

Summary of previous lecture

- Gaussian filters for:
 - Smoothing
 - Derivatives
 - Laplace operator
- Non-linear filters for edge-preserving smoothing
- Smoothing filters for:
 - Noise reduction
 - Image simplification
 - Shading correction
 - Edge sharpening
- First order derivatives used for edge detection
- Second order derivatives used for line detection

Today's lecture: filtering II

- Filtering can be used for analysis & detection
- Special Applications of Filtering:
 - Scale spaces
 - Gabor filter (for pattern detection)
 - Canny's edge detector
 - Hough transform (for line detection)
 - Radon transform
 - Template matching

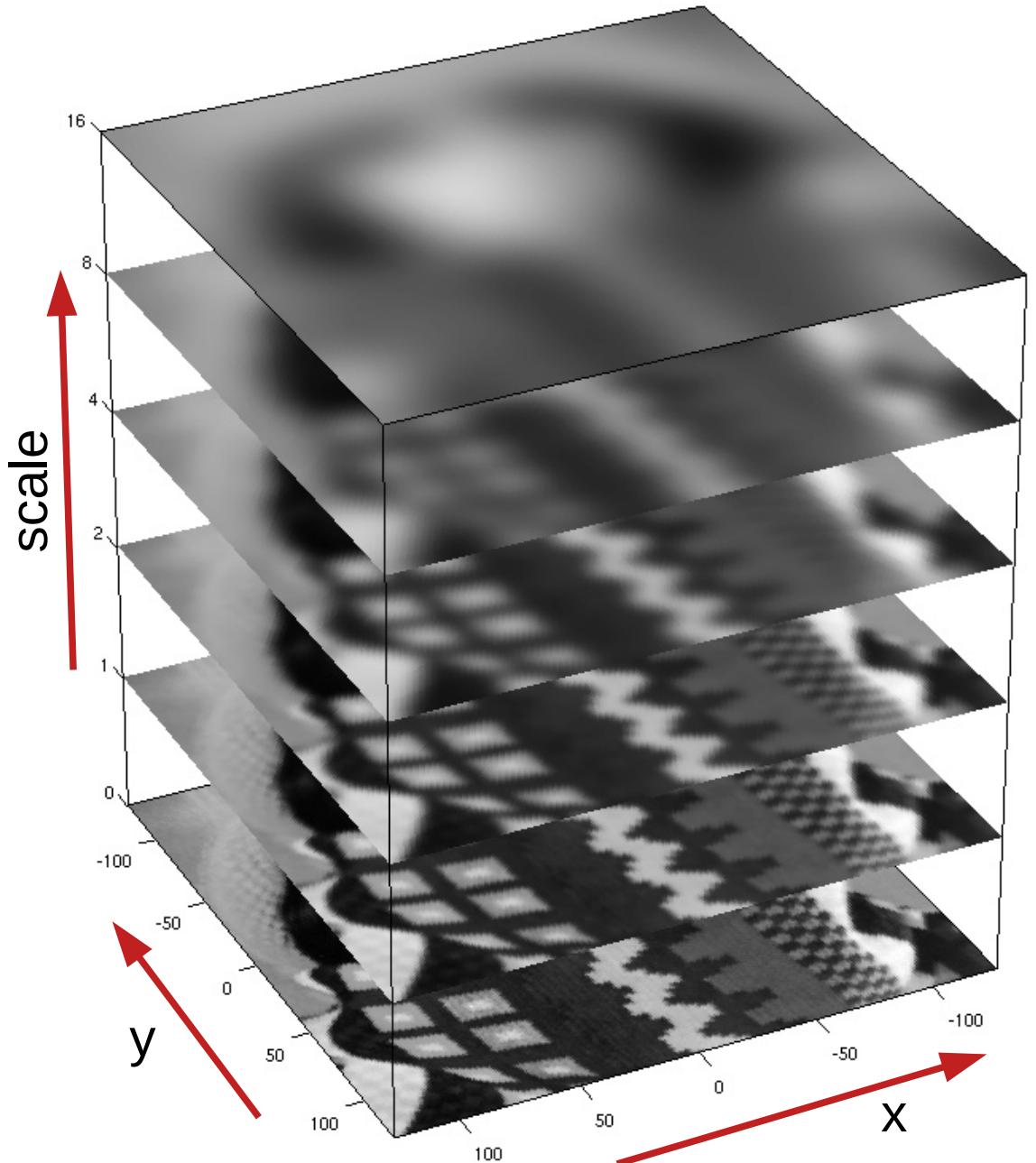
Scale spaces

- All filters had a scale parameter
- Choosing the right scale is problem-dependent
 - Often leads to “magic numbers”
- Smoothing simplifies image for better detecting objects
- Smoothing also shifts object boundaries
- Combine smoothing at many scales to detect objects and its boundaries
- Add dimension to image: $n\text{-D} + \text{scale} = (n+1)\text{-D}$

Scale space

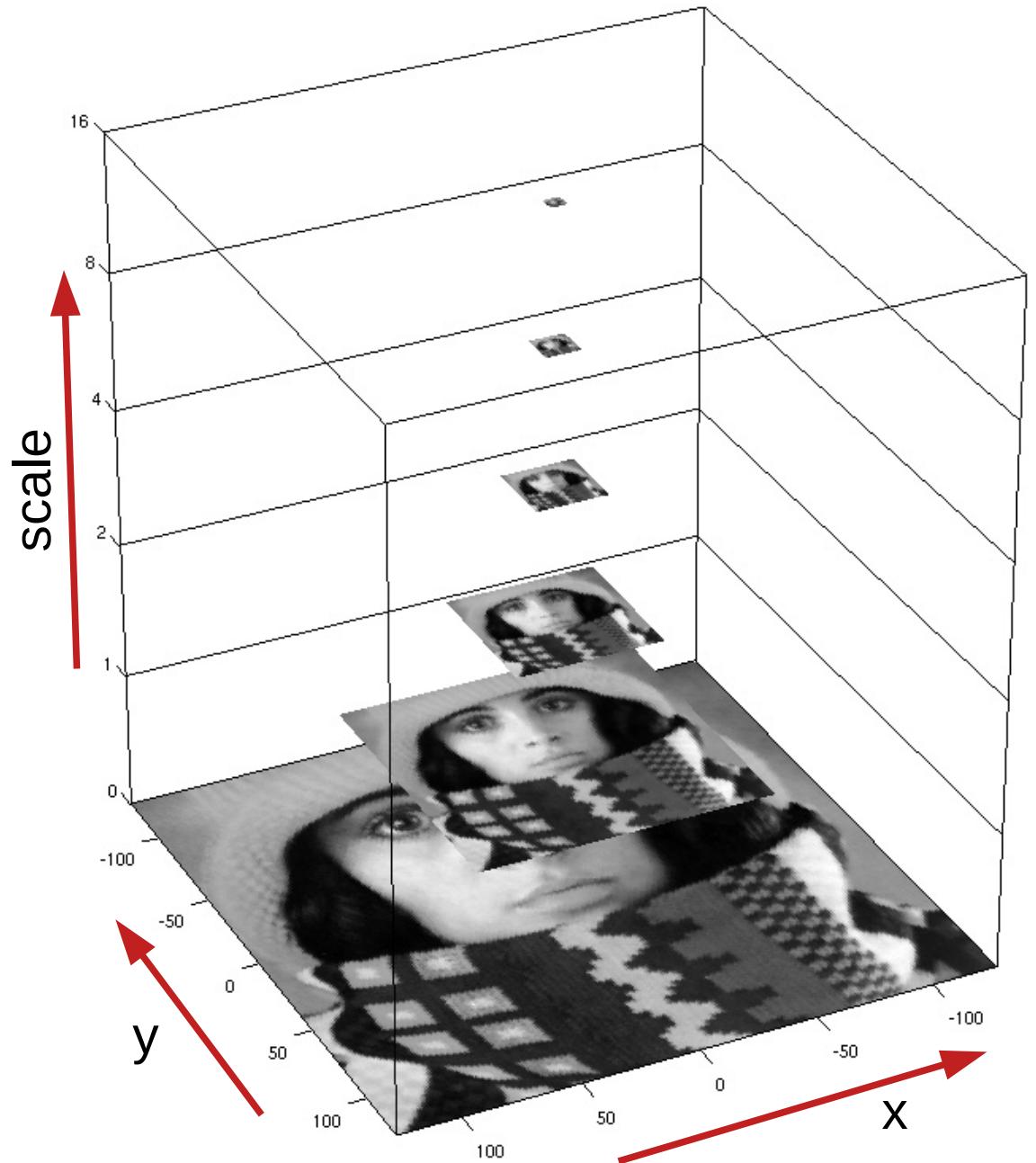
For Gaussian filter,
the “scale” axis
represents time in
the linear diffusion
equation

Non-linear scale
spaces:
• non-linear diffusion
• closings or openings
• wavelet transforms

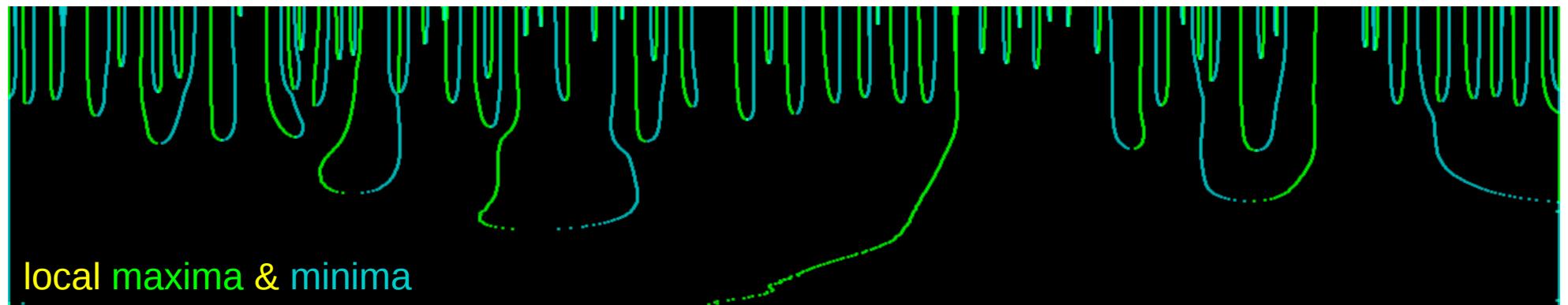


Scale pyramid

Because the filter reduces the information content of the image, it can be downsampled with increasing scale to reduce memory usage



Gaussian scale space



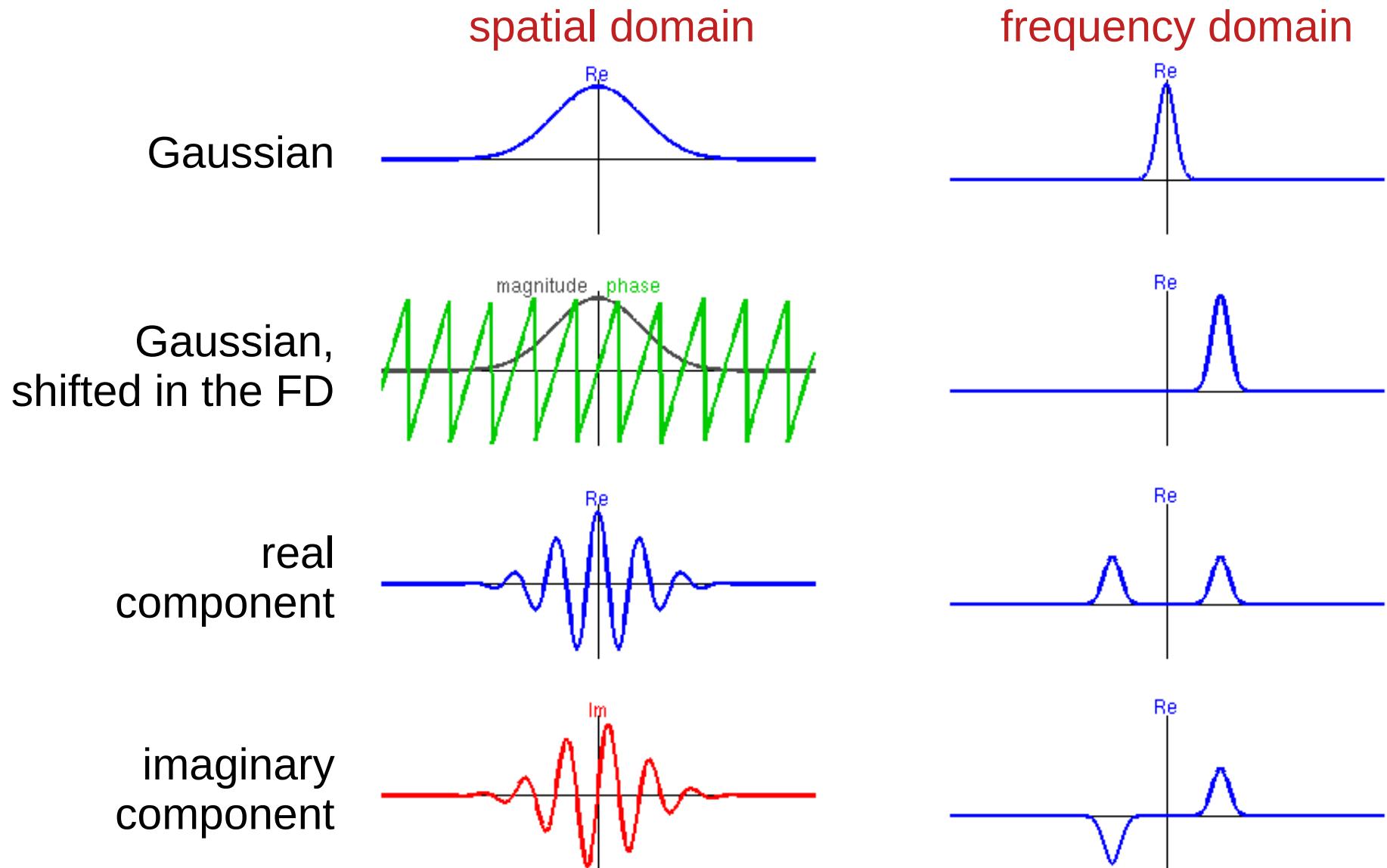
Gabor filter

- We want to show where in the image are grid patterns with a chosen period and orientation
 - Select an area in the Fourier Domain
- We want the resulting spatial filter to be small
 - Use a Gaussian region in the Fourier Domain
- We want the filter to be insensitive to phase
 - Use an even and an odd filter

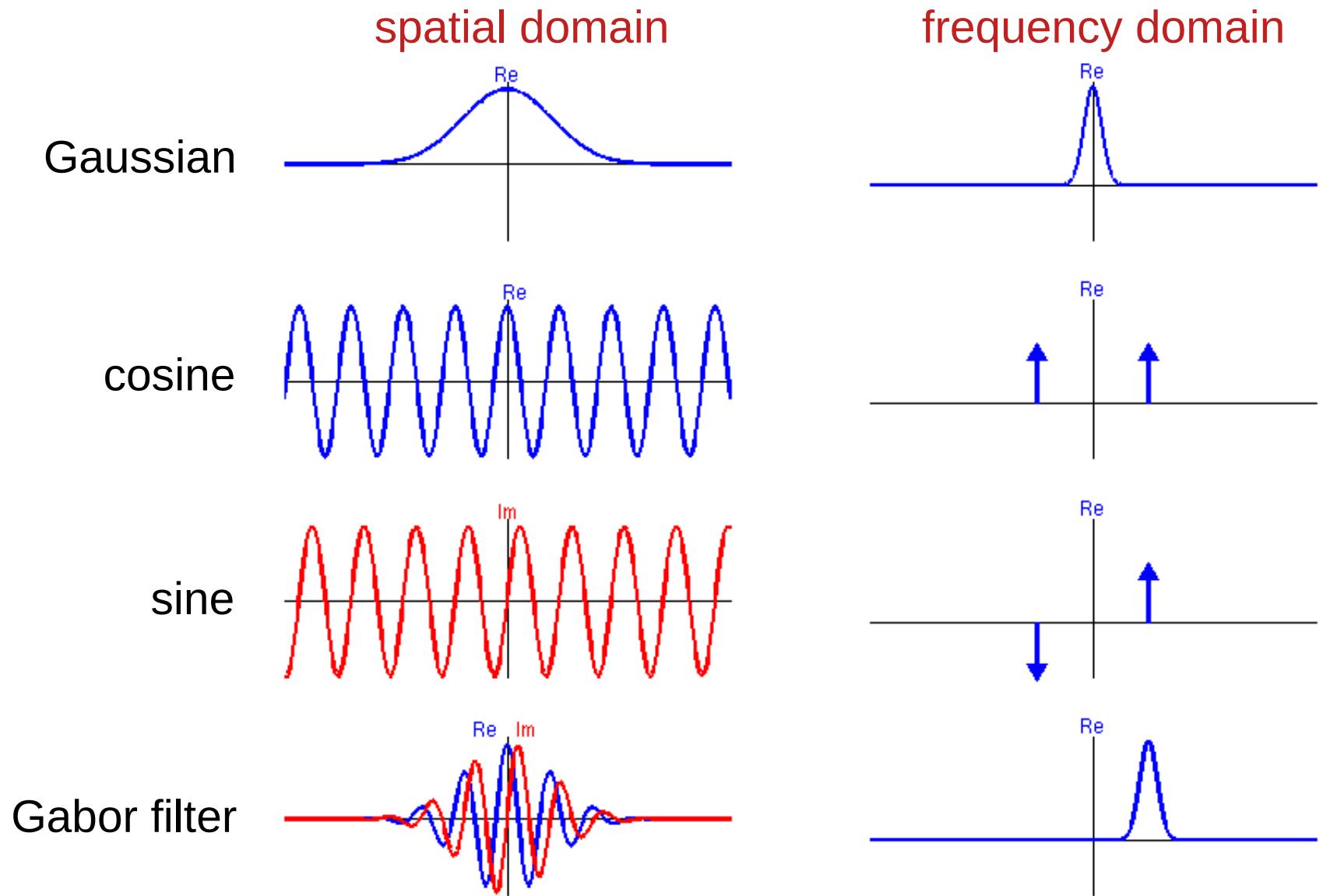
$$F(u, v) = \exp\left(\frac{1}{2}\{(u-u_0)^2 + (v-v_0)^2\}\sigma^2\right) = G(u-u_0, v-v_0)$$

$$\begin{aligned} f(x, y) &= \frac{1}{2\pi\sigma^2} \exp\left(\frac{x^2+y^2}{2\sigma^2}\right) \exp(iu_0x+iv_0y) \\ &= G(x, y)(\cos(u_0x+v_0y)+i\sin(u_0x+v_0y)) \end{aligned}$$

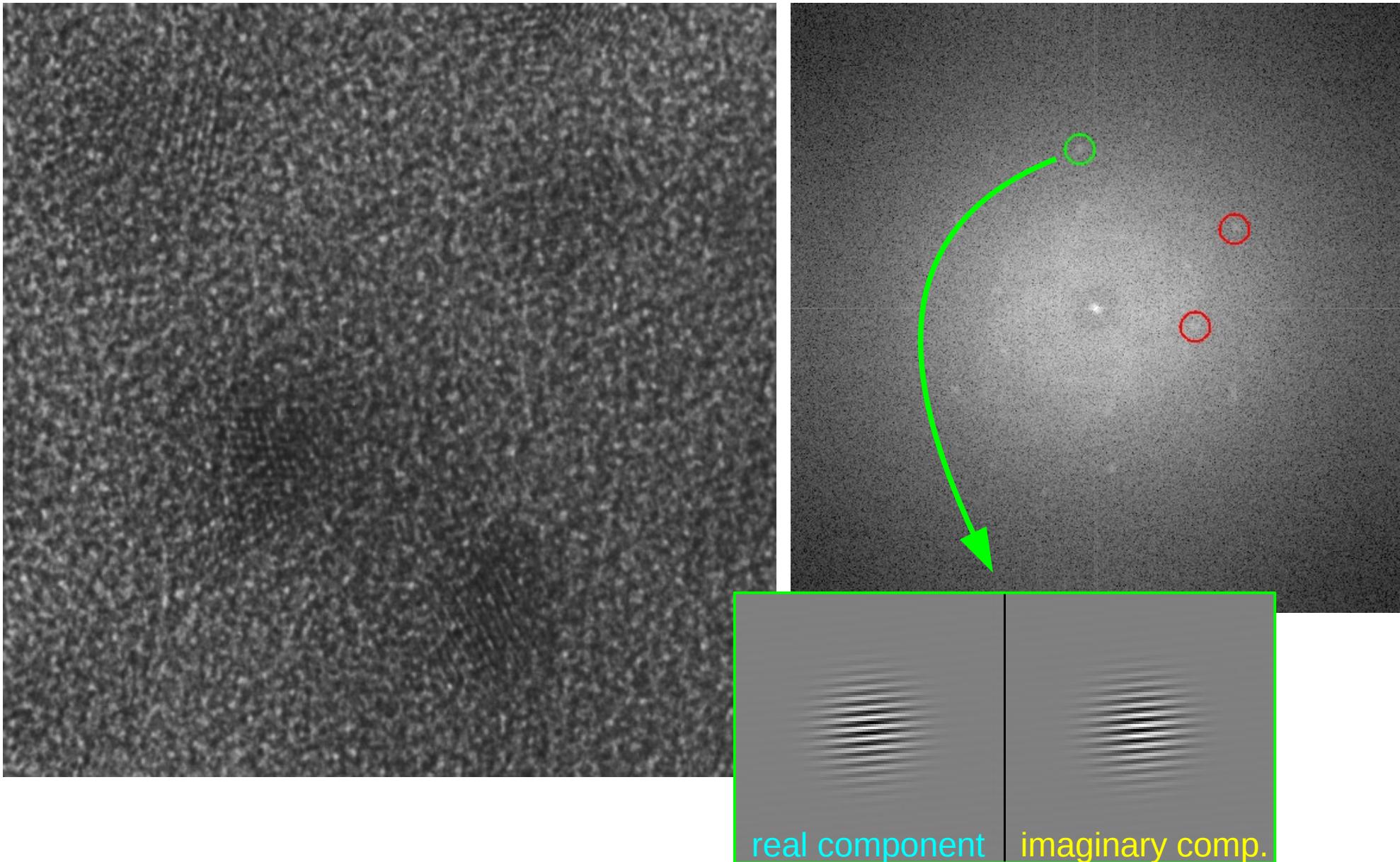
Gabor filter



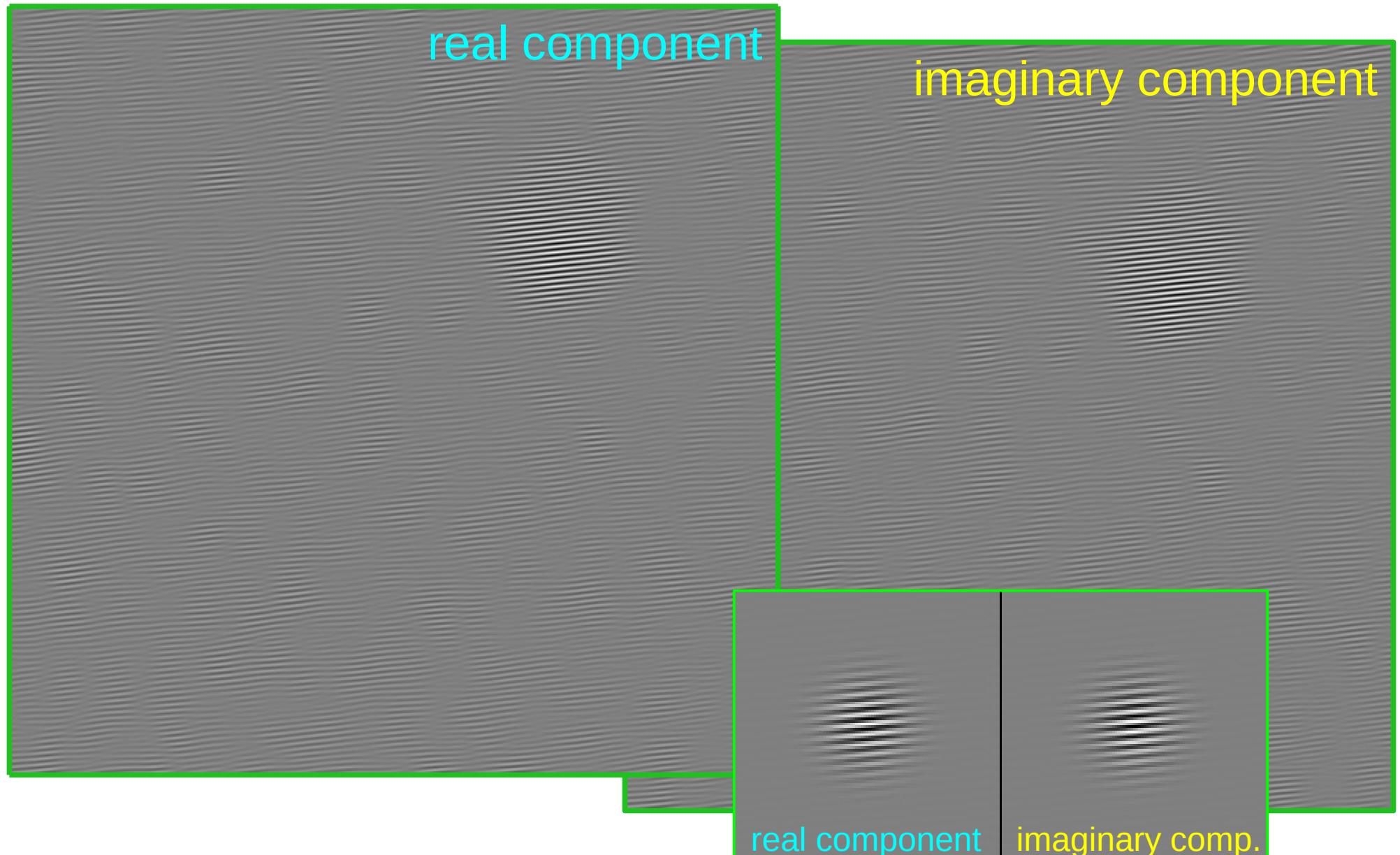
Gabor filter



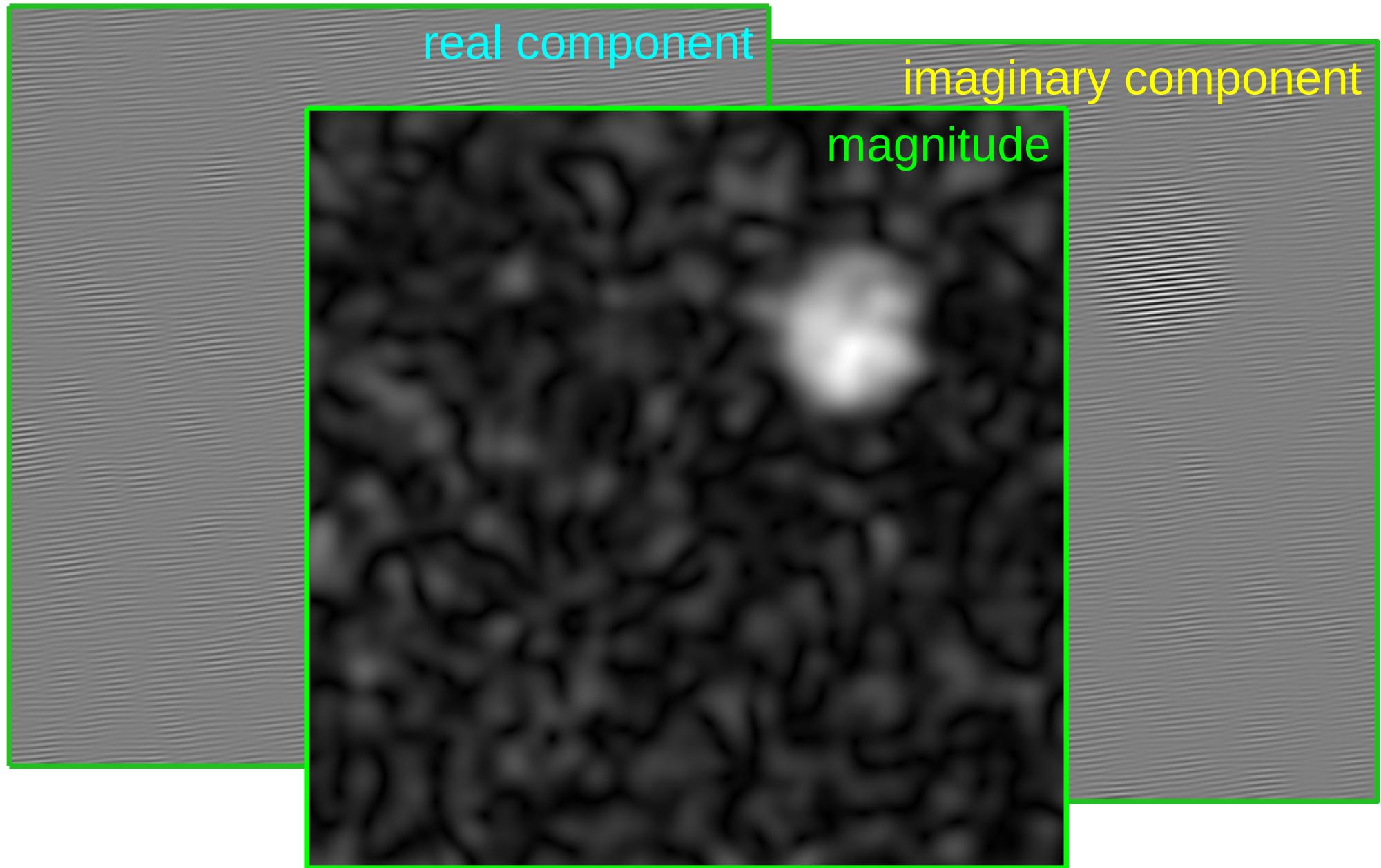
Gabor filter example



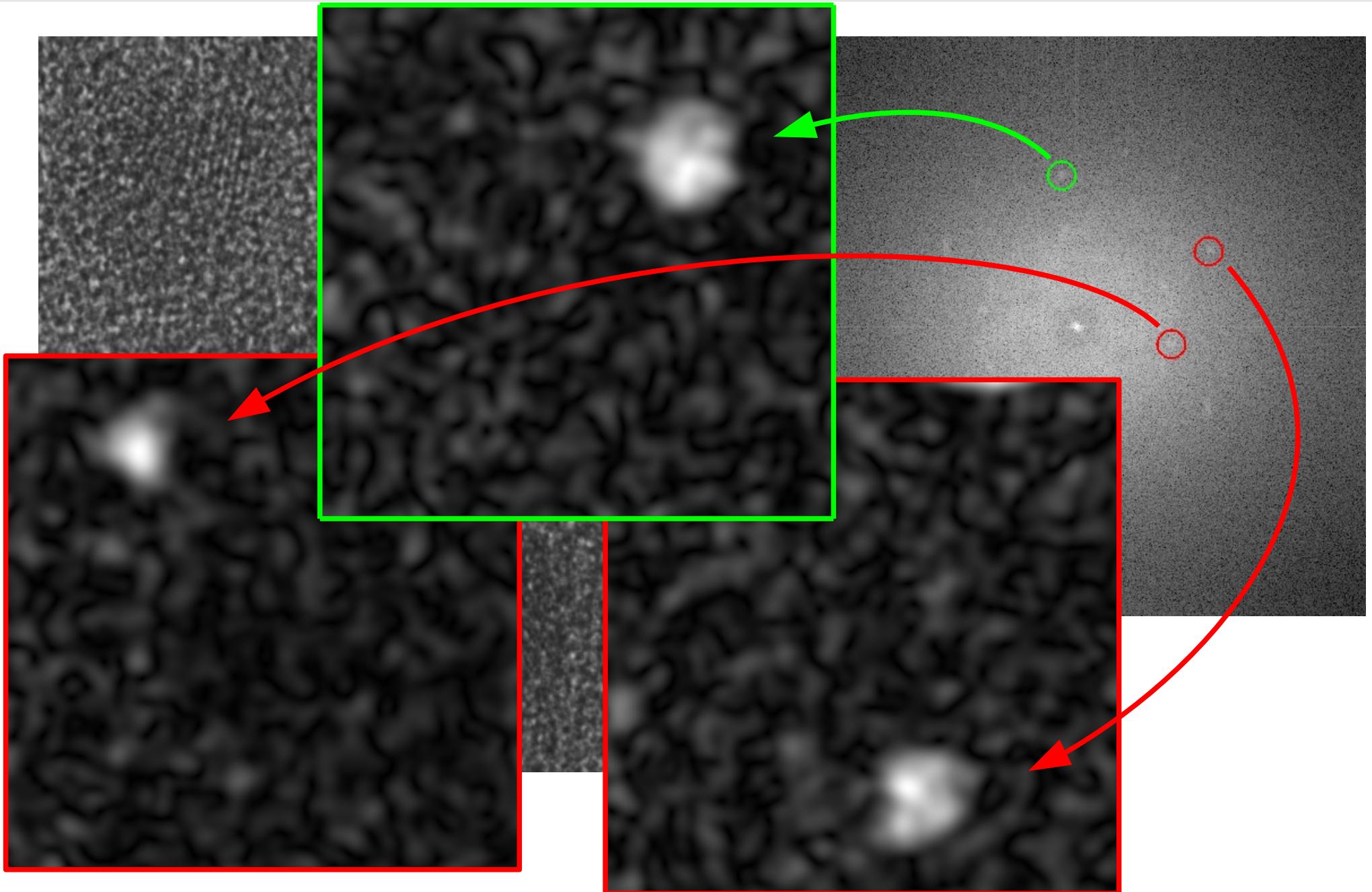
Gabor filter example



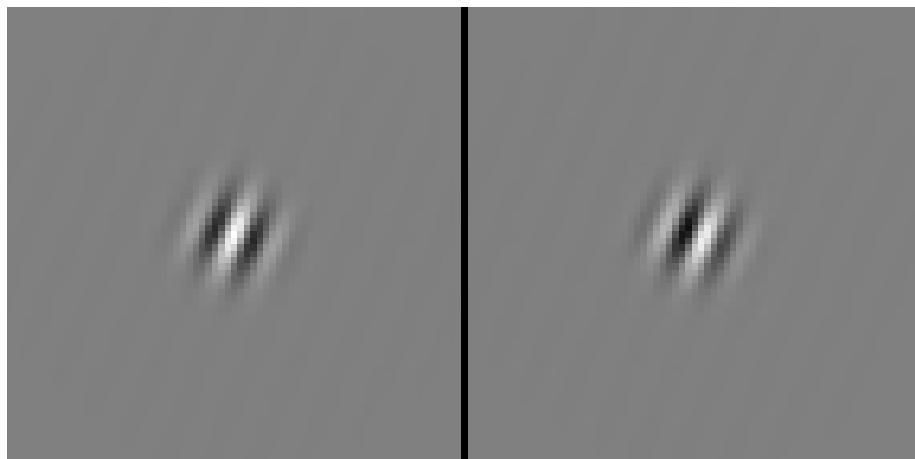
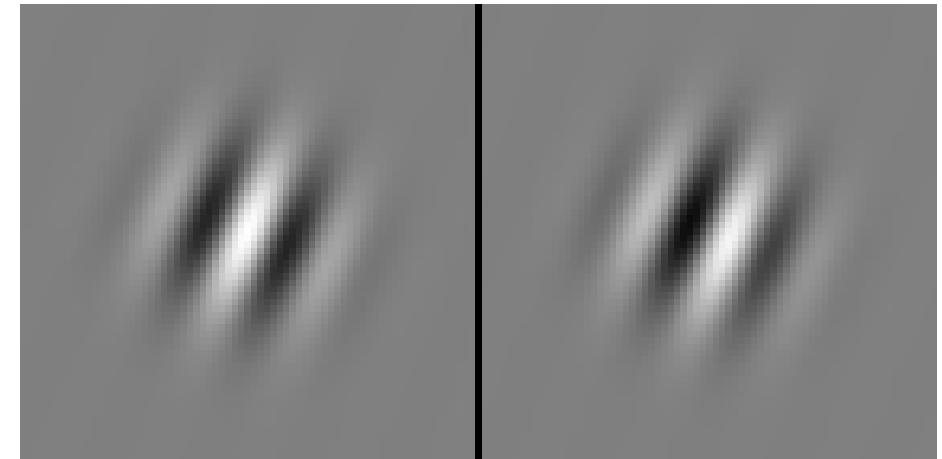
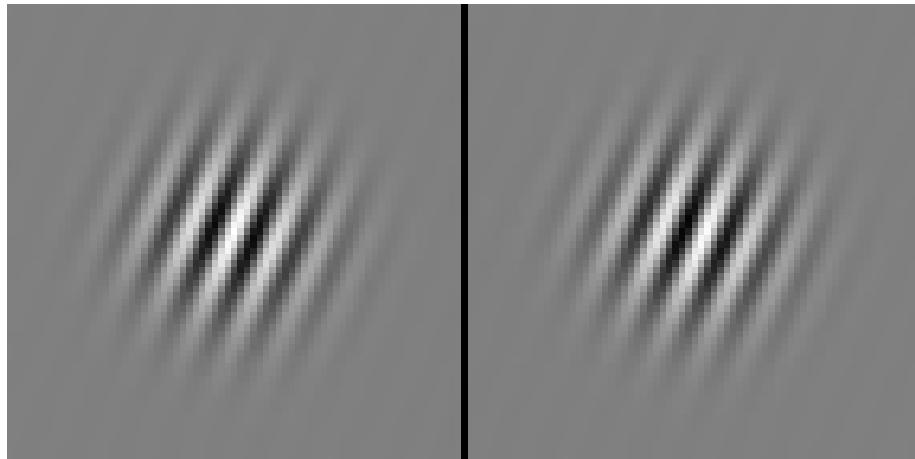
Gabor filter example



Gabor filter example



Gabor filter



smaller $\sigma \Rightarrow$ less specific

lower $\omega \Rightarrow$ needs larger σ

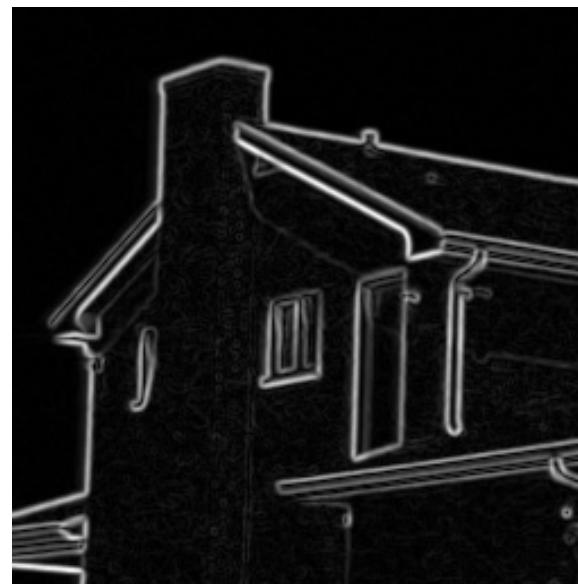
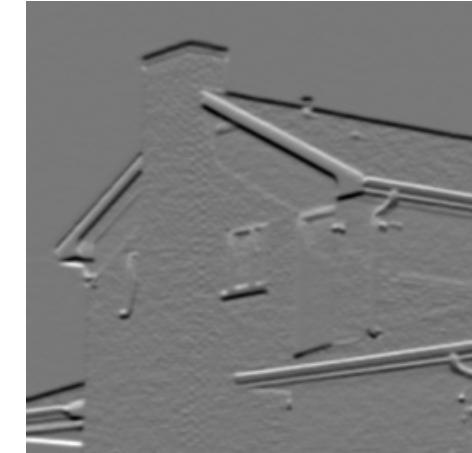
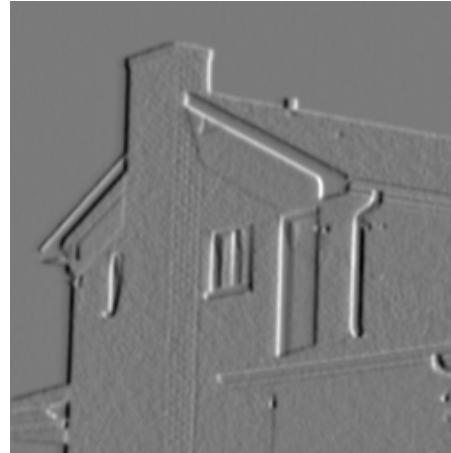
Edge detection

- 1st order derivatives measure gradient
- Gradient is strong at edges
- Maximum of gradient magnitude, in direction of gradient, gives location of edge
 - Canny's edge detector
- Zero crossing of second derivatives (Laplace) gives location of edge
 - Difficult to determine which zero-crossings are relevant
- By smoothing the image, we choose the scale of the edges

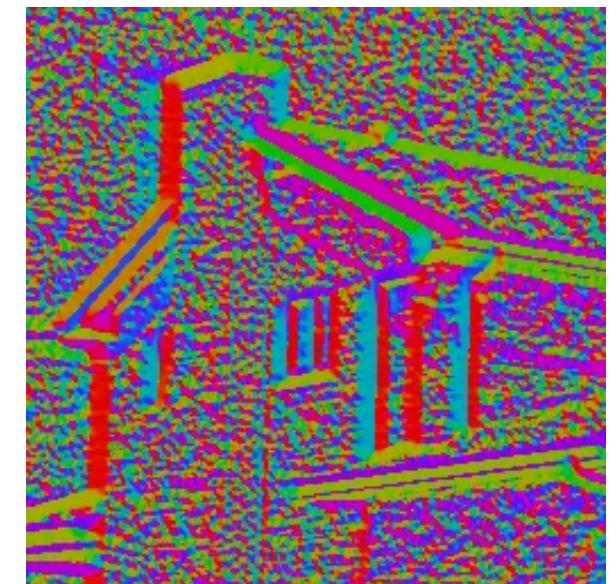
Canny's edge detector

- Compute gradient magnitude and gradient direction
- Find pixels in the gradient magnitude that are local maxima in the gradient direction (ridges of gradient magnitude)
- Many pixels are part of a ridge!
- Prune ridges using hysteresis threshold:
 - keep any ridge line that has a large gradient magnitude in some section
 - a normal threshold would break up edges if a small portion were to have less contrast

Canny's edge detector

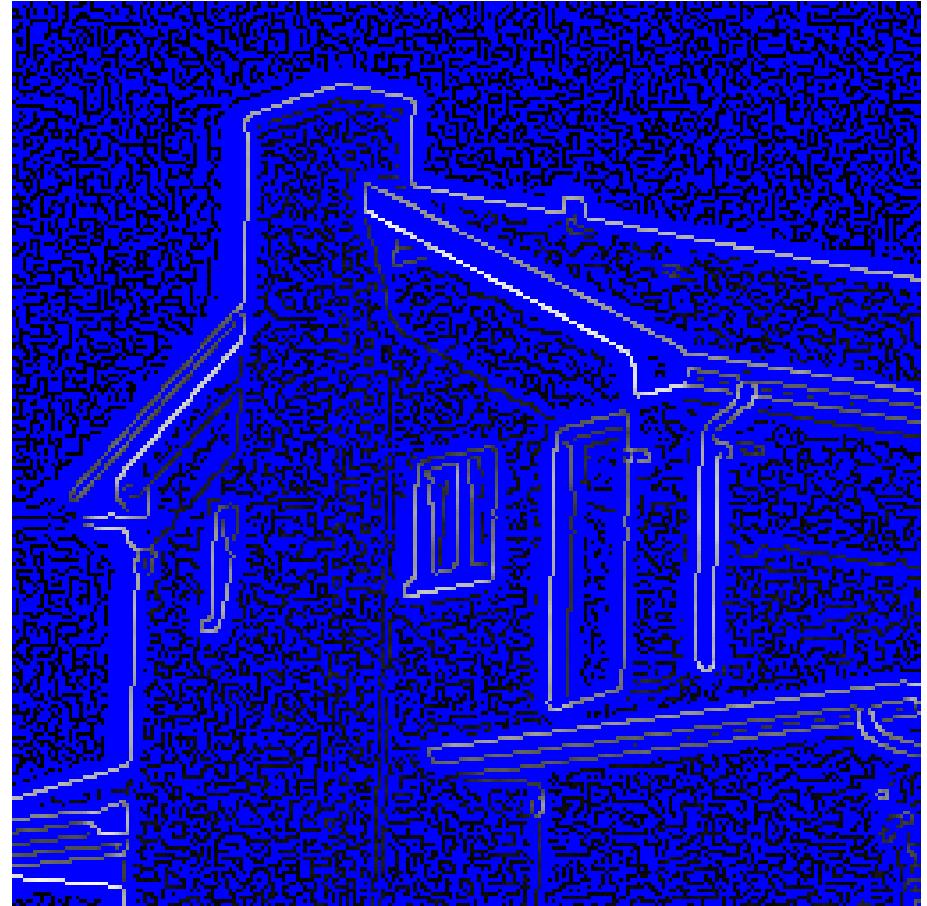


gradient magnitude



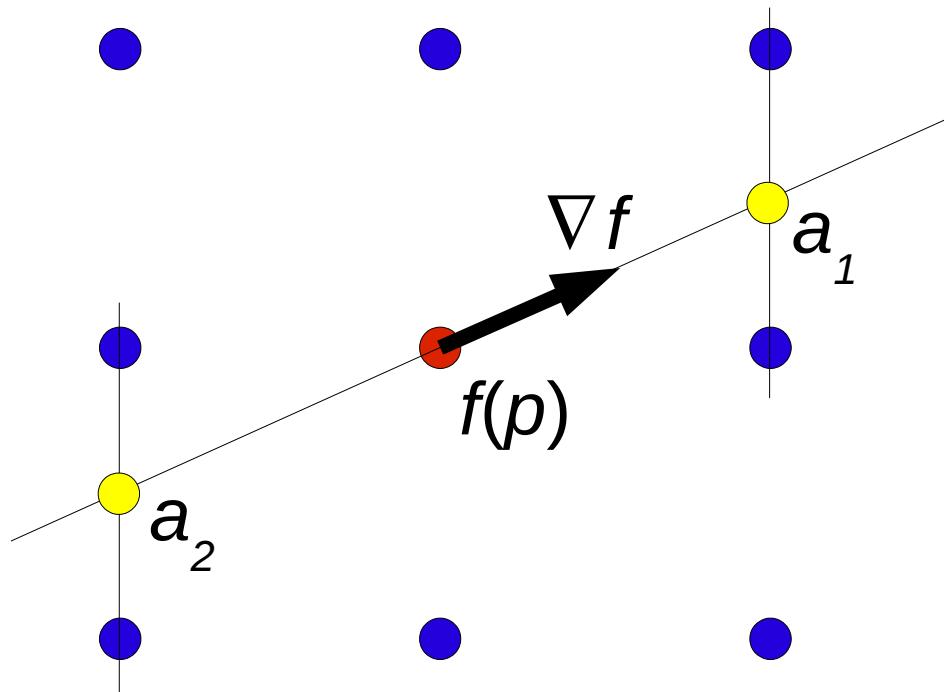
gradient direction

Canny's edge detector



Non-maxima suppression

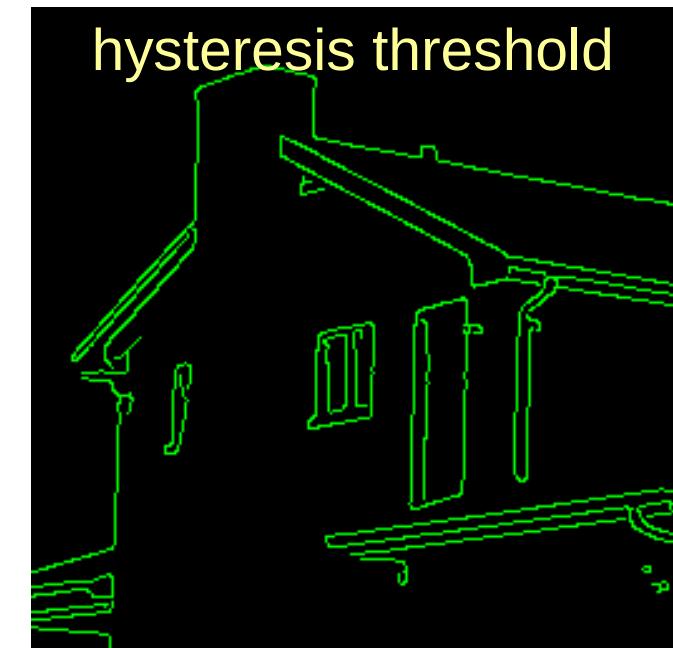
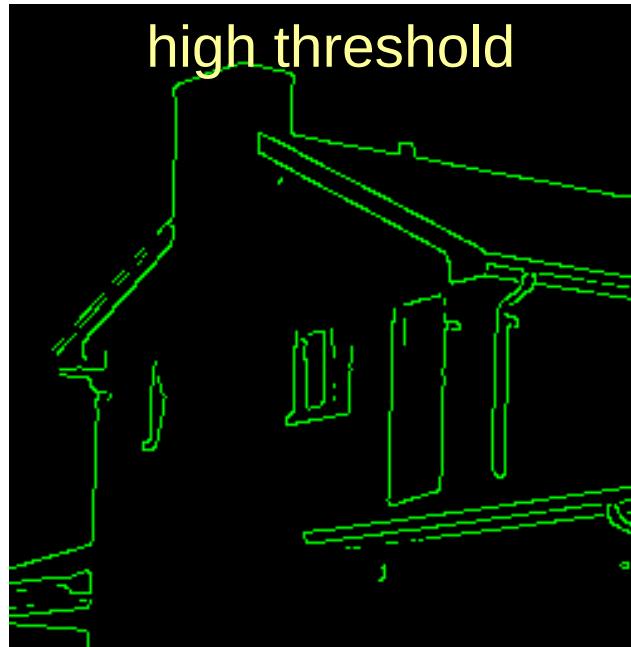
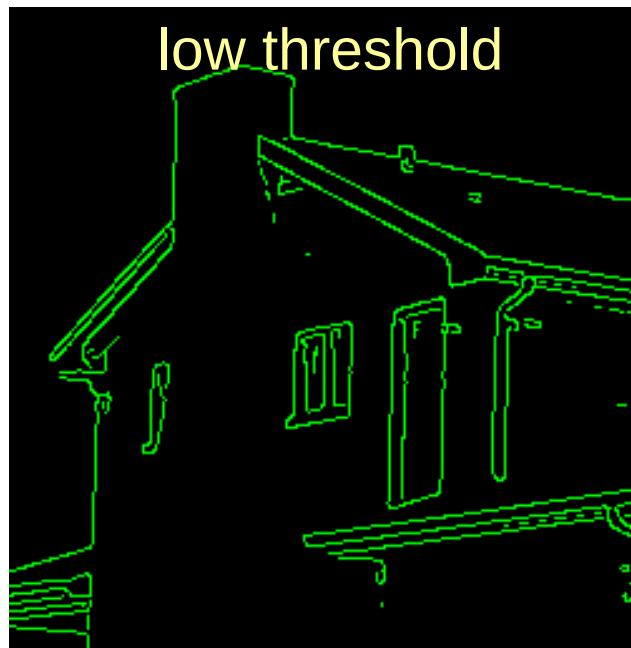
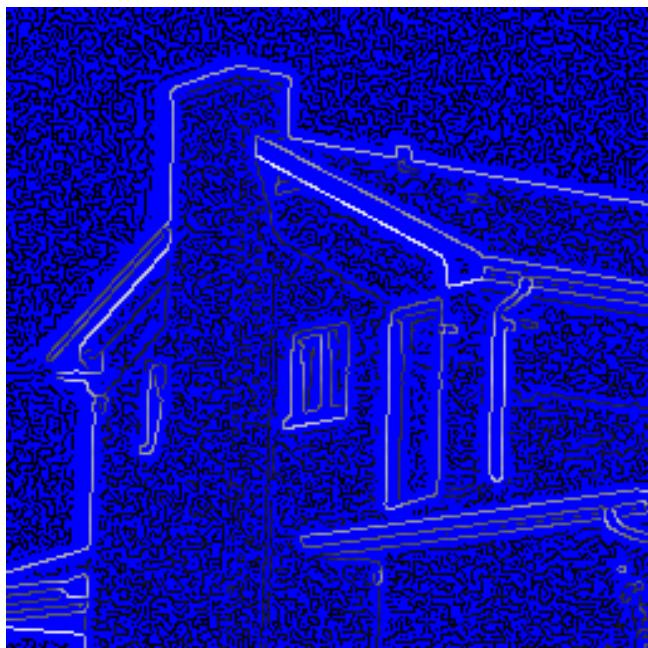
Non-maxima suppression



a_1 & a_2 computed by linear interpolation

For each point p check:
 $f(p) > a_1 \wedge f(p) \geq a_2$
 $f(p) \geq a_1 \wedge f(p) > a_2$
If condition not met, set pixel to 0.

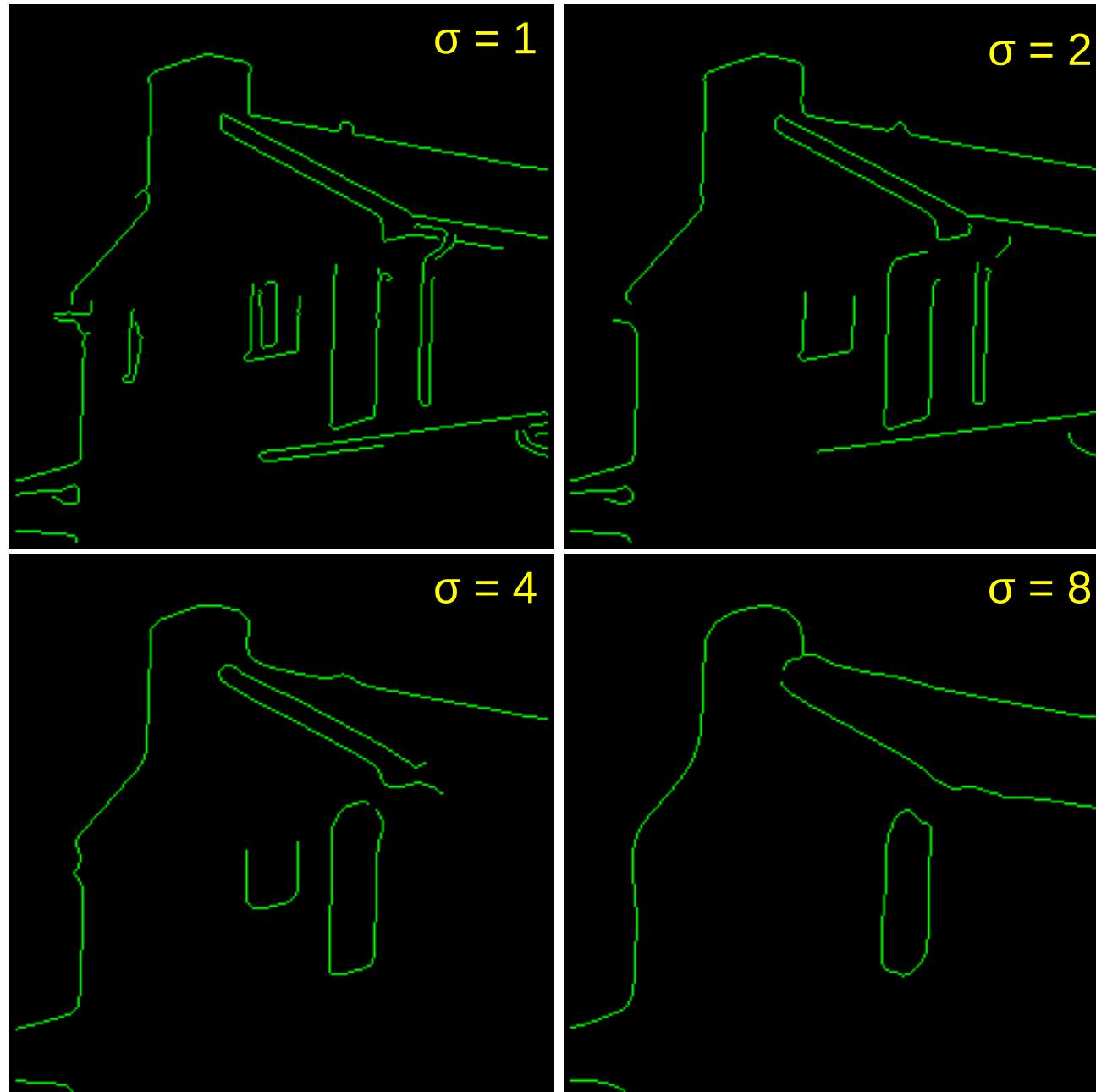
Canny's edge detector



Canny's edge detector



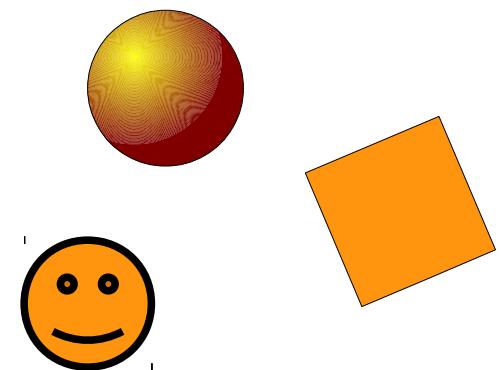
Influence of the σ



Use a scale space!

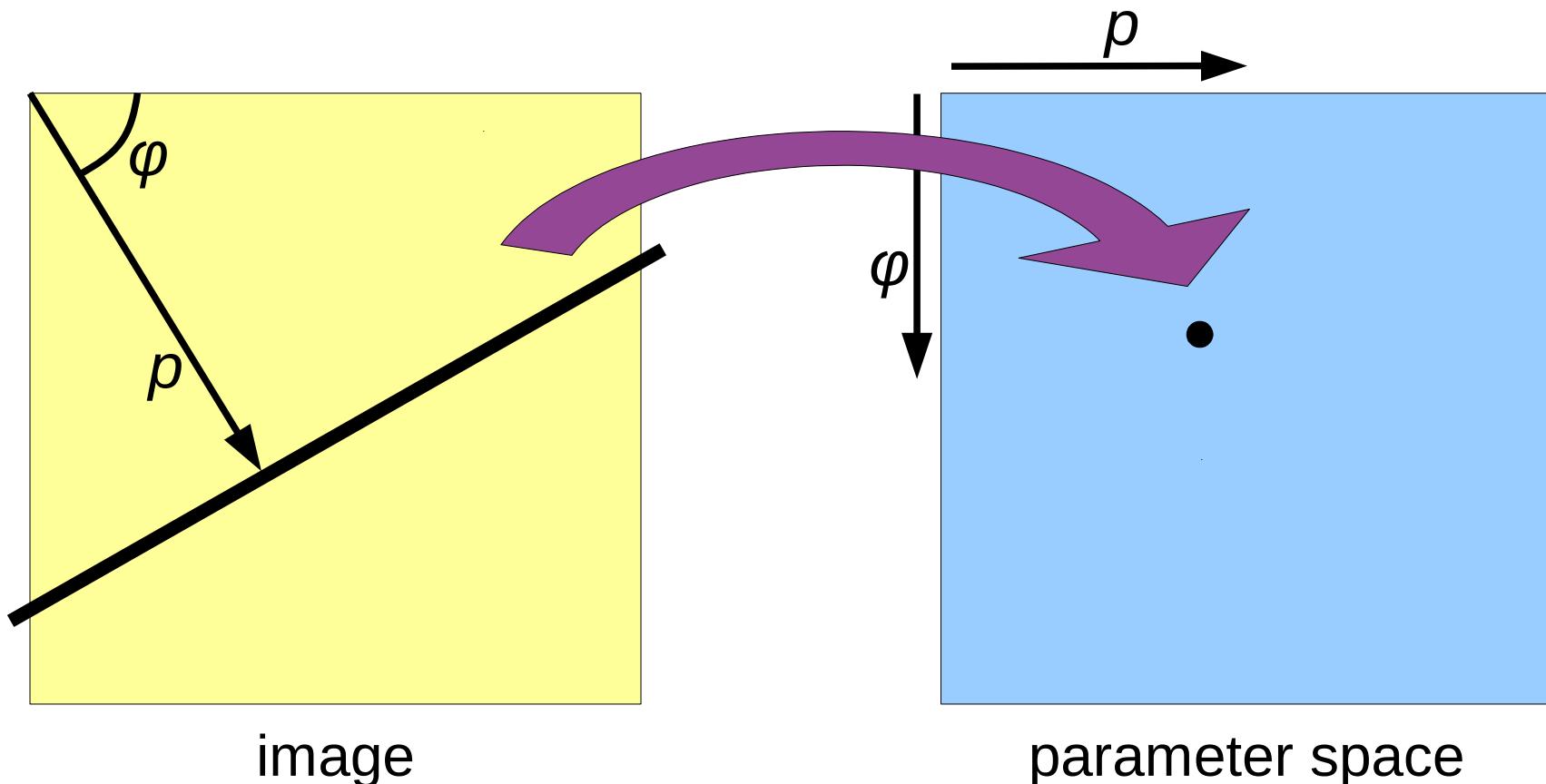
Hough transform

- Detecting parametrized shapes in an image
- Originally for straight lines in 2-D (2 parameters: p , φ)
- Later generalized to any shape:
 - 3-D sphere (4 parameters: x , y , z , r)
 - 2-D square (4 parameters: x , y , a , φ)
 - 2-D smiley (3 parameters: x , y , r)
 - ...
- Creates a “parameter space”
- Converts the shape detection into maxima detection



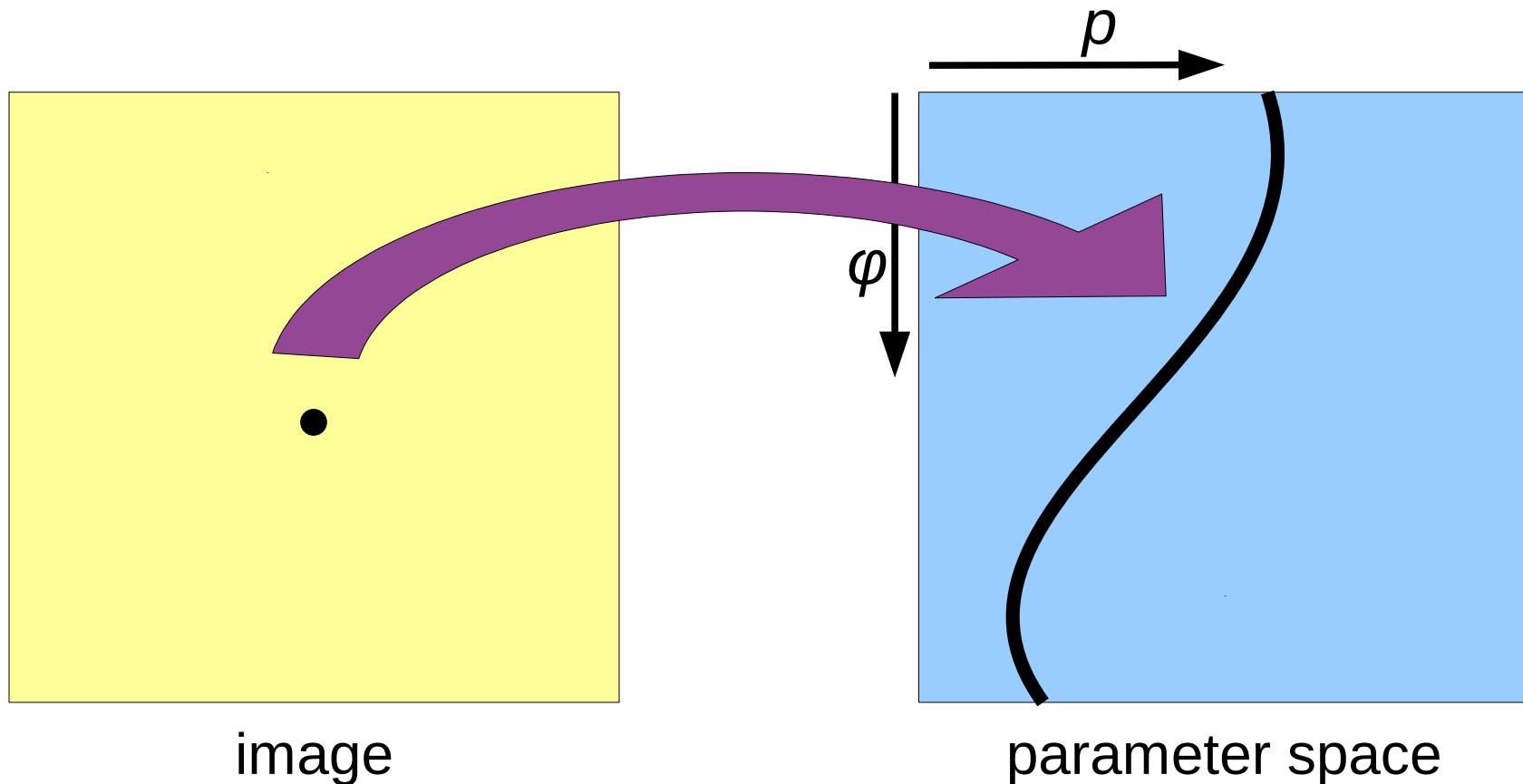
2D Hough transform for lines

- Each point in the parameter shape represents one instance of the shape in the image



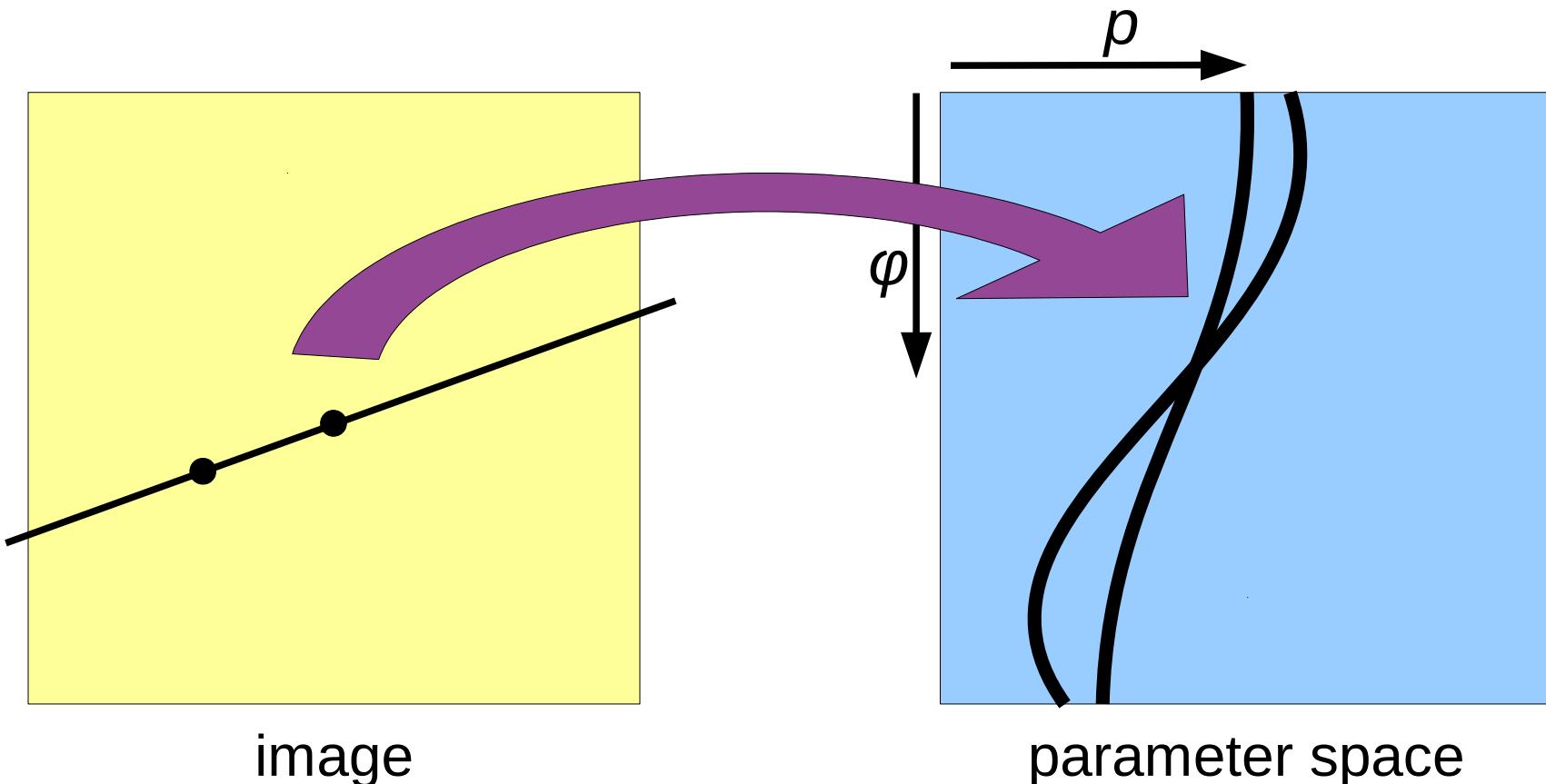
2D Hough transform for lines

- Each point in the image gives evidence for all lines that go through that point

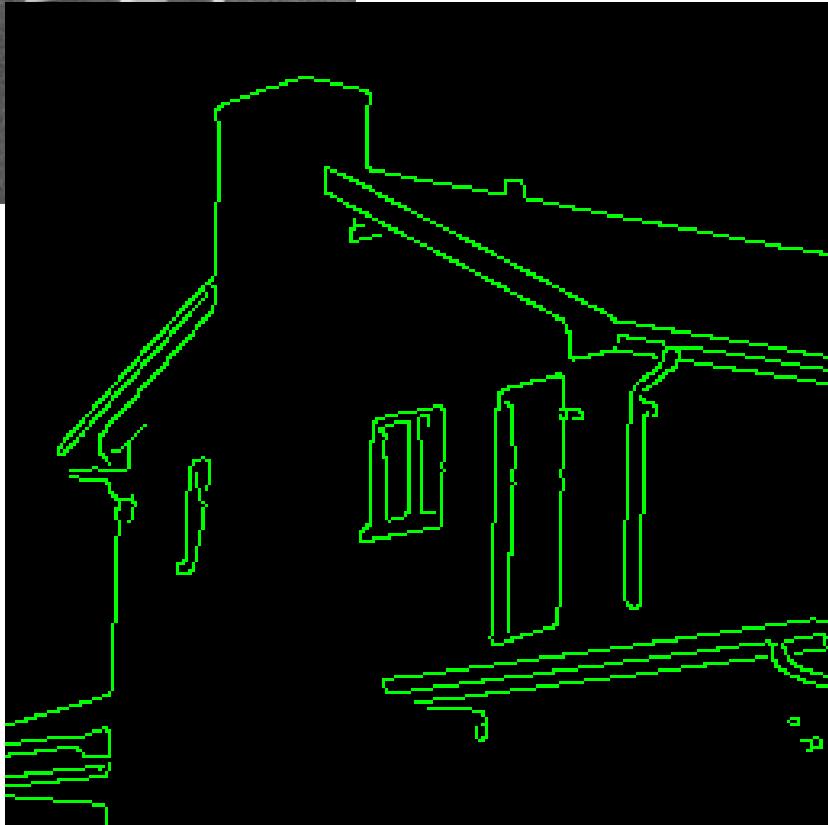


2D Hough transform for lines

- Each point in the image gives evidence for all lines that go through that point



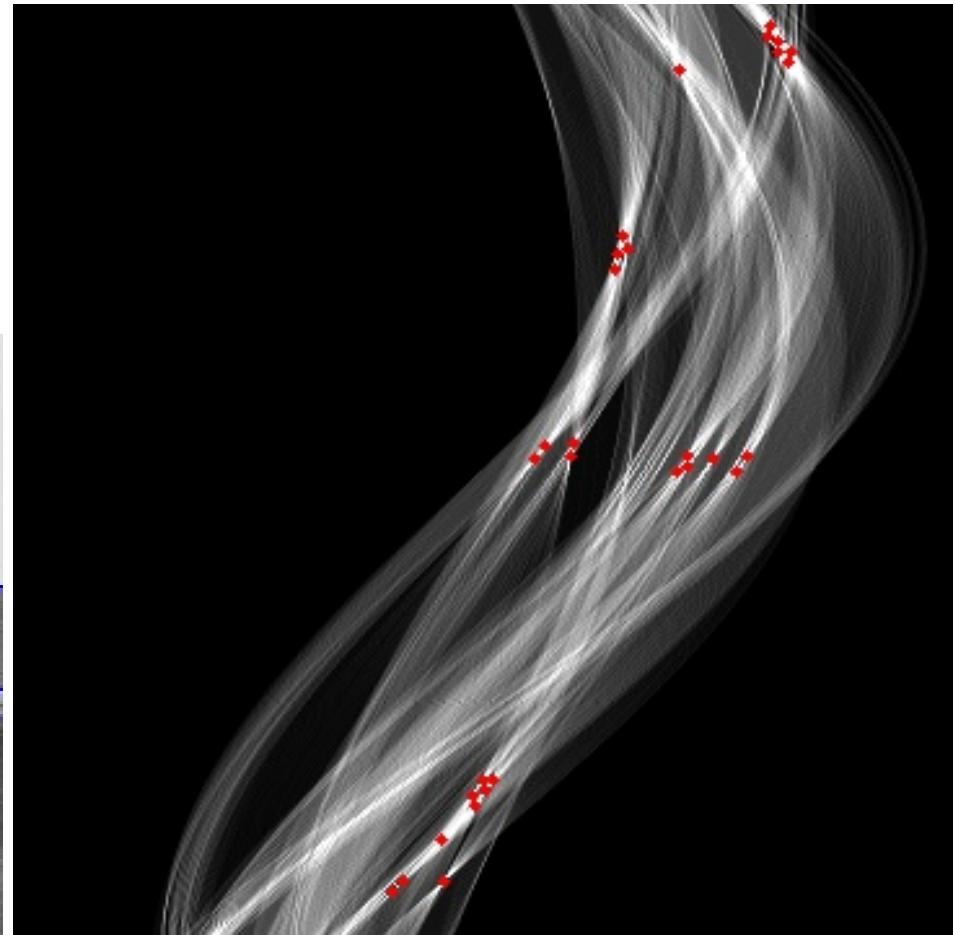
Hough transform example



2D Hough transform for lines

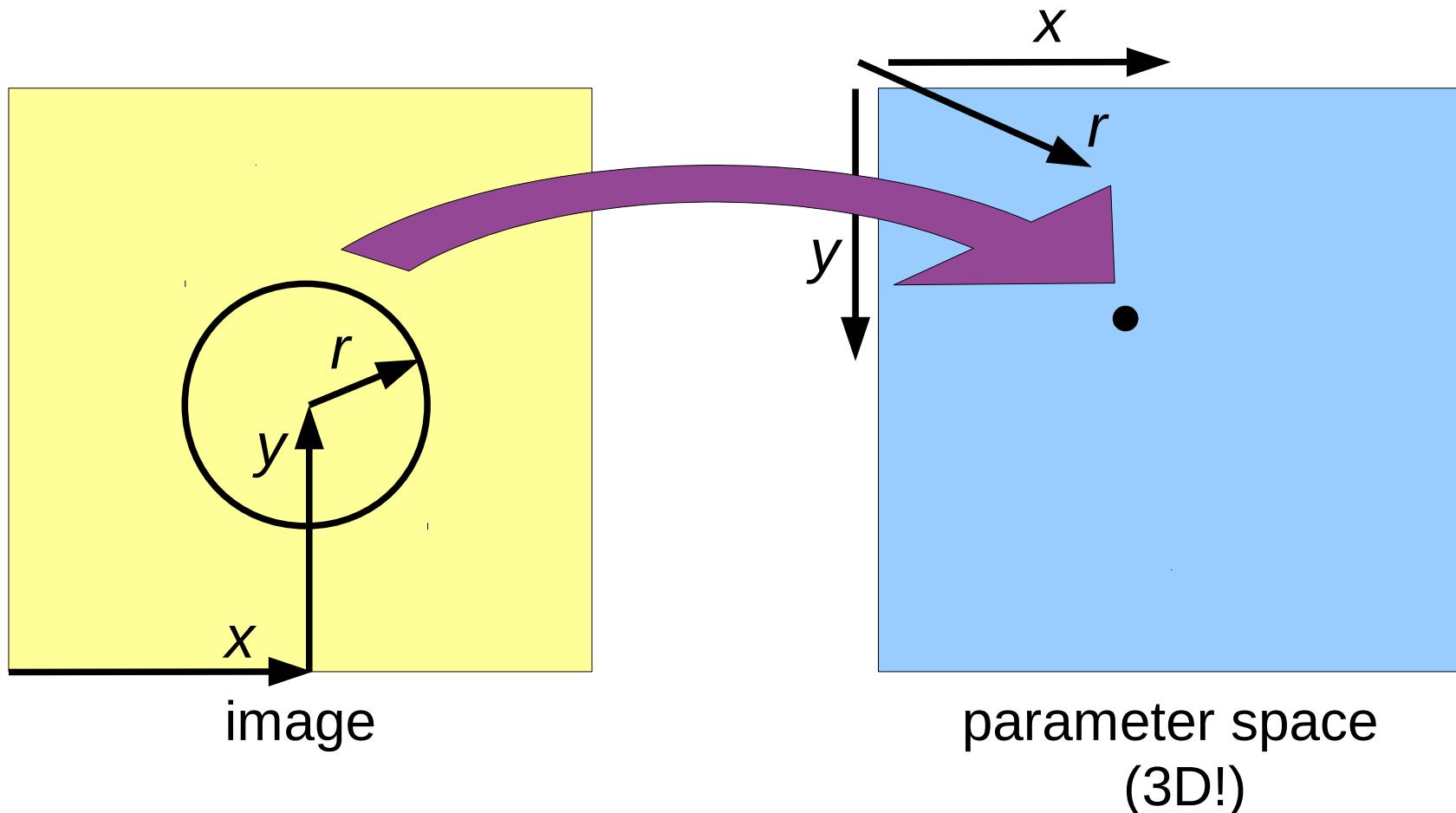
- Each local maximum defines a line that goes across the full image
- We need to detect the line segment in the image
- For each local maximum:
 - Look in the input image where there is support for this line
 - Fill small gaps in the line
 - Is the resulting line long enough?
- In the MATLAB Image Processing Toolbox:
 - hough: create the Hough transform
 - houghpeaks: detect local maxima
 - houghlines: find the line segments

Hough transform example



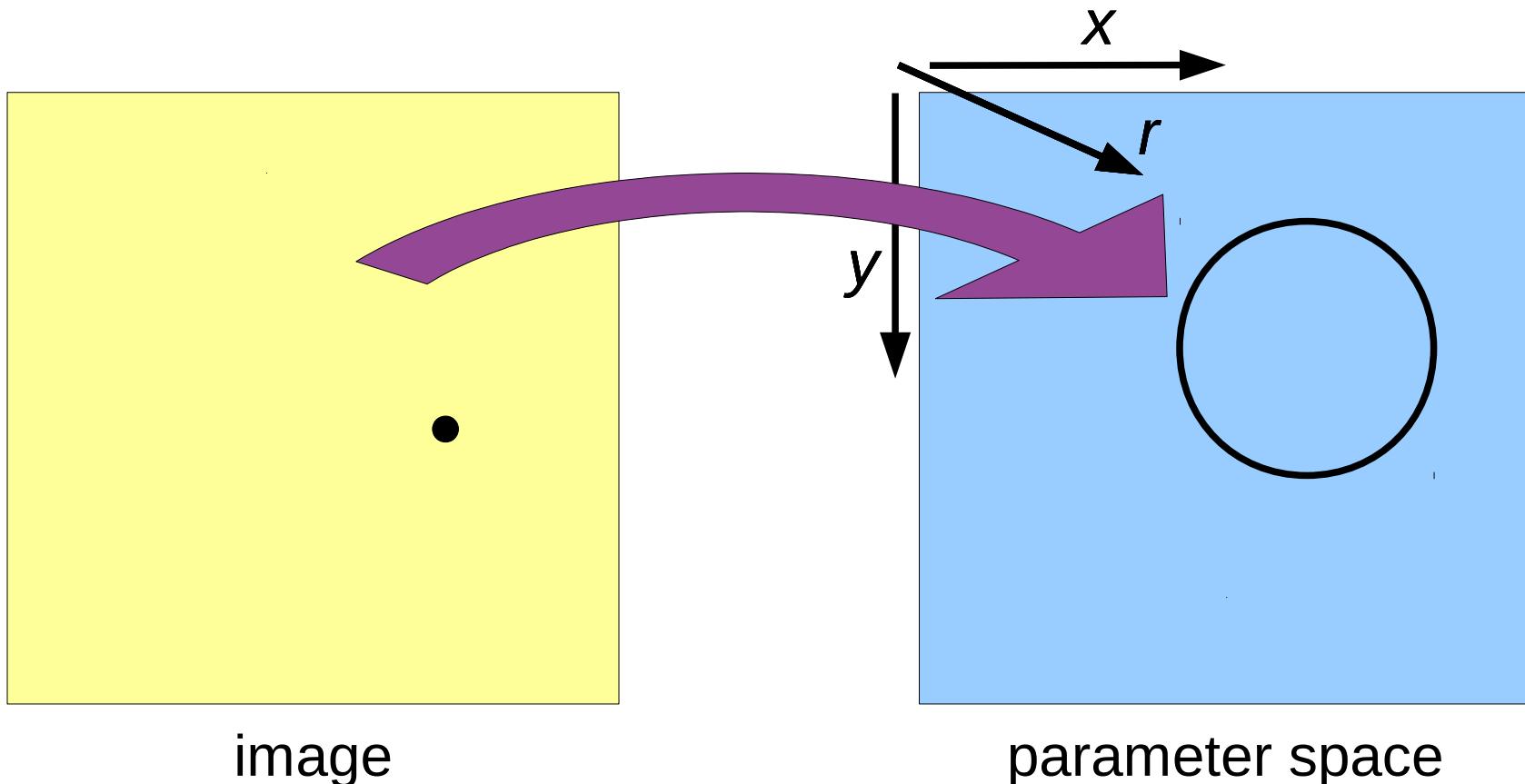
2D Hough transform for circles

- Each point in the parameter shape represents one instance of the shape in the image



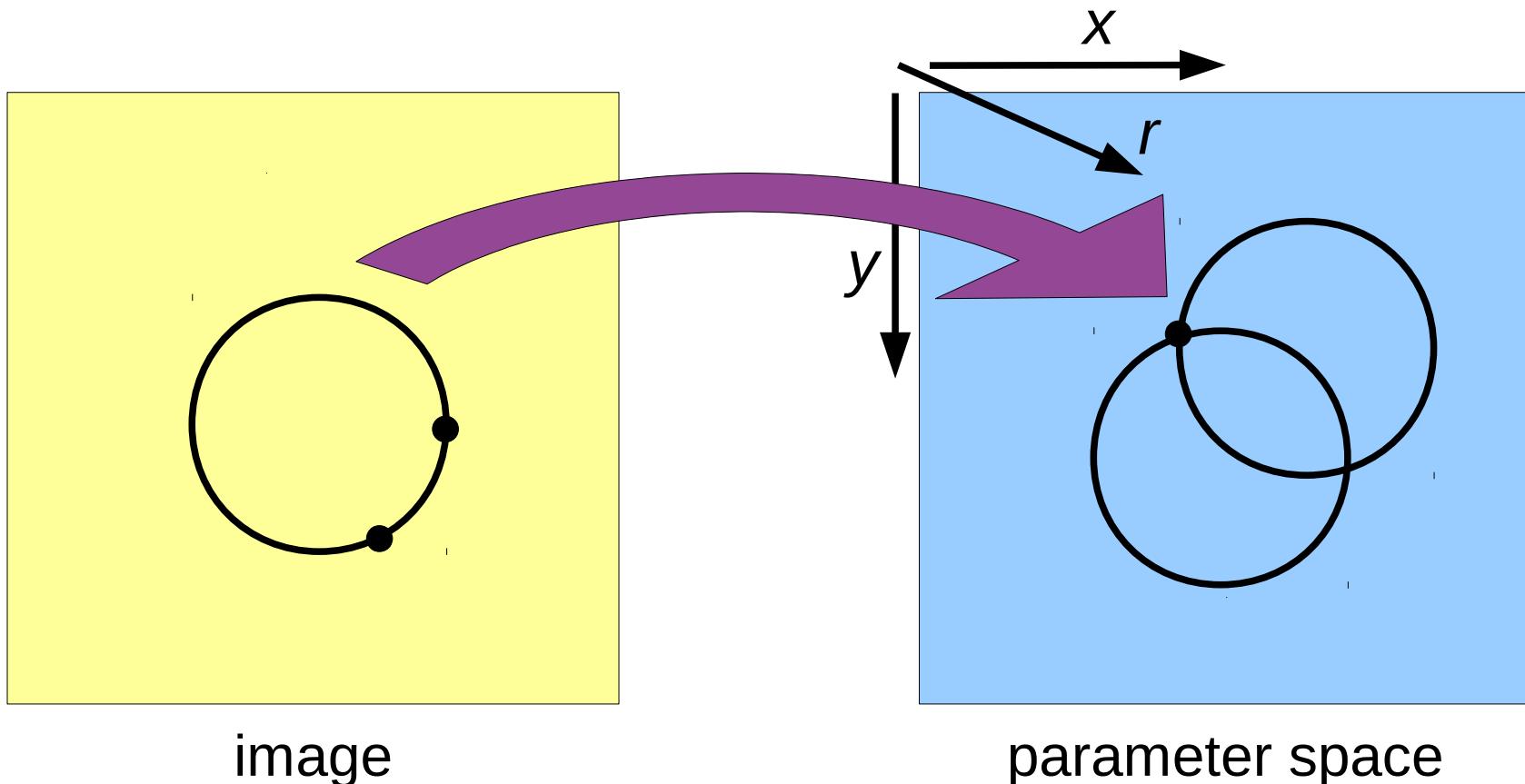
2D Hough transform for circles

- Each point in the image gives evidence for all lines that go through that point



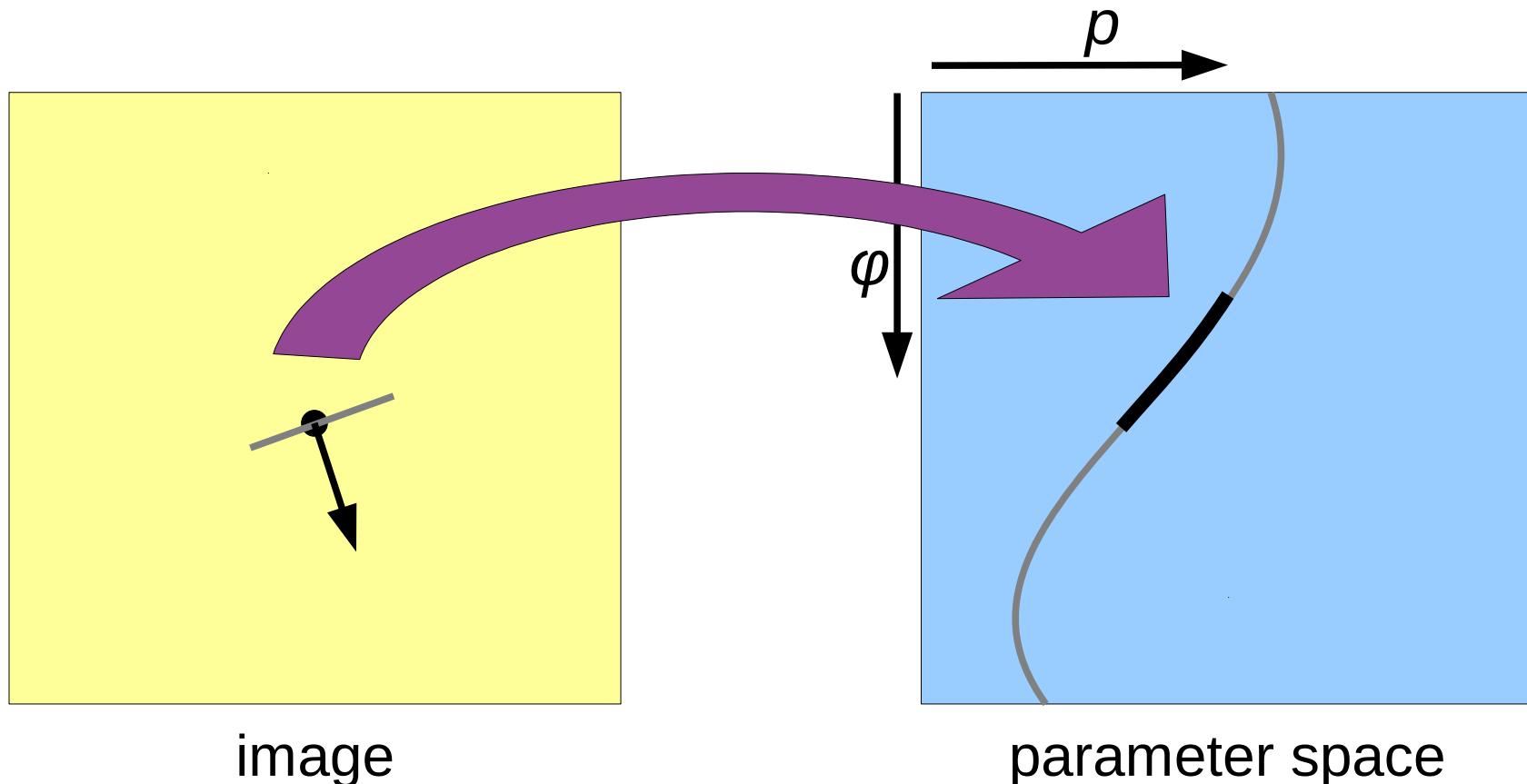
2D Hough transform for circles

- Each point in the image gives evidence for all lines that go through that point



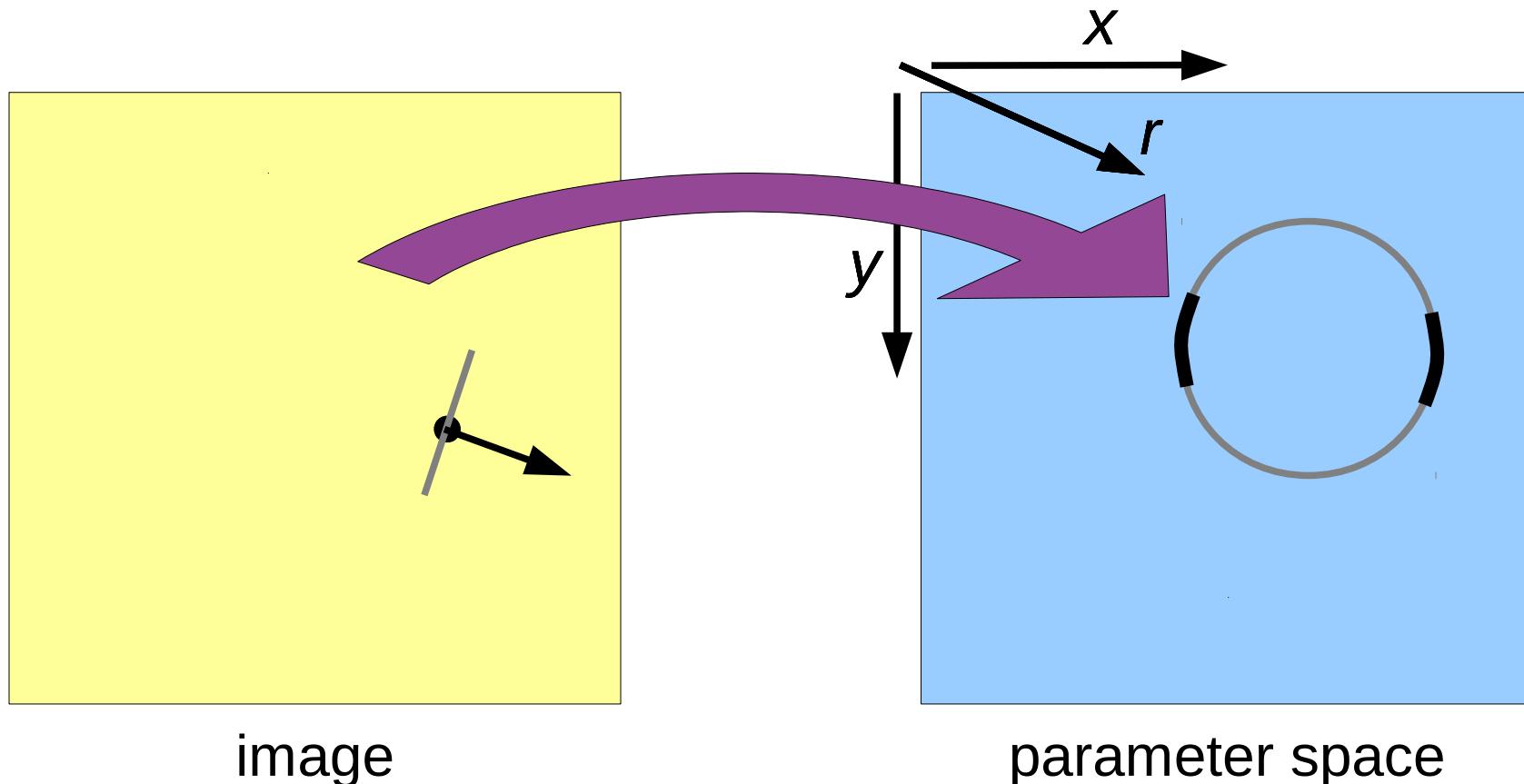
Improving the Hough transform

- Use additional knowledge about the points in the image, e.g. local gradient information

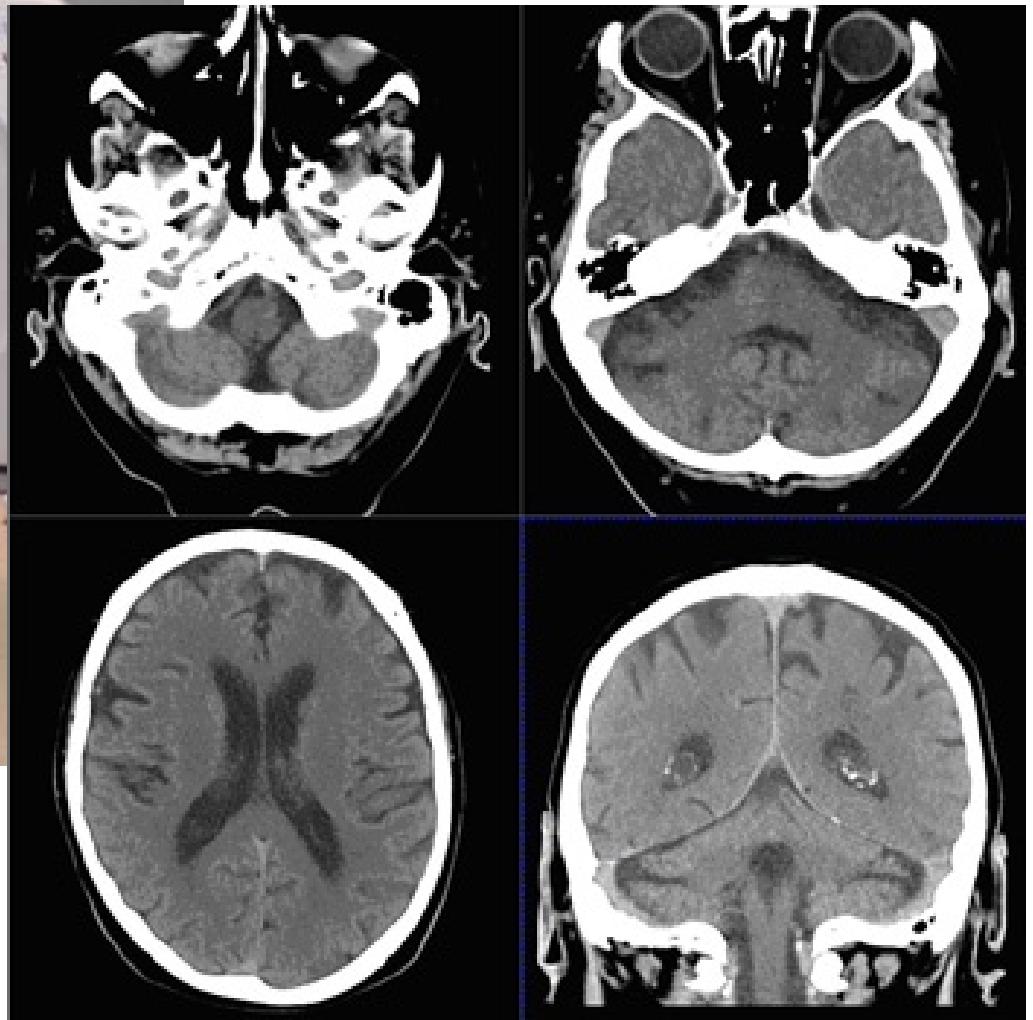


Improving the Hough transform

- Each point in the image gives evidence for all lines that go through that point

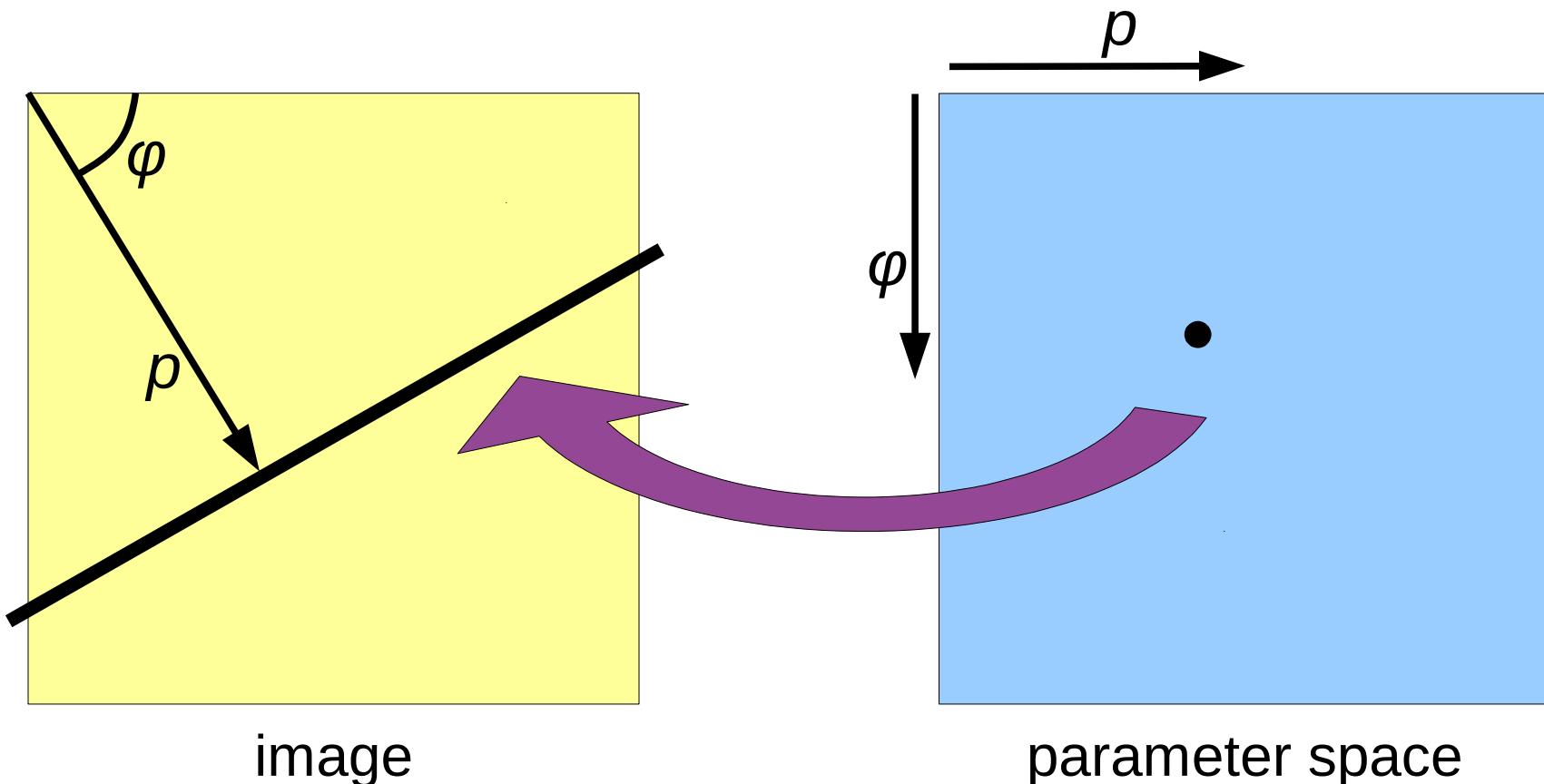


The Radon transform



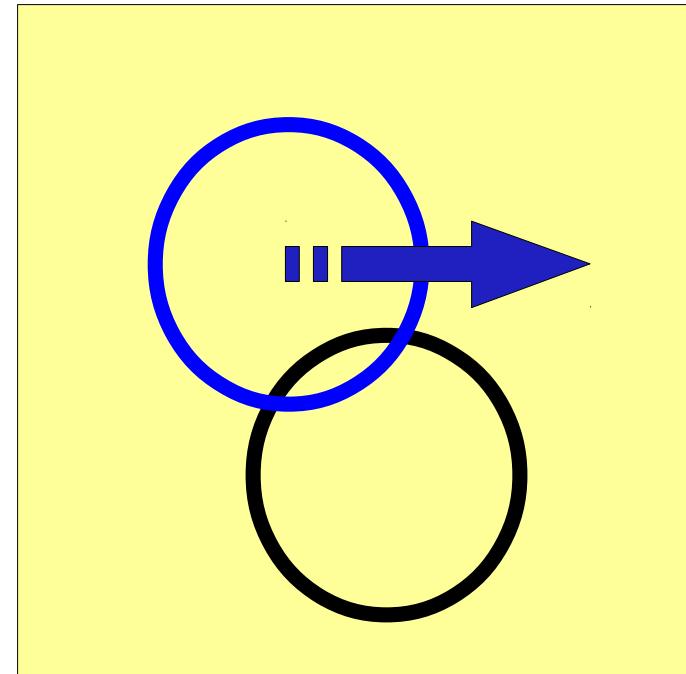
The Radon transform

- For each point in the parameter space, examine how much evidence there is in the image

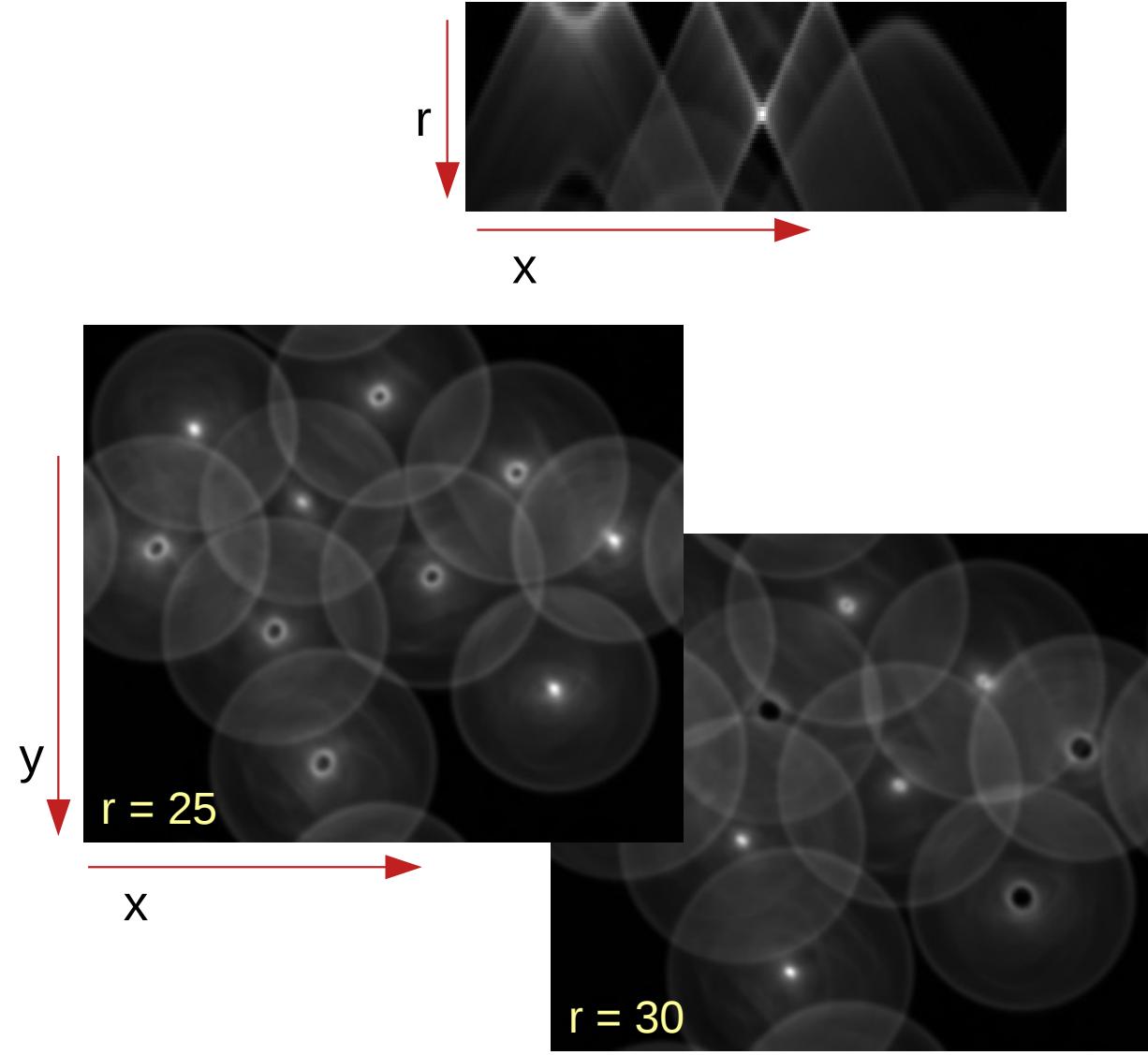
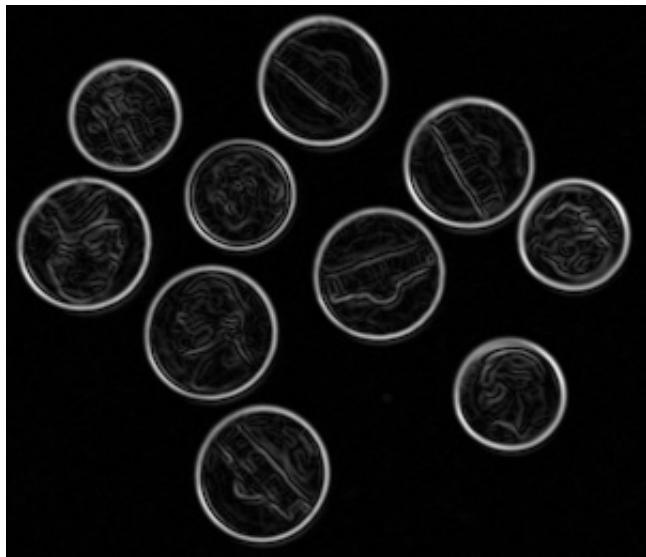


The Radon transform

- For each point in the parameter space, examine how much evidence there is in the image
- This can be done though one convolution for all values of the “position” or “distance” parameters
- This is identical to template matching!

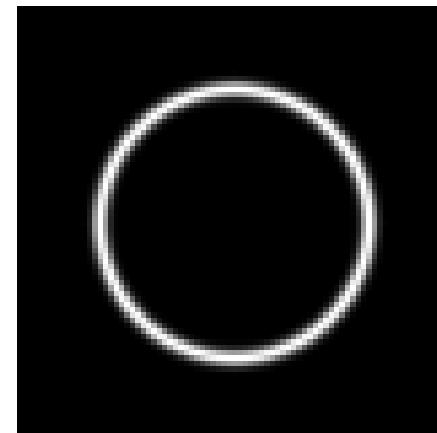


Radon transform example



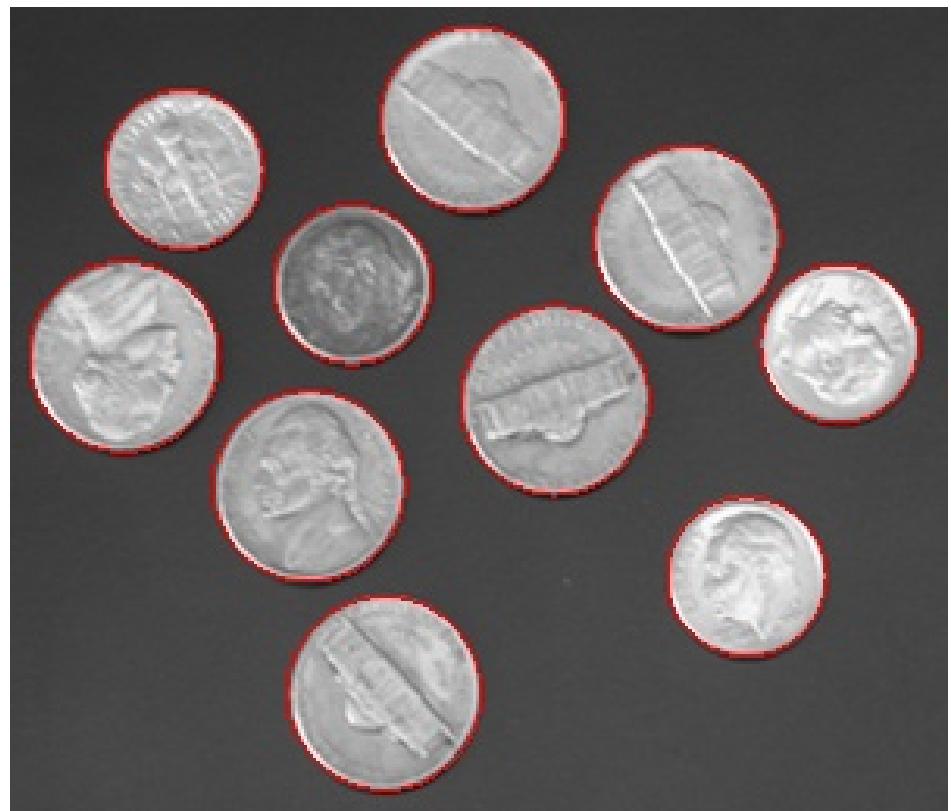
The Radon transform

- Because I'm using convolution to create the parameter space, I can use a grey-value convolution mask!
- Define the circle with a Gaussian profile:
 - Yields a band-limited parameter space
 - The parameter space can be sampled without aliasing
- It is possible to detect peaks with sub-pixel accuracy



Radon transform example

x	y	r
55.10	51.52	24.94
109.00	88.28	24.90
235.10	181.78	24.92
264.83	107.11	24.86
147.59	35.42	29.06
215.97	73.66	29.21
36.06	111.39	29.93
95.17	152.87	30.06
119.33	218.96	30.42
173.80	125.39	29.62



Hough vs Radon

- The resulting parameter space is identical
- Hough = write paradigm
- Radon = read paradigm
- Hough = efficient for binary input images
- Radon = efficient for grey-value input images
- Radon makes it easier to define the parameter space, and how to sample it

Template matching

- I said before: Radon = template matching
 - Linear filter
 - Template is convolution kernel
 - Convolution \approx correlation (mirror the kernel!)

$$g[n] = \sum_k f[n+k] h[k]$$

- Other ways of doing template matching:
 - Mean square error

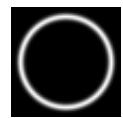
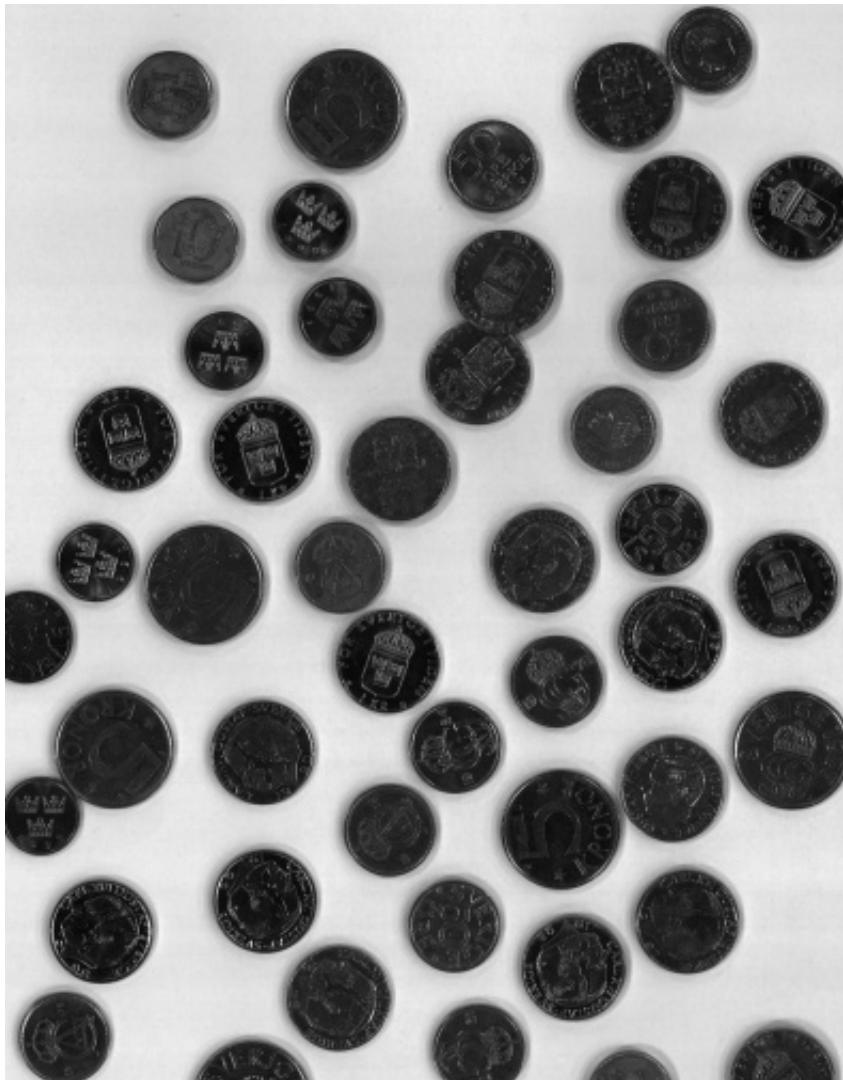
$$g[n] = \frac{1}{N} \sum_k (f[n+k] - h[k])^2$$

- Mean absolute error

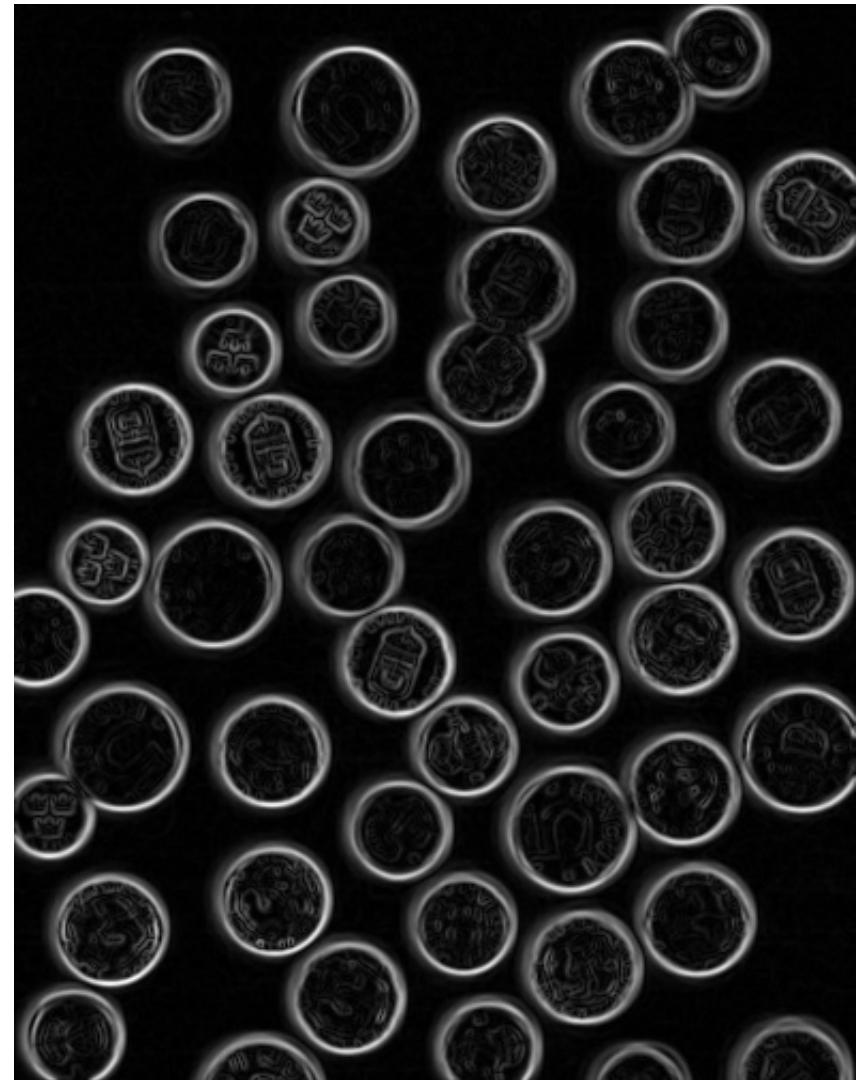
$$g[n] = \frac{1}{N} \sum_k |f[n+k] - h[k]|$$

- ...

Template matching

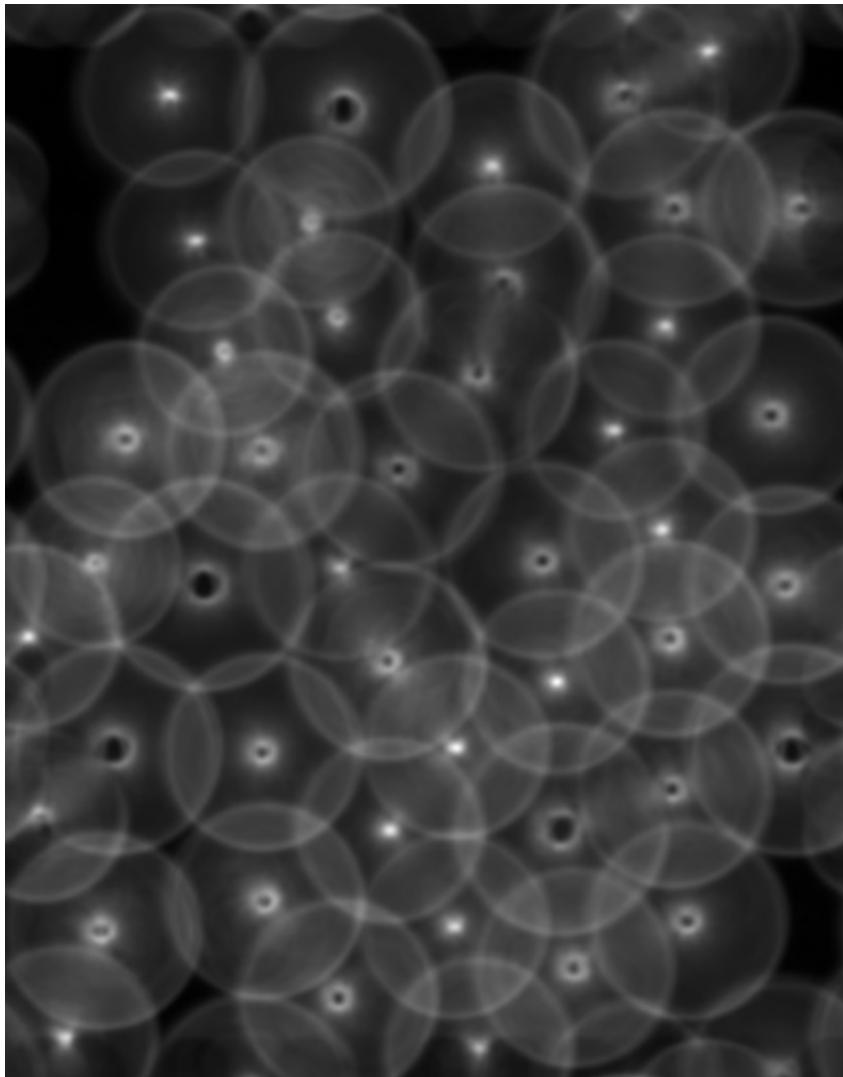


template: circle with radius 21 px

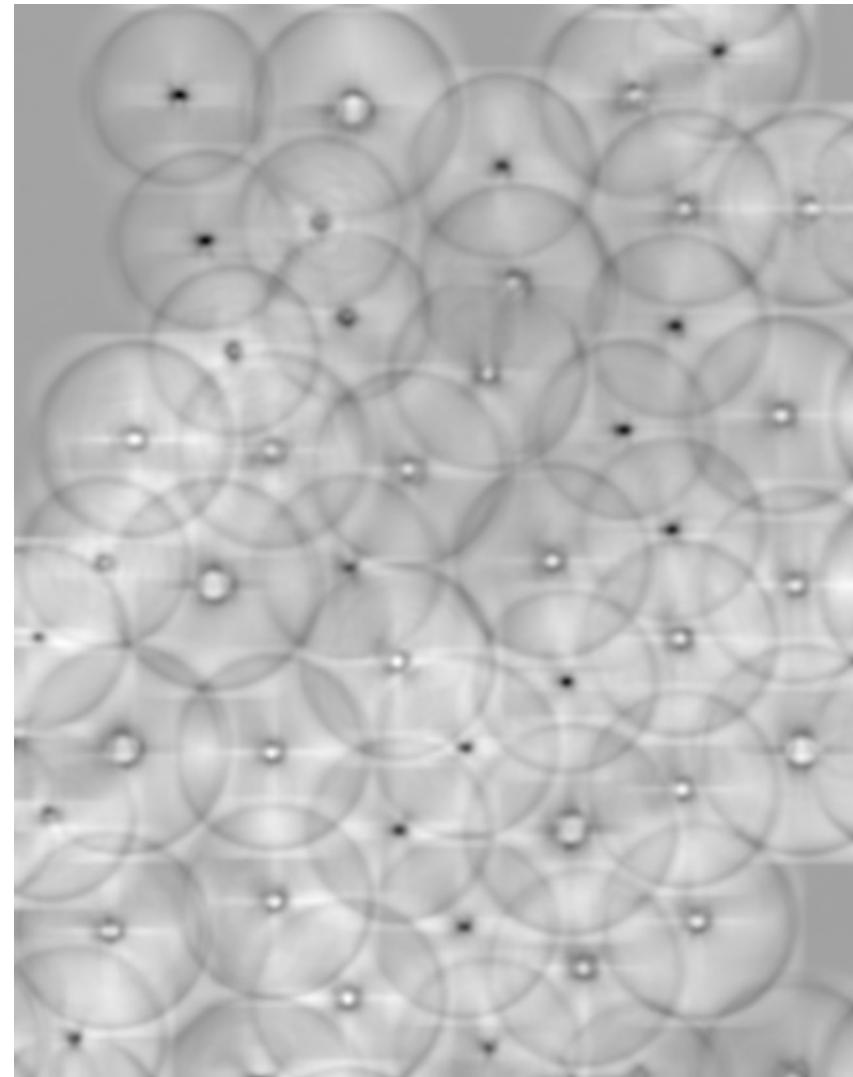


gradient magnitude

Template matching



cross-correlation



mean square error

Summary of today's lecture

- Special Applications of Filtering:
 - Scale spaces
 - Pattern detection: Gabor filter
 - Edge detection: Canny's edge detector
 - Hough transform
 - Radon transform
 - Template matching