CS4475 Project 1

Stuthi Bhat (sbhat307@gatech.edu)
Victoria Choi (vchoi3@gatech.edu)
Steven Li (sli862@gatech.edu)
Raymond Liu (rliu443@gatech.edu)

High Level Overview

Our project implements different types of slide transitions in a few different functions. Given two input images of the same size, a start image and an end image, our project contains multiple functions that each generate a different transition between the two, including a sliding transition, a fade transition, a circle transition, a flash transition, and a split transition. The transition is generated as a set of images in a specified directory, which can be combined in a video editor to create a cohesive video. The transitions are verbally explained below:

`slideTransition()`
Beginning with the start image, the `slideTransition()` function moves the start image to the left, gradually cropping more and more off the left edge. At the same time, the left edge of the end image slides in from the right, following the left image as it continues sliding left. At the end, the start image will have completely slid off the screen, leaving only the end image.

`fadeTransition()`
Beginning with the start image, the `fadeTransition()` function gradually decreases the brightness of the image until it becomes black. Then, it gradually increases the brightness of the end image until it reaches full brightness.

`circleTransition()`
Beginning with the start image, the `circleTransition()` function cuts a small circle in the center of the image through which the end image is peaking through. The size of the circle increases until it covers the entire screen, at which point the end image will be showing.

`flashTransition()`
Beginning with the start image, the `flashTransition()` function gradually increases the brightness of the image until it becomes white. Then, it gradually decreases the brightness of the end image until it reaches normal brightness.

`splitTransition()`
Beginning with the start image, the `splitTransition()` function splits the start image down the middle and slides the left half leftward the the right half rightward off the display. The middle reveals the end image. By the end, both halves will have slid off the image, leaving just the end image.

Shown on the next page are input images and a link to the final video produced by these functions in the same order listed above (normally, you only need two input images, but we are using 6 for the sake of demonstration and so we can demonstrate all 5 transitions in one video).

We took the following images during our recent trip to Switzerland. We hope you enjoy the very short summary of our trip!

Input Images

Video Link:

https://drive.google.com/file/d/153pcWLN__aYkkRUtaCrKRAFRRtR_vUD_/view?usp=sharing

Low Level Description

This section outlines the low level workflow procedure of our project. It will be delineated in steps.

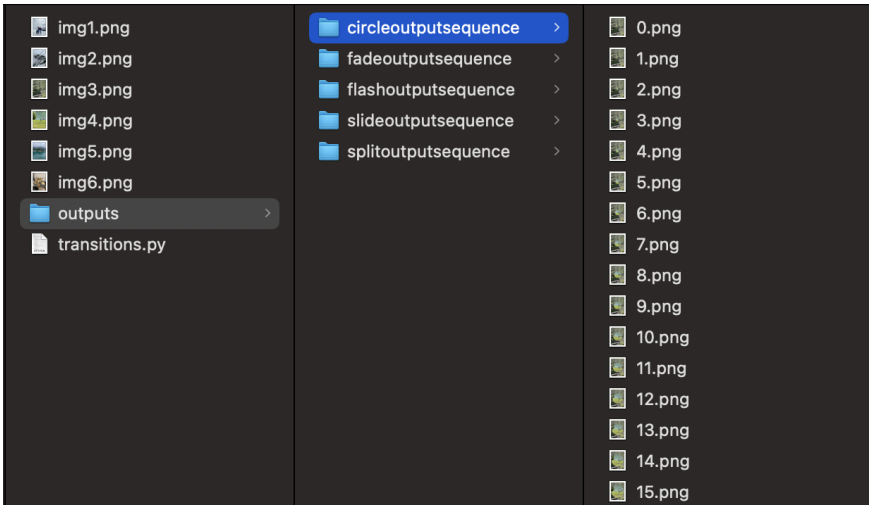
Inputs: Two images of the same size, a start and and end image (we assume both input images are of the same size)

1. Create and navigate to a directory called `<yourDirectory>` (or whatever name you please) somewhere in your files where you would like your transition sequences to be saved. This directory will hold other directories containing transition sequences. Record the path name to this directory, ensuring that it ends in a slash.
    a. **Example**: `/Users/StevenLi/Desktop/CS4475/Project1/outputs/`
2. Add two images called `<start.ext>` and `<end.ext>` into `<yourDirectory>`. **Ensure they have the same size.** If they do not, crop them before continuing the following steps. Also, add the `transition.py` file to the directory.
3. Within `<yourDirectory>`, create another directory called `<yourTransition>` (or whatever name you please). Record the name of this directory, ensuring that it ends in a slash. This will be the directory in which the transition sequence is saved.
    a. **Example**: `slideoutputsequence/`
4. Open the `transitions.py` script. At the top of the file, input the path names of `<yourDirectory>` and `<yourTransition>` you recorded earlier into the indicated variables. At the bottom of the file, input the names of the image files into the indicated variables. At the very bottom, uncomment the function corresponding to the transition you would like to perform. Edit the `frameCount` parameter depending on how many total images you would like in your final sequence.
5. In your command line, run they command `python transitions.py`
6. Check `<yourTransition>`. It should contain new images.
7. Open a video editing application of your choice. Import all images in the sequence as static images in order. Decrease the time frame of each image to a fraction of a second (no longer than 0.1 seconds per image). Set the transition between images to none.
8. Export the video from the image editor. You will be left with a video of the transition.
9. Optionally, if the video editor displays each image for too long, you can speed the video up after exporting.

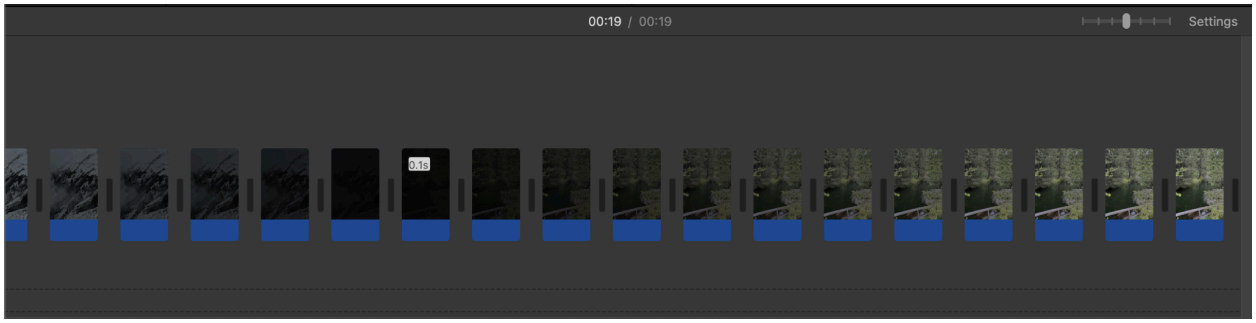
Outputs: A video of the desired transition from the start image to the end image

This video from the high level overview was created through repeated application of the low level procedure we describe above. Here are images showing the process:

Directory Organization



Video Editor (Fade Transition)



Referenced Materials

These were the resources we referenced for help. It includes help with cv2 functions, including image file writing, bit masking, and circle drawing, as well as some HTML help. All other code was written by our team members.

Image File Writing:
https://stackoverflow.com/questions/41586429/opencv-saving-images-to-a-particular-folder-of-choice

Bit Masking:
https://docs.opencv.org/4.x/d0/d86/tutorial_py_image_arithmetics.html

Circle Drawing:
https://www.geeksforgeeks.org/python-opencv-cv2-circle-method/

HTML Images, Video, Formatting:
https://www.w3schools.com/html/html_images.asp
https://www.w3schools.com/html/html5_video.asp
https://www.w3schools.com/html/html_entities.asp

Point Breakdown

We will accept a 30/30 breakdown.