Perception | Edges & Points

Autonomous Mobile Robots

Margarita Chli – University of Edinburgh
Paul Furgale, Marco Hutter, Martin Rufli, Davide Scaramuzza, Roland Siegwart
Edge Detection

- Edge contours in the image correspond to important scene contours.
- Ultimate goal of edge detection: an idealized line drawing.

- Edges correspond to sharp changes of intensity
- Change is measured by 1\textsuperscript{st} order derivative in 1D
- Big intensity change $\Rightarrow$ magnitude of derivative is large
- Or 2\textsuperscript{nd} order derivative is zero.
Edge Detection | 1D edge detection

- Image intensity shows an obvious change

\[ I(x) \]

\[ \frac{d}{dx} I(x) \]
**Edge Detection** | solution: smooth first

\[
I(x)
\]

\[
G_\sigma(x)
\]

\[
s(x) = I(x) * G_\sigma(x)
\]

\[
s'(x) = \frac{d}{dx} s(x)
\]

Edges occur at maxima/minima of \( s'(x) \)
**Edge Detection** | derivative theorem of convolution

- \[ s'(x) = \frac{d}{dx} \left( G_\sigma(x) * I(x) \right) = G'_\sigma(x) * I(x) \]

- This saves us one operation:
  \[ I(x) \]

- \[ G'_\sigma(x) = \frac{d}{dx} G_\sigma(x) \]

- \[ s'(x) = G'_\sigma(x) * I(x) \]

Edges occur at maxima/minima of \( s'(x) \).
Edge Detection | zero-crossings

- Locations of Maxima/minima in $s'(x)$ are equivalent to zero-crossings in $s''(x)$

$$I(x)$$

$$G''_\sigma(x) = \frac{d^2}{dx^2} G_\sigma(x)$$

$$s''(x) = G''_\sigma(x) \ast I(x)$$

Edges occur at zero-crossings of $s''(x)$
Edge Detection | 2D Edge detection

- Find gradient of smoothed image in both directions

\[
\nabla S = \nabla (G_\sigma \ast I) = \begin{bmatrix}
\frac{\partial (G_\sigma \ast I)}{\partial x} \\
\frac{\partial (G_\sigma \ast I)}{\partial y}
\end{bmatrix} = \begin{bmatrix}
\frac{\partial G_\sigma}{\partial x} \ast I \\
\frac{\partial G_\sigma}{\partial y} \ast I
\end{bmatrix} = \begin{bmatrix}
G'_\sigma (x)G_\sigma (y) \ast I \\
G_\sigma (x)G'_\sigma (y) \ast I
\end{bmatrix}
\]

- Discard pixels with \(|\nabla S|\) (i.e. edge strength), below a certain threshold

- **Non-maximal suppression**: identify local maxima of \(|\nabla S|\) ⇒ detected edges

\[
I : \text{Original image ("Lenna")}
\]

\[
|\nabla S| : \text{Edge strength}
\]

\[
\text{Thresholding}\mid \nabla S \mid
\]

\[
\text{Non-maximal suppression} \Rightarrow \text{edge image}
\]

Usually use a separable filter such that:

\[
G_\sigma (x, y) = G_\sigma (x)G_\sigma (y)
\]
Point Features | example: create a panorama

Generated using AUTOSTITCH (freeware)

Images from [Brown and Lowe, ICCV 2003]

How to create a panorama:

- detect corresponding points across images in order to align them
  - We need to:
    - detect the same points independently in different images ⇒ repeatable detector
    - identify the correct correspondence of each point ⇒ reliable & distinctive descriptor

- Point features used in robot navigation, object/place recognition, 3D reconstruction, …
How do we identify corners?

Key: around a corner, the image gradient has two or more dominant directions

Shifting a window in any direction should give a large change in intensity in at least 2 directions.

“flat” region: no intensity change

“edge”: no change along the edge direction

“corner”: significant change in at least 2 directions
Point Features | how do we implement this?

- Two image patches of size $P$ one centered at $(x, y)$ and one centered at $(x + \Delta x, y + \Delta y)$

The Sum of Squared Differences between them is:

$$SSD(\Delta x, \Delta y) = \sum_{x, y \in P} (I(x, y) - I(x + \Delta x, y + \Delta y))^2$$

- Let $I_x = \frac{\partial I(x, y)}{\partial x}$ and $I_y = \frac{\partial I(x, y)}{\partial y}$. Approximating with a 1st order Taylor expansion:

$$I(x + \Delta x, y + \Delta y) \approx I(x, y) + I_x(x, y)\Delta x + I_y(x, y)\Delta y$$

- This produces the approximation

$$SSD(\Delta x, \Delta y) \approx \sum_{x, y \in P} (I_x(x, y)\Delta x + I_y(x, y)\Delta y)^2$$

- Which can be written in a matrix form as

$$SSD(\Delta x, \Delta y) \approx \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} M \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$
Point Features | how do we implement this?

\[ SSD(\Delta x, \Delta y) \approx [\Delta x \quad \Delta y] M \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \]

- \( M \) is the “second moment matrix”

\[ M = \sum_{x,y \in P} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \]

- Since \( M \) is symmetric \( \Rightarrow \) if \( \lambda_1 \) and \( \lambda_2 \): the eigenvalues of \( M \) \( \Rightarrow \) \( M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R \)

- The Harris detector analyses \( \lambda_1 \) and \( \lambda_2 \) to decide if we are in presence of a corner or not \( \Rightarrow \) i.e. looks for large intensity changes in at least 2 directions

- Visualize \( M \) as an ellipse with axis-lengths determined by \( \lambda_1 \) and \( \lambda_2 \), and orientation determined by \( R \):

\[ \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} M \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = const \]
Does the patch $P$ describe a corner or not?

- **No structure:** $\lambda_1 \approx \lambda_2 \approx 0$
  SSD is almost constant in all directions, so it’s a flat region

- **1D structure:** $\lambda_1 \gg \lambda_2$ is large (or vice versa)
  SSD has a large variation only in one direction, which is the one perpendicular to the edge.

- **2D structure:** $\lambda_1, \lambda_2$ are both large
  SSD has large variations in all directions and then we are in presence of a corner.

Computation of $\lambda_1$ and $\lambda_2$ is expensive $\Rightarrow$ use “cornerness function” instead:

$$C = \lambda_1 \lambda_2 - \kappa (\lambda_1 + \lambda_2)^2 = \text{det}(M) - \kappa \cdot \text{trace}^2(M)$$

$k$ is between 0.04 and 0.15

- Last step of Harris corner detector: extract local minima of the cornerness function
Point Features | harris corner properties

- Harris detector: probably the most widely used & known corner detector
- The detection is invariant to
  - Rotation
  - Linear intensity changes
  - note: to make the matching invariant to these we need a suitable descriptor and matching criterion (e.g. SSD on patches is not rotation- or affine- invariant)
- The detection is NOT invariant to
  - Scale changes
- Geometric affine changes: an image transformation which distorts the neighborhood of the corner, can distort its ‘cornerness’ response
Point Features | SIFT features [Lowe, IJCV 2004]

- **SIFT**: Scale Invariant Feature Transform
- SIFT features are reasonably invariant to changes in: rotation, scaling, small changes in viewpoint, illumination
- Very powerful in capturing + describing distinctive structure, but also computationally demanding

**Main SIFT stages:**
1. Extract keypoints + scale
2. Assign keypoint orientation
3. Generate keypoint descriptor
**Point Features | SIFT detector (keypoint location + scale)**

Keypoint detection

1. Scale-space pyramid: subsample and blur original image
2. Difference of Gaussians (DoG) pyramid: subtract successive smoothed images
3. Keypoints: local extrema in the DoG pyramid
Point Features | SIFT orientation and descriptor

**Keypoint orientation** (to achieve rotation invariance)

- Sample intensities around the keypoint
- Compute a histogram of orientations of intensity gradients
- **Keypoint orientation** = histogram peak

**Keypoint descriptor**

- SIFT descriptor: 128-long vector
- Describe all gradient orientations relative to the Keypoint Orientation
- Divide keypoint neighborhood in 4×4 regions & compute orientation histograms along 8 directions
- SIFT descriptor: concatenation of all 4×4×8 (=128) values
FAST: Features from Accelerated Segment Test

- Studies intensity of pixels on circle around candidate pixel C

- Area centered at C is a FAST corner if a set of N contiguous pixels on a circle are significantly darker/brighter than C

- Typical FAST mask: test for 12 contiguous pixels on a 16-pixel circle

- Very fast detector
Features for Robotics | BRIEF descriptor [Calonder et. al, ECCV 2010]

- **BRIEF**: Binary Robust Independent Elementary Features
- Goal: high speed (in description and matching)

- **BRIEF Binary** descriptor = concatenation of simple intensity tests between random pixel pairs
- Pattern of random pixel pairs: pre-selected

- Not scale/rotation invariant (extensions exist…)
- Allows **very fast** Hamming Distance matching: count the number of bits that are different in the descriptors matched
- BRISK: **Binary Robust Invariant Scalable Keypoints**
- Detect corners in scale-space based on FAST
- High-speed (faster than SIFT, SURF)
- Rotation and scale invariant
Features for Robotics | BRISK descriptor

- **Binary**, formed by pairwise intensity comparisons (like BRIEF)
- **Pattern** defines intensity comparisons in the keypoint neighborhood
- **Red circles**: size of the smoothing kernel applied
- **Blue circles**: smoothed pixel value used
- Compare short- and long-distance pairs for orientation assignment & descriptor formation
- Detection and descriptor speed: \( \approx 10 \) times faster than SURF (and even faster than SIFT)
- Slower than BRIEF, but scale- and rotation- invariant
Open-source code for FAST, BRIEF, BRISK and many more, available at the OpenCV library