Chapter 4 Linear Filters

Simple view:

- An image is a 2D array of discrete values
- A mask is given which is a \( 2k+1 \times 2k+1 \) array

\[
R(i,j) = \sum_{u=i-k}^{i+k} \sum_{v=j-k}^{j+k} H(u-(i-k)+1, v-(j-k)+1) F(u,v)
\]

The average value at a pixel is given by setting:

\[
H = \frac{\text{ones}(2k+1, 2k+1)}{(2k+1)^2}
\]

- Shift invariant: depends on region, not location
- Linear: \( \text{convolve} (F+G) = \text{conv} (F) + \text{conv} (G) \)

Linear filtering
window: kernel

blurring with Gaussian

$$G_r(x, y) = \frac{1}{2\pi \sigma^2} \exp \left( \frac{x^2 + y^2}{2\sigma^2} \right)$$

$\sigma$ is standard deviation units: pixels

discrete version

$$H_{k,j} = \frac{1}{2\pi \sigma^2} \exp \left( \frac{(i-k-1)^2 + (j-k-1)^2}{2\sigma^2} \right)$$

for many kernels, use Matlab fspecial

```matlab
H = fspecial('gaussian', 21, 2)
surf(H)
```
% looks like p.109

Derivatives

$$\frac{df}{dx} = \lim_{\varepsilon \to 0} \frac{f(x + \varepsilon, y) - f(x, y)}{\varepsilon}$$

Set $\varepsilon \approx 1$

$$f(x+1, y) - f(x, y)$$

better to use central symmetric difference:

$$f(x+1, y) - f(x-1, y)$$

+ use kernel $H = [0, 1, 0]$ (book has it backward)
There is a function to apply kernels to images:

\[ \text{filter2} (H, F) \]

Try on trees \( H = \begin{bmatrix} E & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \)

**Shift Invariant Linear Systems**

The book shows that the response of a system (camera) to any signal can be characterized by its response to a simple input (impulse response).

(Image processing goes deeper into the math of this and Fourier transforms.)

**Simple use of Fourier Transform**

Given a function \( f(x) \), it can be represented as a sum of basis functions: (sines, cosines) of various frequencies.

\[ f(x) = \sum \text{basis terms} \sin(t) + \sum \text{basis terms} \sin(2t) \]

The Fourier Transform (FT) helps find which functions contribute to the function (see Matlab).
FT can be zeroed out for higher frequencies in order to smooth the image (or signal).

**Filters as templates**

\[
\text{view as vector} \downarrow \quad \text{dot product}
\]

\[\Rightarrow \text{if } \vec{u} + \vec{w} \text{ are similar then:} \]

\[
\cos \theta = \frac{\vec{u} \cdot \vec{w}}{||\vec{u}|| ||\vec{w}||}
\]

- is near 1 when similar
- near -1 when opposite

See Matlab
Gaussian Pyramid

16 x 16

\( \frac{1}{2^0} \)

Smooth + \( \frac{1}{2^n} \)

re sample size

use nearest 2^n x 2^n