Pattern recognition systems – Lab 3

Hough Transform for line detection

1. Objectives

The main objective of this lab is to implement the Hough Transform for line detection from edge images.

2. Theoretical overview

The classical Hough transform is a method that solves a classical image processing problem: finding lines in an image that contains a set of interest points. The straightforward method of computing lines from each pair of points has an increased computational complexity of O(n^2), and is unusable for a large number of points. The Hough transform was first proposed and patented by Peter Hough [**Hou62**]. It is a real-time method to count how many points are placed on each possible line in the image. It relies on the representation of the lines in the slope-intercept form ($y=ax+b$), and on building the line parameter space, also called Hough accumulator. For each image interest point, all possible lines are considered, and the corresponding elements in the line parameter space are incremented. Relevant image lines are located at the locations of the local maxima in the line parameter space.

The initial proposal was focused on the detection of lines in video images, based on a slope and free-term line representation. This representation is not optimal because it is not bounded: in order to represent all the possible lines in an image, the slope and the intercept terms should vary between $-\infty$ and $+\infty$. The work of Duda and Hart [**Dud72**] made the Hough transform more popular in the computer vision field. The main problem of the original Hough transform (unbounded parameters) is solved by using the so-called normal parameterization. The normal parameterization of a line consists of representing the line by the normal vector form the origin to the line. The normal representation (1) is commonly referred as ρ-θ representation (Fig. 1).

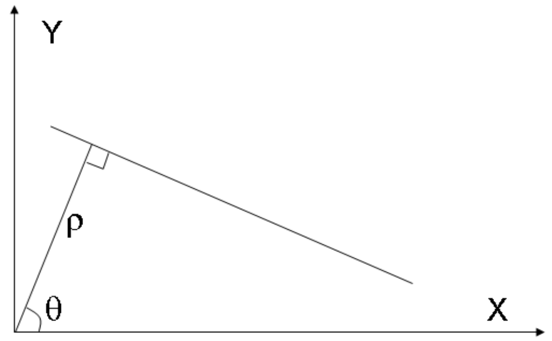


Fig. 1. Line represented by its normal vector at angle θ and at distance ρ along the normal vector from the origin.

$$\rho = x\cos(\theta) + y\sin(\theta) \tag{1}$$

Aside the fact that the parameters are bounded, parameter quantization plays an important role in decreasing the computational complexity of the method. Quantization determines the size of the Hough accumulator. For each of the two line parameters a quantization level must be established, depending on the desired accuracy (ex. the accuracy of ρ can be of 10, 1, 0.5 pixels etc, the accuracy of θ can be of 10 deg, 1 deg, 0.5 deg etc). The parameters ρ and θ have a limited range considering the fact that an image has a limited size. The maximum value for ρ is the diagonal ρ_{max} of the image. Depending on the interval selected for θ, there are several equivalent configurations for the line parameter range (the first one is proposed in the original work):

$$\begin{aligned}&1.\ \theta \in [-90^\circ, 90^\circ)\ \textit{or}\ \theta \in [0, 180^\circ),\ \ \rho \in [-\rho_{\max}, +\rho_{\max}]\\ &2.\ \theta \in [0, 360^\circ),\ \ \rho \in [0, +\rho_{\max}]\end{aligned} \tag{2}$$

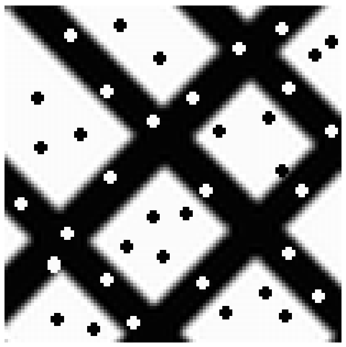
Let us assume that the Hough accumulator H represents the quantized line parameter space. The quantization steps for ρ and θ are $\Delta\rho$ and $\Delta\theta$ respectively, and their maximum values are ρ_{max} and θ_{max}. The accumulator will have a size of ($\rho_{max}/\Delta\rho$ x $\theta_{max}/\Delta\theta$). H is built with the following simple steps:

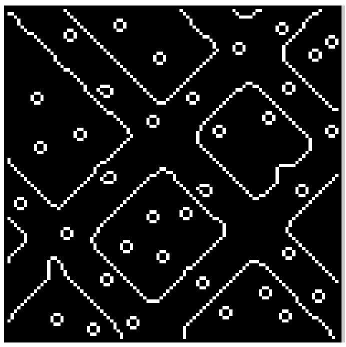
1. Initialize H with 0.
2. For each edge point $P(x, y)$ compute all possible lines and increment the associated locations from H:

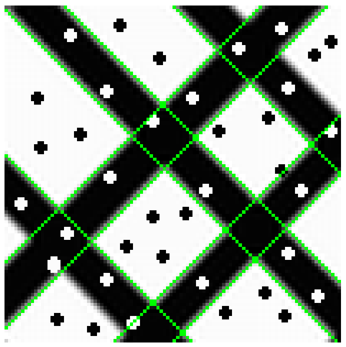
```
for each $\theta$ from 0 to $\theta_{max}$ (with a step of $\Delta\theta$)
    compute $\rho = x\cos(\theta) + y\sin(\theta)$
    if $\rho \in [0, +\rho_{\max}]$ increment $H(\rho, \theta)$
end for
```

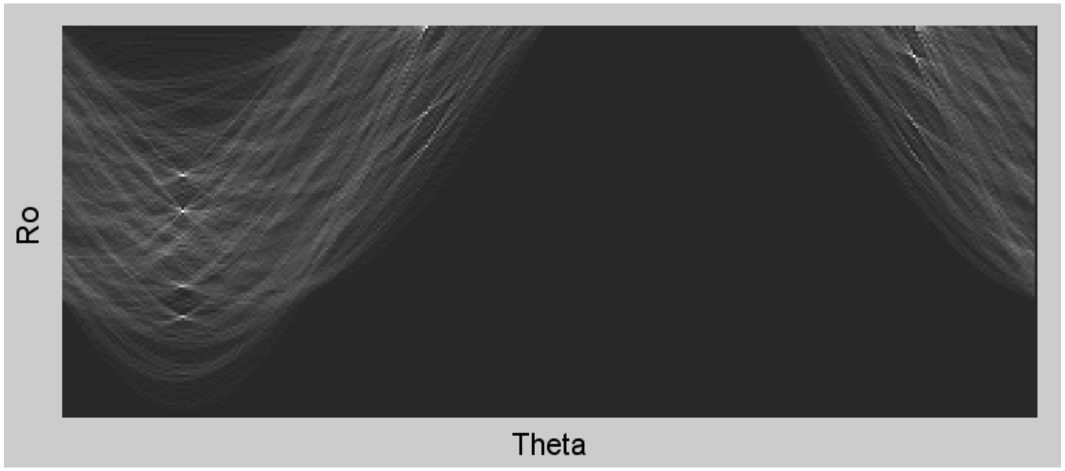
The increment operation of a Hough location can also be weighted, if different weights are desired for each interest point. Once the accumulator is built, relevant lines are extracted as the local peaks of the accumulator. An example of line detection based on the Hough transform is presented below (Fig. 2). The parameter ranges for this example are [0, 360] degrees for θ, [0, 144] pixels for ρ. The parameter accuracy is 1 degree for θ and 1 pixel for ρ.

Choosing the correct level of quantization is important. If the quantization is too fine then the resolution increases, but so does the processing time, and also the chance that collinear points will fall into different accumulator bins (this might cause multiple detections and the fragmentation of certain lines).

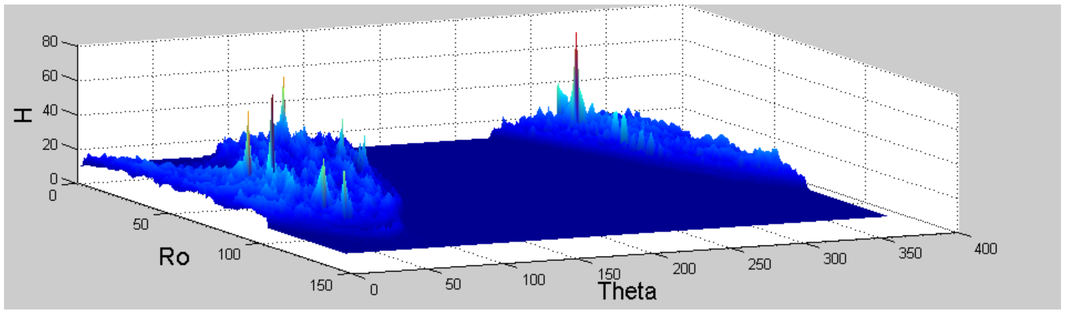

a.


b.


c.



d.



e.

Fig. 2. **a.** An image containing a pattern with straight borders, corrupted by salt-and-pepper like noise, **b.** The edges detected with the Canny edge detector, **c.** The most relevant image lines are displayed with green, associated to the most relevant 8 peaks from the Hough accumulator, **d.** The Hough accumulator displayed using an intensity encoding, **e.** The Hough accumulator displayed in 3D, using color encoding.

Although the Hough transform is widely used for line detection, it can also work with more complex curves as long as an adequate parameterization is available. Duda and Hart [**Dud72**] also proposed the detection of circles based on the Hough transform: a 3D parameter space is needed and each interest point is transformed into a right circular cone in the parameters space (all the possible circles containing the point). Later, Ballard generalized the Hough transform to detect arbitrary non-analytical shapes **[Bal81].**

3. Implementation Details

Use the simplest configuration for parameter quantization: 1 pixel for ρ and 1 degree for θ. Use the second option for the parameters range from formula (2). The size of the Hough accumulator will be *360 x (D + 1)*, where *D* is the image diagonal.

```
Mat Hough(360,D+1,CV_32SC1); //matrix with int values
```

Modify the accumulator like:

```
Hough.at<int>(theta, ro)++;
```

The accumulator needs to be normalized to have values in the range 0-255 to be viewed as a grayscale image. Use the `normalize` method or find the maximum from the accumulator and use:

```
Mat houghImg;
Hough.convertTo(houghImg, CV_8UC1, 255.f/maxHough);
```

In order to locate peaks in the accumulator, you will test for each Hough element if it is a local maximum in a squared window (*n x n*) centered on the element. Use a custom structure to memorize and sort the local maxima.

```
struct peak{
     int theta, ro, hval;
     bool operator < (const peak& o) const {
          return hval > o.hval;
     }
};
```

4. Practical Work

1. Compute the Hough accumulator based on the edge image and display it as a grayscale image.
2. Locate the largest *k* local maxima from the accumulator. Try using different window sizes such as: 3x3, 7x7 or 11x11.
3. Draw lines that correspond to the peaks found. Use both the original image and the edge image for visualization.

5. References

[**Hou62**] P. Hough, “Method and means for recognizing complex patterns”, US patent 3,069,654, 1962.
[Dud72] R. O. Duda and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," *Comm. ACM*, Vol. 15, pp. 11–15, 1972.
[Bal81] D. H. Ballard, "Generalizing the Hough Transform to Detect Arbitrary Shapes", *Pattern Recognition*, Vol.13, No.2, p.111-122, 1981.