

## Laboratory – Hough Transform for Lines

### Learning goals

- understand the principle of the Hough transform
- can implement the Hough transform for lines
- learn to express a vectorized result from the voting space

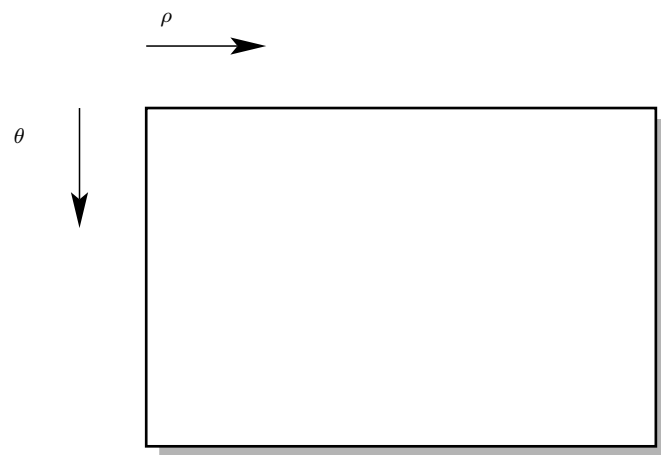
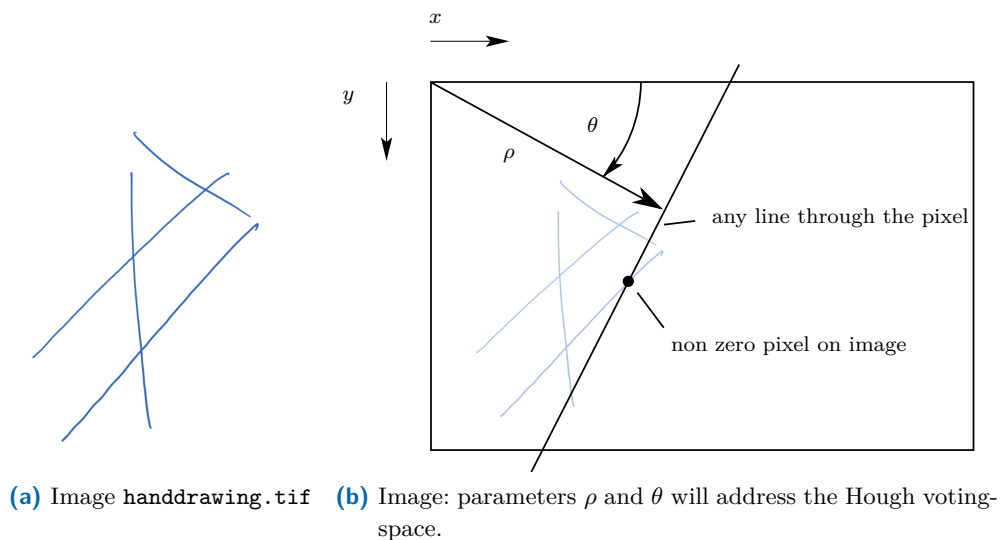


Abbildung 1: Hough voting space.

## Introduction

The image `handdrawing.tif` shows some hand drawn lines. The Hough transform shall be used to recognize them. More precisely, straight lines shall be found that approximate them.

Implement the Hough transform for lines  $HT_L(\theta, \rho)$ . Your implementation should work for binary- as well as grayscale `uint8` images. Before you start implementing the algorithm, answer the following questions for you:

- a) Given a gray value  $g(x, y)$  at image coordinates  $(x, y)$ , how do you compute the locations i.e. inclination  $\theta$  and radius  $\rho$  for the respective votes? More specifically, for a given inclination angle  $\theta$  how do you determine the radius  $\rho$ ?
- b) What are appropriate ranges for the angle  $\theta$  and the radius  $\rho$ ? And what would be an appropriate discrete step for each?
- c) The parameters  $\theta$  and  $\rho$  address the place in the voting space where to vote. However, the matrix representing the voting space must be addressed with indices ranging from 0 to say  $n$ . Think about, how to implement the mapping from parameters to indices and back?

## 1 Exercises

Investigate the prepared code and write additional class methods:

- a) Precompute the normal vector for each discrete value of  $\theta$ . You will need them to compute votes for each nonzero gray value. <sup>1</sup>
- b) Define a mapping function from parameters  $(\theta, \rho)$  to indices in voting space.
- c) Write a voting function, that loops through the image pixels and adds a vote for each potential angle  $\theta$  to the voting space.
- d) Run the program and display an intensity image of the voting space.
- e) Determine the indices at the peaks of the voting space matrix e.g. via the call `peaks = houghpeaks(H, numpeaks, Name, Value)`. Map the indices of the voting space matrix back to the hough parameters.
- f) For each peak, draw the corresponding line on the original image. For this, computing a start point and an end point of each line. These points can be expressed as a linear combination of the normal vector  $\mathbf{n}$  to the line and its orthogonal, the vector  $\mathbf{u}$  parallel to the line<sup>2</sup>. Draw the line on the image with the command `cv.line()`, which allows the starting and end the point to lie outside the image borders.

<sup>1</sup>MATLAB uses the column major, Python and C use the row major memory layout. If you want the two components of the vector to be placed in adjacent memory cells, in MATLAB you must put the vectors into a matrix of dimension  $2 \times n$ , just like in linear-algebra theory. In C and Python it is the other way round.

<sup>2</sup> $\mathbf{u} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \mathbf{n}$