Imperial College London

60006 Computer Vision (Term 2)

Tutorial 1: Image Formation, Image Filtering

1. Let (x, y) be the coordinate of a point in an image. Its homogeneous coordinate $(x, y, 1)$ is often used in computer vision or graphics, for example, transforming an image by scaling, translation, rotation etc. The transformation can be described by the following equation,

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = A \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

where $(x, y, 1)$ and $(x', y', 1)$ denote the coordinate before and after transformation, A is a 3x3 transformation matrix. For the following examples of A, describe what kind of transformation it performs.

1.1 Example 1

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 5 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Solution:

This is a translation operation. It translates the image along the x-axis by 10 pixels and along the y-axis by 5 pixels. The pixel (10, 5) in the new image corresponds to the pixel (0, 0) in the old image.

1.2 Example 2

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Solution:

This is a scaling operation. It zooms the image by 5-fold. The pixel (5, 5) in the new image corresponds to the pixel (1, 1) in the old image.

2. For a RGB image, at each pixel, there are intensity values for three channels (R: red; G: green; B: blue). For example, [255, 0, 0] represents pure red. [0, 255, 0] represents pure green. [0, 0, 255] represents pure blue. Each channel is represented by an integer between 0 and 255, i.e. an 8-bit unsigned char.

   2.1 For a RGB image of size 1280x960 pixels, without image compression, how many bytes are needed for storing the image?

Solution:

Each pixel requires 3 bytes for storage, so for all the pixels, we require

$$1280 \times 960 \times 3 \text{ byte} \approx 3.69 \text{ MB}$$

   2.2 There are algorithms to convert a RGB image into a grayscale image, where each pixel only has one intensity value, which represents the brightness. For example, one algorithm is recommended by ITU-R Recommendation BT.601, which is formulated as the following equation,

$$Y = 0.299R + 0.587G + 0.114B$$

Could you work out what grayscale values pure red, pure green and pure blue respectively convert to?

Solution:

The grayscale intensity for pure red is

$$Y = 0.299 \times 255 + 0.587 \times 0 + 0.114 \times 0 = 76.2$$

For pure green, it is

$$Y = 0.299 \times 0 + 0.587 \times 255 + 0.114 \times 0 = 149.7$$

For pure blue, it is

$$Y = 0.299 \times 0 + 0.587 \times 0 + 0.114 \times 255 = 29.1$$

3. Suppose that we have an image that is corrupted by Gaussian white noise $Y = I + n$, where I denotes the clean image, $n \sim N(0, \sigma^2)$ denotes the Gaussian white noise and Y denotes the corrupted image. Perform image filtering using a low-pass filter is one approach for denoising, but it may also result in loss of fine details. Another approach is to denoise to take a lot of images of the same object and combine them. For example, you can take a lot of pictures of the moon in the sky from the same angle. Each image is described by $Y_i = I + n_i$ $(i = 1, 2, \dots, N)$, where n_i is one independent realisation of the noise. Then you can take the average, $Y = \frac{1}{N}\sum_{i=1}^{N} Y_i$. Please derive the mean and variance of the noise in the combined image Y.

Solution:

The noise in the combined image is described by,

$$e = Y - I = \frac{1}{N}\sum_{i=1}^{N} n_i$$

The mean is calculated by,

$$E[e] = \frac{1}{N}\sum_{i=1}^{N} E[n_i] = 0$$

The variance is calculated by,

$$\begin{aligned} var[e] &= E[(e - E[e])^2] \\ &= \frac{1}{N^2} E\left[\sum_{i=1}^{N} n_i^2 + \sum_{i=1}^{N}\sum_{j \neq i} n_i n_j\right] \\ &= \frac{1}{N}\sigma^2 \end{aligned}$$

The standard deviation is

$$std[e] = \frac{1}{\sqrt{N}}\sigma$$

4. Consider a 2D Gaussian filter

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

4.1 Derive its first derivate $\frac{\partial h}{\partial x}$ and $\frac{\partial h}{\partial y}$.

Solution:

$$\frac{\partial h}{\partial x} = -\frac{x}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$\frac{\partial h}{\partial y} = -\frac{y}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

4.2 The Gaussian filter has an infinitive support. People often truncate and approximate the filter. For example, the following 3x3 filter approximates the 2D Gaussian filter,

1/36	1/9	1/36
1/9	4/9	1/9
1/36	1/9	1/36

The following filter approximates the 1D Gaussian filter,

1/6	2/3	1/6

Please check that whether the 2D Gaussian filter is equivalent to the convolution of a 1D Gaussian filter along x-axis and a 1D Gaussian filter along y-axis.

Solution:

The 1D Gaussian filters along the x-axis and y-axis are

1/6	2/3	1/6

and

1/6
2/3
1/6

After performing the convolution, we will see that it is equivalent to the 2D Gaussian filter. This indicates that the 2D Gaussian filter is separable as two small 1D Gaussian filters.

If it is not obvious to perform 2D convolution, you can treat the Gaussian filters along the x-axis and y-axis respectively as

0	0	0
1/6	2/3	1/6
0	0	0

and

0	1/6	0
0	2/3	0
0	1/6	0

Then move the y-axis Gaussian filter on the x-axis Gaussian filter pixel by pixel and work out the convolution.

4.3 What is the computational cost when we convolve an NxN image with a 3x3 2D Gaussian filter?

Solution:

$9N^2$ multiplications + $8N^2$ additions

4.4 What is the computational cost when we convolve an NxN image with two 1D Gaussian filters, respectively with size 1x3 and 3x1?

Solution:

$6N^2$ multiplications + $4N^2$ additions

4.5 In general, what is the computational cost when we convolve an NxN image with a KxK 2D Gaussian filter? Use the big O notation.

Solution:

K^2N^2 multiplications + $(K^2 - 1)N^2$ additions

Using the big O notation, the cost is $O(K^2N^2)$.

4.6 What is the computational cost when we convolve an NxN image with two 1D Gaussian filters, respectively with size 1xK and Kx1?

Solution:

$2KN^2$ multiplications + $2(K - 1)N^2$ additions

Using the big O notation, the cost is $O(KN^2)$.