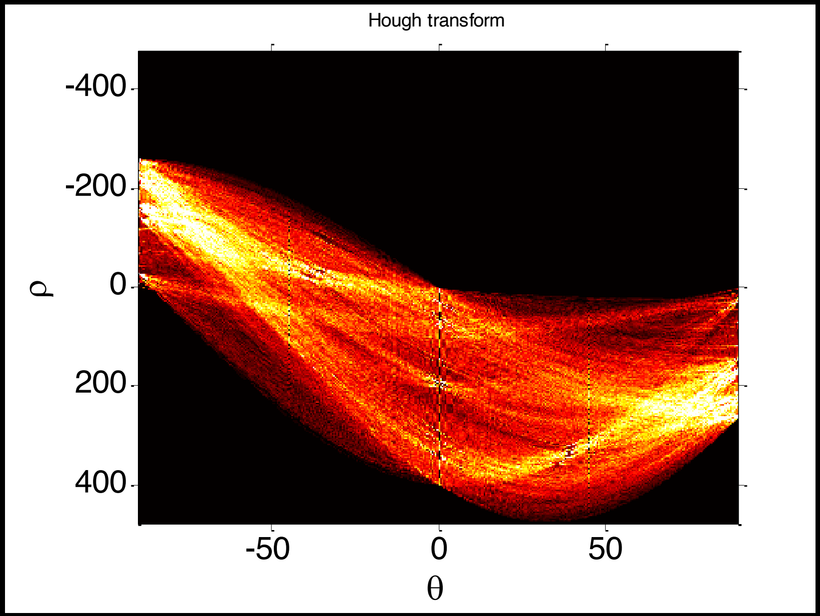
Image Analysis

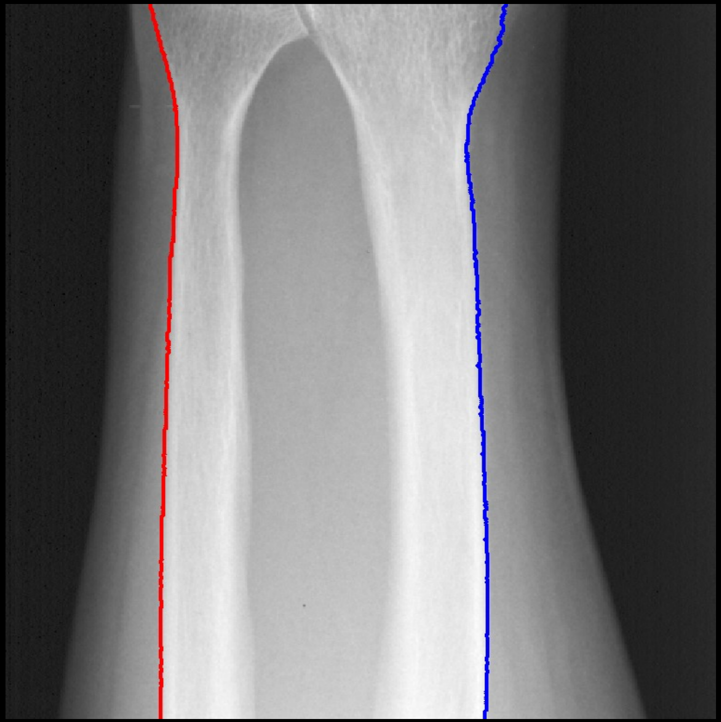
Tim B. Dyrby
Rasmus R. Paulsen
DTU Compute

tbdy@dtu.dk

http://www.compute.dtu.dk/courses/02502

Lecture 8 – Hough Transformation and Path Tracing





Go to **www.menti.com** and use the code **8743 4620**

Quiz testi… …ow long time did it take to develop the Dijkst…

Breaking NEWS!

One morning I was shopping in Amsterdam with my young fiancée, and tired, we sat down on the café terrace to drink a cup of coffee and I was just thinking about whether I could do this, and I then designed the algorithm for the shortest path. As I said, it was a twenty-minute invention.

— *Edsger Dijkstra, in an interview with Philip L. Frana, Communications of the ACM, 2001*[3]

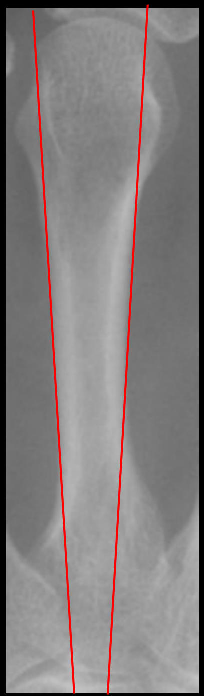
3 years	1 year	2 weeks	1 hour	20 min	37 sec
0	0	0	0	0	0

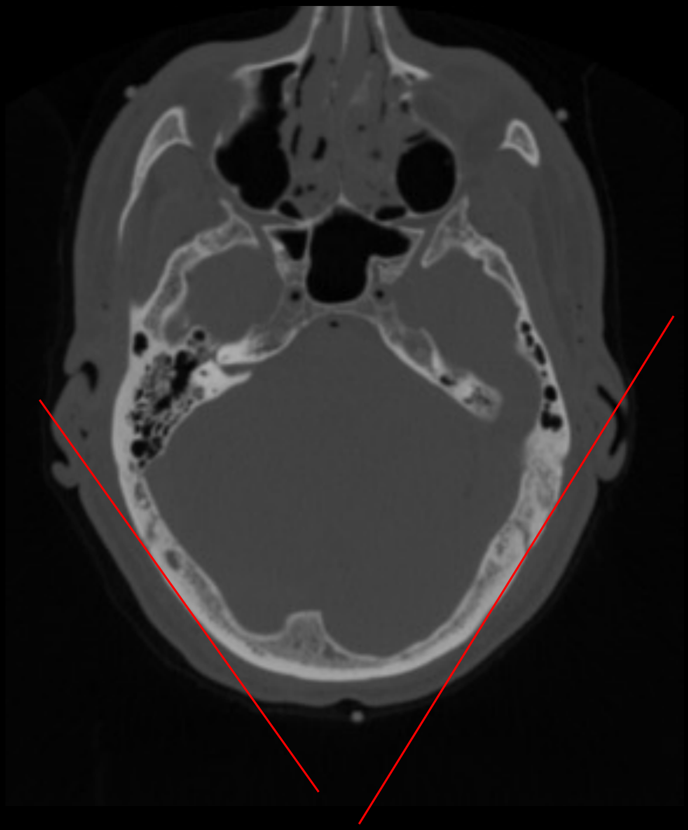
What can you do after today?

- Use the Hough transform for line detection
- Describe the slope-intercept, the general form and the normalised form of lines
- Describe the connection between lines and the Hough space
- Use edge detection to enhance images for use with the Hough transform
- Use dynamic programming to trace paths in images
- Describe how an image can be used as a graph
- Describe the fundamental properties of a cost image
- Compute the cost of path
- Compute an accumulator image for path tracing
- Compute a back tracing image for path tracing
- Choose appropriate pre-processing steps for path tracing
- Describe how circular structures can be located using path tracing

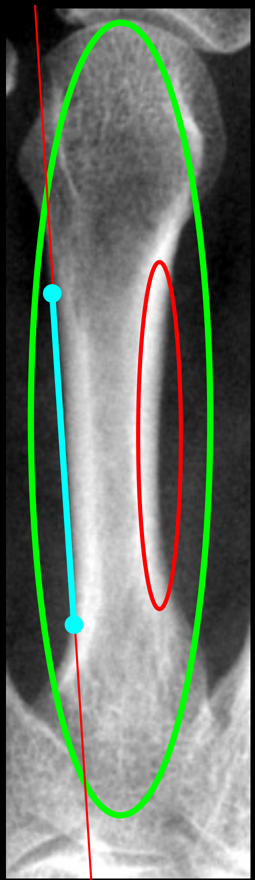
Line Detection

- Find the lines in an image



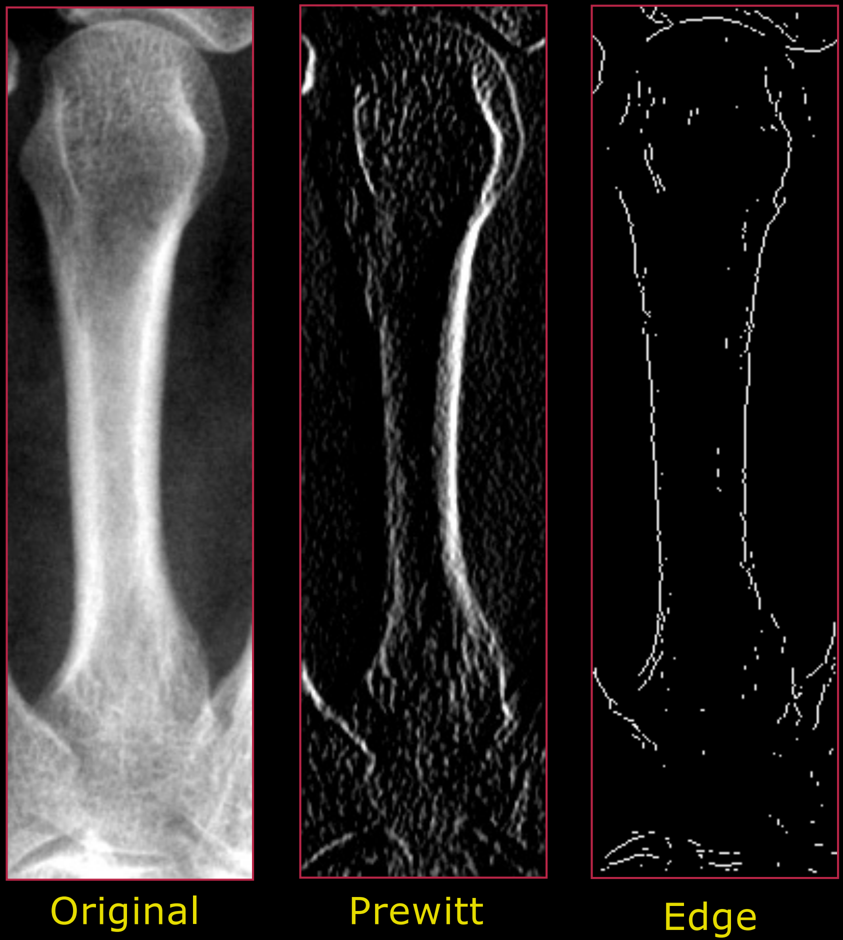


What is a line?



- It can be the entire object
  - Large scale
- Can also be the border between an object and the background
  - Small scale
- Normally only locally defined

Enhancing the lines



- We want to locate the borders
  - Enhance them
- Filtering (Prewitt)
- Edge detection

Prewitt:

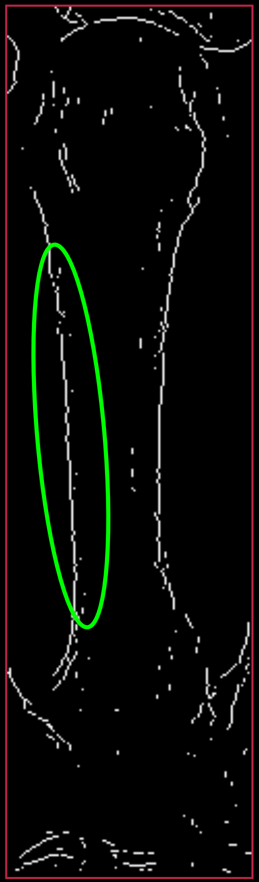
Vertical

-1	0	1
-1	0	1
-1	0	1

Horizontal

-1	-1	-1
0	0	0
1	1	1

What is a line II?



- Result of the edge filter is a selection of white pixels
- Some of them define a line
  - Not a perfect straight line
  - “Linelike”
- How do we find the collection of points that define a line?

Mathematical line definition

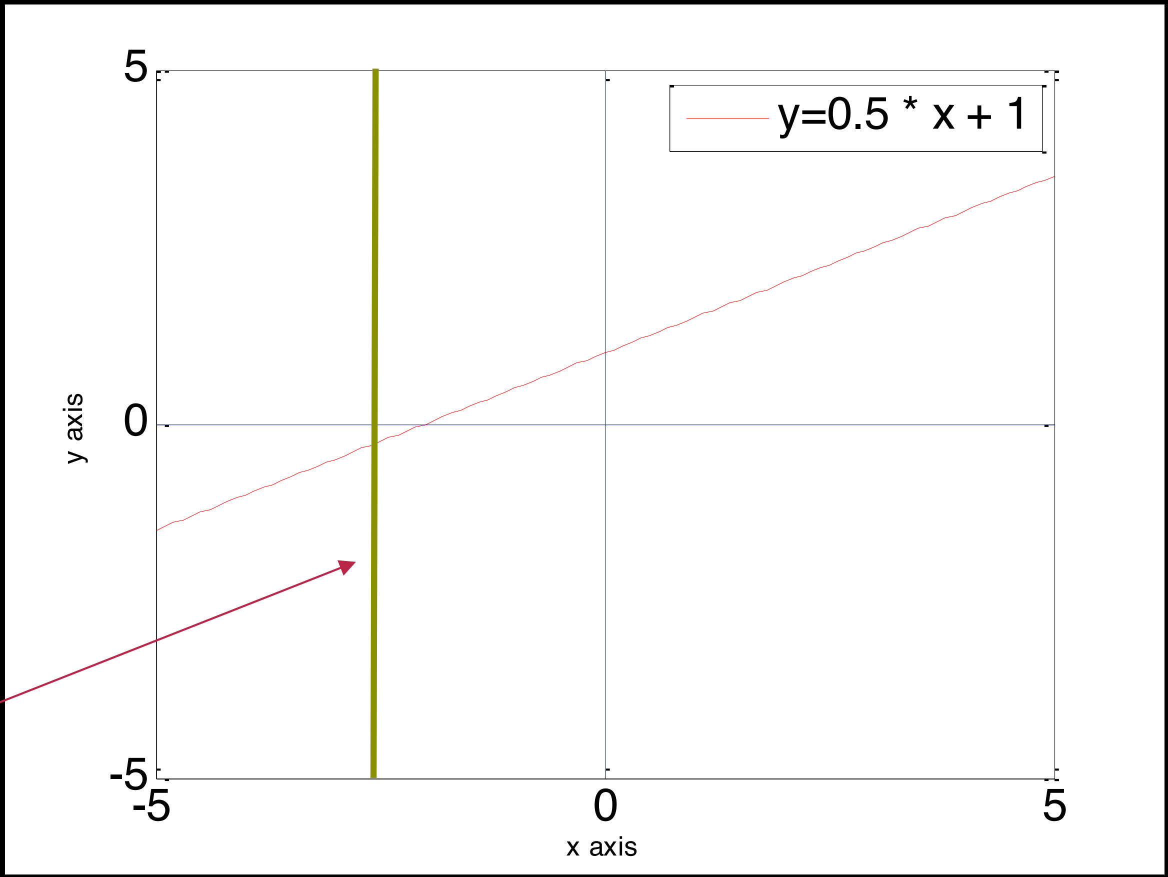
- The classical definition (slope-intercept form)

$$y = ax + b$$

Slope

Intercept

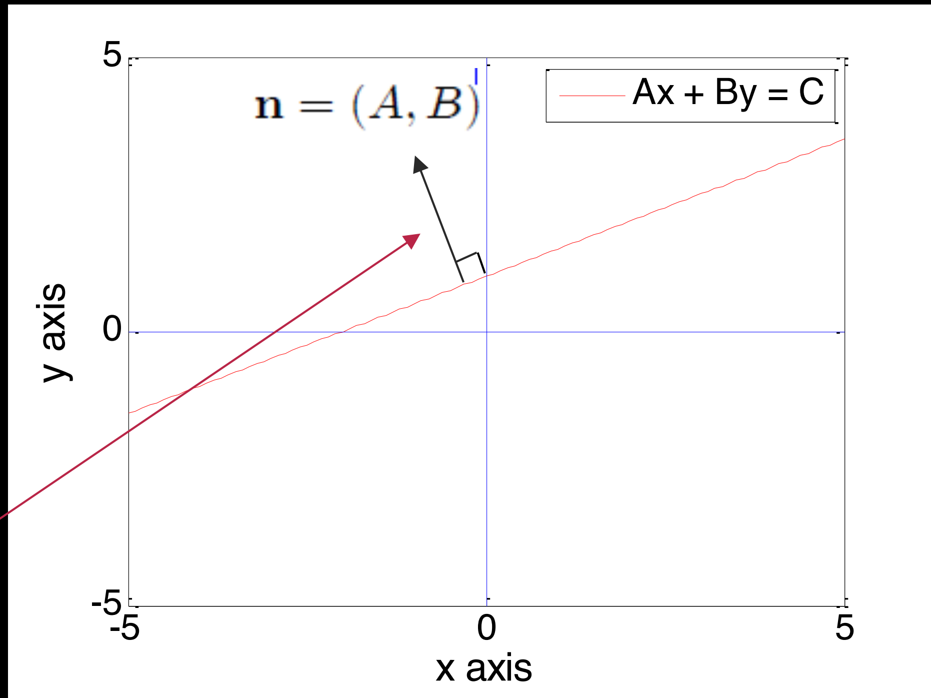
Can not represent lines that are vertical



Mathematical line definition

- General definition (the normal form)

$$Ax + By = C$$



- With

$$A^2 + B^2 = 1$$

Line normal

Mathematical line definition

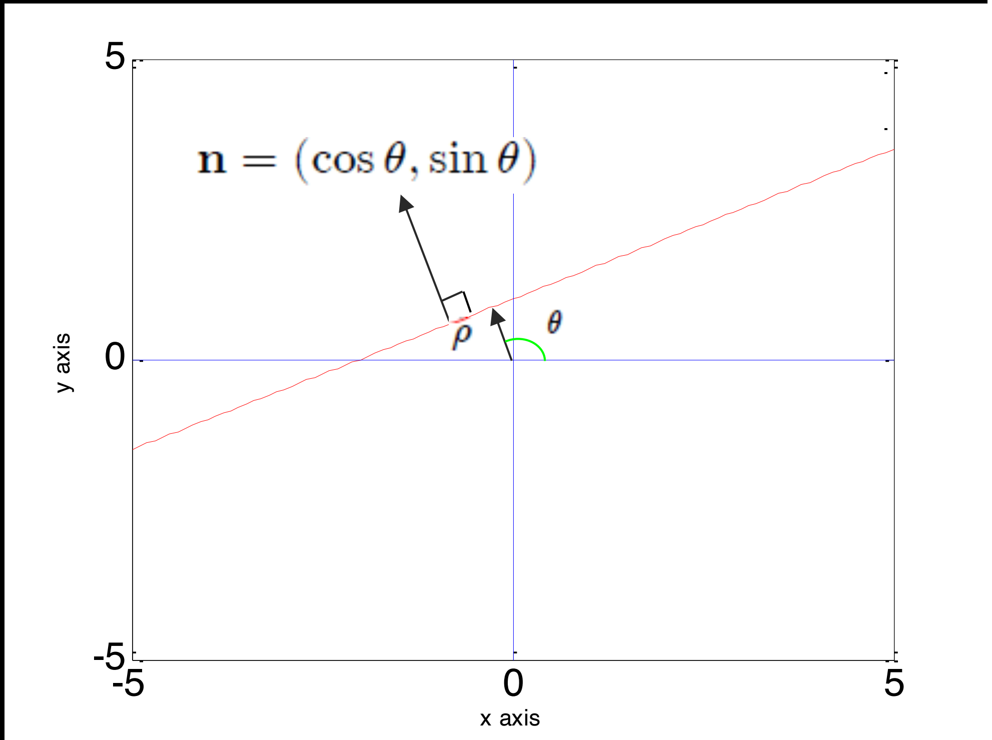
- Normal form parameterisation

$$x\cos\theta + y\sin\theta = \rho$$

- where
  - ρ is the distance from the origin
  - θ is the angle

$$(\cos\theta)^2 + (\sin\theta)^2 = 1$$

$$A^2 + B^2 = 1$$

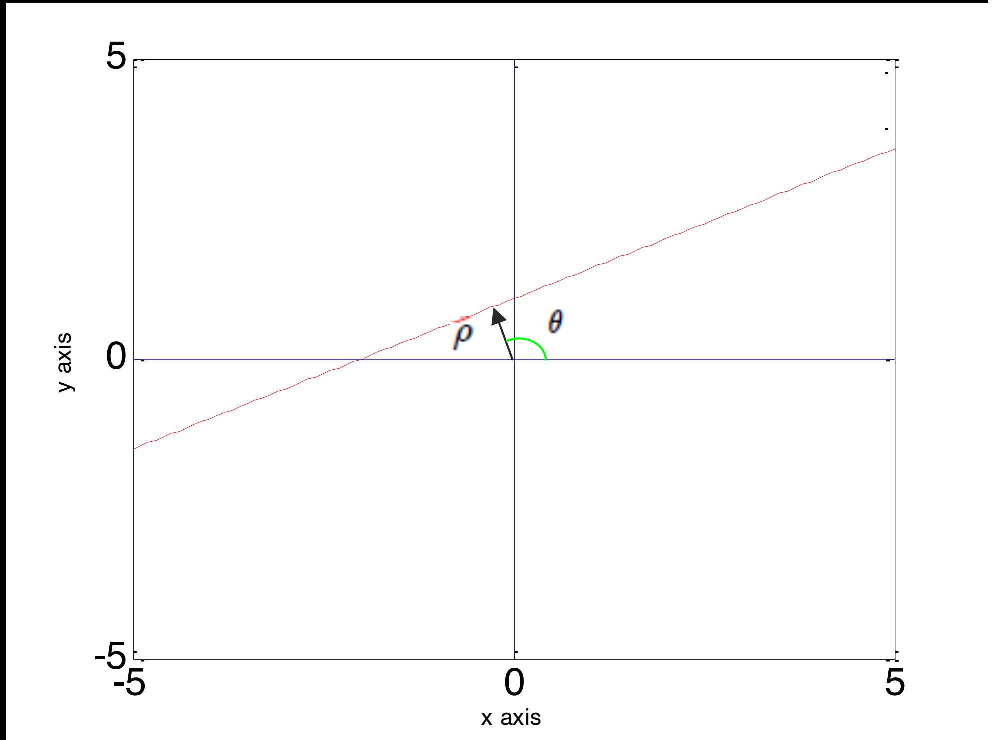


Mathematical line definition

- Normal form parameterisation

$$x\cos\theta + y\sin\theta = \rho$$

- Therefore a line can be defined by two values
  - ρ
  - θ
- A line can therefore also be seen as a *point* in a (θ,ρ)-space



Relation between forms

- From normal from to the slop-intercept form

The normal form: $p = x\cos\theta + y\sin\theta$

The slope-intercept form: $y = ax + b$

Start: $p = x\cos\theta + y\sin\theta$

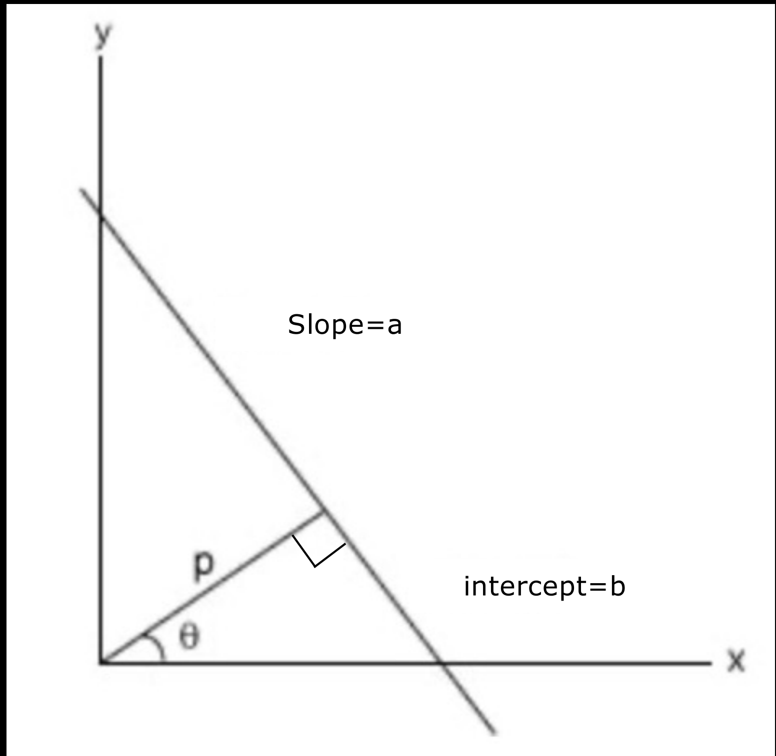
$$-x\cos\theta + p = y\sin\theta$$

$$-x\,cot\theta + p\,cosec\theta = y$$

$$y = x * \boxed{(-cot\theta)} + \boxed{p(cosec\theta)}$$

Slope=a

Intercept=b



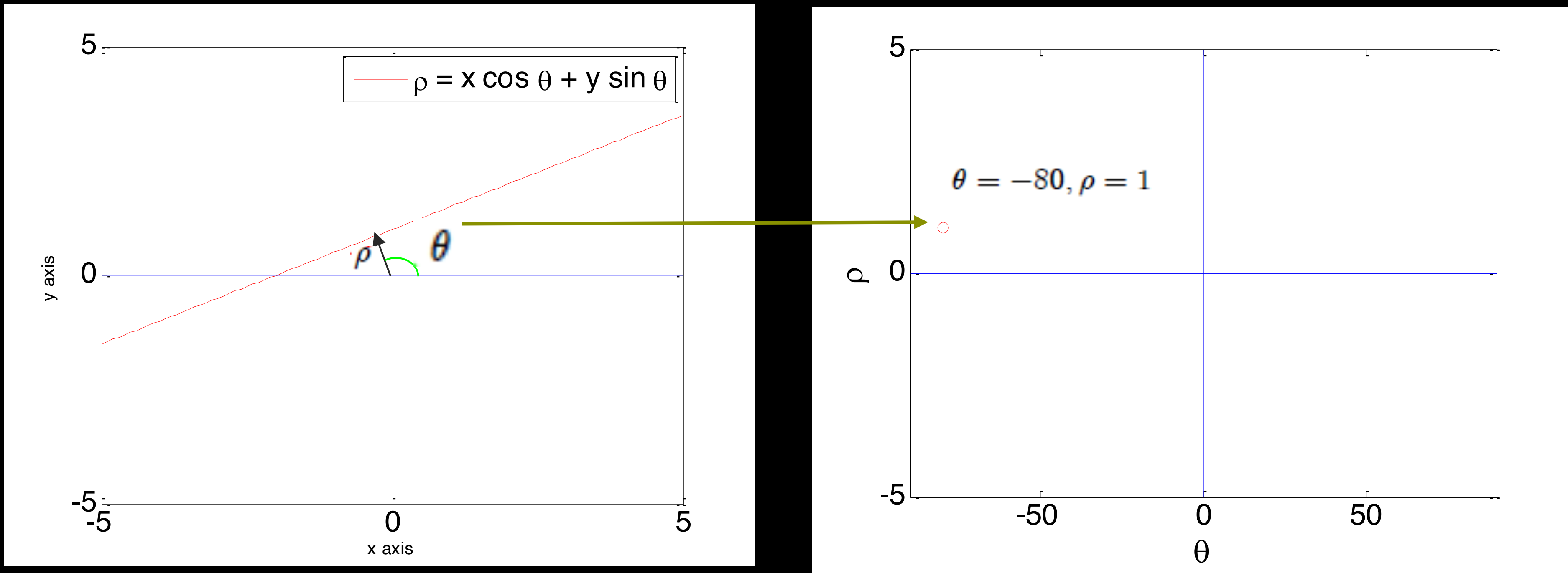
Something about angles

$\theta \in [0^o, 180^o]$ In the course notes

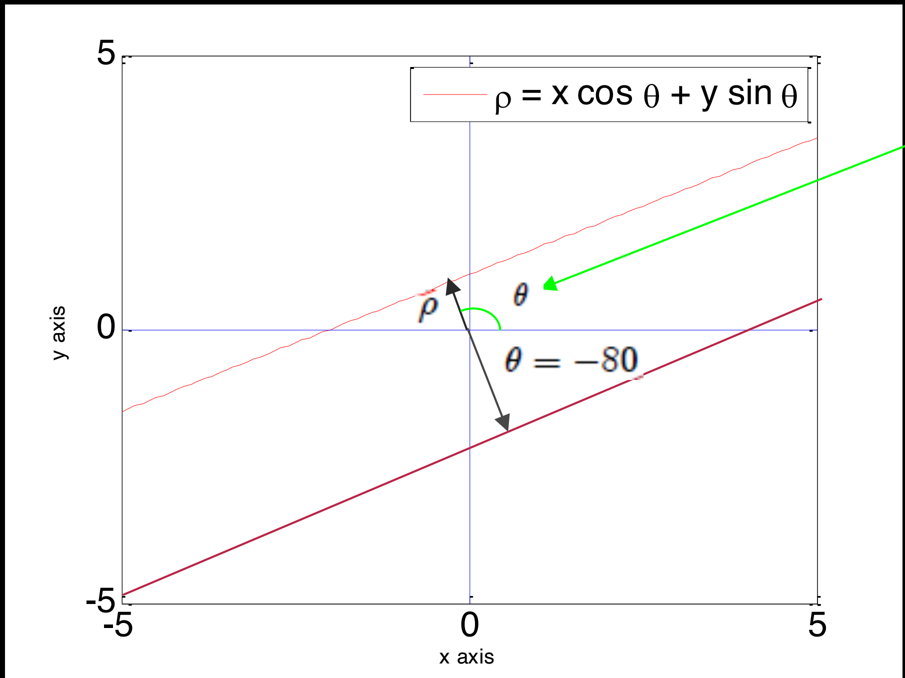
$\theta \in [-90^o, 90^o]$ In Python and in this presentation

Hough Space

$$-90^o < \theta < 90^o$$



More about angles



$$\theta = -80^o$$

Why?

$$\theta = 100^o$$

but Python only allows

$$-90^o < \theta < 90^o$$

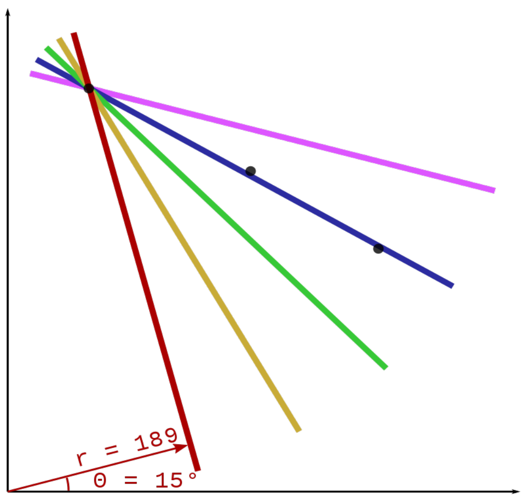
look at the mirror-projection of the normal

ρ is used to determine if it is the "upper" or "lower line"

Hough space: Let's vote for a general line

- A tool to find a line through points.
  1) Define the origin
  2) Select a point coordinate: (x,y)
  3) For different **θ**'s, map a line in the normal form through the selected point
  4) Map each line as parameters in the Hough space: (r,***θ***) $i.e.\ r = x\cos\theta + y\sin\theta$
  5) "Vote" which line fit best through all points: Have similar Hough space parameters

Points



Hough space

θ	r
15	189.0
30	282.0
45	355.7
60	407.3
75	429.4

Quiz 1: Hough space

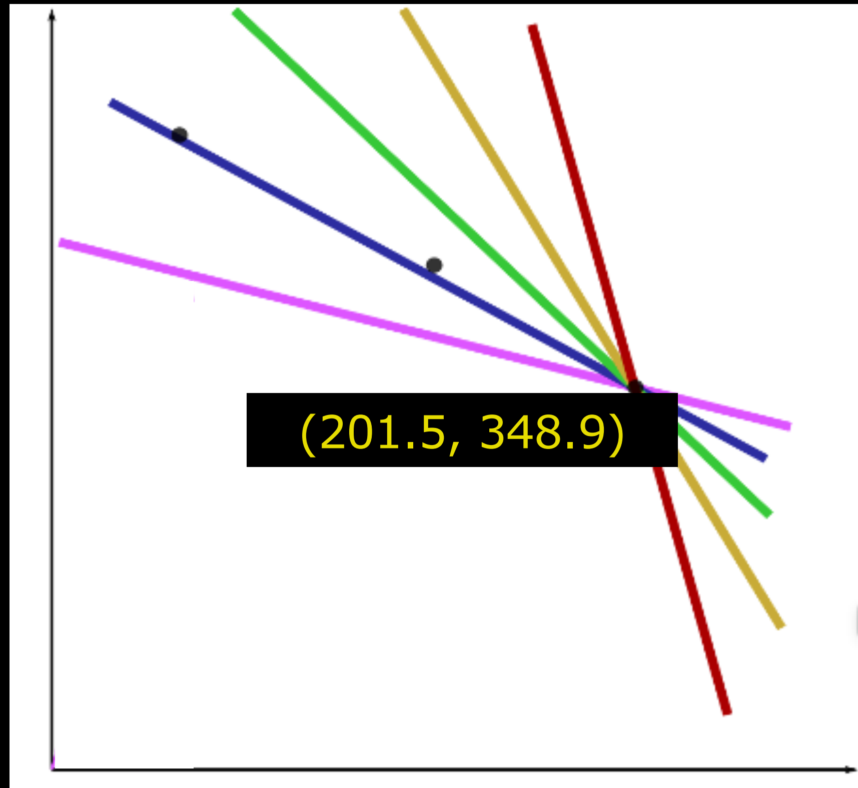
If we select θ to 60 degree what is r when the point is (x,y)=(201.5, 348.9)?

A) 137.1
B) 402.9
C) -25.4
D) 370
E) -298.3

Solution:

$$x \cos\theta + y \sin\theta = r$$

$$201.5 \cos(60 * 0.0175) + 348.9 \sin 60 * 0.0175 = 402.9$$

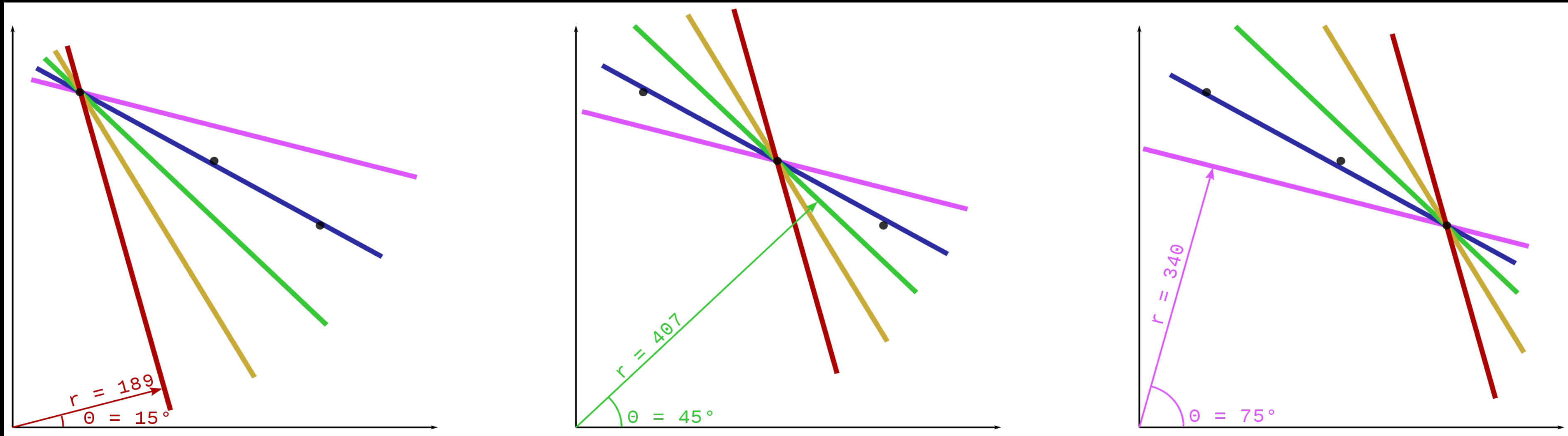


θ	r
15	419.0
30	443.6
45	438.4
60	????
75	340.1

Hough space

- A tool to find a line through points.
  1) Define the origin
  2) Select a point coordinate: (x,y)
  3) For different **θ**'s, map a line in the normal form through the selected point
  4) Map each line as parameters in the Hough space: (r,***θ***) $i.e.\ r = x\cos\theta + y\sin\theta$
  5) "Vote" which line fit best through all points: Have similar Hough space parameters

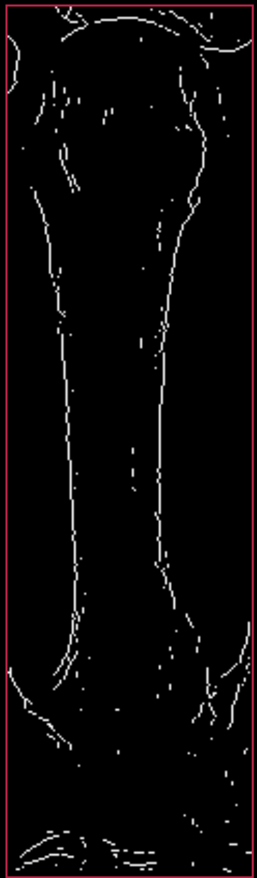
Points



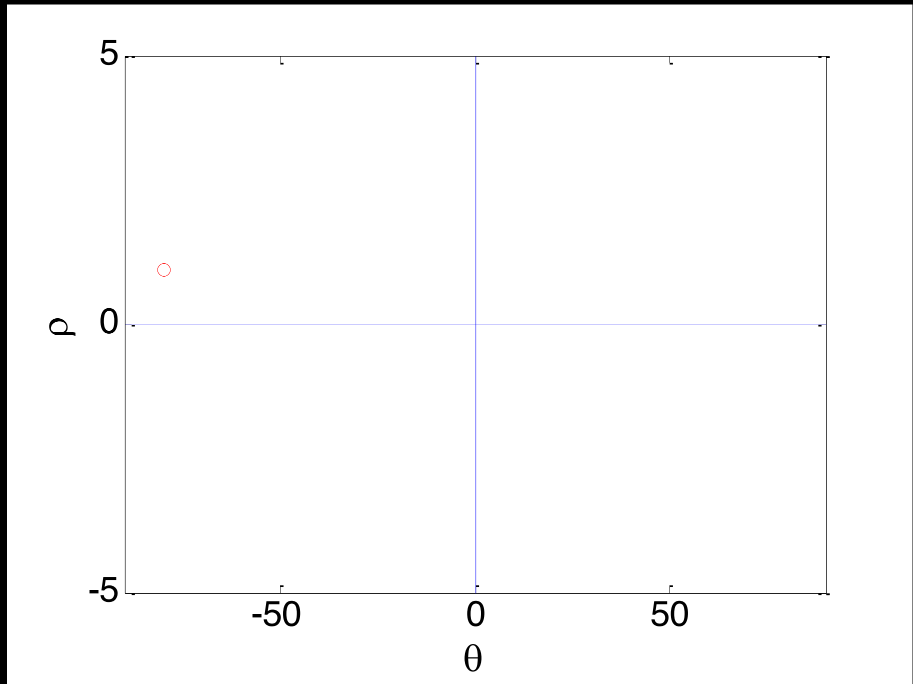
Hough space

θ	r	θ	r	θ	r
15	189.0	15	318.5	15	419.0
30	282.0	30	376.8	30	443.6
45	355.7	45	407.3	45	438.4
60	407.3	60	409.8	60	402.9
75	429.4	75	385.3	75	340.1

How do we use the Hough space?

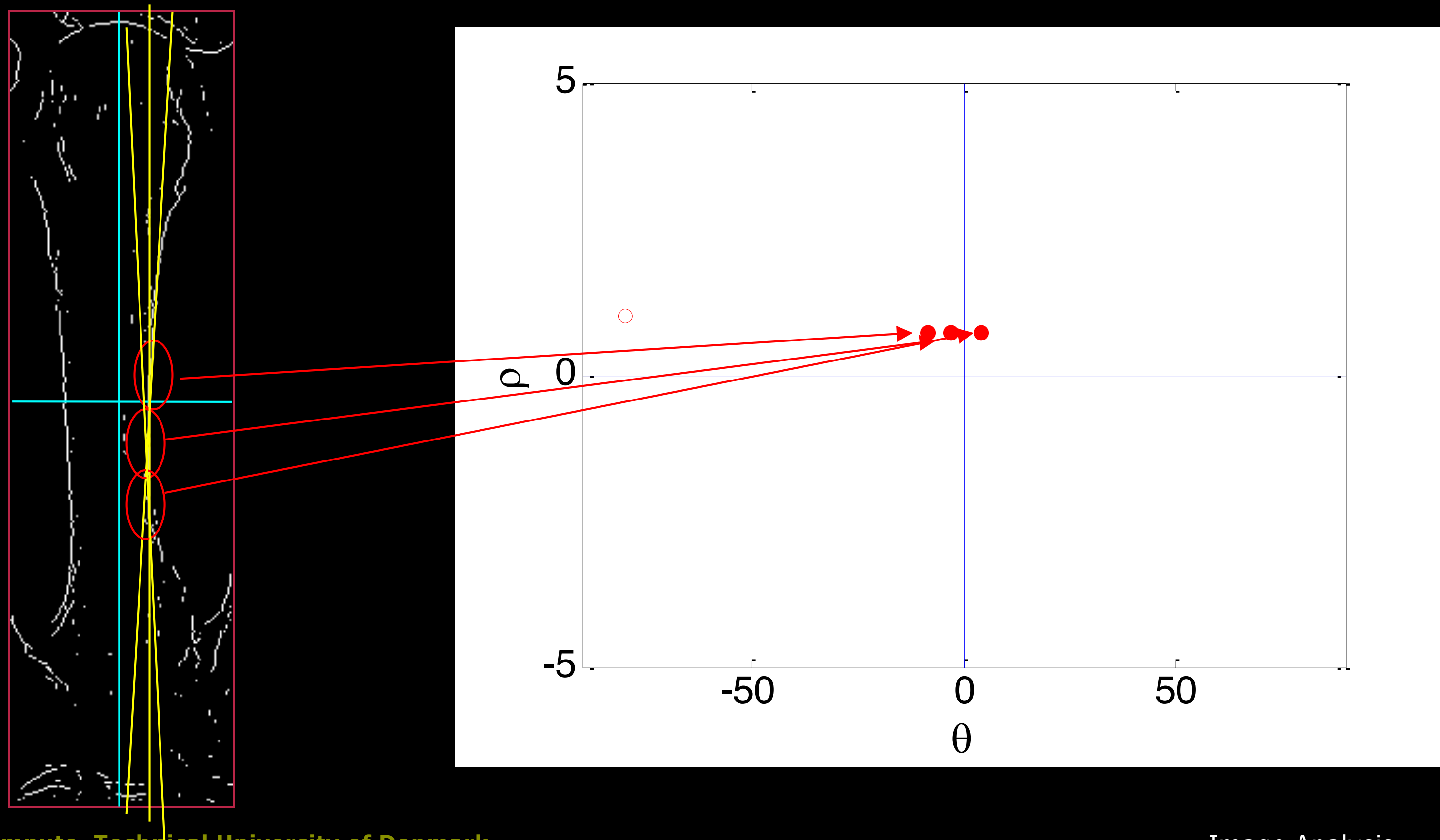


?



How do we use the Hough space?

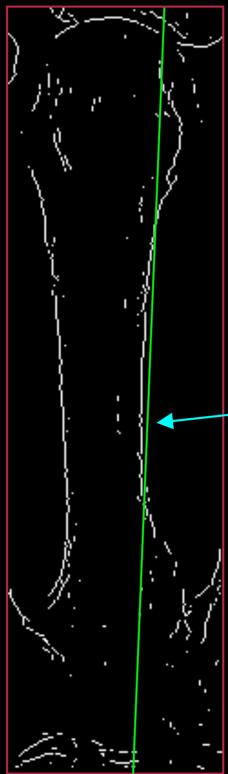
- What if every little “line-segment” was plotted in the Hough-space?

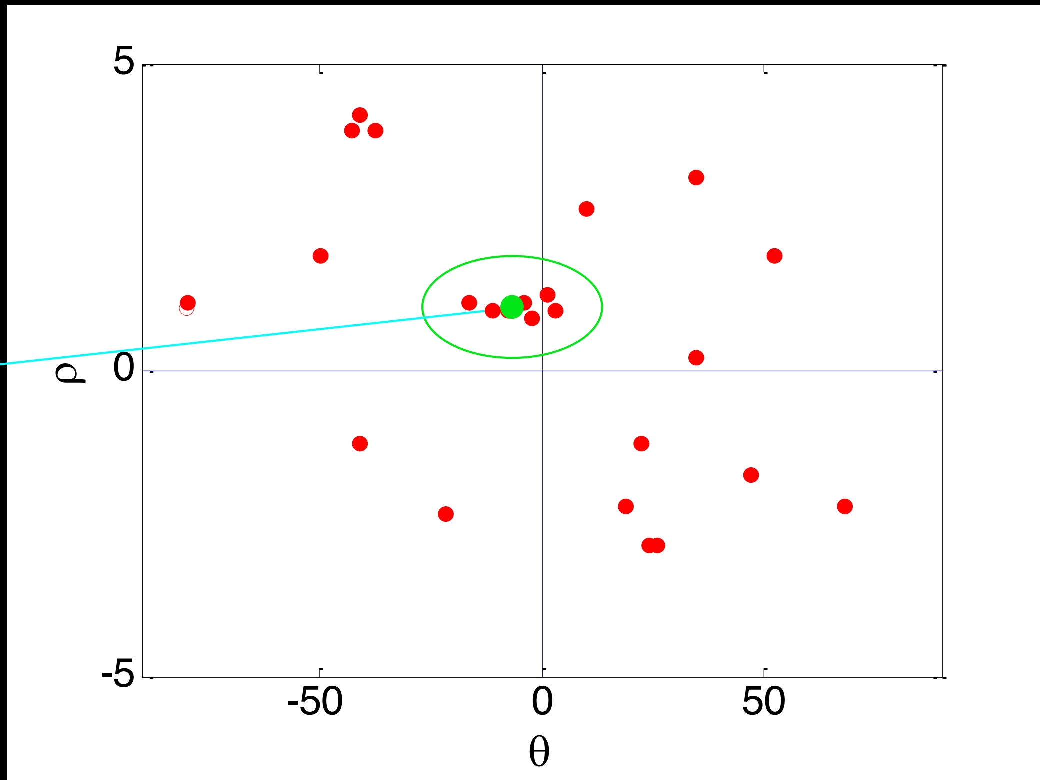


Filled Hough-Space

- All “line segments” in the image examined
- A “global line” can now be found as a cluster of points

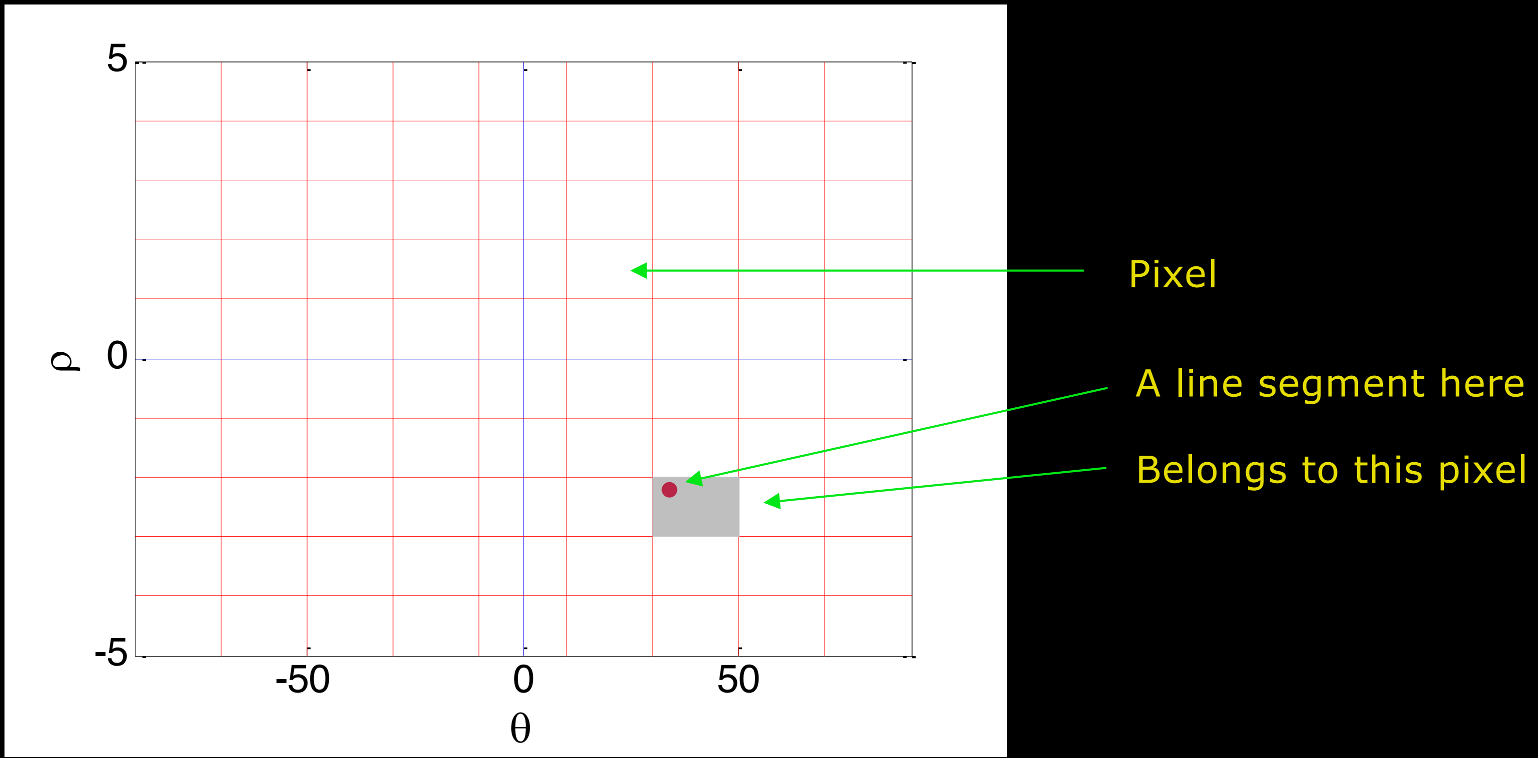
In practice it is difficult to identify clusters





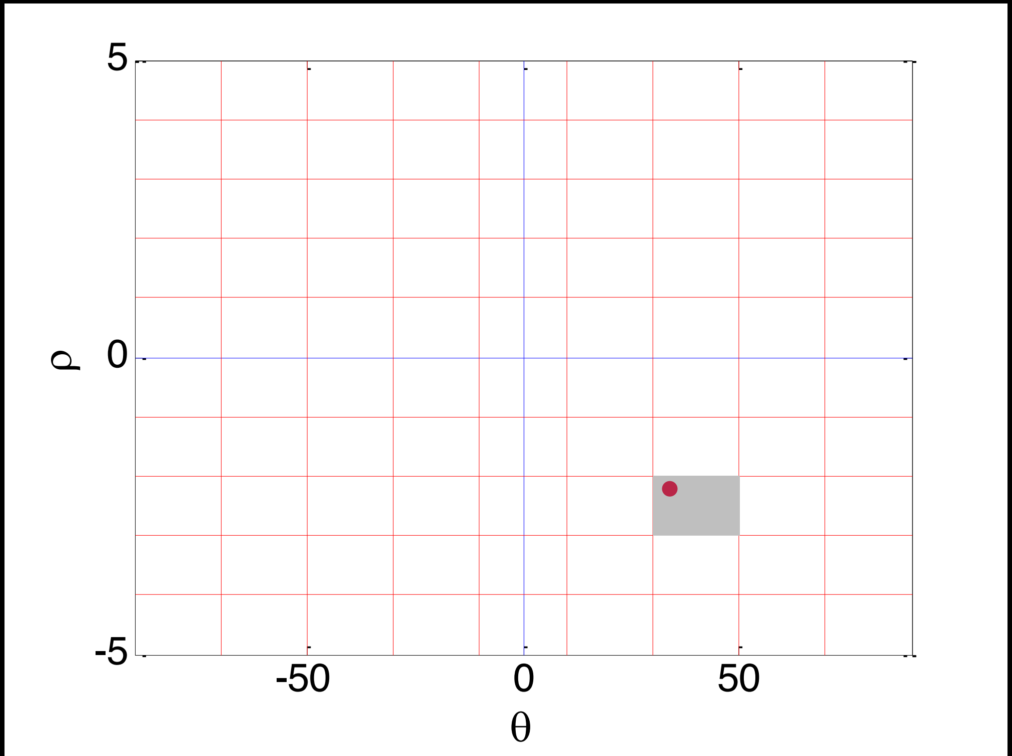
Hough transform in practise

- Hough Space is represented as an image
- It is *quantisized* – made into finite boxes



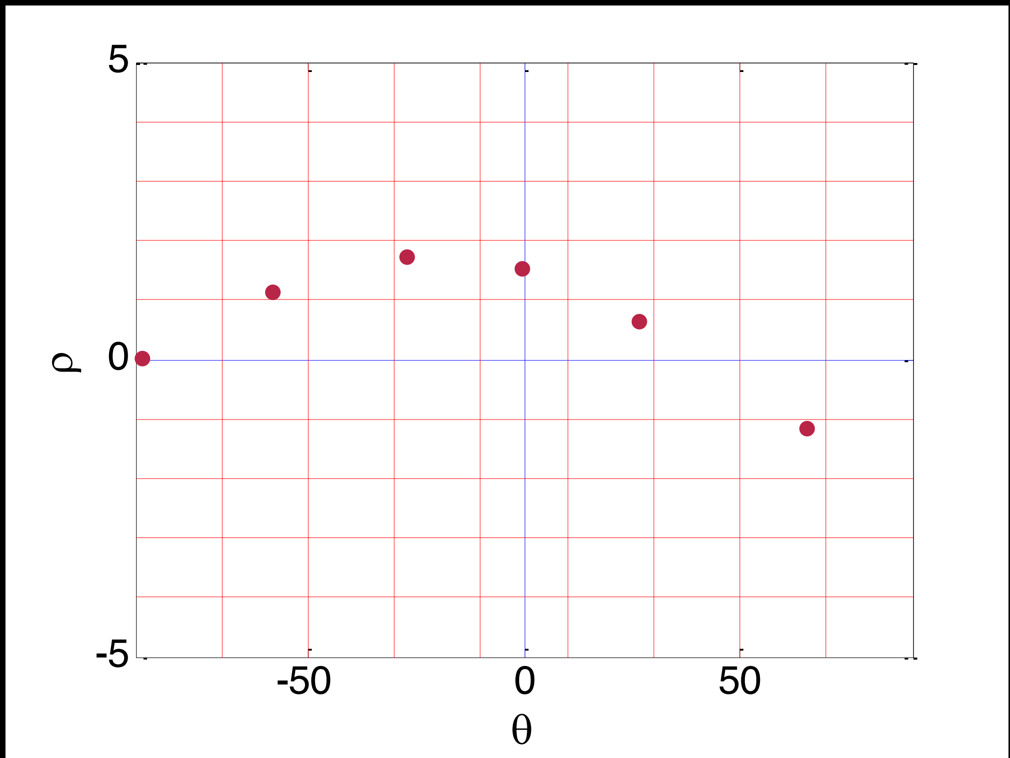
Hough transform as a voting scheme

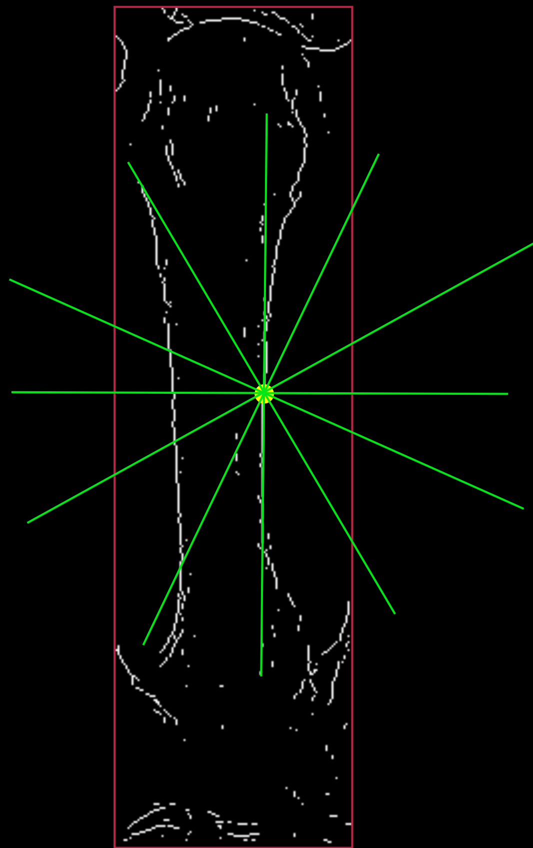
- The pixels in the Hough space are used to *vote* for lines.
- Each *line segment* votes by putting *one vote* in a pixel
- The pixels are also called *accumulator cells*



Hough transform per pixel

- In practise we do not use line segments
- Each pixel in the input image votes for all potential lines going through it.





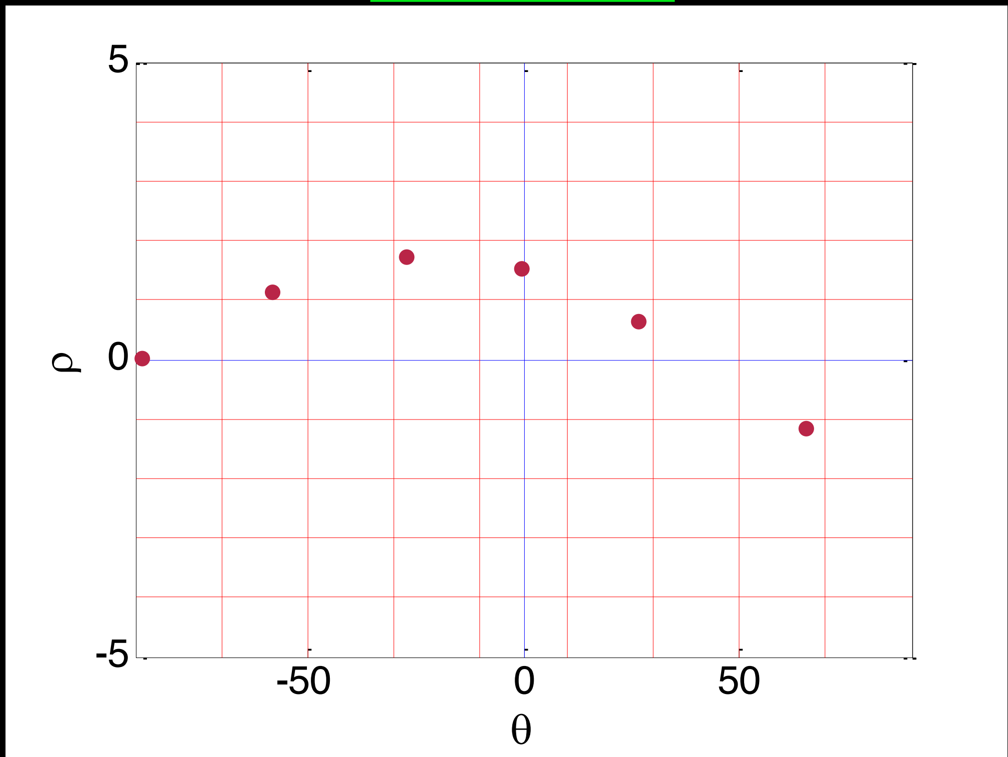
Hough transform per pixel

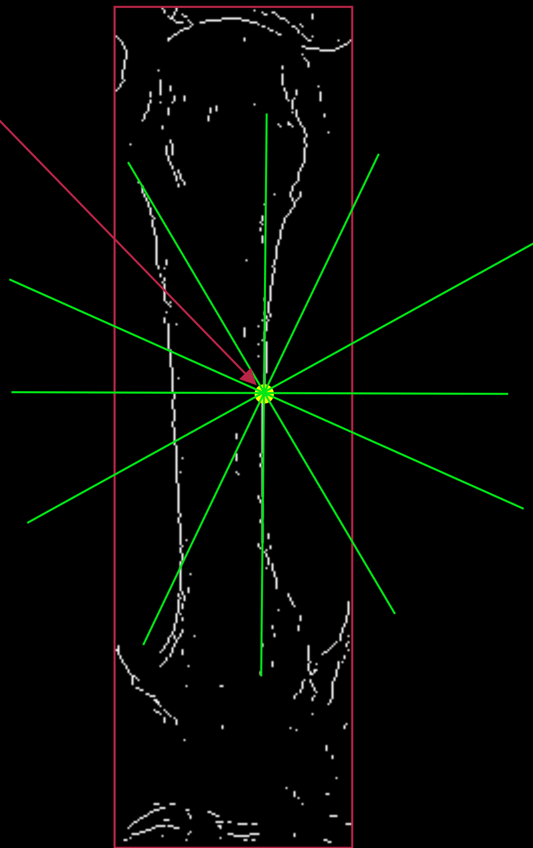
$$x \cos\theta + y \sin\theta = \rho$$

Go through all θ and calculate ρ

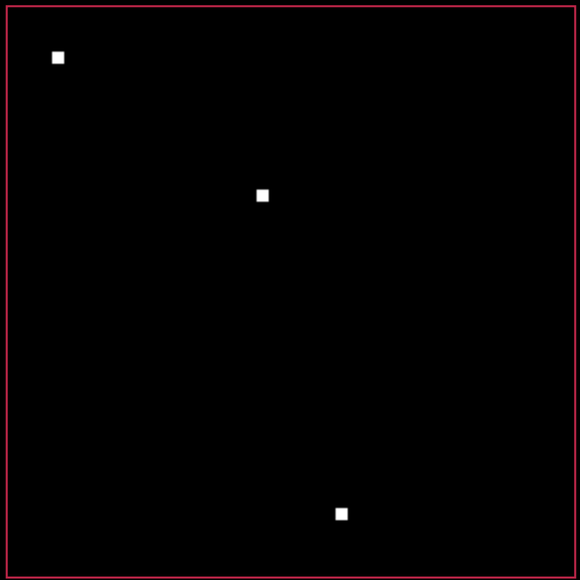
(x, y) are fixed

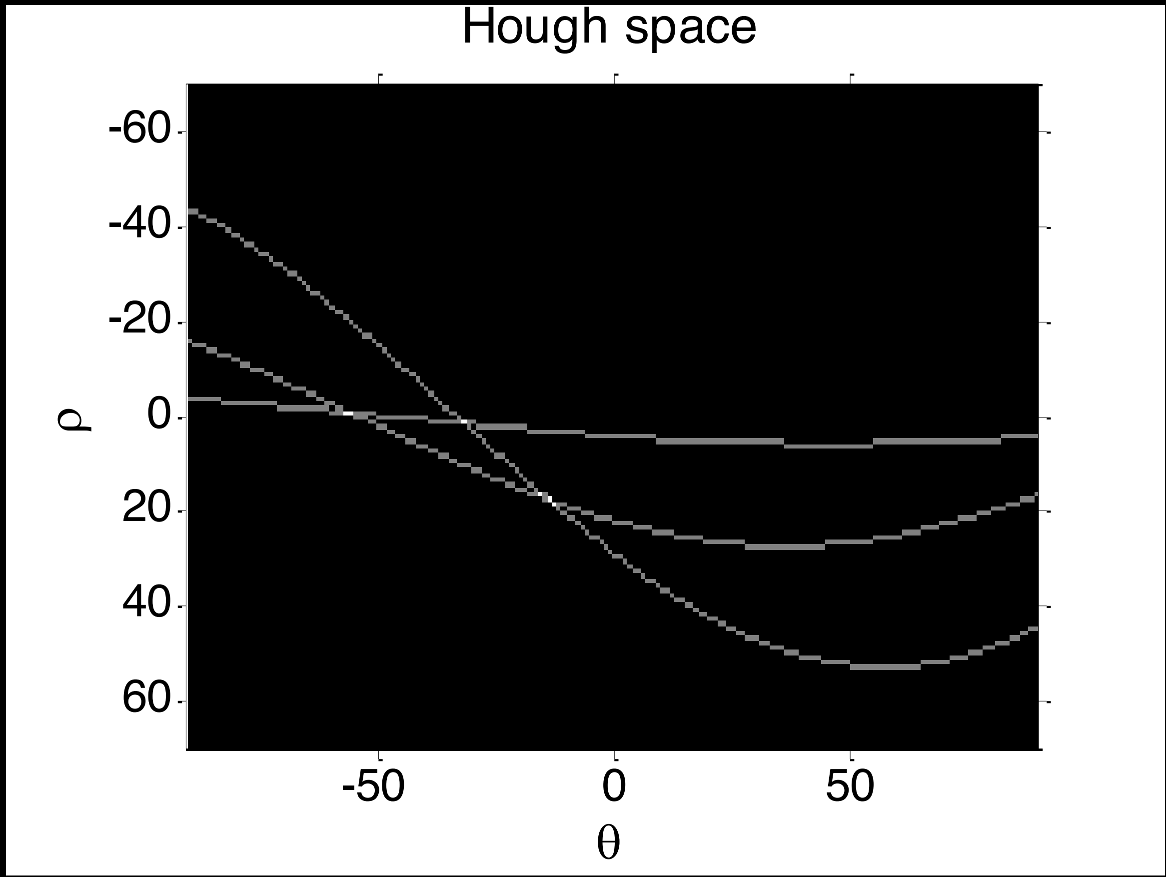
Sinusoid!



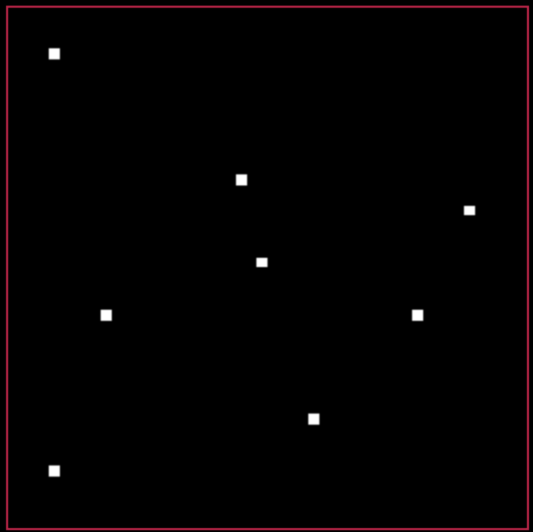


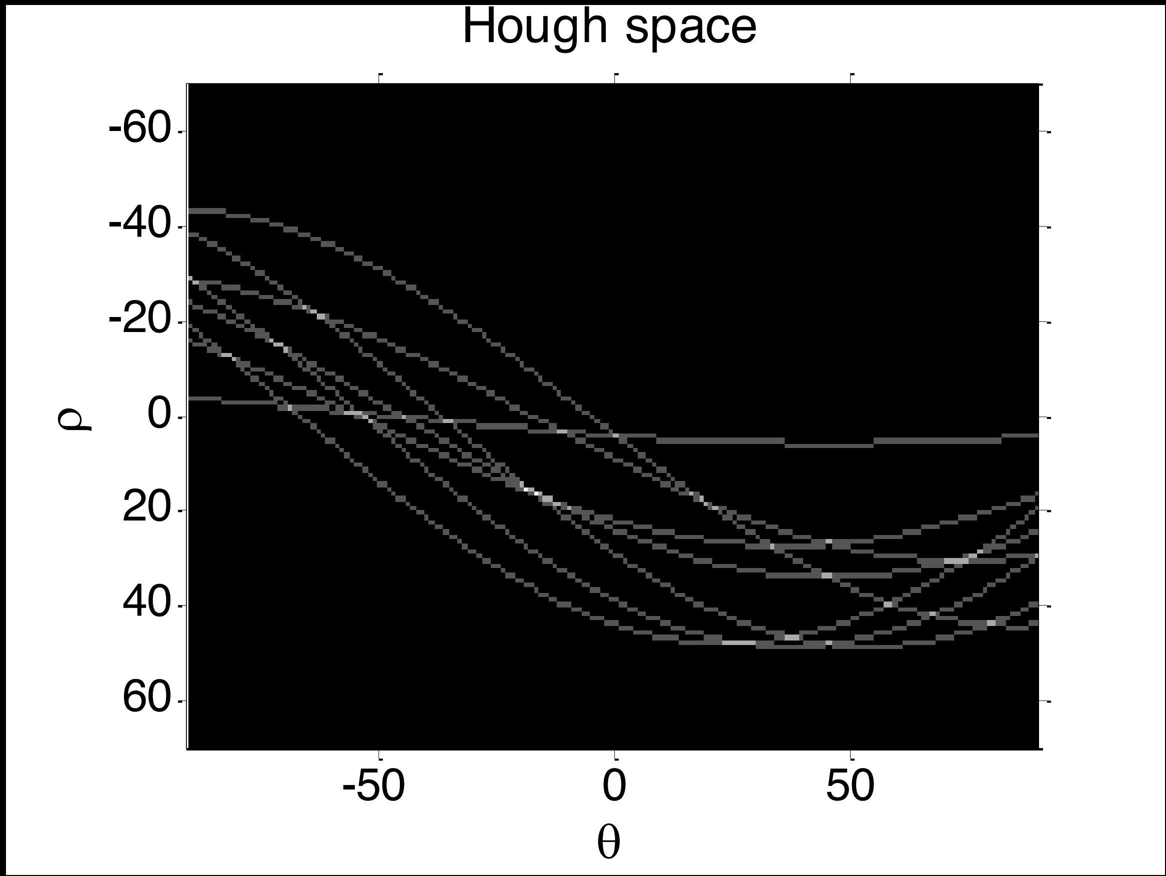
Real Hough Transform





Real Hough Transform II

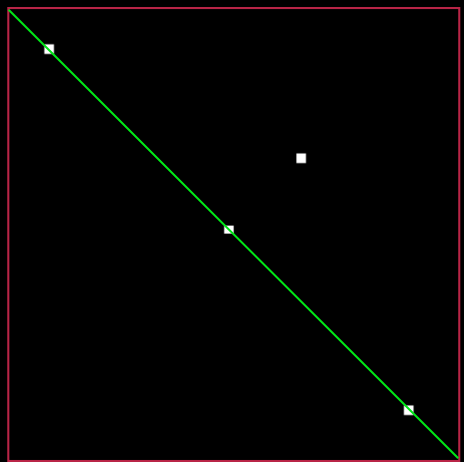


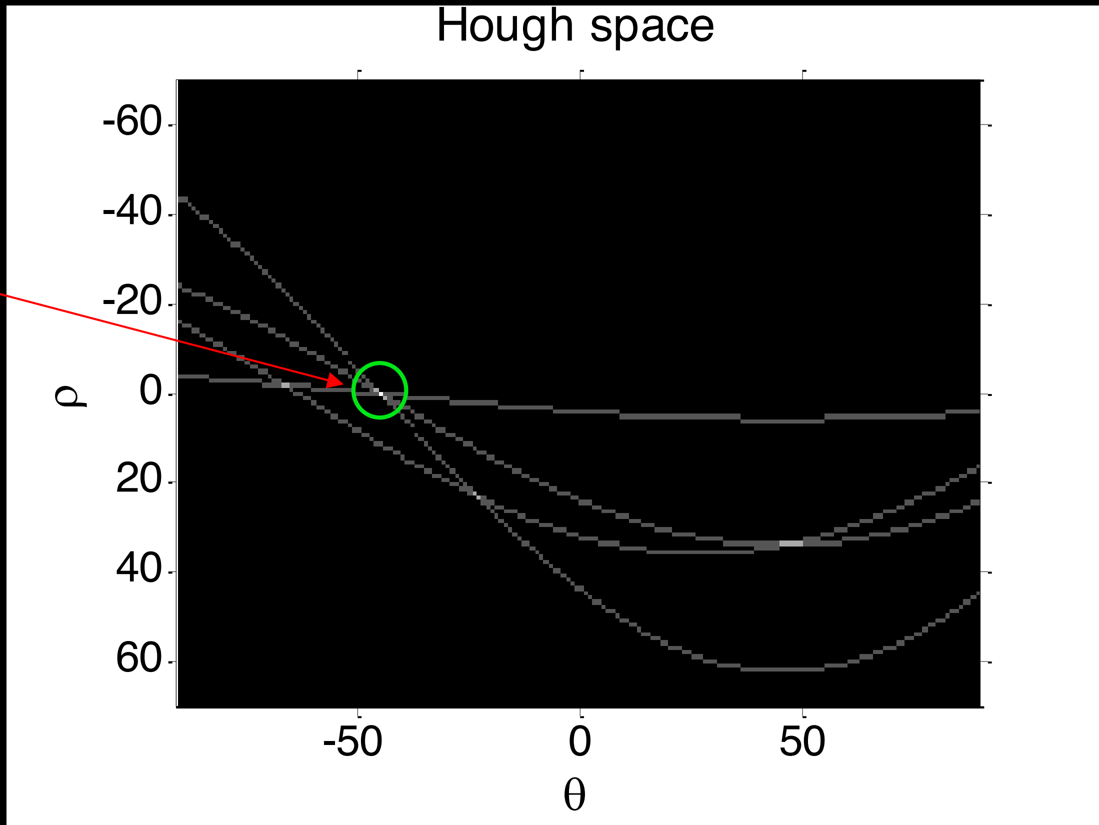


Real Hough Transform and lines

Spot the line!

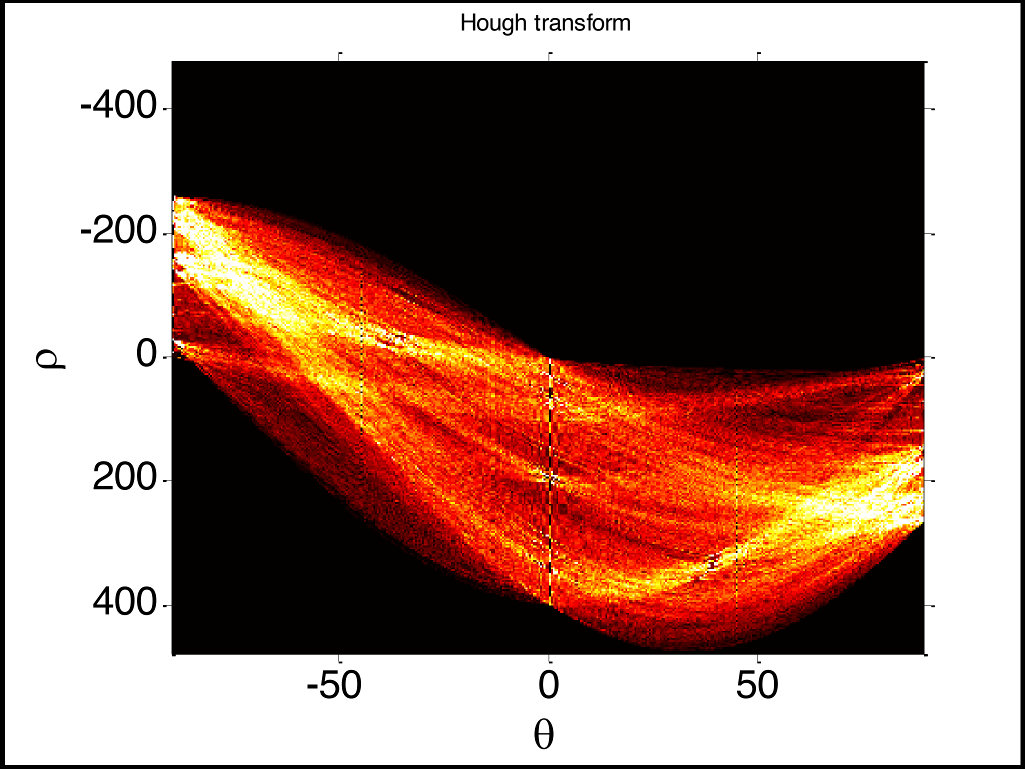
A maximum where Hough pixel has value 3



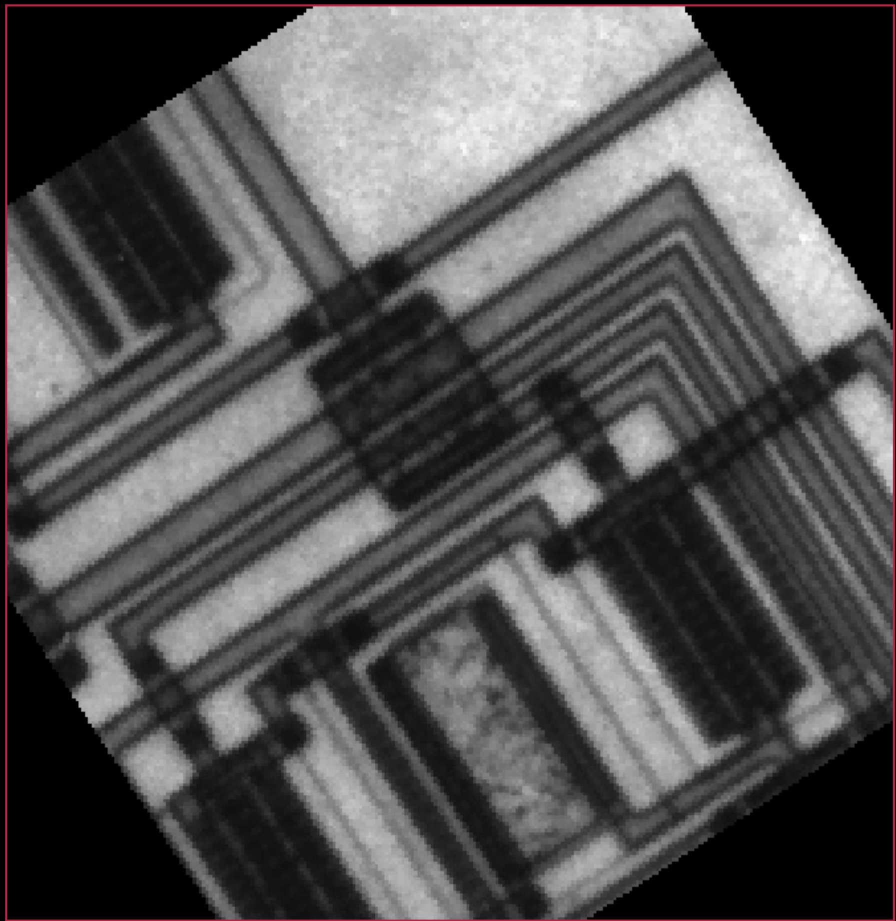


Finding the lines in Hough space

- The lines are found in Hough space where *most pixels have voted for there being a line*
- Can be found by searching for maxima in Hough Space

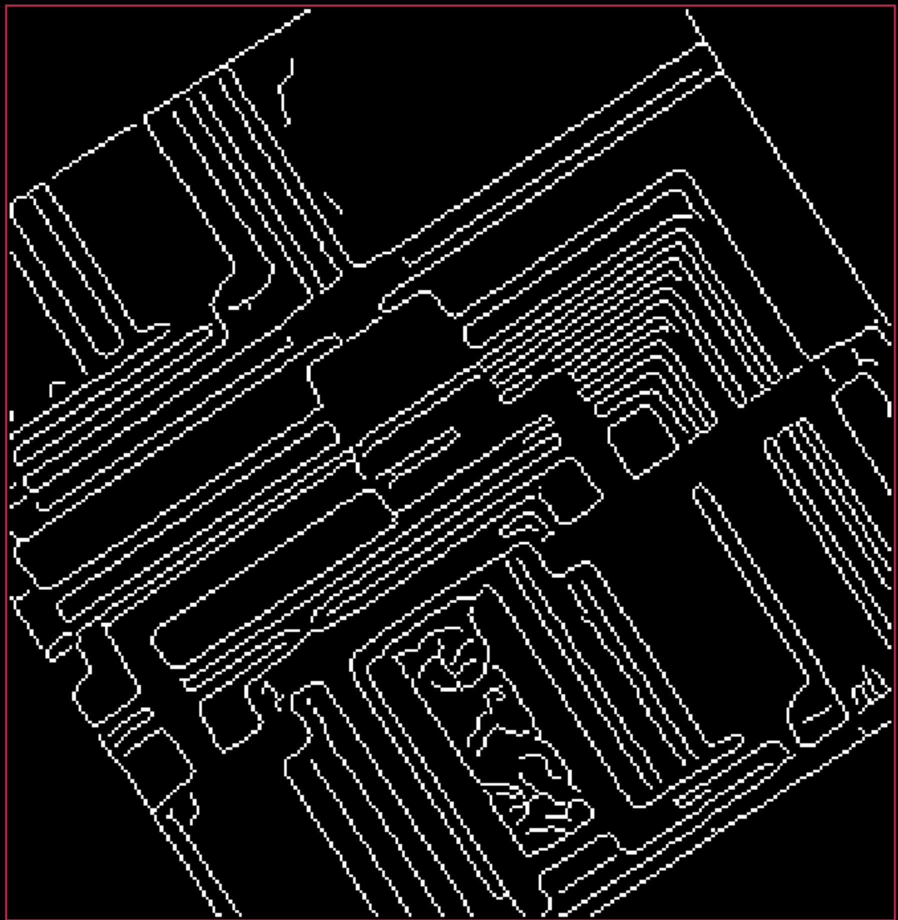


The practical guide to the Hough Transform



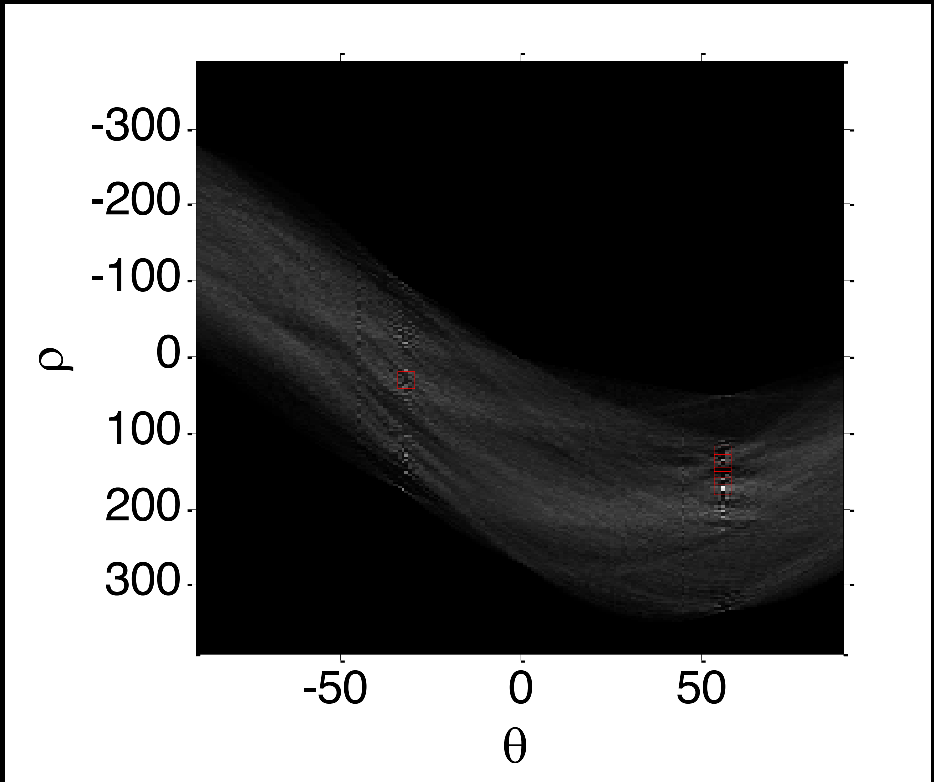
- Start with an input image

The practical guide to the Hough Transform



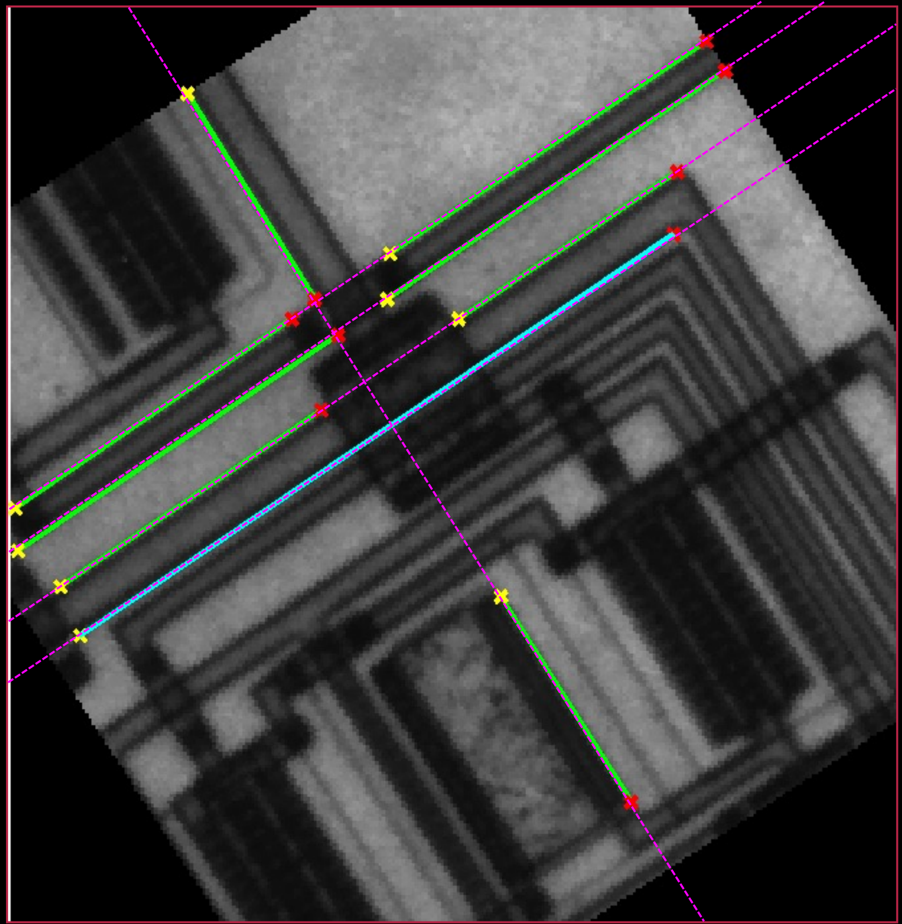
- Detect edges and create a binary image

The practical guide to the Hough Transform



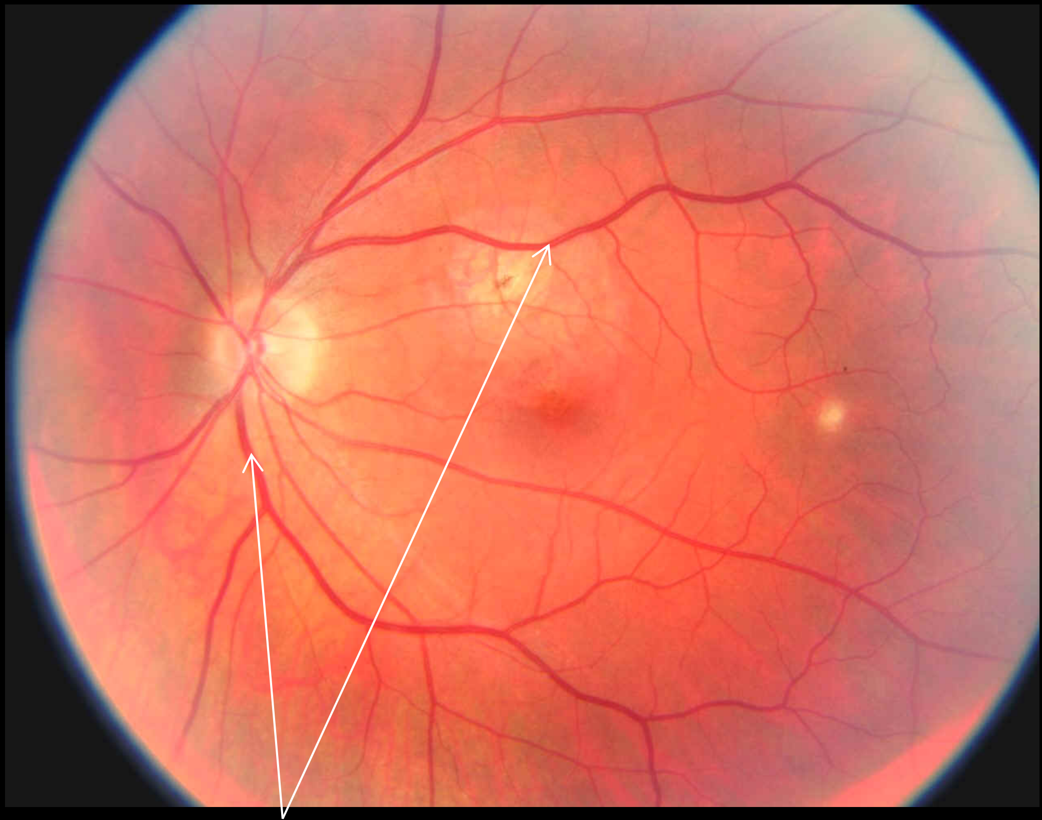
- Compute Hough transform and locate the maxima
- We select the 5 highest points

The practical guide to the Hough Transform



- Draw the 5 lines corresponding to the found maxima (purple)
- The full lines (green)
- Here the cyan line is the longest

Path Tracing

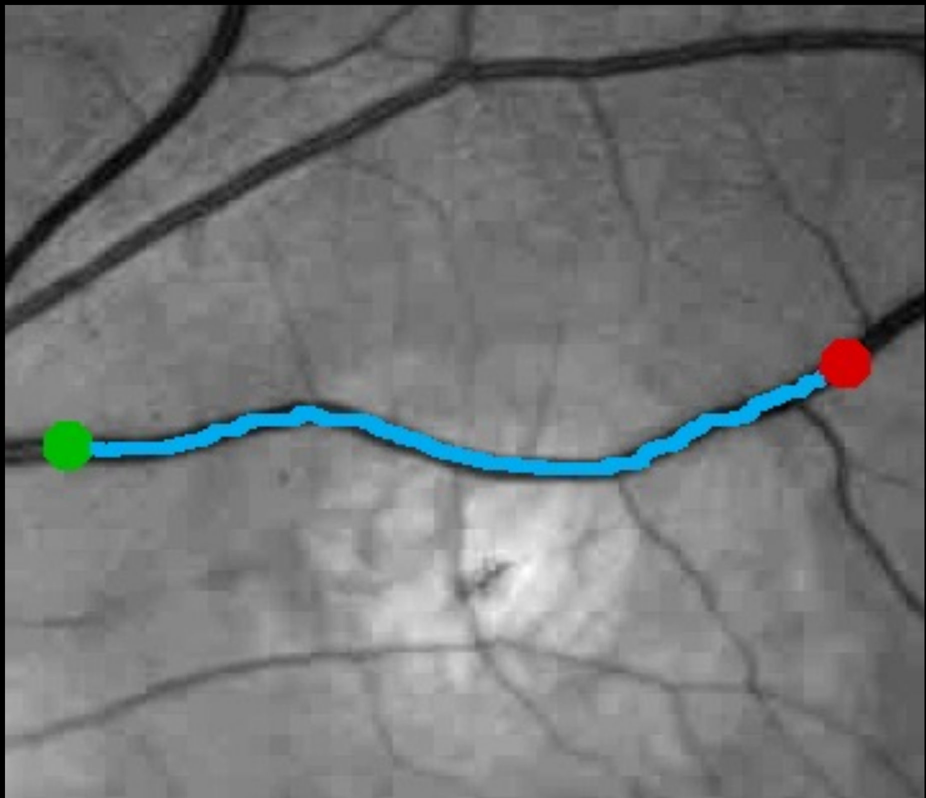


Arteries and veins

Fundus image

- The diameter as function of the distance to the optic cup tells something about the patients health
- We need to find the arteries and veins
- Path tracing is one solution

Path tracing



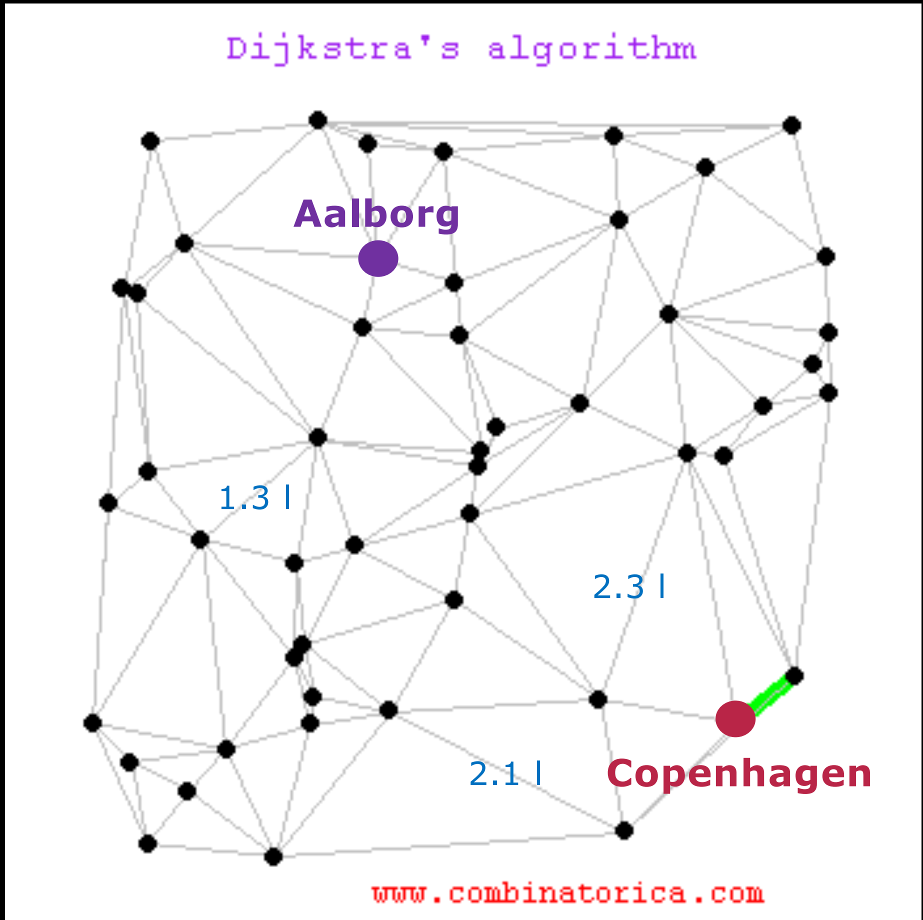
- A path is defined as a curve in an image defined as *something that is different from the background*
- In this case it is a dark line
- Pre-processing can for example turn edges into dark lines.

Dynamic Programming

- Break up large problem into many small sub-problems
- A classic algorithm:
  - Dijkstra’s algorithm
  - One source to all nodes shortest path
- We will look at a simplified variant

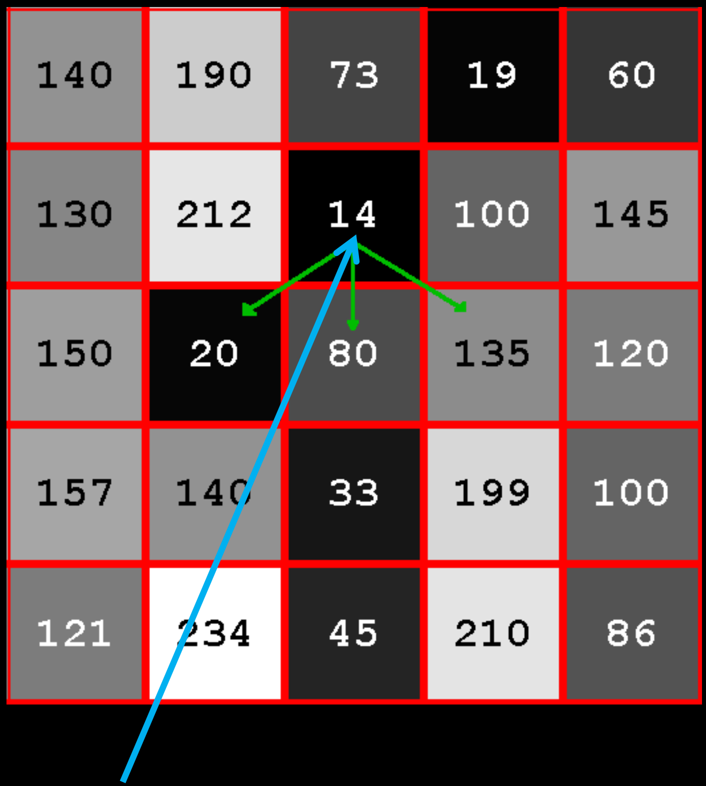
Dijkstra, E. W. (1959). "A note on two problems in connexion with graphs". Numerische Mathematik. 1: 269–271.

Path tracing



- A GPS device uses path tracing
- Based on *graph algorithms*
  - A city is a node
  - A road is an edge. The weight of the edge is the fuel cost
- How do we come from Copenhagen to Aalborg using the least fuel?
- Dijkstra’s algorithm

Images as graphs



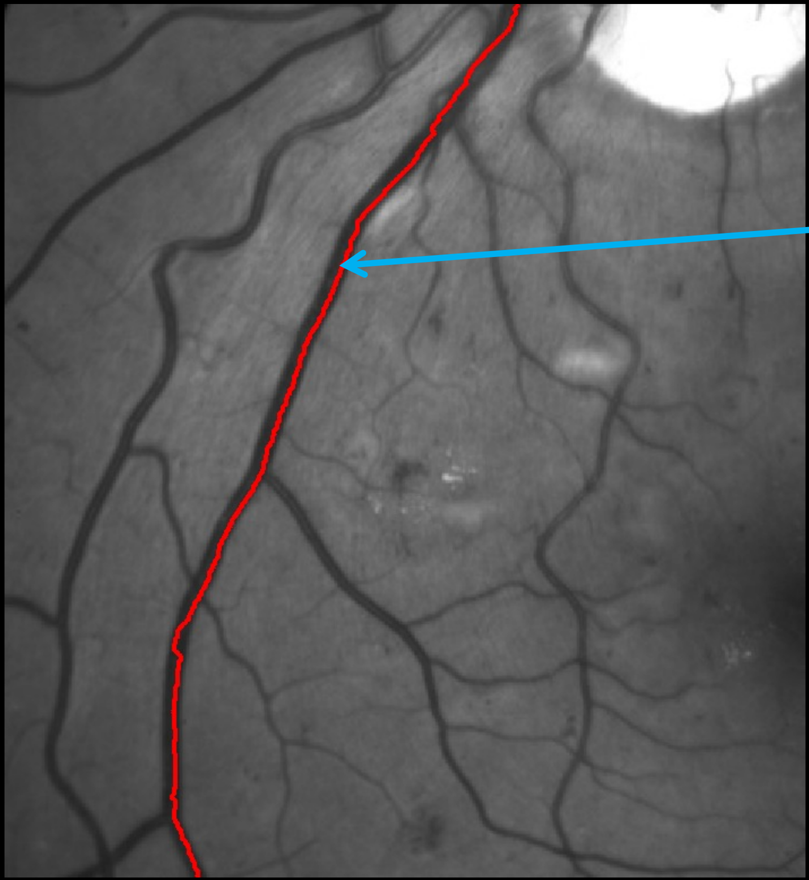
$$\mathcal{C}(2,3) = 14$$

- Each pixel is a node
- Pixel neighbours are connected by edges
- The edge cost ($c(r,c)$)is the pixel value
- Directed graph
- Imagine a car driving on the image
- Called a *cost image*

Simplified problem

- Track dark lines
- Path going from top to bottom
- No sharp turns – smooth
- Problem:
  - from the top to the bottom
  - Sum of pixel values should be minimal

Simplified problem



- Pixel value at (r,c) equals the cost
  $C(r,c)$
- The path P consist of pixels
- The sum of pixel values in the path

$$C_{tot} = \sum_{(r,c)\in P} C(r,c)$$

Path cost

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

P = [(1,3), (2,3), (3, 2), (4,3), (5,4)]

- A path is defined as (r,c) coordinates

$$C_{tot} = \sum_{(r,c)\in P} C(r,c)$$

Quiz 2: Total cost – what is C_{tot}?

A) 167

B) 350

C) 403

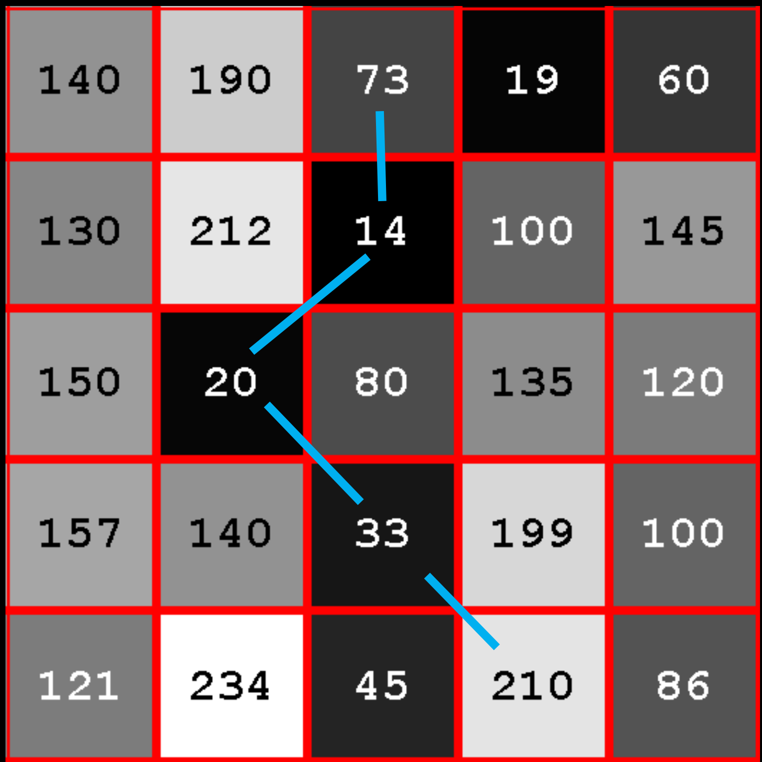
D) 270

E) 345

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

P = [(1,3), (2,3), (3, 2), (4,3), (5,4)]

Path cost



P = [(1,3), (2,3), (3, 2), (4,3), (5,4)]

- This is *NOT* the optimal path
- How do we compute the path *P* that has minimum C_{tot}?

- Test all possible paths?
  - No! Impossible amount of possibilities

Quiz 3: Path Cost

A) 196

B) 154

C) 201

D) 185

E) 132

A path has been found in the image P=[(1,4),(2,4),(3,5),(4,5),(5,5),(6,4)].

What is the total cost of the path?

208	157	234	19	145	79
62	121	73	14	120	135
237	90	193	135	3	42
89	212	192	199	86	154
50	149	97	238	41	67
64	140	145	33	203	167

Figur 1: Grayscale billede

Path restriction: The rules

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

- Path is only allowed to
  - Go down
  - Move one pixel left or right
- Longer jumps not allowed

Accumulator image

140	190	73	19	60
270	285	33	119	164
420	53	113	168	239
210	193	86	312	268
314	320	131	296	354

- Keeps track of the accumulated cost for efficient paths finding
- Path ending here has cost 296
- We use 5 steps to find the shortest path

Computing the accumulator image

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

140	190	73	19	60
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Step 1: Copy first row of input image

Computing the accumulator image

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

140	190	73	19	60
270	285	33	119	164
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Step 2: Fill second row

$$\mathbf{A}(r,c) = \mathbf{I}(r,c) + \min\left(\mathbf{A}(r-1,c-1), \mathbf{A}(r-1,c), \mathbf{A}(r-1,c+1)\right)$$

Computing the accumulator image

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

140	190	73	19	60
270	285	33	119	164
420	53	113	168	239
210	193	86	312	268
314	320	131	296	354

Step 3: Fill all rows by looking at the previous row

$$\mathbf{A}(r,c) = \mathbf{I}(r,c) + \min\left(\mathbf{A}(r-1,c-1), \mathbf{A}(r-1,c), \mathbf{A}(r-1,c+1)\right)$$

Quiz 4: Accumulator Image

A) 57

B) 167

C) 301

D) 241

E) 145

An optimal path has been found in the image. What is the value of the accumulator image in the marked pixel?

117	163	74	210
223	244	171	57
132	61	110	170
241	172	17	215

Using the accumulator image

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

140	190	73	19	60
270	285	33	119	164
420	53	113	168	239
210	193	86	312	268
314	320	131	296	354

Step 4: The end of the optimal path can now be found

The backtracing image

140	190	73	19	60
270	285	33	119	164
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

0	0	0	0	0
1	3	4	4	4
1	3	3	3	4
2	2	2	3	4
2	3	3	3	5

- Keeps track of where the path *came* from
- Each pixel stores the column number

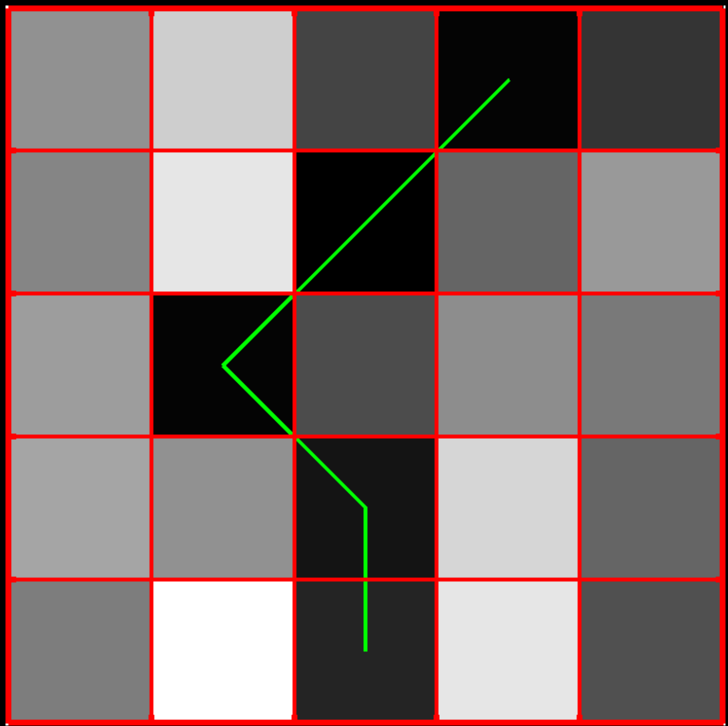
Using the backtracing image

140	190	73	19	60
270	285	33	119	164
420	53	113	168	239
210	193	86	312	268
314	320	131	296	354

0	0	0	0	0
1	3	4	4	4
1	3	3	3	4
2	2	2	3	4
2	3	3	3	5

Step 5: Trace the path in the backtracing image

Using the backtracing image



0	0	0	0	0
1	3	4	4	4
1	3	3	3	4
2	2	2	3	4
2	3	3	3	5

Quiz 5: Backtracing

A) 1

B) 2

C) 3

D) 4

E) 5

An optimal path has been found in an image. The backtracing image is seen below and the optimal path ends in the marked pixel.

What is the optimal path?

0	0	0	0	0
1	3	3	3	5
2	2	2	4	4
1	1	4	5	5
1	1	4	4	4

1. $\mathcal{P} = [(1,3),(2,2),(3,1),(4,1),(5,2)]$
2. $\mathcal{P} = [(1,3),(2,2),(3,2),(4,2),(5,2)]$
3. $\mathcal{P} = [(1,2),(2,2),(3,2),(4,1),(5,2)]$
4. $\mathcal{P} = [(1,3),(2,1),(3,1),(4,1),(5,2)]$
5. $\mathcal{P} = [(1,2),(2,1),(3,1),(4,2),(5,2)]$

Pre-processing

- We would like to track paths that are not dark curves

Quiz 6 : X-ray preprocessing

A) Gaussian smoothing

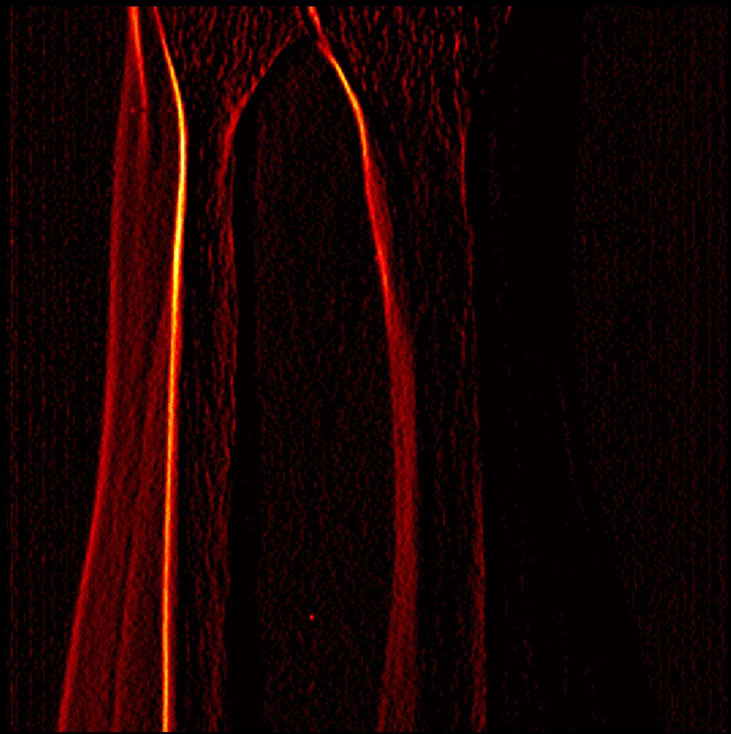
B) 255-I

C) Gradient filter

D) Registration

E) Morphological operation

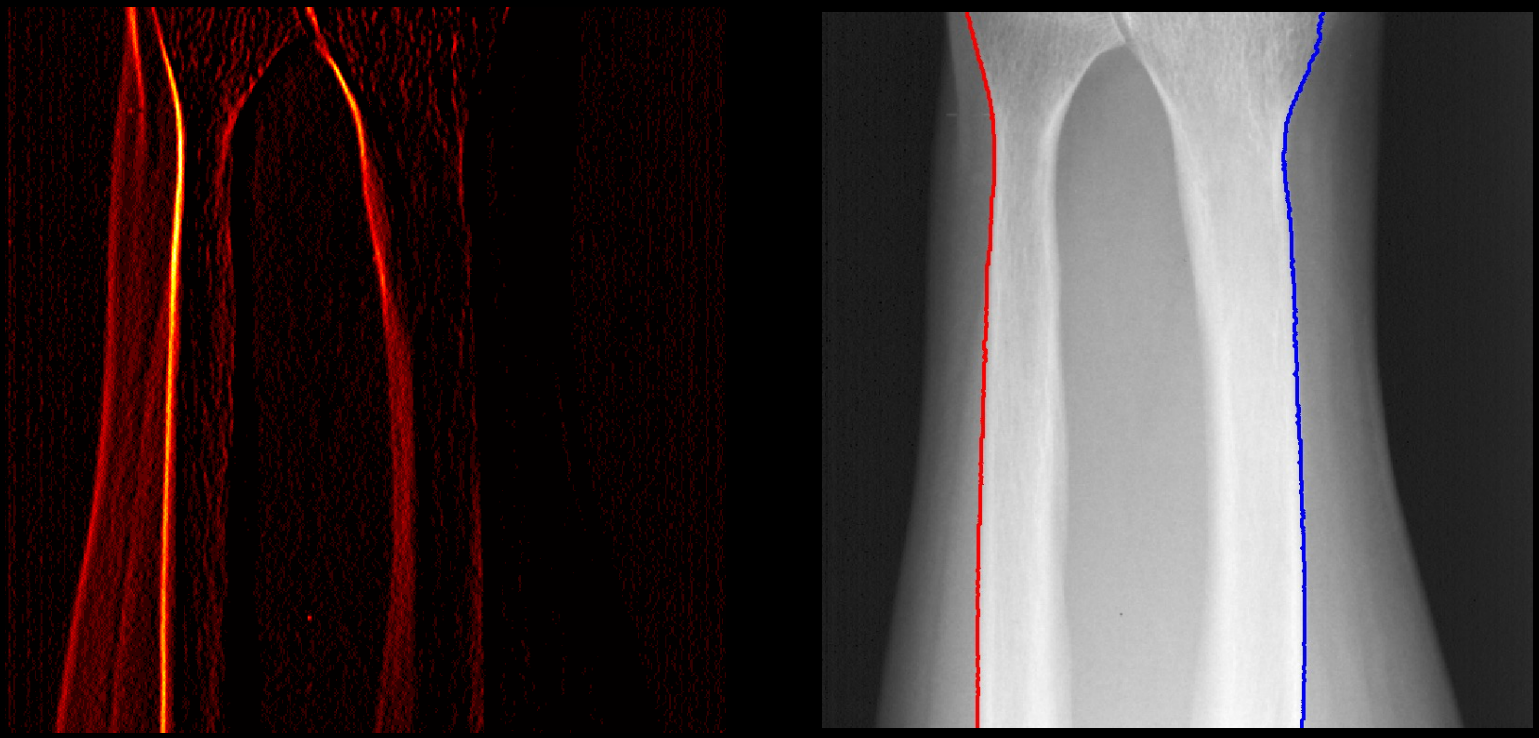
Pre-processing



Edge filtered image

(Gaussian smoothing followed by Prewitt)

Path tracing on pre-processed image



Paths found on pre-processed image and intensity inverted

Quiz 7: Optimal Path 2

A 5 x 5 image is filled with values given the gray level run length encoding: 2, 180, 1, 15, 3, 112, 1, 8, 4, 177, 1, 20, 4, 195, 1, 12, 3, 242, 2, 25, 3, 9. After that the optimal path is found. What is the total cost?

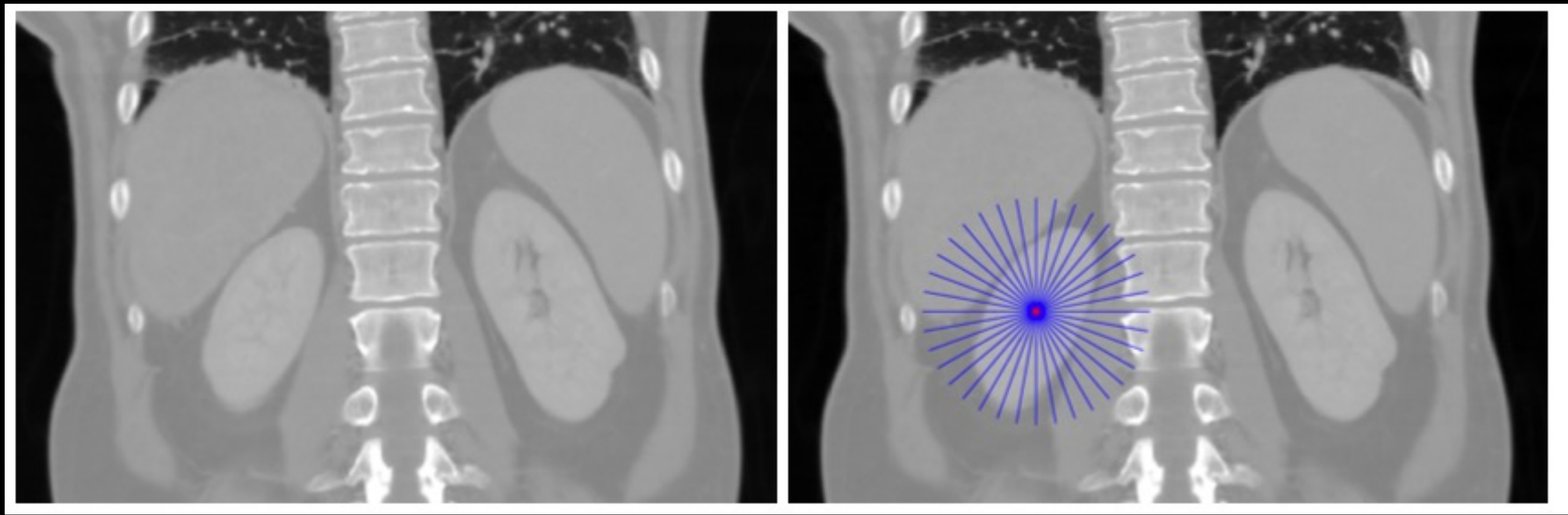
A) 81
B) 64
C) 11
D) 73
E) 51

Solution:

15+8+20+12+9=64

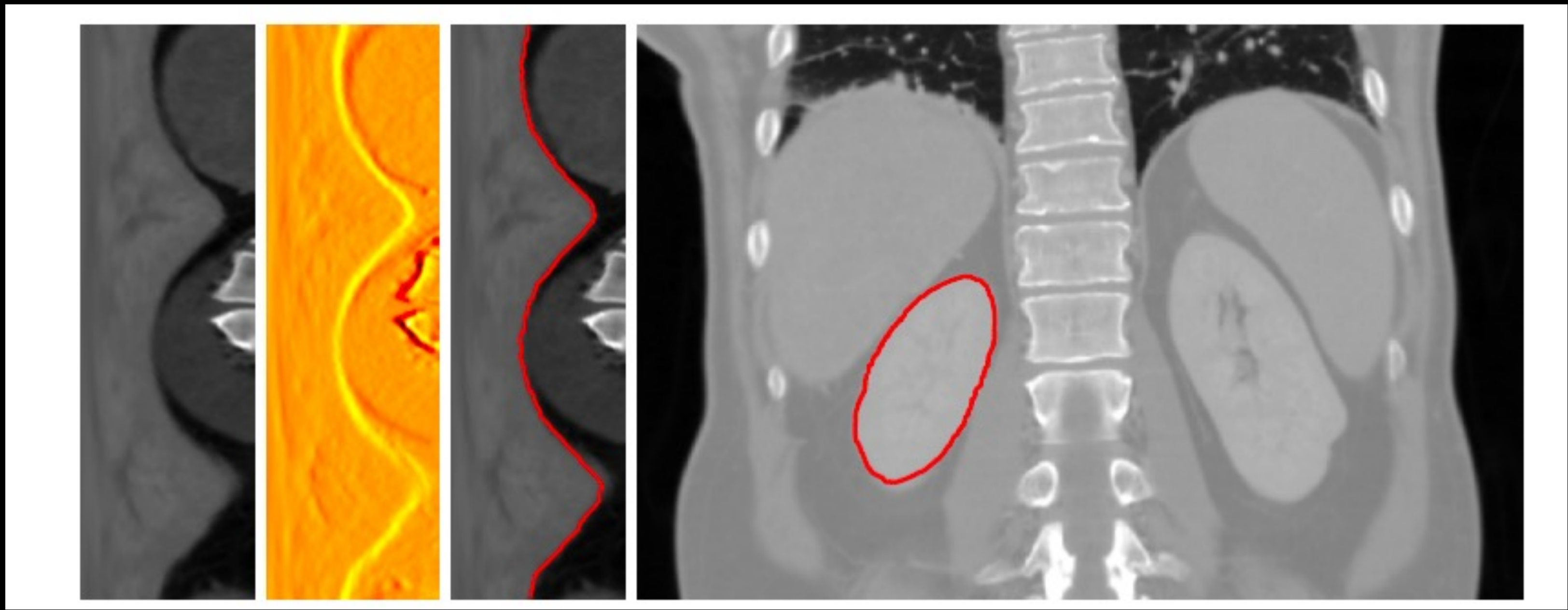
180	180	15	112	112
112	8	177	177	177
177	20	195	195	195
195	12	242	242	242
25	25	9	9	9

Locating Circular Structures



- Define origin inside structure
- Send out spokes

Locating Circular Structures



- Each spoke is a line in a new image (surface- layer detection)
- Prewitt
- Dijkstra’s algorithm
- Map back the spokes into image

What did you learn today?

- Use the Hough transform for line detection
- Describe the slope-intercept, the general form and the normalised form of lines
- Describe the connection between lines and the Hough space
- Use edge detection to enhance images for use with the Hough transform
- Use dynamic programming to trace paths in images
- Describe how an image can be used as a graph
- Describe the fundamental properties of a cost image
- Compute the cost of path
- Compute an accumulator image for path tracing
- Compute a back tracing image for path tracing
- Choose appropriate pre-processing steps for path tracing
- Describe how circular structures can be located using path tracing

Lecture 9 – Industry presentations

JLIVision
FOSS Analytics
Dalux
Videometer
IHfood
TrackMan
Novo Nordisk
Radiobotics
Visiopharm
Claas E-systems