Computer Vision

Lectured by Wenjia Bai

Typed by Aris Zhu Yi Qing

March 17, 2022

Contents

1 Image Filtering

1.1 Definition

- **Kernel**: a small matrix used to apply effects, e.g. blurring.

- **Separable kernel**: kernels that can be separated as two or more simple filters.

- **Padding**: The action of adding pixels around the borders (e.g. with value 0) so that applying filters will not reduce the size of the image.

- **Low-pass (smoothing) filter**: filters that keep the low-frequency signals, e.g. MA filter

- **High-pass (sharpening) filter**: filters that highlight the high-frequency signals, e.g. (identity + (identity - MA)) filter, or

$$\begin{pmatrix} -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} \\ -\frac{1}{8} & 2 & -\frac{1}{8} \\ -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} \end{pmatrix}.$$

- **Denoising filter**: filters to remove noise, e.g. median filter, non-local means, block-matching and 3D filtering (BM3D), etc.

1.2 Common Filters

1.2.1 Moving Average (MA) Filter

- In a 2D case, the MA kernel is a $\mathbb{R}^{K\times K}$ matrix in the following form

$$\frac{1}{K^2}\begin{pmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{pmatrix}$$

  with a time complexity of $O(N^2K^2)$, where N is the legnth of image.

- MA kernel is separable, for instance

$$\begin{pmatrix} \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \end{pmatrix} * \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix},$$

  reducing the time complexity to $O(N^2K)$.

- Purpose:
  - remove high-frequency signal (noise or sharpness)
  - result in a smooth but blurry image

1.2.2 Identity Filter

The identity filter kernel is

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

1.2.3 Gaussian Filter

- The Gaussian kernel is a 2D Gaussian distribution

$$h(i,j) = \frac{1}{2\pi\sigma^2}e^{-\frac{i^2+j^2}{2\sigma^2}}$$

  with $i,j=0$ as the centre of the kernel.

- While its support is infinite, small values outside $[-k\sigma, k\sigma]$ can be ignored, e.g. $k=3$ or $k=4$.

- 2D Gaussian filter is separable with

$$h(i,j) = h_x(i) * h_y(j)$$

  where

$$h_x(i) = \frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{i^2}{2\sigma^2}},$$

  because

$$\begin{aligned} f[x,y] * h[x,y] &= \sum_i \sum_j f[x-i, y-j]h[i,j] \\ &= \sum_i \sum_j f[x-i, y-j] \left(\frac{1}{2\pi\sigma^2}e^{-\frac{i^2+j^2}{2\sigma^2}} \right) \\ &= \sum_i \left(\sum_j f[x-i, y-j] \frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{j^2}{2\sigma^2}} \right) \frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{i^2}{2\sigma^2}} \\ &= \sum_i (f * h_y)[x-i] \frac{1}{\sqrt{2\pi}}e^{-\frac{i^2}{2\sigma^2}} \\ &= (f * h_y) * h_x \end{aligned}$$

- Derivative of Gaussian filter h is

$$\frac{\mathrm{d}(f*h)}{\mathrm{d}x} = f * \frac{\mathrm{d}h}{\mathrm{d}x} = f * \frac{-x}{\sqrt{\pi}\sigma^3}e^{-\frac{x^2}{2\sigma^2}}.$$

  Thus, the smaller the σ, the more detail in the magnitude map; larger σ suppresses noise and results in a smoother derivative. Different σ help find edges at different scale.

1.2.4 Median Filter

- non-linear
- often used for denoising
- Move the sliding window, and replace the centre pixel using the median value in the window.

1.3 Impulse Response

- For continuous signal, an **impulse** is a Dirac delta function $\delta(x)$, with

$$\delta(x) = \begin{cases} +\infty, & \text{if } x = 0 \\ 0, & \text{otherwise} \end{cases}$$

such that $\int_{-\infty}^{\infty} \delta(x)\mathrm{d}x = 1$. For discrete signal, an impulse is a Kronecker delta function $\delta[i]$, with

$$\delta[i] = \begin{cases} 1, & \text{if } i = 0 \\ 0, & \text{otherwise.} \end{cases}$$

- The **impulse response** h is the output of a filter when the input is an impulse. It completely characterises a **linear time-invariant** filter.
  - shifting the input signal k steps corresponds to the same output signal but shifted by k steps as well, e.g. assuming $f[n] = \delta[n]$, $g[n] = h[n]$,
    - $g[n] = 10f[n]$ is time-invariant and amplifies the input by a constant.
    - $g[n] = nf[n]$ is *not* time-invariant since the amount it amplies the input depends on the
  - if input $f_1[n]$ leads to $g_1[n]$, $f_2[n]$ leads to $g_2[n]$, we will have

$$\text{output}(\alpha f_1[n] + \beta f_2[n]) = \alpha g_1[n] + \beta g_2[n].$$

1.4 Convolution

- **Convolution**: output g can be described as the convolution between an input f and impulse response h as

$$g[n] = f[n] * h[n] = \begin{cases} \displaystyle\sum_{m=-\infty}^{\infty} f[m]h[n-m] & \text{discrete} \\ \displaystyle\int_{m=-\infty}^{\infty} f(m)h(n-m) & \text{continuous} \end{cases}$$

- Note that if we describe input signal $f[n]$ as

$$f[n] = \sum_{i=0}^{n} f[i]\delta[n-i]$$

and we known the output of $\delta[n]$ is $h[n]$, we can write the output as

$$g[n] = \sum_{i=0}^{n} f[i]h[n-i]$$

- commutative, i.e.

$$f[n] * h[n] = h[n] * f[n]$$

- associative, i.e.

$$f * (g * h) = (f * g) * h$$

- distributivity, i.e.

$$f * (g + h) = f * g + f * h \quad \text{and} \quad \frac{\mathrm{d}(f * g)}{\mathrm{d}x} = \frac{\mathrm{d}f}{\mathrm{d}x} * g = f * \frac{\mathrm{d}g}{\mathrm{d}x}$$

- In 2D discrete case for image filtering,

$$\begin{aligned} g[m,n] = f[m,n] * h[m,n] &= \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f[i,j]h[m-i,n-j] \\ &= \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f[m-i,n-j]h[i,j] \end{aligned}$$

if the dimension of the kernel is $(2M+1) \times (2N+1)$, we can write

$$(f * h)[m,n] = \sum_{i=-M}^{M} \sum_{j=-N}^{N} f[m-i,n-j]h[i,j]$$

with $h[0,0]$ being the centre of the filter, (m,n) being the location in the image which the kernel's center is on.

- If a big filter f_b can be separated into convolution g and h, we can first convolve with g, then h

$$f * f_b = f * (g * h) = (f * g) * h.$$

2 Edge Detection

2.1 Detection

	finite difference	convolution kernel
Forward difference	$f'[x] = f[x+1] - f[x]$	$[1, -1, 0]$
Backward difference	$f'[x] = f[x] - f[x-1]$	$[0, 1, -1]$
Central difference	$f'[x] = (f[x+1] - f[x-1])/2$	$[1, 0, -1]$

2.2 Edge Detection Filters

2.2.1 Prewitt Filter

Along the x-axis and the y-axis, we have respectively

$$\begin{pmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}.$$

They are separable, i.e.

$$\begin{pmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} * \begin{pmatrix} 1 & 0 & -1 \end{pmatrix}.$$

2.2.2 Sobel Filter

Along the x-axis and the y-axis, we have respectively

$$\begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}.$$

They are also separable, i.e.

$$\begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} * \begin{pmatrix} -1 & 0 & -1 \end{pmatrix}$$

2.2.3 Magnitude and Orientation Calculation

Let h_x denotes the horizontal filter, h_y denotes the vertical filter, we can compute the magnitude and the orientation as

$$\begin{aligned} g_x &= f * h_x && \text{derivative along } x\text{-axis} \\ g_y &= f * h_y && \text{derivative along } y\text{-axis} \\ g &= \sqrt{g_x^2 + g_y^2} && \text{magnitude of the gradient} \\ \theta &= \arctan(g_y, g_x) && \text{angle of the gradient} \end{aligned}$$

2.3 Canny Edge Detection

2.3.1 Criteria for Good Edge Detector

- good detection: low probability of FP/FN on marking edge points
- good localisation: mark as close as the centre of true edge
- single response: only one response to a single edge

2.3.2 Algorithm

1. perform Gaussian filtering to suppress noise
2. calucalte the gradient magnitude $M(x, y)$ and direction
3. apply Non-Maximum Suppression (NMS) to get a single response for each edge

$$M(x,y) = \begin{cases} M(x,y) & \text{if local maximum} \\ 0 & \text{otherwise} \end{cases}$$

4. perform hyteresis thresholding to find potential edges with two thresholds t_{low} and t_{high}

$$\begin{cases} M(x,y) \geq t_{\text{high}} & \text{accept} \\ M(x,y) < t_{\text{low}} & \text{reject} \\ \text{Otherwise} & \text{iteratively check neighbouring pixels and accept if connected to an edge pixel.} \end{cases}$$

5. evaluation/performance
    - good detection — FP reduced by Gaussian smoothing and FN reduced by hysteresis thresholding to find weak edges
    - good localisation — NMS finds locations based on gradient magnitude and direction
    - single response — NMS finds one single point in the neighbourhood

3 Hough Transform

- **Hough transform** is a transform from image space to parameter space, e.g. from an edge map to the two parameters of a line.
- output is a parametric model, given the input edge points
- each edge point vote for possible models in the parameter space
- Example:
    - use slope intercept $b = y - mx$ to be the line model
    - each edge point poll vote for different b and m values (also lines in parameter space)
    - In practice, we use 2D bins to divide the parameter space; each point increases the vote by 1 in one of the bins, as shown in figure 1.
    - But the parameter space could be too large, $[-\infty, +\infty]$; use *normal form* instead

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

    so at least for one dimension, $\theta \in [0, \pi)$.

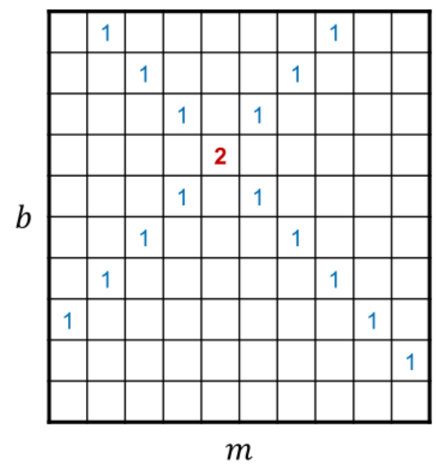

Figure 1: hough transform grid

- Line detection by Hough transform:
    1. Initialize the bins $H(\rho, \theta)$ to all zeros.
    2. For each (x, y) and each θ, calculate ρ and $H(\rho, \theta) + +$.
    3. Find (ρ, θ) where $H(\rho, \theta)$ is a local maximum and larger than a threshold.
        - local maximum so there can be multiple solutions, to reduce FN
        - larger than a threshold so as to reduce FP
    4. The detected lines are given by $\rho = x \cos\theta + y \sin\theta$.
- robust to noise/occlusion
    - edge map is often generated after image smoothing
    - broken/unoccluded edge points can still vote and contribute to line detection
- Circle detection by Hough transform
    - parameter space also forms circles
    - If radius r is also unknown, then 3D parameter space $H(a, b, r)$
    - parameterize to be $x = a + r\cos\theta$ and $y = b + r\sin\theta$, since we know θ form edge detection, we can narrow the voting area to move along θ for a distance r.
- Pros and Cons:
    + detects multiple instances

+ robust to image noise
+ robust to occlusion
- Computational complexity is quite high. For each edge point, we need to vote to a 2D or even 3D parameter space.
- need to carefully set parameters such as those for edge detectors, the threshold for the accumulator or range of circle radius.

4 Interest Point Detection

4.1 Definition

- **Interest point**: points that we are interested in and are useful for subsequent image processing and analysis. They are also called *key-points*, *landmarks*, *low-level* features.