VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590 018

An Internship Project Report
On

“Cartoonify Image"

Submitted in partial fulfillment of the requirements for the VIII Semester of degree of **Bachelor of Engineering in Information Science and Engineering** of Visvesvaraya Technological University, Belagavi

by

Ameena Taj Qureshi
1RN15IS136

Under the Guidance of

Mr. R Rajkumar
Associate Professor
Department of ISE

ESTD:2001
An Institute with a Difference

Department of Information Science and Engineering

RNS Institute of Technology

Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,
Channasandra, Bengaluru-560098

2021-2022

RNS INSTITUTE OF TECHNOLOGY

Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,
Channasandra, Bengaluru - 560098

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the Internship work entitled ***"Cartoonify Image"*** has been successfully completed by **Ameena Taj Qureshi (1RN15IS136)** a bonafide student of **RNS Institute of Technology, Bengaluru** in partial fulfillment of the requirements of 8th semester for the award of degree in **Bachelor of Engineering in Information Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during academic year **2021-2022**. The internship report has been approved as it satisfies the academic requirements in respect of internship work for the said degree.

Mr. R Rajkumar
Internship Guide
Associate Professor
Department of ISE

Dr. Suresh L
Professor and HoD
Department of ISE
RNSIT

Dr. M K Venkatesha
Principal
RNSIT

External Viva

Name of the Examiners — **Signature with Date**

1. ____________________ **1.** ____________________

2. ____________________ **2.** ____________________

DECLARATION

I, **Ameena Taj Qureshi [USN: 1RN15IS136]** student of VIII Semester BE, in Information Science and Engineering, RNS Institute of Technology hereby declare that the Internship work entitled ***Cartoonify Image*** has been carried out by us and submitted in partial fulfillment of the requirements for the *VIII Semester degree of* ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi* during academic year 2021-2022.

Place : Bengaluru

Date :

AMEENA TAJ QURESHI

(1RN15IS136)

TABLE OF CONTENTS

CHAPTER 1

CHAPTER 2

INTRODUCTION

CHAPTER 3

REQUIREMENT SPECIFICATION

CHAPTER 4

DESIGN AND IMPLEMENTATION

CHAPTER 5

CHAPTER 6

CHAPTER 7

CHAPTER 8

CHAPTER1

COMPANY OVERVIEW

We are an innovative organization with a goal to impart the aspiring learners with rigorous training and appropriate exposure for a promising future. We believe that pragmatic abilities weigh more over the educational degree. We strive forward to provide an existence where you can find your energy and transform it into your vocation. With a vision of a world loaded with opportunities and conceivable outcomes, we'll help you build a completely certain and assured foundation to stake guarantee on your skills for a brilliant future that awaits

An e-learning platform with a holistic environment to upskill students for Industry. Recognizing us as the avenue to provide mentor based training and internship programs directly from the oracles of the industry.

A vision to empower and upskill the youth of the future with absolute requisites and stance to seize better employment opportunities.

A mission to endow our mentees with skills in compliance with the emerging market requirements to become industry ready.

COURSES OFFERED

- DATA SCIENCE
- MACHINE LEARNING
- WEB DEVELOPMENT
- DIGITAL MARKETING

CHAPTER 2

INTRODUCTION

INTERNSHIP DEFINITION

An internship is a period of work experience offered by an organization for a limited period of time. Typically, an internship consists of an exchange of services for experience between the intern and the organization. Internships are used to determine if the intern still has an interest in that field after the real- life experience. In addition, an internship can be used to create a professional network that can assist with letters of recommendation or lead to future employment opportunities. The benefit of bringing an intern into full-time employment is that they are already familiar with the company, their position, and they typically need little to no training. Internships provide students the ability to participate in a field of their choice to receive hands-on learning about a particular future career, preparing them for full-time work following graduation.

INTERNSHIP BENEFITS

1. **Gain valuable work experience**

   The hands-on work experience interns receive is invaluable and cannot be obtained in a classroom setting, making this one of the most important benefits of internships**.** Interns have the opportunity to apply acquired knowledge to real work experiences, witnessing firsthand the day- to-day job duties they can expect to encounter in their chosen field. In addition to learning the specialized skills of a particular field, transferable skills such as communication, teamwork, and computer proficiency are also obtained in an internship, fully preparing interns to enter the workforce upon graduation.

2. **Explore a career path**

   Exploring is an important part of the college experience, and internships are a great way for students to acquaint themselves with the field they are interested in. Some students begin college with a major or career path in mind, and end up changing their minds later on.

Taking on an internship while in college allows students to work in their desired field, helping them decide if the field is right for them. By graduation, students who interned are more likely to feel confident they chose the right degree.

3. Give yourself an edge in the job market

One of the most important internship benefits is that college graduates who already have some work experience in the form of an internship stand out to potential employers. Internship experience makes a college grad more marketable as they usually require less training and can handle more responsibilities. You may also receive a higher starting salary than those who do not have internship experience and are entering the workforce or starting a new career.

4. Develop and refine skills

You can learn a lot about your strengths and weaknesses during an internship. Internships allow for feedback from supervisors and others who are established in the field, and offer a unique learning opportunity that you may not have again as a working adult. Embrace the mistakes you make as an intern and the many things that you won ‘t know. Ask questions, observe, and take risks to get the most out of your internship training experience.

5. Receive financial compensation

Many internships are paid which means you can gain valuable work experience and make money at the same time. A paid internship will provide money to fund your college tuition and expenses.

6. Network with professionals in the field

In the working world, it‘s all about who you know. As an intern, you will be surrounded by professionals in the industry. Internships are more than just about earning credit, getting a grade,or making money; internships provide an opportunity to learn from the people around you, ask questions and impress. The professionals you encounter during an internship can be your future colleagues or the connection to your first job.

7. **Gain confidence**

   Internships allow you to test out specific techniques learned in the classroom before entering the working world. It‘s an opportunity to apply what you have learned in a safe environment where mistakes are expected – rather than learn the hard way in your first job out of college.

8. **Transition into a job**

   Many companies use internships as a way to enhance their recruitment efforts. In some cases, a company may decide to hire an intern at the end of the assignment. Even if a job offer doesn‘t happen right away, an intern who makes a favorable impression could receive an offer.

INTRODUCTION TO PYTHON

Python is widely used general purpose ,high level programming language. It was initially designed by Guido van Rossum in 1991and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code. Python is a programming language that lets you work quickly and integrate systems more efficiently. Python was designed for readability, and has some similarities to the English language with influence from mathematics. Python uses new lines to complete a command as opposed to other programming language which often use semicolons or parentheses.

The most recent major version of Python is Python 3, which we shall be using in this tutorial. Python can be used on a server to create web applications. It can be used alongside software to create workflows. It can connect to database system and also read and modify files.

Python can be used to handle big data and perform complex mathematics. It can be used for rapid prototyping, or for production ready software development. It works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc). It runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.

Features of Python programming language

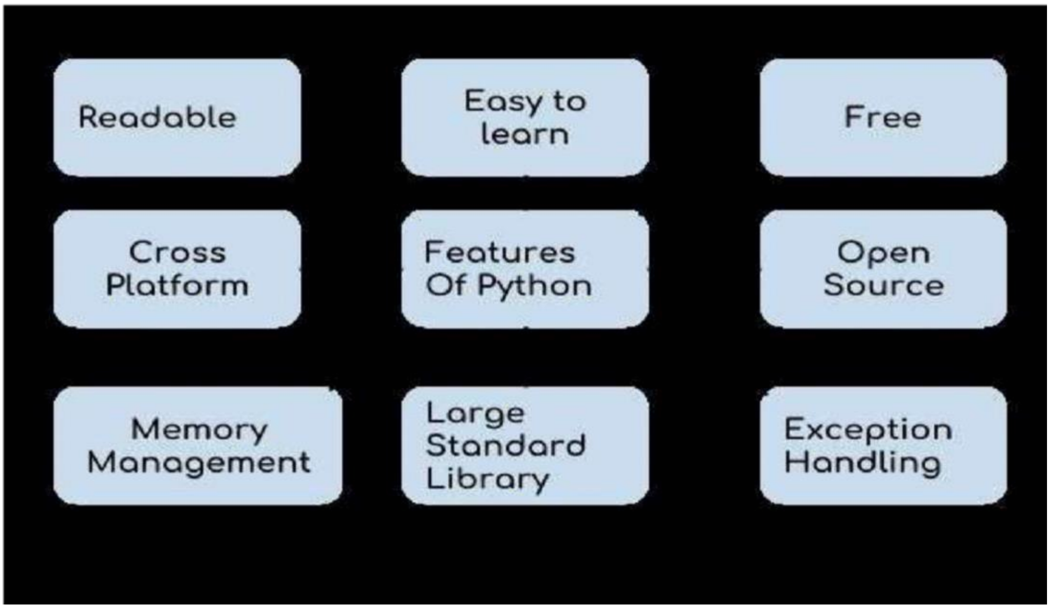


Figure : Features of Python 1.

1. **Readable:** Python is a very readable language.

2. **Easy to Learn:** Learning python is easy as this is a expressive and high level programming language which means it is easy to understand the language and thus easy to learn.

3. **Cross platform:** Python is available and can run on various operating systems such as Mac, Windows, Linux, Unix etc. This makes it a cross platform and portable language.

4. **Open Source:** Python is a open source programming language.

5. **Large standard library:** Python comes with a large standard library that has some handy codes and functions which we can use while writing code in Python.

6. **Free:** Python is free to download and use. This means you can download it for free and use it in your application. Python is an example of a FLOSS (Free/Fiber Open Source Software), which means you can freely distribute copies of this software, read its source code and modify it.

7. **Supports exception handling:** If you are new, you may wonder what is an exception? Exceptionis an event that can occur during program exception and can disrupt the normal flow of program. Python

supports exception handling which means we can write less error prone code and can test various scenarios that can cause an exception later on.

8. **Advanced features:** Supports generators and list comprehensions. We will cover these features later.
9. **Automatic memory management:** Python supports automatic memory management which means the memory is cleared and freed automatically. You do not have to bother clearing the memory.

Applications of Python

1. **Web development** – Web framework like Django and Flask are based on Python. They help you write server side code which helps you manage database, write backend programming logic, mapping urls etc.

2. **Machine learning** – There are many machine learning applications written in Python. Machine learning is a way to write a logic so that a machine can learn and solve a particular problem on its own. For example, products recommendation in websites like Amazon, Flipkart, eBay etc. is a machine learning algorithm that recognizes user‘s interest. Face recognition and Voice recognition in your phone is another example of machine learning.
3. **Data Analysis** – Data analysis and data visualization in form of charts can also be developed using Python
4. **Scripting** – Scripting is writing small programs to automate simple tasks such as sending automated response emails etc. Such type of applications can also be written in Python programming language.
5. **Game development** – You can develop games using Python.

6. **Embedded applications** – You can develop Embedded applications in Python.

7. **Desktop applications** – You can develop desktop application in Python using library like TKinter or QT.

MACHINE LEARNING

Machine learning (ML) is the study of computer algorithms that improve automatically through experience. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop conventional algorithms to perform the needed tasks.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

MACHINE LEARNING APPROACHES

Early classifications for machine learning approaches sometimes divided them into three broad categories, depending on the nature of the "signal" or "feedback" available to the learning system.

These were:

- **Supervised learning** : The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs.

- **Unsupervised learning** : No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning).

- **Reinforcement learning** : A computer program interacts with a dynamic environment in which it must perform a certain goal (such as driving a vehicle or playing a game against an opponent) As it navigates its problem space, the program is provided feedback that's analogous.

CHAPTER 3

REQUIREMENT SPECIFICATION

HARDWARE REQUIREMNETS

- 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.

SOFTWARE REQUIREMNETS

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

CHAPTER 4

DESIGN AND IMPLEMENTATION

MODULE DESCRIPTION

The generative model creates new instances of data that resemble 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 output image is fake or real.[8] The Generator and discriminator both are neural networks and both run in competition with each other in the training phase.

The final output is generated with the convolution block. Later the discriminator checks if the image generated from the generator 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 very efficient since it provides high- quality cartoonized images. To transform videos into animated or Cartoonized videos, we have additionally used the cv2(Computer vision application) which is a library in python the video is divided into frames depending on the specified time period fps=0.5s One can change the time period and No. of frames will differ. Later on, these frames will be brought together (joined) by os.join to convert into video files (.mp4 or .avi).

Conversion of Video is basically similar to getting an animated video out of the normal one. The video is first divided into frames and saved into an array, then passed through the generator and discriminator with the help of OpenCV to get the animated(cartoonized video).

IMPLEMENTATION

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.

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 images.

PROJECT DEVELOPMENT

REQUIREMENTS: Install PyCharm, which is one of many IDEs available for python. PyCharm is an integrated development used in computing programming, specifically for the python language. Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes. Project and code navigation specialized project views, file structure view and quick jumping between file, classes, methods and usages. It is cross platform provides code analysis, graphical debugger, an integrated unit tester integration with version control systems.

Step 1: Creating a python project

- Open PyCharm and select Create New Project.
- Select the location and a name for your project (medium tutorial in this case)
- Expand the Project Interpreter option and select Existing interpreter
- Locate your environment by clicking on the three dots on the extreme

Step 2: Typing the source code

- The source code of the desired with all required package is installed. Which involves various module

like numpy, pandas, matplotlib etc.

Step 3: Import the csv file

- 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 4: Running the Application

Step5:Run/Debug Configuration

CHAPTER 5

SOURCE CODE:

```
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'))


# confirm that image is chosen
if originalmage is None:

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


    ReSized1 = cv2.resize(originalmage, (960, 540))
    #plt.imshow(ReSized1, cmap='gray')



    #converting an image to grayscale
```

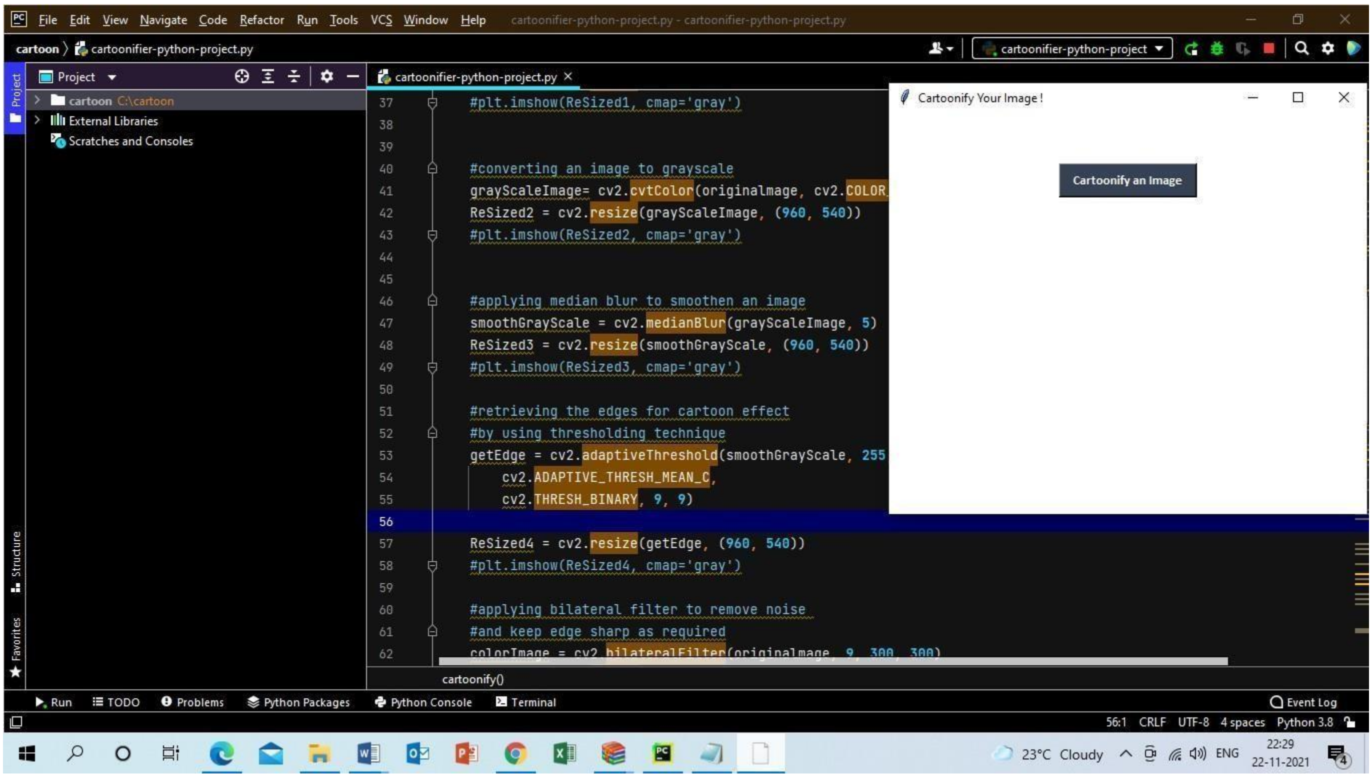
```
  grayScaleImage= cv2.cvtColor(originalmage, cv2.COLOR_BGR2GRAY)
  ReSized2 = cv2.resize(grayScaleImage, (960, 540))
#plt.imshow(ReSized2, cmap='gray')



 #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(originalmage, 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()
```

CHAPTER 6

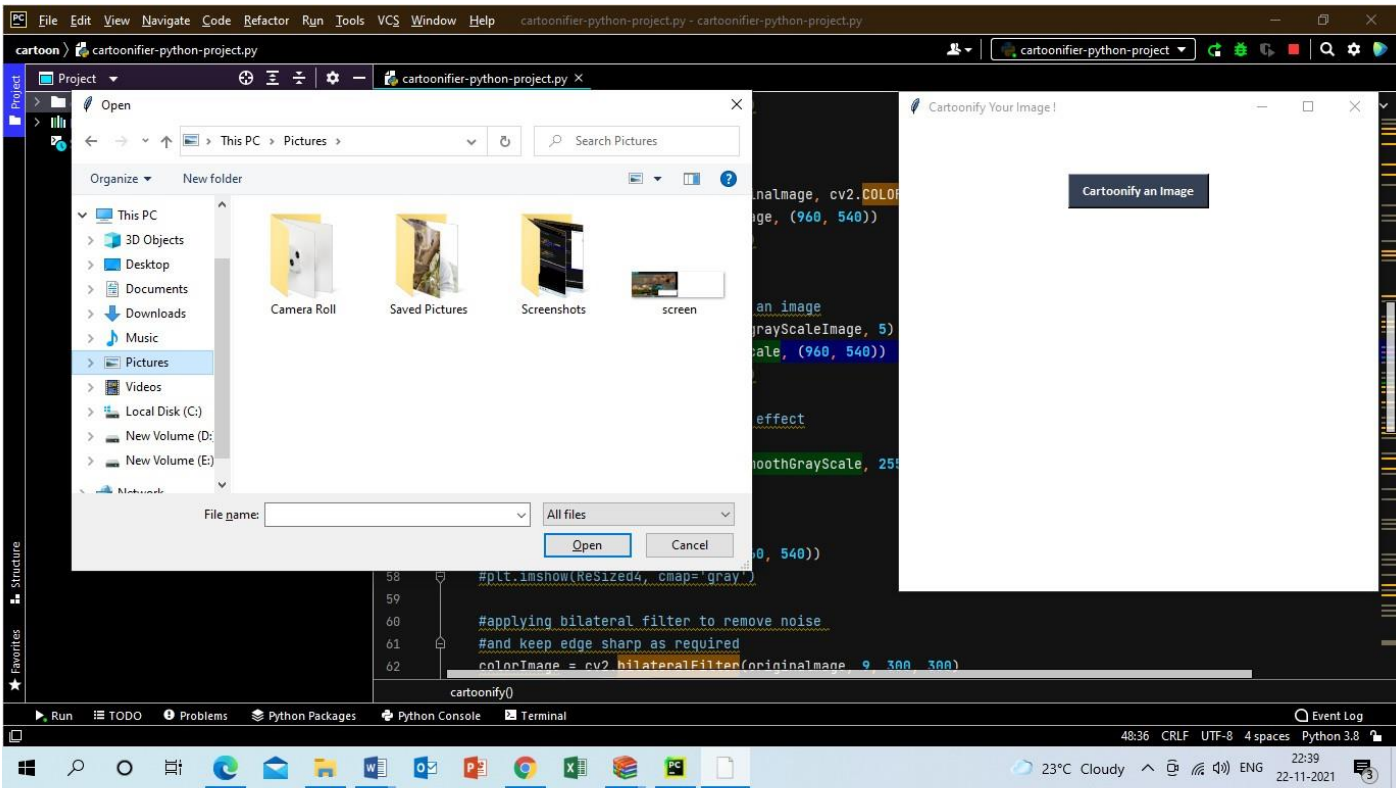
SNAPSHOTS

Home page



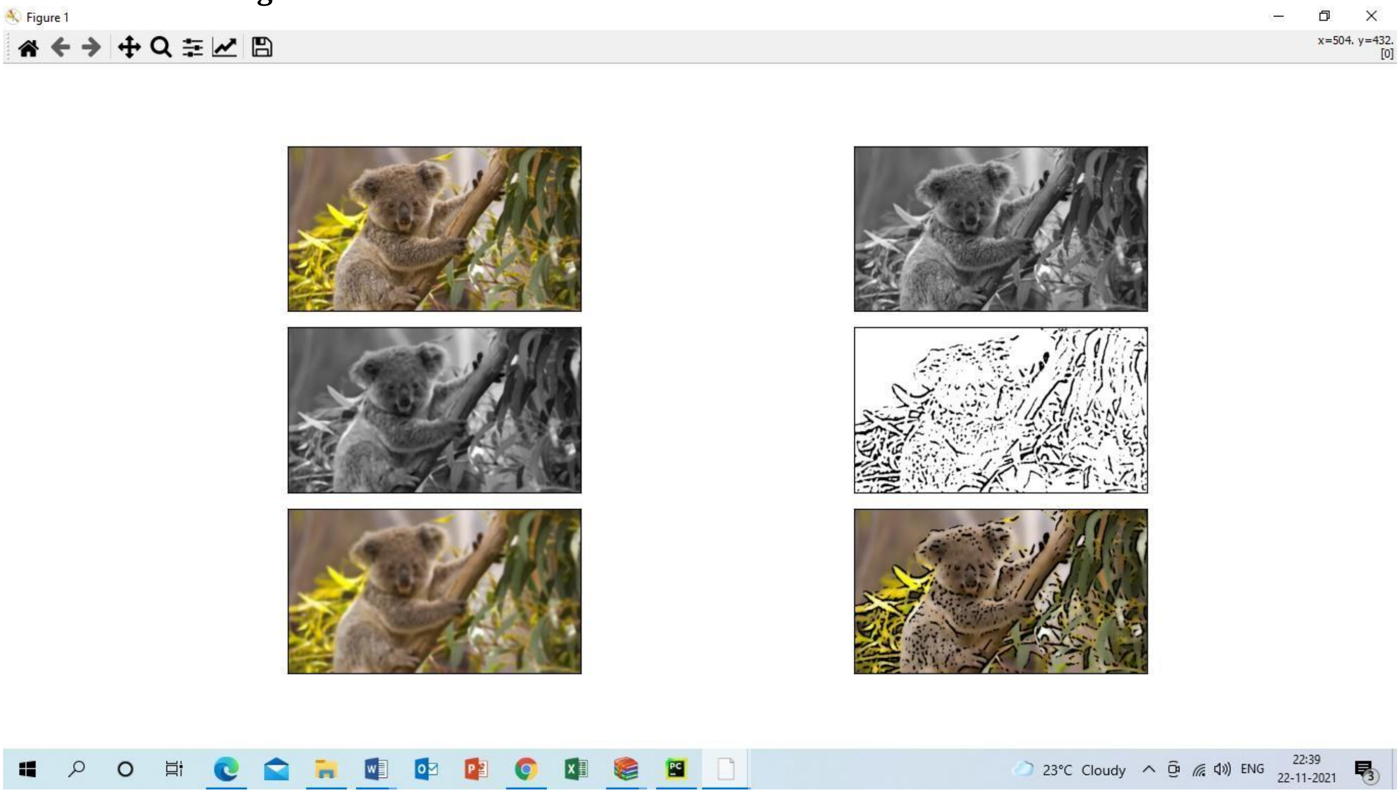
```
#plt.imshow(ReSized1, cmap='gray')


#converting an image to grayscale
grayScaleImage= cv2.cvtColor(originalmage, cv2.COLOR
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(originalmage, 9, 300, 300)
```

Selecting the image

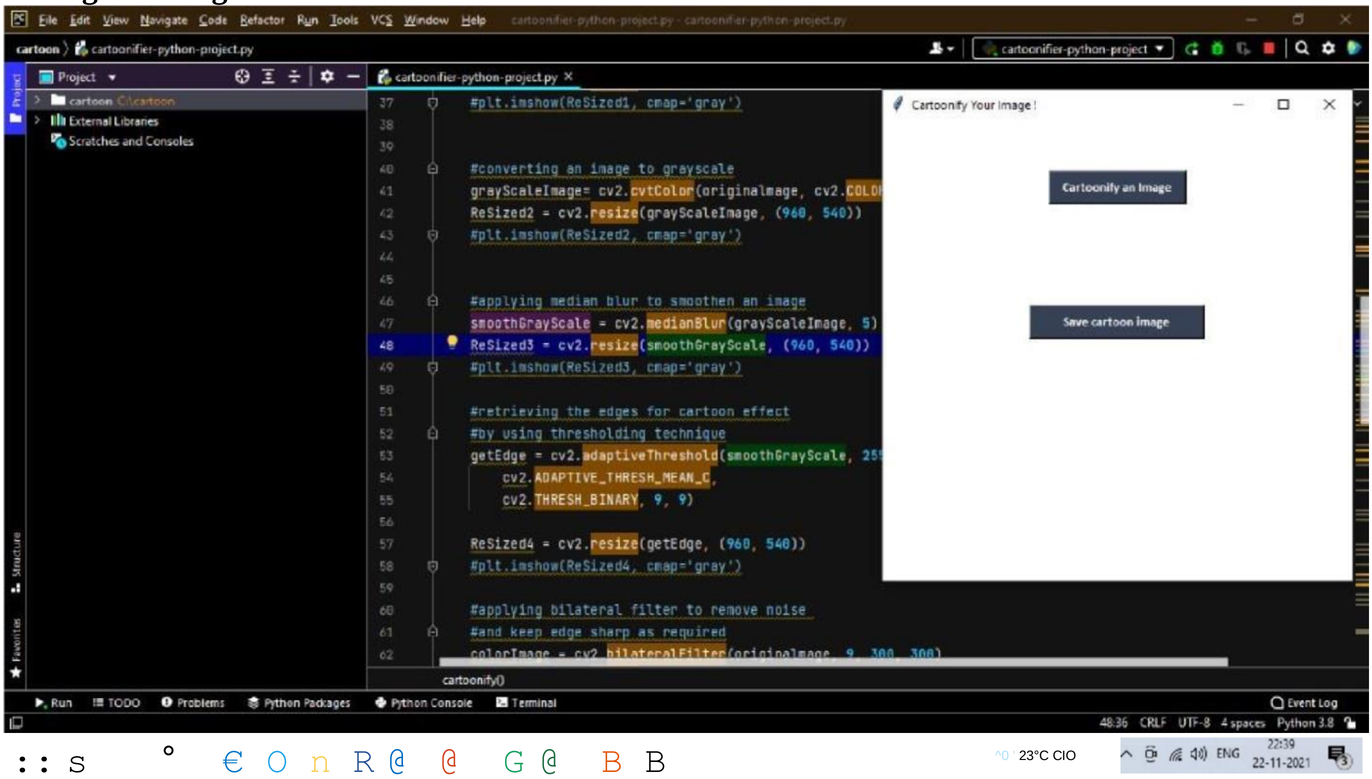


```
nalmage, cv2.COLOR
age, (960, 540))

an image
grayScaleImage, 5)
ale, (960, 540))

effect
oothGrayScale, 255

0, 540))
    #plt.imshow(ReSized4, cmap='gray')

    #applying bilateral filter to remove noise
    #and keep edge sharp as required
    colorImage = cv2.bilateralFilter(originalmage, 9, 300, 300)

cartoonify()
```

Cartoonified Image



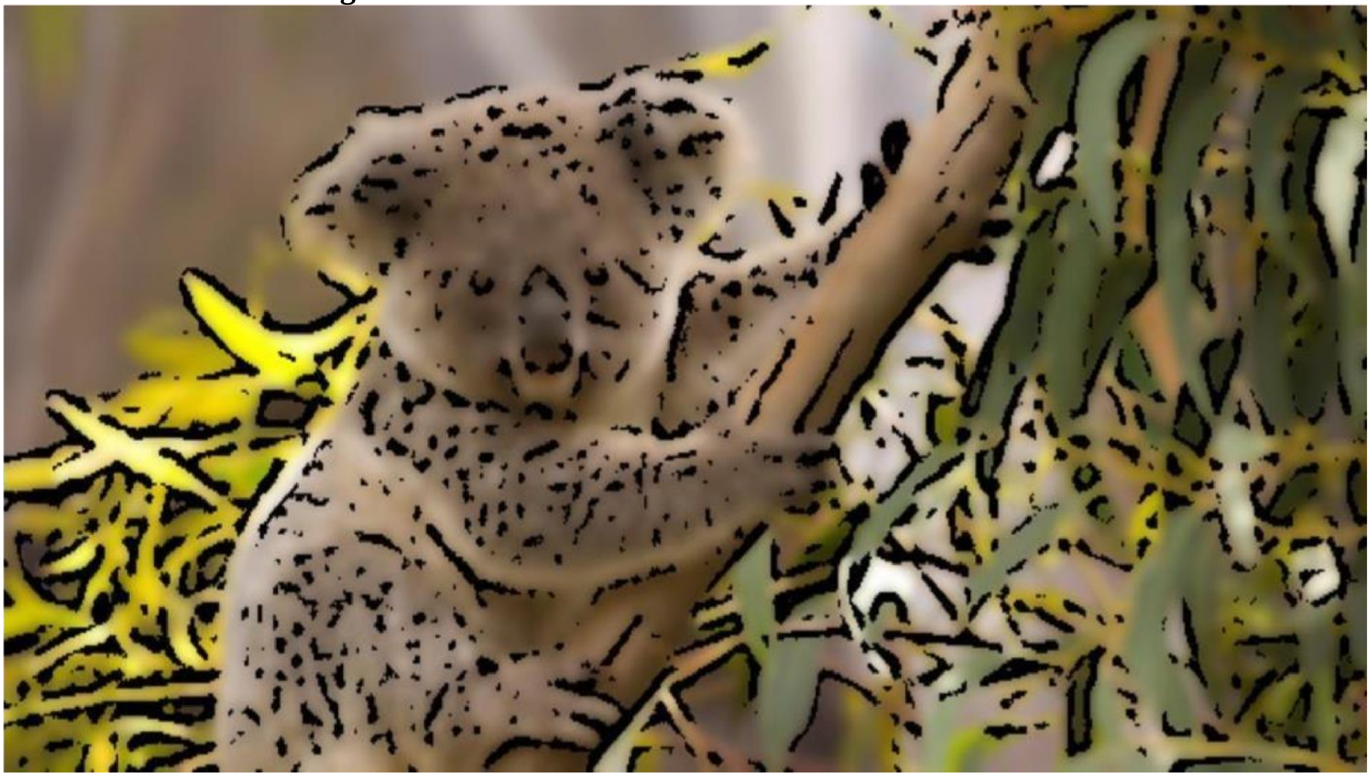
Saving the Image


File Edit View Navigate Code Refactor Run Tools VCS Window Help
cartoonifier-python-project.py - cartoonifier-python-project.py
cartoon 〉 cartoonifier-python-project.py
cartoonifier-python-project
Project
cartoon C:\cartoon
External Libraries
Scratches and Consoles
cartoonifier-python-project.py
Cartoonify Your Image !
Cartoonify an Image
Save cartoon image
cartoonify()
Run
TODO
Problems
Python Packages
Python Console
Terminal
Event Log
48:36 CRLF UTF-8 4 spaces Python 3.8
Structure
Favorites

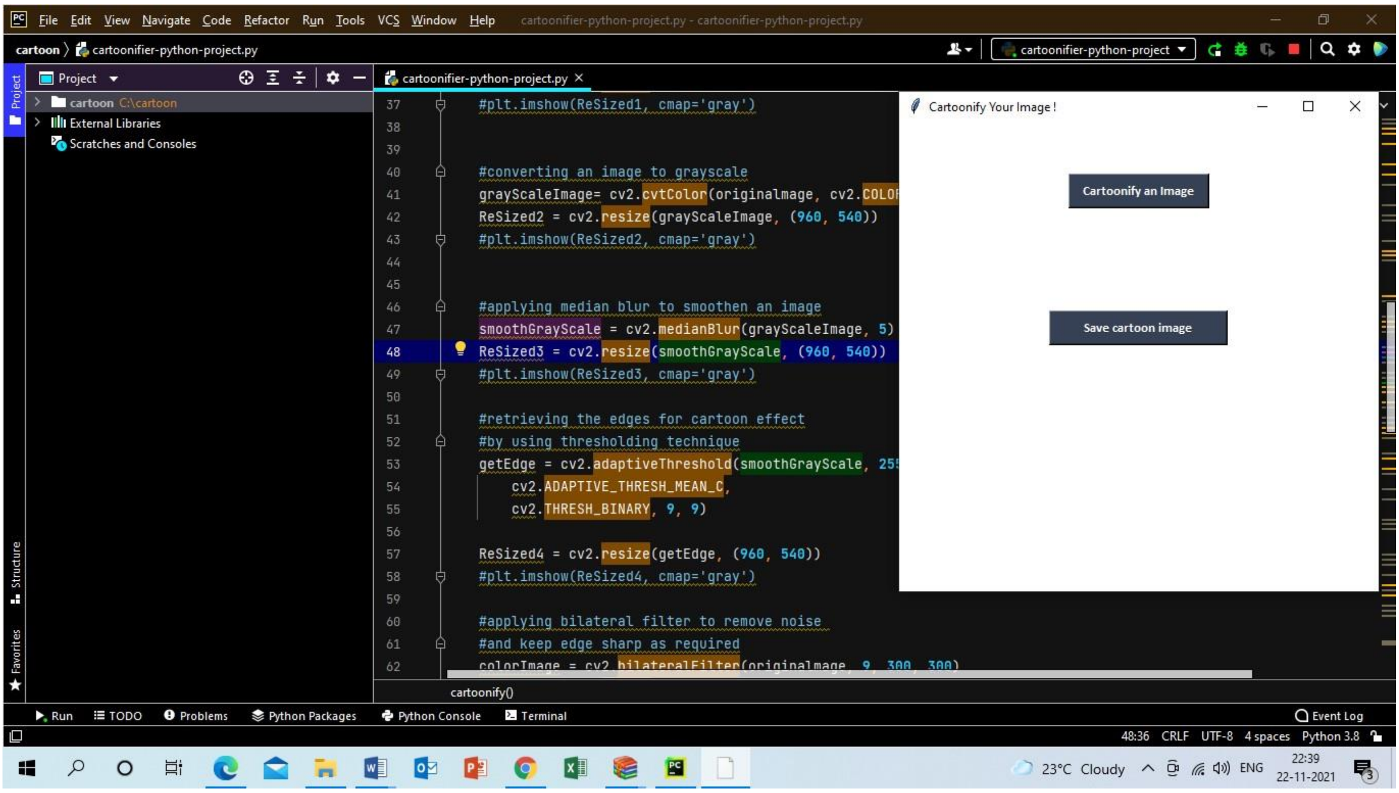
```
#plt.imshow(ReSized1, cmap='gray')


#converting an image to grayscale
grayScaleImage= cv2.cvtColor(originalmage, cv2.COLO
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, 25
    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(originalmage, 9, 300, 300)
```

: : s ° € O n R @ @ G @ B B ^0 23°C CIO ENG 22:39 22-11-2021

Saved Cartoonified Image



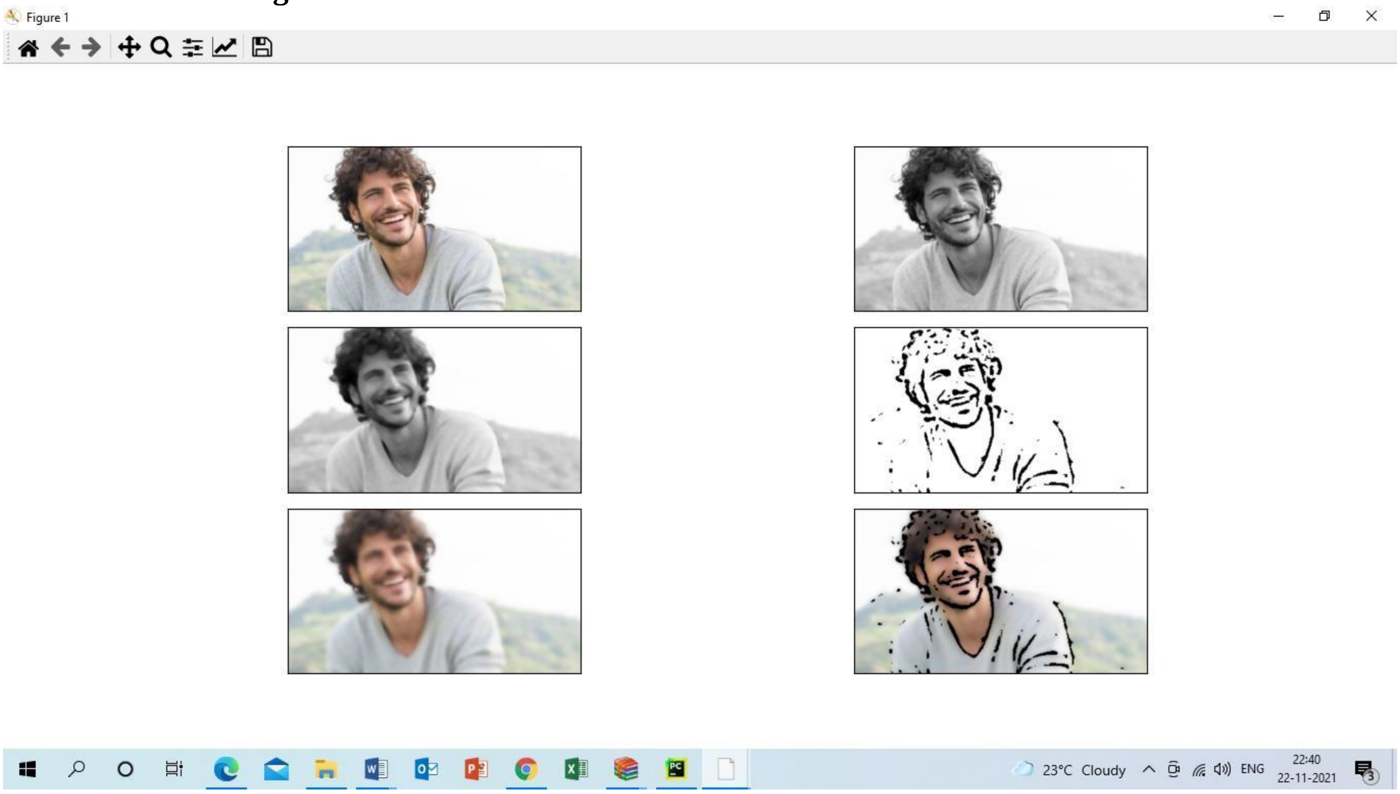
To select another Image



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


#converting an image to grayscale
grayScaleImage= cv2.cvtColor(originalmage, cv2.COLOR
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, 25
    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(originalmage, 9, 300, 300)
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help
cartoonifier-python-project.py - cartoonifier-python-project.py
cartoon 〉 cartoonifier-python-project.py
cartoonifier-python-project
Project
cartoon C:\cartoon
External Libraries
Scratches and Consoles
cartoonifier-python-project.py
Cartoonify Your Image !
Cartoonify an Image
Save cartoon image
cartoonify()
Run
TODO
Problems
Python Packages
Python Console
Terminal
Event Log
48:36 CRLF UTF-8 4 spaces Python 3.8
23°C Cloudy
ENG
22:39
22-11-2021

Cartoonified Image



CHAPTER 7

CONCLUSION

I have shown that how image can be converted to cartoon. I also stated the examples on how image is converted to cartoon. Hardware and software requirements of image to cartoon conversion are also shown . The systematic working of image to cartoon conversion and respective algorithm and formulae is shown with neat diagram in this project. Also I have stated challenges and problems one can face while cartoonifying the captured image.

CHAPTER 8

REFERENCES

[1]. A Neural Algorithm of Artistic Style, 2016 - Leon A. Gatys, Alexander S. Ecker, Matthias Bethge

[2]. Image Style Transfer Using Convolutional Neural Networks, 2016 - Leon A. Gatys, Alexander S. Ecker, Matthias Bethge

[3]. Perceptual Losses for Real-Time Style Transfer and Super-Resolution, 2016 - Justin Johnson, Alexandre Alahi, Li Fei-Fei

[4]. Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks, 2016 - Chuan Li, Michael Wand

[5]. Texture networks: Feed-forward synthesis of textures and stylized images, 2016 - D. Ulyanov, V. Lebedev, A. Vedaldi, V. Lempitsky

[6]. Demystifying Neural Style Transfer, 2017 - Yanghao Li, Naiyan Wang, Jiaying Liu, Xiaodi Hou

[7]. A Learned Representation For Artistic Style, 2017 - Vincent Dumoulin, Jonathon Shlens, Manjunath Kudlur

[8]. Deep Photo Style Transfer, 2017 - Fujun Luan, Sylvain Paris, Eli Shechtman, Kavita Bala

[9]. Neural Style Transfer: A Review, 2018 - Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, Mingli Song

[10]. StyleBank: An Explicit Representation for Neural Image Style Transfer, 2017 - Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, Gang Hua

[11]. Conditional Fast Style Transfer Network, 2017 - Keiji Yanai, Ryosuke Tanno

[12]. Characterizing and Improving Stability in Neural Style Transfer, 2017 - Agrim Gupta, Justin Johnson, Alexandre Alahi, Li Fei-Fei