CIS 5810 Extra Project

Spring 2023

1 Track 1: Face swapping

1.1 Overview

The aim of this project is to automatically detect and swap faces between videos. Given two videos, you will automatically swap faces between the two videos. How you formulate this pipeline is completely up to you. However, we do want to see at least one set of videos where the face from one video has been replaced onto the face in another video without the emotion of the target face coming through.

1.2 Implementation

1.2.1 Broad goals

Seamlessly swapping faces is a non-trivial process. To swap faces between two videos, you will need to complete the following tasks. Firstly, you will need to detect faces and/or facial landmarks in the two videos. Once you detect these features in the videos, you will have to estimate the transformation from one face to another. We suggest exploring concepts such as affine transforms or homography, or triangulation or thin-plate splines. Once, you have the transformation between the two faces, you may swap the two faces. You may also experiment with various optical flow methods such as Kanade-Lucas-Tomasi, or Meanshift or Camshift if you have to track faces in the videos. You are not limited to these and are highly encouraged to play around with other approaches that you may find. You will have to compensate for the changes in exposure, lighting and shadows, etc. between the two videos while swapping the faces.

1.2.2 Suggested pipeline

We will provide you with a dataset of videos containing face(s), to be called source faces/videos. The videos that you select to replace face(s), will be referred to as replacement faces/videos. You are required to swap faces between the two videos. However, while you replace the faces, keep in mind that the

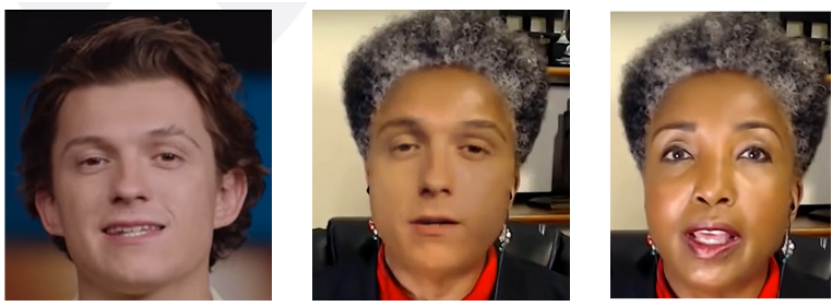


Figure 1: Left to right: source image, target image, face of subject in target image swapped with face in source image

Figure 2: Left to right: two images with the faces of the subjects swapped with each other - note that basic face swapping effect is achieved, although there is a lot that can be improved (e.g. smoothing face color, face direction alignment etc.)

source face's emotion should be present on the target body. In other words, the source face should not emote the target face. This is the only mandatory requirement. Once you are done with this, you are allowed to experiment and showcase your creativity! You are highly encouraged to use your own videos.

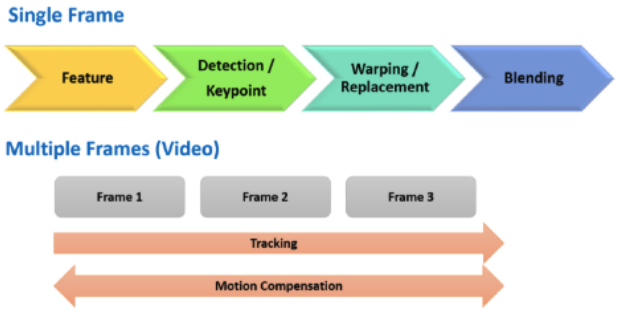


Figure 3: Possible pipeline for face swap

A suggested pipeline is as follows:

- Replacement video/s selection
  Select a second video with a face or faces to swap with.
- Source and replacement face and facial landmarks detection
  Detect faces and facial landmarks in the source and replacement videos. We recommend the Dlib toolkit or any deep learning based facial landmark detection.
- Feature extraction
  Since the emotion of the replacement face should not seep through to the source face, the features you use to control the warp should just be the ones along the convex hull of the face. However, to control the warp better, you could use other features within the face as long as they don't change the emotions of the source face.
- Face swapping
  For each frame, compute appropriate image transforms that warps the replacement face to the source face and vice-versa. Apply these transforms to both the faces. You may have to compute the convex hull of the source face and the replacement face while swapping the faces.
- Video refinement (optional)
  Make the face swapping seem natural. Use gradient domain blending or any other technique that makes the swapped faces look real with their new bodies. Compensate for exposure, lighting and shadows, poses, skin tone, etc. Incorporate optical flow techniques to robustly track faces.

- Video to video replacement
  Keep in mind that this is a video to video face replacement. The two videos may have the same number of frames or they may not. You should account for this while performing the replacement.

- Eternal glory (extra-credit)
  You will be awarded if you implement cool and creative features apart from the ones mentioned above. You can also attempt face swapping within a single video. Having your code run close to real time will be another way to earn a lot of extra credit!

2 Track 2: Style transfer

2.1 Overview

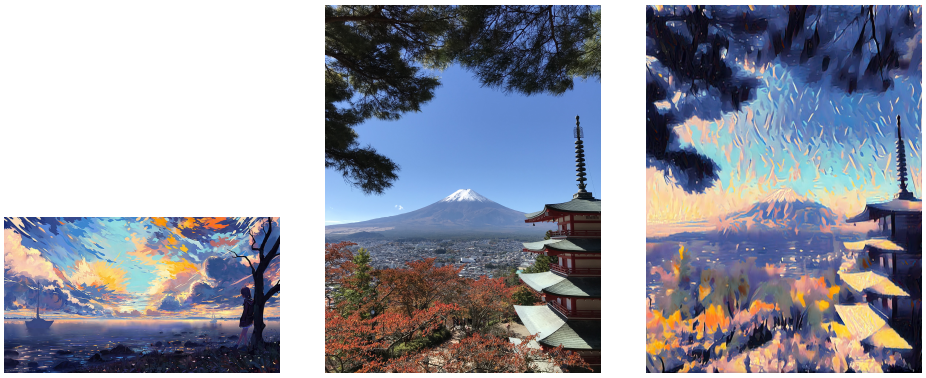
Neural style transfer (NST) represent a category of algorithms designed to take an input image and reproduce it with a specified artistic style. The Neural-Style algorithm [1] was the first such algorithm which provided a simple an efficient way to generate stylized images. Through this track you will develop a modified version of NST which starts off with being able to generate artistic style images before moving on to tackle generating stylized videos. You may focus on the implementations provided in [1] but you are encouraged to explore more recent works and incorporate anything interesting from those into your projects to generate photo-real images. A few images showing what will be achieved through this project are shown in Fig. 4.

2.2 Implementation

2.2.1 Broad goals

The first goal of your implementation should be to develop a code-base capable of taking a reference style image and an input image that is to be stylized - and then output a synthesized stylized image. The second goal will be to extend your model to take in a reference style image and an input video - and output a coherent synthesized stylized video. In designing and implementing your pipeline, we recommend you to experiment with different network architectures, loss functions (content and style), and optimizers. It is important to report results for your experiments and explain qualitatively as well as quantitatively as to why you made a certain design choice. You are expected to compile all of your findings in your final submission (which includes reporting all of your design choice, describing your system/metrics, and tabulating the comparisons). Remember to start small, this track can be memory and compute intensive therefore start by stylizing smaller images, before moving on to synthesizing larger images.

Style transfer in videos is very similar to images except that you need to add



(a) Style Image (b) Input Image (c) Stylized Input

(d) Style Image (e) Input Image (f) Stylized Input

Figure 4: Example of Style Transfer. Notice that the Style Image does not even need to be of the same size!

another component to the loss function such as temporal consistency loss, to ensure that the generated scene does not change a lot across consecutive frames. When tackling video synthesis, a key element is to ensure consistency between frames. A suggested reference paper is [2]. You can implement the loss given in this paper. You can also think about concepts taught in the course such as optical flow or feature tracking across frames to ensure a consistent stylized video. After you have a working pipeline, you can also think about how you can reduce the memory consumption, and speed up your code to scale beyond the smaller images while also being faster.

3 Track 3: No Cameraman Left Behind

3.1 Overview

This track is based on a project created by former students of CIS 5810 (Jeffrey Li, Sindhura Mente, Amar Mohantry, Sriya Reddi) which was named No Cameraman Left Behind. In their project Jeffrey, Sindhura, Amar, and Sriya addressed an issue many of us have faced where in taking group pictures, the photographer has to be left out. Their objective was: given a source image of only the photographer and a target image of the rest of the group - to implement a program (using image processing and deep learning techniques) that combines

both parties into a realistic final image. Now you will be able to work on solving this problem as well using various methods and making sure no one in the group is left out of the photo!

Figure 5: Left to right: source image, target image, final output image

Figure 6: Left to right: source image, target image, final output image

3.2 Implementation

3.2.1 Broad goals

For this track, you are able to start with a source image of only the photographer and a target image of the rest of the group. The first goal is to extract the relevant information from your source images. The second goal is to output a natural, realistic, and visually pleasing final image with everyone together. The third goal is to analyze the outcomes and challenges that will come into play when your initial images are taken (such as positions/distances/angles of the subjects, camera angles, lighting, shadows, backgrounds, etc.). You will need to create a system/metric to compare qualitatively and/or quantitatively how your good your output is (versus a benchmark across various situations). For this track, we suggest you start your experimentation with images that minimize the variable challenges (as described above) before incorporating them in.

4 Track 4: Celebrity Recognition

4.1 Overview

This track is based on a project created by former students of CIS 5810 (Ryan French, Sukanya Joshi, Luisa Silva). Ryan, Sukanya, and Luisa are big fans of actor Brad Pitt's movies, such as Ocean's Eleven and Mr. & Mrs. Smith. In their project, they built and trained a deep learning model to detect and recognize Brad Pitt in video streams. Do you have a favorite celebrity? If yes, you can implement your own methodology to detect and recognize your target subject(s) both in images and videos.

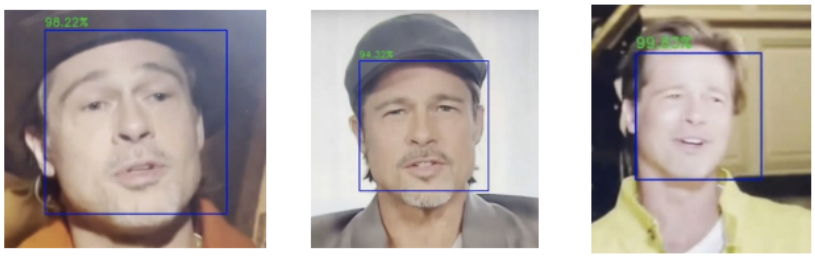


Figure 7: Detecting and recognizing Brad Pitt's face - displaying percentages of accuracy

Figure 8: Detecting and recognizing Brad Pitt's face - displaying percentages of accuracy

4.2 Implementation

4.2.1 Broad goals

The first goal of this track is to research and build a deep learning model using PyTorch that will recognize and track your celebrity's face in videos. Feel free to experiment with transfer learning, where you can use a pre-trained neural network model and modify it to suit your project's needs. For detecting human faces (before conducting facial recognition on the detected faces), you do not have to create your own model and can use existing facial detection models from OpenCV (or other frameworks/libraries). The second goal is to compile and build a relevant and impactful dataset that you will use to train your deep learning model. Building datasets can be very time consuming and you will need to think of an efficient way to compile this for training. When compiling your dataset be aware of such factors as image sizes, face angles, and quantity of data needed. The third goal is to analyze how successful your outcomes are under various conditions and discuss the areas for further improvement. You will need to create a system/metric to compare qualitatively and/or quantitatively how your good your output is (versus a benchmark across various situations). Your implementation must work on video streams. Can you detect and recognize more than one celebrity in a video?

5 Track 5: 360 image creation

* This track does not require you to own a 360 VR headset.

5.1 Overview

The aim of this project is to obtain a 360 image which can be viewed in a virtual reality (VR) headset and simulated much like the 360 images available over Google Maps (or any other known platform). You will need to use multiple images from the same location, at various orientations, and display them on a spherical mesh grid, in a way that resembles the natural view of our surroundings as we perceive them.

5.2 Implementation

5.2.1 Broad goals

The first goal in this track is to use stitching on multiple images within a spherical meshgrid. It is crucial to decide how you will fold a stitched 2D image into a 360 projection. To help you display your images on spherical grids, you will need to look at various projection methods. The second goal is for you to design your results in a way that can be used on any 360 viewing application, and provide the user an immersive experience of that location. You can use a 360 image viewing application to display your final results, however the challenge

Figure 9: An example of the final 360 image

is for you to display the final output in the correct form. You can use various feature extracting techniques between two or more images, as well as any of the methods taught in the course. We suggest experimenting with various tools, and also analyzing their implementation in Google Street View. You are allowed (encouraged) to use a VR viewing application (such as 360 Panorama Viewer) to see the final results. You do not need a VR headset to work on the project, as the 360 image viewing application can use your device's motion to display the panoramic view in 360.

6 Track 6: Open-ended

6.1 Overview

If you have an idea or are interested in some problem not covered by the other projects, you can explore them in this track. There are no restrictions on what you can explore (as long as it is relevant to this course). As this is an open-ended option, the TAs will keep track of your progress to provide support and guidance. It is entirely possible that what you thought would work, won't, and that is an entirely expected part of research. You should expect that your methodology will change between the initial proposal and the final project. If you still do not have satisfactory results for the final report, as long as you sufficiently showcase your experimentation and demonstrate a well-thought approach to the problem, you will not be strongly penalized for this!

Some possible directions:

- Novel research in an area related to the course material. Students pursuing individual research projects are encouraged to relate the course project to

their outside research, but the course project should be something new for the course.

- Taking some of the course algorithms and applying them to a novel or interesting problem of your choice.
- Study and implement an existing idea that was either discussed in class or some other state-of-the-art method.

References

[1] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. "Image Style Transfer Using Convolutional Neural Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[2] Manuel. Ruder, Alexey. Dosovitskiy, and Thomas Brox. "Artistic Style Transfer For Videos and Spherical Images". In: *International Journal of Computer Vision (IJCV)*. Aug. 2018.