**SYNOPSIS**

Cartooning an Image Using Open CV andPython

# Submitted By Rahul Kumar

**(2000290140094)**

# Shubhranshu Yadav

**(2000290140119)**

# Rekha Rani

**(2000290140101)**

# Submitted in partial fulfillment of the Requirements for the Degree of

MASTER OF COMPUTER APPLICATIONS

# Under the Supervision of Dr. Shashank Bhardwaj (Assistant Professor), Mr. Ankit Verma (Assistant Professor)



**Submitted to**

**DEPARTMENT OF COMPUTER APPLICATIONS**

# KIET Group of Institutions, Ghaziabad Uttar Pradesh-201206

**(MAR 2022)**

## Dr. Ajay Shrivastava

**Head, Department of Computer Applications KIET Group of Institutions, Ghaziabad**

# CERTIFICATE

Certified that **Rahul Kumar (2000290140094), Rekha Rani(2000290140101), Shubhranshu Yadav (2000290140119)** have carried out the project work having “**Cartooning an Image**” for **Master of Computer Applications** from Dr. A.P.J. Abdul Kalam Technical University (AKTU**)** (formerly UPTU), Technical University, Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself / herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

## Date:

**Rahul Kumar (2000290140094)**

## Rekha Rani (2000290140101)

**Shubhranshu Yadav(2000290140119)**

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

## Date:

**Dr. Shashank Bhardwaj (Assistant Professor),**

## Department of Computer Applications KIET Group of Institutions, Ghaziabad

**Signature of Internal Examiner Signature of External Examiner**

# ABSTRACT

**Image Processing** – In the field of the research processing of an image consisting of identifying an object in an image, identify the dimensions, no of objects, changing the images to blur effect and such effects are highly appreciated in this modern era of media and communication. There are multiple properties in the Image Processing. Each of the property estimates the image to be produced more with essence and sharper image. Each Image is examined to various grid. Each picture element together is viewed as a 2-D Matrix. With each of the cell store different pixel values corresponding to each of the picture element.

# Acknowledgment

Success in life is never attained single handedly. My deepest gratitude goes to my thesis supervisor, **Dr. Shashank Bhardwaj** for his guidance, help and encouragement throughout my researchwork. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Ajay Kumar Shrivastava**, Professor and Head, Department of Computer Applications, for his insightful comments and administrativehelp at various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me moral support and other kind of help. Without their support, completion ofthis work would not have been possible in time. They keep my life filled with enjoyment and happiness.

## Rahul Kumar (2000290140094)

**Rekha Rani (2000290140101)**

**Shubhranshu Yadav (2000290140119)**

**TABLE OF CONTENTS**

**Chapter 1** - Introduction

* 1. Introduction to Project
  2. Objective of Project
  3. Problem Definition
  4. Methodology
  5. Project Scope
  6. Hardware/ Software Requirement

**Chapter 2-** Feasibility

* 1. Feasibility Study
  2. Technical Feasibility
  3. Behavioral Feasibility

**Chapter 3-** Literature Survey

* 1. Literature Survey
  2. Existing systems

**Chapter 4-** System Design

* 1. Introduction
  2. Modules Description
  3. Conceptual Models
     1. Data Flow Diagram
  4. DFD Levels and Layers

**Chapter 5-** Flow-chart

* 1. Flow-chart
  2. Use Case Diagram
  3. Block Diagram
     1. Entity Relationship Model
     2. Entities
     3. Relationships
     4. Attribute

**Chapter 6-** Algorithm

* 1. Algorithm
  2. Identify an Image
  3. Color to the RGB Image
  4. ER-Notations

**Chapter 7-** Implementation

* 1. Home Page
  2. Open the Image Button
  3. Output
  4. OpenCV
  5. Color Quantization
  6. Objective of Result

**Chapter 8-** Coding

8.1 Module wise code

**Chapter 6-** Conclusion

**Chapter 7-** References

# CHAPTER 1

**INTRODUCTION**

## INTRODUCTION TO PROJECT

Advanced technology has become the integral part of our life. To satisfy the need of the society, almost in each work, we use the technology. In current era computer science is major subject. It has many real life applications such as cloud computing, remote monitoring, Wirelesssensor network, uncertainty), internet of things, Neural network, artificial intelligence, internet Security, and so on. Technology is the mode by which user can store, fetch, communicate and utilize the information. The image processing plays a major role in all computers related applications. The image processing appears in many real-life applications, e.g, home security, banking system, education sector, defense system, Railway, and so on. In this manuscript we discuss about the cartooning of image.

Each of these methodologies offers a rapid contribution to human interest. Each confined methodology helps in filtering the picture element that forms to an image.

There are various factors that enables to produce the essence of an image. The concerns are contrasting and appropriate color mixing, matching between any two pixels connecting two cells, accurate placing of objects together combined to form image features. In the recent times there happened to be drastic changes in ample fields. The uplift of these fields enhances in betterment of the society. In the field of medicine, these processing of images enable to extract the fullest accuracyof the images.

Image Processing is widely processed in the medical field such as in the MRI/ET scans. The amount of research in the image processing has helped to acquire early detection of tumors. There plays a vital role in the field of image processing and in the field of Biology. This researchbound to save livelihood as early detection can be identified and effective treatment can be started off. These extended concepts have enabled to build better security systems which ensure safety. The security/surveillance systems have managed to build systems depending on the imageprocessing algorithms,

The recent technology of fingerprint unlocks, face detection unlock has resulted in developing an efficient security. These Bio metric systems perhaps have been now installed on tosmaller devices as well for the simpler usage. With the recent success apprehended by the socialmedia is duly with the techniques installed to enhance the user experience. e.g; – Facebook confines with the auto tag mechanism to automatically suggest the person’s name and not by manually tagging each person on the image.

The basic concept in this algorithm involves the technique of converting the RGB colorimage to an accurate, cartooned image without multiple filtration's or blurred image without proper facilitation of edge detection. This user interface allows to apply the animation effects. This naturally provides an artistic effect and comics as well with wide range of pictures.

## OBJECTIVE OF PROJECT

Creating a cartoon like effect is time and space consuming. Existing solutions to provide cartoon like effect to images are complex. Some solutions involve installing complex photo editing software like photo-shop and other involve performing some task by user. Our project shows a way to carry out the task of Applying effects is more suitable, space efficient and takes minimum user efforts, for example toony photos is an existing website to perform such task but it is difficult to use as user has to markdown points & lines on the image to apply effects which is not user friendly also the options are limited. Hence there is a dire need for a project which is user friendly and performs the task of applying effects to images very well.

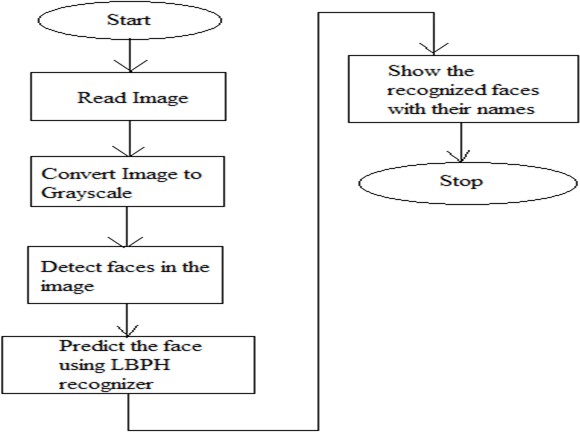
## PROBLEM DEFINITION

Training of networks for different style images is time consuming and requires lots of computation hardware(GPU). Different content images may produce slightly different styled images. Precision of cartoon like effect entirely depends on type of content image provided.

## METHODOLOGY

Methodology In this project we are accepting input image and then extracting edges, gray image, then applying median blur with Bilateral Filter technique to convert input image to cartoon image. To run this project you need to have 64 bit operating system and then install python3.7 software which i will send you with code. Install this python software and then in first or second installation screen in bottom you will find checkbox saying add path to variable. You just select that checkbox and continue with installation. After installation open command prompt and execute below commands. Your system must connect to internet before executing below commands.

* + - pip install numpy
    - pip install scipy
    - pip install Open CV-Python
    - pip install pillow



**Flow-chart of the methodology used for Cartooning Image**

* 1. **PROJECT SCOPE**

The project successfully delivered on all requirement specification specified by the user. Care was ensured during the design to make sure data integrity is maintained and to avoid all forms of redundancies associated with data. The user is assured a very friendly interface, behind which there are wide ranging technical details thatwent in. The user guide is a mere formality because, the project was specially created bearing in mind interaction and designs that would make users feel as though they have used a system such as this.

The technical document that is provided in the synopsis of this project will help developers understandthe internal workings of the system.

## HARDWARE/ SOFTWARE REQUIREMENT

**Hardware–**

## Software–

1. On Smart Phone
   * Processor should be dual core.
   * Processor speed be 1.3 GHz.
   * ROM 1 GB.
   * RAM 512 MB.
2. On Personal Computer (PC)
   * Processor should be dual core.
   * Processor speed be 1.5 GHz.
   * RAM more than 4 GB.
   * ROM more than 30 GB.

Visual Studio Code. Python

# CHAPTER 2 FEASIBILITY

## FEASIBILITY STUDY

Feasibility study is used to check the visibility of the project under the consideration of the theoretical project and its various types of feasibility are conducted below are important which are represented are as follows :-

## Technical Feasibility

Technical Feasibility determines whether the work for the project can be done with the existing equipment’s software technology and available personals technical feasibility is concerned with the specified equipment and software that will satisfy the user requirement this project is feasible on technical remark also as the proposed system is beneficiary in term of having a soundproof system with new technical equipment’s install on the system that proposed system can run on any machine supporting window and work on the best software and hardware that had been used while designing the system so it wouldbe visible in all technical term of feasibility3.

## Behavioral Feasibility

Behavioral Feasibility is the measure of how the society is looking towards ourproject, what is the reaction of people who are going to use this in upcoming future.

It includes how strong the reaction of user will be towards the development of new system that involves device use in their daily life for connecting with faculties at college

* 1. **Literature Survey**

# CHAPTER 3 LITERATURE SURVEY

Library Cartoons: A Liter oons: A Literature Review of Libr perspective on Library-themed Car y-themed Cartoons, Caricatur oons, Caricatures, and Comics Julia B. Chambers is a MLIS upand-comer at San Jose State University's School of Library and Information Science. To comprehend contrasting perspectives on past occasions, antiquarians, political theory researchers, and sociologists have examined political and publication kid's shows with topics going from decisions to financial approach to human rights. However sparse examination has been devoted to kid's shows with library topics. The creator of this paper inspects peer-explored writing regarding the matter of library kid's shows, including verifiable foundation, examination of ongoing subjects, and contentions for advancing library-themed kid's shows, exaggerations, and funnies. The creator finds a huge hole in the writing on this theme and presumes that data experts would profit by an extensive substance investigation of library-themed kid's shows to improve comprehension of the essentialness of libraries during noteworthy occasions, survey public view of libraries, and distinguish patterns after some time.

Researchers have examined and dissected the impact and estimation of publication kid's shows in the United States since the beginning of the twentieth century, not long after kid's shows turned into a standard element in East Coast papers. In a 1933 article, American craftsmanship and scholarly pundit Elizabeth Luther Cary contended that American exaggeration gave understanding into history, uncovering perspectives or elective mentalities that papers and history books have in any case neglected to record. Twenty years afterward, Stephen Becker (1959), creator of Comic Art in America, agreed that early instances of exaggeration served to make up for editorial shortfalls, in some cases going about as the solitary satisfactory source for editorial excessively indecent or touchy to show up in composed publications. Richard Felton Outcault's Yellow Kid publication kid's shows, distributed in 1896 in the New York World, are one model: "[Yellow Kid] brought something new and disturbing into American homes: the ghettos, and ghetto children, and customary savagery, and slang, and the arrogance of destitution" (Becker, 1959, p. 13).

## Existing System

Our project shows a website to carry out the task of Applying effects is more suitable, space efficient and takes minimum user efforts, for example toony photos is an existing website to perform such task but it is difficult to use as user has to markdown points & lines on the image to apply effects which is not user friendly also the options are limited. Hence there is a dire need for a website which is user friendly and performs the task of applying effects to images very well.

# CHAPTER 4 SYSTEM DESIGN

## Introduction

System design is the solution of a “how to approach to the creation of the new system. It is composed of several steps. It facilitates the understanding and provides the procedural details necessary for implementation of the system recommended in the feasibility study. Emphasis is given on translating the performance requirements into design specification. Design goes through logical and physical stages of development.

Logical design reviews the present physical system; prepares input and output specification; make editing; security and control specification; details the implementation plan, and prepare logical design walk through. The physical design maps out the details of the physical system; plans the system implementation plan and specifies hardware and software. System design translates the system requirement into the ways of the system as recommended in the feasibility study. Thus the system design is the translation from user- oriented document to a programmer or a database personal oriented document. System design is a highly creative process that can be greatly facilitated by the following:-

* + - Strong Problem Definition
    - Pictorial description of the Existing System
    - Set of Requirements of the new system

## Module Description

Basically, we are going to use a series of filters and image conversions.

First we downscale the image and then apply bilateral filter to get a cartoon flavor. Then again we upscale the image.

Next step is getting a blurred version of the original image. Now, we don’t want the colors to interfere in this process. We only want the blurring of the boundaries. For this, we first convert the image to gray – scale and then we apply the media blur filter.

Next step is to identify the edges in the image and then add this to the previously modified images to get a sketch pen effect. For this first we are using adaptive threshold. We can experiment with other types of threshold techniques also. Because Computer Vision is all about experimenting.

In step 5, we compile the final images obtained from the previous steps.

## Conceptual Models Data Flow Diagram

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into detail, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design). A DFD shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored. It does not show information about process timing or whether processes will operate in sequence or in parallel, unlike a traditional structured flowchart which focuses on control flow, or a UML activity workflow diagram, which presents both control and data flows asa unified model. Data flow diagrams are also known as bubble charts. DFD is a designingtool used in the top-down approach to Systems Design.

## Symbols and Notations Used in DFDs-

Using any convention’s DFD rules or guidelines, the symbols depict the four componentsof data flow diagrams –

**External entity:** an outside system that sends or receives data, communicating with the system being diagrammed. They are the sources and destinations of information entering or leaving the system. They might be an outside organization or person, a computer system,or a business system. They are also known as terminators, sources and sinks or actors. Theyare

typically drawn on the edges of the diagram.

**Process:** any process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the data flow based on business rules.

**Data store:** files or repositories that hold information for later use, such as a databasetable or a membership form.

**Data flow:** the route that data takes between the external entities, processes, and data stores. It portrays the interface between the other components and is shown with arrows, typically labelled with a short data name, like “Billing details.”

# DFD levels and layers

A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0, 1 or 2, and occasionally goto even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish.

**DFD Level 0** is also called a Context Diagram. It’s a basic overview of the whole systemor process being analyses or modelled. It’s designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers.

**DFD Level 1** provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system, as you break down the high-level process of the Context Diagram into its sub processes.

**DFD Level 2** then goes one step deeper into parts of Level 1. It may require more text toreach the necessary level of detail about the system’s functioning.

Progression to Levels 3, 4 and beyond is possible, but going beyond Level 3 is uncommon. Doing so can create complexity that makes it difficult to communicate, compare or model effectively.

Using DFD layers, the cascading levels can be nested directly in the diagram, providing a cleaner look with easy access to the deeper dive.

## When it's used?

The DFD is an excellent communication tool for analysts to model processes and functional requirements. One of the primary tools of the structured analysis efforts of the 1970's it was developed and enhanced by the likes of Your don, Mc Menamin, Palmer, Gane and Sarson. It is still considered one of the best modeling techniques for eliciting and representing the processing requirements of a system.

Used effectively, it is a useful and easy to understand modeling tool. It has broad application and usability across most software development projects. It is easily integrated with data modeling, workflow modeling tools, and textual specs. Together with these, it provides analysts and developers with solid models and specs. Alone, however, it has limited usability. It is simple and easy to understand by users and can be easily extended and refined with further specification into a physical version for the design and development teams.

The different versions are Context Diagrams (Level 0), Partitioned Diagrams (single process only -- one level), functionally decomposed, leveled sets of Data Flow Diagrams.

## Data Store

It is a repository of information. In the physical model, this represents a file, table, etc. In the logical model, a data store is an object or entity.

## DataFlow

DFDs show the flow of data from external entities into the system, showed how the data moved from one process to another, as well as its logical storage.

There are only four symbols:

* Squares representing **external entities**, which are sources or destinations of data.
* Rounded rectangles representing **processes**, which take data as input, do something to it, and output it.
* Arrows representing the **data flows**, which can either, be electronic data or physical items.
* Open-ended rectangles representing **data stores**, including electronic stores such as databases or XML files and physical stores such as or filing cabinets or stacks of paper.

There are several common modeling rules for creating DFDs:

* All processes must have at least one data flow in and one data flow out.
* All processes should modify the incoming data, producing new forms of outgoing data.
* Each data store must be involved with at least one data flow.
* Each external entity must be involved with at least one data flow.
* A data flow must be attached to at least one process.

DFDs are nothing more than a network of related system functions and indicate from where information is received and to where it is sent. It is the starting point in the system that decomposes the requirement specifications down to the lowest level detail.

The four symbols in DFD, each of which has its meaning. They are given below:

* External entities are outside to system but they either supply input data in the system or use the system output. These are represented by square of rectangle. External entities that
* supply data into a system are sometimes called Sources. External entities that use system data are sometimes called sinks.
* Data flow models that passages of data in the system and are represented by line by joining system components. An arrow indicates the direction of the flow and the line is labeled by the name of the data flow.
* Process show that the systems do. Each process has one or more data inputs and one

or data outputs. Circles in DFD represent them. Each high level process may be consisting of more than one lower level processes. Process will be expanded in sequence level DFD. A circle or a bubble represents a process that transforms incoming data flow into outgoing data-flow.

The high level processes in a system are:

* + Receivable process.
  + Verifiable process.
  + Disposal process.

File or data store is a repository of data. They contain data that is retained in the system. Process can enter data into data store or retrieved data from the data store. An open rectangle is a data store, data at rest.

# CHAPTER- 5

# FLOWCHAT

## FLOWCHART

Flowchart is a graphical representation of an algorithm. Programmers often use it as a program-planning tool to solve a problem. It makes use of symbols which are connected among them to indicate the flow of information and processing. The process of drawing a flowchart for an algorithm is known as “flow charting”.

Basic Symbols used in Flowchart Designs-

**Terminal:** The oval symbol indicates Start, Stop and Halt in a program’s logic flow. A pause/halt is generally used in a program logic under some error conditions. Terminal is the first and last symbols in the flowchart.

**Input/Output**: A parallelogram denotes any function of input/output type. Program instructions that take input from input devices and display output on output devices are indicated with parallelogram in a flowchart.

**Processing:** A box represents arithmetic instructions. All arithmetic processes such as adding, subtracting, multiplication and division are indicated by action or process symbol.

**Decision:** Diamond symbol represents a decision point. Decision based operations such as yes/no question or true/false are indicated by diamond in flowchart.

**Connectors:** Whenever flowchart becomes complex or it spreads over more than one page, it is useful to use connectors to avoid any confusions. It is represented by a circle.

**Flow lines:** Flow lines indicate the exact sequence in which instructions are executed. Arrows represent the direction of flow of control and relationship among different symbols of flowchart.

## Use Case Diagram

In the Unified Modelling Language (UML), a use case diagram can summarize the detailsof your system's users (also known as actors) and their interactions with the system. To build one, you'll

use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:

* Scenarios in which your system or application interacts with people, organizations,or external systems
* Goals that your system or application helps those entities (known as actors) achieve

## Use case Diagram Components

To answer the question, "What is a use case diagram?" you need to first understand its building blocks. Common components include:

**Actors:** The users that interact with a system. An actor can be a person, an organization, or an outside system that interacts with your application or system. They must be external objects that produce or consume data.

**System:** A specific sequence of actions and interactions between actors and thesystem. A system may also be referred to as a scenario.

**Goals:** The result of most use cases. A successful diagram should describe the activities and variants used to reach the goal.

## Use case diagram symbols and notation

The notation for a use case diagram is straightforward and doesn't involve as many typesof symbols as other UML diagrams.

## Use cases

Horizontally shaped ovals that represent the different uses that a user might have.

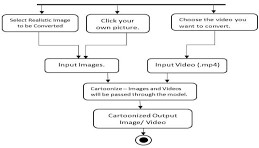
**Actors:** Stick figures that represent the people employing the use cases.

**Associations:** A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.

**System boundary boxes:** A box that sets a system scope to use cases. All use cases outside the box would be considered outside the scope of that system. Forexample, Psycho Killer is outside the scope of occupations in the chainsaw example found below.

**Packages:** A UML shape that allows you to put different elements into groups. Just as with component diagrams, these groupings are represented as file folders.

# BLOCK DIAGRAM



## Entity-Relationship Model

Simply stated the ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity-Relationship diagram which is used to visually represent data objects. Since Chen wrote his paper the model has been extended and today it is commonly used for database.

## Basic Constructs of E-R Model

The ER model views the real world as a construct of entities and association between entities.

## Entities

Entities are the principal data object about which information is to be collected. Entities are classified as independent or dependent (in some methodologies, the terms used are strong and weak, respectively). An independent entity is one that does not rely on another for identification. A dependent entity is one that relies on another for identification.

## Relationships

A Relationship represents an association between two or more entities. Relationships are classified in terms of degree, connectivity, cardinality, and existence.

## Attributes

Attributes describe the entity of which they are associated. A particular instance of an attribute is a value. The domain of an attribute is the collection of all possible values an attribute can have. The domain of Name is a character string.

## Classifying Relationships

Relationships are classified by their degree, connectivity, cardinality, direction, type, and existence. Not all modeling methodologies use all these classifications.

## Degree of a Relationship

The degree of a relationship is the number of entities associated with the relationship. The n-ray relationship is the general form for degree n. Special cases are the binary, and ternary, where the degree is 2 and 3 respectively.

## Connectivity and Cardinality

The connectivity of a relationship describes the mapping of associated entity instances in the relationship. The values of connectivity are "one" or "many". The cardinality of a relationship is the actual number of related occurrences for each of the two entities. The basic types of connectivity for relations are: one-to-one, one-to-many, and many-to-many.

## Direction

The direction of a relationship indicates the originating entity of a binary relationship. The entity from which a relationship originates is the parent entity; the entity where the relationship terminates is the child entity.

The direction of a relationship is determined by its connectivity type .An identifying relationship is one in which one of the child entities is also a dependent entity. A non- identifying relationship is one in which both entities are independent.

## Existence

Existence denotes whether the existence of an entity instance is dependent upon the existence of another, related, entity instance. The existence of an entity in a relationship is defined as either mandatory or optional.

## Generalization Hierarchies

A generalization hierarchy is a form of abstraction that specifies that two or more entities that share common attributes can be generalized into a higher level entity type called a super type or generic entity. The lower-level of entities become the sub type, or categories, to the super type. Sub types are dependent entities.

# CHAPTER 6

* 1. **ALGORITHM**

# ALGORITHM

The process to create a cartoon effect image can be initially branched into 2 divisions –

* + 1. To detect, blur and bold the edges of the actual RGB color image.
    2. To smooth, quantize and the conversion of the RGB image to gray scale. The results involved incombining the image and help achieve the desired result

## IDENTIFYING AN IMAGE

Finding smooth outline that represents or bounds the shape of the image is an important propertyto achieve a quality image. All Edge processing tasks are:

* **MEDIAN FILTER** – This filter helps in reducing the noise created during the down scaling theimage and later converting the original image to cartoon image by applying the bilateral filter. Any extreme specks are smoothed over.
* **EDGE DETECTION** – At first the noise of the image is removed within the image. Later the smoothed image is filtered using horizontal and vertical direction by dividing the cells of thepicture elements (both x and y dimensions.)
* **MORPHOLOGICAL OPERATIONS** – This serves the purpose to Bolden and smoother the outline of the edges variably. The pixels that are highlighted but seems far are removed. Hencethe edge lines reduce to thinner outline.
* **EDGE FILTERING** – Two divisions of the constituent regions, any region that pertain below acertain threshold is removed. Small outline identified by the detection method is removed fromthe final image.

## COLOR TO THE RGB IMAGE

The most important aspect is to eliminate the color regions and apply cartoon effects. Through this algorithm, the colors are smothered on multiple filtration's so as to create an equalcolor regions.

* **BILATERAL FILTERING** – The important role of this filter is to smooth the images without creating any sort of noise also while preserving the edges. Filtering is performed by reading animage from the file and storing it in a matrix object. Initially creating an empty matrix to store the result and applying bilateral filter. This totally depends on the kernel size and testing by running more no of iterations.
* **QUANTIZE COLORS** – The last step of the conversion involves the step of

reducing the number of colors in each pixel.

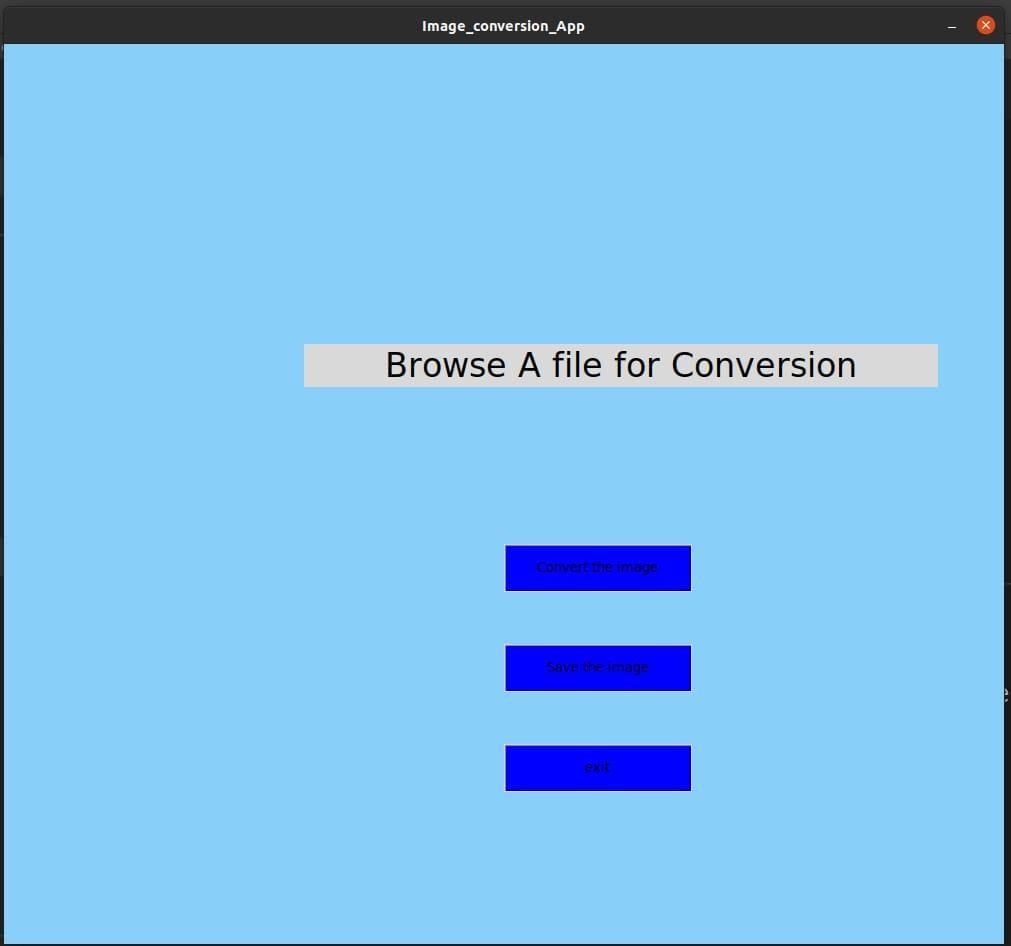
## ER Notation

The symbols used for the basic ER constructs are:

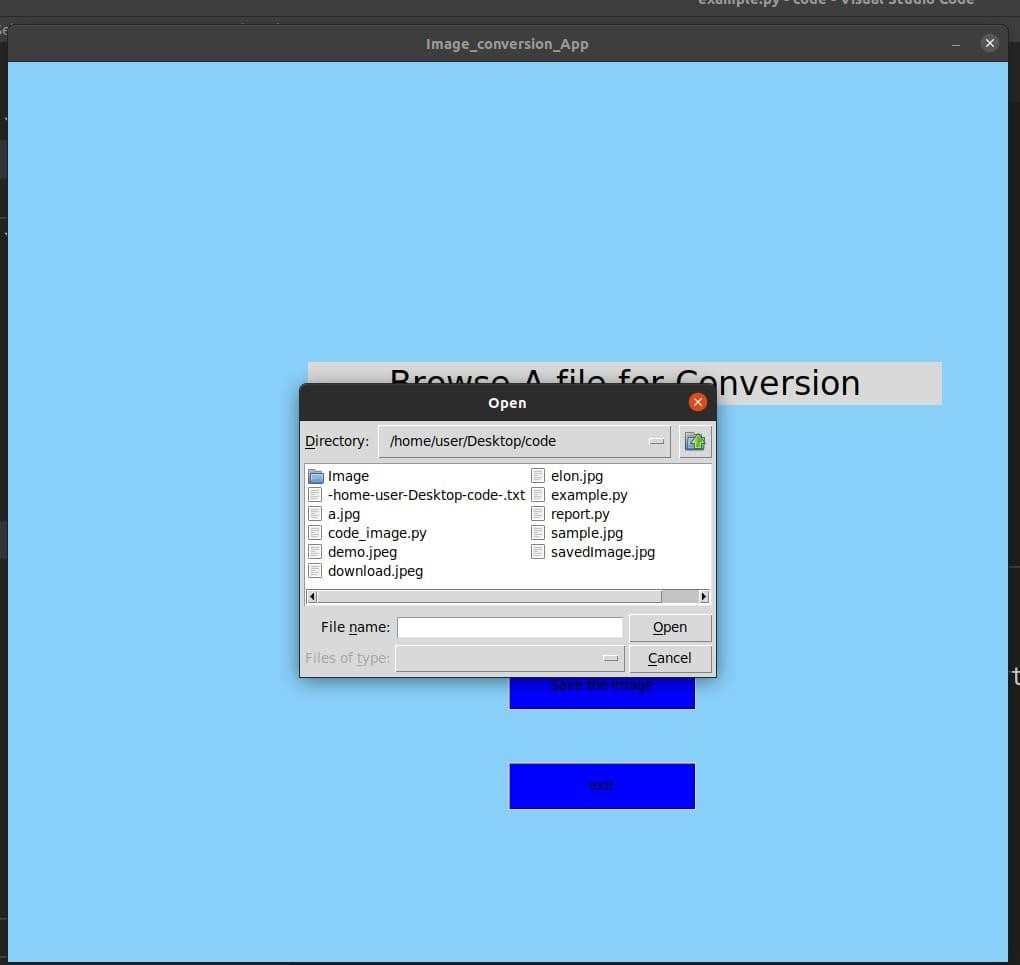
1. Entities are represented by labeled rectangles. The label is the name of the entity.
2. Relationships are represented by a solid line connecting two entities. The name of the relationship is written above the line. Relationship names should be verbs.
3. Attributes, when included, are listed inside the entity rectangle. Attributes which are identifiers are underlined. Attribute names should be singular nouns.
4. Cardinality of many is represented by a line ending in a crow's foot. If the crow's foot is omitted, the cardinality is one.
5. Existence is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional.
6. Existence is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional.

# CHAPTER 7 IMPLEMENTATION

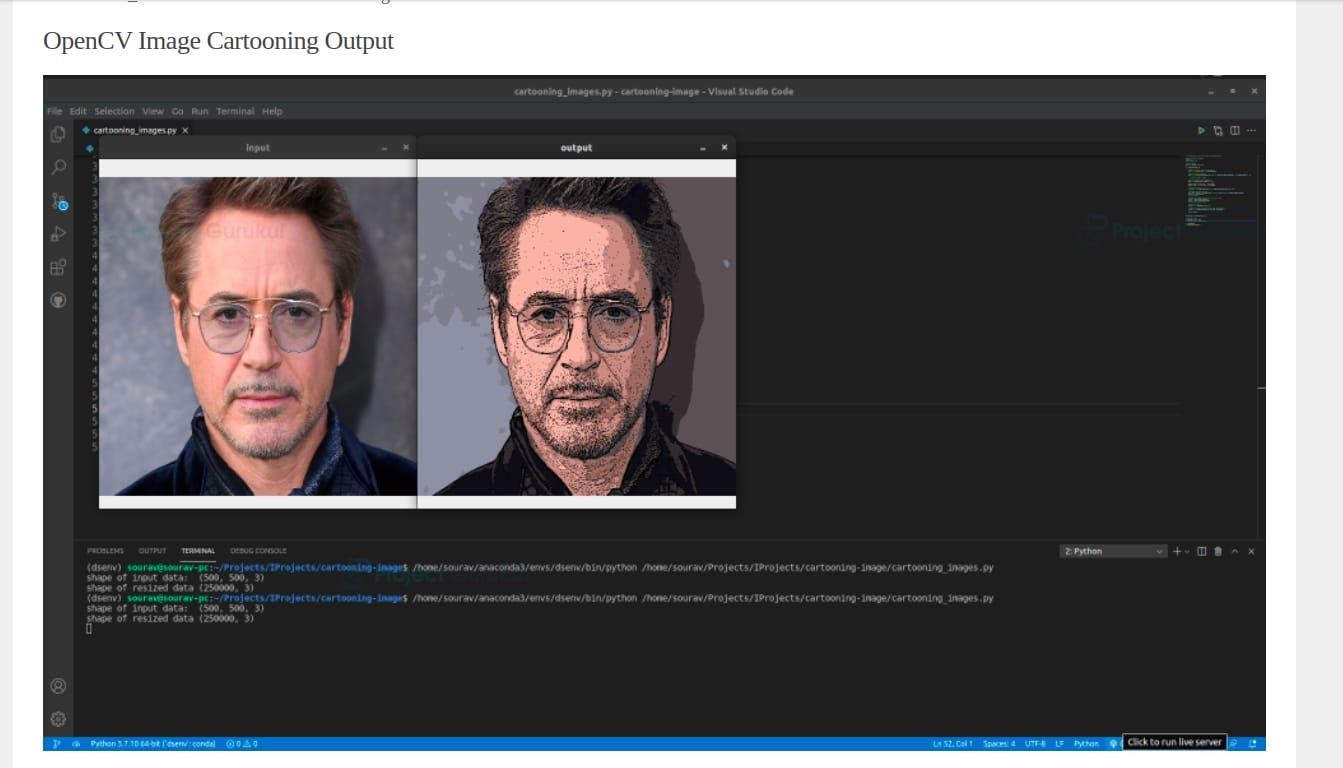
## HOME PAGE



* 1. **OPEN THE IMAGE BUTTON**



## OUTPUT



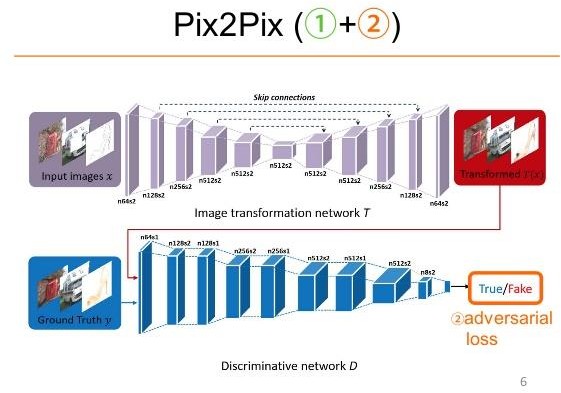
**ORIGINAL IMAGE**



## CARTOONING IMAGE



**ARCHITECTURE**



## Open CV

Python is the pool of libraries. It has numerous libraries for real-world applications. One such library is Open CV. Open CV is a cross-platform library used for Computer Vision. It includes applications like video and image capturing and processing.

OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 14 million. The library is used extensively in companies, research groups and by governmental bodies. It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time

It is majorly used in image transformation, object detection, face recognition, and many other stunning applications.

# OpenCV's application areas include:

* + - 2D and 3D feature toolkits
    - Egomotion estimation
    - Facial recognition system
    - Gesture recognition
    - Human–computer interaction (HCI)
    - Mobile robotics
    - Motion understanding
    - Object identification
    - Segmentation and recognition
    - Stereopsis stereo vision: depth perception from 2 cameras
    - Structure from motion (SFM)
    - Motion tracking
    - Augmented reality

To support some of the above areas, OpenCV includes a statistical machine learning library that contains:

Boosting

* Decision tree learning
* Gradient boosting trees
* Expectation-maximization algorithm
* k-nearest neighbour algorithm
* Naive Bayes classifier
* Artificial neural networks
* Random forest
* Random forest
* Support vector machine (SVM)
* Deep neural networks (DNN)

## NumPy

NumPy is an acronym for "Numeric Python" or "Numerical Python". It is an open source extension module for Python, which provides fast precompiled functions for mathematical and numerical routines. Furthermore, NumPy enriches the programming language Python with powerful data structures for efficient computation of multi-dimensional arrays and matrices. The implementation is even aiming at huge matrices and arrays. Besides that the module supplies a large library of high-level mathematical functions to operate on these.

matrices and arrays. It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

* A powerful N-dimensional array object.
* Sophisticated (broadcasting) functions.
* Tools for integrating C/C++ and FORTRAN code.
* Useful linear algebra, Fourier Transform, and random number capabilities.

# TensorFlow

TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow.

# Guiding Principles

**User Friendliness:** Keras is an API designed for human beings, not machines. It puts user experience front and centre. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear and actionable feedback upon usererror.

**Modularity:** A model is understood as a sequence or a graph of standalone, fully configurable modules that can be plugged together with as little restrictions as possible. In particular, neural layers, cost functions, optimizer, initialization schemes, activationfunctions, regularization schemes are all standalone modules that you can combine to create new models.

**Easy Extensibility:** New modules are simple to add (as new classes and functions), and existing modules provide ample examples. To be able to easily create new modules allows for total expressiveness, making Keras suitable for advanced research.

**Work with Python:** No separate models configuration files in a declarative format. Models are described in Python code, which is compact, easier to debug, and allows forease of extensibility.

## Color Quantization

Color quantization or vector quantization is an image processing technique that reduces the number of colors to show the image while preserving the overall appearance quality. It basically uses the K-Means clustering method to perform the task. K-means clustering is a method of finding a group of data. Each group is called a cluster.

What is the Adaptive threshold technique?

Thresholding is an image segmentation technique by setting all pixels whose values are above the threshold as foreground and the rest of the pixels as background. But in adaptive thresholding, these values are calculated for smaller regions.

Cartoonifier Project Prerequisites:

1. Python – 3.x (We used 3.7.10 for this project)
2. OpenCV – 4.5

Please run below command to install opencv to install the package. pip install opencv-python

3. Numpy – 1.19.5

## What we are going to build

At the end of this article, we aim to transform images into its cartoon. Yes, we will **CARTOONIFY** the images. Thus, we will build a python application that will transform an image into its cartoon using Open CV.

We aim to build an application which looks like:

**Step 1**: Importing the required modules We will import the following modules:

* + **CV2**: Imported to use Open CV for image processing.
  + **easygui**: Imported to open a file box. It allows us to select any file from our system.
  + **Numpy**: Images are stored and processed as numbers. These are taken as arrays. We use Numpy to deal with arrays.
  + **Imageio**: Used to read the file which is chosen by file box using a path.
  + **Matplotlib**: This library is used for visualization and plotting. Thus, it is imported to form the plot of images.
  + **OS**: For OS interaction. Here, to read the path and save images to that path.

**Step 2:** Building a File Box to choose a particular file

In this step, we will build the main window of our application, where the buttons, labels, and images will reside. We also give it a title by title() function.

def upload(): ImagePath=easygui.fileopenbox() cartoonify(ImagePath)

**Step 3:** Transforming an image to grayscale: contoured\_image = image

gray = cv2.cvtColor(contoured\_image, cv2.COLOR\_BGR2GRAY)

edged = cv2.Canny(gray, 70, 100)

**Step 4:** Retrieving the edges of an image Cartoon effect has two specialties:

1.Highlighted Edges 2.Smooth colors

In this step, we will work on the first specialty. Here, we will try to retrieve the edges and highlight them. This is attained by the adaptive thresholding technique. The threshold value is the mean of the neighborhood pixel values area minus the constant C. C is a constant that is subtracted from the mean or weighted sum of the neighborhood pixels. Thresh\_binary is the type of threshold applied, and the remaining parameters determine the block size.

**Step 5:** Preparing a Mask Image:

def ColorQuantization(image, K=4):

Z = image.reshape((-1, 3)) Z = np.float32(Z)

criteria = (cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER, 10000, 0.0001)

compactness, label, center = cv2.kmeans

In the above code, we finally work on the second specialty. We prepare a lightened color image that we mask with edges at the end to produce a cartoon image. We use bilateral Filter which removes the noise. It can be taken as smoothening of an image to an extent.

The third parameter is the diameter of the pixel neighborhood, i.e, the number of pixels around a certain pixel which will determine its value. The fourth and Fifth parameter defines QuantizationColor. These parameters are used to give a sigma effect, i.e make an image look vicious and like water paint, removing the roughness in colors.

Yes, it’s similar to BEAUTIFY or AI effect in cameras of modern mobile phones.

**Step 6:** Giving a Cartoon Effect:

image = cv2.imread(filename) resized\_image = resizeImage(image)

coloured = ColorQuantization(resized\_image) contoured = findCountours(coloured)

So, let’s combine the two specialties. This will be done using MASKING. We perform bitwise and on two images to mask them. Remember, images are just numbers?

Yes, so that’s how we mask edged image on our “BEAUTIFY” image.

This finally CARTOONIFY our image!

**Step 7:** Functionally of save button:

final\_image = contoured save\_q = input("Save the image? [y]/[n] ") if save\_q == "y":

cv2.imwrite("cartoonized\_"+ filename, final\_image) print("Image saved!")

Here, the idea is to save the resultant image. For this, we take the old path, and just change the tail (name of the old file) to a new name and store the cartoonified image with a new name in the same folder by appending the new name to the head part of the file.

For this, we extract the head part of the file path by os.path.dirname() method. Similarly, final\_image = contoured save\_q = input("Save the image? [y]/[n] ") is used to extract the extension of the file from the path.

Here, newName stores “Cartoonified\_Image” as the name of a new file. final\_image = contoured save\_q = input("Save the image? [y]/[n] ") joins the head of path to the newname and extension. This forms the complete path for the new file.

imwrite() method of cv2 is used to save the file at the path mentioned. cv2.imwrite("cartoonized\_"+ filename, final\_image)is used to assure that no color get extracted or highlighted while we save our image. Thus, at last, the user is given confirmation that the image is saved with the name and path of the file.

## OBJECTIVE OF RESULT

* + 1. Following area unit the most objectives planned and accomplished during this analysis work.Rapid image processing with high detection rates
    2. To provide High accuracy model as compare with current existing models.
    3. To provide very low false positive rate.

# CHAPTER 8

**CODING**

import numpy as np import cv2

filename = 'elon.jpg'

def resizeImage(image):

scale\_ratio = 0.30

width = int(image.shape[1] \* scale\_ratio) height = int(image.shape[0] \* scale\_ratio) new\_dimensions = (width, height)

resized = cv2.resize(image, new\_dimensions, interpolation=cv2.INTER\_AREA) return resized

def findCountours(image):

contoured\_image = image

gray = cv2.cvtColor(contoured\_image, cv2.COLOR\_BGR2GRAY) edged = cv2.Canny(gray, 70, 100)

contours, hierarchy = cv2.findContours(edged, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_NONE)

cv2.drawContours(contoured\_image, contours, contourIdx=-1, color=1, thickness=1)

cv2.imshow('Image after countouring', contoured\_image) cv2.waitKey(0)

cv2.destroyAllWindows() return contoured\_image

def ColorQuantization(image, K=4):

Z = image.reshape((-1, 3)) Z = np.float32(Z)

criteria = (cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER, 10000, 0.0001)

compactness, label, center = cv2.kmeans

( Z, K, None, criteria, 1, cv2.KMEANS\_RANDOM\_CENTERS)

center = np.uint8(center) res = center[label.flatten()]

res2 = res.reshape((image.shape)) return res2

if name == " main ":

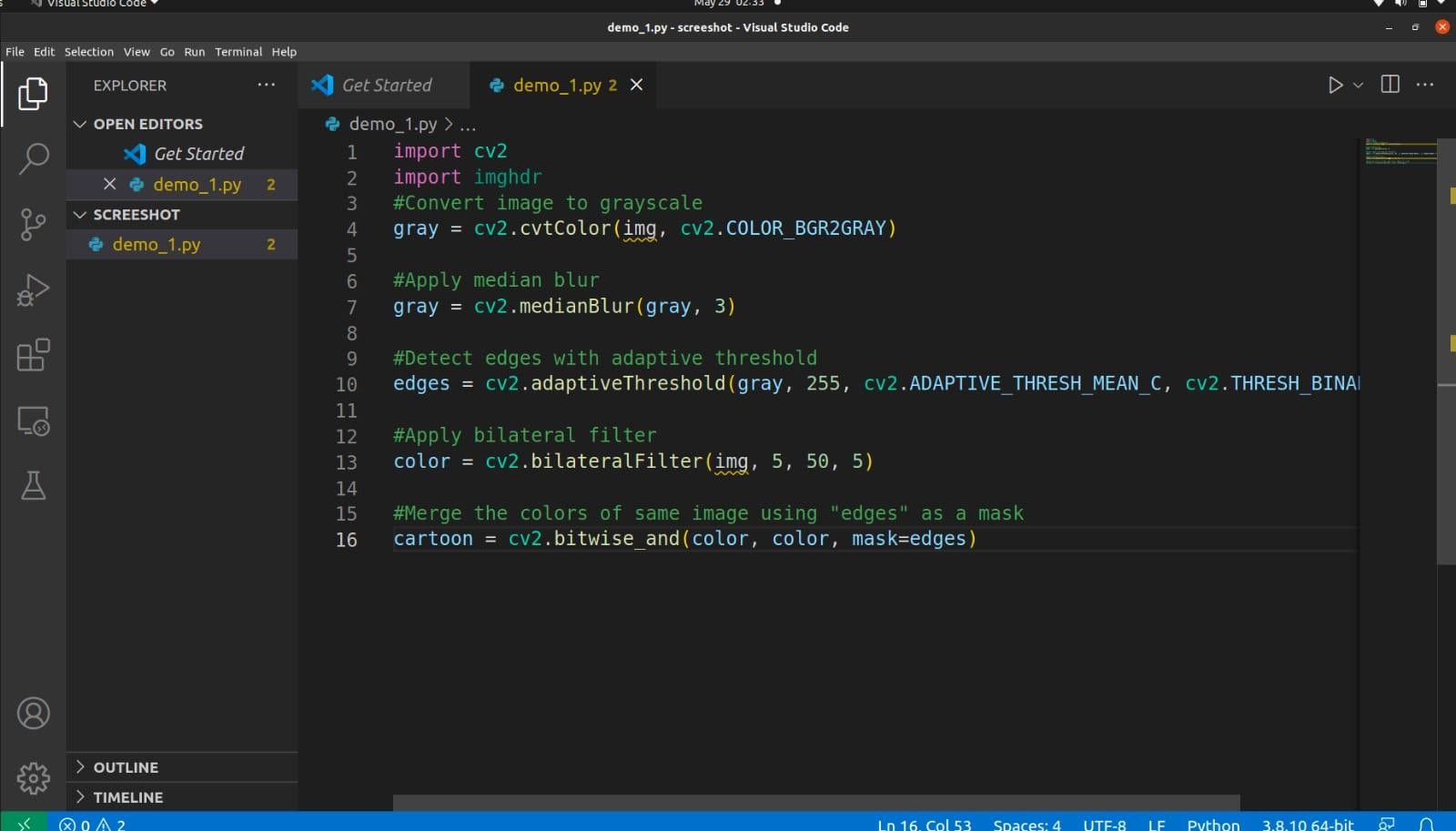
image = cv2.imread(filename) resized\_image = resizeImage(image)

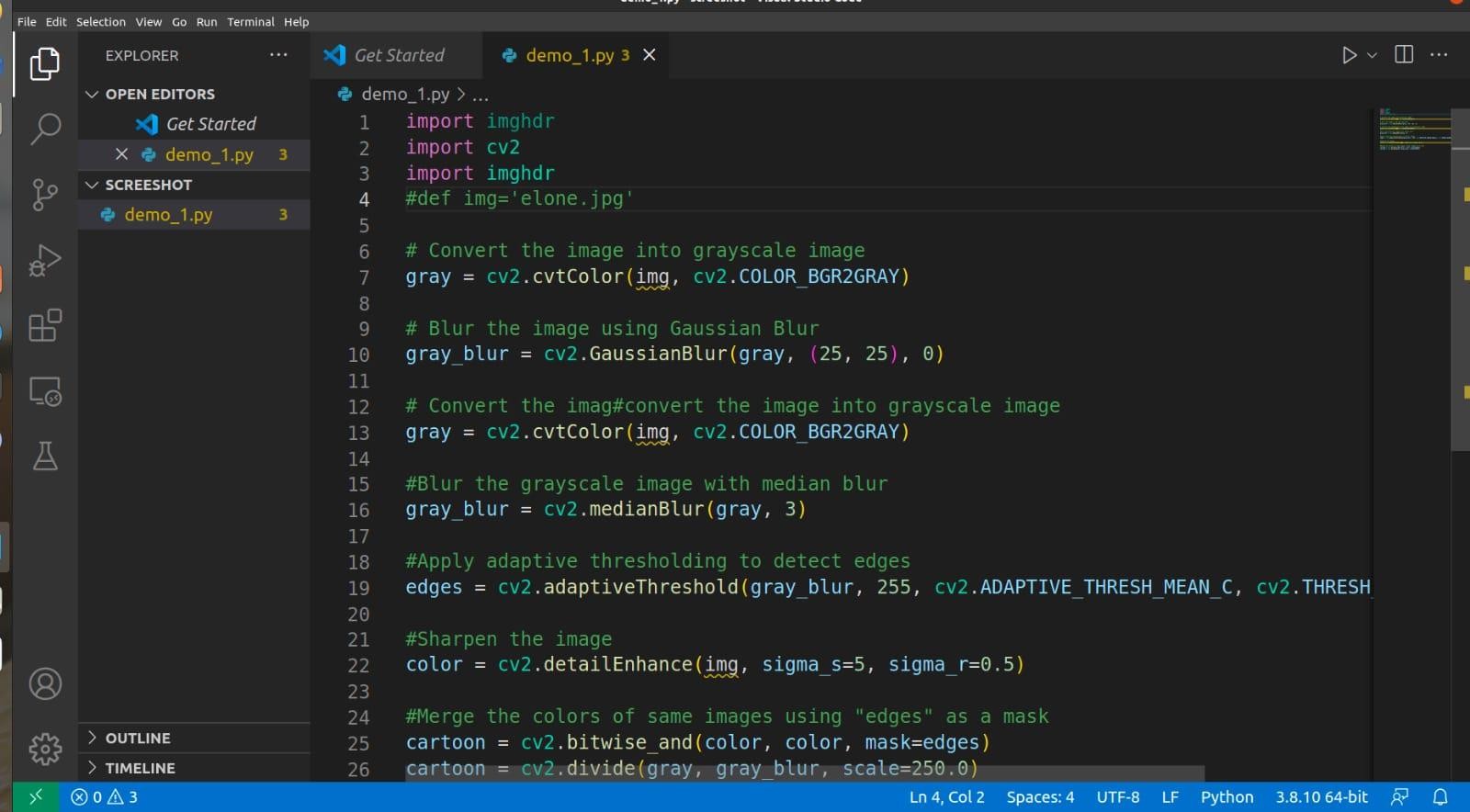
coloured = ColorQuantization(resized\_image) contoured = findCountours(coloured)

final\_image = contoured save\_q = input("Save the image? [y]/[n] ") if save\_q == "y":

cv2.imwrite("cartoonized\_"+ filename, final\_image) print("Image saved!")

**8.1 Module wise code**

**e**



# CHAPTER 9

# TESTING

## Testing

Software Testing is evaluation of the software against requirements gathered from users and system specifications. Testing is conducted at the phase level in software development life cycle or at module level in program code. Software testing comprises of

Validation and Verification.

## Unit Testing

While coding, the programmer performs some tests on that unit of program to know if it is error free. Testing is performed under white-box testing approach. Unit testing helps developers decide that individual units of the program are working as per requirement and are error free.

## Integration Testing

Even if the units of software are working fine individually, there is a need to find out if the units if integrated together would also work without errors. For example, argument passes and data updating etc.

## System Testing

The software is compiled as product and then it is tested. This can be accomplished using one or more of the following tests:

## Performance testing

This test proves how efficient the software is. It tests the effectiveness and average time taken bythe software to do desired task. Performance testing is done by means of load testing and stress testing where the software is put under high user and data load under various environment conditions.

## Acceptance Testing

When the software is ready to hand over to the customer it must go through last phase of testing where it is tested for user-interaction and response. This is important because even if the software matches all user requirements and if user does not like the way it appears or works, it may be rejected.

## Functionality testing

Tests all functionalities of the software against the requirement.

## Alpha testing

The team of developer themselves perform alpha testing by using the system as if it is being used in work environment. They try to find out how user would react to some action in software and how the system should respond to inputs.

## Beta testing

After the software is tested internally, it is handed over to the users to use it under their production environment only for testing purpose. This is not yet the delivered product. Developers expect that users at this stage will bring minute problems, which were skipped to attend.

# CHAPTER 10 CONCLUSION

First of all, the basic tools to handle the titled problems of the thesis are incorporated. It

includes origin and history of image processing, different types of uncertain environment, existing methods for cartoon imaging. Amid the previous three decades, the topic of image processing has gained vital name and recognition among researchers because of their frequent look in varied and widespread applications within the field of various branches of science and engineering. As an example, image processing is helpful to issues in signature recognition, digital video processing, Remote Sensing and finance. Conclusion and Future Directions Firstly, we use high-resolution camera to take picture of the internal structure of the wire. Secondly, we use Open-CV image processing functions to implement image pre- processing. Thirdly we use morphological opening and closing operations to segment image because of their blur image edges. The main attraction ofthe paper is to solve different types of images having one object, two object and three object which can’t be solved by any of the exiting methods but can be solved by our proposed method.

# CHAPTER 11

# 

# REFERENCES

1. Yi Yang, Yueting Zhuang, Dacheng Tao, Dong Xu, Jun Yu, and Jiebo Luo; "Recognizing Cartoon Image Gestures for Retrieval and Interactive Cartoon Clip Synthesis", IEEE Transactions On Circuits And Systems For Video Technology, Vol. 20, No. 12, Pp. 1745-1756, December 2010.
2. Zhang Liang,Yueting Zhuang , Yi Yang and Jun Xiao; "Retrieval-Based Cartoon Gesture Recognition and Applications Via Semi-Supervised Heterogeneous Classifiers Learning ", Pattern Recognition, 2013.
3. Sungyoung Kim, Soyoun Park, and Minhwan Kim; "Central Object Extraction for Object-Based Image Retrieval", E. M. Bakker et al. (Eds.): CIVR 2003, LNCS 2728, Pp. 39-49, 2003.
4. Yi Yang, Yueting Zhuang, Dong Xu, Yunhe Pan, Dacheng Tao and Steve Maybank; " Retrieval Based Interactive Cartoon Synthesis via Unsupervised Bi-Distance Metric Learning", MM’09, Beijing, China, Pp. 311-320, 2009.
5. Savitha Sivan and Thusnavis Bella Mary; "An Optimized Feature Extraction Technique for Content Based Image Retrieval", International Journal of Image Processing and Vision Sciences (IJIPVS), Pp. 35-40, 2013
6. Saikrishna, Yesubabu, Anandarao, Sudha Rani; "A Novel Image Retrieval Method Using Segmentation and Color Moments", Advanced Computing: An International Journal (ACIJ), Vol.3, No.1, Pp. 75-80, 2012.
7. Miki Haseyama and Atsushi Matsumura; "A Trainable Retrieval System For Cartoon Character Images", in Proc. ICME, Pp. 393–396, Jul. 2003.
8. Miki Haseyama and Atsushi Matsumura; "A Cartoon Character Retrieval System Including Trainable Scheme", in Proc. ICIP, Pp. 37–40, Sep. 2003.
9. Ahmed Talib, Massudi Mahmuddin, Husniza Husni and Loay E. George; "A Weighted Dominant Color Descriptor for Content-Based Image Retrieval", J. Vis. Commun. Image R., (Elsevier), Pp. 345-360, 2013. Mustafa M. Abd Zaid et al, International Journal of Computer Science and Mobile Computing, Vol.3 Issue.12, December- 2014, pg. 482-491 © 2014, IJCSMC All Rights Reserved 491.
10. Ruggero Milanese; "Detecting Salient Regions in an Image: from Biological Evidence

to Computer Implementation", Ph.D. Thesis, University of Geneva, 1993.

1. Jun Yu and Seah; "Fuzzy Diffusion Distance Learning for Cartoon Similarity Estimation", Journal of Computer Science and Technology, Vol. 26, No. 2, Pp. 203-216, 2011.

[ 12] Jun Yu, Jun Cheng and Dacheng Tao; "Interactive Cartoon Reusing by Transfer Learning", J. Signal Process, Vol. 92, Pp. 2147–2158, 2012.

1. OpenCV – Python tutorials.
2. OpenCV: Computer Vision Projects with Python by Joseph Howse, Prateek Joshi, Michael Beyeler.
3. How to create a cartoon effect – Opencv with Python by Sergio Canu.
4. Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks.[[arxiv](https://arxiv.org/abs/1611.07004)]
5. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative adversarial nets. NIPS 2014 [[arxiv](https://arxiv.org/abs/1406.2661)]
6. Ian Goodfellow. Tutorial: Generative Adversarial Networks. NIPS 2016 [[arxiv](https://arxiv.org/abs/1701.00160)]
7. PyTorch Docs. [<https://pytorch.org/docs/stable/index.html>]