60006 - Tutorial 1
Image Formation, Image Filtering

Xin Wang

January 21, 2022

Question 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:

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

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:

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

Calculating the following equation results in:

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

This corresponds to a *translation transformation* that shifts a pixel 10 in the x-axis and 5 in the y-axis.

1.2: Example 2:

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

Calculating the following equation results in:

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

This corresponds to a *scaling transformation* that zooms the image 5 in the x-axis and 5 in the y-axis i.e. zooming the image 5 times.

Question 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 1280×960 pixels, without image compression, how many bytes are needed for storing the image?

In a 1280×960 image, there are 1228800 pixels. Each pixel needs to represent a three channels which is represented by 8×3 bits. In totals there are:

$$
\begin{aligned}
1228800 \text{ pixels} \times 24 &= 29491200 \text{ bits} \\
&= 3686400 \text{ bytes}
\end{aligned}
$$

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?

Pure Red: RGB $= [255, 0, 0]$ and grayscale is $[76.245, 0, 0]$ Pure Green: RGB $= [0, 255, 0]$ and grayscale is $[0, 149.685, 0]$ Pure Blue: RGB $= [0, 0, 255]$ and grayscale is $[0, 0, 29.07]$

Question 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.

Performing 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 denoising is to take a lot of images of the same object and then 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 $Y_i = I + n_i (i = 1, 2, ..., N)$, 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.

The noise in the combined image e is defined as:

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

Since each independent realisation of noise is of type Gaussian white noise, the mean of the noise in the comnbined image is:

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

The variance is calculated by:

$$\begin{aligned} var[e] &= E[(e - E[e])^2] \\ &= \frac{1}{N}\sigma^2 \end{aligned}$$

Question 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 the first derivative: $\frac{\partial h}{\partial x}$ and $\frac{\partial h}{\partial y}$

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

4.2: The Gaussian filter has an infinitive support. People often truncate and approximate the filter.

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

1D Gaussian filter along x-axis is $[1/6, 2/3, 1/6]$ and along y-axis is $\begin{bmatrix} 1/6 \\ 2/3 \\ 1/6 \end{bmatrix}$. It is equivalent to the 2D Gaussian filter.

4.3: What is the computational cost when we convolve an $N \times N$ image with a 3x3 2D Gaussian filter?

- At each pixel, there are $K \times K$ (9) multiplications and $K^2 - 1$ (8) summations
- In total, there are N^2K^2 multiplications and $N^2(K^2 - 1)$ summations
- Complexity is $O(N^2K^2)$

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

- At each pixel, there are K multiplications and $K - 1$ summations
- In total, there are $2N^2K$ multiplications and $2N^2(K - 1)$ summations
- Complexity is $O(N^2K)$

4.5: In general, what is the computational cost when we convolve an $N \times N$ image with a $K \times K$ 2D Gaussian filter? Use the big O notation.

- At each pixel, there are $K \times K$ multiplications and $K^2 - 1$ summations
- In total, there are N^2K^2 multiplications and $N^2(K^2 - 1)$ summations
- Complexity is $O(N^2K^2)$