MID TERM REPORT

**On**

**OBJECT-REMOVAL**

**Submitted by**

**RITESH KUMAR SINGH/PREETI PANDEY**

**Roll No: 171500262/171500237**

**PROJECT GUIDE**

**PIYUSH VASHISTHA**

Department of Computer Engineering & Applications

**Institute of Engineering & Technology**



**GLA University**

**Mathura- 281406, INDIA**

**2019**

# DELARATION

We hereby declare that the work presented in this project report entitled “**Object Removal**”, submitted towards completion of mini project in Sixth semester of B.Tech. GLA University, Mathura, is an authenticated record of our original work carried out by us under the guidance of Mr. Piyush Vashistha. Due acknowledgements have been made in the text to all other material used. The project was done in full compliance with the requirements and constraints of the prescribed curriculum.

Place: Mathura Ritesh Kumar Singh & Preeti Pandey

171500262 & 171500237

# ACKNOWLEDGEMENTS

We express our deepest gratitude to our project guide, Mr. Piyush Vashistha, Associate Professor, GLA University, Mathura, for providing all the material possible, providing valuable suggestions and encouraging throughout the project work.

We also thank all the staff members of our college and technicians for their help in making this project a successful one.

Finally, we take this opportunity to extend our deep appreciation to our family and friends, for all that they meant to us during the crucial times of the completion of our project.

**ABSTRACT**

In this project we have implemented a tool to Remove Object selected regions from an image. Removing Object refers to the art of removing unwanted parts of image and removing them based on the background information. The tool provides a user interface wherein the user can open an image for removing, select the parts of the image that he wants to remove. The tool would then automatically erase the selected area according to the background information. The image can then be saved. The removing is based on the exemplar-based approach. The basic aim of this approach is to find examples (i.e. patches) from the image and replace the lost data with it. Applications of this technique include the restoration of old photographs and damaged film; removal of superimposed text like dates, subtitles etc.; and the removal of entire objects from the image like microphones or wires in special effects.

# TABLE OF CONTENTS

1. **Introduction…………………………………………………………………………….…1** 
   1. Currently existing technologies…………………………………………………....2
   2. Problem definition and scope.…………………………………………………......2
2. **Description of Hardware and Software Used………………………………………….3**
   1. Hardware…………………………………………………………………………3
   2. Software………………………………………………………………………….3
3. **Theoretical Tools – Analysis and Development……………………………………….3** 
   1. Use Case Diagram….…………………………………………………………….3
4. **Development of Project …………………………………………..................................4** 
   1. User Interface Module…………………………………………………………...4
   2. Object removal ….……………………………………………………………...5
   3. Integration of the two modules…………………………………………………7
5. **Testing and Analysis……………………………………………………………….…7**
   1. Implementation Code ………………………………………………………....7
   2. Comparison with Another Algorithm………………………………………....11
   3. Comparison with Criminisi’s Approach……………………………………....11
   4. Comparison based on time with Criminisi’s approach……………………….12
   5. Real Life Examples……………………………………………………………12
6. **Conclusions…………………………………………………………………………….15**
7. **Recommendations and Future Work………………………………………………....15**

**References……………………………………………………………………………................15**

## 1. Introduction

The aim of the project is to develop a tool to remove selected regions from an image. Object Removing is the art of removing the parts of an image and reconstructing them based on the background information. This must be done in an undetectable way. The term removing 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. Figure 1 shows an example of this technique where a building (manually selected as the target region) is replaced by information from the remaining of the image in a visually plausible way. The algorithm automatically does this in a way that it looks “reasonable” to the human eye. Details that are hidden/occluded completely by the object to be removed cannot be recovered by any mathematical method. Therefore, the objective for object removal is not to recover the original image, but to create some image that has a close resemblance with the original image.



### (a) (b)

**Figure 1:** Removing large objects from an image.(a) Original image. (b) The building that had been selected manually has been removed from the image and the information from background is merged into the missing region.

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 damaged 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.

### 1.1 Currently Existing Technologies

Currently there are very few accepted technologies, tools for carrying out the work of Object Removal. It is still in the beginning stages and a lot of researches are being carried out to explore this area.

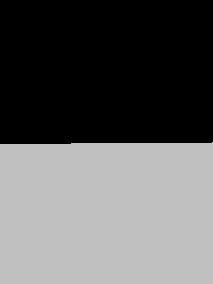
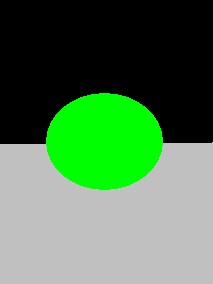
Due to the lack of such software’s, the restorers manually do the work of removing objects as in museums etc. It provides several algorithms including detection algorithms (which covers the problem of finding target areas), removing (discovers the problem of filling detected cracks and missing thin parts of the images, paintings and frescos), Restoration (deals with removing noise etc.) along with several other algorithms.

Another software that deals with the solution to the problem of inpainting. It provides tools for selecting the region to be removed and then applies some algorithm to achieve the desired result..

### 1.2 Problem Definition and Scope

The object of the project is to remove some portions of the image, to make it more legible and restore its unity. The whole scope of the problem can be stated as:

* Remove the regions from the image that have been marked by the user for removing. The user may mark more than one region.
* Given an image and a region to be removed**,** removing would try to construct an image and remove the marked region in a visually plausible way.
* This can be done by using information from surrounding areas and merge the removed region into the image so seamlessly that a typical viewer is not aware of it. The quality of the result will depend on what is missing. If the inpainting region is small and the surrounding area is without much texture, the result will be good. Large areas with lots of information lost are harder to reconstruct, because information in other parts of the image is not enough to get an impression of what is missing. If the human brain is not able to imagine what is missing, equations will not make it either. 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.



**(a) (b)**

**Figure 2**: Example of object removal. Marked portion of image is shown in green color. The image on the right-hand side is the object removed image.

## 2. Description of Hardware and Software Used

### 2.1 Hardware Used

The tool was developed, and the testing performed on 2.83 GHz Intel Core 2 Quad Processor with 2 GB RAM.

### 2.2 Software Used

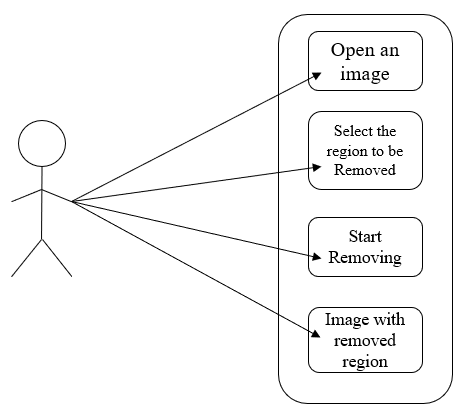
We have used NetBeans 6.8 for the development of the tool. The reason for selecting NetBeans ahead of MATLAB for making the tool was that NetBeans is completely free software. All the libraries that we used are provided with the default distribution of Java Development Kit.

## 3. Theoretical Tools – Analysis and Development

### 3.1 Use Case Diagram

The following represents the use case model for our system. The user performs the following tasks:

1. Open an Image.
2. Select the region to be removed.
3. Start removing.
4. Image with removed region.



**Figure 3:** Use Case Diagram

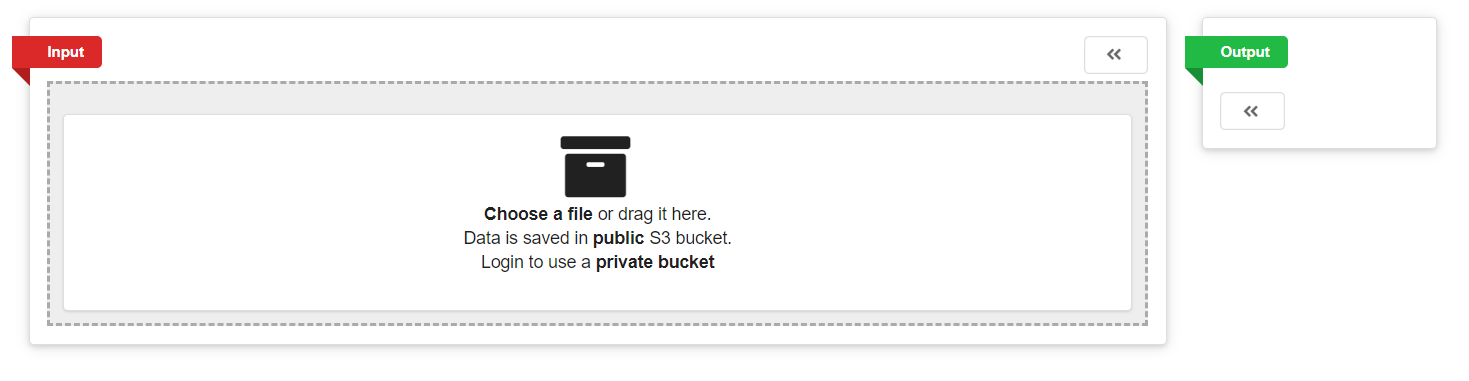
## 4. Development of Project

We have divided the project into two modules. The first module is the user interface module and the second is the object removal module.

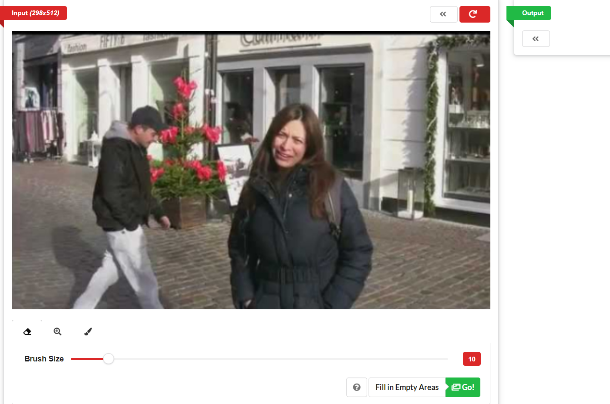
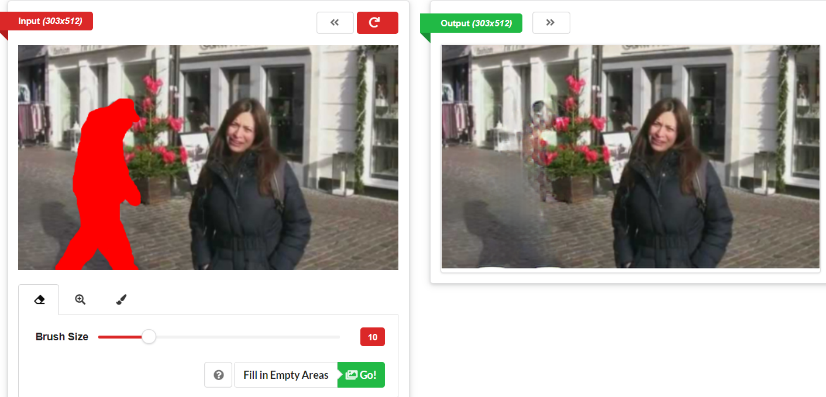
**4.1 User Interface Module:**

The user interface module is concerned with presenting the user with an interface wherein he could select the region to be removed by marking the region. The user can select as many regions as he wants in the image. The regions need not be spatially connected. This is useful in the context of object removal as the output of removed object image may depend on how well the area to be removed is selected. The following summarizes the responsibilities of the user interface module:

1. Open an image.
2. Select the region to be removed.
3. Click on “GO” to remove the object that is selected.
4. Output the image of removed object.

****

**Figure 4:** InitialInterface

**Figure 5(a):** Image uploaded **(b):** Object Removed Image

**Figure 5(b)**: Selecting an object using our tool. The user is required to mark the object of the region.

### 4.2 Object Removal

This module is concerned with the removing the object from the image. It receives the image from the user where the region to be removed is marked in red. As mentioned earlier, we have chosen red color because of its use in the creation of special effects in movies etc.

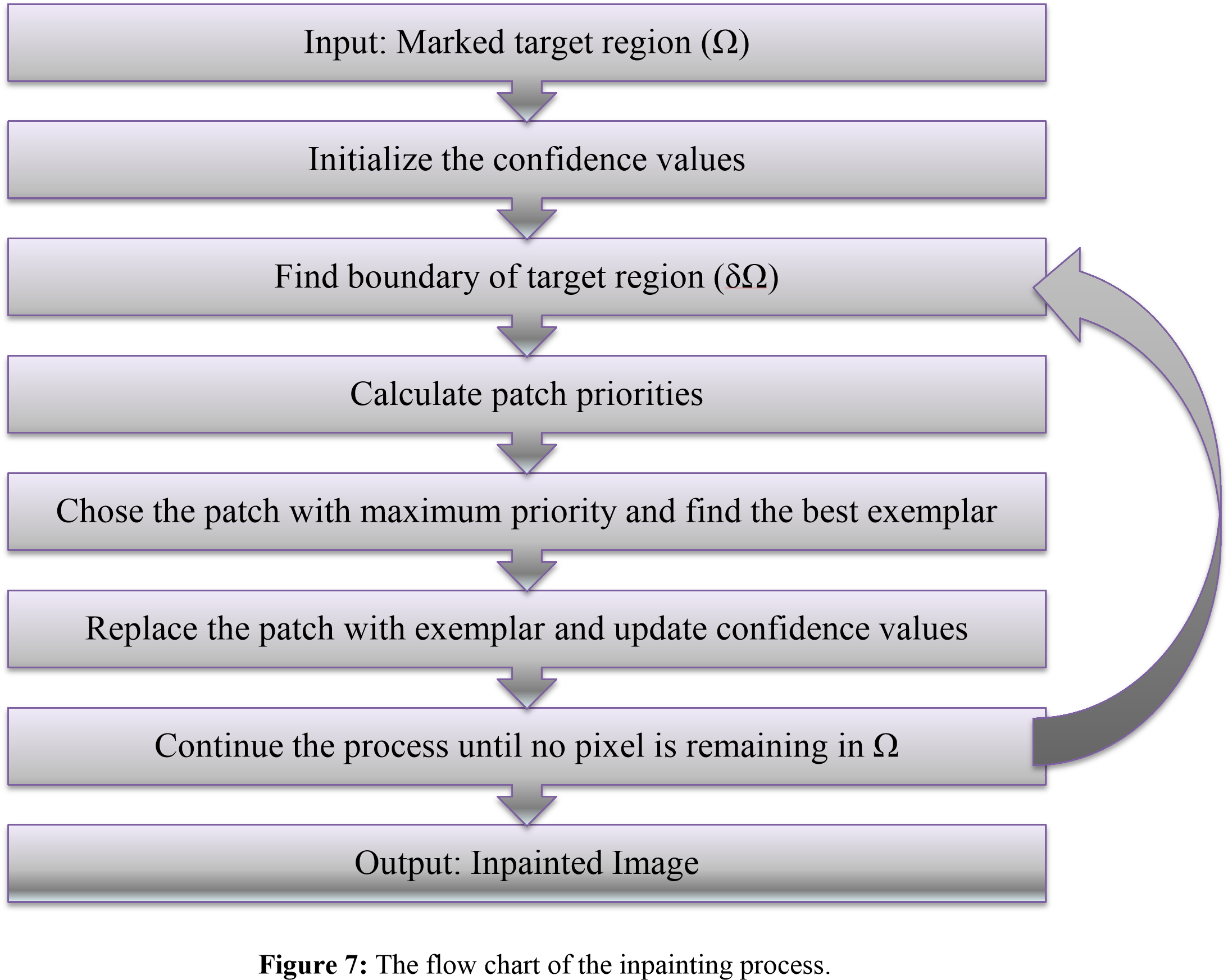
Let us first describe the terms used in removing literature.

1. The object to be removed is represented as **I**.
2. The target region (i.e., the region to be removed) is represented as **Ω**.
3. The source region (i.e., the region from the image which is not to be removed and from where the information can be extracted to reconstruct the target region) is represented as **Φ**.

Φ = I - Ω

1. The boundary of the target region (i.e., the pixels that separate the target region from the source region) is represented as **δΩ**.

As with all other exemplar-based algorithms, this algorithm replaces the target region patch by patch. This patch is generally called the template window, ψ. The size of ψ must be defined for the algorithm. This size is generally kept being larger than the largest texture element in the source region. We have kept the default patch size of 9 x 9 but we may have to vary it for some images. Once these parameters are assigned the remaining process is completely automatic. The algorithm now proceeds as follows:



**Figure 6:** Flow chart of the process of object removing.

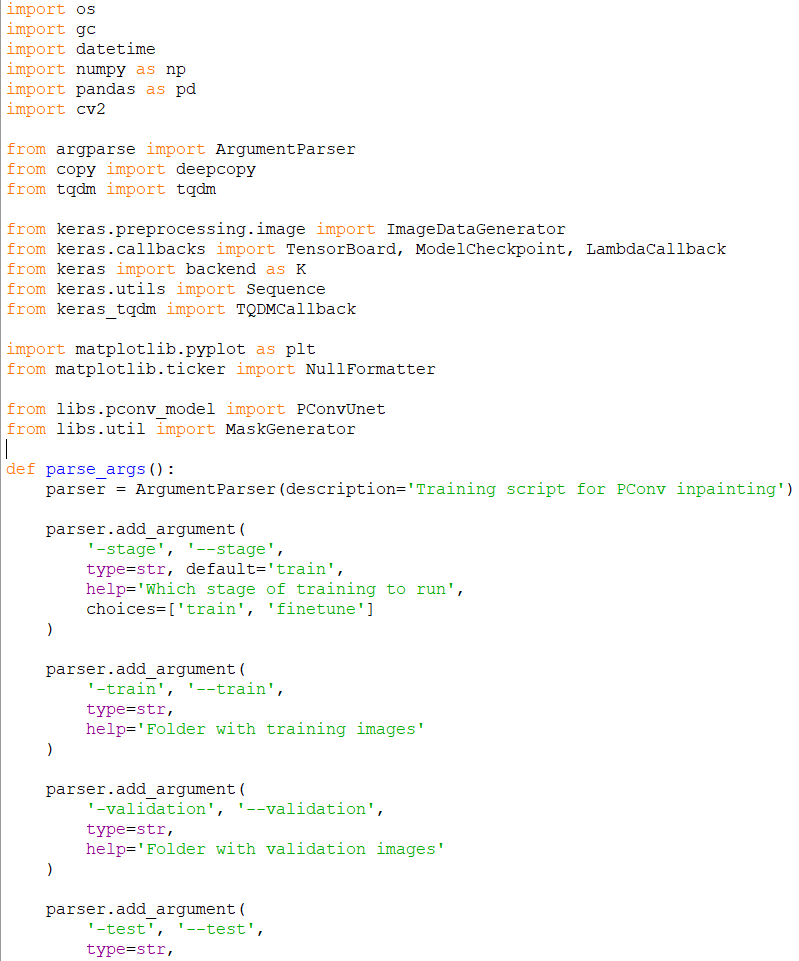
### 4.3 Integration of the two modules

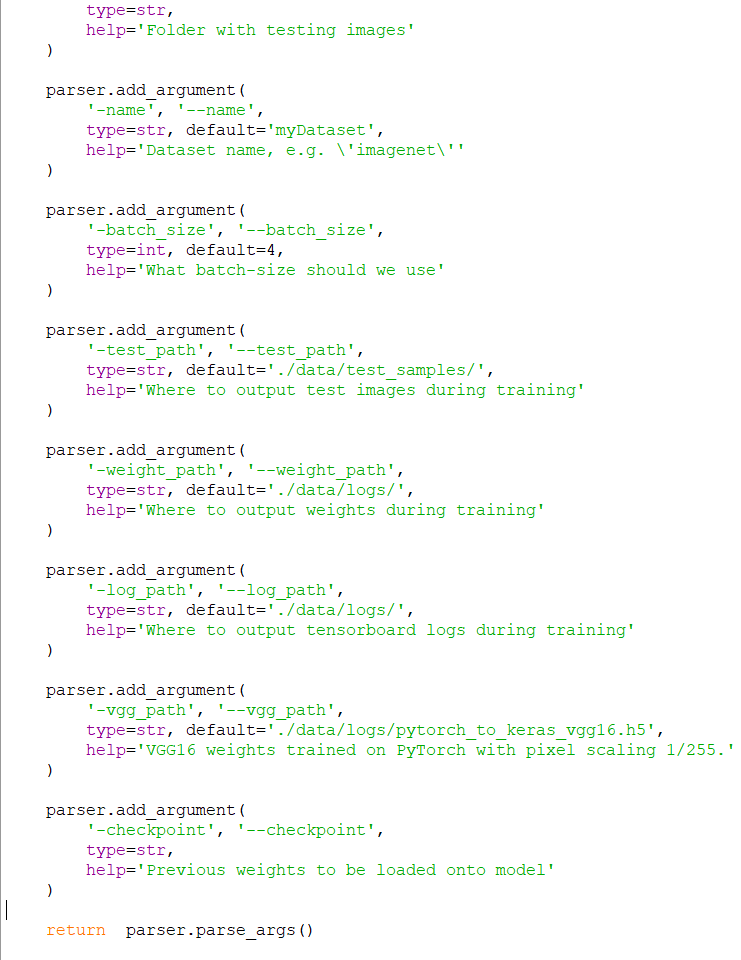
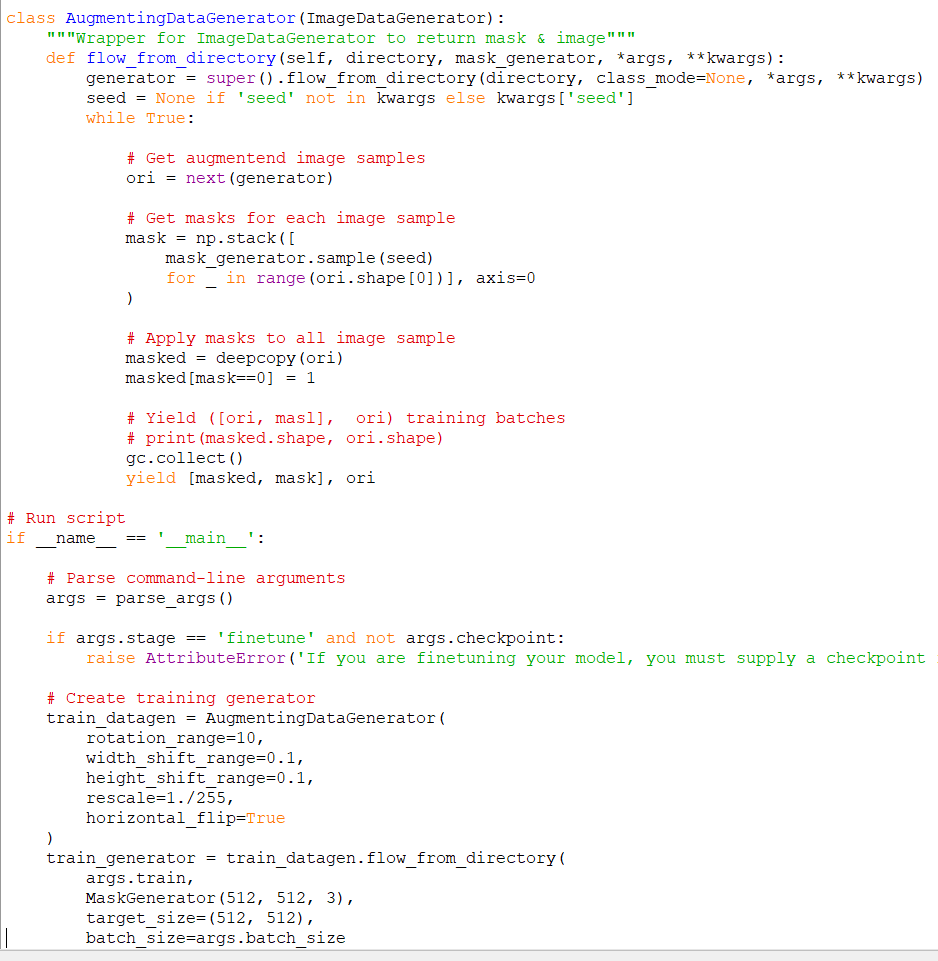
Once, we have the two modules, the next step is the integration of the module. The integration was performed so that first the user is presented with the user interface and when he selects the option to remove the object from the image, the object removal module is called from within the user interface module. The object removal module sends the updated image to the user interface module at every step of the process and thus the user can see the removing as it progresses. This is important because since the user can see the updates at every moment, it keeps him informed of how much removing has been done and how much is left.

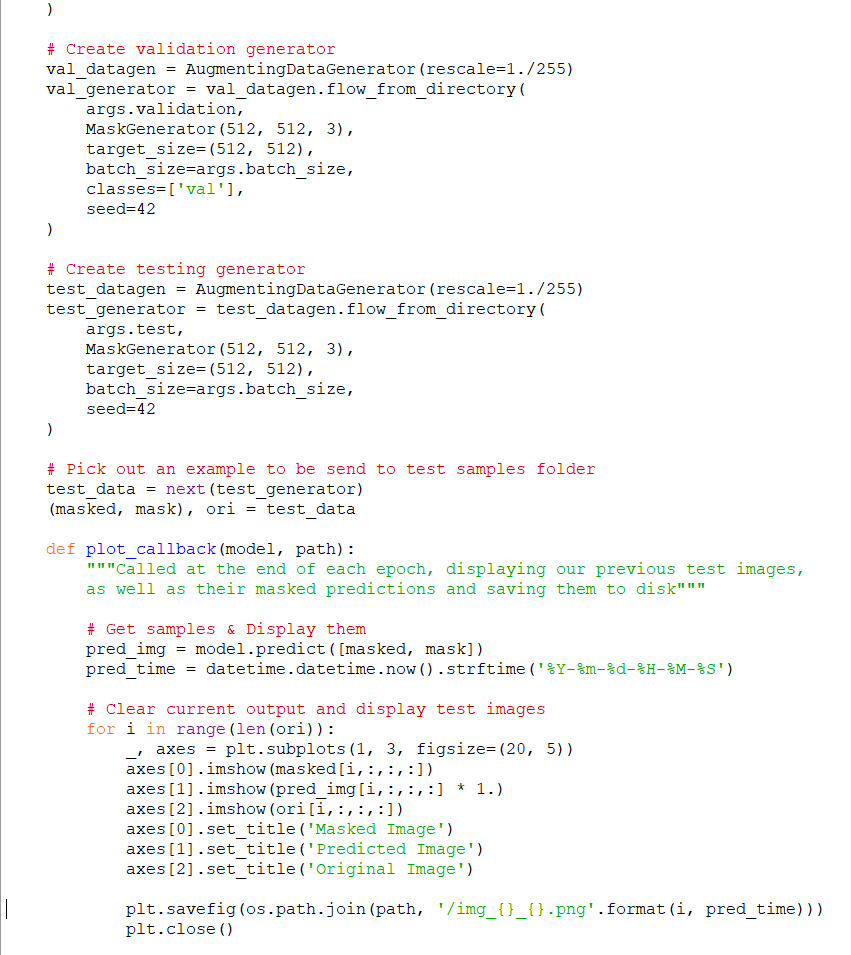
## 5. Testing and Analysis

We applied our algorithm to several images ranging from simple images to the images with complex textures. We have also made comparisons with several other algorithms which we present side by side in this section. In most of the experiments, the patch size was set to 9 x 9. We will state appropriately wherever a different patch size was taken by us and the reasons for the difference. All experiments were run on a 2.83 Core 2 Quad Processor with 2 GB RAM.

**5.1 Implementation Code**

****

** **

****

****

### 5.2 Comparison with Another algorithm

First, we present a brief comparison of our approach with Another Algorithm.

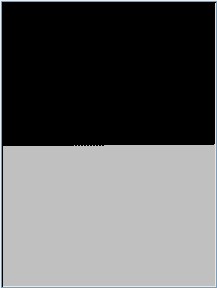
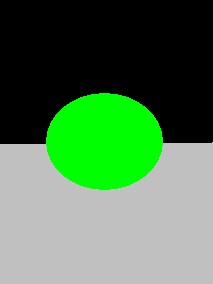


(a) (b) (c)

**Figure 7:** Comparison with the onion peel algorithm. (a) The input image, (b) The image with our Algorithm. (c) The results of inpainting using another algorithm.

This difference in output from another algorithm occurs because of the concentric outward to inward fill algorithm. On the other hand, our algorithm fills the center of the board first where the isophones flows into the patch and then starts filling the outer region.

### 5.3 Comparison with Criminisi’s approach



(a) (b) (c)

**Figure 8:** Comparison with Criminisi’s approach. (a) Image to be removed. (b) Result using our algorithm. (c) Result using our implementation of Criminisi’s approach.

Next let us compare some outputs that we obtained with the algorithm proposed by Criminisi.

This result that we obtained was after we started using the extension that we earlier proposed which involved the use of variance if the patch with minimum error was found to be the same.

**5.4 Comparison based on time taken with Criminisi’s approach**

Also using the fast object removal algorithm, the time taken in inpainting the image is considerably reduced. For example, for the following image, we present the comparison (based on time taken) between our approach and our implementation of Criminisi’s Approach.



1. (b) (c) (d)

**Figure 9:** Comparison with Criminisi’s Approach (based on time taken). (a) The input image of size 416 x 316 [2]. (b) The image with island to be removed marked in green color. (c) The output using our algorithm. Time taken was 2 minutes and 5 seconds. (d) The output using our implementation of Criminisi’s approach. The time taken was 2 minutes 35 seconds.

### 5.5 Real Life Examples

We now show a few more examples from real scenes.

We have an image with two persons. One person in the image is the unwanted person that we now want to remove. Other objects that we want to remove from this image. We manually select the regions through the user interface and perform removing. This could be done by selecting one object at a time or selecting multiple objects at the same time. We present the example after selecting multiple objects at the same time.

1. (b)



(c)

**Figure 10:** Example of removing unwanted person (a) The original image with an unwanted person. (b) Image with the unwanted person and scratches selected. (c) Image with the scratches and unwanted person removed.

Another example that we now mention presents the image of an eagle along with water. The eagle selected to be removed has been removed by the algorithm by reconstructing the complex texture from water. Note that the white lines near the leg of the eagle are preserved in the image.

(a) (b)

**Figure 11:** Image with complex texture. (a) Original image. (b) Image with the eagle removed.

The final example that we present here shows an image and walks through the image with objects with different complexities removed at each step.



(a) (b)



(c) (d)

**Figure 12:** Removing multiple objects from a photograph. (a) The original image. (b) The image with the sign board and one cat removed. (c) The image with another cat removed. (d) The image with both the men removed.

## 6. Conclusions

During this project we have implemented the basic algorithm presented by A. Criminisi et. al. and then worked from thereon to improve it based on our observations. This presents an algorithm that can remove objects from the image in a way that it seems reasonable to the human eye.

Our approach employs an exemplar based inpainting along with a priority term that defines the filling order in the image. In this algorithm, pixels maintain a confidence value and are chosen based on their priority that is calculated using confidence and data term. The confidence term defines how much sure we are about the validity of that pixel whereas data term is focused towards maintaining the linear structures in the image. This approach is capable of propagating both linear structures and 2 dimensional textures into the target region. This technique can be used to fill small scratches in the image/photos as well as to remove larger objects from them. It is also computationally efficient and works well with larger images.

## 7. Recommendations and Future Work

We are looking forward to improving the algorithm so that the computational complexity is improved while retaining the quality of removing object and if possible, we would also like to improve the object removal algorithm. Also, the inpainting algorithm presented here is not capable enough to be used for inpainting videos, i.e. removing some scratches or some objects from videos. We are also exploring towards this area to make it more robust so that it can be used with videos. We have tried using the same algorithm for different frames of a video, the results are good.

## References

1. <https://en.wikipedia.org/wiki/Inpainting>
2. <https://en.wikipedia.org/wiki/Computer_vision>
3. <https://en.wikipedia.org/wiki/Object_detection>
4. <https://www.deeplearningbook.org/>
5. Digital object Removal by Harald Grossauer