MCTE 4323
Assignment
Zhao Yaqi
1739844

#1

a

In convolution, a small matrix called a kernel is moved over the image, with each step performing element-wise multiplication between the kernel and the image pixels, followed by a sum to obtain a single output pixel. This results in the creation of a new image, where the values of the output pixels are influenced by the values of the input pixels and the values of the kernel. Convolution is commonly used for tasks such as edge detection, blurring, and sharpening.

Correlation, on the other hand, is the operation of comparing two signals to find the similarity between them. It is similar to convolution, except the kernel is not flipped before being multiplied with the image pixels. Correlation is mostly used for pattern recognition tasks, such as template matching.

Convolution is generally preferred over correlation in image processing and recognition because convolution can process an image in a more dynamic and flexible way, as it allows for the creation of custom kernels to perform specific tasks. In mathematical terms, the associativity of convolution can be expressed as: $(f * g) * h = f * (g * h)$. This property allows for simplification and optimization of image processing algorithms that involve multiple convolutions. For example, if an image needs to be processed with three different kernels, the associativity of convolution allows for the convolutions to be performed in any order, which can reduce computation time and improve performance.

In practice, associativity of convolution is an important property in image processing, and it has been widely used in computer vision and image recognition algorithms.

b

The associativity of convolution states that: $(f * g) * h = f * (g * h)$, where f, g, and h are functions and * represents the convolution operation. This means that convolving an image with filter H1 and then convolving the result with filter H2 gives the same result as when you first convolve with H2 and then with H1.

c

Padding: Padding involves adding additional pixels to the edges of the image to maintain the same size after the convolution operation. The added pixels are typically set to a default value such as 0. For example, if we have an image of size 6x6 and a convolutional kernel of size 3x3, we can pad the image with zeros so that it becomes 8x8. Then, we can perform the convolution operation, which results in an output image of size 6x6.

Using a kernel size of 1x1 is another technique that can be used to ensure that the output size of a convolution operation is the same as the input size. A 1x1 convolutional kernel has the effect of performing a linear combination of the values in a small neighbourhood of the image. This linear combination is performed by element-wise multiplication between the kernel values and the pixel values in the neighbourhood, followed by summation. Since the size of the kernel is 1x1, the output size will be the same as the input size, assuming the stride is set to 1.

d

The equation of a line, y = mx + c, is a linear equation that represents a straight line in the Cartesian coordinate system. However, when used in the Hough parameter space, it can lead to a mathematical flaw. In the Hough parameter space, lines are represented by points (m, c) in a two-dimensional space. The Hough transform is used to detect lines in an image by converting the image into this parameter space and searching for peaks in the distribution of points. The flaw in the equation y = mx + c arises when it is used to represent lines that are nearly vertical. For nearly vertical lines, the slope m becomes very large, which can cause numerical instability and lead to inaccurate results. This is because the slope m becomes very sensitive to small changes in x, which can result in significant changes in the value of y. To overcome this mathematical flaw, the Hough transform is often modified to use a different representation for lines, such as the polar representation, which is less sensitive to numerical instability when representing nearly vertical lines. The polar representation uses the angle and distance of a line from the origin, rather than its slope and y-intercept. This representation is more stable and accurate, even for lines that are nearly vertical.

e

The first step in the Canny edge detection process is to apply Gaussian filtering to the input image to reduce noise. If Gaussian filtering is not applied before the Canny edge detection process, the output will likely be affected by noise and other unwanted artefacts. The presence of noise can cause false edges to be detected, making the output image difficult to interpret and less useful for further processing. Without Gaussian filtering, the Canny edge detector may also detect edges that are not real, such as small isolated fluctuations in the image intensity. These false edges can negatively impact the accuracy of other image processing and recognition tasks that rely on the output of the Canny edge detector. This seems to happen in image D.

Image E seems to detect the correct edges, but the edges are very thick. The edges in the output of the Canny edge detector can appear too thick or too broad if the parameters of the algorithm are not properly set. This can occur in several ways:

- Low threshold values: The Canny edge detector uses two threshold values, a low and a high threshold, to determine which edges to keep and which to discard. If the low threshold is set too low, many edges will be detected, including edges that are not real or that are only partially visible. This can cause the edges to appear too thick.
- High threshold values: If the high threshold is set too high, few edges will be detected, and the edges that are detected will be broad and indistinct. This can also cause the edges to appear too thick.
- Large kernel size for Gaussian filter: The Canny edge detector requires that the input image be smoothed using a Gaussian filter before edge detection. If the size of the Gaussian filter kernel is too large, it can cause the edges to appear too thick by blurring the edges and reducing the contrast between adjacent pixels.
- Unsuitable image size or resolution: The size and resolution of the input image can also impact the appearance of the edges in the output of the Canny edge detector. If the image is too small or has low resolution, the edges may appear too thick because the edges will not be well defined.

Image C seems like the correct answer but can be improved.

#2

a)

$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ 0 & 0 & P_{33} \end{bmatrix} \begin{bmatrix} X \\ y \\ 1 \end{bmatrix}$$

b)

$$P = \begin{bmatrix} 5 & -14 & 2 & 17 \\ -10 & -5 & -10 & 50 \\ 10 & 2 & -11 & 19 \end{bmatrix} \qquad X = \begin{bmatrix} 0 \\ 4 \\ 8 \\ 1 \end{bmatrix}$$

i) $\left(\frac{0}{1} , \frac{4}{1} , \frac{8}{1} \right) = (0, 4, 8)$

ii) $\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = PX = \begin{bmatrix} -23 \\ -50 \\ -61 \end{bmatrix}$

$u = \frac{-23}{-61} = \frac{23}{61}$

$v = \frac{-50}{-61} = \frac{50}{61}$

$(u, v) = \left(\frac{23}{61} , \frac{50}{61} \right)$

c

An affine transformation is appropriate for viewing a planar scene under certain conditions. These conditions include:

- Planar scenes: An affine transformation is most appropriate when the objects in the scene lie on a single plane. This is because an affine transformation maps lines to lines, preserving their relative positions and angles.
- Orthographic views: An affine transformation is appropriate when the desired view of the scene is an orthographic projection, such as a top-down view or a side view. In this type of view, parallel lines in the scene remain parallel in the output image.
- Isometries: An affine transformation is appropriate when the desired view of the scene is an isometry, such as a rotation or a translation. Isometries preserve distances and angles, so the objects in the scene retain their relative sizes and shapes in the output image.
- Linear deformations: An affine transformation is appropriate when the desired view of the scene involves linear deformations, such as scaling, shearing, or reflection. These types of transformations can be performed using an affine matrix.

In general, an affine transformation is appropriate when the desired view of the scene involves linear transformations that preserve the relative positions and angles of the objects in the scene.

d

The degrees of freedom to solve for in an affine camera model (a) depend on the number of parameters in the model. An affine camera model has 6 degrees of freedom: 3 for rotation and translation, and 3 for scaling and shearing. To estimate the calibration parameters, the minimum number of calibration points needed is 3. This is because an affine transformation can be determined by 3 non-collinear points in the world and their corresponding projections in the image. However, it is recommended to use more than 3 points for a more accurate and robust estimation of the parameters.