Nick Thompson

Dr. Morago

CSC-340-001

19 September 2019

Programming Assignment 0 - Rotation and Error Evaluation

- Summary of the assignment goals:

  My function “rotate_image” accepts the “rotation_angle” parameter. It then calls my “multiply_matrices” function with the rotation matrix and matrix of the x and y coordinate of a point. My “multiply_matrices” function can accept any size matrices but will return a “Value Error” if multiplication between the two matrices is not possible. I also scale the image up on a canvas that is 150% larger than the largest dimension of the original image, so I am able to rotate the image without clipping the corners of the original image. I calculate the absolute color error between the colors of the original image’s pixels and the colors of the image after it has been rotated 360º. Every time the image is rotated, I calculate the pixel displacement for that rotation and add it to the total pixel displacement of the image.

- 3+ images showing your code at code at intermediate stages (image rotated x degrees) with captions saying what angle stepsize is being used and the total rotation of the image so far (i.e. Image rotated 60º with steps of 20º, 3 rotation operations applied so far.):

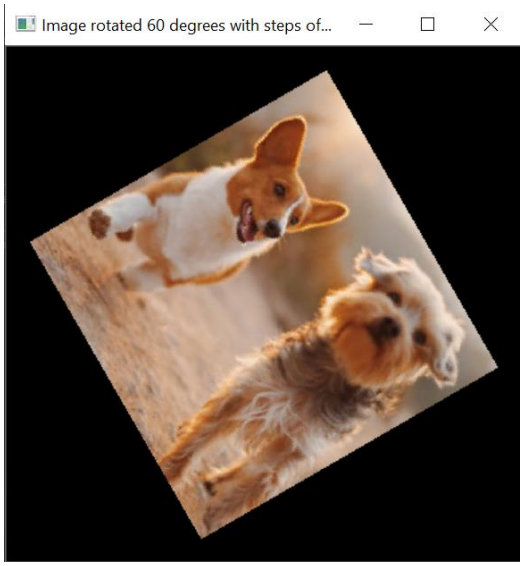

Image rotated 60º with steps of 60º, 1 rotation applied so far.

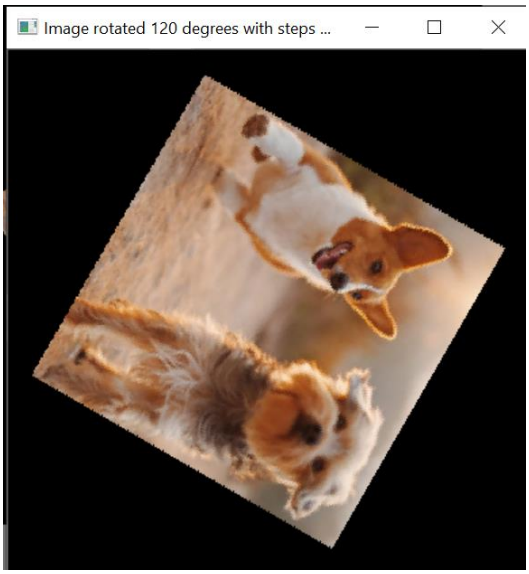

Image rotated 120º with steps of 60º, 2 rotations applied so far.

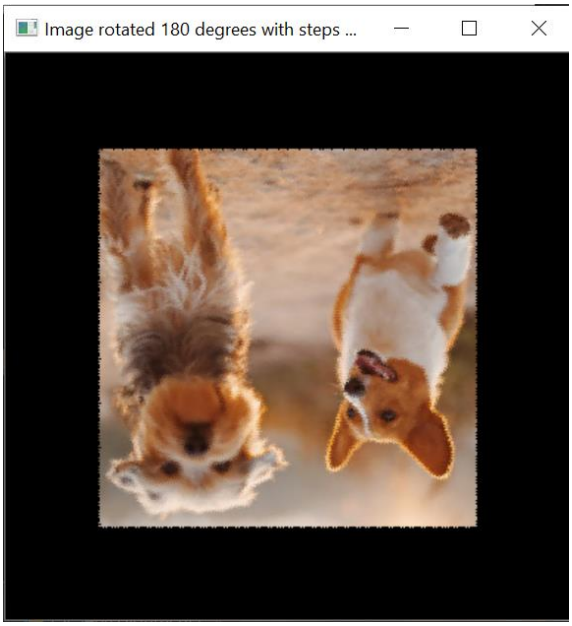

Image rotated 180º with steps of 60º, 3 rotations applied so far.

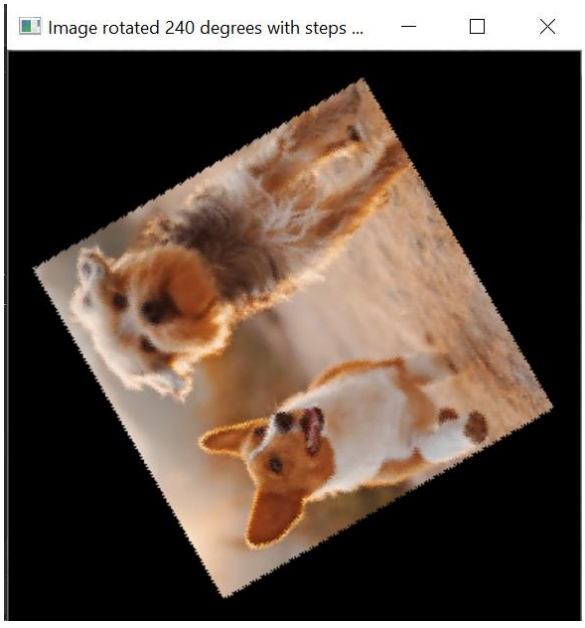

Image rotated 240º with steps of 60º, 4 rotations applied so far.

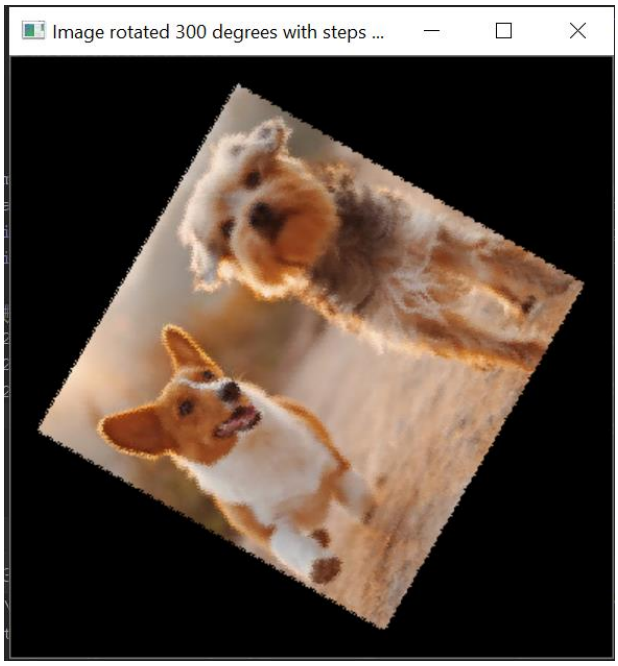

Image rotated 300º with steps of 60º, 5 rotations applied so far.

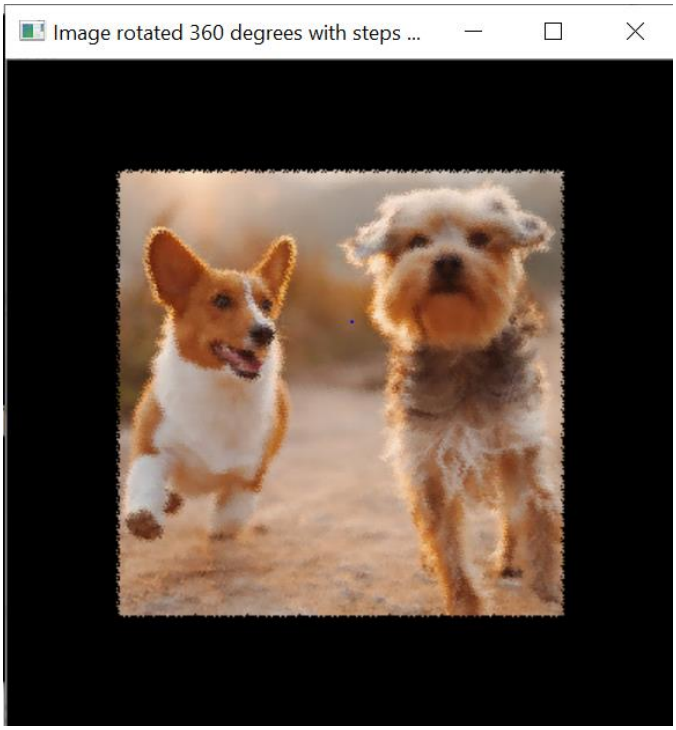

Image rotated 360º with steps of 60º, 6 rotations applied.

- A chart with the following information filled in:

Angle Step Size	# Rotations	Absolute Color Error	Pixel Displacement	(# Rotations) * (Pixel Displacement)
45	8	3.964502970377604	112.41475224565579	899.3180179652463
60	6	5.9210459391276045	146.89337657615758	881.3602594569454
90	4	0.0	207.77391053132274	831.095642125291
120	3	2.265996297200521	254.39701365741442	763.1910409722433
180	2	0.0025685628255208335	293.8366838747392	587.6733677494784
360	1	0.007904052734375	0.0018107096354166667	0.0018107096354166667

- Conclusions drawn from the completed chart:

  There seems to be a correlation between the absolute color error and the pixel displacement. It seems that the further each pixel travels, the more color error exists. This seems to be only be the case for step sizes that cause the image to be on a diagonal angle and due to those angles having more rounding error. As we can see from the data, there is very little color error for 90º, 180º, and 360º because there was less rounding error. Also, because the pixel displacement is not 0 for the 360º rotation, we can tell that there is still some rounding error for that step size. As we multiply our #rotations by pixel displacement, we see a straight decrease.

- Discussion of any issues encountered:

  I encountered many issues. The first issue I encountered was figuring out how to scale the canvas to be the appropriate size to avoid clipping the corners of the original image. I then rotated the larger image which caused index errors since I was going out of the bounds of my matrix. I corrected that error by allowing clipping of the black portion of my larger image; because only the black portion is being clipped, I was able to avoid clipping the original image. I also had issues with black streaks appearing on my image. I corrected that issue by rotating my matrix the opposite way so all my indices would be updated. After I corrected these issues, I had trouble scaling my image back down to the original size so I could calculate the absolute color error. Furthermore, I was not taking the absolute value of my absolute error calculation so some of the values were cancelling each other out. Finally, the last issue I encountered was calculating the pixel displacement using the wrong coordinates.