16720A: Computer Vision-Homework 1

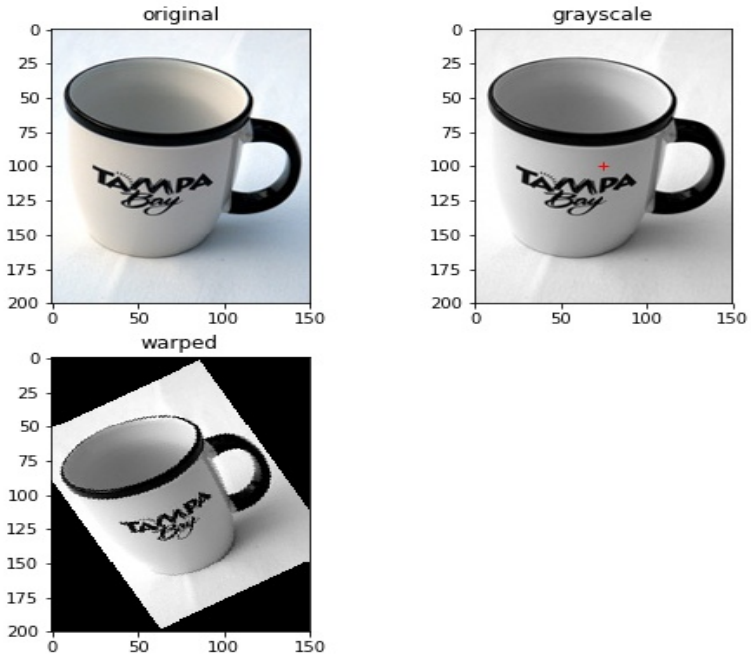
[redacted]

September 13, 2019

1 Color Channel alignment

Using Sum of Squared Differences

2 Image warping



3 Hough transform

3.1 Theory Questions

2.1)

$$xcos(\theta) + ysin(\theta) = \rho \tag{1}$$

Multiply and divide by $(x^2 + y^2)^{0.5}$
Using the law of triangles, assume

$$sin(\phi) = \frac{x}{(x^2 + y^2)^{0.5}} \tag{2}$$

$$cos(\phi) = \frac{y}{(x^2 + y^2)^{0.5}} \tag{3}$$

Substitute (2) and (3) in (1):

$$(x^2 + y^2)^{0.5} * (sin(\phi)cos(\theta) + cos(\phi)sin(\theta)) = \rho \tag{4}$$

Assume $A = (x^2 + y^2)^{0.5}$ Hence, each image point (x, y) results in a sinusoid in (ρ, θ) Hough space.

$$\rho = Asin(\theta + \phi) \tag{5}$$

Here, A is the amplitude of the sine wave and ϕ is the phase shift. Also from equations (2) and (3):

$$tan(\phi) = \frac{y}{x} \tag{6}$$

2.2) Traditionally, a line $y = mx + c$ can be represented using slope m and intercept b in space. But if the line is vertical, this poses a problem as the slope of the line is now tan 90 which is infinity. Slope m would have unbounded values. For computational ease and better edge detection, we parameterize the line in terms of (ρ, θ) instead of slope and intercept (m, c). Also the accumulator array becomes very large because the value of slope ranges from $(-\infty, \infty)$.
Normal form of line is

$$xcos(\theta) + ysin(\theta) = \rho \tag{7}$$

Slope intercept form of line is

$$y = mx + c \tag{8}$$

Substitute (7) in (8)

$$y = \frac{\rho}{sin(\theta)} - \frac{xcos(\theta)}{sin(\theta)} \tag{9}$$

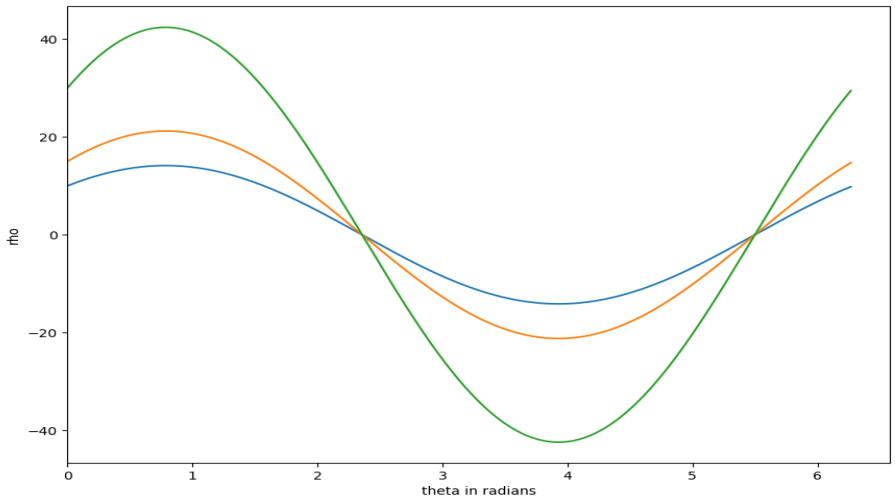
Compare equation (9) with equation (8)

$$m = \frac{-cos(\theta)}{sin(\theta)} \tag{10}$$

$$c = \frac{\rho}{sin(\theta)} \tag{11}$$

2.3) $\rho = (H^2 + W^2)^{0.5}$
θ ranges from 0 to 360 degrees

2.4)
Visualization of the graphs The intersection point by observation is $\rho = 0$, $\theta = 2.3565$.



Using equations (10) and (11),
$m = 1.0006$, which is approximately 0
$c = 0$.

3.2

Anaconda virtual environment was installed with OpenCV

3.3

The Hough code is generated

3.4 Write up

1 and 2. The inbuilt functions I used are with their significance:
cv2.GaussianBlur and cv2.Canny and cv2.HoughLinesP

Cv2.GaussianBlur	
Image	Input image
(sigmaX, sigmaY)	Standard deviations of kernel in X and Y direction
Border Type	Specifies image boundaries while kernel is applied on image borders.

Cv2.Canny –	outputs an edge map of the image
Image	Single channel 8 bit input image
Threshold 1	First threshold for the hysteresis procedure – probability of an edge below this is 0
Threshold 2	Second threshold of hysteresis procedure – probability of an edge above this is 1
Aperture Size	Aperture size for Sobel operator – corner size of detection
L2 Gradient	Use L1 or L2 to calculate image gradient magnitude

Cv2.HoughLinesP	Output vector of lines.
Edges	Input image of edges from canny
rho	Distance resolution of the accumulator in pixels
theta	Angle resolution of the accumulator in radians
threshold	Minimum number of votes for a line to be included
Min line length	Minimum length of line
Max line gap	Maximum gap between line segments

3.

Cv2.GaussianBlur	
Image	Original image
(sigmaX, sigmaY)	(5, 5)
Border Type	cv2.BORDER_DEFAULT
Cv2.Canny	
Image	Gray, blur image
Threshold 1	100
Threshold 2	220
Aperture Size	3
L2 Gradient	False
Cv2.HoughLinesP	
Edges	Image of edges
rho	800
theta	Pi/180
threshold	100
Min line length	0
Max line gap	0

Figure 1: Image01

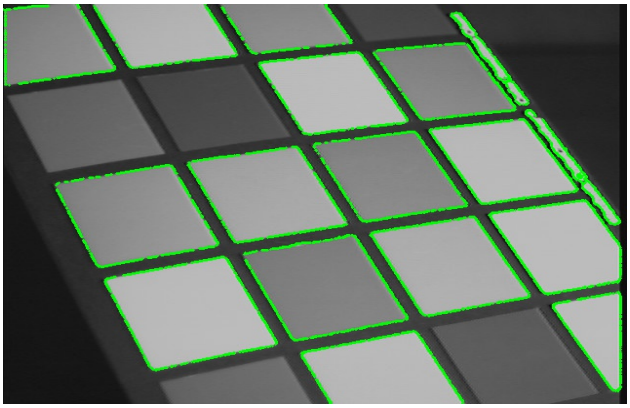


Figure 2: Image01

Elaborating on the mathematical significance and iteration performed for figure 1. For gaussian blurring, I stuck to the default parameters with a kernel size of (5, 5) in x and y directions.

For canny image detection, threshold 1 is 100 and threshold 2 is 220. No edges are detected below 100 whilst edges are always detected above an intensity of 220. For intensities lying within this range, votes are cast by each pixel to gauge the probability of an edge existing there. I varied these values until all possible edges could be detected. Varying these edges I realized that I could filter out the noise but if I changed it too much or increased the upper limit drastically, important features of the image were lost.

For hough line detection, I passed the image with detected edges. The lines were spaced very far apart with a rho of 1. So I increased rho to the maximum possible value it could accommodate which is 800 - the diagonal of the image. Significant improvement was seen. However varying theta seemed to have no significant impact and I left it at $\pi/180$.

Varying the number of votes and minimum line length seemed to have the most impact. Minimum line length failed at corners of the image. Increasing the number of votes made it less susceptible to detecting false lines but at the same time ran the risk of losing small existing lines. Same goes for minimum line length. Whenever I noticed gaps in the image, I increased maximum gap width so that gaps between lines could be filled in for increased continuity,

Cv2.GaussianBlur	
Image	Original image
(sigmaX, sigmaY)	(5, 5)
Border Type	cv2.BORDER_DEFAULT
Cv2.Canny	
Image	Gray, blur image
Threshold 1	50
Threshold 2	150
Aperture Size	3
L2 Gradient	False
Cv2.HoughLinesP	
Edges	Image of edges
rho	400
theta	Pi/180
threshold	100
Min line length	0
Max line gap	0

Figure 3: Image02

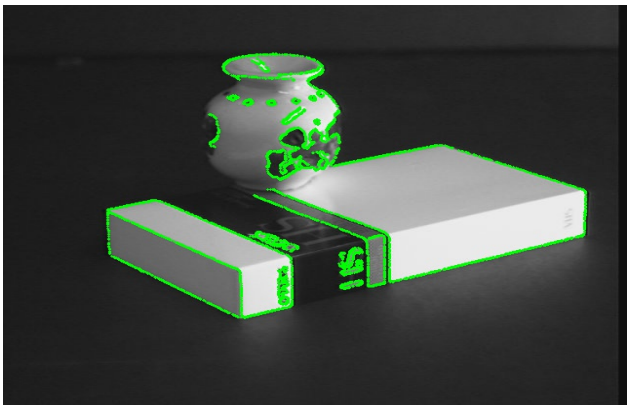


Figure 4: Image02

Cv2.GaussianBlur	
Image	Original image
(sigmaX, sigmaY)	(5, 5)
Border Type	cv2.BORDER_DEFAULT
Cv2.Canny	
Image	Gray, blur image
Threshold 1	30
Threshold 2	100
Aperture Size	3
L2 Gradient	False
Cv2.HoughLinesP	
Edges	Image of edges
rho	400
theta	Pi/180
threshold	100
Min line length	0
Max line gap	0

Figure 5: Image03

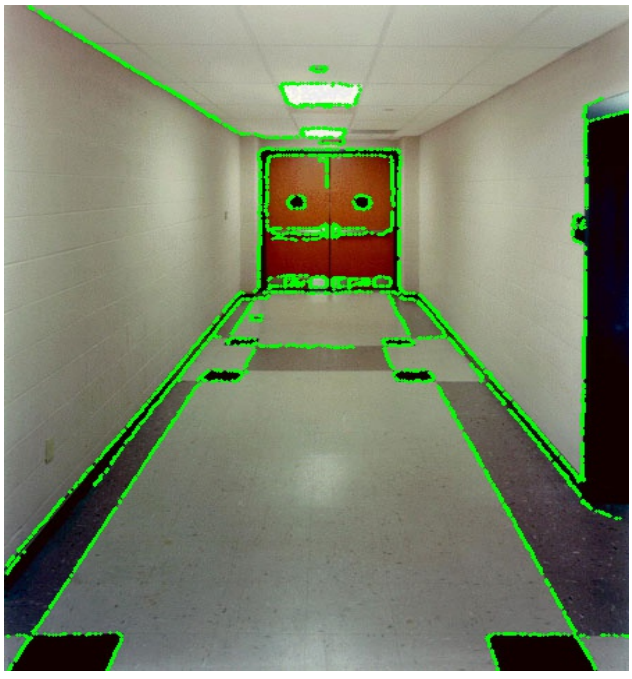


Figure 6: Image03

Cv2.GaussianBlur	
Image	Original image
(sigmaX, sigmaY)	(5, 5)
Border Type	cv2.BORDER_DEFAULT
Cv2.Canny	
Image	Gray, blur image
Threshold 1	10
Threshold 2	100
Aperture Size	3
L2 Gradient	False
Cv2.HoughLinesP	
Edges	Image of edges
rho	800
theta	Pi/180
threshold	200
Min line length	0
Max line gap	0

Figure 7: Image05

Figure 8: Image05

Cv2.GaussianBlur	
Image	Original image
(sigmaX, sigmaY)	(5, 5)
Border Type	cv2.BORDER_DEFAULT
Cv2.Canny	
Image	Gray, blur image
Threshold 1	10
Threshold 2	150
Aperture Size	3
L2 Gradient	False
Cv2.HoughLinesP	
Edges	Image of edges
rho	800
theta	Pi/180
threshold	250
Min line length	0
Max line gap	0

Figure 9: Image08

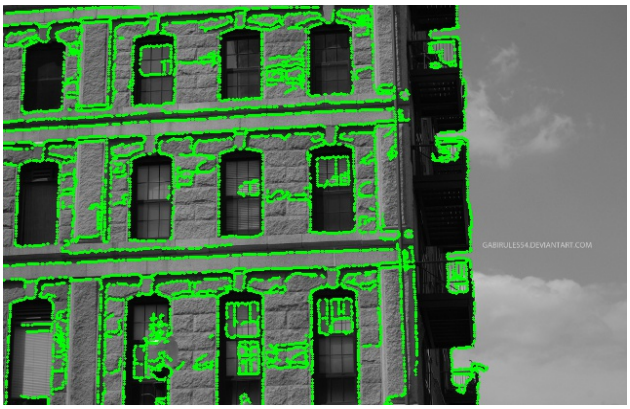


Figure 10: Image08