CS335 Exercise 10
"JMorph" (completed)
Due Sunday December 12

1. Introduction

The goal of this project is to provide a user interface and backend driver that supports the specification and rendering of a piecewise (triangular) image morph between two images. Once specified, the morph will be rendered as a sequence of images that can be wrapped as a video (using **`ffmpeg`** to create an mp4) and used in standard video editing software. Your work in this phase will tweak the frontend and *complete the backend of the framework* by applying the specified transitions to the underlying images and by adding additional features.

2. Features

Now that the user can position a grid of control points on the start and end images, you must render the morph that warps and cross-dissolves between the two images. This means that the user must be able to read in the desired images (you must use a menu and a file browser widget). The computed warp should follow the positions of the two sets of triangles. The cross-dissolve should linearly scale the intensities between start and end images over the interpolation interval. Render tween images as JPEGs.

The following additional features are **required** in your completed project:

- Handle Start/End Images of Different Sizes: Scale the images on input so that the start/end image sizes are the same. It is not sufficient simply to disallow different sizes.
- Start/End Image Enhancement: implement the ability to apply a simple intensity adjustment to the start and end images. In particular, you must allow the user to adjust/change the *brightness* of the images. You may design how the interface will allow these changes to be specified. The morph should use the current image intensity values (after any image processing adjustments have been applied).
- Grid Resolution: Implement the ability to change the grid resolution. Provide the ability to choose a square resolution setting from (5x5) to (20x20), i.e., (5x5), (6x6), … (20x20)

The following features are optional and will receive **extra credit** if you implement them correctly:

- Drag Constraints on Control Points: implement a *constraint* so that no control point can be moved to create overlapping triangles.
- Control Point Group Move: implement the ability to select a group of control points (within a bounding box) and drag/move their position as a group.
- Project Save/Read: ability to save a project (all parameters/state: control point positions, parameters, image files) and read it in order to continue work over multiple sessions.

3. What to Turn In

You must submit a complete Java solution as an IntelliJ project. In addition, you must submit a *complete movie file* created by your tool that demonstrates a morph (or set of morphs edited together). Create the movie by converting the JPEG frames from your tool into a single movie file with ffmpeg[1]. Make sure the output video uses the mp4 codec.

[1] https://ffmpeg.org/about.html