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: $n$-D + scale = ($n$+1)-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

scale space

local maxima & minima

inflection points
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 FD

- We want the filter to be insensitive to phase
  - Use an even and an odd filter
Gabor filter

Gaussian

spatial domain

frequency domain

cosine
sine

Gabor filter
Gabor filter

Gaussian

Gaussian, shifted in the FD

real component

imaginary component

spatial domain

frequency domain

Gaussian,
shifted in the FD

real component

imaginary component
Gabor filter example
Gabor filter example
Gabor filter example

real component

imaginary component

magnitude
Gabor filter example
Gabor filter

smaller $\sigma \Rightarrow$ less specific

lower $\omega \Rightarrow$ needs larger $\sigma$
Edge detection

- 1\textsuperscript{st} 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

For each point $p$ check:

$$f(p) > a_1 \land f(p) \geq a_2$$

$$f(p) \geq a_1 \land f(p) > a_2$$

If condition not met, set pixel to 0.

$a_1$ & $a_2$ computed by linear interpolation
Canny’s edge detector

- low threshold
- high threshold
- hysteresis threshold
Canny’s edge detector
Influence of the $\sigma$

Use a scale space!
Hough transform

- Detecting parametrized shapes in an image
- Originally for straight lines in 2-D (2 parameters: \( p, \phi \))
- Later generalized to any shape:
  - 3-D sphere (4 parameters: \( x, y, z, r \))
  - 2-D square (4 parameters: \( x, y, a, \phi \))
  - 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

![Diagram of 2D Hough transform for lines](image)
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 through 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

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>55.10</td>
<td>51.52</td>
<td>24.94</td>
</tr>
<tr>
<td>109.00</td>
<td>88.28</td>
<td>24.90</td>
</tr>
<tr>
<td>235.10</td>
<td>181.78</td>
<td>24.92</td>
</tr>
<tr>
<td>264.83</td>
<td>107.11</td>
<td>24.86</td>
</tr>
<tr>
<td>147.59</td>
<td>35.42</td>
<td>29.06</td>
</tr>
<tr>
<td>215.97</td>
<td>73.66</td>
<td>29.21</td>
</tr>
<tr>
<td>36.06</td>
<td>111.39</td>
<td>29.93</td>
</tr>
<tr>
<td>95.17</td>
<td>152.87</td>
<td>30.06</td>
</tr>
<tr>
<td>119.33</td>
<td>218.96</td>
<td>30.42</td>
</tr>
<tr>
<td>173.80</td>
<td>125.39</td>
<td>29.62</td>
</tr>
</tbody>
</table>
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