CAP4453-Robot Vision

Lecture 7- Finding Features
(introduction to feature engineering, local features, corners)

Dr. Ulas Bagci
bagci@ucf.edu
Outline

Non-Deep Learning Approaches (Introduction to Feature Engineering)

• Ex1. Key-point Features
  – Harris corner detection
• Ex2. Affine Invariance

• Read Szeliski, Chapter 4.
• Read Shah, Chapter 2.
• Read/Program CV with Python, Chapters 1 and 2.
Motivation for Feature Finding: **Matching**
Motivation for Feature Finding: Harder case
How do we process scenes for recognition/matches?

Do we see larger objects first or smaller?
Do we pay attention to large homogeneous regions or sharp corners?
Do we see first general or specific?
How much should we pay attention to specifics?
....
What are the features that we look for matching images? (e.g.)
Finding Features in Videos

Subsequent studies have shown that many complex actions can be recognized on the basis of such 'point-light displays', including:

- facial expressions,
- Sign Language,
- arm movements,
- and various full-body actions.
Features are used in ...

- Automate object tracking
- Point matching for computing disparity
- Motion based segmentation
- Object Recognition
- 3D Object Reconstruction
- Robot Navigation
- Image Retrieval/Indexing
- ....
What is an interest point?

- **Expressive texture**
  - The point at which the direction of the boundary of object changes abruptly
  - Intersection point between two or more edge segments
What is an interest point?
Properties of Interest Points

• Detect all (or most) true interest points
• No false interest points
• Well localized
• Robust with respect to noise
• Efficient detection
Possible Approaches for Corner Detection (ex. interest point)

• Based on brightness of images
  • Usually image derivatives
• Based on boundary extraction
  • First step edge detection
  • Curvature analysis of edges
Correspondence Across Views

• Correspondence: matching points, patches, edges, or regions across images
Goals for KeyPoints

Detect points that are *repeatable* and *distinctive*
Overview of KeyPoint Matching

1. Find a set of distinctive keypoints
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region
5. Match local descriptors

\[ d(f_A, f_B) < T \]
Major Trade-offs

Detection of interest points

More Repeatable
- Robust detection
- Precise localization

More Interest Points
- Robust to occlusion
- Works with less texture

Description of patches

More Distinctive
- Minimize wrong matches

More Flexible
Local Features: Main Components

1) **Detection**: Identify the interest points

2) **Description**: Extract feature vector descriptor surrounding each interest point.

3) **Matching**: Determine correspondence between descriptors in two views

\[
x_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]
\]

\[
x_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]
\]
Goal: interest operator repeatability

• We want to detect (at least some of) the same points in both images.

No chance to find true matches!

• Yet we have to be able to run the detection procedure *independently* per image.
Goal: descriptor distinctiveness

• We want to be able to reliably determine which point goes with which.

• Must provide some invariance to geometric and photometric differences between the two views.

Kristen Grauman
Some patches can be localized or matched with higher accuracy than others.

Why?
A COMBINED CORNER AND EDGE DETECTOR

Chris Harris & Mike Stephenson
Plessey Researchoke Manor, United Kingdom
© The Plessey Company plc. 1988

Consistency of image edge detection is of prime importance for 3D interpretation of image sequences using feature tracking algorithms. To cater for image regions containing texture and isolated features, a combined corner and edge detector based on the local auto-correlation function is utilized, and it is shown to perform well in consistency on natural imagery.

INTRODUCTION

The problem we are addressing in Alvey Project MM149 is that of using computer vision to understand the uncontrolled 3D world, in which the viewed scenes will in general contain too wide a diversity of objects for top-down recognition techniques to work. For example, we desire to obtain an understanding of natural scenes, containing roads, buildings, trees, bushes, etc., as typified by the two frames from a sequence illustrated in Figure 1.

The solution to this problem that we are pursuing is to use a computer vision system based upon motion analysis of a monocular image sequence from a mobile camera. By extraction and tracking of image features, representations of the 3D analogues of these features can be constructed.

To enable explicit tracking of image features to be performed, the image features must be discrete, and not form a continuum (like texture, or edge pixels (edges)). For this reason, our earlier work has concentrated on the extraction and tracking of feature-points or corners, since they are discrete, reliable and meaningful. However, the lack of connectivity of feature-points is a major limitation in our obtaining higher level descriptions, such as surfaces and objects. We need the richer information that is available from edges.

THE EDGE TRACKING PROBLEM

Matching between edge images on a pixel-by-pixel basis works for stereo, because of the known epipolar camera geometry. However for the motion problem, where the camera motion is unknown, the aperture problem prevents us from undertaking explicit edgel matching. This could be overcome by solving for the motion beforehand, but we are still faced with the task of tracking each individual edge pixel and estimating its 3D location from, for example, Kalman Filtering. This approach is unattractive in comparison with assembling the edges into edge segments, and tracking these segments as the features.

Now, the uncontrolled imagery we shall be considering will contain both curved edges and texture of various scales. Representing edges as a set of straight line fragments, and using these as our discrete features will be inappropriate, since curved lines and texture edges can be expected to fragment differently on each image of the sequence, and so be untrackable. Because of ill-conditioning, the use of parametrised curves (eg. circular arcs) cannot be expected to provide the solution, especially with real imagery.

- Edge detectors often fail in corners. Why?
- Corner point can be recognized in a window
- Shifting a window in any direction should give a large change in intensity
- LOCALIZING and UNDERSTANDING shapes...

Figure 1. Pair of images from an outdoor sequence.
Basic Idea in Corner Detection

“flat” region:
no change in all directions

“edge”:
no change along the edge direction

“corner”:
significant change in all directions
Template Matching

\[
\begin{bmatrix}
-4 & 5 & 5 \\
-4 & 5 & 5 \\
-4 & -4 & -4
\end{bmatrix}
\quad
\begin{bmatrix}
5 & 5 & 5 \\
-4 & 5 & -4 \\
-4 & -4 & -4
\end{bmatrix}
\]
Template Matching

Complete set of eight templates can be generated by successive 90 degree of rotations.
Template Matching

Complete set of eight templates can be generated by successive 90 degree of rotations.

Why the summation of filter is 0?
Template Matching

Complete set of eight templates can be generated by successive 90 degree of rotations.

Why the summation of filter is 0?

Insensitive to absolute change in intensity!
Corner Detection using the Hessian Matrix

- Corners are characterized by high-curvature of intensity values.

\[ H(p) = \begin{bmatrix} I_{xx}(p) & I_{xy}(p) \\ I_{xy}(p) & I_{yy}(p) \end{bmatrix} \]
Corner Detection using the **Hessian Matrix**

- Corners are characterized by high-curvature of intensity values.

\[
H(p) = \begin{bmatrix}
I_{xx}(p) & I_{xy}(p) \\
I_{xy}(p) & I_{yy}(p)
\end{bmatrix}
\]
Corner Detection using the **Hessian Matrix**

- Corners are characterized by high-curvature of intensity values.

\[
H(p) = \begin{bmatrix}
I_{xx}(p) & I_{xy}(p) \\
I_{xy}(p) & I_{yy}(p)
\end{bmatrix}
\]

Hessian Matrix at pixel p.
Corner Detection using the **Hessian Matrix**

- Corners are characterized by high-curvature of intensity values.

\[
H(p) = \begin{bmatrix}
I_{xx}(p) & I_{xy}(p) \\
I_{xy}(p) & I_{yy}(p)
\end{bmatrix}
\]

- Hessian Matrix at pixel \( p \).
- Eigenvalues of the Hessian indicate corner features if both eigenvalues are large!
Corner Detection using the **Hessian Matrix**

- Corners are characterized by high-curvature of intensity values.

\[
H(p) = \begin{bmatrix}
I_{xx}(p) & I_{xy}(p) \\
I_{xy}(p) & I_{yy}(p)
\end{bmatrix}
\]

Hessian Matrix at pixel \( p \).

Eigenvalues of the Hessian indicate corner features if both eigenvalues are large!

One large, one small eigenvalues -> edge regions

Two small eigenvalues indicate -> flat regions
Corner Detection using the Hessian Matrix

- Corners are characterized by high-curvature of intensity values.

\[ H(p) = \begin{bmatrix} I_{xx}(p) & I_{xy}(p) \\ I_{xy}(p) & I_{yy}(p) \end{bmatrix} \]

Hessian matrix summarizes Distribution of gradients.

One large, one small eigenvalues -> edge regions

Two small eigenvalues indicate -> flat regions
Reminder: Eigenvalues/Eigenvectors

\[ H(p) = \begin{bmatrix} I_{xx}(p) & I_{xy}(p) \\ I_{xy}(p) & I_{yy}(p) \end{bmatrix} \]

\[ \text{det}(H) = I_{xx}I_{yy} - I_{xy}^2 \]

\[ \text{Tr}(H) = I_{xx} + I_{yy} \]
Reminder: Eigenvalues/Eigenvectors

\[ H(p) = \begin{bmatrix} I_{xx}(p) & I_{xy}(p) \\ I_{xy}(p) & I_{yy}(p) \end{bmatrix} \]

\[ \det(H) = I_{xx}I_{yy} - I_{xy}^2 \]

\[ \text{Tr}(H) = I_{xx} + I_{yy} \]

The eigenvalues of the matrix \( H \) are solutions of its characteristics polynomial

\[ \det(H - \lambda I_2) = 0 \]
The eigenvalues of the matrix $H$ are solutions of its characteristic polynomial

$$\det(H - \lambda I_{2}) = 0$$

$$\det\left( \begin{bmatrix} I_{xx}(p) - \lambda & I_{xy}(p) \\ I_{xy}(p) & I_{yy}(p) - \lambda \end{bmatrix} \right) = 0$$

where $I_{xx}(p)$, $I_{xy}(p)$, and $I_{yy}(p)$ are the elements of the matrix $H(p)$.
Reminder: Eigenvalues/Eigenvectors

\[ H(p) = \begin{bmatrix} I_{xx}(p) & I_{xy}(p) \\ I_{xy}(p) & I_{yy}(p) \end{bmatrix} \]

\[ \det(H) = I_{xx}I_{yy} - I_{xy}^2 \]

\[ \text{Tr}(H) = I_{xx} + I_{yy} \]

The eigenvalues of the matrix H are solutions of its characteristics polynomial

\[ \det(H - \lambda I_2) = 0 \]

\[ \det( \begin{bmatrix} I_{xx}(p) - \lambda & I_{xy}(p) \\ I_{xy}(p) & I_{yy}(p) - \lambda \end{bmatrix} ) = 0 \]

\[ (I_{xx}(p) - \lambda)(I_{yy}(p) - \lambda) - I_{xy}(p)^2 = 0 \]
Harris and Stephens Corner Detector

• Instead of using Hessian of image I, use first derivative of smoothed I (i.e., L)

\[
G(p, \sigma) = \begin{bmatrix}
L_x^2(p, \sigma) & L_x(p, \sigma)L_y(p, \sigma) \\
L_x(p, \sigma)L_y(p, \sigma) & L_y^2(p, \sigma)
\end{bmatrix}
\]
Harris and Stephens Corner Detector

• Instead of using *Hessian* of image I, use first derivative of smoothed I (i.e., L)

\[ G(p, \sigma) = \begin{bmatrix} L_x^2(p, \sigma) & L_x(p, \sigma)L_y(p, \sigma) \\ L_x(p, \sigma)L_y(p, \sigma) & L_y^2(p, \sigma) \end{bmatrix} \]

• Instead of calculating eigenvalues, we will consider *cornerness* feature:
Harris and Stephens Corner Detector

• Instead of using *Hessian* of image I, use first derivative of smoothed I (i.e., L)

\[
G(p, \sigma) = \begin{bmatrix}
L_x^2(p, \sigma) & L_x(p, \sigma)L_y(p, \sigma) \\
L_x(p, \sigma)L_y(p, \sigma) & L_y^2(p, \sigma)
\end{bmatrix}
\]

• Instead of calculating eigenvalues, we will consider *cornerness* feature:

\[
\mathcal{H}(p, \sigma, \alpha) = \det(G) - \alpha \cdot \text{Tr}(G)
\]

*for small alpha (=1/25)*
Harris and Stephens Corner Detector

\[ \mathcal{H}(p, \sigma, \alpha) = \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2) \]
Harris and Stephens Corner Detector

\[ \mathcal{H}(p, \sigma, \alpha) = \lambda_1 \lambda_2 - \alpha.(\lambda_1 + \lambda_2) \]

• The same behavior as Hessian based detector, but now we directly use simple determinant and trace functions!

\[ \mathcal{H}(p, \sigma, \alpha) = \text{det}(G) - \alpha.\text{Tr}(G) \]
Harris and Stephens Corner Detector

\[ \mathcal{H}(p, \sigma, \alpha) = \lambda_1 \lambda_2 - \alpha (\lambda_1 + \lambda_2) \]

- The same behavior as Hessian based detector, but now we directly use simple determinant and trace functions!

\[ \mathcal{H}(p, \sigma, \alpha) = \det(G) - \alpha \cdot \text{Tr}(G) \]

Other \textit{cornerness} measures:

\[ \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2} \quad \text{or} \quad \lambda_1 - \alpha \lambda_2 \]

Harmonic mean \quad Triggs
Square of derivatives

Gaussian Smoothing
Feature Extraction: Corners

9300 Harris Corners Pkwy, Charlotte, NC
Invariant Local Features

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters.
Invariance and Covariance

- We want corner locations to be **invariant** to photometric transformations and **covariant** to geometric transformations
  - **Invariance**: image is transformed and corner locations do not change
  - **Covariance**: if we have two transformed versions of the same image, features should be detected in corresponding locations
Affine Transformation

translation
Affine Transformation

- Translation
- Rotation
Affine Transformation

- Translation
- Rotation
- Scaling
Affine Transformation

Corners are invariant to

- translation ✔
- rotation ✔
- scaling ✗
Questions?