AVD624 - Computer Vision

Assignment

Quiz II Solutions

Submitted by

Vatsalya Gupta
SC19B098
B.Tech. ECE VI Semester

Department of Avionics
Indian Institute of Space Science and Technology
Thiruvananthapuram - 695 547
April 2022

Question - 1

Fourier Transform and image matching.

(a) Suppose you have 3 grayscale images I_1, I_2 and I_3, all of the same size. You would like to determine whether I_1 is more similar to I_2 or I_3. Write two different ways that you know for matching images.

(b) Write important properties of Fourier Transform. For the images given below draw the most likely Fourier Transform magnitude spectrum plot.

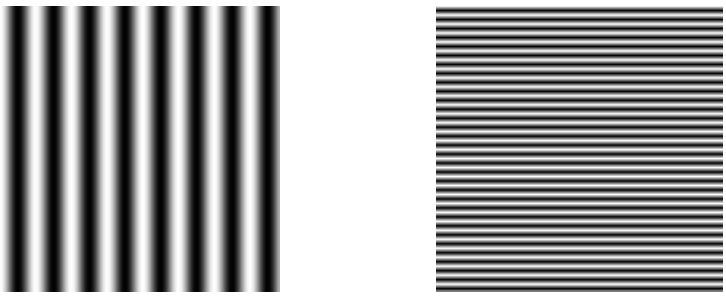

Figure 1: Two images. Plot Fourier magnitude spectrum (approximately).

(c) Suppose I is an image of size $N \times M$ and suppose that $\hat{I}$ is the Fourier Transform of image I. If $\hat{I}(0,0) = 1$ and $\hat{I}(k,l) = 0$ for $(k,l) \neq (0,0)$. Give the most precise description of image I. Justify your answer.

(a) There are several methods for finding similarity between images. We can use template matching techniques, such as comparing normalised cross correlation or sum of squared differences. Other techniques, such as assigning a score to feature descriptors using SIFT, can be used to determine whether a given image is more similar to one image or another.

(b) The Fourier transform magnitude spectra for the given images are as follows:

Figure 2: Fourier transform magnitude spectrum for the given images.

(c) Since the Fourier transform of the image has only a single frequency component and no high frequency content. So, the image can be of a flat region with each pixel having an intensity of $1/(N \times M)$ (to normalise the Fourier transform magnitude to 1).

Question - 2

Edge Detection, RANSAC and Hough Transform.

(a) Explain the algorithm of Hough transform for detecting lines.

(b) RANSAC is a powerful method for a wide range of model fitting problems as it is easy to implement. In class, we've seen how RANSAC can be applied to fitting lines. However, RANSAC can handle more complicated fitting problems as well, such as fitting circles. In this problem we will solve for the steps needed to fit circles with RANSAC.

(i) What is the minimum number of points we must sample in a seed group to compute an estimate for a uniquely defined circle?

(ii) If we obtain a good solution that has few outliers, we want to refit the circle using all of the inliers, and not just the seed group. For speed purposes, we would like to keep the same center of the estimated circle from the seed group, and simply refit the radius of the circle. The equation for our circle is given as:

$$(x - x_c)^2 + (y - y_c)^2 = r^2$$

where (x_c, y_c) are the coordinates of the center of the circle, and r is the radius. The error we would like to minimize is given by:

$$E(x_c, y_c, r) = \sum_{i=1}^{m} (\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} - r)^2$$

where m is the number of inliers. Derive the new radius r that minimizes this least squares error function.

(iii) One of the benefits to RANSAC is that we are able to calculate the failure rate for a given number of samples. Suppose we know that 30% of our data is outliers. How many times do we need to sample to assure with probability 20% that we have at least one sample being all inliers? You can leave your answer in terms of log functions.

(a) We use the polar parametrisation form: $xcos\theta - ysin\theta = d$. Then, follow the below steps.

1. Initialize $H[d, \theta] = 0$.
2. for each edge point $I[x, y]$ in the image
   for $\theta = [\theta_{min}$ to $\theta_{max}]$ // some quantisation
   $d = xcos\theta - ysin\theta$
   $H[d, \theta] += 1$.
3. Find the value(s) of (d, θ) where $H[d, \theta]$ is maximum.
4. The detected line in the image is given by $d = xcos\theta - ysin\theta$.

(b) (i) There are 3 degrees of freedom in a circle. These include the coordinates (x, y) of the centre and the radius r. Therefore, we need at least 3 points to compute an estimate for a uniquely defined circle.

(ii) We need to minimise the least squares error function, so let us take the partial derivative with respect to r.

$$\frac{\partial E}{\partial r} = -2\sum_{i=1}^{m}(\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} - r) = 0$$

$$\implies Radius\ (r) = \frac{1}{m}\sum_{i=1}^{m}(\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2})$$

(iii) We know that $1 - (1 - (1 - e)^s)^N = p$. Here $e = 0.3, s = 3$ and $p = 0.2$. So we get,

$$1 - (1 - (1 - 0.3)^3)^N = 0.2 \implies N = \frac{log(1 - 0.2)}{log(1 - (0.7)^3)} = 0.5312$$

i.e. we need to sample only once to get the required inlier probability.

Question - 3

Segmentation.

(a) Explain the K-means clustering algorithm. Describe how it can be used in segmentation.

(b) Explain the working of Otsu's image threshold algorithm. Histogram of a 3 bit image is shown in the following table

Gray level	0	1	2	3	4	5	6	7
Number of pixels	2	3	2	1	2	3	2	1

Find optimal threshold using Otsu.

(a) The K-means is a clustering algorithm. Clustering is the problem of grouping similar data points and represent them with a single token. It involves organising data into classes such that there is high intra-class similarity and low inter-class similarity. The class labels and the number of classes are directly found from the data (as opposed to classification tasks). The outline of the K-means algorithm in segmentation is as follows:

1. Pick K points as the initial centroids from the dataset, either randomly or any K.
2. Find the Euclidean distance of each point in the dataset with the identified K points (cluster centroids).

3. Assign each data point to the closest centroid using the distance found in the previous step.
4. Find the new centroid by taking the average of the points in each cluster group.
5. Repeat 2 to 4 for a fixed number of iteration or till the centroids don't change.

(b) Otsu's algorithm exhaustively searches for the threshold that minimises the intra-class variance ($\sigma_w^2(t)$) or maximises the inter-class variance ($\sigma_b^2(t)$). The algorithm can be can be summarised as follows:

1. Compute histogram and probabilities of each intensity level.
2. Set up initial $w_i(0)$ and $\mu_i(0)$.
3. Step through all possible thresholds $t = 1, ...$ maximum intensity. Update w_i and μ_i, and compute the inter-class variance ($\sigma_b^2(t)$).
4. Desired threshold corresponds to the maximum $\sigma_b^2(t)$.
5. Repeat 2 to 4 for a fixed number of iteration or till the centroids don't change.

The intensity histogram for the given 3-bit image is symmetrically distributed. So, we can easily categorise it into 2 separate classes. As seen in Otsu's algorithm, we can compute the inter-class variance and check at what value of threshold is it maximum. By observation also, we can see that the optimal threshold will be at gray level 3, i.e. categorising gray levels 0 to 3 as one class and gray levels 4 to 7 as another class.

Question - 4

Feature Extraction.

(a) Briefly explain the working of Harris corner detection method. When Harris corner will fail to detect corner. Why?

(b) Consider the following image.

I

0	0	1	4	9
1	0	5	7	11
1	4	9	12	16
3	8	11	14	16
8	10	15	16	20

d/dx

-1	0	1

d/dy

-1
0
1

Figure 3: An image for Harris corner.

Compute the Harris matrix

$$H = \sum_{(x,y)\in W} \begin{bmatrix} I_x(x,y)^2 & I_x(x,y)I_y(x,y) \\ I_x(x,y)I_y(x,y) & I_y(x,y)^2 \end{bmatrix}$$

for the 3 by 3 highlighted window. In the above formula $I_x = dI/dx, I_y = dI/dy$ and W is the window highlighted in the image.

(i) First, compute the derivatives using the differentiation kernels shown above. No normalisation (division by 2) is needed.

(ii) Now compute the Harris Matrix based on the derivative matrices.

(iii) Compute the Harris cornerness score for $C = det(H) - k\ trace(H)^2$ for $k = 0.4$. What do we have here? A corner? An edge? Or a flat area? Why?

(a) The high-level pseudocode for Harris corner detection is as follows:

- Take the grayscale of the original image.
- Apply a Gaussian filter to smooth out any noise.
- Apply Sobel operator to find the x and y gradient values for every pixel in the grayscale image.
- For each pixel p in the grayscale image, consider a 3×3 window around it and compute the corner strength function. Call this its Harris value.
- Find all pixels that exceed a certain threshold and are the local maxima within a certain window (to prevent redundant dupes of features).
- For each pixel that meets the criteria in previous step, compute a feature descriptor.

The Harris corner detection algorithm is sensitive to scale change and corners detected throughout the entire image under complex background, thus extracting more false corners. This may lead to large amount of calculation and a high rate of error matching.

(b) (i)

$$I_x = \frac{dI}{dx} = \begin{bmatrix} 4 & 7 & 6 \\ 8 & 8 & 7 \\ 8 & 6 & 5 \end{bmatrix} \quad , \quad I_y = \frac{dI}{dy} = \begin{bmatrix} 4 & 8 & 8 \\ 8 & 6 & 7 \\ 6 & 6 & 4 \end{bmatrix}$$

(ii)

$$\sum_{(x,y)\in W} I_x(x,y)^2 = 4^2 + 7^2 + 6^2 + 8^2 + 8^2 + 7^2 + 8^2 + 6^2 + 5^2 = 403$$

$$\sum_{(x,y)\in W} I_y(x,y)^2 = 4^2 + 8^2 + 8^2 + 8^2 + 6^2 + 7^2 + 6^2 + 6^2 + 4^2 = 381$$

$$\sum_{(x,y)\in W} I_x(x,y) I_y(x,y) = 4*4+7*8+6*8+8*8+8*6+7*7+8*6+6*6+5*4 = 385$$

$$H = \begin{bmatrix} 403 & 385 \\ 385 & 381 \end{bmatrix}$$

(iii) $C = det(H) - k\ trace(H)^2 = 5318 - 0.04 * (784)^2 = -19268.24$
Since the Harris cornerness score is negative, it indicates an edge.

■ ■ ■