VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

AN AUTONOMOUS INSTITUTE

(Approved by AICTE, New Delhi, Govt of T.S and Affiliated to JNTU, Hyderabad) Accredited by NBA and NAAC with ‘A++’ Grade

Bachupally, Hyderabad –
500090 Telangana, India.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Estd.1995

SEVEN HABITS OF HIGHLY EFFECTIVE PEOPLE

1. Be Proactive.
2. Begin with the end in mind.
3. Put first things first.
4. Think Win-Win.
5. First Understand, then be understood.
6. Synergies.
7. Sharpen Your Saw.

We have followed the above 7 steps during the course of our project work

G. GEETHIKA SRI	**17071A0481**
K. MANUMITHA	**17071A0487**
K. RADHIKA	**17071A0494**
S. KAIVALYA RAO	**17071A04B1**

RESTORATION OF OLD DAMAGED PHOTOS BY LATENT SPACE TRANSLATION

INDUSTRY ORIENTED MAJOR PROJECT WORK SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENT FOR THE AWARD OF THE DEGREE OF
BACHELOR OF TECHNOLOGY
IN

ELECTRONICS & COMMUNICATION ENGINEERING

Submitted By

G. GEETHIKA SRI	17071A0481
K. MANUMITHA	17071A0487
K. RADHIKA	17071A0494
S. KAIVALYA RAO	17071A04B1

UNDER THE SUPERVISION OF

Ms. K. DEEPTHI

ASSISTANT PROFESSOR

VNRVJIET

TAMASOMA JYOTIRGAMAYA

Estd.1995

Department of Electronics and Communication Engineering

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

AN AUTONOMOUS INSTITUTE

(Approved by AICTE, New Delhi, Govt of T.S and Affiliated to JNTU, Hyderabad) Accredited by NBA and NAAC with ‘A++’ Grade

Bachupally, Hyderabad –
500090 Telangana, India.

2020-21

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

AN AUTONOMOUS INSTITUTE

(Approved by AICTE, New Delhi, Govt of T.S and Affiliated to JNTU, Hyderabad) Accredited by NBA and NAAC with 'A++' Grade

Bachupally, Hyderabad –
500090 Telangana, India.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the project report entitled "**RESTORATION OF OLD DAMAGED PHOTOS BY LATENT SPACE TRANSLATION"** is submitted under my supervision and is being submitted by **G.GEETHIKA SRI (17071A0481), K.MANUMITHA(17071A0487), K.RADHIKA(17071A0494), S.KAIVALYA RAO(17071A04B1)** in partial fulfillment for the award of the Degree of **Bachelor of Technology** in Electronics and Communication Engineering, of VNR VJIET, Hyderabad during the academic year 2020-21.

Certified further that to the best of my knowledge the work presented in this thesis has not been submitted to any other University or Institute for the award of any Degree.

SUPERVISOR	**HEAD OF THE DEPARTMENT**
Ms. K. DEEPTHI	**DR. Y. PADMA SAI,**
Assistant Professor	**Head of Department of ECE**
VNR VJIET, Hyd.	**VNR VJIET, Hyd.**

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

AN AUTONOMOUS INSTITUTE

(Approved by AICTE, New Delhi, Govt of T.S and Affiliated to JNTU, Hyderabad) Accredited by NBA and NAAC with 'A++' Grade

Bachupally, Hyderabad –
500090 Telangana, India.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Estd.1995

DECLARATION

We do declare that the project report entitled **"RESTORATION OF OLD DAMAGED PHOTOS BY LATENT SPACE TRANSLATION"** submitted to the department of Electronics and Communication Engineering (ECE), Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology, Hyderabad, in partial fulfillment of the requirement for the award of the degree of **BACHELOR OF TECHNOLOGY** in Electronics and Communication Engineering is the bona-fide record of the project report presented under the Supervision of **Ms. K.DEEPTHI**, Assistant Professor, VNRVJIET.

Also, we declare that the matter embodied in this project report has not been submitted by me in full or in any part there for the award of any degree/diploma of any other institution or university previously.

PLACE: Hyderabad
DATE:

G. GEETHIKA SRI
K. MANUMITHA
K. RADHIKA
S. KAIVALYA RAO

ACKNOWLEDGEMENTS

We wish to express our deep sense of gratitude to **Ms. K. DEEPTHI**, Assistant Professor, VNR VJIET for his valuable guidance and constant encouragement in all respects during the course of our project work.

Our sincere thanks to **Dr. Y. PADMA SAI**, Professor and Head of Department of Electronics and Communication Engineering, VNRVJIET, Hyderabad for the encouragement and guidance provided during the course of our project work.

We are thankful to the Principal **Dr. C.D. NAIDU**, VNRVJIET, Hyderabad, for giving us permission to carry out this project.

We are thankful to all the staff members of ECE department, VNRVJIET for helping us during this project.

We are thankful to all the project committee members of ECE department, VNR VJIET for helping us during this project.

Finally, we are very thankful to our family members and our friends for their great moral support.

G. GEETHIKA SRI
K. MANUMITHA
K. RADHIKA
S. KAIVALYA RAO

ABSTRACT

Through a deep learning approach, we propose to restore old photos that suffer from severe degradation. The degradation in real photos in complex and the domain gap between synthetic images and real old photos makes the network fail to generalize unlike conventional restoration tasks which will be solved through supervised learning.

By proposing a singular novel triplet domain translation network by leveraging real photos alongside massive synthetic image pairs. And the change between these two latent spaces is acquired with synthetic paired data. This translation concludes well to synthetic photos because the domain gap is closed the compact latent space. This system is used for converting an image which is damaged into a new version of that image.

CONTENTS

CHAPTER 1 INTRODUCTION

CHAPTER 2 FEASIBILITY STUDY

CHAPTER 3 LITERATURE SURVEY

CHAPTER 4 IMAGE DEFFECTS

CHAPTER 5 SYSTEM ANALYSIS

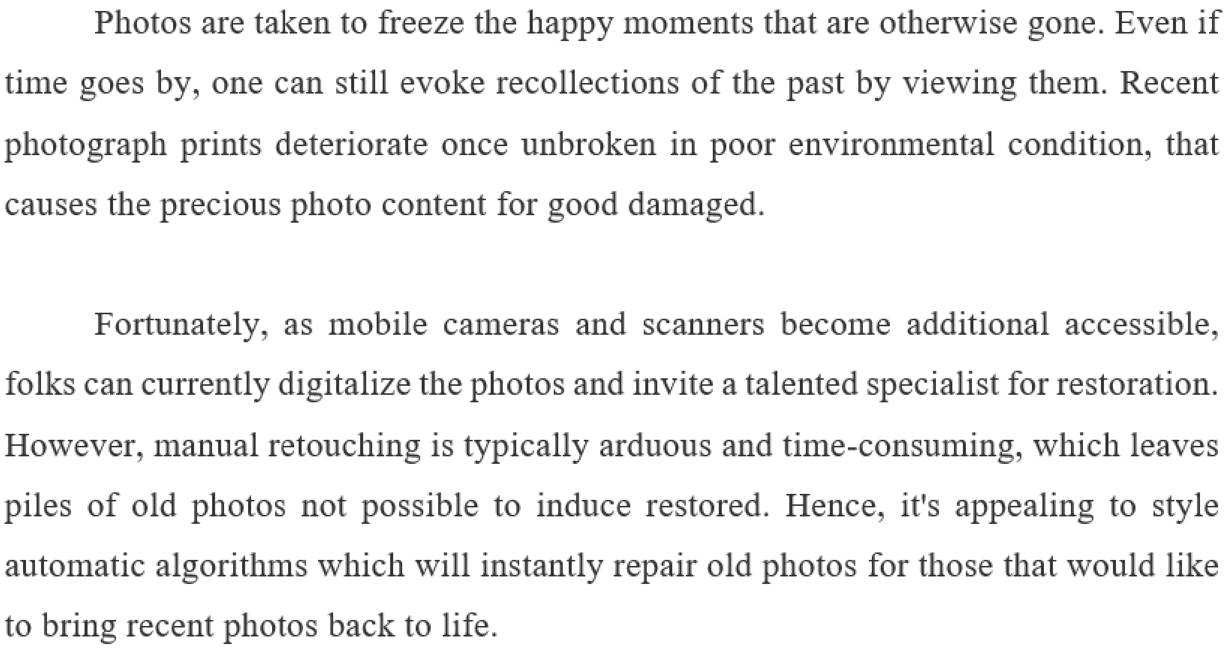
CHAPTER 6 SYSTEM IMPLEMENTATION

CHAPTER 7 METHODOLOGY AND PROCESS FLOW

CHAPTER 8 SYSTEM DESIGN

CHAPTER-1
INTRODUCTION

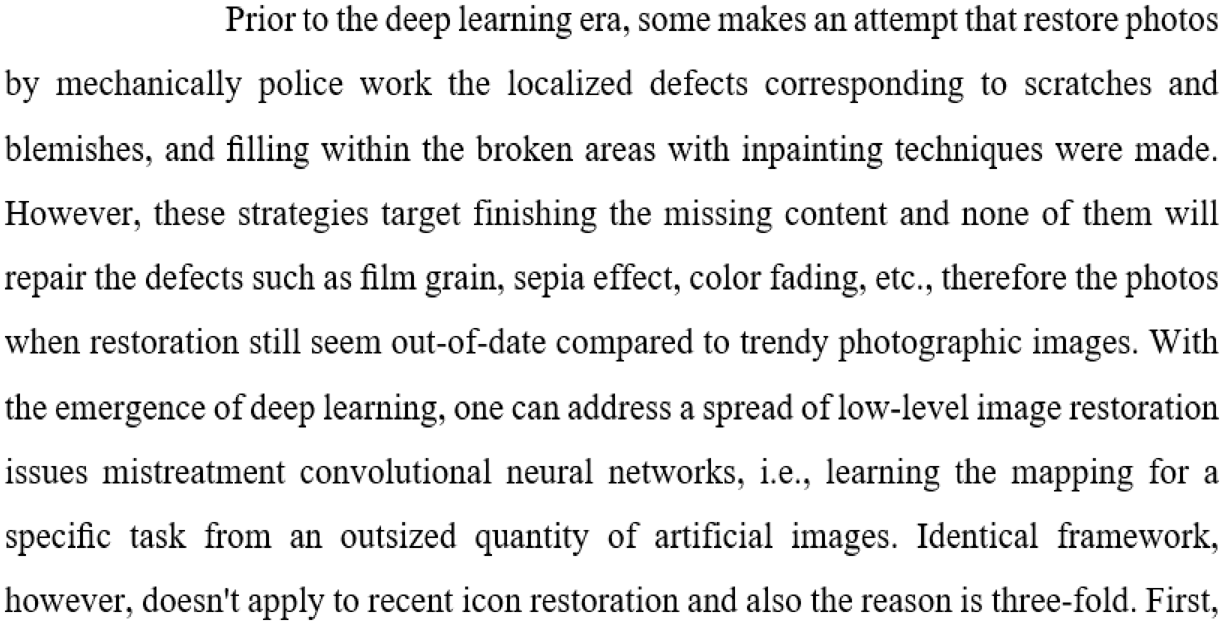
1.1 PURPOSE



Photos are taken to freeze the happy moments that are otherwise gone. Even if time goes by, one can still evoke recollections of the past by viewing them. Recent photograph prints deteriorate once unbroken in poor environmental condition, that causes the precious photo content for good damaged.

Fortunately, as mobile cameras and scanners become additional accessible, folks can currently digitalize the photos and invite a talented specialist for restoration. However, manual retouching is typically arduous and time-consuming, which leaves piles of old photos not possible to induce restored. Hence, it's appealing to style automatic algorithms which will instantly repair old photos for those that would like to bring recent photos back to life.

1.2 IMAGE DEGRADATION



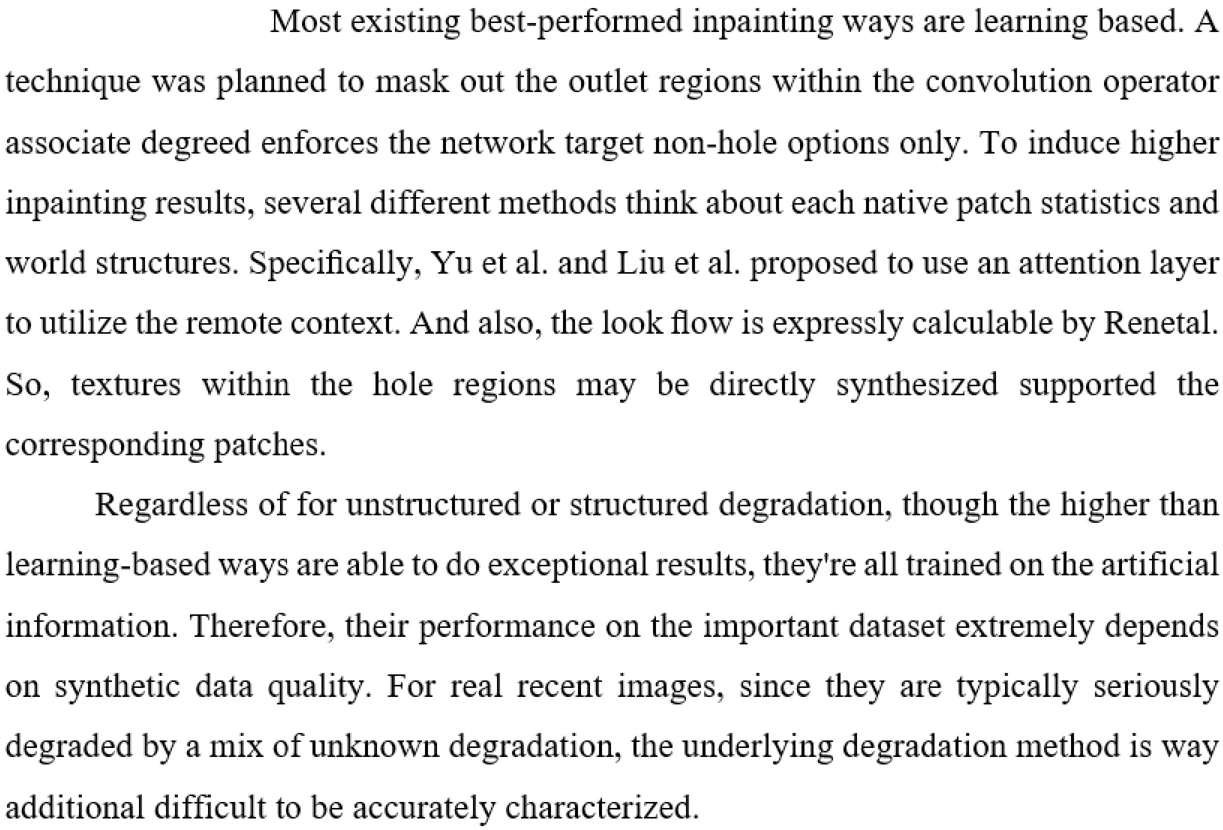
Prior to the deep learning era, some makes an attempt that restore photos by mechanically police work the localized defects corresponding to scratches and blemishes, and filling within the broken areas with inpainting techniques were made. However, these strategies target finishing the missing content and none of them will repair the defects such as film grain, sepia effect, color fading, etc., therefore the photos when restoration still seem out-of-date compared to trendy photographic images. With the emergence of deep learning, one can address a spread of low-level image restoration issues mistreatment convolutional neural networks, i.e., learning the mapping for a specific task from an outsized quantity of artificial images. Identical framework, however, doesn't apply to recent icon restoration and also the reason is three-fold. First, the degradation method of old photos is very complex, and there exists no degradation model which will realistically render the old photo.

1.3 EXISTING SYSTEM

Most existing best-performed inpainting ways are learning based. A technique was planned to mask out the outlet regions within the convolution operator associate degreed enforces the network target non-hole options only. To induce higher inpainting results, several different methods think about each native patch statistics and world structures. Specifically, Yu et al. and Liu et al. proposed to use an attention layer to utilize the remote context. And also, the look flow is expressly calculable by Renetal. So, textures within the hole regions may be directly synthesized supported the corresponding patches.

Regardless of for unstructured or structured degradation, though the higher than learning-based ways are able to do exceptional results, they're all trained on the artificial information. Therefore, their performance on the important dataset extremely depends on synthetic data quality. For real recent images, since they are typically seriously degraded by a mix of unknown degradation, the underlying degradation method is way additional difficult to be accurately characterized. In different words, the network trained on synthetic data only, can suffer from the domain gap downside and perform badly on real old images. we tend to model real old photo restoration as a replacement triplet domain translation problem and a few new techniques are adopted to attenuate the domain gap.

1.3.1 SINGLE DEGRADATION IMAGE RESTORATION



Most existing best-performed inpainting ways are learning based. A technique was planned to mask out the outlet regions within the convolution operator associate degreed enforces the network target non-hole options only. To induce higher inpainting results, several different methods think about each native patch statistics and world structures. Specifically, Yu et al. and Liu et al. proposed to use an attention layer to utilize the remote context. And also, the look flow is expressly calculable by Renetal. So, textures within the hole regions may be directly synthesized supported the corresponding patches.

Regardless of for unstructured or structured degradation, though the higher than learning-based ways are able to do exceptional results, they're all trained on the artificial information. Therefore, their performance on the important dataset extremely depends on synthetic data quality. For real recent images, since they are typically seriously degraded by a mix of unknown degradation, the underlying degradation method is way additional difficult to be accurately characterized.

1.3.2 MIXED DEGRADATION IMAGE RESTORATION

In Mixed Degradation, we use methods like crafting a Toolchain for Image Restoration by Deep Reinforcement Learning. The advantage is that a toolbox consisting of small-scale Convolutional Networks of different complexities but rely on supervised learning from synthetic data and hence cannot generalize to real photos which is a major drawback. Another method is based on attention and Adaptive Selection of Operations for Image Reconstruction in the Presence of Unknown Combined Distortions. The main advantage of this procedure is the novel triplet domain translation network by hiding real photos along with massive synthetic image pairs but only focus on unstructured defects and does not support structured defects like images in the painting. We can also use the Deep Image Prior technique which is a type of Convolutional Neural Network used to improve a given image with no pre training data other than the image itself. Since the deep neural network inherently resonates with low-level image statistics and thereby can be utilized as an image before blind image restoration without external training data. But it is hard to restore in-the-wild images corrupted by mixed factors.

1.3.3 DRAWBACKS OF EXISTING SYSTEMS

- Can handle only single degradations such as de-noising, de-blurring.
- Resolution of images is poor.
- Missing portions of images are not handled.
- Images having scratches are not restored.

1.4 PROPOSED SYSTEM

We use novel triplet domain translation network to get back the mixed degradation in old photos. The domain gap is reduced between old photos and clean images, and the translation to clean images is learned in latent space. Our method suffers less from generalization issues compared with prior methods. Furthermore, We propose a partial nonblack that restores the latent features by leveraging the global context, So the scratches can be in-painted with better structural consistency. Our method demonstrates good performance in restoring severely degraded old photos.

CHAPTER-2
FEASIBILITY STUDY

A feasibility study involves taking a judgment call on whether a project is doable, worth the investment and evaluates the project's potential for success. A feasibility analysis is employed to work out the viability of a thought, like ensuring a project is legally and technically feasible also as economically justifiable. The feasibility of the project is analyzed during this phase and business proposal is put forth with a really general plan for the project and a few cost estimates. This is to make sure that the proposed system isn't a burden. The two criteria to guage feasibility are cost required and value to be delivered. Conducting a feasibility study before technical development and project implementation is always beneficial as it gives us a clear picture of the proposed project. The project may require too many resources and may cost more than what we earn back by taking on a project which isn't profitable. For feasibility analysis, some understanding of the main requirements for the system is important. There are six types of feasibility study i.e. separate areas that a feasibility study examines.

2.1 TECHNICAL FEASIBILITY

Technical feasibility involves the analysis of the hardware, software, and alternative technical needs of the projected system. It helps to work out whether or not the technical resources accessible meet capability and whether the team is capable of changing the ideas into operating systems. Any system developed shouldn't have a high demand on the available technical resources. This may cause high demands being placed on the client. The developed system must have a modest requirement. In our project, the technology concerned is Deep

Learning. The language won’t to implement the concepts is Python Programming and also the tool used to execute the code is Google Colab.

2.2 ECONOMIC FEASIBILITY

Economic feasibility typically helps in assessing the viability, cost analysis, and benefits associated with the project before financial resources are allocated and enhances project credibility, helping decision-makers determine the positive economic benefits that the proposed project will provide.

This study is administered to see the economic impact that the system will wear the organization. The amount of fund that can be poured into the research and development of the system is limited. The expenditures must be justified. The developed system was made within the budget and this was achieved because all technologies used are freely available. This system is designed such that it requires minimal cost and time and eliminates the need for manual work as we trained the model using deep learning technology.

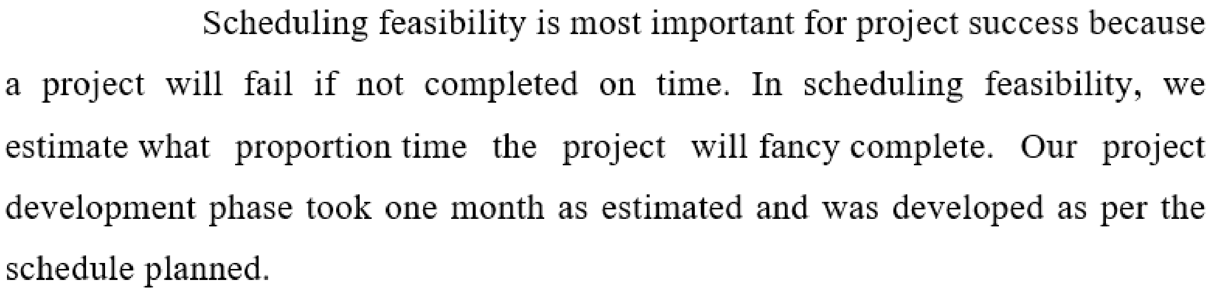
2.3 LEGAL FEASIBILITY

Legal feasibility ensures that the proposed system doesn’t conflict with legal requirements like zoning laws, data protection acts or social media laws. Our project ensures legal data access and gives prominence to data security.

2.4 OPERATIONAL FEASIBILITY

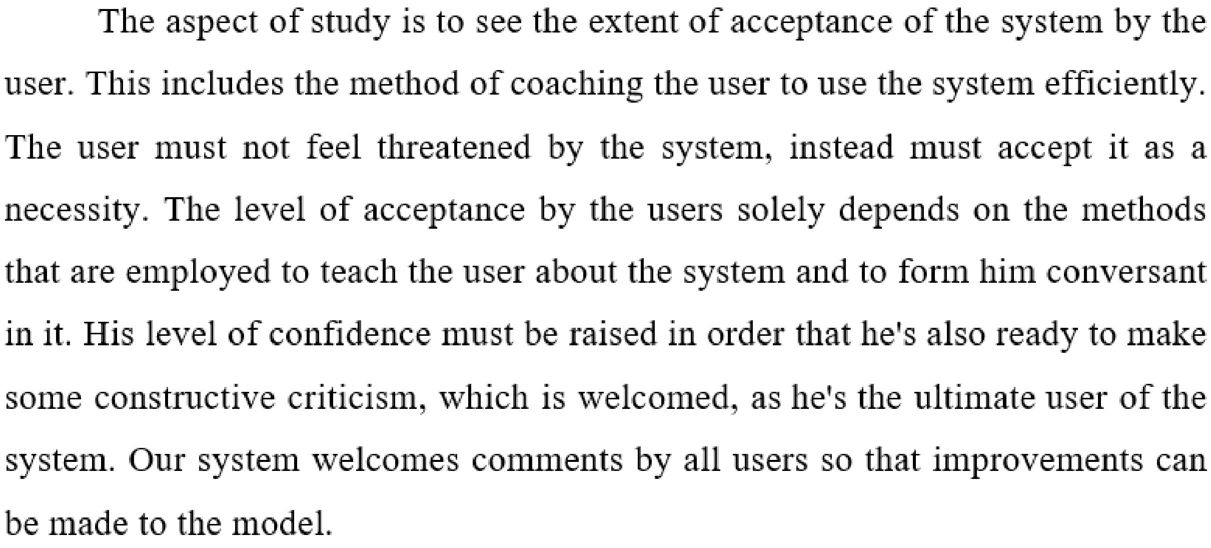
Operational feasibility studies examine how a project plan satisfies the requirements identified in the requirement analysis phase of system development. The system involves design-dependent parameters such as reliability, maintainability, supportability, usability, disposability, sustainability, affordability, and others. It minimizes the drawbacks of the current system by building a model that can handle multiple defects both structured and unstructured defects in an image simultaneously.

2.5 SCHEDULING FEASIBILITY



Scheduling feasibility is most important for project success because a project will fail if not completed on time. In scheduling feasibility, we estimate what proportion time the project will fancy complete. Our project development phase took one month as estimated and was developed as per the schedule planned.

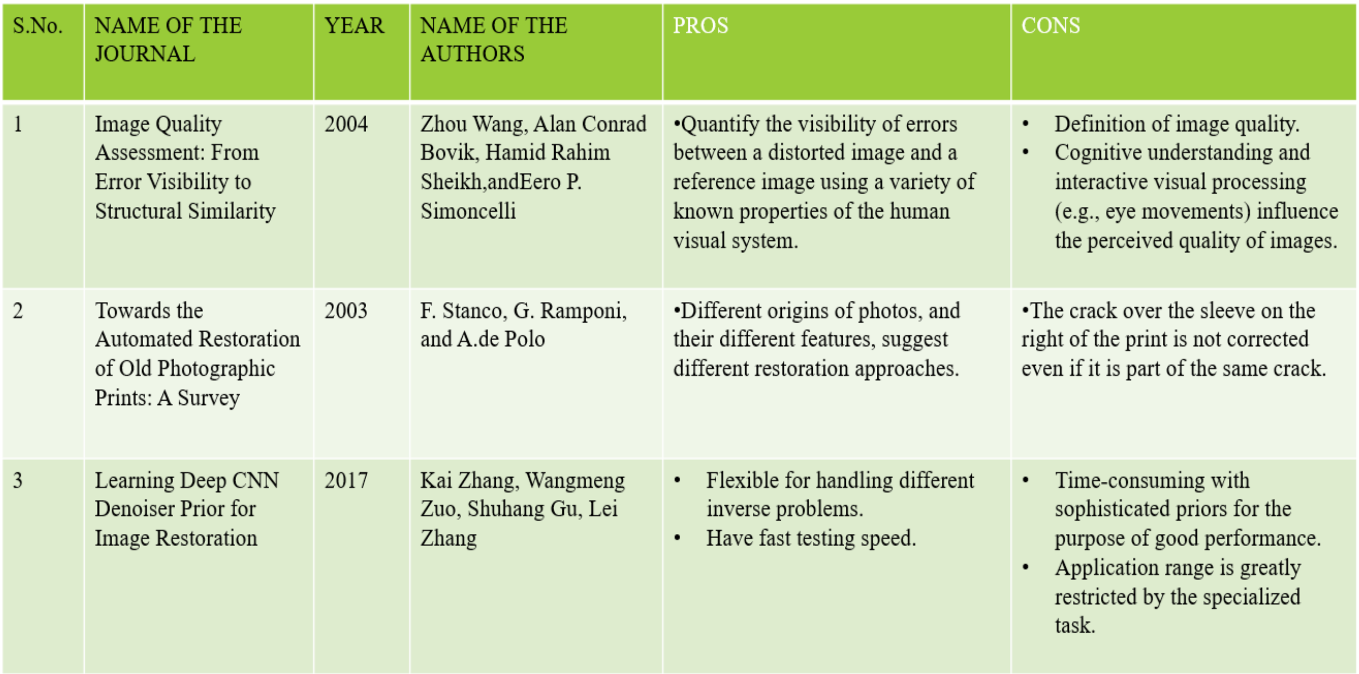
2.6 SOCIAL FEASIBILITY



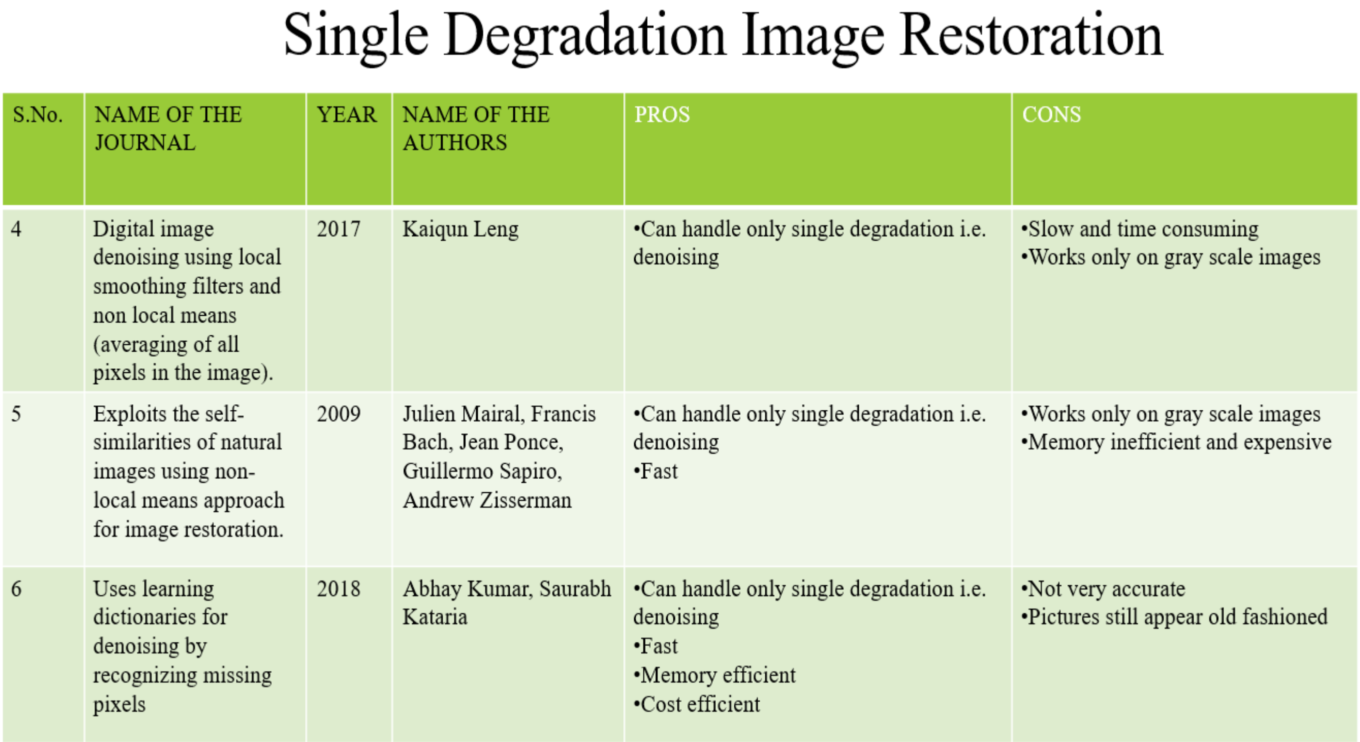
The aspect of study is to see the extent of acceptance of the system by the user. This includes the method of coaching the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to teach the user about the system and to form him conversant in it. His level of confidence must be raised in order that he's also ready to make some constructive criticism, which is welcomed, as he's the ultimate user of the system. Our system welcomes comments by all users so that improvements can be made to the model.

CHAPTER 3
LITERATURE SURVEY

3.1 RESEARCH PAPERS REFERRED



S.No.	NAME OF THE JOURNAL	YEAR	NAME OF THE AUTHORS	PROS	CONS
1	Image Quality Assessment: From Error Visibility to Structural Similarity	2004	Zhou Wang, Alan Conrad Bovik, Hamid Rahim Sheikh,andEero P. Simoncelli	•Quantify the visibility of errors between a distorted image and a reference image using a variety of known properties of the human visual system.	• Definition of image quality. • Cognitive understanding and interactive visual processing (e.g., eye movements) influence the perceived quality of images.
2	Towards the Automated Restoration of Old Photographic Prints: A Survey	2003	F. Stanco, G. Ramponi, and A.de Polo	•Different origins of photos, and their different features, suggest different restoration approaches.	•The crack over the sleeve on the right of the print is not corrected even if it is part of the same crack.
3	Learning Deep CNN Denoiser Prior for Image Restoration	2017	Kai Zhang, Wangmeng Zuo, Shuhang Gu, Lei Zhang	• Flexible for handling different inverse problems. • Have fast testing speed.	• Time-consuming with sophisticated priors for the purpose of good performance. • Application range is greatly restricted by the specialized task.



Single Degradation Image Restoration

S.No.	NAME OF THE JOURNAL	YEAR	NAME OF THE AUTHORS	PROS	CONS
4	Digital image denoising using local smoothing filters and non local means (averaging of all pixels in the image).	2017	Kaiqun Leng	•Can handle only single degradation i.e. denoising	•Slow and time consuming •Works only on gray scale images
5	Exploits the self-similarities of natural images using non-local means approach for image restoration.	2009	Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, Andrew Zisserman	•Can handle only single degradation i.e. denoising •Fast	•Works only on gray scale images •Memory inefficient and expensive
6	Uses learning dictionaries for denoising by recognizing missing pixels	2018	Abhay Kumar, Saurabh Kataria	•Can handle only single degradation i.e. denoising •Fast •Memory efficient •Cost efficient	•Not very accurate •Pictures still appear old fashioned

Mixed Degradation Image Restoration

S.No.	NAME OF THE JOURNAL	YEAR	NAME OF THE AUTHORS	PROS	CONS
7	Crafting a Toolchain for Image Restoration by Deep Reinforcement Learning	2018	Ke Yu, Chao Dong, Liang Lin, Chen Change Loy	A toolbox consisting of small-scale convolutional networks of different complexities	Rely on supervised learning from synthetic data and hence cannot generalize to real photos.
8	Attention-based Adaptive Selection of Operations for Image Restoration in the Presence of Unknown Combined Distortions	2019	Masanori Suganuma, Xing Liu, Takayuki Okatani	Novel triplet domain translation network by leveraging real photos along with massive synthetic image pairs	They only focus on unstructured defects and do not support structured defects like image inpainting
9	Deep Image Prior	2018	Victor Lempitsky, Andrea Vedaldi, Dmitry Ulyanov	Deep neural network inherently resonates with low-level image statistics and thereby can be utilized as an image prior for blind image restoration without external training data	To restore in-the-wild images corrupted by mixed factors.

Low Level Image Restoration

S.No.	NAME OF THE JOURNAL	YEAR	NAME OF THE AUTHORS	PROS	CONS
10	Image Colorization: A Survey and Dataset	2020	Saeed Anwar, Md Tahir, Chongyi Li, Ajmal Mian, Fahad Shahbaz Khan, Abdul Wahab Muzaffar	• Colorization problem has been handled	• Can handle only single image colorization problem. • Only plain networks are handled.
11	Joint Transmission Map Estimation and Dehazing using Deep Networks	2019	He Zhang, Vishwanath Sindagi	• It provides an end-to-end learning framework, where the inherent transmission map and dehazed result are learned jointly from the loss function.	• Find a way to solve the ambiguity problem between image color and depth in dehazing. • Create a more robust model for dehazing in inhomogeneous atmosphere.
12	FFDNet: Toward a Fast and Flexible Solution for CNN based Image Denoising	2018	Kai Zhang, Wangmeng Zuo, Lei Zhang	• The ability to handle a wide range of noise level. • Faster speed than benchmark BM3D even on CPU without sacrificing denoising performance.	Only noise is handled any other aspect is not handled.

CHAPTER - 4

IMAGE DEFECTS

4.1 CLASSIFICATION OF IMAGE DEFECTS:

Pictures are liable to damage when they are not stored carefully. Images that are safely stored undergo less damage than the images that are left out in the open. These images that are left open in living spaces are subjected to stains, scratches, foldings and many other such defects. These defects are classified into two types.

a) Structured Defects

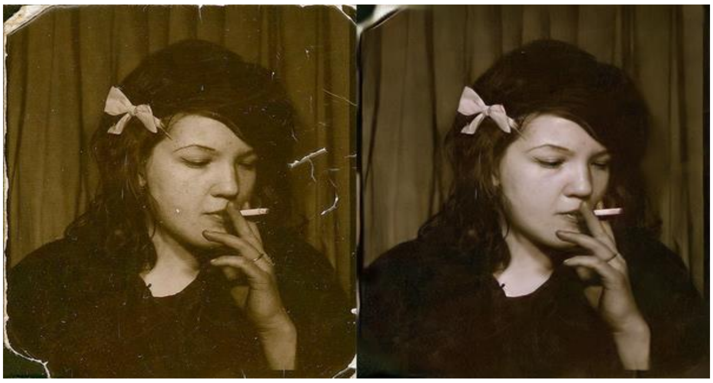
b) Unstructured Defects

4.1.1 STRUCTURED DEFECTS:

Structured defects occur commonly in a similar or recognizable way. Scratches and dust spots are also called as structured defects.

- *SCRATCHES:*

A scratch is a cut or damage to a surface of an image with something sharp or rough. Scratches on an image are formed due defects in old films or bad handling of an image. Scratches can also be considered as noise, as they have high contrast compared to their neighbors.



- *DUST SPOTS:*

When we snap a picture, if camera sensor isn't totally perfect, we will likely see dust spots in pictures. They are most prominently found when our device lens isn’t properly cleaned and also when our image is left with stains.

4.1.2 UNSTRUCTURED DEFECTS:

Unstructured defects are novel and don’t occur frequently. These types of defects are difficult to recognize at one glance. Image Noise, Blurring and Low resolution are some of the unstructured defects.

- *IMAGE NOISE:*

Image noise can be defined as a random variation in the brightness or color information of an image. This occurs due to internal and external factors, namely: background lighting, sensors and film used in the camera. The most common type of noise found in old images is “*Film grain*”. This is mainly because of sensor nonuniformities, which was quite common in old times due to the Camera and Film quality.

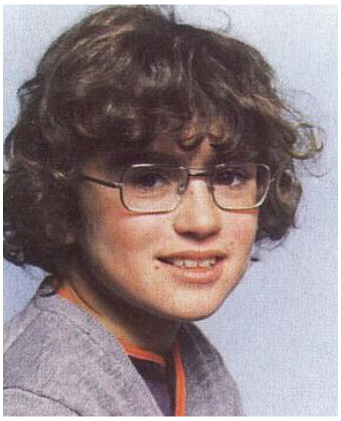
- *BLURRING:*

  The most common reason for blurred images is due to camera shake or sudden movement of an object while a photo is being taken. The camera quality in the past produced photos, which almost look blurred to eyes even today.

- *LOW RESOLUTION:*

  The level of detail in an image is referred to as image resolution. The image appears better at a higher resolution. A popular resolution type is “Pixel count”.

  In the past, the resolution of film frames available was much less than what we have today, and hence we find lesser resolution in old photos, which is also an aspect to be considered during restoration.



CHAPTER - 5

SYSTEM ANALYSIS

5.1 SYSTEM FUNCTIONALITY

Our application is a GUI-based application that implements few image restoration techniques on grayscale and color images without using existing image processing libraries. The features implemented in the application are given below:

- Inverse filtering
- Truncated inverse filtering
- Minimum mean square error (Weiner) filtering
- Constrained least-squares filtering

Additionally, the application provides the following features for ease of use:

- Open: opens a dialog box to select the input image
- Save: opens a dialog box to choose the location for saving the current image
- Compute PSNR/ SSIM: calculates the PSNR/ SSIM values for both input and restored image when the undegraded image is known
- Load/ Clear true image: provides an option to choose/ clear the undegraded image for computation of PSNR/ SSIM for the current input image
- Choose blur kernel: provides an option to choose a blur kernel from a predefined set of kernels.

PSNR: Peak signal-to-noise (PSNR) is that the ratio between the utmost possible power of an image and therefore the power of corrupting noise that affects the standard of its representation. To estimate the PSNR of a picture, it's necessary to match that image to a perfect clean image with the maximum possible power.

SSIM: The structural similarity index measure is a perception-based model that considers image degradation as perceived change in structural information and prediction of image quality.

5.2 TECHNOLOGY

5.2.1 WHAT IS PYTHON?

Python is a high-level programing language which is interpretive and also general purpose in nature. It was introduced in the year 1992 and was then further developed by Guido van Rossum. It includes featuring a particular design that declares code readability. It bears innumerable programming models. These models include object-oriented, imperative, functional, and procedural. It also has a library which is inclusive. The exponents of this programming language known as interpreters are handy for several operating systems. CPython, is a backing model of Python, which is an open-source software that marks a joint effort in the development of the model. Python and CPython are administered by the non-profit Python Software Foundation. “Python may be a using more than one paradigm, and thus called multi-paradigm programing language. Various varieties of paradigms are supported just by extensions, including design by logic programming a. Python has various features like dynamic name resolution, object-oriented programming, as well as structured programming, and a lot of its features support functional programming”. Listed below are the functional python libraries incorporated for the project development which is, IMAGE RECONSTRUCTION OF OLD DAMAGED PHOTOS:

1) Opencv2
2) Numpy
3) Pyqt

5.3 SYSTEM REQUIREMENTS

Software Requirements Specification (SRS) – Requirements that are essential for any software system with the entire information of that particular system that is to be developed. It includes cases which hold the information about the connections between the user and the software. Besides this, the Software Requirements

Specification (SRS) also includes non-functional requirements, which are those that have limitations for both the design and the implementation i.e., performance, quality or any of the design constraints. A business analyst also called as system analyst is in charge of the business needs of their customers and also the stakeholders to help solve business problems and then alongside, provide a better solution. Basically, any project has three types of requirements:

• BUSINESS REQUIREMENTS: According to the business terms, keeping the supply value in view, delivering what is needed the utmost is important.

• PRODUCT REQUIREMENTS: This type of requirement include properties that a particular system or any product need. It is one of the best ways to acquire business requirements also.

• PROCESS REQUIREMENTS: It contains the activities that are carried out by the developing team.

5.4 SYSTEM ARCHITECTURE

5.4.1. Software Requirements:

● Technology requirement is Machine Learning Deep Learning

● Python and its widely used libraries opencv2(for reading/writing of images and color space conversion only), NumPy (for array operations)

● Pyqt4 library (for GUI)

5.4.2. Hardware Requirements:

● 64-bit Operating System

● 1 TB hard disk

● 8 GB RAM

● Intel Core i3 or more processor

CHAPTER – 6

SYSTEM IMPLEMENTATION

6.1 IMPLEMENTATION

Following is the process flow diagram:

1. Data Acquisition
2. Preprocessing
3. Feature Extraction
4. Feature Discrimination
5. Decision

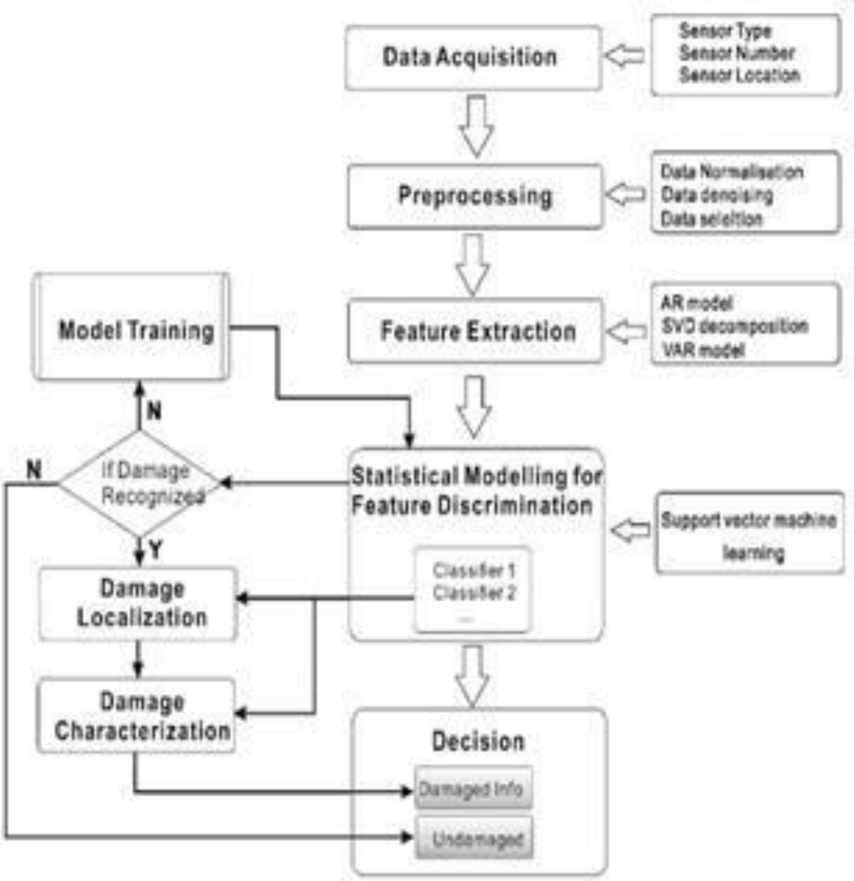


Fig-1 Flow diagram of process flow

1. **Data Acquisition:**

Data Acquisition measures bodily global situations and conditions along with electricity, sound, temperature and so on, that is performed via the use of diverse sensors which pattern the surrounding’s analog signals and transform them to digital signals the use of an analog to digital convertor. The resulting virtual numeric values can then be immediately manipulated via a laptop, allowing the evaluation, storage, and presentation of that information.

2. **Preprocessing:**

Preprocessing is the procedure of reworking our information earlier than including it to the set of rules. Records preprocessing is a method that won’t convert the data right into a smooth information set. In different phrases, whenever the info is accrued from distinct resources it's accumulated inside the uncooked layout which isn't always feasible for the evaluation.

3. **Feature Extraction:**

Feature Extraction ambitions to lessen the variety of capabilities in a dataset by way of growing new features from current ones after which deleting the original features. The received decreased set of capabilities need to then be able to summarize maximum of the statistics contained within the original set of capabilities.

4. **Feature Discrimination:**

Feature Discrimination reduces the wide variety of input variables whilst developing a predictive version. Filter-primarily based feature choice methods use statistical measures to achieve the correlation or dependence among enter variables in order to be filtered to decide the foremost applicable features.

5. **Decision:**

The educated model is then loaded and then a graphical consumer interface that forecast the broken and undamaged information.

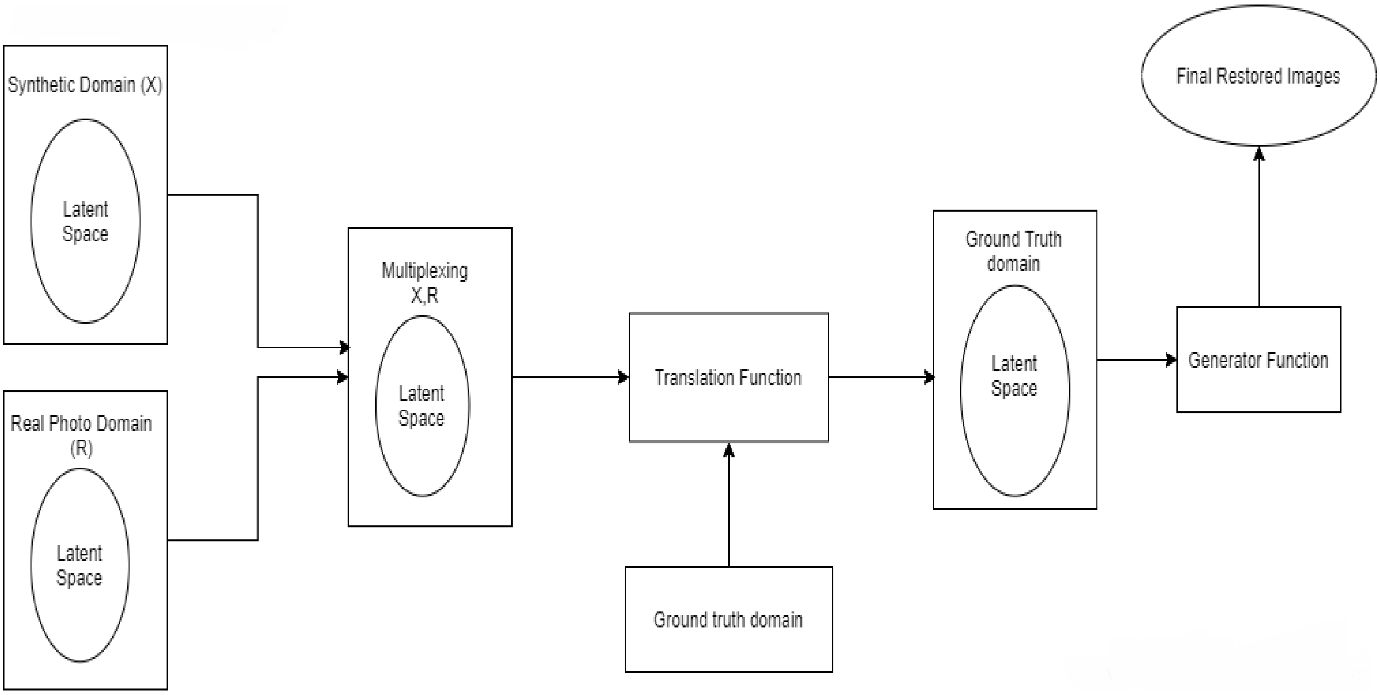
CHAPTER – 7

METHODOLOGY AND PROCESS FLOW

7.1 METHODOLOGY

For image restoration using old photos and real photos, the strategy is to perform mapping between the real photo domain and synthetic photo domain. These mappings occur exactly at the regions of defects of damaged photos and are restored with the help of a decoder. The final image would have higher resolution, lesser film grain and covered scratches.

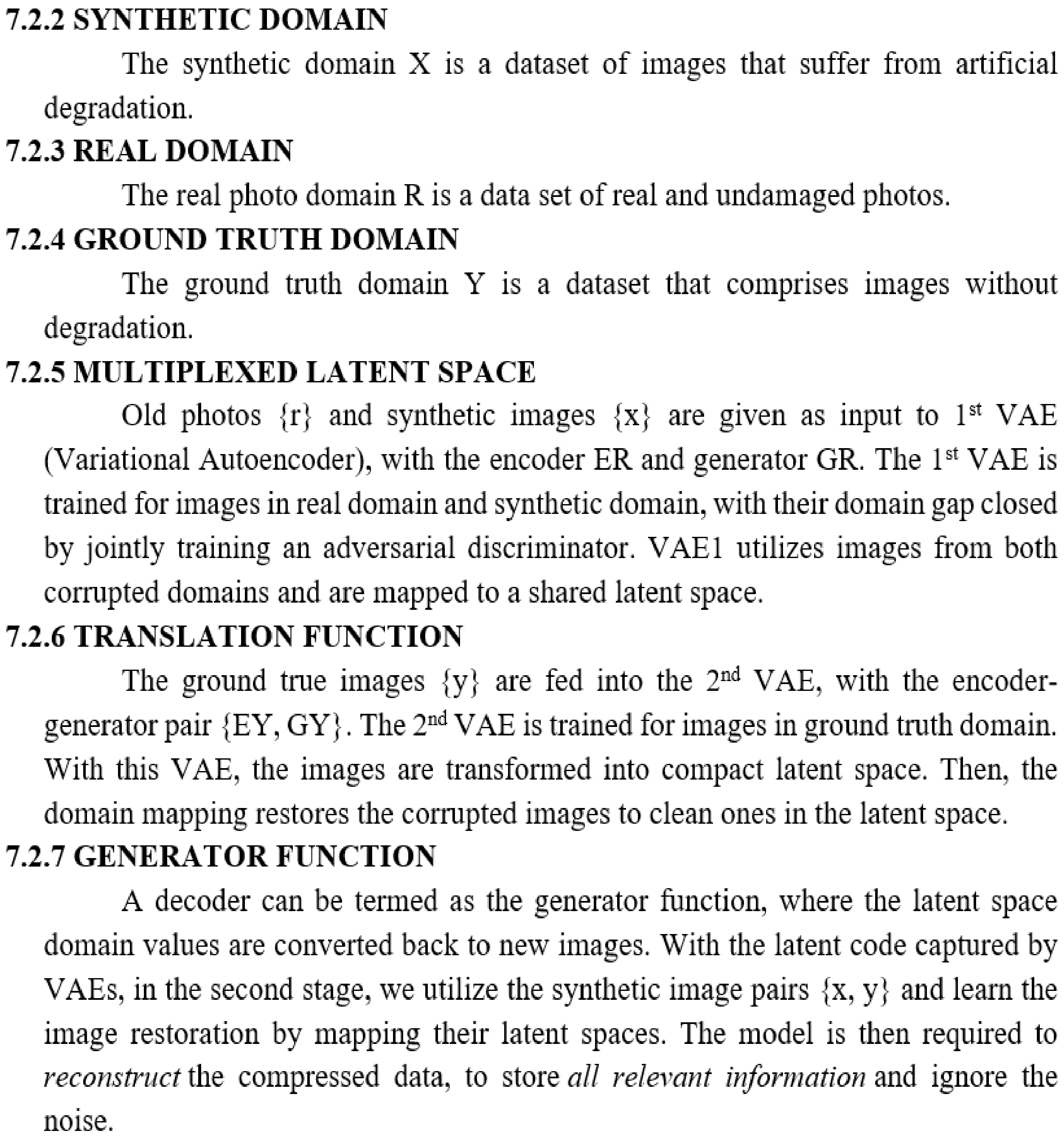
7.1.1 BLOCK DIAGRAM



7.2 COMPONENTS OF PROPOSED SYSTEM

7.2.1 LATENT SPACE

The word “latent” means “hidden”. When the model learns through a certain data parameter, the *spatial property* of the image is first *reduced* before it is ultimately increased. So, when the dimensionality is reduced, this is a form of lossy compression. This value of compression allows us to get rid of any irrelevant information, and only focus on the most important features. **This 'compressed state' is called *Latent Space Representation of data*.**



7.2.2 SYNTHETIC DOMAIN

The synthetic domain X is a dataset of images that suffer from artificial degradation.

7.2.3 REAL DOMAIN

The real photo domain R is a data set of real and undamaged photos.

7.2.4 GROUND TRUTH DOMAIN

The ground truth domain Y is a dataset that comprises images without degradation.

7.2.5 MULTIPLEXED LATENT SPACE

Old photos {r} and synthetic images {x} are given as input to 1st VAE (Variational Autoencoder), with the encoder ER and generator GR. The 1st VAE is trained for images in real domain and synthetic domain, with their domain gap closed by jointly training an adversarial discriminator. VAE1 utilizes images from both corrupted domains and are mapped to a shared latent space.

7.2.6 TRANSLATION FUNCTION

The ground true images {y} are fed into the 2nd VAE, with the encoder-generator pair {EY, GY}. The 2nd VAE is trained for images in ground truth domain. With this VAE, the images are transformed into compact latent space. Then, the domain mapping restores the corrupted images to clean ones in the latent space.

7.2.7 GENERATOR FUNCTION

A decoder can be termed as the generator function, where the latent space domain values are converted back to new images. With the latent code captured by VAEs, in the second stage, we utilize the synthetic image pairs {x, y} and learn the image restoration by mapping their latent spaces. The model is then required to *reconstruct* the compressed data, to store *all relevant information* and ignore the noise.

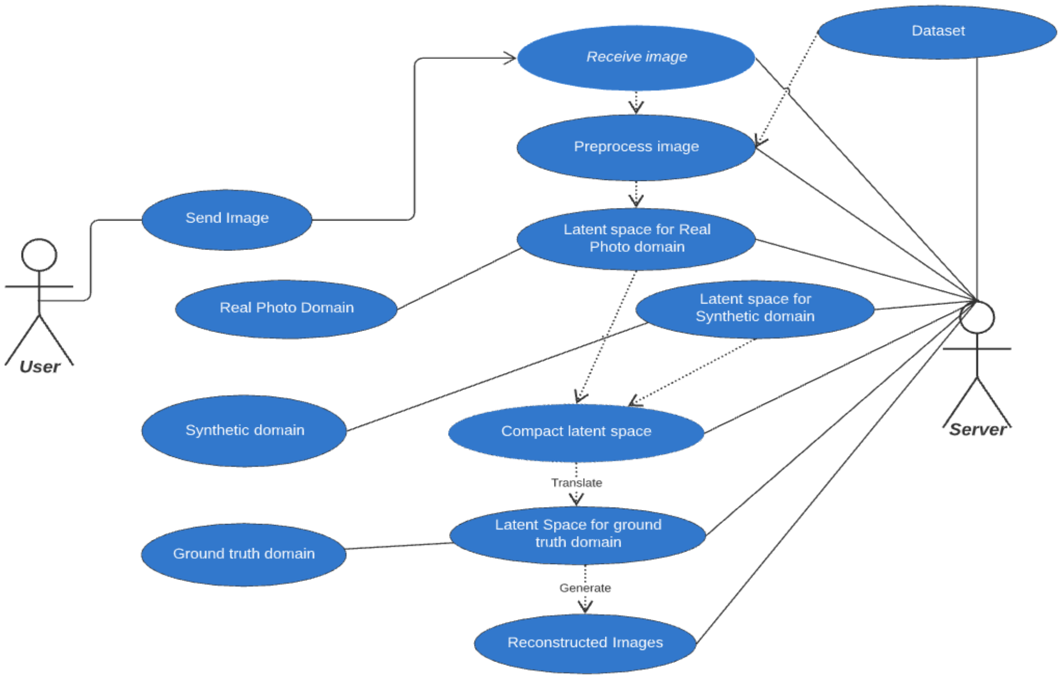
CHAPTER - 8
SYSTEM DESIGN

8.1 USE CASE DIAGRAM

8.1.1 DEFINITION

A key concept of use case modeling here is that it helps us design the Image reconstruction of old damaged photos from the end user's perspective. It is an effective technique for communicating the application behavior in the user's terms by specifying all externally visible system behavior.

8.1.2 USE CASE DIAGRAM



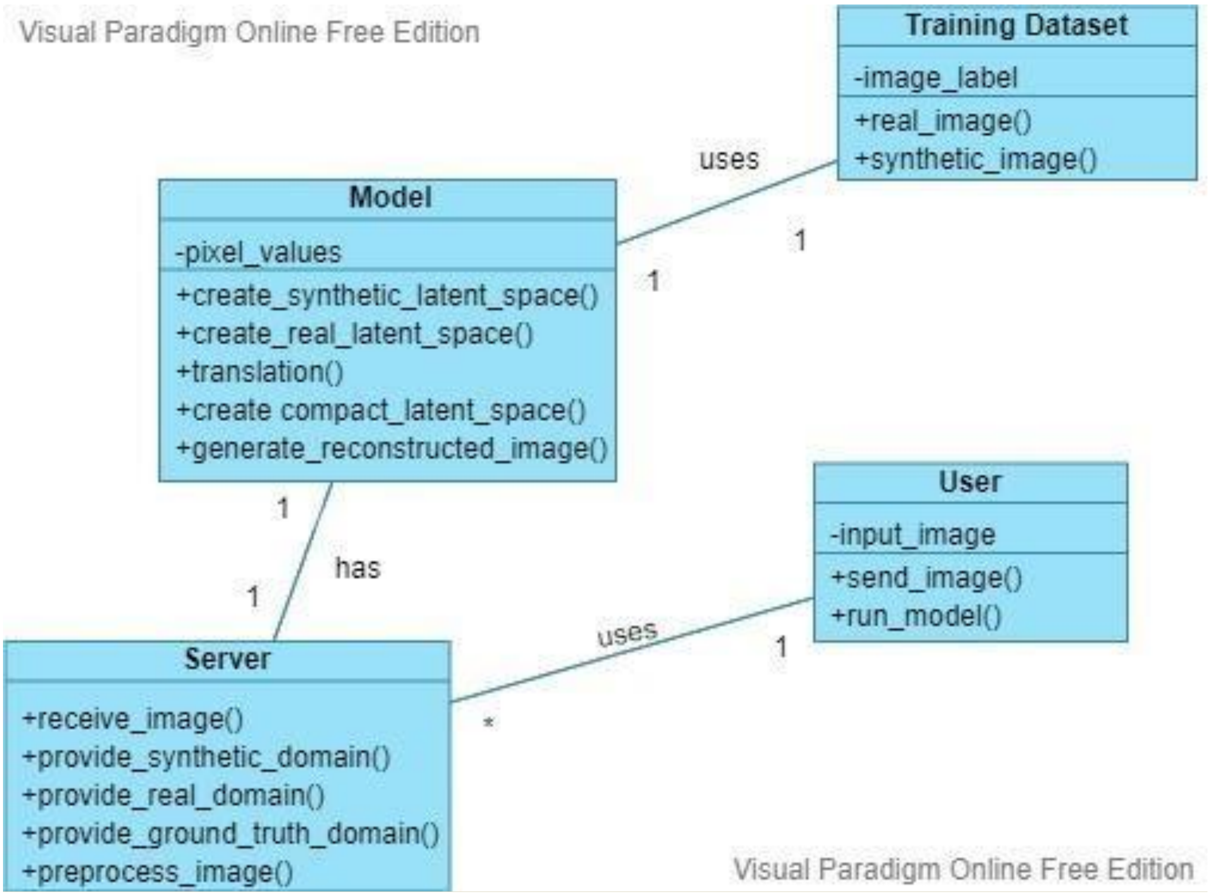
USE CASE DIAGRAM

8.2 CLASS DIAGRAM

8.2.1 DEFINITION

The UML class diagram below is a type of static structure diagram that describes the structure of an Image reconstruction of old damaged photos by showing the classes such as model, server, user, training dataset, their attributes, operations (or methods), and the relationships among objects.

8.3.2 CLASS DIAGRAM



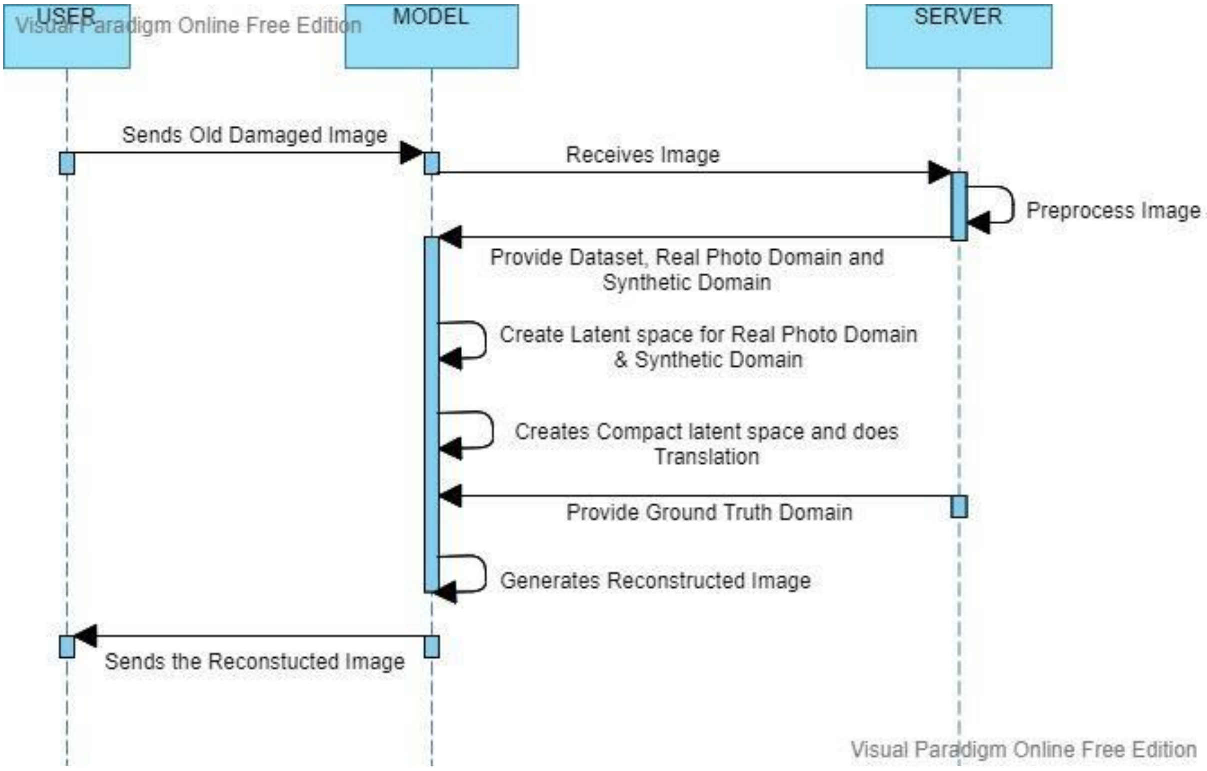
Visual Paradigm Online Free Edition
Training Dataset
-image_label
+real_image()
+synthetic_image()
uses
1
1
Model
-pixel_values
+create_synthetic_latent_space()
+create_real_latent_space()
+translation()
+create compact_latent_space()
+generate_reconstructed_image()
1
has
1
User
-input_image
+send_image()
+run_model()
uses
1
*
Server
+receive_image()
+provide_synthetic_domain()
+provide_real_domain()
+provide_ground_truth_domain()
+preprocess_image()
Visual Paradigm Online Free Edition

8.4 SEQUENCE DIAGRAM

8.4.1 DEFINITION

A UML sequence diagram represents the interaction between objects in sequential order (i.e., the order in which the interactions occur from uploading the dataset to retrieving the response). The terms ‘event diagrams’ or ‘event scenarios’ are used to refer to a sequence diagram. Sequence diagrams describe the order in which the objects in a system function.

8.4.2 SEQUENCE DIAGRAM


USER
Visual Paradigm Online Free Edition
MODEL
SERVER
Sends Old Damaged Image
Receives Image
Preprocess Image
Provide Dataset, Real Photo Domain and Synthetic Domain
Create Latent space for Real Photo Domain & Synthetic Domain
Creates Compact latent space and does Translation
Provide Ground Truth Domain
Generates Reconstructed Image
Sends the Reconstucted Image
Visual Paradigm Online Free Edition

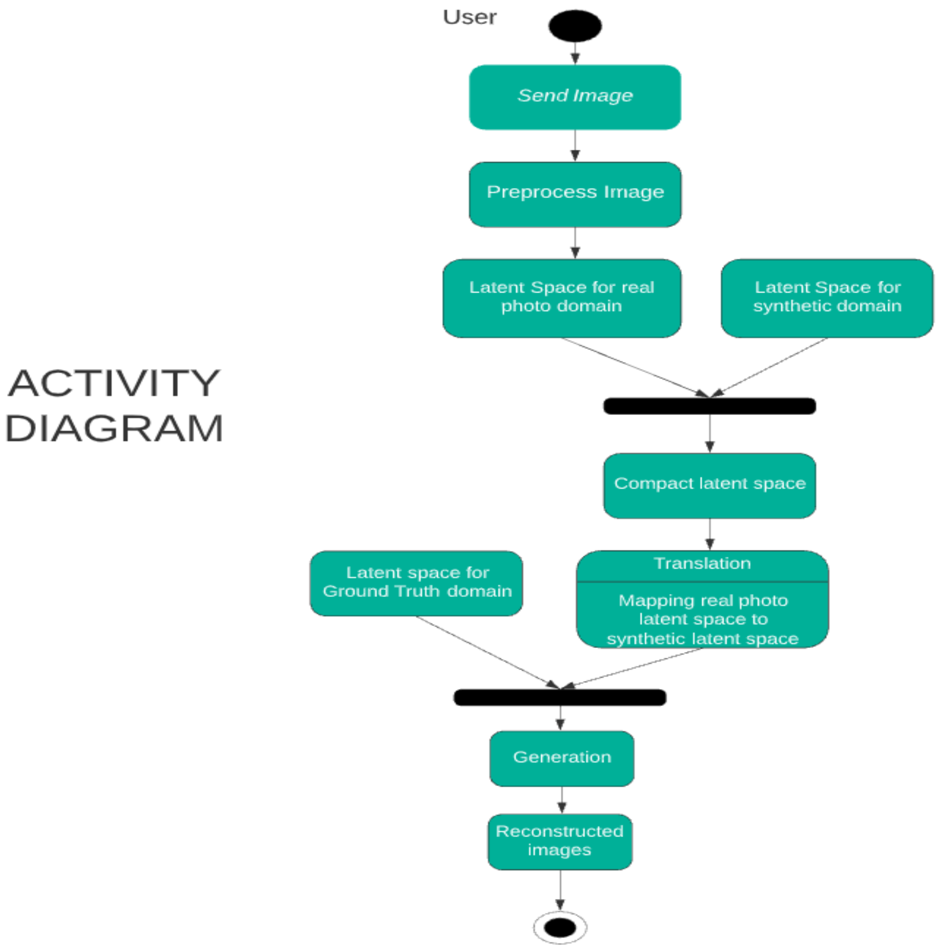
8.5 ACTIVITY DIAGRAM

8.5.1 DEFINITION

UML Activity diagrams helps to describe the dynamic aspects of the system. It is a flowchart for representing flow from one activity to another activity. It can be narrated as an operation of the system. The flow can be branched, sequential or concurrent.

It is used for Modeling workflow, business requirements, high-level understanding of the system's functionalities, investing business requirements at a later stage.

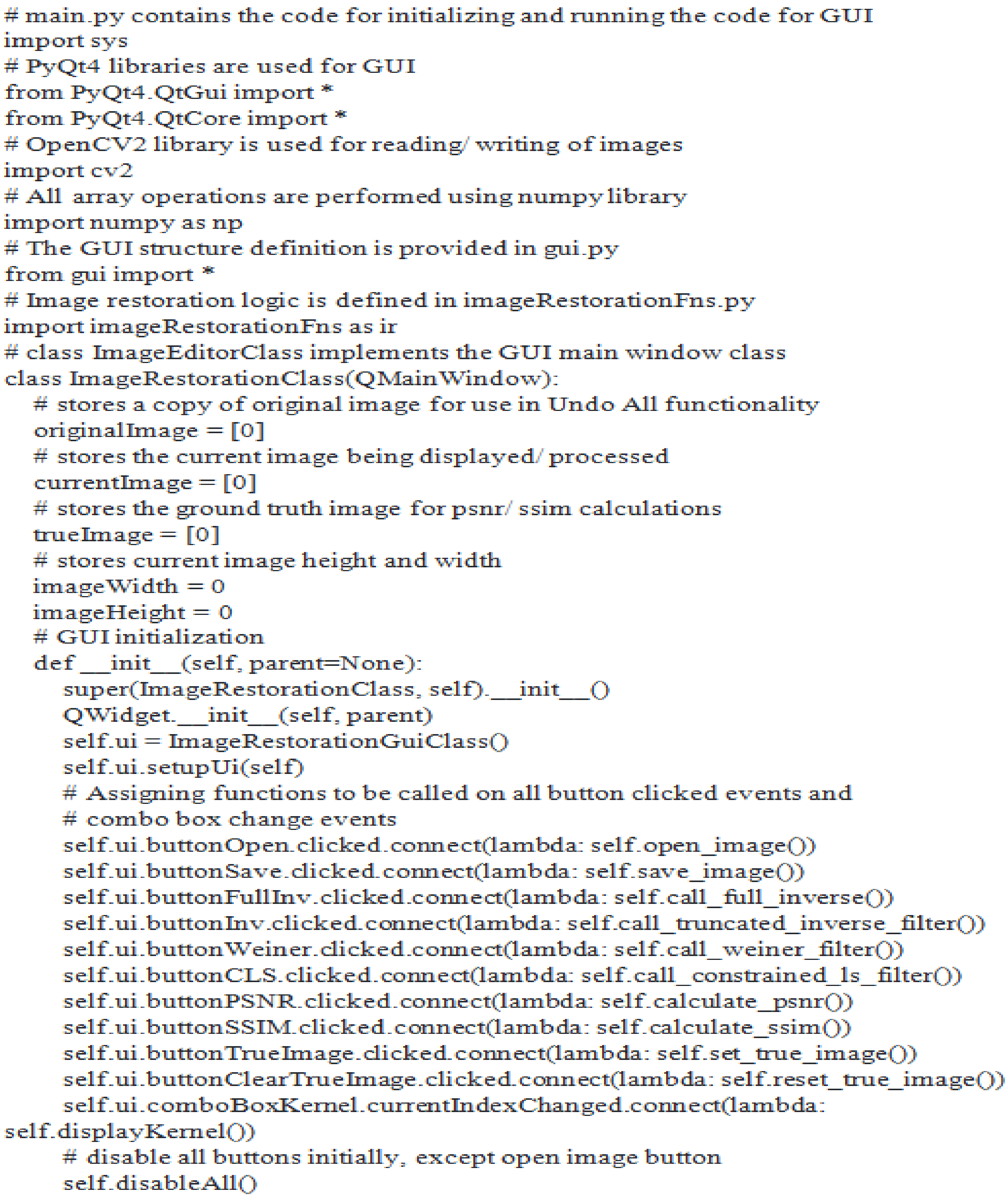
8.5.2 ACTIVITY DIAGRAM


ACTIVITY DIAGRAM
User
Send Image
Preprocess Image
Latent Space for real photo domain
Latent Space for synthetic domain
Compact latent space
Latent space for Ground Truth domain
Translation
Mapping real photo latent space to synthetic latent space
Generation
Reconstructed images

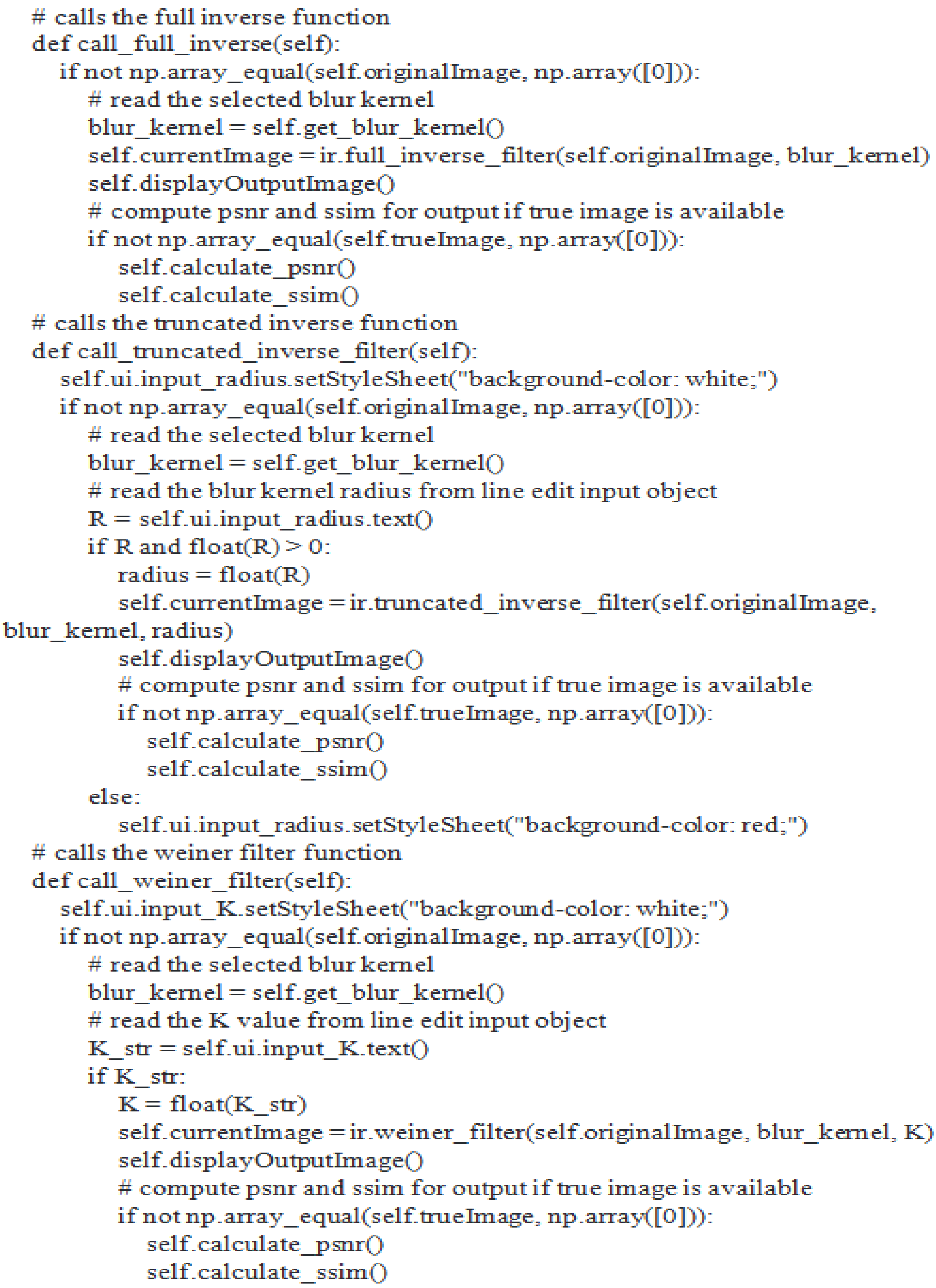
CHAPTER - 9

IMPLEMENTATION

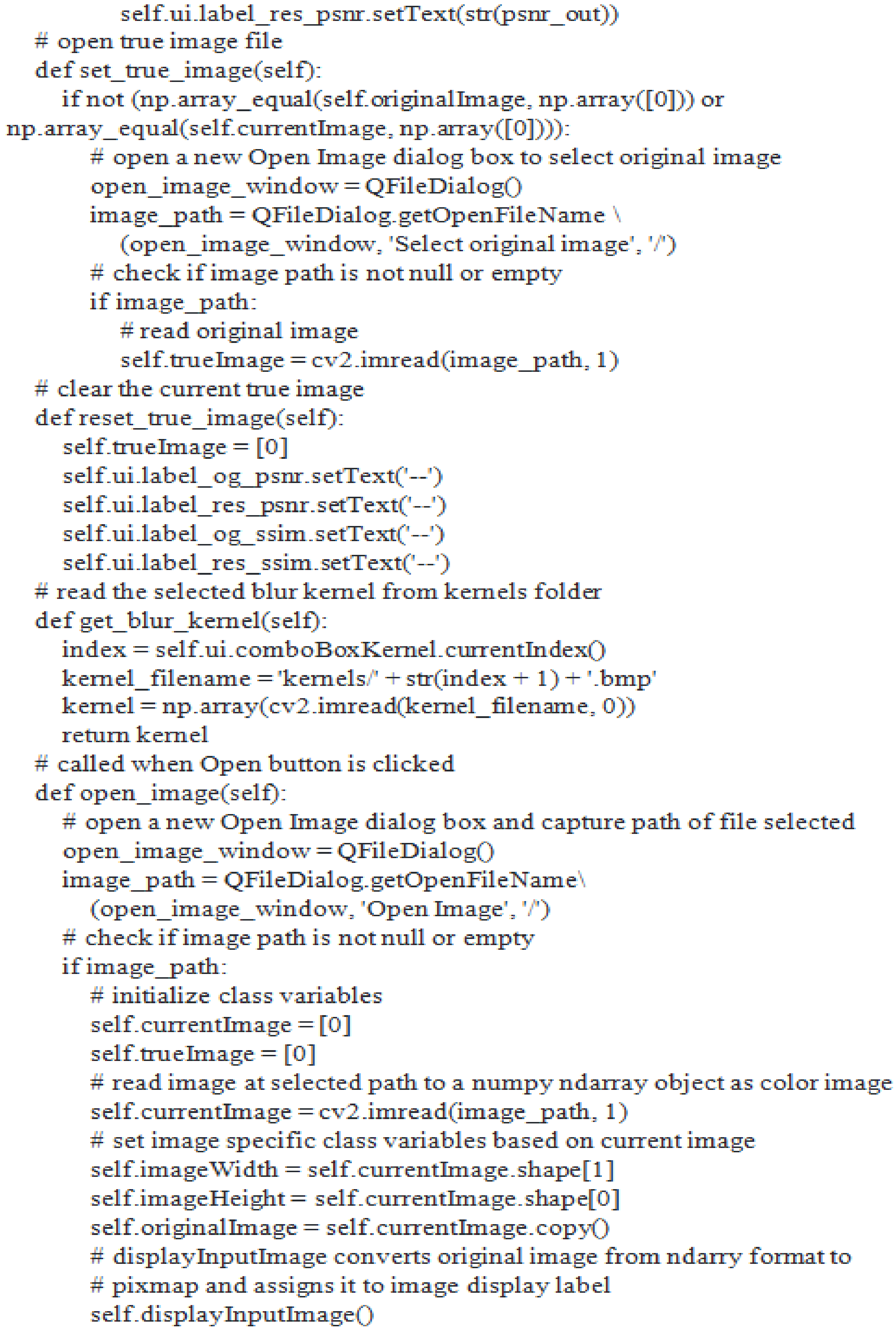
9.1 CODE IMPLEMENTATION



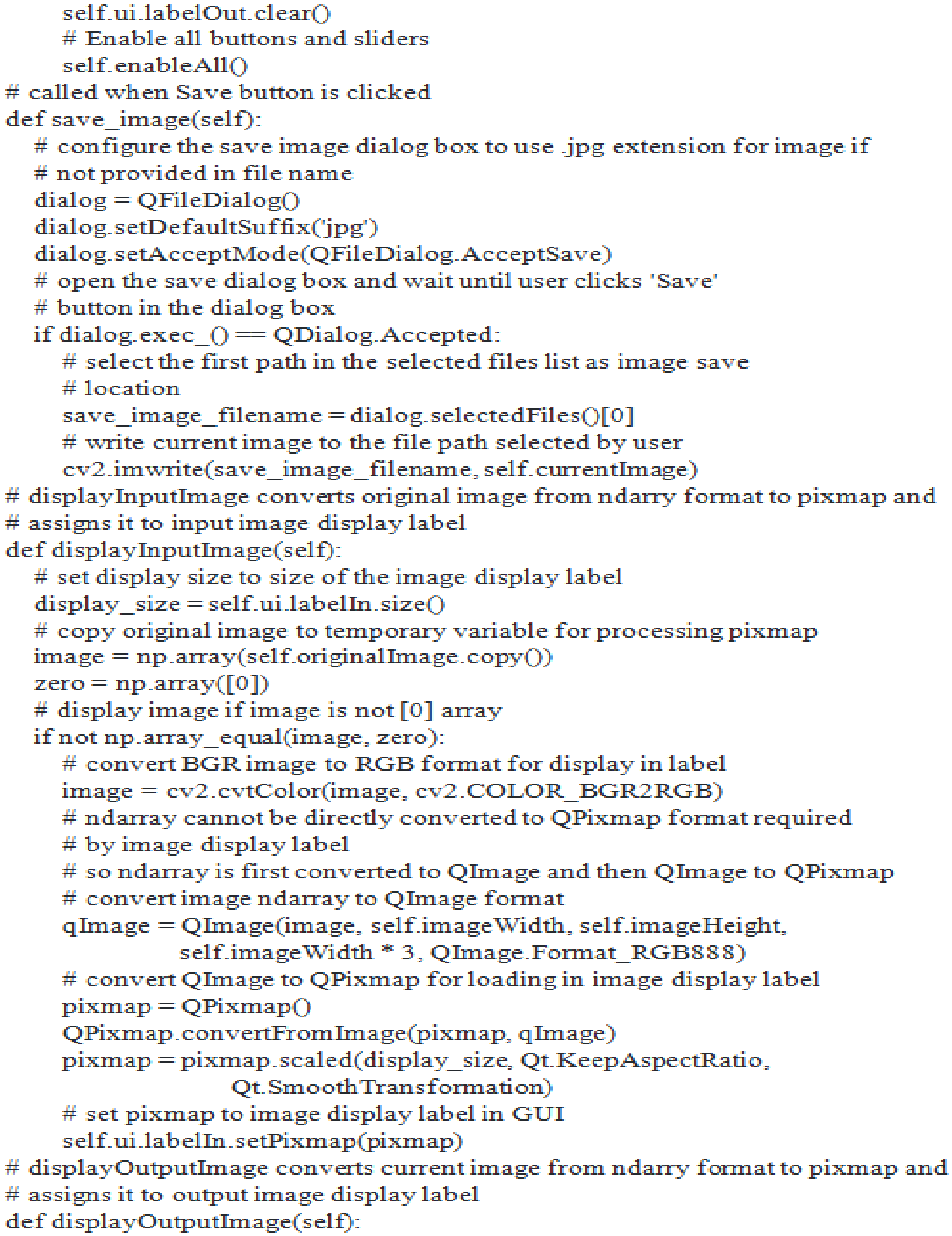
```
# main.py contains the code for initializing and running the code for GUI
import sys
# PyQt4 libraries are used for GUI
from PyQt4.QtGui import *
from PyQt4.QtCore import *
# OpenCV2 library is used for reading/ writing of images
import cv2
# All array operations are performed using numpy library
import numpy as np
# The GUI structure definition is provided in gui.py
from gui import *
# Image restoration logic is defined in imageRestorationFns.py
import imageRestorationFns as ir
# class ImageEditorClass implements the GUI main window class
class ImageRestorationClass(QMainWindow):
    # stores a copy of original image for use in Undo All functionality
    originalImage = [0]
    # stores the current image being displayed/ processed
    currentImage = [0]
    # stores the ground truth image for psnr/ ssim calculations
    trueImage = [0]
    # stores current image height and width
    imageWidth = 0
    imageHeight = 0
    # GUI initialization
    def __init__(self, parent=None):
        super(ImageRestorationClass, self).__init__()
        QWidget.__init__(self, parent)
        self.ui = ImageRestorationGuiClass()
        self.ui.setupUi(self)
        # Assigning functions to be called on all button clicked events and
        # combo box change events
        self.ui.buttonOpen.clicked.connect(lambda: self.open_image())
        self.ui.buttonSave.clicked.connect(lambda: self.save_image())
        self.ui.buttonFullInv.clicked.connect(lambda: self.call_full_inverse())
        self.ui.buttonInv.clicked.connect(lambda: self.call_truncated_inverse_filter())
        self.ui.buttonWeiner.clicked.connect(lambda: self.call_weiner_filter())
        self.ui.buttonCLS.clicked.connect(lambda: self.call_constrained_ls_filter())
        self.ui.buttonPSNR.clicked.connect(lambda: self.calculate_psnr())
        self.ui.buttonSSIM.clicked.connect(lambda: self.calculate_ssim())
        self.ui.buttonTrueImage.clicked.connect(lambda: self.set_true_image())
        self.ui.buttonClearTrueImage.clicked.connect(lambda: self.reset_true_image())
        self.ui.comboBoxKernel.currentIndexChanged.connect(lambda:
self.displayKernel())
        # disable all buttons initially, except open image button
        self.disableAll()
```



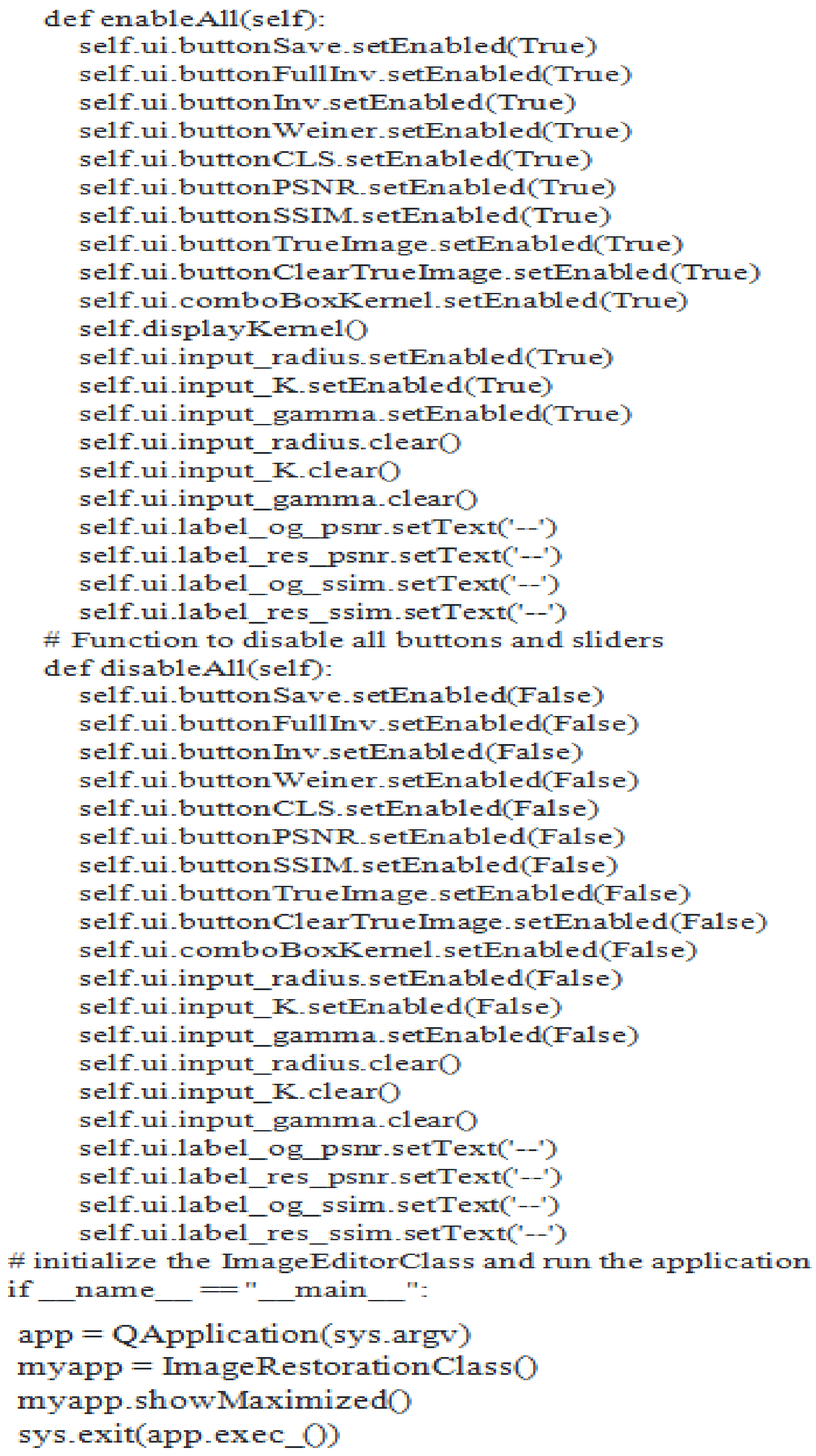
```
    # calls the full inverse function
    def call_full_inverse(self):
        if not np.array_equal(self.originalImage, np.array([0])):
            # read the selected blur kernel
            blur_kernel = self.get_blur_kernel()
            self.currentImage = ir.full_inverse_filter(self.originalImage, blur_kernel)
            self.displayOutputImage()
            # compute psnr and ssim for output if true image is available
            if not np.array_equal(self.trueImage, np.array([0])):
                self.calculate_psnr()
                self.calculate_ssim()
    # calls the truncated inverse function
    def call_truncated_inverse_filter(self):
        self.ui.input_radius.setStyleSheet("background-color: white;")
        if not np.array_equal(self.originalImage, np.array([0])):
            # read the selected blur kernel
            blur_kernel = self.get_blur_kernel()
            # read the blur kernel radius from line edit input object
            R = self.ui.input_radius.text()
            if R and float(R) > 0:
                radius = float(R)
                self.currentImage = ir.truncated_inverse_filter(self.originalImage,
blur_kernel, radius)
                self.displayOutputImage()
                # compute psnr and ssim for output if true image is available
                if not np.array_equal(self.trueImage, np.array([0])):
                    self.calculate_psnr()
                    self.calculate_ssim()
            else:
                self.ui.input_radius.setStyleSheet("background-color: red;")
    # calls the weiner filter function
    def call_weiner_filter(self):
        self.ui.input_K.setStyleSheet("background-color: white;")
        if not np.array_equal(self.originalImage, np.array([0])):
            # read the selected blur kernel
            blur_kernel = self.get_blur_kernel()
            # read the K value from line edit input object
            K_str = self.ui.input_K.text()
            if K_str:
                K = float(K_str)
                self.currentImage = ir.weiner_filter(self.originalImage, blur_kernel, K)
                self.displayOutputImage()
                # compute psnr and ssim for output if true image is available
                if not np.array_equal(self.trueImage, np.array([0])):
                    self.calculate_psnr()
                    self.calculate_ssim()
```



```
                self.ui.label_res_psnr.setText(str(psnr_out))
    # open true image file
    def set_true_image(self):
        if not (np.array_equal(self.originalImage, np.array([0])) or
np.array_equal(self.currentImage, np.array([0]))):
            # open a new Open Image dialog box to select original image
            open_image_window = QFileDialog()
            image_path = QFileDialog.getOpenFileName \
                (open_image_window, 'Select original image', '/')
            # check if image path is not null or empty
            if image_path:
                # read original image
                self.trueImage = cv2.imread(image_path, 1)
    # clear the current true image
    def reset_true_image(self):
        self.trueImage = [0]
        self.ui.label_og_psnr.setText('--')
        self.ui.label_res_psnr.setText('--')
        self.ui.label_og_ssim.setText('--')
        self.ui.label_res_ssim.setText('--')
    # read the selected blur kernel from kernels folder
    def get_blur_kernel(self):
        index = self.ui.comboBoxKernel.currentIndex()
        kernel_filename = 'kernels/' + str(index + 1) + '.bmp'
        kernel = np.array(cv2.imread(kernel_filename, 0))
        return kernel
    # called when Open button is clicked
    def open_image(self):
        # open a new Open Image dialog box and capture path of file selected
        open_image_window = QFileDialog()
        image_path = QFileDialog.getOpenFileName\
            (open_image_window, 'Open Image', '/')
        # check if image path is not null or empty
        if image_path:
            # initialize class variables
            self.currentImage = [0]
            self.trueImage = [0]
            # read image at selected path to a numpy ndarray object as color image
            self.currentImage = cv2.imread(image_path, 1)
            # set image specific class variables based on current image
            self.imageWidth = self.currentImage.shape[1]
            self.imageHeight = self.currentImage.shape[0]
            self.originalImage = self.currentImage.copy()
            # displayInputImage converts original image from ndarry format to
            # pixmap and assigns it to image display label
            self.displayInputImage()
```



```
        self.ui.labelOut.clear()
        # Enable all buttons and sliders
        self.enableAll()
# called when Save button is clicked
def save_image(self):
    # configure the save image dialog box to use .jpg extension for image if
    # not provided in file name
    dialog = QFileDialog()
    dialog.setDefaultSuffix('jpg')
    dialog.setAcceptMode(QFileDialog.AcceptSave)
    # open the save dialog box and wait until user clicks 'Save'
    # button in the dialog box
    if dialog.exec_() == QDialog.Accepted:
        # select the first path in the selected files list as image save
        # location
        save_image_filename = dialog.selectedFiles()[0]
        # write current image to the file path selected by user
        cv2.imwrite(save_image_filename, self.currentImage)
# displayInputImage converts original image from ndarry format to pixmap and
# assigns it to input image display label
def displayInputImage(self):
    # set display size to size of the image display label
    display_size = self.ui.labelIn.size()
    # copy original image to temporary variable for processing pixmap
    image = np.array(self.originalImage.copy())
    zero = np.array([0])
    # display image if image is not [0] array
    if not np.array_equal(image, zero):
        # convert BGR image to RGB format for display in label
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        # ndarray cannot be directly converted to QPixmap format required
        # by image display label
        # so ndarray is first converted to QImage and then QImage to QPixmap
        # convert image ndarray to QImage format
        qImage = QImage(image, self.imageWidth, self.imageHeight,
                        self.imageWidth * 3, QImage.Format_RGB888)
        # convert QImage to QPixmap for loading in image display label
        pixmap = QPixmap()
        QPixmap.convertFromImage(pixmap, qImage)
        pixmap = pixmap.scaled(display_size, Qt.KeepAspectRatio,
                               Qt.SmoothTransformation)
        # set pixmap to image display label in GUI
        self.ui.labelIn.setPixmap(pixmap)
# displayOutputImage converts current image from ndarry format to pixmap and
# assigns it to output image display label
def displayOutputImage(self):
        # set display size to size of the image display label
        display_size = self.ui.labelOut.size()
        # copy current image to temporary variable for processing pixmap
        image = np.array(self.currentImage.copy())
        zero = np.array([0])
        # display image if image is not [0] array
        if not np.array_equal(image, zero):
            # convert BGR image to RGB format for display in label
            image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
            # ndarray cannot be directly converted to QPixmap format required
            # by image display label
            # so ndarray is first converted to QImage and then QImage to QPixmap
            # convert image ndarray to QImage format
            qImage = QImage(image, self.imageWidth, self.imageHeight,
                            self.imageWidth * 3, QImage.Format_RGB888)
            # convert QImage to QPixmap for loading in image display label
            pixmap = QPixmap()
            QPixmap.convertFromImage(pixmap, qImage)
            pixmap = pixmap.scaled(display_size, Qt.KeepAspectRatio,
                                   Qt.SmoothTransformation)
            # set pixmap to image display label in GUI
            self.ui.labelOut.setPixmap(pixmap)
    # displayKernel converts selected kernel image from ndarry format to pixmap and
    # assigns it to kernel display label
    def displayKernel(self):
        # set display size to size of the kernel display label
        display_size = self.ui.labelKernelDisplay.size()
        # copy kernel image to temporary variable for processing pixmap
        kernel = np.array(self.get_blur_kernel())
        zero = np.array([0])
        # display image if kernel is not [0] array
        if not np.array_equal(kernel, zero):
            # ndarray cannot be directly converted to QPixmap format required
            # by kernel display label
            # so ndarray is first converted to QImage and then QImage to QPixmap
            # convert kernel ndarray to QImage format
            qImage = QImage(kernel, kernel.shape[1], kernel.shape[0], kernel.shape[1],
QImage.Format_Indexed8)
            # convert QImage to QPixmap for loading in image display label
            pixmap = QPixmap()
            QPixmap.convertFromImage(pixmap, qImage)
            pixmap = pixmap.scaled(display_size, Qt.KeepAspectRatio,
                                   Qt.SmoothTransformation)
            # set pixmap to kernel display label in GUI
            self.ui.labelKernelDisplay.setPixmap(pixmap)
    # Function to enable all buttons and sliders
```



```
    def enableAll(self):
        self.ui.buttonSave.setEnabled(True)
        self.ui.buttonFullInv.setEnabled(True)
        self.ui.buttonInv.setEnabled(True)
        self.ui.buttonWeiner.setEnabled(True)
        self.ui.buttonCLS.setEnabled(True)
        self.ui.buttonPSNR.setEnabled(True)
        self.ui.buttonSSIM.setEnabled(True)
        self.ui.buttonTrueImage.setEnabled(True)
        self.ui.buttonClearTrueImage.setEnabled(True)
        self.ui.comboBoxKernel.setEnabled(True)
        self.displayKernel()
        self.ui.input_radius.setEnabled(True)
        self.ui.input_K.setEnabled(True)
        self.ui.input_gamma.setEnabled(True)
        self.ui.input_radius.clear()
        self.ui.input_K.clear()
        self.ui.input_gamma.clear()
        self.ui.label_og_psnr.setText('--')
        self.ui.label_res_psnr.setText('--')
        self.ui.label_og_ssim.setText('--')
        self.ui.label_res_ssim.setText('--')
    # Function to disable all buttons and sliders
    def disableAll(self):
        self.ui.buttonSave.setEnabled(False)
        self.ui.buttonFullInv.setEnabled(False)
        self.ui.buttonInv.setEnabled(False)
        self.ui.buttonWeiner.setEnabled(False)
        self.ui.buttonCLS.setEnabled(False)
        self.ui.buttonPSNR.setEnabled(False)
        self.ui.buttonSSIM.setEnabled(False)
        self.ui.buttonTrueImage.setEnabled(False)
        self.ui.buttonClearTrueImage.setEnabled(False)
        self.ui.comboBoxKernel.setEnabled(False)
        self.ui.input_radius.setEnabled(False)
        self.ui.input_K.setEnabled(False)
        self.ui.input_gamma.setEnabled(False)
        self.ui.input_radius.clear()
        self.ui.input_K.clear()
        self.ui.input_gamma.clear()
        self.ui.label_og_psnr.setText('--')
        self.ui.label_res_psnr.setText('--')
        self.ui.label_og_ssim.setText('--')
        self.ui.label_res_ssim.setText('--')
# initialize the ImageEditorClass and run the application
if __name__ == "__main__":

 app = QApplication(sys.argv)
 myapp = ImageRestorationClass()
 myapp.showMaximized()
 sys.exit(app.exec_())
```

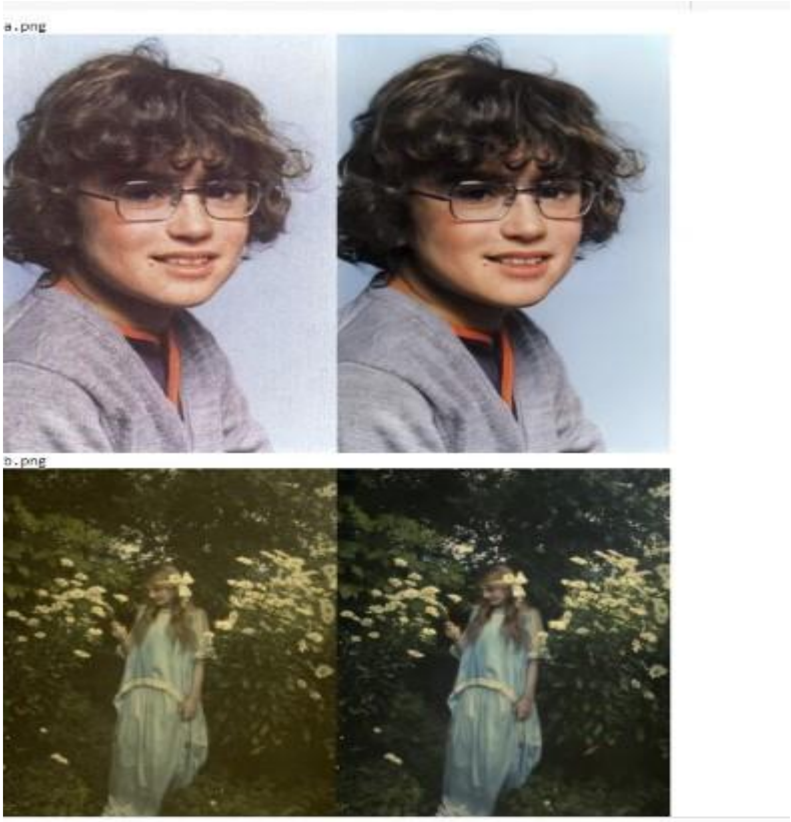
CHAPTER - 10

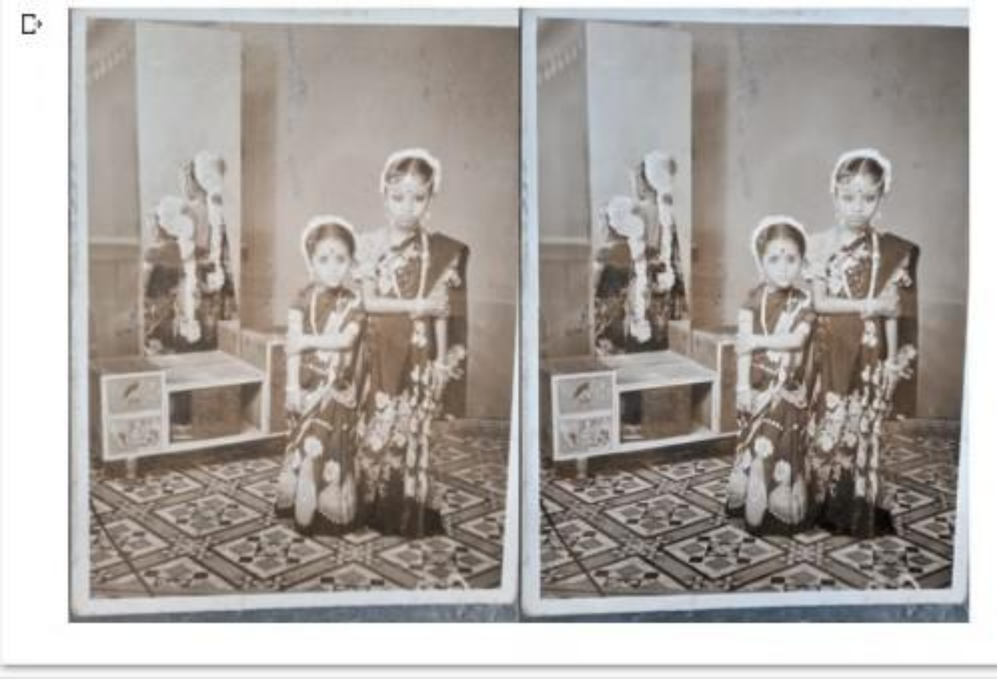
TESTING AND RESULTS

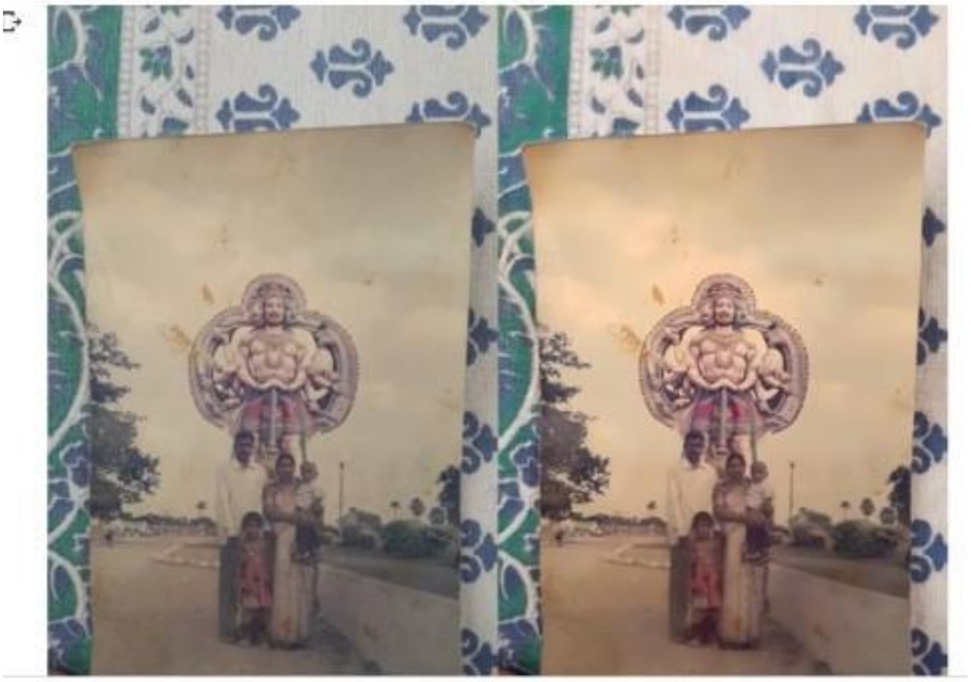
10.1 TEST CASES

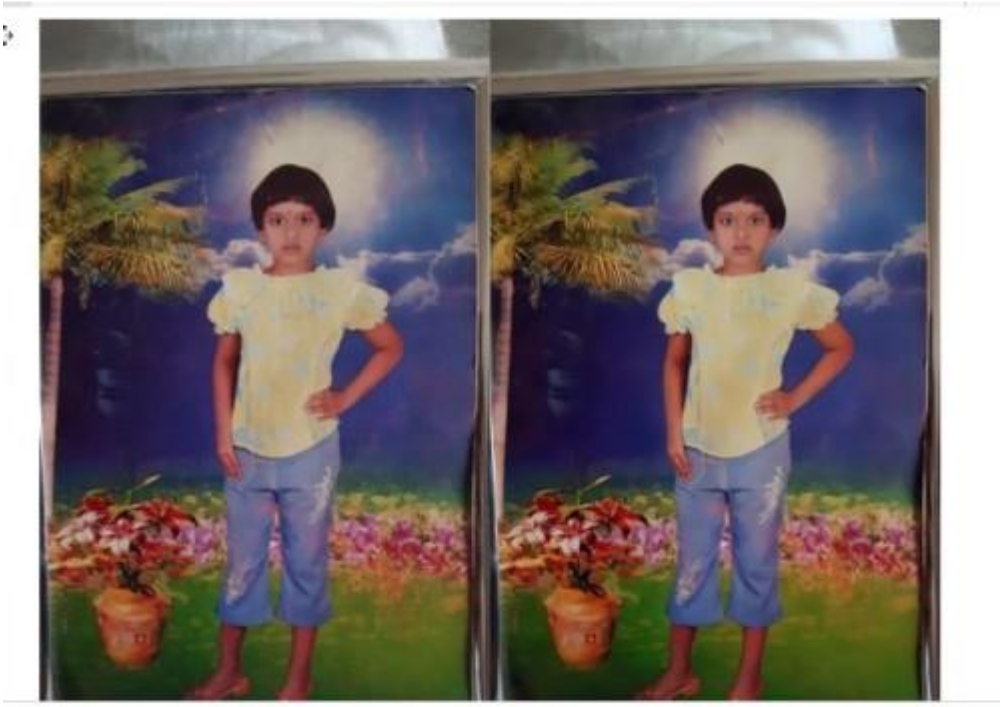
10.2 RESULTS

10.2.1 RESULTS FOR IMAGES WITH UNSTRUCTURED DEFECTS

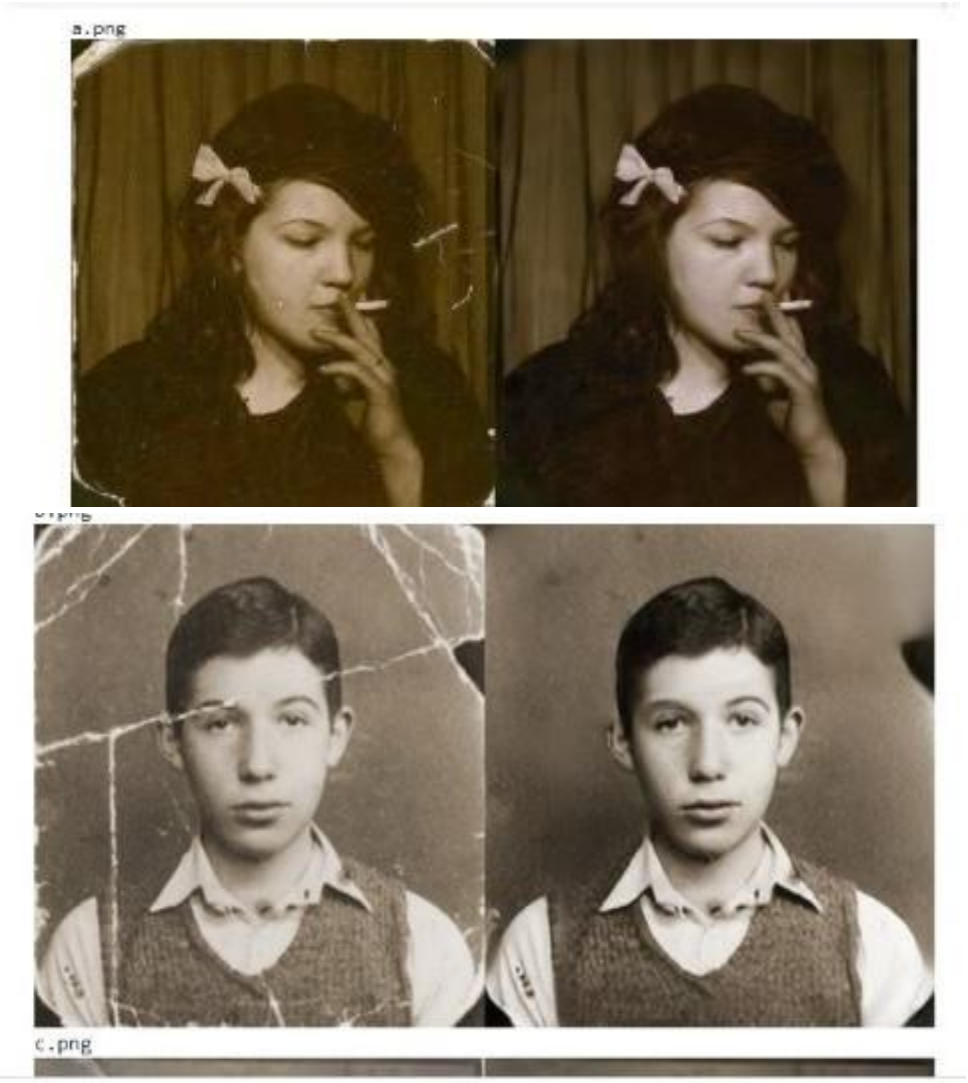








10.2.2 RESULTS FOR IMAGES WITH STRUCTURED DEFECTS



CHAPTER - 11
CONCLUSION

11.1 CONCLUSION

The best image restoration results were obtained using a constrained least square filter on images degraded by blurring and noise. Hence, “we can automatically and instantly repair old photos for those who wish to bring them back to life with more accuracy and less degradation”. Our project helps to restore old damaged photos with high accuracy and less time. Also, it was observed that imperfections in kernel estimation and linear degradation model approximation can affect the results of restoration considerably.

11.2 FUTURE SCOPE

It can be used in Medical Field for scanning purposes like CT scan and MRI Scans. Our method demonstrates good performance in restoring severely degraded old photos. However, our method cannot handle complex shading. This is because our dataset contains very few old photos with such defects. One could possibly address this limitation using our framework by explicitly considering the shading effects during synthesis or adding more such photos as training data. As an extension to our model, in the future we can work with reconstruction of damaged MRI scan images, space research images and old films to enhance their resolution. It will help in surveillance purposes also to make sure a much better data can be retrieved from collected data.

CHAPTER - 12
BIBLIOGRAPHY

12.1 REFERENCES

- F. Stanco, G. Ramponi, and A. De Polo, “Towards the automated restoration of old photographic prints: a survey,” in The IEEE Region 8 EUROCON 2003. Computer as a Tool.,vol. 2. IEEE, 2003, pp. 370–374.
- V. Bruni and D. Vitulano, “A generalized model for scratch detection,” IEEE transactions on image processing, vol. 13, no. 1, pp. 44–50, 2004.
- R.-C. Chang, Y.-L. Sie, S.-M. Chou, and T. K. Shih, “Photo defect detection for image inpainting,” in Seventh IEEE International Symposium on Multimedia (ISM’05). IEEE, 2005, pp. 5–pp.
- I. Giakoumis, N. Nikolaidis, and I. Pitas, “Digital image processing techniques for the detection and removal of cracks in digitized paintings,” IEEE Transactions on Image Processing, vol. 15, no. 1, pp. 178–188, 2005.
- K. Zhang, W. Zuo, S. Gu, and L. Zhang, “Learning deep cnn denoiser prior for image restoration,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3929–3938.
- K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising,” IEEE Transactions on Image Processing, vol. 26, no. 7, pp. 3142–3155, 2017.
- C. Dong, C. C. Loy, K. He, and X. Tang, “Learning a deep convolutional network for image super-resolution,” in European conference on computer vision. Springer, 2014, pp. 184–199.
- L. Xu, J. S. Ren, C. Liu, and J. Jia, “Deep convolutional neural network for image deconvolution,” in Advances in Neural Information Processing Systems, 2014, pp. 1790–1798.
- W. Ren, S. Liu, H. Zhang, J. Pan, X. Cao, and M.-H. Yang, “Single image dehazing via multi-scale convolutional neural networks,” in European conference on computer vision. Springer, 2016, pp. 154– 169.
- A. Buades, B. Coll, and J.-M. Morel, “A non-local algorithm for image denoising,” in 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05), vol. 2. IEEE, 2005, pp. 60–65. JOURNAL OF LATEX CLASS FILES, VOL. 14, NO. 8, AUGUST 2015 14

- J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, “Nonlocal sparse models for image restoration,” in 2009 IEEE 12th international conference on computer vision. IEEE, pp. 2272–2279.
- M. Elad and M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” IEEE Transactions on Image processing, vol. 15, no. 12, pp. 3736–3745,2006.
- J. Mairal, M. Elad, and G. Sapiro, “Sparse representation for color image restoration,” IEEE Transactions on image processing, vol. 17, no. 1, pp. 53–69, 2007.
- J. Yang, J. Wright, T. S. Huang, and Y. Ma, “Image super-resolution via sparse representation,” IEEE transactions on image processing, vol. 19, no. 11, pp. 2861–2873, 2010.
- J. Xie, L. Xu, and E. Chen, “Image denoising and inpainting with deep neural networks,” in Advances in neural information processing systems, 2012, pp. 341–349.
- X. Mao, C. Shen, and Y.-B. Yang, “Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections,” in Advances in neural information processing systems, 2016, pp. 2802–2810.
- S. Lefkimmiatis, “Universal denoising networks: a novel cnn architecture for image denoising,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 3204–3213.
- D. Liu, B. Wen, Y. Fan, C. C. Loy, and T. S. Huang, “Non-local recurrent network for image restoration,” in Advances in Neural Information Processing Systems, 2018, pp. 1673–1682.
- J. Kim, J. Kwon Lee, and K. Mu Lee, “Accurate image superresolution using very deep convolutional networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 1646–1654
- S. Nah, T. Hyun Kim, and K. Mu Lee, “Deep multi-scale convolutional neural network for dynamic scene deblurring,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3883–3891.
- O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas, “Deblurgan: Blind motion deblurring using conditional adversarial networks,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8183–8192.
- G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, “Image inpainting for irregular holes using partial convolutions,” in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 85–100.

- K. Yu, C. Dong, L. Lin, and C. Change Loy, “Crafting a toolchain for image restoration by deep reinforcement learning,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 2443–2452.
- D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Deep image prior,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 9446–9454.
- Z. Shen, W.-S. Lai, T. Xu, J. Kautz, and M.-H. Yang, “Deep semantic face deblurring,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8260–8269.
- A. Bulat and G. Tzimiropoulos, “Super-fan: Integrated facial landmark localization and super-resolution of real-world low resolution faces in arbitrary poses with gans,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 109–117.
- K. Grm, W. J. Scheirer, and V. Struc, “Face hallucination using ˇ cascaded super-resolution and identity priors,” IEEE Transactions on Image Processing, vol. 29, no. 1, pp. 2150–2165, 2019.
- S. Menon, A. Damian, S. Hu, N. Ravi, and C. Rudin, “Pulse: Self-supervised photo upsampling via latent space exploration of generative models,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 2437–2445.
- J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for realtime style transfer and super-resolution,” in European conference on computer vision. Springer, 2016, pp. 694–711.
- X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7794–7803.
- T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, “Semantic image synthesis with spatially-adaptive normalization,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 2337–2346.
- T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2019, pp. 4401–4410. JOURNAL OF LATEX CLASS FILES, VOL. 14, NO. 8, AUGUST 2015 15
- E. Agustsson and R. Timofte, “Ntire 2017 challenge on single image super-resolution: Dataset and study,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, July 2017.

- T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, “High-resolution image synthesis and semantic manipulation with conditional gans,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8798–8807.
- K. Nazeri, E. Ng, T. Joseph, F. Qureshi, and M. Ebrahimi, “Edgeconnect: Generative image inpainting with adversarial edge learning,” 2019.