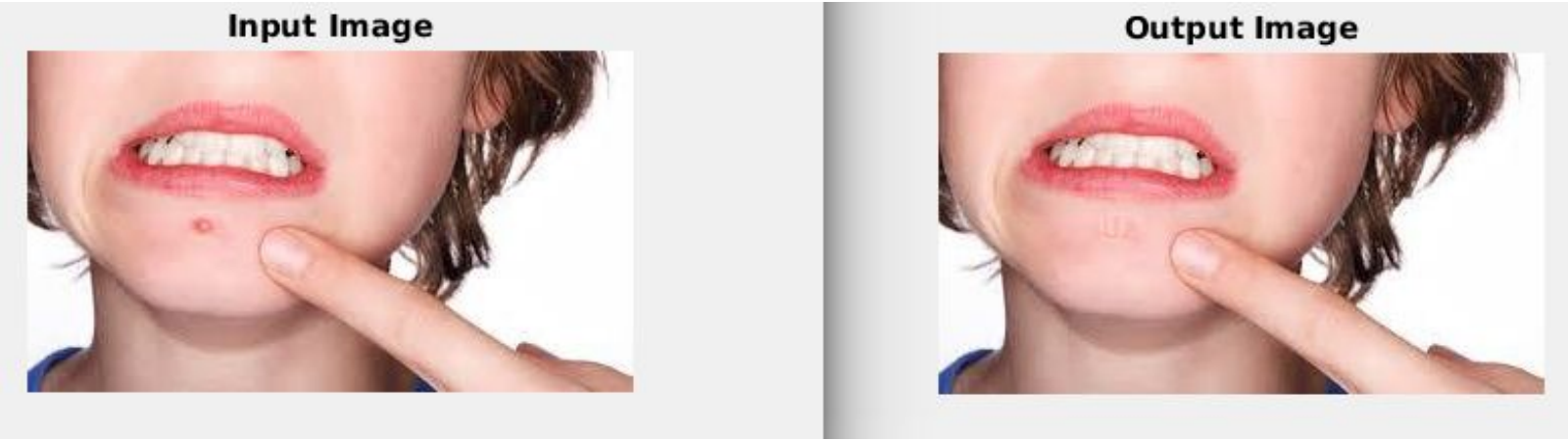




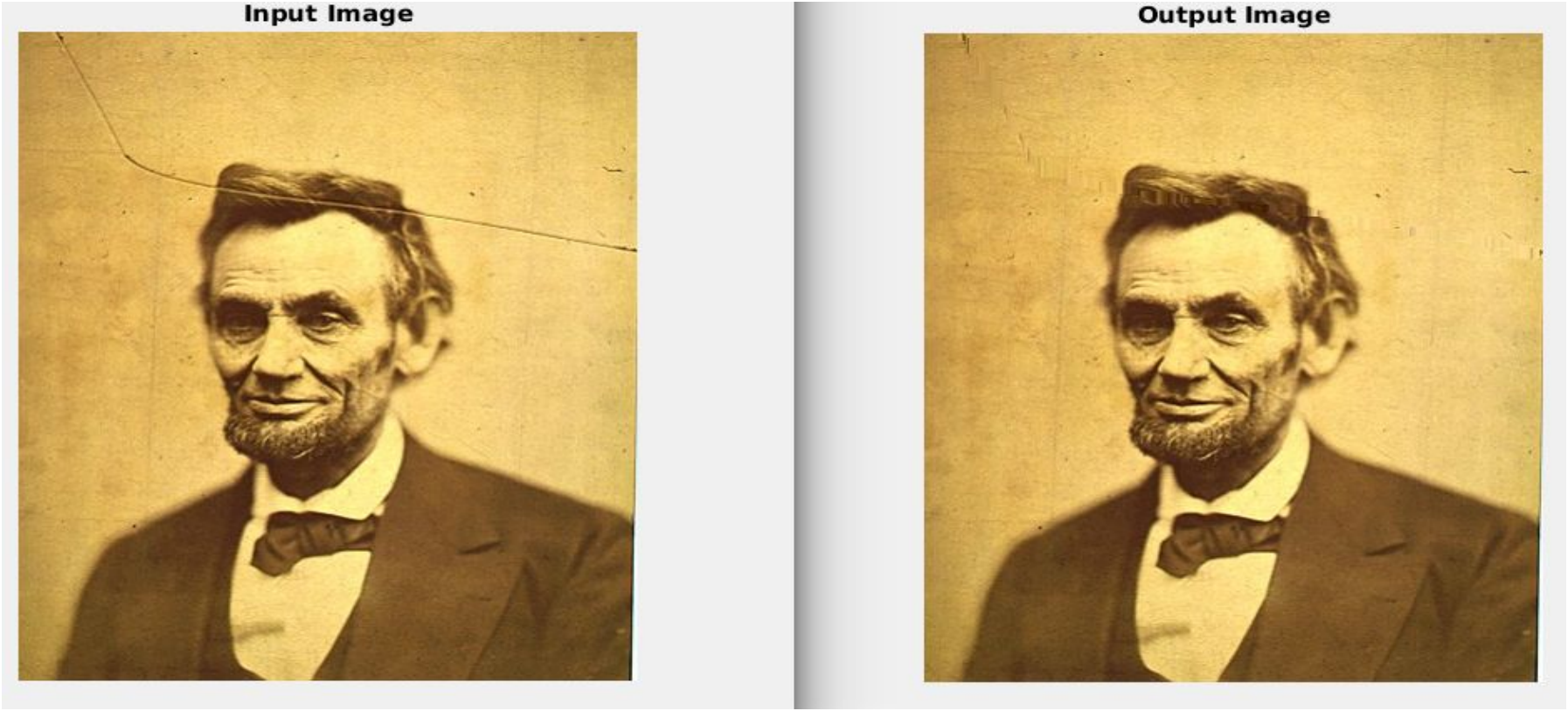
Nearest Neighbour Algorithm

In this Algorithm,we consider the intensity of nearest neighbour of a pixel for inpainting the damaged portions in the Image.

Result obtained using nearest neighbour algorithm



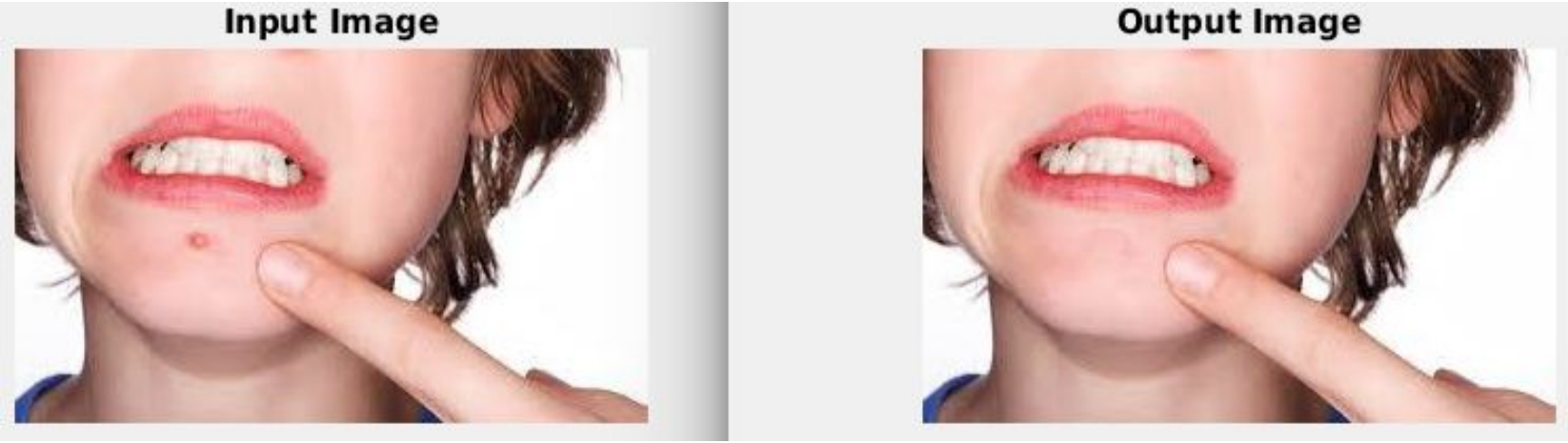
Results obtained in the nearest neighbour Algorithm



Median Diffusion Algorithm for Inpainting

In this algorithm ,we consider a pixel in the inpainting region and find the median of the neighbourhood values of the pixel and replace the intensity value of the pixel with the median pixel intensity.This fills the damaged portions of the Image.

Results obtained in the Median Algorithm:-



Observations:

We use four algorithms (Gaussian Average, PDE, Nearest Neighbour and Median Diffusion). We observed that our main algorithm gives better outputs compared to other algos. Median Diffusion and Nearest Neighbour will work only if the surroundings are smooth.

For main algorithm, Diffusion Barriers are used for getting better edges in the restored image.

Applications

1. Restoring damaged photographs, films and paintings.
2. Removal of occlusions, such as text, subtitles, stamps and publicity from images.
3. To produce special effects.
4. To remove some objects from the images (like in Fig. A removing insect from the image).
5. To recover pixels lost while compression and transmission of images.

Image Compression

Input Image

Image Compression

Output Image

Limitations

These algorithms cannot be used if the region to be inpainted is large and if it includes high contrast edges and complex background.