Submission before: 07.12.2015
Discussion on: 08.12.2015
Submission on stud.ip, submission folder for sheet.
Please submit a zip file containing the .m files for Matlab programming tasks.

Exercise 1 *(Understanding Hough Transform – 5p)*

This exercise is intended to help you understand (linear) Hough transform (you may use the buildin function `hough` here).

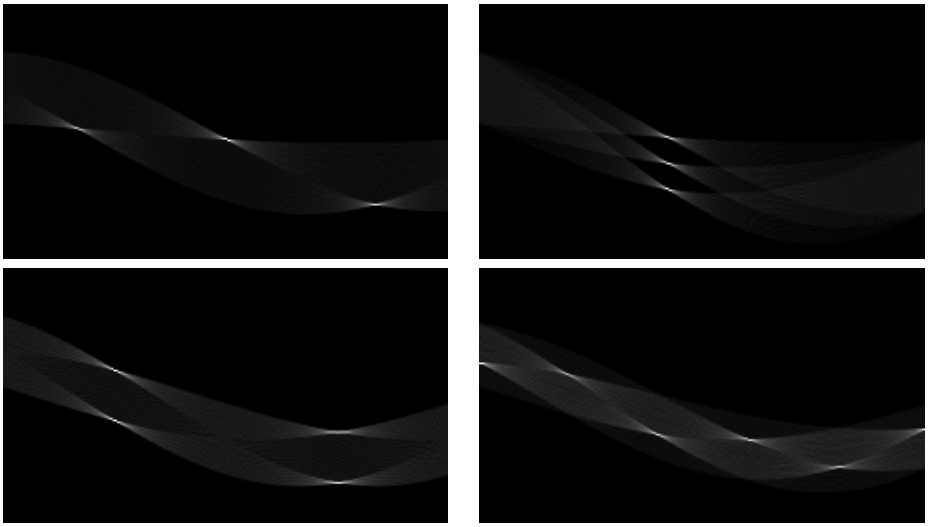
(a) First look at the Hough transform of a single point. Create an image with a single point and display its Hough transform. Move that point around.

(b) Now place multiple points in a row. Again plot the Hough transform.

(c) Draw a single line and plot the Hough transform. Vary position and angle.

(d) Draw complex figures like triangles and other polygons and display their Hough transform.

(e) Look at the following images that display results of Hough transforms. How do the original figures look like?



Exercise 2 *(Implementing Hough Transform – 5p)*

Now implement a simple version of the linear Hough transform (ignoring gradient information).[1] Proceed in two steps (and test your function by applying it to examples from exercise 1):

1. First define an accumulator space with the X-axis representing the angle values, and the Y-axis representing the distance to the origin. Use angle values in the range $[0, \pi]$. `linspace` may be a useful function here.

2. Then, for every white pixel calculate the corresponding curve in Hough space and add it to the accumulator space: the calculation of the distance to the origin can be performed based on the (x, y) coordinates of the pixel in the original image (image space) and the angle θ. The formula is given in the lecture slides. Since we don't use gradient information, i.e. we make no assumption about the angle θ, you will have to iterate through all possible values of θ and calculate the distance for these values.

[1] Pseudocode for that algorithm can be found at Wikipedia:
`https://de.wikipedia.org/wiki/Hough-Transformation#Einfacher_Algorithmus`

Exercise 3 *(Circular Hough Transform – 10p)*

It was suggested in the lecture, that circular Hough transform can be used to implement a simple form of "eye tracking". Try this using your webcam.
Hints:

- You may start with a single snapshot before applying your function to a video stream.
- You may use the MATLAB function `imfindcircles`, which implements circular Hough transform (results can be displayed using `viscircles`). You probably have to specifiy appropriate values for parameters like `Sensitivity`, `EdgeThreshold`, etc.
- You may try to apply some preprocessing to detect edges or regions prior to Hough transform (you may use anything you know from the lecture; MATLAB functions are fine here).
- `imfindcircles` may be inappropriate, as it is designed to detect full circles, while eyes are often partially occluded. You may implement your own Hough transforms, which detects this kind of incomplete circles.
- You may apply additional knowledge to detect eyes, e.g. that they usually occur in pairs. You may also use the approximate size and position from last frame.