

A Project Report

on

CARTOONIFY AN IMAGE USING OPENCV IN PYTHON

*Submitted in partial fulfillment of the
requirement for the award of the degree
of*

B. TECH COMPUTER SCIENCE ENGINEERING



(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

**Under The Supervision of
DR. T. POONGODI**

Submitted By:

Utkarsh Bhardwaj

18SCSE1010049

Saurabh Kumar

18SCSE1010541

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA
INDIA MONTH**

December 2021



**SCHOOL OF COMPUTING SCIENCE AND
ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA**

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the project, entitled “**CARTOONIFY AN IMAGE USING OPENCV .**” in partial fulfillment of the requirements for the award of the **Bachelor of Technology in Computer Science and Engineering** submitted in the **School of Computing Science and Engineering** of Galgotias University, Greater Noida, is an original work carried out during the period of **JULY-2021 to DECEMBER-2021**, under the supervision of **DR. T. Poongodi, Associate Professor, Department of Computer Science and Engineering**, of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

SAURABH KUMAR- 18SCSE1010541

UTKARSH BHARDWAJ- 18SCSE1010049

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

DR. T. POONGODI

ASSOCIATE PROFESSOR

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of **18SCSE1010541- SAURABH KUMAR, 18SCSE1010049- UTKARSH BHARDWAJ** has been held on _____ and his/her work is recommended for the award of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING.**

Signature of Examiner(s)

Signature of Supervisor(s)

Signature of Project Coordinator

Signature of Dean

Date:

Place:

ABSTRACT

Aim of the project is to put forward a solution for transforming snapshots or videos of real-world into animated photos (Cartoon Images) or Video. The earlier method of transformation requires complicated computer graphics and skills. The idea of the paper is based on designated snapshots and videos which are converted to an art form such as painting. Amongst all the techniques usable, the application of a Generative Adversarial Network (GAN) called Cartoon GAN will be used for the styling real-world images that use 2 loss functions namely, content loss and adversarial loss for getting a sharp and clear image. With the help of GAN, it is possible to convert video as well to its cartoonized version and the development of the project shows that our Proposed Idea provides high quality cartooned images and videos. We would apply an OpenCV library in the python while making the project. Now let us understand what OpenCV library in Python is. Python is the pool of libraries. It has multifold libraries for real- world operations. One corresponding library is OpenCV. OpenCV is a cross-platform library used for Computer Vision. It includes operations like tape and image capturing and processing. It's majorly used in image conversion, object finding, face recognition, and multiple other stunning operations. Since the user for cartoon picture retrieval system targets to urge relevant pictures to question image from information inside same object (i.e. a user has cartoon image with object —Dorall, during this case the user can target to urge all relevant image with —Dorall character), so a vital step in cartoon image retrieval is shaping the thing inside cartoon image. In this paper, associate degree economical technique for objects extraction from cartoon pictures is introduced; it's supported general assumptions associated with color and locations of objects in cartoon pictures, the objects square measure usually drawn close to the middle of the image, the background colors that the more oft drawn close to the sides of cartoon image, and also the object colors is a smaller amount bit for the sides. The processes of color division, seed filling and located the thing ghost are used. The results of conducted tests indicated that the system have promising potency for extracting each single or multi object(s) lay in easy and sophisticated backgrounds of cartoon pictures.

LIST OF FIGURE

S.no	TABLE	PAGE No.
1	Working model of GAN	8
2	Activity diagram	8

LIST OF CONTENTS

Title	Page No.
Abstract	I
List of Figures	II
Chapter 1 Introduction	
1.1 Introduction	5
1.2 Formulation of Problem	5
Chapter 2 Literature Survey	
2.1 Methodology	6
2.2 tools used and modules used	7
Chapter 3 implementation and code	
3.1 Step by Step code Explanation	14
3.2 Code	18
3.3 Project overview	22
Chapter 4 Result, Conclusion and future scope	26

INTRODUCTION:

Cartoons are ordinarily used in variety of kinds of uses. As we know cartoons are artistically made it requires elegant and fine natural cultural savvy. While portraying Cartoons in humongous reckoning for any animated Filmmaking it gets time- consuming for the artist as they need to define the sketch of the cartoon duly to get a good result. We all know that robustness plays an important place in the world of cinema, so to overcome the problem faced by the artist we've created a program with the help of GAN that not only converts images but it also converts tape into a jazziness.

A couple of ages ago, the styling of images consists of a particular Demesne named “non-photorealistic depiction”. The Traditional algorithm was being developed on the basis of the demesne for the styling of images and they were successful in naming any images by adding designs, texture, personality, etc. With the help of the algorithm, multiplex software was developed to convert real images (shot) into cartoon images some of the systems failed while some of the systems gave results but didn't satisfy all the Necessaries. Either, cartoon images are intricate compared to real life images.

To satisfy all the necessities of converting real images i.e., pic into cartoon image we've taken the help of Cartoon GAN which is one of the employments of Generative Adversarial Network (GAN). This program will ease natural work. In the following image, a real image is converted into an amped image with the help of the use developed known as “Cartoon GAN” in less duration of time. Not only of the conversion of images but we've also successfully converted tape clips into peppiness with the help of CV2 which is a library in python. In this manner time is being saved, fine work is generated within an ample quantum of time, which will give a great room for vibrance sedulity to make as momentous as movie or vibrance clips. This system gives a more accurate result of converting these images plus it also converts Videotape into a vigorousness clip compared to the prior techniques.

PROBLEM FORMULATION:

- Cartoons are commonly used in various kinds of applications. As we know cartoons are artistically made it requires elegant and fine human artistic skills. While portraying Cartoons in humongous numbers for any animated movies it gets time-consuming for the artist as they need to define the sketch of the cartoon properly to get a good result. We all know that animation plays an important role in the world of cinema
- Not only that but also the quality of the image or video is also needs to be of best quality otherwise the result would be not good.

SCOPE OF THE PROJECT:

User will be provided with a set of pretrained style images to choose from. Based on the chosen style and the content image provided by the user, the Resulting image with cartoon like effect is generated by the program. The implementation is based on of the combination of Gatys' A Neural Algorithm of Artistic Style, Johnson's Perceptual Losses for Real-Time Style Transfer and SuperResolution, and Ulyanov's Instance Normalization

NEED OF THE 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 photoshop and other involve performing some task by user. Our research 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 mark down 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.

Following is our brief research on existing solutions:

A. Cartoon Effect

The majority of photo editing websites offer the so called Cartoon Effect. The main advantages of online photo to cartoon effect apps are simplicity and quickness. You'll have to upload a photo from your computer or from the web, find Cartoon Effect in the tool set or choose between styles or variants of this funny photo effect (like in case of www.picturetopeople.org, Kuso Cartoon) and press the button Apply (or Go). The image processing varies from several seconds up to 1-2 minutes. However, as all quick online solutions these apps have drawbacks. A lot of photo online photo editing tools are rather humdrum because they are deprived of enhancement features. In these apps cartoonization is limited to 1-click operation. Besides, sometimes colors may become blurred and it leads to an unsatisfactory result. Such apps as www.converttocartoon.com, Photo.to, AnyMaking and others belong to this group. At the same time there are online photo editors with more advanced tools. They have a variety of adjustment options. For example, BeFunky helps you modify sketch brightness, contrast, smoothness and other details. Fig 1

B. Pencil Sketch

Another means of cartoonization is making a pencil sketch out of your digital photograph. Whenever you apply Cartoon effect your images turn bright and cheerful. If you want to render a solid atmosphere and achieve respectability in your online profile pencil sketch creation will suit your needs better. The image manipulation procedure is just the same as described for the cartoon effect. select the desired effect, push the button Apply and you are done. The application does its job instantly by itself. PhotoSketcher, Fotosketcher, Dumpr, Tuxpi photo editor and many other applications give you an opportunity to convert your snaps to life-like pencil sketches. Fig 2 Besides, you can decorate your profile photo with a cute photo frame and even create a photo with your favorite cartoon character. Amaze your nearest and dearest, friends and coworkers with a cool profile photo, stand out from the crowd and attract more followers and fans in social networks. You know, the first impression is the strongest)). How to turn photo into cartoon online or on Windows/Mac Moreover, sharing a photo cartoon on social media could attract more attention when others just post standard photos. We are going to share how to turn photo to cartoon on Windows, Mac, and online in this tutorial. With these photo editors and our guides, you can create cartoon at any time, even if you have not learnt any knowledge about painting. If you are ready, let's start right now.

ALREADY PRESENT PAPERS ON THIS PROJECT:

Year of The Paper	Title of the Paper	Methodology Used	Advantage	Limitation
2015	A Neural Algorithm of Artistic Style (Leon A. Gatys, Alexander S. Ecker, Matthias Bethge)	An artificial system based on a Deep Neural Network that creates artistic images of high perceptual quality. we take pre-trained neural network and define a 3 Component loss function that will enable us to achieve our end goal of style transfer and then optimize over that loss function.	First Ever Successful Attempt to create Artistic style transfer using Deep Neural Network. All previous attempts/algorithms to achieve style transfer had failed. The system uses neural representations to separate and recombine content and style of arbitrary images, providing a neural algorithm for the creation of artistic images.	While the Gatys et al. method produced beautiful neural style transfer results, the problem was that it was quite slow.
2016	Perceptual Losses for Real-Time Style Transfer and Super-Resolution (Justin Johnson, Alexandre Alahi, Fei-Fei Li)	proposes the use of perceptual loss functions for training feed-forward networks for image transformation tasks. a feed-forward network is trained to solve the optimization problem proposed by Gatys et al. in real-time.	Johnson et al. (2016) built on the work of Gatys et al., proposing a neural style transfer algorithm that is up to three orders of magnitude faster. The Johnson et al. method frames neural style transfer as a super-resolution-like problem based on perceptual loss functions.	While the Johnson et al. method is certainly fast, the biggest downside is that you cannot arbitrarily select your style images like you could in the Gatys et al. method.
2017	Instance Normalization: The Missing Ingredient for Fast Stylization (Dmitry Ulyanov, Andrea Vedaldi, Victor Lempitsky)	The change is limited to swapping batch normalization with instance normalization, and to apply the latter both at training and testing times. The resulting method can be used to train high-performance architectures for real-time image generation.	it was found that swapping batch normalization for instance normalization (and applying instance normalization at both training and testing) in Gatys et al. method, leads to even faster real-time performance and arguably more aesthetically pleasing results as well.	None

Table 1

What is an Image?

Visual representation of a real-life object (a person or any other object) in a two-dimensional form is called an image. An image is nothing but a collection of pixels in different color spaces. (Singh, 2019). From this definition, two terms are prominent in understanding what an image really is, these are the terms; 'two-dimensional' and 'pixels'. Two-dimensional form. When an object is in the two-dimensional form (2D) it means simply that there are only two dimensions of measurements that are used to define it. This could be the common width and height, or the geometric x-axis and y-axis. This property of an image is very important in carrying out mathematical operations to an image. In other words, we are simply saying that we can define or map an image on to an x-y plane. (Rafael C. Gonzalez) simplifies this by defining an image as a two-dimensional function, $F(x, y)$, where x and y are spatial coordinates and the amplitude of F at any pair of coordinates (x, y) is called the intensity of that image at that point. When x , y , and amplitude values of F are finite, we call it a digital image. In other words, an image can be defined by a two-dimensional array specifically arranged in rows and columns. Pixels. To understand more about what an image is, we can think about what exactly makes up an image. A complete image is a set that consists of small samples. These samples are called pixels. They are the smallest elements in any digital image. So, pixels are subsamples of an image that, when get combined, give us the complete image.

Image formats

In this section, I will introduce some file formats that we may encounter on our journey in image processing. It is important to know and understand the image format so that we can carry out various appropriate techniques when doing image processing. JPEG (or JPG) - Joint Photographic Experts Group JPEGs are known for their "lossy" compression, meaning that the quality of the image decreases as the file size decreases. You can use JPEGs for projects on the web, in Microsoft Office documents, or for projects that require printing at a high resolution. Paying attention to the resolution and file size with JPEGs is essential in

order to produce a nice-looking project. PNG - Portable Network Graphics PNGs are amazing for interactive documents such as web pages, but are not suitable for print. While PNGs are "lossless," meaning you can edit them and not lose quality, they are still low resolution. The reason PNGs are used in most web projects is that you can save your image with more colors on a transparent background. This makes for a much sharper, web-quality image. GIF - Graphics Interchange Format GIFs are most common in their animated form, which are all the rage on Tumblr pages and in banner ads. In their more basic form, GIFs are formed from up to 256 colors in the RGB colorspace. Due to the limited number of colors, the file size is drastically reduced. This is a common file type for web projects where an image needs to load very quickly, as opposed to one that needs to retain a higher level of quality. (Lee, 2018) TIFF - Tagged Image File A TIF is a large raster file that doesn't lose quality. This file type is known for using "lossless compression," meaning the original image data is maintained regardless of how often you might copy, re-save, or compress the original file. Despite TIFF images' ability to recover their quality after manipulation, you should avoid using this file type. It can take forever to load. TIFF files are also commonly used when saving photographs for print. (Lee, 2018) PSD - Photoshop Document PSDs are files that are created and saved in Adobe Photoshop, a popular graphics editing software. This type of file contains "layers" that make modifying the image much easier to handle. The largest disadvantage to PSDs is that Photoshop works with raster images as opposed to vector images. (Lee, 2018) raster images are pixel-based, they suffer a malady called image degradation. Just like photographic images that get blurry and imprecise when blown up, a raster image gets jagged and rough. (connection, 2020) vector images. vector-based graphics are more malleable than raster images — thus, they are much more versatile, flexible and easy to use. The most obvious advantage of vector images over raster graphics is that vector images are quickly and perfectly scalable. (connection, 2020) PDF - Portable Document Format PDFs were invented by Adobe with the goal of capturing and reviewing rich information from any application and device. If a designer saves your vector logo in PDF format, you can view it without any design editing software and they have the

ability to use this file to make further manipulations. (Lee, 2018) EPS - Encapsulated Postscript EPS is a file in vector format that has been designed to produce high-resolution graphics for print. Almost any kind of design software can create an EPS. The EPS extension is more of a universal file type (much like the PDF) that can be used to open vector-based artwork in any design editor. (Lee, 2018)

What is Image processing?

Now that we have a good idea of what exactly an Image is, we can move on to understanding the subject of many conversations in the sector research and computer science. The subject is image processing. In this data age that we live in, data scientists, students and researchers have been coming up with advanced ways and technics to extract information from any piece of data. Images have not been left out in this process. Images have proven to be every useful because of the human vision ability that can quickly assimilate large amounts of information within a very short time. It is said that a picture is worth a thousand words, this is particularly true because it is much easier for the brain to understand and remember what it interprets with vision that any other sense. Computers scientists, data scientists and researchers have been trying to give this interpretation ability and data extraction for better understanding to computers. By definition, image processing is a method to convert an image into digital form and perform some operations on it, in order to get an enhanced image or to extract some useful information from it. it is a type of signal dispensation in which the input is an image and the output is an image or characteristics associated with that image. (R. Suganya, 2018) What are the uses of Image processing? Today there is almost no area of technical endeavour that is not impacted in some way by digital image processing. As optics, imaging sensors, and computational technology advanced, image processing has become more commonly used in many different areas. Some areas of application of digital image processing include image enhancement for better human perception, image compression and transmission, as well as image representation for automatic machine perception (ElizaYingzi Du). We can cover

only a few of their applications in the context and space of the current discussion. In general, the fields that use digital image processing techniques can be divided into criminology, morphology, microscopy, photography, remote sensing, medical imaging, forensics, transportation and military application but not limited to.

Criminology / Forensics: The growth of sophisticated image processing and editing software has made the manipulation of digital images easy and imperceptible to the naked eyes. This has increased the demand to assess the trustworthiness of digital images when used in crime investigation, as evidence in court of law and for surveillance purposes (Ms. Neha Singh, 2015). Ideally, the image will be clear, with all persons, settings, and objects reliably identifiable. Unfortunately, though, that is not always the case, and the photograph or video image may be grainy, blurry, of poor contrast, or even damaged in some way. In such cases, investigators may rely on computerized technology that enables digital processing and enhancement of an image. (Thomas)

Medical Imaging: This is a technology that can be used to generate images of a human body (or part of it). These images are then processed or analysed by experts, who provide clinical prescription based on their observations. Ultrasonic, X-ray, Computerized Tomography (CT) and Magnetic Resonance Imaging (MRI) are quite often seen in daily life, though different sensory systems are individually applied. (James J. Park, 2017)

Remote Sensing: This is technology of employing remote sensors to gather information about the Earth. Usually the techniques used to obtain the information depend on electromagnetic radiation, force fields, or acoustic energy that can be detected by cameras, radiometers, lasers, radar systems, sonar seismographs, thermal meters. Image processing is almost always the first step of any remote sensing application project but it is often given greater significance than it deserves. In fact, one of the main objectives of image processing is to optimise visualisation of particular thematic dataset. Visual interpretation is therefore essential. Thematic maps are the most important products of remotely sensed imagery and they are derived either by visual interpretation or image segmentation (computerised classification). (Mason, 2016)

Military: digital image processing has been widely deployed for defence and security applications such as small target detection and tracking, missile guidance,

vehicle navigation, wide area surveillance, and automatic/aided target recognition. One goal for an image processing approach in defence and security applications is to reduce the workload of human analysts in order to cope with the ever-increasing volume of image data that is being collected. A second, more challenging goal for image processing researchers is to develop algorithms and approaches that will significantly aid the development of fully autonomous systems capable of decisions and actions based on all sensor inputs (Eliza Yingzi Du).

Transportation: This is a new area that has just been developed in recent years. One of the key technological progresses is the design of automatically driven vehicles, where imaging systems play a vital role in path planning, obstacle avoidance and servo control. With the development of digital image processing technology, traffic video monitoring technology based on image processing technology has become an important frontier research field of intelligent transportation system. (Wenshuai Ji, 2016)

Industrial inspection/ quality control: A major area of digital image processing is in automated visual inspection of manufactured goods. A system has a controller board for a CD_ROM drive. A typical image processing task with products like this is to inspect them for missing parts. Detecting anomalies like there is a major theme of industrial inspection that includes other products such as wood and cloth. (Oprea, Lita, Jurianu, Visan, & Cioc)

Digital Camera Images: Digital cameras generally include dedicated digital image processing chips to convert the raw data from the image sensor into a color-corrected image in a standard image file format. Images from digital cameras often receive further processing to improve their quality, a distinct advantage that digital cameras have over film cameras. The digital image processing typically is executed by special software programs that can manipulate the image in many ways. Many digital cameras also enable viewing of histograms of images, as an aid for the photographer to understand the rendered brightness range of each shot more readily. (Florin)

Image processing operations

There are a lot of operations that could be executed in image processing. The following are some of the operations but the list is not limited to these listed

operations.

- Changing Colorspaces
- Geometric Transformations of Images
- Image Thresholding
- Smoothing Images
- Morphological Transformations
- Image Gradients
- Canny Edge Detection
- Image Pyramids
- Image Transforms in OpenCV

In this section, I will demonstrate how some of the above operations can be done with the python library called OpenCV. What is python and why python?

According to the official Python execute summary, Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. (Python, 2020) The above statement explains both the what and why question. Python is equipped with the necessary packages and tools that make it possible for user to do many tasks and operations such as image processing. Apart from OpenCV, python also has the Pillow library that can be installed and used for basic operations like cropping and

resizing. If we want to perform more advanced operations using Pillow, we must always convert the image to digital form, because by default the images in pillow are not automatically in digital form. We also have Scikit learn and Matplotlib that are also used for image processing.

Changing Colorspaces

Our aim here is to change or convert the color spaces in an image for example, from BGR to Gray. A color space is defined as all the possible tones generated by a color model. A color model is actually a set of equations and/or rules used to calculate a given color and are capable of describing a wide range of tones from the visible spectrum of light. (Marco A. Pérez Cisneros, 2019). This property of images is important because according to (Shreyank N. Gowda, 2019) Image classification is a fundamental application in image processing. Recently, deeper networks and highly connected networks have shown state of the art performance for image classification tasks. Most datasets these days consist of a finite number of color images. These color images are taken as input in the form of RGB images and classification is done without modifying them. Now that we know the importance of changing Colorspaces, we will now execute the operation using python and OpenCV as earlier mentioned.

Geometric Transformations of Images

According to (Rhody), Geometric transformations are widely used for image registration and the removal of geometric distortion. Common applications include construction of mosaics, geographical mapping, stereo and video. A spatial transformation of an image is a geometric transformation of the image coordinate system. It is often necessary to perform a spatial transformation to:

- Align images that were taken at different times or with different sensors
- Correct images for lens distortion
- Correct effects of camera orientation
- Image morphing or other special effects

Transformation OpenCV provides two transformation functions, `cv2.warpAffine` and `cv2.warpPerspective`, with which you can have all kinds of transformations. `cv2.warpAffine` takes a 2x3 transformation matrix while

`cv2.warpPerspective` takes a 3x3 transformation matrix as input. (Revision, 2020)

Scaling

is just resizing of the image. OpenCV comes with a function `cv2.resize()` for this purpose. The size of the image can be specified manually, or you can specify the scaling factor. Different interpolation methods are used. Preferable interpolation methods are `cv2.INTER_AREA` for shrinking and `cv2.INTER_CUBIC` (slow) & `cv2.INTER_LINEAR` for zooming. By default, interpolation method used is `cv2.INTER_LINEAR` for all resizing purposes.

Translation

Translation is the shifting of object's location. (Revision, 2020). In the image below, notice the difference in the image from the top left edge in the original photo and the image that has been translated

Rotation According to (Revision, 2020), Rotation of an image for an angle θ is achieved by the transformation matrix of the form

$$M = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

But OpenCV provides scaled rotation with adjustable centre of rotation so that you can rotate at any location you prefer.

Modified transformation matrix is given by where:

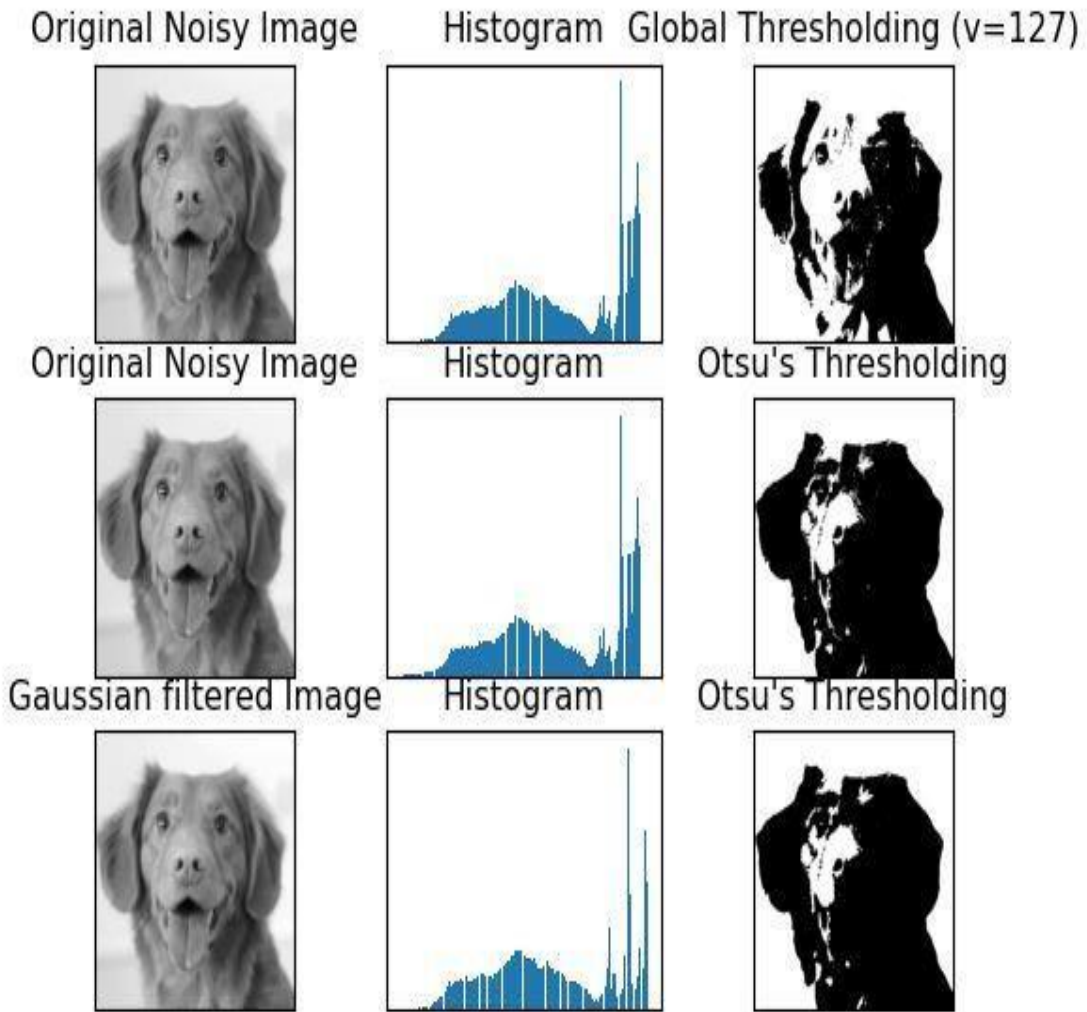
$$\begin{bmatrix} \alpha & \beta & (1 - \alpha) \cdot \text{center.x} - \beta \cdot \text{center.y} \\ -\beta & \alpha & \beta \cdot \text{center.x} + (1 - \alpha) \cdot \text{center.y} \end{bmatrix}$$

To find this transformation matrix, OpenCV provides a function, `cv2.getRotationMatrix2D`

$$\alpha = scale \cdot \cos \theta,$$

$$\beta = scale \cdot \sin \theta$$

Figure 1



Perspective Transformation

For perspective transformation, you need a 3x3 transformation matrix. Straight lines will remain straight even after the transformation. To find this transformation matrix, you need 4 points on the input image and corresponding points on the

output image. Among these 4 points, 3 of them should not be collinear. Then transformation matrix can be found by the function `cv2.getPerspectiveTransform`. Then apply `cv2.warpPerspective` with this 3x3 transformation matrix (Revision, 2020).

Simple Thresholding

If pixel value is greater than a threshold value, it is assigned one value (may be white), else it is assigned another value (may be black). The function used is `cv2.threshold`. First argument is the source image, which should be a grayscale image. Second argument is the threshold value which is used to classify the pixel values. Third argument is the `maxVal` which represents the value to be given if pixel value is more than (sometimes less than) the threshold value. OpenCV provides different styles of thresholding and it is decided by the fourth parameter of the function. Different types are; (Revision, 2020) `cv2.THRESH_BINARY` `cv2.THRESH_BINARY_INV` `cv2.THRESH_TRUNC` `cv2.THRESH_TOZERO` `cv2.THRESH_TOZERO_INV`

Adaptive Thresholding

In the previous section, we used a global value as threshold value. But it may not be good in all the conditions where image has different lighting conditions in different areas. In that case, we go for adaptive thresholding. In this, the algorithm calculates the threshold for a small region of the image. So, we get different thresholds for different regions of the same image and it gives us better results for images with varying illumination. (Revision, 2020) Adaptive Method - It decides how thresholding value is calculated. • `cv2.ADAPTIVE_THRESH_MEAN_C`: threshold value is the mean of neighbourhood area. • `cv2.ADAPTIVE_THRESH_GAUSSIAN_C`: threshold value is the weighted sum of neighbourhood values where weights are a gaussian window. (Revision, 2020)

Smoothing Images.

According to (Revision, 2020) the main aim of image smoothing is to blur images

with various low pass filters and apply custom made filters to images (2D convolution). OpenCV provides a function, `cv2.filter2D ()`, to convolve a kernel with an image. As an example, we will try an averaging filter on an image. A 5x5 averaging filter kernel can be defined as follows: Filtering with the above kernel results in the following being performed: for each pixel, a 5x5 window is centered on this pixel, all pixels falling within this window are summed up, and the result is then divided by 25. This equates to computing the average of the pixel values inside that window. This operation is performed for all the pixels in the image to produce the output filtered image.

Image Blurring (Image Smoothing)

Image blurring is achieved by convolving the image with a low-pass filter kernel. It is useful for removing noise. It actually removes high frequency content (e.g.: noise, edges) from the image resulting in edges being blurred when this is filter is applied. (Well, there are blurring techniques which do not blur edges). OpenCV provides mainly four types of blurring techniques. (Revision, 2020) Averaging According to (Revision, 2020), this is done by convolving the image with a normalized box filter. It simply takes the average of all the pixels under kernel area and replaces the central element with this average. This is done by the function `cv2.blur()` or `cv2.boxFilter()`. We should specify the width and height of kernel. A 3x3 normalized box filter would look like this:

How to turn photo into cartoon online or on Windows/Mac

Moreover, sharing a photo cartoon on social media could attract more attention when others just post standard photos. We are going to share how to turn photo to cartoon on Windows, Mac, and online in this tutorial. With these photo editors and our guides, you can create cartoon at any time, even if you have not learnt any knowledge about painting. If you are ready, let's start right now.

The following are the steps to convert photo to cartoon in windows MAC: -

Method 1: Convert Photo into Cartoon Online Many people prefer to use online photo editors. They are compatible to more platforms and allow you to edit photos at anytime and anywhere. There are lots of online cartoon photo editor on the internet, you can choose one of them to make your photo into cartoon. → Cartoon a Picture Online on Toonyphotos.com •

Step 1. Enter the toonyphotos.com on the browser and you can see the introduction of its photos to cartoons editing function.

- Step 2. Click Start Turning Photos into Cartoons button to enter the cartoon photo editing webpage and then click Choose Photo button to choose the photo from your computer. After uploading the photo, you start to outline the region you need to turn into cartoon. Stop outlining by clicking the right mouse button. Click Render to start cartoonize your photo.

- Step 3. After applying, you can click Download to directly download the cartoon photo or click View to preview the photo before downloading. If you are not satisfied with the photo, you may need to outline more region to get more detail cartoonized photo. Fig 3 → Convert photo into cartoon with Photoshop If you have Photoshop, you can also use it to turn photo into cartoon effect in your Windows 10/8/7 or Mac (macOS High Sierra included).

The photo-to-cartoon effect in Photoshop is a popular effect and surprisingly easy to achieve.

Here in this tutorial, you can learn to turn photos to cartoons using an assortment of filters and simple brushwork.

- Step 1. Open Photoshop and add your photo into the program.

- Step 2. Press Cmd/Ctrl+Shift+Alt+E to merge a new layer, then Cmd/Ctrl+J to

copy it. Rename the layer Sketch.

- Step 3. Make a new layer, drag it below the sketch layer, then go to Edit>Fill Layer. Set Use to White and click OK. Next highlight the sketch layer and click the Add Layer Mask icon in the Layers Panel. Grab the Brush tool and set color to black, then paint to tidy up the skin, clothes and hair.
- Step 4. Change the Blend Mode of the sketch layer to Multiply, then make a new layer and drag it below. Grab the Brush tool and choose a color for the skin, then begin painting. Make more new layers and paint different colors for the hair, eyes, mouth and jeans.
- Step 5. Duplicate the background layer. Drag it to below the sketch layer. Go to Filter>Brush Strokes>Ink Outlines. Leave the default settings and hit OK. Alt-click the Add Layer Mask icon to add a mask that hides the layer, then paint with white to reveal the tattoos. Next highlight the white layer.
- Step 6. Go to Filter>Textures>Grain. Set Grain Type: Vertical, Intensity 100, Contrast 0. Go to Filter>Blur>Gaussian Blur. Set Radius 4px. Go to Filter>Distort>Polar Coordinates and pick Rectangular to Polar. Hit Cmd/Ctrl+U, check Colorize, set Hue 211.

Paint white over any lines that show through.

Convert photo to cartoon with Paint.net Paint.net is a free photo editor for Windows, depending on the .Net framework. As one of the best alternatives for Microsoft Paints, Paint.net offers more effects and features, including make cartoon photos with personal images.

- Step 1: Import photo
- Step 2: Add effect

- Step 3: Simulate cartoon
- Step 4: Fill background, Then save the paint to local disk.

Convert photo to cartoon using Adobe illustrator

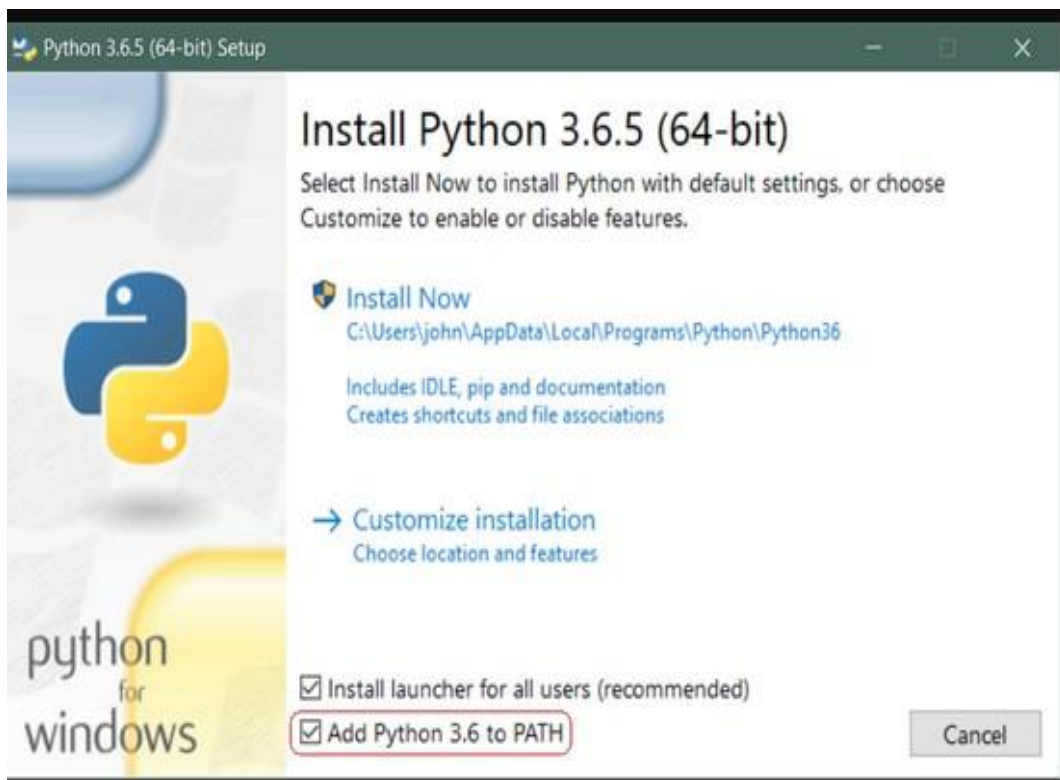
- Step 1: Get Yourself a Picture and Upload to Adobe Illustrator

Choose your favorite picture and upload it to Illustrator. Select a large enough picture so that you will be able to draw over it easily. This program is vector based, that means you will be able to scale the final artwork without losing its form. It will always look great if you want to make it larger or smaller

Installing Python and OpenCv

Download the Python 3 Installer

- Open a browser window and navigate to the Download page for Windows at python.org.
- Underneath the heading at the top that says Python Releases for Windows, click on the link for the Latest Python 3 Release - Python 3.x.x. (As of this writing, the latest is Python 3.6.5.)

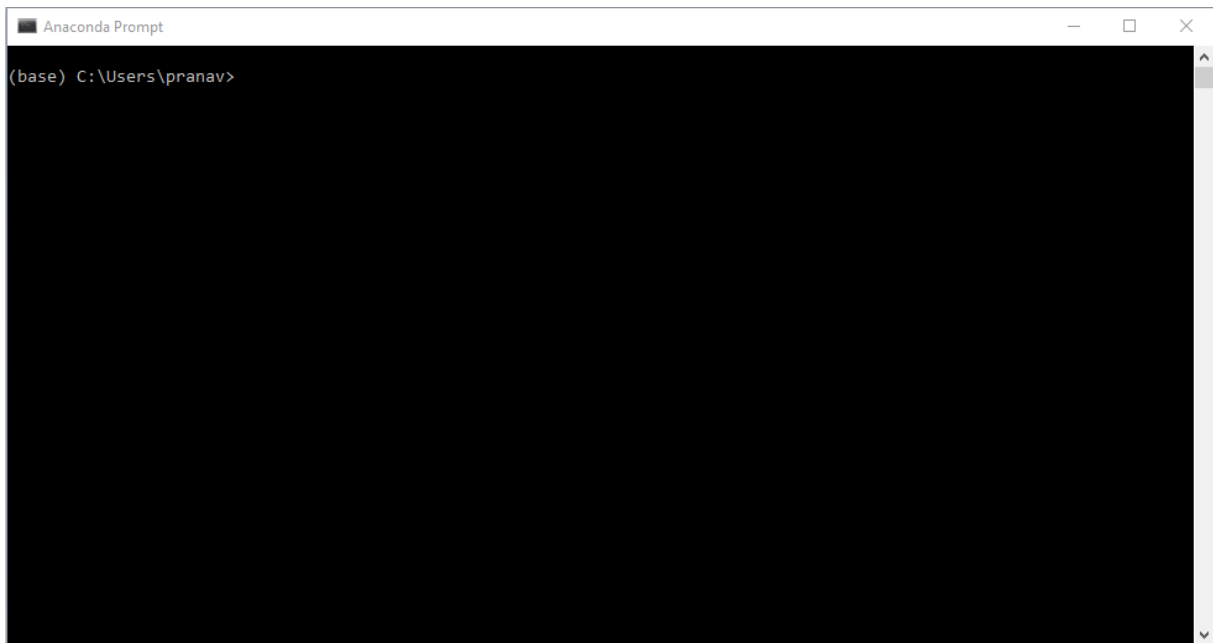


- Scroll to the bottom and select either Windows x86-64 executable installer for 64-bit or Windows x86 executable installer for 32-bit.
- Once you have chosen and downloaded an installer, simply run it by double-clicking on the downloaded file. A dialog should appear .
- Then just click Install Now. That should be all there is to it. A few minutes later you should have a working Python 3 installation on your system. 4.1.2

Installing OpenCV

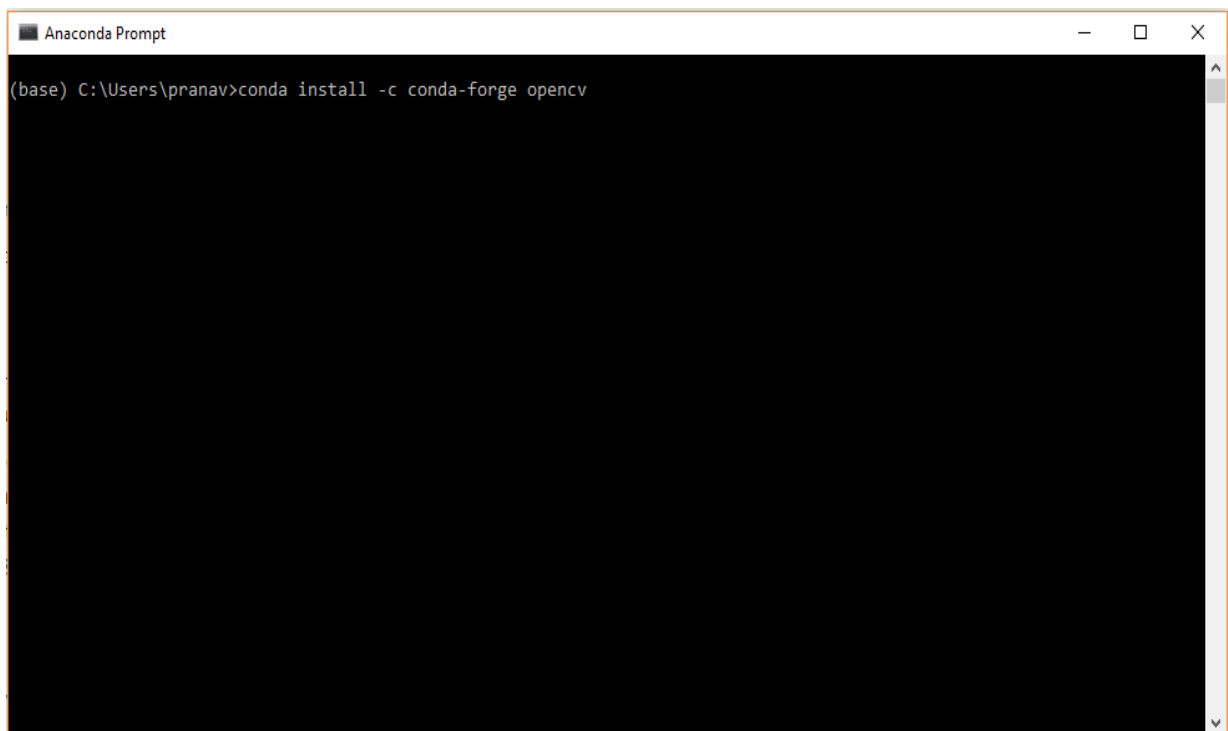
Launch the Anaconda prompt from the start menu:

We should see a similar prompt if we went according to instructions.



```
Anaconda Prompt
(base) C:\Users\pranav>
```

To install the OpenCV we need to type the following command at the prompt:



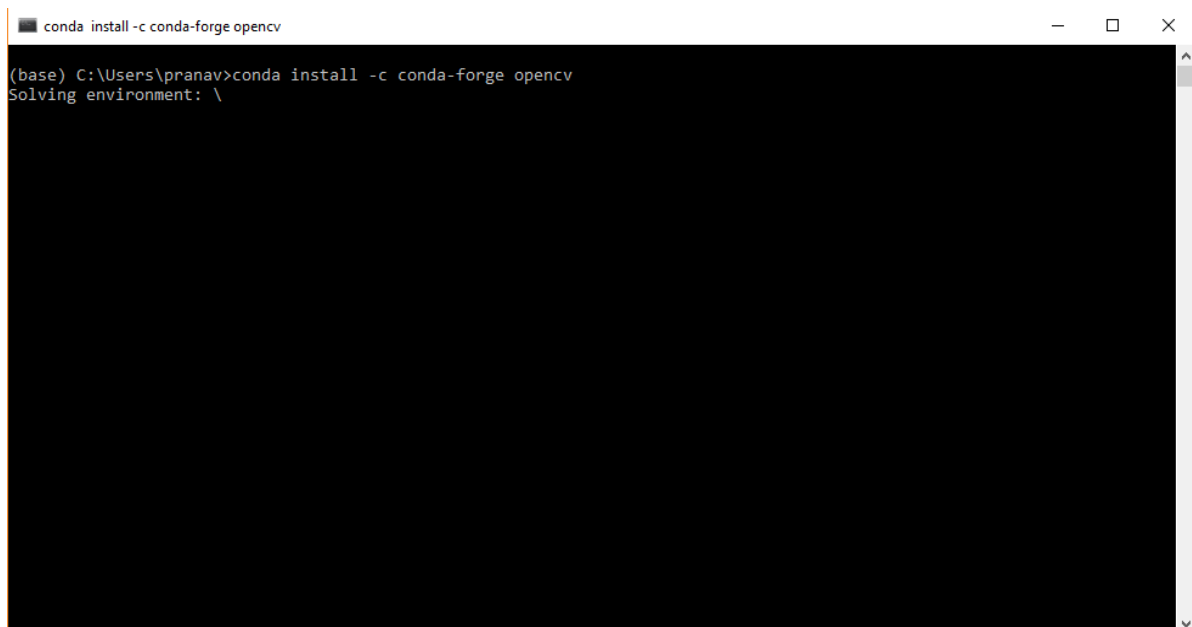
```
Anaconda Prompt
(base) C:\Users\pranav>conda install -c conda-forge opencv
```

`conda install -c conda-forge opencv .`

prompt will show that it is “solving environment”.

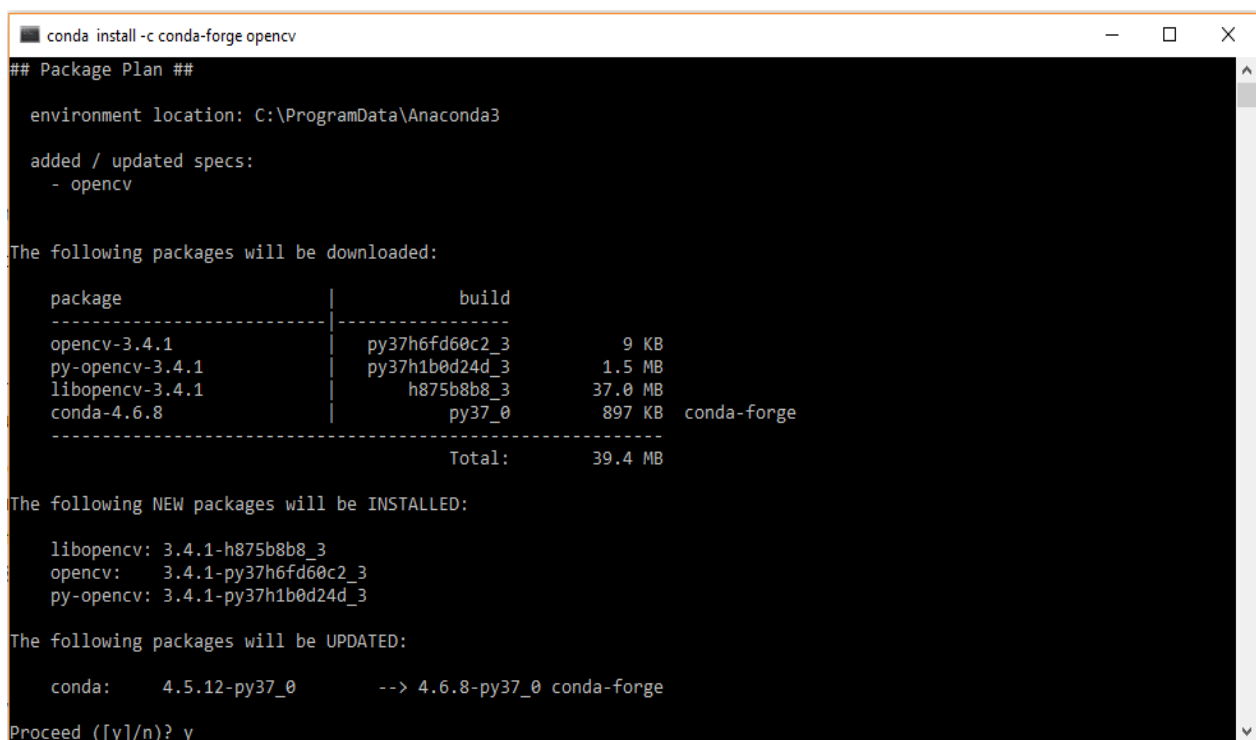
It takes quite a bit of time if you are on slower Internet connection.

Try to use faster Internet, preferably a wired connection for uninterrupted and best results. if you are in a work or institution based Internet connection



```
conda install -c conda-forge opencv
(base) C:\Users\pranav>conda install -c conda-forge opencv
Solving environment: \
```

then it is likely that an HTTP timeout will occur and we need to enter the command once again and restart the procedure.



```
conda install -c conda-forge opencv
## Package Plan ##
environment location: C:\ProgramData\Anaconda3
added / updated specs:
- opencv

The following packages will be downloaded:

package | build | size
-----|-----|-----
opencv-3.4.1 | py37h6fd60c2_3 | 9 KB
py-opencv-3.4.1 | py37h1b0d24d_3 | 1.5 MB
libopencv-3.4.1 | h875b8b8_3 | 37.0 MB
conda-4.6.8 | py37_0 | 897 KB conda-forge
-----|-----|-----
Total: | | 39.4 MB

The following NEW packages will be INSTALLED:

libopencv: 3.4.1-h875b8b8_3
opencv: 3.4.1-py37h6fd60c2_3
py-opencv: 3.4.1-py37h1b0d24d_3

The following packages will be UPDATED:

conda: 4.5.12-py37_0 --> 4.6.8-py37_0 conda-forge

Proceed ([y]/n)? y
```

Once the environment is resolved by conda it will list the packages that will be installed, namely: opencv, libopencv, py-opencv.

Enter y to proceed with the installation.

```
Administrator: Anaconda Prompt - python
Please update conda by running
$ conda update -n base -c defaults conda

## Package Plan ##

environment location: C:\ProgramData\Anaconda3

added / updated specs:
- opencv

The following NEW packages will be INSTALLED:

libopencv: 3.4.1-h875b8b8_3
opencv:    3.4.1-py37h6fd60c2_3
py-opencv: 3.4.1-py37h1b0d24d_3

The following packages will be UPDATED:

conda:      4.5.12-py37_0      --> 4.6.8-py37_0 conda-forge

Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

We can verify if the installation was successful by launching the python interpreter. opencv is referred to as cv2 in python. Type at the prompt: • import cv2 : if the prompt is displayed then opencv then python has successfully imported the opencv library.

But we should also verify the version of opencv so we need to type: • print(cv2.version) : as of March 2019 the version displayed is 3.4.1 which is the officially supported version of opencv by anaconda environment. If we want to work with a different version then while installing .

```
python
(base) C:\Users\pranav>python
Python 3.7.1 (default, Dec 10 2018, 22:54:23) [MSC v.1915 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> print(cv2.__version__)
3.4.1
>>>
```

LITERATURE SURVEY:

An preface to image conflation with GAN. A couple of times back, there had been tremendous growth in the exploration of GAN (Generative Adversarial Network) (9). GAN was put forward in the time 2014 where it was introduced in colorful operations similar as deep literacy, natural language processing (NLP). From this paper, we explored the different styles of image conflation similar as direct system, Hierarchal system and iterative system (3).

They spoke about two styles of image conflation which are “textbook-to-image conversion” and “image-to- image restatement”. In textbook-to- photo conversion, current styles worked well on a pre-defined dataset where each image contains one object similar as Caltech-UCSD Birds (6) and Oxford102 (7), but the performance on complex datasets similar as MSCOCO (14) is important poor. While some of the models were successful in producing realistic images of apartments in LSUN (16) because the apartments didn’t contain any living effects. So, they got a successful photo of the room because living effects are more complicated to convert also stationary objects. This was the current limitation in this model and it was necessary to learn different generalities of the object. To ameliorate the performance of GAN and enhance affair in the task they trained different types of models that would induce a single object and train another model which would learn to combine colorful objects according to textbook descriptions, and that CapsNet (10). Now coming to image- image restatement, they bandied some general models from supervised to unsupervised settings which are pixel-wise loss (13), cyclic loss (14) and tone- distance loss (12). Away from this they also proposed some image – image restatement model for face editing, videotape vaticination and, image super-resolution. We can say that image- image restatement was a catchy operation of GAN which had a great compass for mobile operation. Although during the exploration unsupervised system were seem to be more popular compared to the supervised

system

Bus-painter cartoon image generation from sketch using by using cGAN. The authors studied colorful problems faced by sketch artist while sketching colorful

black and white cartoon delineations, for consideration coloring of colorful sketches, mixing of the different colors to get a unique shade for a particular sketch. According to their exploration, some difficulties were faced by the user to get a unique or asked color they need after mixing two or further colors. So, to overcome this problem an operation was introduced which was known as “sketch-to- image conflation while using tentative generative inimical networks” (cGAN) (3). Latterly on, they plant that the operation faced a problem and failed to give the asked affair. To avoid this issue, they constructed the Bus-painter model which could automatically induce suitable colors for a sketch. Their operation was grounded on tentative GAN with ‘Unit’ (2) structure which allowed the affair image to have both low- position information of sketch as well as learned high- position color information. They also innovated further constraints grounded on the pix2pix (20) model to gain finer oil. Then they worked on the autopainter to alter to color control so that their network could alter the combined result which could satisfy the stoner by colorful colors. The result showed that the bus-painter could induce a polished amped image from the two given datasets.

In malignancy of the guaranteed result, the system suffered from difficulties of conforming parameters just like other Learning models. Also, the combination network structure Redounded in lower speed for training.

METHODOLOGY:

GAN i.e. “Generative Adversarial Networks” algorithm is used for the fulfillment. GAN is a combination of a generative model and a discrimination AL model. The generative model creates new prototypes of data that Duplicate the training data. The discriminator is the model used for testing the data and comparing it with the image from the Generator. Discriminator decides whether the production image is fake or real. The Generator and discriminator both are part of neural networks and both run within competition with each other in the training phase. The track are repeated multiple times so that the instituter and discriminator get a better result after the replay of Line. The working can be conjured by the figure given below.

Now coming to the design, we've used for about 300 images for training the project from Flickr (Flickr is an image hosting service and tape hosting service). Compare to the antecedent approach of cartoon GAN they have training data that contains the real- world photograph and the cartoon image but, in our game, we only train the real- world image that's captured with the help of a camera. Presently the training data cohere of (photograph) real- world images which were passed through 200 days, trained to produce the cartoon picture with the help of father were it reconstructs the cartoon image with the help of the antagonistic function (that comes other loss function) and the images are trained again on the same content loss (that falls under loss function). The final handwork is generated with the complicacy block. Thereafter the discriminator checks if the image generated is real or fake. To classify whether an image is fake or real is comparatively a less demanding task as compared to generating an image Using GANs is really productive since it provides high- quality cartoonized images. To transfigure videotapes into animated or Cartoonized videotapes, we have further used the cv2 (Computer vision play) which is a library in python the vid is divided into frames which are dependent on the specified time period

$$\text{fps} = 0.5 \text{ s}$$

You can change the time period and Number of frames will differ. Thereafter on, these frames will be brought together (joined) by `os.join` to convert into videotape strings (. mp4 or. avi).

Conversion of Video is largely analogous to getting an animated tape out of the normal bone. The tape is first divided into frames and saved into an array, either passed through the author and discriminator with the help of OpenCV to get the amped (cartoonized videotape). The below exertion plate shows the working of the scheme. Where it gives an overall brief idea on how the images and videotape taken from the camera are converted into cartoon style.

TOOLS USED:

Python

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace.

Platform: PyCharm.

MODULES USED:

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

PROPOSED SYSTEM:

We propose to use neural style transfer which is a machine learning algorithm, which involves two images, first is the input image from the user and second is the style image which is used to apply the style on the input image. Following are the examples of images generated using neural style transfer. Fig 4 We propose to create a website, which consists of image upload functionality using which the user can upload his image, the uploaded image is then processed by server using Neural style transfer algorithm and the resulting image is presented to the user on the website. Which then user can download & share. Neural fast style transfer is used by Apps such as <https://deepart.io>, Prisma, Artisto etc. We decided to choose this approach over traditional image filters (e.g. using image filters such as median & bilateral filters to posterize an Image) as Neural fast style transfer is quite new and challenging technique which uses machine learning & image processing to produce various styled images based on variety of input & style images. The algorithm can be implemented in Python/JavaScript/Lua to perform neural style transfer. We will use Python to implement the backend and the front end of the website will be in HTML, CSS & JS. Basically, in Neural Style Transfer we have two images- style and content. We need to copy the style from the style image and apply it to the content image. By, style we basically mean, the patterns, the brushstrokes, etc. we will provide a set of style images which a user can use to apply different kinds of Cartoon like effects to his image.

ALGORITHM:

Our implementation uses TensorFlow to train a fast style transfer network. We use roughly the same transformation network as described in Justin Johnson et. al, except that batch normalization is replaced with Ulyanov's instance normalization. We use a loss function close to the one described in Gatys, using VGG19 instead of VGG16 and typically using "shallower" layers than in Johnson's implementation (e.g. we use relu1_1 rather than relu1_2). Empirically, this results in larger scale style features in transformations. Fig 8 •

Step 1. Train a feed forward neural network for each style

Step 2. fetch the uploaded Content image(i.e. uploaded user image) & trained feed forward network for the selected style image (i.e. style selected by user)

Step 3. Calculate the Cost/Loss function of Generated Image G using

$$f(G) = f_{\text{Content}}(C,G) + f_{\text{Style}}(S,G)$$

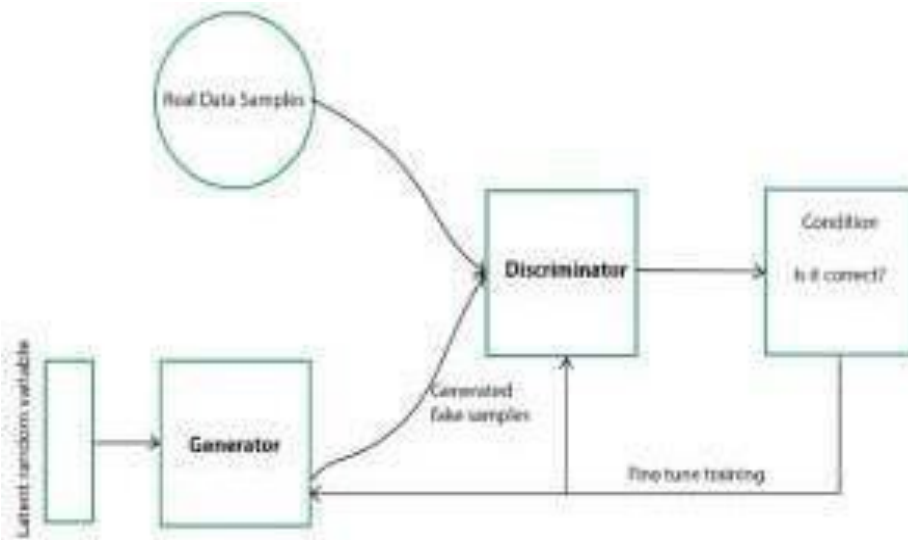
where C : Content Image &

S: Style Image $f_{\text{Content}}(C,G)$ – Content Loss Function $f_{\text{Style}}(S,G)$ – Style Loss Function

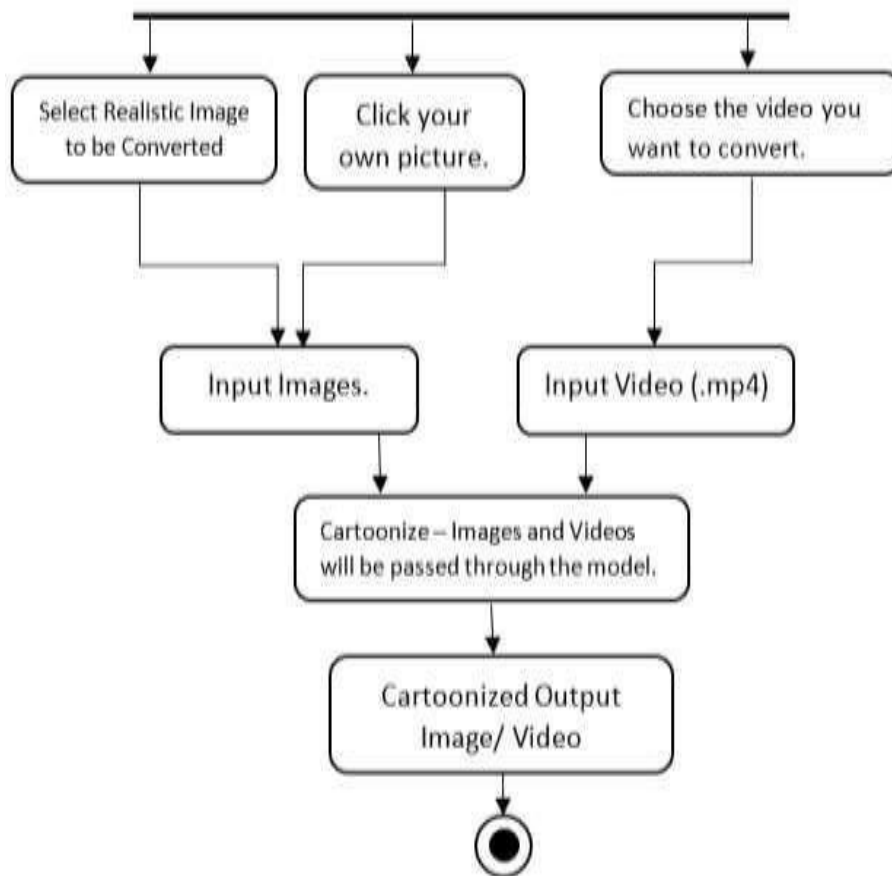
Step 4. After minimizing the loss function $f(G)$, Stylize the Image by single forward pass through image transformation network •

Step 5. Display Styled Image to the user

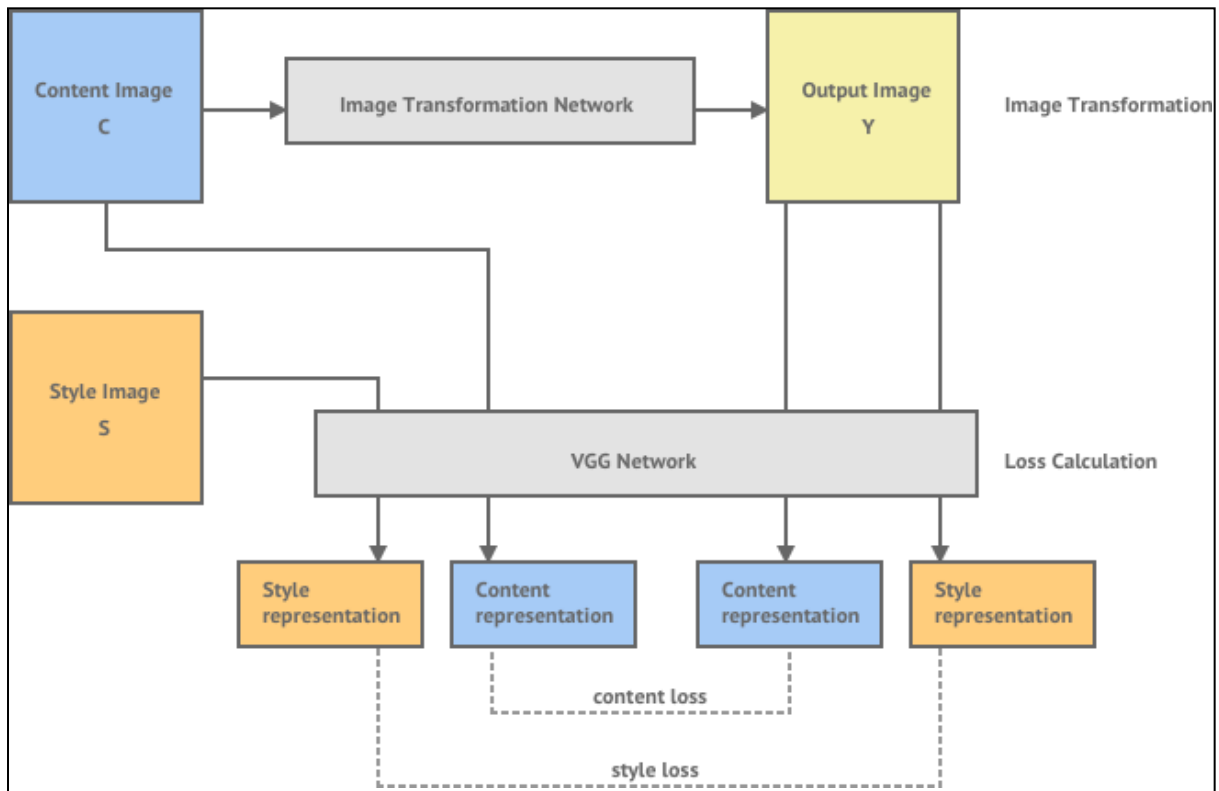
WORKING MODEL OF GAN:



ACTIVITY DIAGRAM:



BLOCK DIAGRAM :



H/W AND S/W REQUIREMENT

A. *Hardware Requirements:*

- Windows 10, 64 bits PC or 64 Bit Mac OS X High Sierra computers
- Any CPU (Intel i5/ i7/ Xeon recommended).
- Nvidia GPUs (minimum 2GB Recommended)
- At least 8 GB RAM, 10 GB HDD Free Space.

B. *Software Requirements:*

- Anaconda Framework for Python IDE & Packages (Recommended), Jupyter notebook, Tensorflow machine Learning library
- Programming Language: Python

STEP BY STEP CODE EXPLANATION:

Step 1: Importing the required modules

```
import cv2 #for image processing
import easygui #to open the filebox
import numpy as np #to store image
import imageio #to read image stored at particular path
import sys
import matplotlib.pyplot as plt
import os
import tkinter as tk
from tkinter import filedialog
from tkinter import *
from PIL import ImageTk, Image
```

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

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

Step 3: how image is stored

```
originalImage = cv2.imread(ImagePath)
originalImage = cv2.cvtColor(originalImage, cv2.COLOR_BGR2RGB)
#print(image) # image is stored in form of numbers
# confirm that image is chosen
if originalImage is None:
    print("Can not find any image. Choose appropriate file")
    sys.exit()
ReSized1 = cv2.resize(originalImage, (960, 540))
```

Step 4: Transforming an image to grayscale

```
grayScaleImage = cv2.cvtColor(originalImage, cv2.COLOR_BGR2GRAY)
ReSized2 = cv2.resize(grayScaleImage, (960, 540))
```

Step 5: Smoothing a grayscale image

```
smoothGrayScale = cv2.medianBlur(grayScaleImage, 5)
ReSized3 = cv2.resize(smoothGrayScale, (960, 540))
```

Step 6: Retrieving the edges of an image

```
getEdge = cv2.adaptiveThreshold(smoothGrayScale, 255,  
cv2.ADAPTIVE_THRESH_MEAN_C,  
cv2.THRESH_BINARY, 9, 9)  
ReSized4 = cv2.resize(getEdge, (960, 540))
```

Step 7: Preparing a Mask Image

```
colorImage = cv2.bilateralFilter(originalImage, 9, 300, 300)  
ReSized5 = cv2.resize(colorImage, (960, 540))
```

Step 8: Giving a Cartoon Effect

```
cartoonImage = cv2.bitwise_and(colorImage, colorImage, mask=getEdge)  
ReSized6 = cv2.resize(cartoonImage, (960, 540))
```

Step 9: Plotting all the transitions together

```
images=[ReSized1, ReSized2, ReSized3, ReSized4, ReSized5, ReSized6]  
fig, axes = plt.subplots(3,2, figsize=(8,8), subplot_kw={'xticks':[], 'yticks':[]},  
gridspec_kw=dict(hspace=0.1, wspace=0.1))  
for i, ax in enumerate(axes.flat):  
ax.imshow(images[i], cmap='gray')  
//save button code  
  
plt.show()
```

Step 10: Functionally of save button

```
def save(ReSized6, ImagePath):  
#saving an image using imwrite()  
newName="cartoonified_Image"  
path1 = os.path.dirname(ImagePath)  
extension=os.path.splitext(ImagePath)[1]  
path = os.path.join(path1, newName+extension)  
cv2.imwrite(path, cv2.cvtColor(ReSized6, cv2.COLOR_RGB2BGR))  
I = "Image saved by name " + newName + " at " + path  
tk.messagebox.showinfo(title=None, message=I)
```

Step 11: Making the main window

```
top=tk.Tk()  
top.geometry('400x400')  
top.title('Cartoonify Your Image !')  
top.configure(background='white')  
label=Label(top,background='#CDCDCD', font=('calibri',20,'bold'))
```

Step 12: Making the Cartoonify button in the main window

```
upload=Button(top,text="Cartoonify an Image",command=upload,padx=10,pady=5)
upload.configure(background='#364156', foreground='white',font=('calibri',10,'bold'))
upload.pack(side=TOP,pady=50)
```

Step 13: Making a Save button in the main window

```
save1=Button(top,text="Save cartoon image",command=lambda: save(ImagePath,
ReSized6),padx=30,pady=5)
save1.configure(background='#364156', foreground='white',font=('calibri',10,'bold'))
save1.pack(side=TOP,pady=50)
```

Step 14: Main function to build the tkinter window

```
top.mainloop()
```

CODE:

```
import cv2 #for image processing

import easygui #to open the filebox

import numpy as np #to store image

import imageio #to read image stored at particular path

import sys

import matplotlib.pyplot as plt

import os

import tkinter as tk

from tkinter import filedialog

from tkinter import *

from PIL import ImageTk, Image

top=tk.Tk()

top.geometry('400x400')

top.title('Cartoonify Your Image !')

top.configure(background='white')

label=Label(top,background='#CDCDCD', font=('calibri',20,'bold'))

def upload():
```

```
ImagePath=easygui.fileopenbox()
```

```
cartoonify(ImagePath)
```

```
def cartoonify(ImagePath):
```

```
    # read the image
```

```
    originalImage = cv2.imread(ImagePath)
```

```
    originalImage = cv2.cvtColor(originalImage, cv2.COLOR_BGR2RGB)
```

```
    #print(image) # image is stored in form of numbers
```

```
    # confirm that image is chosen
```

```
    if originalImage is None:
```

```
        print("Can not find any image. Choose appropriate file")
```

```
        sys.exit()
```

```
    ReSized1 = cv2.resize(originalImage, (960, 540))
```

```
    #plt.imshow(ReSized1, cmap='gray')
```

```
    #converting an image to grayscale
```

```
    grayScaleImage= cv2.cvtColor(originalImage, cv2.COLOR_BGR2GRAY)
```

```
    ReSized2 = cv2.resize(grayScaleImage, (960, 540))
```

```
    #plt.imshow(ReSized2, cmap='gray')
```

```
#applying median blur to smoothen an image

smoothGrayScale = cv2.medianBlur(grayScaleImage, 5)

ReSized3 = cv2.resize(smoothGrayScale, (960, 540))

#plt.imshow(ReSized3, cmap='gray')

#retrieving the edges for cartoon effect

#by using thresholding technique

getEdge = cv2.adaptiveThreshold(smoothGrayScale, 255,

    cv2.ADAPTIVE_THRESH_MEAN_C,

    cv2.THRESH_BINARY, 9, 9)

ReSized4 = cv2.resize(getEdge, (960, 540))

#plt.imshow(ReSized4, cmap='gray')

#applying bilateral filter to remove noise

#and keep edge sharp as required

colorImage = cv2.bilateralFilter(originalImage, 9, 300, 300)

ReSized5 = cv2.resize(colorImage, (960, 540))

#plt.imshow(ReSized5, cmap='gray')
```

```

#masking edged image with our "BEAUTIFY" image

cartoonImage = cv2.bitwise_and(colorImage, colorImage, mask=getEdge)

ReSized6 = cv2.resize(cartoonImage, (960, 540))

plt.imshow(ReSized6, cmap='gray')

# Plotting the whole transition

images=[ReSized1, ReSized2, ReSized3, ReSized4, ReSized5, ReSized6]

fig, axes = plt.subplots(3,2, figsize=(8,8), subplot_kw={'xticks':[], 'yticks':[]},
gridspec_kw=dict(hspace=0.1, wspace=0.1))

for i, ax in enumerate(axes.flat):

    ax.imshow(images[i], cmap='gray')

save1=Button(top,text="Save cartoon image",command=lambda: save(ReSized6,
ImagePath),padx=30,pady=5)

save1.configure(background='#364156', foreground='white',font=('calibri',10,'bold'))

save1.pack(side=TOP,pady=50)

plt.show()

def save(ReSized6, ImagePath):

    #saving an image using imwrite()

```

```
newName="cartoonified_Image"
```

```
path1 = os.path.dirname(ImagePath)
```

```
extension=os.path.splitext(ImagePath)[1]
```

```
path = os.path.join(path1, newName+extension)
```

```
cv2.imwrite(path, cv2.cvtColor(ReSized6, cv2.COLOR_RGB2BGR))
```

```
I= "Image saved by name " + newName + " at " + path
```

```
tk.messagebox.showinfo(title=None, message=I)
```

```
upload=Button(top,text="Cartoonify an Image",command=upload,padx=10,pady=5)
```

```
upload.configure(background='#364156', foreground='white',font=('calibri',10,'bold'))
```

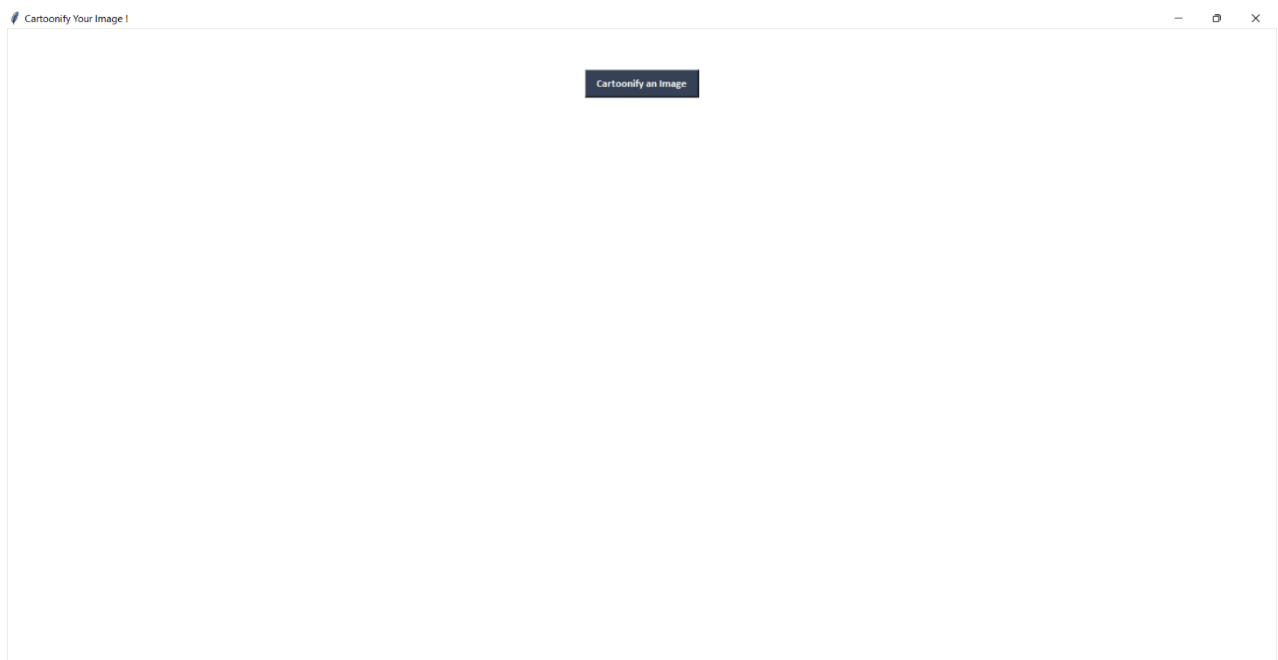
```
upload.pack(side=TOP,pady=50)
```

```
top.mainloop()
```

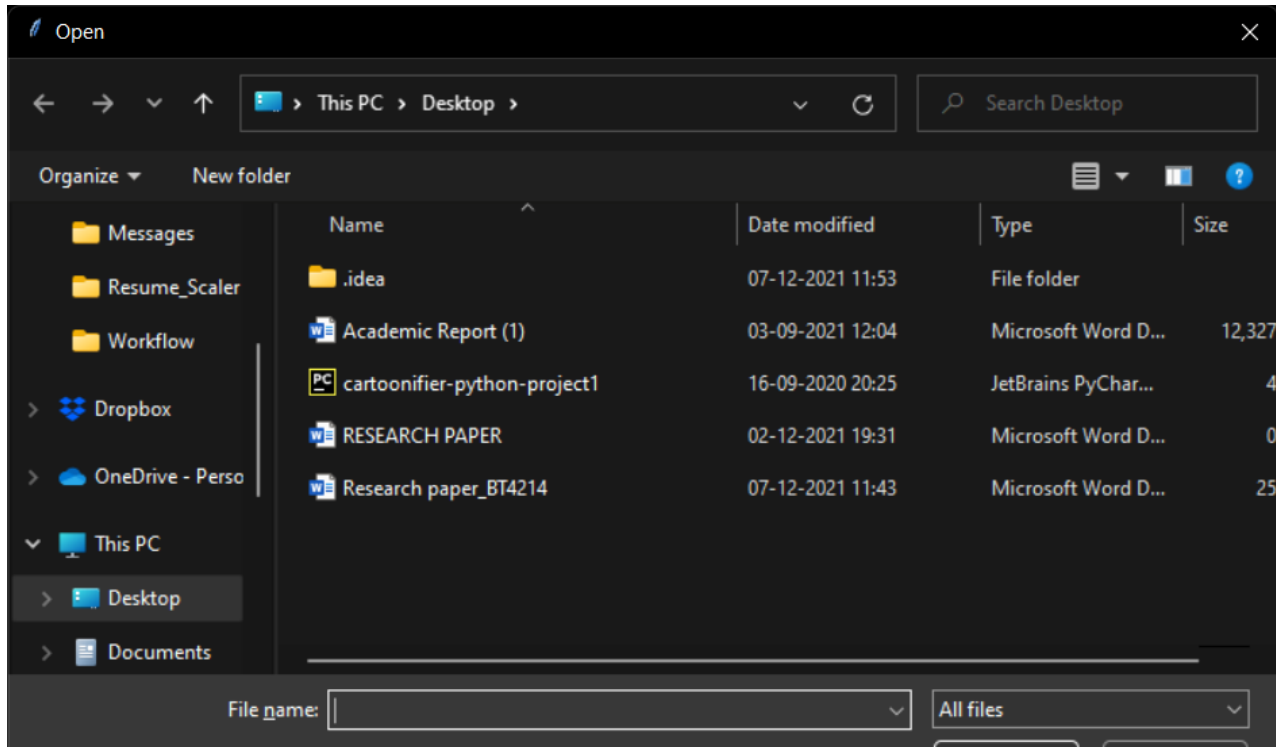

PROJECT OVERVIEW:

STEP 1): After executing above commands then double click on 'run.bat' file from Cartoon folder to get below screen.

STEP 2): In above screen click on 'Upload Image' button and select input image



STEP 3): After uploading image click on ‘Cartoonify Image’ button to get below screens

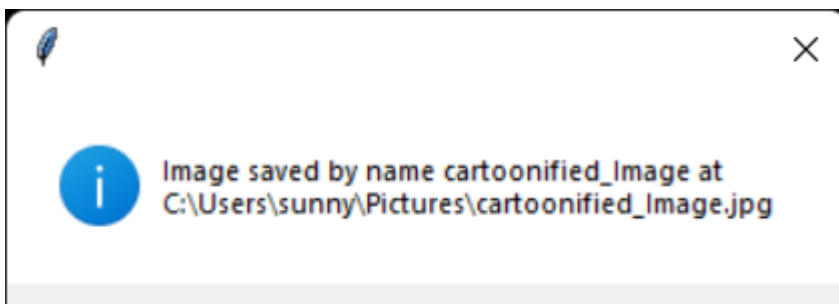


STEP 4): We will get above carton converted images and same images will saved inside output folder

Figure 1



STEP 5: click on save file.



RESULT:

The following project shows that the first image is the original image that we want to cartoonify and next photo is the cartooned photo of the image which is done with the help of GAN and opencv.

CONCLUSION:

In this paper with the use of CartoonGAN, where GAN stands for Generative Adversarial Network is used to transfigure images (shots) to the finest outlined image (amped image). With the help of the loss function and its two types named as Inimical loss and Content loss, we got a flexible as well as a clear edge defined images. Also with the help of CV2 which is known as Computer vision, we've converted videotape into vitality (cartooned image)

FUTURE SCOPE:

This project showed that the conversion of the image to cartooned image using GAN and also with the help of python library opencv.

In future we would like to convert these images into a more high definitive images that would help in future though we tried to use loss function to make sure that image is a high definitive we still failed.

Next we would convert it into 4K or at least in HD image.

REFERENCES:

- Eliza Yingzi Du, R. I.-H. (n.d.). Advanced Image Processing for Defense and Security Applications. *EURASIP Journal on Advances in Signal Processing*.
- Florin, T. (n.d.). Color processing in a digital camera pipeline . *Toadere Florin, "Color proceProc. SPIE 7297, Advanced Topics in Optoelectronics, Microelectronics, and Nanotechnologies IV, 729716*.
- James J. Park, V. L. (2017). *Advances in Computer Science and Ubiquitous Computing*. Taiwan: Springer.
- Lee, J. (2018). *10 Types of Image File Extensions and When to Use Them*. Retrieved from hubspot.com: <https://blog.hubspot.com/insiders/different-types-of-image-files>
- Marco A. Pérez Cisneros, D. Z. (2019). Color Spaces Advantages and Disadvantages in Image Color Clustering Segmentation. *ResearchGate*.
- Mason, J. G. (2016). *Image processing and GIS for remote Sensing*. John Wiley & Sons, Ltd.
- Ms. Neha Singh, D. S. (2015). Digital Image Forensics: Progress and Challenges. *Conference paper 31st National convention of Electronics and Telecommunication Engineers*.
- Oprea, S., Lita, I., Jurianu, M., Visan, D. A., & Cioc, I. B. (2008). *Digital image processing applied in drugs industry for detection of broken aspirin tablets*. IEEE.
- Python. (2020). *What is Python*. Retrieved from python: <https://www.python.org/doc/essays/blurb/>
- R. Suganya, S. R. (2018). *Big Data in Medical Image Processing*. CRC Press.
- Rafael C. Gonzalez, . E. (2013). *Digital Image Processing Using MATLAB*. McGraw Hill Education.
- Revision, A. M. (2020). *Docs OpenCV-Python Tutorials » Image Processing in OpenCV* . Retrieved from
OpenCV.
- Rhody, H. (n.d.). Geometric Image Transformations. *Geometric Image Transformations*.
- Shreyank N. Gowda, C. Y. (2019). ColorNet: Investigating the Importance of Color Spaces for Image Classification. *ACCV 2018*.
- Singh, H. (2019). *Practical Machine and Image Processing For Facial Recognition, Object Detection*., ISBN- 13 (electronic): 978-1-4842-4149-3.

Thomas, R. (n.d.). The Official Newsletter Of The National Association Of Investigative Specialists.

Investigative Article.

Wenshuai Ji, a. X. (2016). Research on the Application of Digital Image Processing Technology in Intelligent Transportation. *2nd International Conference on Materials Engineering and Information Technology Applications (MEITA 2016)*