Project 3: Image Morphing

By Zach Howell, zgh23

I completed image morphing and ran my program (p3.py) on three sets of images – the small cat/panda, the large cat/panda, and Ian/Pavan (two other CS students). I did not attempt any extra credit.

# Source Images

Small Cat/Panda (200x200)



Large Cat/Panda (450x450)

Same as above, not pasting in as would have to scale down

Ian/Pavan (500x500)



(Not to scale)

# Results

# Intermediate Frames

Here are some intermediate frames from small cat/panda. The full sequence is in a the attached folder framesCatPanda\_sm

|  |  |  |  |
| --- | --- | --- | --- |
| Frame 2/8 | Frame 4/8 | Frame 5/8 | Frame 7/8 |
| C:\Users\Zach\AppData\Local\Microsoft\Windows\INetCache\Content.Word\2.png | C:\Users\Zach\AppData\Local\Microsoft\Windows\INetCache\Content.Word\4.png | C:\Users\Zach\AppData\Local\Microsoft\Windows\INetCache\Content.Word\5.png | C:\Users\Zach\AppData\Local\Microsoft\Windows\INetCache\Content.Word\7.png |

Intermediate frames from Ian/Pavan

|  |  |  |  |
| --- | --- | --- | --- |
| Frame 2/8 | Frame 4/8 | Frame 5/8 | Frame 7/8 |
| C:\Users\Zach\AppData\Local\Microsoft\Windows\INetCache\Content.Word\2.png | C:\Users\Zach\AppData\Local\Microsoft\Windows\INetCache\Content.Word\4.png | C:\Users\Zach\AppData\Local\Microsoft\Windows\INetCache\Content.Word\5.png | C:\Users\Zach\AppData\Local\Microsoft\Windows\INetCache\Content.Word\7.png |

# Videos

For cat/panda, see attached video CatPanda.mp4

For Ian/Pavan, I made a 90 frame video. It’s also attached is called IanPavan.mp4

# Plots

The first thing I did was find the Delaunay triangulation for the cat, the panda, and some point in between. These are the plots I got:

|  |  |  |
| --- | --- | --- |
| Cat Triangulation | Intermediate Triangulation (at 0.5) | Panda Triangulation |
| C:\Users\Zach\AppData\Local\Microsoft\Windows\INetCache\Content.Word\start_cp_sm.png |  | C:\Users\Zach\AppData\Local\Microsoft\Windows\INetCache\Content.Word\end_cp_sm.png |

Both the cat and panda triangulation have fairly similar shapes, so I’m glad these images were chosen. The intermediate triangulation also visually looks like a transition between the other two, which is good. This worked pretty quickly, and I’m confident it’s working correctly.

# Process and Errors

I got a number of errors before getting some satisfactory results. I tried to implement all of the operations as numpy operations on the whole image (avoiding for loops), and mostly succeeded. However, one thing I couldn’t figure out with numpy was bilinear interpolation. I got some results that looked like this:



Where my destination points were not being filled in correctly (I was just leaving blank 0s at spots that didn’t have a corresponding source point). I fixed that by just switching to a for loop iterating over every pixel in the blank destination image and adding together/rounding all 4 surrounding pixels in the source.

I also got an error where my triangles were not mapping correctly. This resulted in a blended image looking like this:



And I fixed it by changing by one variable in one line of code:

simplices = sourceTri.points[sourceTri.simplices[simplexIs]]

#TO

simplices = sourceTri.points[destTri.simplices[simplexIs]]

Turns out the points in each Delaunay triangulation remain in the same order, but the order of the triangles is likely to change for some intermediate frames. By getting the point indices from the destination triangulation (in this case, where I’m finding what simplex each point is in) and then using those point indices in the source triangulation, I was able to get the triangulations to line up properly.