# CHAPETER 3

## FEATURE POINT DETECTORS

**Table of Contents**

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Introduction</td>
<td>31</td>
</tr>
<tr>
<td>3.2 Requirement of a corner detector</td>
<td>33</td>
</tr>
<tr>
<td>3.3 Kitchen-Rosenfeld corner detector</td>
<td>34</td>
</tr>
<tr>
<td>3.4 Harris corner detector</td>
<td>34</td>
</tr>
<tr>
<td>3.5 KLT corner detector</td>
<td>36</td>
</tr>
<tr>
<td>3.6 SUSAN corner detector</td>
<td>37</td>
</tr>
<tr>
<td>3.7 FAST corner detector</td>
<td>39</td>
</tr>
<tr>
<td>3.8 Proposed method</td>
<td>41</td>
</tr>
<tr>
<td>3.8.1 Steerable filters</td>
<td>41</td>
</tr>
<tr>
<td>3.8.2 Steerable Harris Detector</td>
<td>47</td>
</tr>
<tr>
<td>3.9 Summary</td>
<td>49</td>
</tr>
</tbody>
</table>
3. FEATURE POINT DETECTORS

3.1 INTRODUCTION

Traditionally, the term detector has been used as a tool that extracts the various image features from the images. A wide variety of feature detectors exist e.g., a corner detector, blob detector or edge detector. These detectors are used to detect the local features. The detected features are useful in computer vision and image processing, for example recognizing objects, matching, image alignment and mosaicing applications etc..

Corners are important local features in images. Corners are not much affected by illumination and have the property of rotational invariance. Without losing image data information, extracting corners can minimize the processing of data. The features detected by feature point detectors in an image also referred as interest/corner points. The corner features are identified by their high intensity gradient changes in both directions, vertical and horizontal. Let us consider an example; if a triangle object is present in the image then its three corners are very good feature points.

The corner detector [24] broadly categorized into two main groups:

i). Contour/Edge-based and

ii). Intensity based.
In first group algorithm, the first step is to identify the edges. After this process, the designer has to search for the intersection points or maxima curvature. The intensity based [25] methods estimate a measure to indicate the presence of a corner directly from the image gray values. This kind of method is characterized by its fast speed and its independence to other local features. In this chapter, we focused on the second group of feature detectors such as SUSAN [17], Kitchen-Rosenfeld [16], Harris[19], KLT [23], and FAST [6,26]. The Harris corner detector is, one of the most successful algorithms in the intensity-based approach. The following are some of important methods, which are used in the second group algorithm. Briefly they are described below.

- Smith and Brady method (SUSAN): They calculated cornerness value based on the geometrical criteria.
- Kitchen-Rosenfield method: To find out the cornerness value, they calculated the first and second order derivatives.
- Harris method: To find out the cornerness value, they proposed to calculate the first order derivatives only.
- KLT detector: To find out corner points, it will take the information from inter point displacement technique.
- FAST (Features from Accelerated Segment Test) detector: Introduced by Rosten and Drummond [26] builds on the SUSAN
computes the fraction of pixels within a neighborhood which have similar intensity to the centre pixel.

The use of feature points is a key step to find corresponding points between two or multiple images that correspond to the same object. These correspondences used for stereo matching, image registration, stitching and aligning the images into mosaic image, object recognition/detection, motion tracking and robot vision.

3.2 REQUIREMENTS OF A CORNER DETECTOR

Successful corner detectors have to satisfy a number of criteria as shown in table 3.1. First, all true corners should be detected.

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Properties</th>
<th>Good corner detectors</th>
<th>Bad corner detectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>All &quot;true corners&quot; should be detected.</td>
<td><img src="image" alt="Corner" /></td>
<td><img src="image" alt="Bad Corner" /></td>
</tr>
<tr>
<td>2</td>
<td>No &quot;false corners&quot; should be detected.</td>
<td><img src="image" alt="No Corner" /></td>
<td><img src="image" alt="Bad Corner" /></td>
</tr>
<tr>
<td>3</td>
<td>Corner points should be well localized</td>
<td><img src="image" alt="Localized Corner" /></td>
<td><img src="image" alt="Bad Corner" /></td>
</tr>
<tr>
<td>4</td>
<td>Detector should have a high repeatability rate (good stability).</td>
<td><img src="image" alt="High Repeatability" /></td>
<td><img src="image" alt="Bad Corner" /></td>
</tr>
<tr>
<td>5</td>
<td>Detector should be robust with respect to noise.</td>
<td><img src="image" alt="Robust Corner" /></td>
<td><img src="image" alt="Bad Corner" /></td>
</tr>
</tbody>
</table>
3.3 KITCHEN-ROSENFELD CORNER DETECTOR

In the corner detectors literature, it is reported that, Kitchen-Rosenfeld algorithm [16] is one of the earliest corner detectors method. This algorithm aims to calculate the ‘cornerness’ value $C$, which is based up on the local gradient magnitude and the rate of change gradient direction. The $C$ is given by

$$C = \frac{IN_{xx}IN_x^2 + IN_{yy}IN_y^2 - 2IN_{xy}IN_xIN_y}{IN_x^2 + IN_y^2} \quad (3.1)$$

Where $IN$ is the grey-level and $IN_{xx}$, $IN_{yy}$ are the directional second derivative of $IN$. similarly $IN_x$, $IN_y$ are the directional first derivative of $IN$. $IN_{xy}$ is the partial derivate of $IN$ with respect to $x$ and $y$ consequently. Points in an image are declared as corners, if the ‘cornerness value’ meets some threshold requirement. This detector uses second order derivatives and hence more sensitivity to noise.

3.4 HARRIS CORNER DETECTOR (HCD)

This detector will detect corner only when the local autocorrelation function is maximum. Calculate the variation of pixel gradient. Declare the pixel as a corner if gradient of pixel varies in all directions significantly.

$$R = \det(M) - k(\text{trace}(M))^2 \quad (3.2)$$

$$M = G(\sigma) \left[ \begin{array}{ccc} \frac{\partial I}{\partial x}^2 & \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \\ \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} & \frac{\partial I}{\partial y}^2 \end{array} \right] \quad (3.3)$$
Where \( R \) = measurement of corner response at pixel coordinates \((x, y)\) and \( k = \) constant (typical value is 0.04).

\[
\frac{\partial I}{\partial x} = \text{the first order gray gradient of horizontal}
\]

\[
\frac{\partial I}{\partial y} = \text{the first order gray gradient of vertical}
\]

\( G(\sigma) = \) isotropic Gaussian filter with standard deviation \( \sigma \) and the operator \( * \) represents convolution

![Fig. 3.1: Harris operator.](image)

In figure 3.1, the values of \( \lambda_1 \text{or} \lambda_2 \) (\( \lambda_1 \) and \( \lambda_2 \) are Eigen values of \( M \)) plays a key role. By observing these two values, we can determine the cases of edge, corner or area. Based on the principal curvature of partial correlation, there is a possibility to get three different cases. They are

CASE 1: If curvatures \((\lambda_1 \text{ and} \lambda_2)\) are low, then the resultant correlation function shows flat.
CASE 2: If any one curvature ($\lambda_1$ or $\lambda_2$) is low and other is high, then autocorrelation changes in one direction only. It represents the detection of edge.

CASE 3: If both curvatures ($\lambda_1$ and $\lambda_2$) are high, the auto correlation function changes indicate peak value. It represents the detection of corner. This is a fast, simple detector and the computations are based entirely on first derivatives. It suffers from loss in localization accuracy, sensitive to quantization noise.

### 3.5 KLT CORNER DETECTOR

The KLT (Kanade-Lucas-Tomasi) method has two parameters: the threshold on $\lambda_2$, denoted by $\lambda_{thr}$, and the linear size of a square window (neighborhood) $D \times D$.

The KLT corner detector algorithm:

1. Compare C at each point $(x, y)$ of the image.
2. For each image point $P(x, y)$.
   
   (a) Find the smallest value of $\lambda_2$ in $D$-neighborhood of $P$.
   
   (b) If that $\lambda_2 > \lambda_{thr}$, save $P(x, y)$ into a potential list $L$.
3. Sort $L$ in decreasing order of $\lambda_2$.
4. Scan the sorted list from top to bottom. For each current point $P$, delete all lower points in the list in the $D$-neighborhood of $P$. 
The KLT algorithm produces a list of feature points that have \( \lambda_2 > \lambda_{th} \) and the neighborhood \( D \) of these points do not overlap. The parameter selection \( \lambda_{th} \) may be estimated from a histogram of \( \lambda_2 \): usually select \( \lambda_{th} \) to a valley of the histogram.

Select window size \( D \) usually between 2 to 10, by trial and error. For large \( D \), corners may move away from the actual position and some close neighbor corners may be lost. This detector uses explicit calculation of eigenvalue \( \lambda_2 \) and produces good repeatability. But it suffers from localization error and sensitive to geometrical distortions.

### 3.6 SUSAN CORNER DETECTOR

The SUSAN (Smallest Unvalue Segment Assimilating Nucleus) detector was proposed by Smith and Brady[17]. This detector neither uses spatial derivatives nor smoothes the image. It is based on the fact that each point (pixel) within an image has to be connected with a local area of comparable brightness. Apply circular mask for each and every pixel in an image. For each pixel, find out the differences between the brightness of each pixel and the mask. The area of mask should be selected or defined in such a way that the brightness of the nucleus is same. This area of mask with same brightness can be treated as “USAN”.

The figure 3.2 represents the best example for USAN. A dark rectangle on a white background with five circular masks \( a, b, c, d \) and \( e \) are shown at different positions on the simple image. From the
figure, we can conclude that ‘a’ has corner. Because, in ‘a’, the area of USAN is smallest when compared to the b, c, d and e.

![Diagram showing corner detection principles](image)

Fig. 3.2: SUSAN principle (a) circle masks at different places on the image, (b) USAN show in white color.

Detection of corner decision will depend on equation number (3.4). The relation between the selected pixel and nucleus pixel is given as

$$c(\vec{r}, \vec{r}_0) = \begin{cases} 1, & \|I(\vec{r})-I(\vec{r}_0)\| \leq t \\ 0, & \|I(\vec{r})-I(\vec{r}_0)\| > t \end{cases}$$ (3.4)

Where $\vec{r}_0$ = Nucleus position (if the image considered to be in two dimensional)

$\vec{r}$ = Position of selected point within the mask

$I(\vec{r})$ = Brightness of any pixel

$t$ = Threshold values gives information about brightness difference

$c$ = Comparison of output between selected and nucleus.

The Eq.(3.4) as described gives quite good results, but much more stable and sensible equation is
\[ c(\vec{r}, \vec{r}_0) = e^{-\left( \frac{I(\vec{r}) - I(\vec{r}_0)}{t} \right)^6} \]  

The sixth power is used in Eq. (3.5) to obtain the optimum brightness difference. The size of USAN region is given by (3.6):

\[ n(r_0) = \sum_{r \in c(r_0)} c(\vec{r}, \vec{r}_0) \]  

The initial response of corners depend on a SUSAN region. If USAN area is small then there is a chance to get the great response to corners and vice-versa.

\[ R(\vec{r}_0) = \begin{cases} g - n(\vec{r}_0) & \text{if } n(\vec{r}_0) > g \\ 0 & \text{otherwise,} \end{cases} \]  

In (3.7), g is geometric threshold. The g value plays an active role in determining the accuracy. If g is small then the percentage of accuracy is high. The value of g also enhances the information of the corner in an image. Based on non-maximum inhibition, finally we can find corners. SUSAN corner detector makes a better noise robustness and localization compared to previous algorithms, but has an average repeatability rate and more computational cost.

### 3.7 FAST CORNER DETECTOR

The FAST (Features from Accelerated Segment Test) detector, introduced by Rosten and Drummond [26] builds on the SUSAN computes the fraction of pixels within a neighborhood which have similar intensity to the centre pixel. This idea is taken further by Fast
corner detector based on segment test, which computes pixels only on a circle of fixed radius around the point.

Considering a circle of sixteen pixels around the corner point $P$ is shown in figure 3.3.

![Segment point detection in an image patch, the pixel at $P$ is the centre of candidate corner.](image)

There are two possibilities to detect the selected point $P$ as a corner.

Case1: The brightness of $P$ should be less than the surrounding pixels

Case2: The brightness of $P$ should be greater than the surrounding pixels.

To achieve high-speed test, we choose $n$ value as 12. This selection of $n$ value is used to exclude a very large number of non-corners. In the example, we selected four points i.e. 1, 5, 9 and 13.

- If any three pixels brightness value in selected 4 pixels is more than the brightness of $P$, then the $P$ is declared as corner.

- If any three pixels darker value in selected 4 pixels is more than the value of $P$, then the $P$ is declared as corner.
In neither of these is case existed, and then \( P \) cannot be treated as a corner.

Segment test can be applied to the remaining candidates by examining all pixels in the circle. This detector exhibits computationally efficient, but it has several weaknesses:

1. If pixels are adjacent to each other, then multiple features can be detected.
2. The high-speed test does not generalize well for \( n < 12 \).
3. The choice and ordering of the fast test pixels contains implicit assumptions about the distribution of feature appearance.
4. Knowledge from the first 4 tests is discarded when the full segment test criterion is applied.

This detector is computationally efficient, but it suffers from loss in localization accuracy and sensitive to noise.

### 3.8 PROPOSED METHOD

Proposed method uses the steerable filters and Harris corner detectors to detect the feature points.

#### 3.8.1 Steerable filters

Steerable filters are introduced by Freeman [27,28,29] and Adelson[27]. Steerable filters are a class of filters in which filter of arbitrary orientation is synthesized as a linear combination of set of “basis filters”. These filters allowing, one to adaptively “steer” a filter to
any orientation, and to determine analytically the filter output as a function of orientation. Steerable filters may be designed in quadrature pairs to allow adaptive control over phase as well as orientation, to show as how many basis filters are needed to steer a given filter in two-dimensional and three-dimensional cases. First we consider the two-dimensional, circularly symmetric Gaussian function $G$ written in Cartesian coordinate $x$ and $y$:

$$G(x, y) = e^{-(x^2 + y^2)}$$  \hspace{1cm} (3.8)

Where normalization and scaling constants are set to one for convince.

Let $n^{th}$ derivative of a Gaussian in the $x$ direction be $G_n$, represent the rotation operator $(...)^\theta$ for any function $f(x, y)$, the $f(x, y)$ is rotated through an angle $\theta$ about the origin is given by $f^\theta(x, y)$.

The first derivative of a Gaussian $G_1^{\theta\theta}$ is

$$G_1^{\theta\theta} = \frac{\partial e^{-(x^2 + y^2)}}{\partial x} = -2xe^{-(x^2 + y^2)}$$  \hspace{1cm} (3.9)

Now rotate the function by $90^0$, then

$$G_1^{90^\theta} = \frac{\partial e^{-(x^2 + y^2)}}{\partial y} = -2ye^{-(x^2 + y^2)}$$  \hspace{1cm} (3.10)

These Gaussian filter kernels are shown in figure 3.5(b)

$$G_1^\theta = \cos(\theta)G_1^{\theta\theta} + \sin(\theta)G_1^{90^\theta}$$  \hspace{1cm} (3.11)
The $G_i^{\theta_0}$ and $G_i^{90^\circ}$ are basis filters for $G_i^\theta$

and $G_i^\theta$ is Gaussian filter at an arbitrary orientation $\theta$.

The $\cos(\theta)$ and $\sin(\theta)$ terms are the corresponding interpolation functions for those basis filters.

Filtered image can synthesize at an arbitrary orientation by taking linear combination of the filtered image with $G_i^{\theta_0}$ and $G_i^{90^\circ}$. Letting $\ast$ represent convolution, if

$$R_1^{\theta_0} = G_i^{\theta_0} \ast I$$
$$R_1^{90^\circ} = G_i^{90^\circ} \ast I$$

Then the output of a filter tuned to any orientation, let us desire $R_1^{\theta}$, by interpolate exactly from the responses of the basis filters.

$$R_1^{\theta} = \cos(\theta)R_1^{\theta_0} + \sin(\theta)R_1^{90^\circ}$$

The basis filters of the steerable filters are directional derivative operators, thus allowing independent representation of orientations [30], as shown in figure 3.5 (b).

Fig. 3.4: Steerable pyramid transform.
Figure 3.4 shows the steerable pyramid transform decomposes an image into two portions, such as high pass $H_0(-\omega)$ and low pass $L_0(-\omega)$. The low pass portion is further subsampled, and the recursion is performed repeatedly by applying four band pass filters which are oriented at $0^0$, $45^0$, $90^0$, and $135^0$. The recursive portion of the subsystem is shown as dashed box.

Where $H_0$, $L_0$, $B_k$, $L_1$ are the high-pass, the first low pass, directional band pass filters and the second low-pass filters respectively.

Fig. 3.5: (a) original image (b) Phase analyzing filters, $0^0$, $45^0$, $90^0$ and $135^0$ orientations. (c) Filter response ($k=4$) at four orientations (d) the shaded region corresponds to the spectral support of a single diagonal oriented sub band.
The steerable pyramid performs a polar-separable decomposition in the frequency domain, as shown in figure 3.5(d), thus allowing independent representation of scale and orientation.

![Steerable filter block diagram](image)

**Fig. 3.6: Steerable filter block diagram.**

The primary objective is to develop a rotated filter from a linear combination of a fixed set of basis filters. Figure 3.6 represents the general architecture of steerable filter. This architecture will use dedicated bank of filters. These filters convolve the image as it comes in. By using gain masks, we can multiply the filter outputs. For every position, apply interpolation in gain mask that should vary with time constraints. Finally the summations will give the filtered image. There are no restrictions over derivative filters and steerability condition. It may be expressed as a function of signal $f$ and is given by

$$ f^\theta(x, y) = \sum_{m=1}^{M} k_m(\theta)f^\vartheta_m(x, y) $$

(3.15)

where $f^\vartheta(x, y) =$ the function at $(x, y)$ position with respect to angle $\theta$
\( k_m(\theta) \) = the interpolation functions, and

\( M \) = the number of basis functions required to steer the function \( f(x,y) \).

(The steering function for even and odd polynomials are given in Appendix B)

To determine the conditions under which a given function satisfies the steering conditions in Eq.(3.15), let us work in polar coordinates

\[ (r = \sqrt{x^2 + y^2}) \text{ and } \phi = \text{arg}(x,y). \]

The function \( f \) could be expressed as Fourier series in polar angle \( \theta \):

\[
f(r, \phi) = \sum_{n=-N}^{N} a_n(r)e^{j n \phi}\tag{3.16}
\]

Where \( j = \sqrt{-1} \) and \( N \) is the discrete length of coefficients. According to Freeman et.al.[27], that the steering condition in Eq.(3.15) is satisfied for functions expandable in the form of Eq.(3.16), if only the interpolation function \( k_m(\theta) \) are solution of:

\[
c_n(\theta) = \sum_{m=-M}^{M} k_m(\theta)(c_n(\theta))^m\tag{3.17}
\]

where \( e = e^{j \theta} \), and \( n=0,\ldots,N \).

from of Eq.(3.16), \( f^\theta(r,\phi) \) is expressed as:

\[
f^\theta(r,\phi) = \sum_{m=1}^{M} k_m(\theta)g_m(r,\phi)\tag{3.18}
\]

where \( g_m(r,\phi) \) can be any set of functions.
It has been also demonstrated that the minimum number $M$ of basis functions required to steer $f(r, \phi)$ is equal to the number of non-zero Fourier coefficients $a_n(r)$.

### 3.8.2 Steerable Harris Detector (SHD)

The KLT, Harris, FAST and SUSAN corner detector methods are suffering with certain limitations. These methods fail to detect all true corners and/or detect too many false corners due to the presence of noise in the images. To overcome this problem, we propose a corner detector algorithm derived from steerable filters and Harris detector. The proposed algorithm steps are given below:

1. First extract the image features at different orientations with steerable filters.

2. Using with the Harris detectors find the corners in each orientation.

3. Combining all detected corners by performing logical OR operation.

4. Applying dilation to the combining nearby corners to make them into one.

5. Find the centroid of step 4 and identify as a corner.
Fig. 3.7: Proposed SHD corner detection method.

The figure 3.7 shows proposed corner detector algorithm that uses the steerable filters. The captured image is initially applied to the steerable filters. These steerable filters decompose and enhance the image features at different orientations. The most popular HCD is applied to four oriented filters output and corners are detected for each orientation. Combine all the corners for individual orientation by Logical OR operations. It is observed that too many corners are detected in some places of image. To make them as a single corner, use dilation operation and declare as a corner. The experiments carried out with matlab and concluded that if the number of orientations is four it exhibits better localization and detecting minimum number of false corners.
3.9 SUMMARY

Corner detectors are basically classified as contour and intensity based. In this chapter, intensity based algorithms such as Kitchen-Rosenfeld, FAST, KLT, SUSAN and Harris are discussed in detail. Intensity based algorithm is characterized by its fast speed and its independence to other local features. Most of the detectors have problems with regard to noise, localization error and accuracy and geometrical distortions. To overcome these limitations, we attempted to enhance the performance of most successful HCD detector with the combination of steerable filters. The results of the proposed method compared with other four detectors explained and presented in chapter 6. The next chapter explains RANSAC algorithm and is used to separate outliers from inliers.