Computer Vision Assignment 0

Swetanjal Dutta
International Institute of Information Technology, Hyderabad
Gachibowli, Telangana
swetanjal.dutta@research.iiit.ac.in

Abstract

This is a report for Assignment 0 of the Computer Vision Course taught by Dr. Avinash Sharma during Spring Semester of 2020 at IIIT-H. The goal of this assignment is to get familiar with the usage of OpenCV package, the most popular Computer Vision library currently. The assignment also introduces us to a popular Computer Vision task called Chroma Keying.

1. Introduction

The objective of this assignment was to get myself familiar with the usage of OpenCV package in Python, and implement Chroma Keying using Python and OpenCV. After completing the assignment, I feel I have acquired the following skills:

- Using the OpenCV package to do basic operations like reading an image, writing an image, displaying an image, creating frames from a video and creating a video from multiple frames.
- Use the inbuilt web camera to capture images and videos.
- Perform basic image processing operations like thresholding an image, finding edges, difference between images etc.
- Convert images from one type to another. Eg: BGR to RGB; BGR to Grayscale; BGR to binary etc.
- Implement vectorised operations on images
- Implement simple Computer Vision Algorithms like Chroma Keying from scratch.
- Implement complex Computer Vision Algorithms like Face Detection using inbuilt functionalities.

2. Chroma Keying

2.1. Problem Statement

This is the technique of composting(layering) two images or videos based on color hues(chroma range). It is useful when one wants to place an actor in another background/scene. The idea of Chroma Keying is that there are two videos. One video is the foreground video, which is the video of the actor acting in front of a green screen. The other video is the background video which is the video of a scene of interest where one wants to put the actor. Figure 1. and Figure 2. shows an example frame from foreground and background videos respectively.

Figure 1. Example of a frame from the foreground video

Figure 2. Example of a frame from the background video

Our goal is to compost the reporter (in front of the green screen) in the News Channel background. This is achieved using the Chroma Keying algorithm. We achieve results as shown in Figure 3.

Figure 3. Example of a frame from the composted video

2.2. Algorithm

The following is the algorithm of Chroma Keying:

1. Extract frames from both the foreground as well as the background videos.
2. Take a foreground frame and it's corresponding background frame.
3. Find the most frequent color in the foreground frame using histogram.
4. Go through each intensity value in the foreground image and check if it is very close to the most frequent color. This can be done by taking a norm of $abs(img[i][j] - most\ frequent\ color)$ and checking whether it is below a certain threshold ϵ which is empirically decided.
5. If the intensity value at a particular location in the foreground image is close to the most frequent color, we replace the intensity with the intensity value at the corresponding location in the corresponding background frame.
6. If the intensity value at a particular location in the foreground image is not close to the most frequent color, we keep the intensity of the pixel as it is.
7. Repeat for all locations in the frame.
8. Save the frame as a new image.
9. Repeat the steps 2, 3, 4, 5, 6, 7, and 8 for all frames.
10. Finally, we have the output images which can be made into a video.

2.3. Challenges faced

1. The two videos, background video and foreground video have to be of same length. Therefore, had to use some video editing tool like ffmpeg.
2. The resolution of the two videos, the background video and the foreground video have to be the same. Therefore, had to again use a tool like ffmpeg.
3. Even with careful decision of empirical threshold, one can either see the foreground actor being cut in some places or a green color appearing along the borders of the foreground actor.
4. The green reflection cannot be removed from specular surfaces.

2.4. Sample Results

Download the output_data folder from here and place it in the current directory i.e the directory of this report. Download the input_data folder from here and place it in the current directory i.e directory of this folder.

- Inside output_data folder, there exists a folder by the name 'frames'. This contains output of the frame extraction process from a video. Un-comment appropriate lines in cell 6 of Jupyter Notebook to execute this task. Call the function as follows:

```
videoToFrame(path to video,
path to frame folder)
```

  Inside the input_data folder there exists a directory by the name 'videos'. This contains the input video from which the frames are extracted.

- Inside output_data folder, there exists a folder by the name 'videos'. This contains output of the video creation task from frames. Un-comment appropriate lines in cell 6 of Jupyter Notebook to execute this task. Call the function as follows:

```
framesToVideo(path to frame
folder, path to video,
frame rate)
```

  Inside the input_data folder there exists a directory by the name 'frames'. This contains the input frames that are converted to the video.

- Inside output_data folder, there exists a folder by the name 'capture'. This contains output of the frame capture from my web camera. Un-comment appropriate lines in cell 6 of Jupyter Notebook to execute this task. Call the function as follows:

```
captureVideo(path to frames)
```

- Inside output_data folder, there exists a folder by the name 'chroma_key'. This contains outputs of the Chroma Keying algorithm. Un-comment appropriate lines in cell 10 of Jupyter Notebook to execute this task. Call the function as follows:

```
ChromaKeyVideo(path to
foreground video, path to
background video, path to
output video, frame rate)
```

  Inside the input_data folder, there exists videos fg.mp4, bg.mp4, fg1.mp4, bg1.mp4. These are used as background/foreground inputs for Chroma Keying Composition.

- Inside output_data folder, there exists a folder by the name 'face_detection'. This contains outputs of the Face Detection Algorithm from the web camera. Uncomment appropriate lines in cell 12 of Jupyter Notebook to execute this task. Call the function as follows:

```
face_detection_capture(path to
video to be created, frame rate)
```