Kevin Lucas 149188

Hw1-Interpolation

nearest_neighbor

Function was rather easy to implement, it was just a proportion problem and rounding the floats to a whole number.

This function was very fast, compared to the bilinear_interpolation

bilinear_interpolation

At first i did not realize that there were other functions in interpolation.py that can be called to help with the calculations.I had already done all calculations needed within the resize.py and was unable to correctly import the functions in the interpolation.py file, so i left it as is.

I do believe that the functions were rather unneeded as they just calculated single point. (write a whole function to reduce 3 lines of code).

One initial problem i had was when dealing with any edge pixel, you cant pick a correct nearest neighbor & the distance would come out to 0 thus you cant divide(function to calculate intensity). To fix this there is an if else statement to check if its a edge piece and set it to the correct value.

I used the Documentation found on https://en.wikipedia.org/wiki/Bilinear_interpolation to set up my variables and do needed calculations

Overall

Overall once I found a good naming scheme for all the point coordinates, the problem was very simple, just plug it into an equation.

I do feel that the Directions in the README for resize/interpolation.py could have be written better(what do we do with them?) If they were to be called by the bilinear_interpolation, it should have stated that and be listed above the resize/resample.py instructions.

It would also be nice to have some simple test cases with a way to check if our output is correct. Otherwise you're just going "this looks right to me".