Engineering 13300 HW 12
MATLAB 4 – Individual Tasks

Guidelines for Tasks 4-5:
Tasks 4-5 are individual tasks. You may seek help from classmates, the instructional team, or others but the work you submit should be your own. If you collaborate with others and use information developed together or by someone else, ALWAYS document and reference that material. Each student is responsible for submitting their own assignment to Gradescope.

Individual Task 4 (of 5) Image Rotation

Background

Image rotation is an increasingly commonly used feature in many smart-phone image apps. In this problem you will create an implementation of image rotation based on user input in MATLAB. Draw a flowchart and save it as `Ma4_Ind_flowchart_username.pdf`

Problem Steps

1. Open the **ENGR133_MATLAB_UDF_Template.m** file. Complete the header information. Save your script as Ma4_Task4_image_rotate_*username*.m
2. Create a MATLAB program that reads `block.png` image file and outputs it to the screen using the `imshow` function. Properly label this image as **"Color Original Image".**
3. `imread` returns an array of **rows X columns X frames**. The number of frames for a colored image is generally '3' (RGB colors) or '5'. Convert the image into gray scale image using an in-built MATLAB function called `rgb2gray` and label this image as **"Gray Original Image".**
   Hint: You can look up on what the function 'rgb2gray' does in the official MATLAB documentation.
4. Use the `menu` function in MATLAB to ask the user to choose either the colored original image or gray original image to perform the operations mentioned in step 5.
   Give the user the following options:
   a. Choose colored image
   b. Choose gray-scaled image
5. Next, using the `menu` function again, ask the user to select a rotation for either the chosen gray scale image or color image. Give the user the following options:
   a. 90 degrees clockwise
   b. 90 degrees counter-clockwise
   c. 180 degrees rotation
6. For 90º clockwise rotation, use only nested 'for' loops for rotating the image (either gray scale or color).
7. For 90º counter-clockwise rotation, use MATLAB's `rot90` function to rotate in the counter-clockwise direction.
8. For 180º rotation, you can choose to either use nested for loops or transpose/permute to rotate the images.

9. Write each of the above-mentioned operations in step 5 in separate .m files as functions with the following file names
    - Ma4_Task4_90_clockwise_username.m
    - Ma4_Task4_90_counterclockwise_username.m
    - Ma4_Task4_180_username.m
10. Output the modified image to the screen and label the output **Image Rotated XX degrees** where XX indicates the rotation selected by the user (i.e. 90, -90, or 180 degrees)

** NOTE: Do not use any built-in MATLAB functions such as **`imrotate`** to complete this task.

Publish the Ma4_Task4_image_rotate_*username.m* file as a PDF using any one of the test cases and name it Ma4_Task4_image_rotate_*username*.pdf.
Upload Ma4_Task4_image_rotate_*username.m* also.
Make sure all the images are shown (with appropriate titles) in the published document as well.

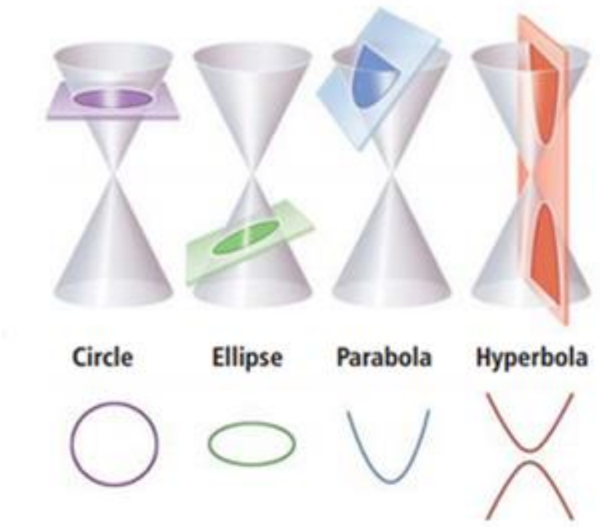
Task 4 files to submit:

- Ma4_Task4_90_clockwise_*username*.m
- Ma4_Task4_90_counterclockwise_*username*.m
- Ma4_Task4_180_*username*.m
- Ma4_Task4_image_rotate_*username*.m
- Ma4_Task4_image_rotate_*username*.pdf (published file)
- Ma4_Ind_flowchart_username.pdf

Individual Task 5 (of 5) Plotting Conic Sections

Background:

A conic section is a plane that forms after cutting at varied angles from a cone (hence, conic sections). The shapes vary according to the angle at which it is cut from the cone.



Source: https://www.ellipsesconicsections.weebly.com/

There are four types of conic sections. They are Circles, Ellipses, Parabolas, and Hyperbolas.

The conic sections appear everywhere in our everyday life. Some examples of a parabola are satellite dish, the motion of a projectile, etc. Some examples of an ellipse include the motion of planets around the Sun, shape of a football, etc. Shape of cooling towers in nuclear reactors, the path of some comets are all examples of hyperbolas. You can find examples of circles in real-life on your own.

For the given values of x, plot all the conic sections with the given parameters in MATLAB and Python.

Circle: Plot a circle with center at (2,-3) and a radius of 5 units.

Hint: Use parametric equation:

x = r*cos(theta) + h

y = r*sin(theta) + k

where r = radius of circle, (h,k)=Cartesian coordinates of the center of the circle

Parabola: For x ranging from -8 to 8 with 100 elements, plot a parabola with the following equation:

$$y = 0.1x^2$$

Ellipse: The parameters of the ellipse are as follows:
a ≡ semi-major axis = 15 units
b ≡ semi-minor axis = 6 units
Hint: Use parametric equation
x = a*cos(theta)
y = b*sin(theta)

Hyperbola: The parameters of the ellipse are as follows:
a ≡ semi-major axis = 2 units
b ≡ semi-minor axis = 1 units
For values of x varying from –10 to 10 with 100 elements, plot the hyperbola with the following equation:

$$\frac{y^2}{b^2} - \frac{x^2}{a^2} = 1$$

NOTE: A hyperbola has two branches. Plot both the branches

Problem Steps:

1. Open the **ENGR133_MATLAB_Template.m** file. Complete the header information. Save your script as Ma4_Task5_conics_*username*.m
2. Create a MATLAB program that plots the conic sections in the same figure window but different subplots.
3. Each conic section must be in a separate quadrant of the figure. Turn the grid 'on'. The figure must have main title, subtitles for the subplots, xlabel and ylabel.

Now, plot the above conic sections in Python as well. The intension is to signify that both programming languages are equally capable of performing such operations. You are allowed to use matplotlib, NumPy and math modules only.

Publish the Ma4_Task5_conics_*username.m* function as a PDF and name it Ma4_Task5_conics_*username*.pdf. Upload Ma4_Task5_conics_*username.m* and Ma4_Task5_conics_ *username*.py also.

Task 5 files to submit:

- Ma4_Task5_conics_*username*.m
- Ma4_Task5_conics_*username*.pdf (published file)
- Ma4_Task5_conics_ *username*.py (Python File)