TOSHKENT SHAHRIDAGI INHA UNIVERSITETI
INHA UNIVERSITY IN TASHKENT

School of Computers and Information Engineering

Spring semester 2020

Multimedia Computing

Technical Report / **Assignment for Mid-term and Final Exam**

Project name: PhotoSuperhero

Name: Mokhlaroyim Tuychiboeva

ID: U1610148

Group: 16-3

Name: Mokhlaroyim Tuychiboeva

ID: U1610148

Group: 16-3

Abstract

Being one of the most popular high-level programming language, Python has become a practical tool in creation of not only complex websites but also various desktop applications. The paper reviews how with the help of desktop camera the person' face can be detected, cut out and replaced with another image. The program applies OpenCV - python library, which works with the Image Processing techniques related to Computer Vision. This report will explain the process and the implementation of the algorithm method and finalizing with experiments results that are achieved in the end of the program.

Keywords: Python, desktop application, OpenCV, Image Processing, face recognition

Introduction

Many people around the world are becoming workaholic nowadays. It appears that very little time is left for spare time. Entertainment is one of the method to make people feel more relaxed. Since it is not denied that many people, interested in "Marvel Comics" regardless their age and gender, would not be against to view themselves in superhero costumes. This has served as a major motive for the creation of the current application, which is called *"PhotoSuperhero"*.

PhotoSuperhero is a desktop application, the primary purpose of which is to entertain a user. A lot of playful and smart applications have come out in recent days. But a uniqueness of this app accounts for usability of the program even among those category of people, who are not familiar with proper use of application. The program requires no extra learned skills and it is not rocket science to open it and to get the desired output. In fact, a user is provided with soft and simple user interface, for the first time, and he/she is supposed to click on Image and press on several hot keys, so that the program itself automatically generates the result without human interaction. It should be noted that the given project is for a while available for desktop version, but in the future the mobile version is also considered.

As for aim of the program, it should detect the face of the person from the image and crop out that region in order to place in another image. In particular, the captured Image of person is processed using OpenCV python library which is applied for all sorts of image and video analysis, such as facial recognition and detection. There are also other python packages are used for creation of graphical user interface (GUI). The result are shown in Experiment Result section. In the end of the report, some limitations of the project are discussed, and some suggestions are made for future work.

Name: Mokhlaroyim Tuychiboeva

ID: U1610148

Group: 16-3

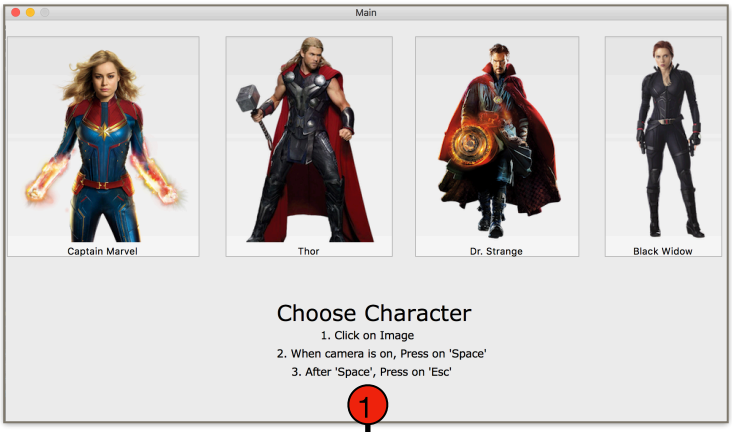
Method

There are two python files, namely "*menu.py"* and "*python.py".*

The first "*menu.py"* file is responsible for GUI of the application. The main logic is the creation of a Class, such as *"Example"* and creating the instance of it in main(). In details, in order to have a friendly user interface, three python libraries has been imported: tkinter - is a package for Python, widgets are supposed to be inherited; tkinter.ttk - is a module which gives access to the Tk themed widget set, and Frame, Label, Style, Button are supposed to be inherited; PILL - is for working with images, and Image, ImageTk are supposed to be inherited. The class *"Example"* inherits from tkinter, the container Frame, so that it serves for object organization and widgets. Here, one class constructer and two functions *"initUI()"* and *"open_dialog()"* are created. In *"initUI()" ,* the image files are opened with PhotoImage() and .pack() geometry manager packs widgets in rows or columns. Once Image has been pressed, the tkinter.ttk widget Button() calls command lambda:self.open_dialog(), where open_dialog() function refers to a python,py function called capture_img() or in other words the other window (Camera Window) pops up.

The second "*python.py"* identifies the face of a person and place it in the selected body of the background image. In order to do this, OpenCV python library is imported to work with image and video processing. Here, only one function *"capture_img()"* encompasses all desired logic. In while loop, the image of the person is captured frame by frame and displayed, and two hot keys work: "Esc" and "Space". The user is supposed to press first on "Space" to take a photo of him/herself and then "Esc" to quit and see the output image. For face detection OpenCV provides such called CascadeClassifier() which takes the parameter, haarcascade_frontalface_default.xml file. After this, using cvtColor(), we convert to gray desired image and gray background image and with for loop, coordinates of the face are identified, and cropped out so that to put in background image.

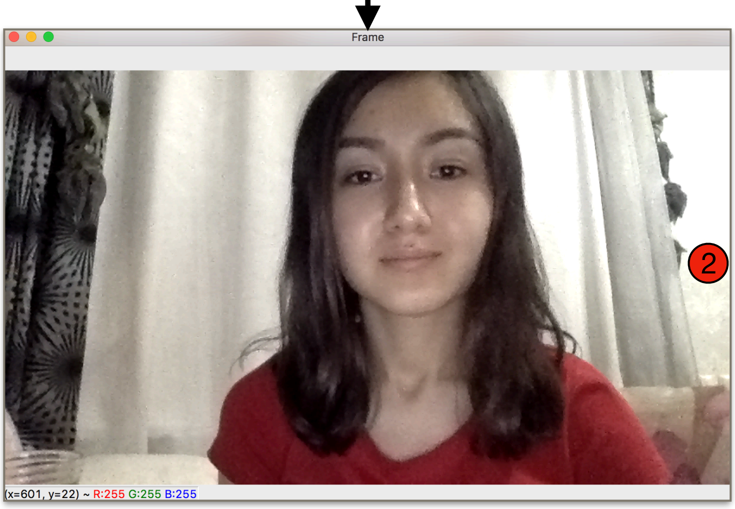
Experiment Results



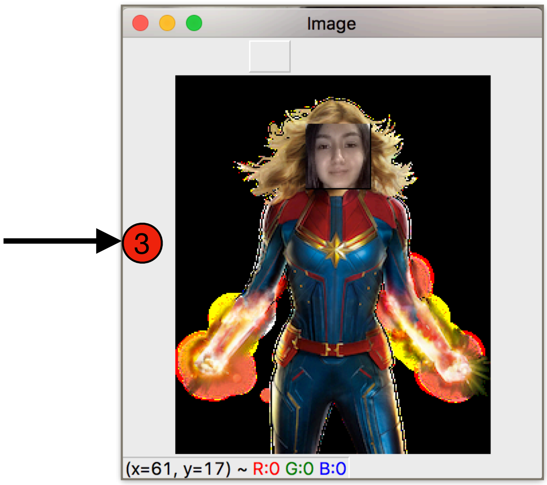
Choose Character

1. Click on Image
2. When camera is on, Press on 'Space'
3. After 'Space', Press on 'Esc'

Figure 1. *Main Window*



(x=601, y=22) ~ R:255 G:255 B:255



(x=61, y=17) ~ R:0 G:0 B:0

Figure 2. *Frame Window*

Figure 3. *Image Window*

Experiment results illustrate the window where four images, or in other words, one of four main characters to be chosen. Once the "*menu.py"* file has been run, the Main window pops up as shown in *Figure 1*. The next step is to click on desired Image. This click is followed by a camera popup, demonstrated in *Figure 2*. The user can press on "Space" more than once, so that to recapture and process the last Image. If to press "Esc" the output result of the Image comes out, which is illustrated in *Figure 3*. As it is seen, there some disadvantages of the program. The face is identified, however, in ideal case the contour of the face must be cropped out and some filters to be applied in order the picture to look more realistic.

References

https://younglinux.info/tkinter/tkinter.php
https://docs.python.org/3/library/tkinter.ttk.html
https://pillow.readthedocs.io/en/stable/handbook/overview.html
https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html