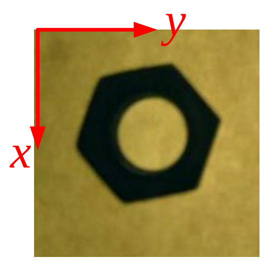
Georgia Institute of Technology
George W. Woodruff School of Mechanical Engineering
ME6406 Machine Vision (Fall 2016)

All programs should be written using MATLAB. MATLAB and its toolboxes are available in ME Computer Lab (MRDC2105). To save the image from the ME6406 website page, point the mouse cursor on the image and click on the mouse's right button. Choose <Save image as>.

Assignment #2: Due **Thursday September 29, 2015**. (Solutions should include m-files, results, and an explanation of your results in a document. All m-files must be submitted electronically in a zipped file through T-square. A confirmation email will be sent on receipt of submission.

1. Feature Points Detection
   a. Use the ***rho-theta signature*** method discussed in class, implement using MATLAB, to locate all the corners of the object in 'HW2.png'. Plot the $\rho\theta$ signature for locating the corners. Show the coordinates of the corners with respect to the coordinate system as shown.
   b. Repeat (a) with the ***median length*** method. Pick an appropriate N (length of the neighbor). Plot median-length as a function of path.
   c. Repeat (a) with the ***curvature*** method to locate the 'peaks' of the curvature along the boundary. Plot the graph that you used to locate the corners. Show the coordinates of the corners with respect to the coordinate system as shown.



HW2.png

2. Template Matching
   a. Forward transformation:

Write a MATLAB function to perform a coordinate-transformation with known parameters (k, θ; x_d, y_d). Use the three template points and the parameters in Table 1 to check the computation. Plot the three points before and after transformation (accurate to three decimal places) on the same figure.

Table 1.	Template			Parameters
	1	2	3	
X	0	7	3	Scale k =0.8; Orientation θ=60°; and Translation (x_d=6, y_d=7)
Y	0	4	8	

   b. Pseudo inverse method:

Wirte a MATLAB function to perform the pseudo-inverse method to determine the four parameters (k, θ; x_d, y_d) for a given set of (template-target) data pairs. To verify your algorithm, use the three template points in Table 1 and the three corresponding simulated "target points in an image) from Part 2(a) to find the four parameters and check these values with Table 1.

   c. Polygon matching:

Write a MATLAB program to perform template matching. Use the template-matching algorithm to identify the match points between template and target (given in Table 2 and Fig. 2), and determine the transformation parameters; namely scale k, orientation θ, and displacement (x_d, y_d).

Table 2	Template				
	1	2	3	4	5
X	2	6	8	5	-3
Y	0	2	6	8	5
	Target				
	a	b	c		
x	2.28	10.621	9.545		
y	16.28	10.318	15.576		

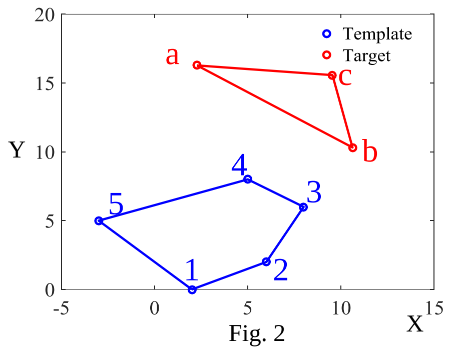


Fig. 2

Note 1: Given n points, there are $\binom{n}{3} = \frac{n!}{(n-3)!3!}$ unique triangles. Use Matlab function nchoosek( ) to compute all possible permutations of vertices to form unique triangles.

Note 2: As there are many triangles to be matched, your program must automate the matching process to avoid tedious manual matching.

Note 3: Determine the 'best' match by finding the one that yields the smallest 'matching error' defined by Eqn (11) in [Lee, *et al*., 1992]. ($E = \sum_{i=1}^{k} \sqrt{(x_{ti} - x_i)^2 + (y_{ti} - y_i)^2}$)

Note 4: The center of the object silhouette is at origin ($X_c = Y_c = 0$)

3. <u>Hough Transform</u>

a. Show that $\begin{bmatrix} x_o & y_o \end{bmatrix}^T = \upsilon \begin{bmatrix} g_x & g_y \end{bmatrix}^T$ where $\upsilon = \left(x g_x + y g_y \right) / \sqrt{g_x^2 + g_y^2}$.

b. With the parameters (x_o, y_o), write a MATLAB program to perform Hough transform (foot-of-normal parameterization) on the image "Camera.png" to find the yellow edge (≈line).

Step 1: For this problem, use the Matlab functions "rgb2gray.m" to convert RGB image to gray-level image, and the "edge.m" function to find the edge image resulting a binary image where the edge pixels are 1.

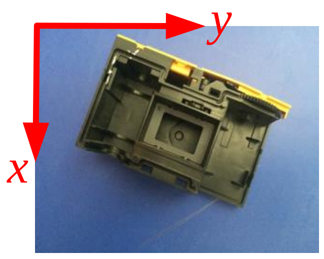
Step 2: Compute the gradients in *x* and *y* directions from the edge image in Step 1.

Step 3: Determine the accumulate matrix for (x_0, y_0).

Step 4: Find (x_0, y_0) and equation of the yellow edge of the camera.

c. Use Hough Transform on the points of inner boundary to find the circle in HW2.png.

Note: The Matlab commands hough.m, houghpeaks.m, houghlines.m are NOT allowed.



Camera.png

Reference: Lee, K-M. and S. Janakiraman, "A Model-based Vision Algorithm for Real-Time Flexible Part-feeding and Assembly," Paper number: MS 92-211. *SME Applied Machine Vision Conf.*, June 1-4, 1992, Atlanta, GA.