IPCV-I

Assignment Questions Guidelines for End Sem & TT2

Q2,Q6,Q9,Q11,Q15,Q18(TT2)

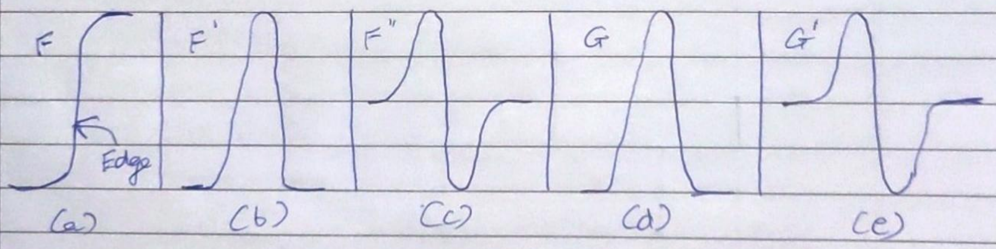
Q1. Short Note on Slant Transform

Q2. Explain Canny's Edge Detection Algorithm

Q.2. Explain Canny's Edge Detection Algorithm.

Canny operator is a first derivative edge detector coupled with noise cleaning. Canny, over the years, have become one of the most popular derivative operators. Like in LoG, a Gaussian function is used to smoothen the noise.

In Canny edge detector, we first smoothen image using Gaussian Low Pass Filter and then take the first derivative. Consider the diagram:



There is similarity between shape of Fig.(c) and Fig.(e). Hence, derivative of bell shape of Gaussian function approximates the second derivative.

Q3. Explain Wiener Filter with advantages & disadvantages

Q4. Explain Boundary Descriptors: Signatures, Fourier Descriptors, Polygon Approximation

Q5. Noise Models

Q6. Illustrate Harris Operator & Different forms of Harris Operators

Harris Corner Detector is a corner detection operator commonly used, in computer vision algorithms, to extract corners and infer features of an image. Harris' corner detector takes the differential of the corner score into account with reference to direction directly and has been proven to be more accurate in distinguishing between edges and corners.

Working of Harris Operator

The Harris operator is defined very simply, in terms of the local components of intensity gradient Ix, Iy in an image. We start by computing the following matrix:

$$\Delta = \begin{bmatrix} \langle I_x^2 \rangle & \langle I_x I_y \rangle \\ \langle I_x I_y \rangle & \langle I_y^2 \rangle \end{bmatrix}$$

where the suffixes indicate partial differentiation of the intensity I; we then use the determinant and trace to estimate the corner signal:

$$C = \frac{\det\Delta}{\operatorname{trace}\Delta}$$

While this definition involves averages, we shall find it more convenient to work with sums of quadratic products of intensity gradients:

$$\Delta = \begin{bmatrix} \Sigma I_x^2 & \Sigma I_x I_y \\ \Sigma I_x I_y & \Sigma I_y^2 \end{bmatrix}$$

Forms of Harris Operator:

There are different forms of the Harris operator that have been developed to improve its performance in certain situations:

Original Harris Operator: The original Harris operator is the formula shown above. It is simple and effective at detecting corners, but it is sensitive to noise and can generate false positives.

Improved Harris Operator: The improved Harris operator adds a constant term to the denominator of the formula to avoid division by zero, which can occur when the eigenvalues are close to zero. The formula is as follows:

$$R = \det(M) - k(\operatorname{trace}(M))^2$$

where

- $\det(M) = \lambda_1 \lambda_2$
- $\operatorname{trace}(M) = \lambda_1 + \lambda_2$
- λ_1 and λ_2 are the eigenvalues of M

Adaptive Harris Operator: The adaptive Harris operator adjusts the smoothing parameter σ based on the local image structure. It uses the eigenvalues of the image structure tensor to estimate the scale of the features in the image and sets the smoothing parameter accordingly.

Multi-scale Harris Operator: The multi-scale Harris operator uses the Harris operator at multiple scales to detect features of different sizes. It applies the Harris operator with different values of σ to the image and combines the results to obtain a multi-scale feature map.

Q7. Condition for Maximum corner signal for corners of different degrees (Corner Signals and shifts for various geometric configurations)

Q8. Evaluate the performance of Harris Operator on other types of features (Performance with crossing points and Junctions)

Q9.SN: Local Invariant Feature Detectors and Descriptors

Local invariant feature detectors and descriptors are widely used in computer vision and image processing for various applications such as image matching, object recognition, and 3D reconstruction. These methods aim to identify distinctive and repeatable features in images that are invariant to changes in scale, rotation, illumination, and viewpoint.

Local feature detectors typically identify interest points or key points in an image, such as corners, blobs, or edges. These key points are then described using a set of local image features, which capture the appearance and geometry of the region around the key point. Local feature descriptors can be designed to be invariant to various types of image transformations, such as rotation, scale, affine distortion, and viewpoint changes.

Some popular local invariant feature detectors and descriptors include Scale-Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF), Oriented FAST and Rotated BRIEF (ORB), and Local Binary Patterns (LBP). These methods have been shown to be effective in many real-world applications, such as object recognition, face recognition, and image retrieval.

Local invariant feature detectors and descriptors have also been used as building blocks for more complex computer vision systems, such as structure-from-motion and visual SLAM (Simultaneous Localization and Mapping). These systems rely on matching and tracking local features across multiple images to estimate the 3D structure and motion of the scene.

Q10. SN: Harris scale and Affine- Invariant Detectors and Descriptors

Q11 SN: The SIFT operators

The Scale-Invariant Feature Transform (SIFT) is a popular local feature detection and descriptor algorithm used in computer vision and image processing. It differs in using the Difference of Gaussians (DoG) instead of the Laplacian of Gaussians (LoG), in order to save computation. This possibility is seen by differentiating G with respect to σ.

$$G(\sigma) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{r^2}{2\sigma^2}\right)$$

$$\frac{\partial G}{\partial \sigma} = \left(\frac{r^2}{\sigma^3} - \frac{2}{\sigma}\right) G(\sigma) = \sigma \, LoG$$

which means that we can approximate LoG as the difference between Gaussians of two scales:

$$LoG \approx \frac{G(\sigma') - G(\sigma)}{\sigma(\sigma' - \sigma)} = \frac{G(k\sigma) - G(\sigma)}{(k-1)\sigma^2}$$

where the use of the constant scale factor k permits scale normalization to be carried out easily between scales.

In fact, it is in the design of the descriptors that SIFT is particularly different from the Harris and Hessian-based detectors. Here the operator divides the support region, at each scale, into a 16x16 sample array and estimates the intensity gradient orientations for each of these. They are then grouped into sets of sixteen 4x4 sub-arrays and orientation histograms are generated for each of these, the directions being restricted to one of eight directions. The final output is a 4x4 array of histograms each containing entries for eight directions—amounting to a total output dimensionality of 4 x 4 x 8 = 5128

Q12 SN: The SURF operators

Q13.Discuss Optical Flow with respect to Images (Discuss

Aperture Problem) Q14Interparte Optical Flow Field (F.ocus of

Expansion(FOE))

Q15 How to deduce the distance of the closest approach of the camera to the fixed object known as coordinates (Using the focus of expansion to avoid collision)

In the notation of Chapter 15, we have the following formulas for the location of an image point (x, y, z) resulting from a world point (X, Y, Z):

$$x = \frac{fX}{Z} \tag{19.8}$$

$$y = \frac{fY}{Z} \tag{19.9}$$

$$z = f \tag{19.10}$$

Assuming the camera has a motion vector $(-\dot{X}, -\dot{Y}, -\dot{Z}) = (-u, -v, -w)$, fixed world points will have velocity (u, v, w) relative to the camera. Now a point (X_0, Y_0, Z_0) will after a time t appear to move to $(X, Y, Z) = (X_0 + ut, Y_0 + vt, Z_0 + wt)$ with image coordinates:

$$(x, y) = \left(\frac{f(X_0 + ut)}{Z_0 + wt}, \frac{f(Y_0 + vt)}{Z_0 + wt}\right) \tag{19.11}$$

and as $t \to \infty$ this approaches the focus of expansion F $(fu/w, fv/w)$. This point is in the image, but the true interpretation is that the actual motion of the center of projection of the imaging system is toward the point:

$$\mathbf{p} = \left(\frac{fu}{w}, \frac{fv}{w}, f\right) \tag{19.12}$$

(This is of course consistent with the motion vector (u, v, w) assumed initially.) The distance moved during time t can now be modeled as:

$$\mathbf{X}_c = (X_c, Y_c, Z_c) = \alpha t\mathbf{p} = f\alpha t\left(\frac{u}{w}, \frac{v}{w}, 1\right) \tag{19.13}$$

where α is a normalization constant. To calculate the distance of closest approach of the camera to the world point $\mathbf{X} = (X, Y, Z)$, we merely specify that the vector

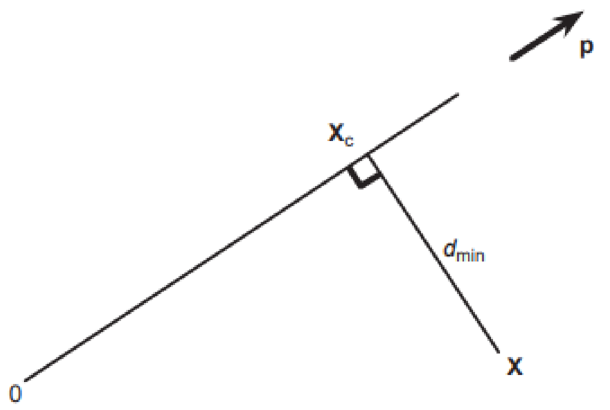


FIGURE 19.5

Calculation of distance of closest approach. Here, the camera is moving from 0 to $\mathbf{X}_c$ in the direction $\mathbf{p}$, not in a direct line to the object at $\mathbf{X}$. d_{min} is the distance of closest approach.

$\mathbf{X}_c - \mathbf{X}$ be perpendicular to $\mathbf{p}$ (Fig. 19.5) so that:

$$(\mathbf{X}_c - \mathbf{X}) \cdot \mathbf{p} = 0 \tag{19.14}$$

$$\text{i.e., } (\alpha t\mathbf{p} - \mathbf{X}) \cdot \mathbf{p} = 0 \tag{19.15}$$

$$\therefore \quad \alpha t\mathbf{p} \cdot \mathbf{p} = \mathbf{X} \cdot \mathbf{p} \tag{19.16}$$

$$\therefore \quad t = \frac{\mathbf{X} \cdot \mathbf{p}}{\alpha(\mathbf{p} \cdot \mathbf{p})} \tag{19.17}$$

Substituting in the equation for $\mathbf{X}_c$ now gives:

$$\mathbf{X}_c = \frac{\mathbf{p}(\mathbf{X} \cdot \mathbf{p})}{\mathbf{p} \cdot \mathbf{p}} \tag{19.18}$$

Hence, the minimum distance of approach is given by:

$$\begin{aligned} d_{\min}^2 &= \left[\frac{\mathbf{p}(\mathbf{X} \cdot \mathbf{p})}{(\mathbf{p} \cdot \mathbf{p})} - \mathbf{X}\right]^2 = \frac{(\mathbf{X} \cdot \mathbf{p})^2}{(\mathbf{p} \cdot \mathbf{p})} - \frac{2(\mathbf{X} \cdot \mathbf{p})^2}{(\mathbf{p} \cdot \mathbf{p})} + (\mathbf{X} \cdot \mathbf{X}) \\ &= (\mathbf{X} \cdot \mathbf{X}) - \frac{(\mathbf{X} \cdot \mathbf{p})^2}{(\mathbf{p} \cdot \mathbf{p})} \end{aligned} \tag{19.19}$$

which is naturally zero when $\mathbf{p}$ is aligned along $\mathbf{X}$. Clearly, avoidance of collisions requires an estimate of the size of the machine (e.g., robot or vehicle) attached to the camera and the size to be associated with the world point feature $\mathbf{X}$. Finally, note that while $\mathbf{p}$ is obtained from the image data, $\mathbf{X}$ can only be deduced from the image data if the depth Z can be estimated from other information. In fact, this information should be available from time-to-adjacency analysis (see below) if the speed of the camera through space (and specifically w) is known.

Q16.Discuss how the depth of the object can be deduced from optical flow (Time to adjacency analysis)

Q17.What could be difficulties with optical flow models

Q18 Explain Stereo from motion with respect to Images

Pending…