Recap: Sliding-Window Object Detection

- If object may be in a cluttered scene, slide a window around looking for it.

- Essentially, this is a brute-force approach with many local decisions.

Recap: AdaBoost

- Want to select the single rectangle feature and threshold that best separates positive (faces) and negative (non-faces) training examples, in terms of weighted error.

Resulting weak classifier:

\[ h(x) = \begin{cases} 
  +1 & \text{if } f_1(x) > \theta_1 \\
  -1 & \text{otherwise} 
\end{cases} \]

For next round, reweight the examples according to errors, choose another filter/threshold combo.
Recap: Viola-Jones Face Detector

- Train with 5K positives, 350M negatives
- Real-time detector using 38 layer cascade
- 6061 features in final layer
- Implementation available in OpenCV: http://sourceforge.net/projects/opencvlibrary/

Limitations of Sliding Windows (continued)

- Not all objects are “box” shaped

Limitations (continued)

- Non-rigid, deformable objects not captured well with representations assuming a fixed 2D structure; or must assume fixed viewpoint
- Objects with less-regular textures not captured well with holistic appearance-based descriptions

Limitations (continued)

- In practice, often entails large, cropped training set (expensive)
- Requiring good match to a global appearance description can lead to sensitivity to partial occlusions

Topics of This Lecture

- Local Invariant Features
  - Motivation
  - Requirements, Invariances
- Keypoint Localization
  - Harris detector
  - Hessian detector
- Scale Invariant Region Selection
  - Automatic scale selection
  - Laplacian-of-Gaussian detector
  - Difference-of-Gaussian detector
  - Combinations
- Local Descriptors
  - Orientation normalization
  - SIFT
Motivation

- Global representations have major limitations
- Instead, describe and match only local regions
- Increased robustness to
  - Occlusions
  - Articulation
  - Intra-category variations

Application: Image Matching

Harder Case

Answer Below (Look for tiny colored squares)

Application: Image Stitching

Harder Still?
Application: Image Stitching

• Procedure:
  1. Detect feature points in both images
  2. Find corresponding pairs
  3. Use these pairs to align the images

Common Requirements

• Problem 1:
  1. Detect the same point independently in both images

We need a repeatable detector!
Invariance: Geometric Transformations

Requirements

- Region extraction needs to be repeatable and accurate
  - Invariant to translation, rotation, scale changes
  - Robust or covariant to out-of-plane (affine) transformations
  - Robust to lighting variations, noise, blur, quantization
- Locality: Features are local, therefore robust to occlusion and clutter.
- Quantity: We need a sufficient number of regions to cover the object.
- Distinctiveness: The regions should contain "interesting" structure.
- Efficiency: Close to real-time performance.

Many Existing Detectors Available

- Hessian & Harris [Beaudet '78], [Harris '88]
- Laplacian, DoG [Lindeberg '98], [Lowe '99]
- Harris-/Hessian-Laplace [Mikolajczyk & Schmid '01]
- Harris-/Hessian-Affine [Mikolajczyk & Schmid '04]
- EBR and IBR [Tuyltelaars & Van Gool '04]
- MSER [Matas '02]
- Salient Regions [Kadir & Brady '01]
- Others...

Those detectors have become a basic building block for many recent applications in Computer Vision.

Keypoint Localization

- Goals:
  - Repeatable detection
  - Precise localization
  - Interesting content

⇒ Look for two-dimensional signal changes

Finding Corners

- Key property:
  - In the region around a corner, image gradient has two or more dominant directions
- Corners are repeatable and distinctive

Corners as Distinctive Interest Points

- Design criteria
  - We should easily recognize the point by looking through a small window (locality)
  - Shifting the window in any direction should give a large change in intensity (good localization)

- “flat” region: no change in all directions
- “edge”: no change along the edge direction
- “corner”: significant change in all directions

Harris Detector Formulation

- This measure of change can be approximated by:
  \[ E(u,v) \approx [u \ v] \begin{bmatrix} M_{xx} & M_{xy} \\ M_{xy} & M_{yy} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \]

  where \( M \) is a 2x2 matrix computed from image derivatives:
  \[ M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \]

  Sum over image region - the area we are checking for corner

  \[ M = \sum_{x,y} w(x,y) I_x I_y \begin{bmatrix} I_x & I_y \\ I_y & I_y \end{bmatrix} \]

  Gradient with respect to \( x \), times gradient with respect to \( y \)

What Does This Matrix Reveal?

- First, let’s consider an axis-aligned corner:

  - This means:
    - Dominant gradient directions align with \( x \) or \( y \) axis
    - If either \( \lambda \) is close to 0, then this is not a corner, so look for locations where both are large.

  - What if we have a corner that is not aligned with the image axes?
Since $M$ is symmetric, we have $M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$ (Eigenvalue decomposition)

We can visualize $M$ as an ellipse with axis lengths determined by the eigenvalues and orientation determined by $R$

Direction of the fastest change

Direction of the slowest change

\[
\omega_{max} \leq \omega_{max}^{1/2}
\]

\[
\omega_{min} \leq \omega_{min}^{1/2}
\]

Classification of image points using eigenvalues of $M$:

- \( \lambda_2 \) and \( \lambda_2 \) are small; \( R \) is almost constant in all directions
- \( \lambda_2 \) increases in all directions

Fast approximation

- Avoid computing the eigenvalues
- \( \alpha \): constant (0.04 to 0.06)

Corner Response Function

\[
R = \det(M) - \alpha \text{trace}(M)^2 = \lambda_1 \lambda_2 - \alpha (\lambda_1 + \lambda_2)^2
\]

Interpreting the Eigenvalues

- \( \lambda_2 \)

\( \lambda_2 \) increases in all directions

Window Function \( w(x,y) \)

\[
M = \sum w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}
\]

- Option 1: uniform window
  - Sum over square window
  - Problem: not rotation invariant

- Option 2: Smooth with Gaussian
  - Gaussian already performs weighted sum
  - Result is rotation invariant

Summary: Harris Detector [Harris88]

- Compute second moment matrix (autocorrelation matrix)
  
  \[
  M_{(x, y)} = g(x) \begin{bmatrix} I_x^2(x, y) & I_x I_y(x, y) \\ I_x I_y(x, y) & I_y^2(x, y) \end{bmatrix}
  \]

- Compute non-maximum suppression

Cornerness function - two strong eigenvalues

\[
R = \det(M) - \alpha \text{trace}(M)^2 = g(I_x)^2 g(I_y)^2 - g(I_x I_y)^2 - \alpha g(I_x^2 + I_y^2)
\]

Harris Detector: Workflow
Harris Detector: Workflow

• Compute corner responses $R$

• Take only the local maxima of $R$, where $R > \text{threshold}$.

Harris Detector: Workflow

• Resulting Harris points

Effect: A very precise corner detector.

Results are well suited for finding stereo correspondences
**Harris Detector: Properties**

- Rotation invariance?
- Scale invariance?

**Corner response** \( R \) is invariant to image rotation

**Ellipse rotates but its shape (i.e. eigenvalues) remains the same**

**Hessian Detector** [Beaudet78]

- Hessian determinant

\[
\text{Hessian}(I) = \begin{bmatrix}
I_{xx} & I_{xy} \\
I_{xy} & I_{yy}
\end{bmatrix}
\]

**Intuition:** Search for strong derivatives in two orthogonal directions

**Note:** these are 2nd derivatives!

**Effect:** Responses mainly on corners and strongly textured areas.

In Matlab:

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


**Hessian Detector** [Beaudet78]

- Hessian determinant

\[
\text{Hessian}(I) = \begin{bmatrix}
I_{xx} & I_{xy} \\
I_{xy} & I_{yy}
\end{bmatrix}
\]

**Slide credit:** Kristen Grauman
Topics of This Lecture

- Local Invariant Features
  - Motivation
  - Requirements, Invariances
- Keypoint Localization
  - Harris detector
  - Hessian detector
- Scale Invariant Region Selection
  - Automatic scale selection
  - Laplacian-of-Gaussian detector
  - Difference-of-Gaussian detector
  - Combinations
- Local Descriptors
  - Orientation normalization
  - SIFT

From Points to Regions...

- The Harris and Hessian operators define interest points.
  - Precise localization
  - High repeatability
- In order to compare those points, we need to compute a descriptor over a region.
  - How can we define such a region in a scale invariant manner?
- I.e. how can we detect scale invariant interest regions?

Naïve Approach: Exhaustive Search

- Multi-scale procedure
  - Compare descriptors while varying the patch size

Slide credit: Krystian Mikolajczyk
Naïve Approach: Exhaustive Search
- Comparing descriptors while varying the patch size
  - Computationally inefficient
  - Inefficient but possible for matching
  - Prohibitive for retrieval in large databases
  - Prohibitive for recognition

\[
 f_A \quad \text{Similarity measure} \quad d(f_A, f_B)
\]

Automatic Scale Selection
- Solution:
  - Design a function on the region, which is “scale invariant” (the same for corresponding regions, even if they are at different scales)
    - Example: average intensity. For corresponding regions (even of different sizes) it will be the same.
  - For a point in one image, we can consider it as a function of region size (patch width)

\[
 f \quad \text{Image 1} \quad \text{Region size} \quad \text{scale} = \frac{1}{2} \quad \text{Region size} \quad \text{Image 2}
\]

Important: this scale invariant region size is found in each image independently!
Automatic Scale Selection

- Function responses for increasing scale (scale signature)

What Is A Useful Signature Function?

- Laplacian-of-Gaussian = “blob” detector

Characteristic Scale

- We define the characteristic scale as the scale that produces peak of Laplacian response


Slide credit: Krystian Mikolajczyk

B. Leibe
• Interest points: Local maxima in scale space of Laplacian-of-Gaussian

\[ L_x(\sigma) + L_y(\sigma) \]

\( \Rightarrow \) List of \((x, y, \sigma)\)

LoG Detector: Workflow
LoG Detector: Workflow

Technical Detail
- We can efficiently approximate the Laplacian with a difference of Gaussians:

\[ L = \sigma^2 \left( G_{\mu}(x, y, \sigma) + G_{\nu}(x, y, \sigma) \right) \]

(D difference of Gaussians)

- Gaussian pyramid.
- No need to compute 2nd derivatives. Gaussians are computed anyway.

Difference-of-Gaussian (DoG)
- Difference of Gaussians as approximation of the LoG.
  - This is used e.g. in Lowe’s SIFT pipeline for feature detection.
- Advantages:
  - No need to compute 2nd derivatives.
  - Gaussians are computed anyway, e.g. in a Gaussian pyramid.

Key point localization with DoG
- Detect maxima of difference-of-Gaussian (DoG) in scale space.
- Then reject points with low contrast (threshold).
- Eliminate edge responses.

DoG - Efficient Computation
- Computation in Gaussian scale pyramid.

Results: Lowe’s DoG
Example of Keypoint Detection

(a) 233x189 image
(b) 832 DoG extrema
(c) 729 left after peak value threshold
(d) 536 left after testing ratio of principle curvatures (removing edge responses)

Harris-Laplace [Mikolajczyk '01]

1. Initialization: Multiscale Harris corner detection
2. Scale selection based on Laplacian (same procedure with Hessian ⇒ Hessian-Laplace)

Harris points

Harris-Laplace points

Summary: Scale Invariant Detection

• Given: Two images of the same scene with a large scale difference between them.
• Goal: Find the same interest points independently in each image.
• Solution: Search for maxima of suitable functions in scale and in space (over the image).

Two strategies
- Laplacian-of-Gaussian (LoG)
- Difference-of-Gaussian (DoG) as a fast approximation

These can be used either on their own, or in combinations with single-scale keypoint detectors (Harris, Hessian).

You Can Try It At Home...

- For most local feature detectors, executables are available online:
  - http://robots.ox.ac.uk/~vgg/research/affine
  - http://www.cs.ubc.ca/~lowe/keypoints/
  - http://www.vision.ee.ethz.ch/~surf

http://www.robots.ox.ac.uk/~vgg/research/affine/detectors.html#binaries
References and Further Reading

• Read David Lowe’s SIFT paper
  - D. Lowe, *Distinctive image features from scale-invariant keypoints*, *IJCV* 60(2), pp. 91-110, 2004

• Good survey paper on Int. Pt. detectors and descriptors

• Try the example code, binaries, and Matlab wrappers
  - Good starting point: Oxford interest point page
    [http://www.robots.ox.ac.uk/~vgg/research/affine/detectors.html#binaries](http://www.robots.ox.ac.uk/~vgg/research/affine/detectors.html#binaries)