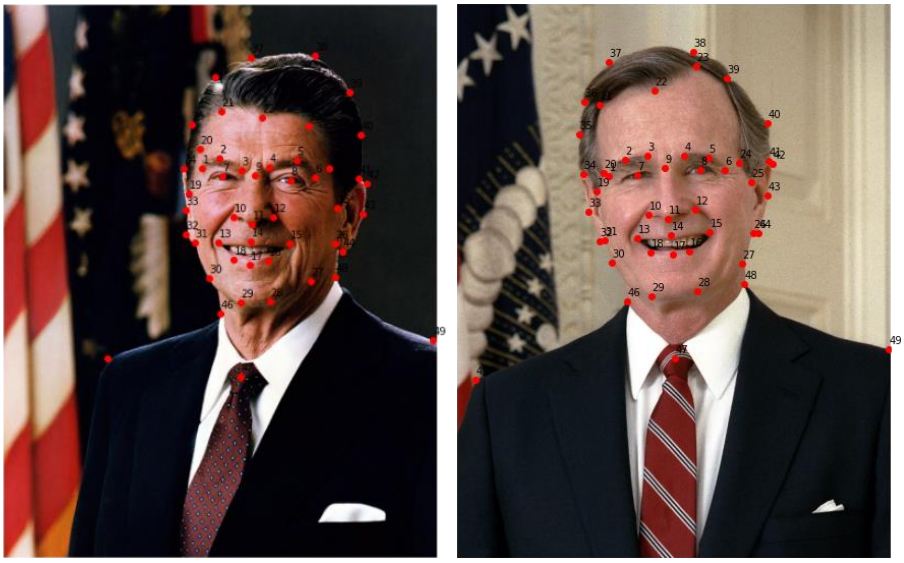
Motivation and impact:
The goal of this project was to implement image morphing. Image morphing is a technique used primarily for special effects in film to create a transition between images and/or objects within images. This requires correspondence points to be defined between objects. For this project, correspondence points were defined manually, but one could use facial recognition tools to automatically produce correspondence points.

Approach:
The following are the steps taken to morph between two images:

1) Define correspondence points on each image.
2) For each frame, linearly interpolate between correspondence points. Keep count of each frame and use the current frame number as the weighting variable for linear interpolation. This effectively produces coordinates that define a shape that is the weighted average of the two images in the morph.
3) Use Delaunay triangulation to create a mesh over correspondence points in the average shape. Use the same mesh for each image.
4) For each triangle in the mesh, compute the affine projection from the triangle in the interpolated shape to the corresponding triangle in each image.
5) For each pixel in each triangle, compute the weighted average of intensities between corresponding pixels in each of the images by mapping from the average image to the original images.
6) Save each frame
7) Combine frames to create a video of the morph.

Results:
To demonstrate image warping, a video was produced which morphs between portraits of the last six United States presidents. The video is found at the following link: https://youtu.be/v1wv3MEpp7M. The video is running at 30 frames per second, and was made using 60 frames per morph.



57 correspondence points were used for each image. 49 of the points were chosen using the convention shown in the images above. Additionally, one point was place in each corner and one point at the midpoint of each edge of the images.

Implementation detail:
The project was implemented in a Jupyter notebook using Python. OpenCV packages were used for reading and writing images to and from files, combining the image into a video, and for help with the affine warping of corresponding triangles. SciPy packages were used for the Delaunay triangulation and computing a least squares solution of the affine transformation matrix.

A function was created that allows the user to select control points from within Jupyter notebook. After selecting all control points for an image, the coordinates of the control points are saved as a text document to the folder containing the images.

Another function was created which takes two images, two sets of control points, and a number in range [0,1]. This function first performs a linear interpolation between each pair of corresponding points in the two images, with the weighted average distance between each point defined by the number passed to the function. Delaunay triangulation is then used to create a triangle mesh of the averaged control points, with each control point being a vertex. The function then loops through each triangle in the mesh, and affine transformation matrices are computed that map the triangles in each image to the corresponding triangle in the average mesh. The transformation matrix is computed using another function, compute_affine(), which is passed the coordinates of the correspondence points in two triangles as arguments, and outputs the transformation matrix.

The next part is done two different ways: a slow way and a fast way. The fast way uses the transformation matrix and cv2.affineWarp function to warp corresponding triangles into the shape of the average triangle and cross-dissolves using the same weighting number that was used to create the average shape. A mask is then used to extract the triangle and place it into an output image. Once this is done for all triangles in the mesh, the output image is saved to a file. The slow implementation does not use cv2.affineWarp function, but instead uses the inverse of the transformation matrix to map from pixel locations in the average triangle to pixel locations in the two morphing images. Then the weighted average of the two pixels is used as the pixel value in the average image. This slow implementation takes much longer to run than the faster method.

A final function loops through all images in a file folder. The folder also contains the associated list of correspondence point coordinates. For each pair of images, the function described above is run n times, where n is the desired number of frames in each morph. The current frame number is used as the weighting variable when interpolating between correspondence points. After all frames have been produced, they are combined into a video using cv2.VideoWriter().

Challenge / innovation
The biggest challenges was in creating the compute_affine() and morph() functions. After realizing that the use of cv2.warpAffine executed much faster than my initial method (the slow implementation described above), I decided to make a second implementation using cv2.warpAffine, which increased the project scope significantly. Additionally, a lot of time was spent experimenting with the correct number and location of correspondence points to produce a realistic morph.

Acknowledgements / Attributions:
Official Portrait of President Ronal Reagan by the White House Photographic Office,
https://en.wikipedia.org/wiki/File:Official_Portrait_of_President_Reagan_1981.jpg

Official Portrait of President George H. W. Bush by the White House Photographic Office, https://en.wikipedia.org/wiki/George_H._W._Bush#/media/File:George_H._W._Bush_presidential_portrait_(cropped).jpg

Official Portrait of President Bill Clinton by Bob McNeely, https://en.wikipedia.org/wiki/Bill_Clinton#/media/File:Bill_Clinton.jpg
Official Portrait of President George W. Bush by Eric Draper, https://en.wikipedia.org/wiki/File:George-W-Bush.jpeg#/media/File:George-W-Bush_edit.jpg

Official portrait of President Barack Obama in the Oval Office, Dec. 6, 2012. (Official White House Photo by Pete Souza), https://www.flickr.com/photos/obamawhitehouse/8390033709/

Official Portrait of President Donald Trump by Shealah Craighead, https://en.wikipedia.org/wiki/Donald_Trump#/media/File:Donald_Trump_official_portrait.jpg