Correspondence
We know how to detect good points
Next question: How to match them?

Two interrelated questions:
1. How do we describe each feature point?
2. How do we match descriptions?
Feature descriptor
Feature matching

• Measure the distance between (or similarity between) every pair of descriptors

<table>
<thead>
<tr>
<th></th>
<th>$y_1$</th>
<th>$y_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$d(x_1, y_1)$</td>
<td>$d(x_1, y_2)$</td>
</tr>
<tr>
<td>$x_2$</td>
<td>$d(x_2, y_1)$</td>
<td>$d(x_2, y_2)$</td>
</tr>
</tbody>
</table>
Invariance vs. discriminability

• Invariance:
  • Distance between descriptors should be small even if image is transformed

• Discriminability:
  • Descriptor should be highly unique for each point (far away from other points in the image)
Image transformations

- Geometric
  - Rotation
  - Scale

- Photometric
  - Intensity change
Invariance

• Most feature descriptors are designed to be invariant to
  • Translation, 2D rotation, scale

• They can usually also handle
  • Limited 3D rotations (SIFT works up to about 60 degrees)
  • Limited affine transformations (some are fully affine invariant)
  • Limited illumination/contrast changes
How to achieve invariance

Design an invariant feature descriptor

• Simplest descriptor: a single 0
  • What’s this invariant to?
  • Is this discriminative?

• Next simplest descriptor: a single pixel
  • What’s this invariant to?
  • Is this discriminative?
The aperture problem
The aperture problem

- Use a whole patch instead of a pixel?
SSD

• Use as descriptor the whole patch
• Match descriptors using euclidean distance
• $d(x, y) = ||x - y||^2$
SSD
NCC - Normalized Cross Correlation

- Lighting and color change pixel intensities
- Example: increase brightness / contrast
- \( I' = \alpha I + \beta \)
- Subtract patch mean: invariance to \( \beta \)
- Divide by norm of vector: invariance to \( \alpha \)
- \( x' = x - <x> \)
- \( x'' = \frac{x'}{||x'||} \)
- similarity = \( x'' \cdot y'' \)
NCC - Normalized cross correlation
Basic correspondence

• Image patch as descriptor, NCC as similarity

• Invariant to?
  • Photometric transformations?
  • Translation?
  • Rotation?
Rotation invariance for feature descriptors

• Find dominant orientation of the image patch
  • This is given by $x_{\text{max}}$, the eigenvector of $\mathbf{M}$ corresponding to $\lambda_{\text{max}}$ (the larger eigenvalue)
  • Rotate the patch according to this angle
Multiscale Oriented PatcheS descriptor

Take 40x40 square window around detected feature
- Scale to 1/5 size (using prefiltering)
- Rotate to horizontal
- Sample 8x8 square window centered at feature
- Intensity normalize the window by subtracting the mean, dividing by the standard deviation in the window

Adapted from slide by Matthew Brown
Detections at multiple scales

Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matter images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.
Invariance of MOPS

- Intensity
- Scale
- Rotation
Color and Lighting
Out-of-plane rotation
Discussion
Better representation than color: Edges
Towards a better feature descriptor

• Match *pattern of edges*
  • Edge orientation – clue to shape

• Be resilient to *small deformations*
  • Deformations might move pixels around, but slightly
  • Deformations might change edge orientations, but slightly
Invariance to deformation by quantization

Between 30 and 45
Invariance to deformation by quantization

\[ g(\theta) = \begin{cases} 
0 & \text{if } 0 < \theta < \frac{2\pi}{N} \\
1 & \text{if } \frac{2\pi}{N} < \theta < \frac{4\pi}{N} \\
2 & \text{if } \frac{4\pi}{N} < \theta < \frac{6\pi}{N} \\
\vdots & \\
N - 1 & \text{if } 2(N - 1)\frac{\pi}{N} < \theta < \frac{2N\pi}{N} 
\end{cases} \]
Spatial invariance by histograms

2 blue balls, one red box

<table>
<thead>
<tr>
<th></th>
<th>balls</th>
<th>boxes</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Rotation Invariance by Orientation Normalization

- Compute orientation histogram
- Select dominant orientation
- Normalize: rotate to fixed orientation

[Lowe, SIFT, 1999]
The SIFT descriptor

SIFT – Lowe IJCV 2004
Basic idea:

- DoG for scale-space feature detection
- Take 16x16 square window around detected feature
  - Compute gradient orientation for each pixel
  - Throw out weak edges (threshold gradient magnitude)
  - Create histogram of surviving edge orientations

Adapted from slide by David Lowe
SIFT descriptor

Create histogram

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor

Adapted from slide by David Lowe
SIFT vector formation

• Computed on rotated and scaled version of window according to computed orientation & scale
  • resample the window
• Based on gradients weighted by a Gaussian
Ensure smoothness

- Trilinear interpolation
  - a given gradient contributes to 8 bins: 4 in space times 2 in orientation
Reduce effect of illumination

• 128-dim vector normalized to 1
• Threshold gradient magnitudes to avoid excessive influence of high gradients
  • after normalization, clamp gradients >0.2
  • renormalize
Properties of SIFT

Extraordinarily robust matching technique

- Can handle changes in viewpoint
  - Up to about 60 degree out of plane rotation
- Can handle significant changes in illumination
  - Sometimes even day vs. night (below)
- Fast and efficient—can run in real time
- Lots of code available:
Summary

• Keypoint detection: repeatable and distinctive
  • Corners, blobs, stable regions
  • Harris, DoG

• Descriptors: robust and selective
  • spatial histograms of orientation
  • SIFT and variants are typically good for stitching and recognition
  • But, need not stick to one
Which features match?
Feature matching

Given a feature in $I_1$, how to find the best match in $I_2$?

1. Define distance function that compares two descriptors
2. Test all the features in $I_2$, find the one with min distance
Feature distance

How to define the difference between two features $f_1, f_2$?

• Simple approach: $L_2$ distance, $||f_1 - f_2||$
• can give good scores to ambiguous (incorrect) matches
Feature distance

How to define the difference between two features $f_1, f_2$?

- Better approach: ratio distance $= ||f_1 - f_2|| / ||f_1 - f_2'||$
  - $f_2$ is best SSD match to $f_1$ in $I_2$
  - $f_2'$ is 2nd best SSD match to $f_1$ in $I_2$
  - gives large values for ambiguous matches
Dense correspondence
Dense correspondence

• Goal: Assign disparity value to each pixel
• Problem: most pixels will be ambiguous
• Solution: propagate from unambiguous to ambiguous pixels
• Basic idea: nearby pixels likely to have same disparity (smoothness)
Dense correspondence

• Goal:
  • Assign disparity value to each pixel

• Basic idea:
  • Disparity image should be smooth

• Energy minimization
  • $\min E(d)$, where $d$ is disparity image
  • $E(d) = E_{\text{data}}(d) + E_{\text{smoothness}}(d)$

$E_{\text{data}}(d)$: scores based on NCC (for example)

$E_{\text{smoothness}}(d) = \sum_{i,j} \rho(d(i,j) - d(i, j + 1)) + \rho(d(i,j) - d(i + 1, j))$
Markov Random Fields

- Probabilistic model
- Undirected graphical model

\[ P(d) \propto e^{-E(d)} \]

- Undirected graph with nodes and edges
- Unary potential on nodes = data term
- Binary potential on edges = smoothness term

\[ E(d) = \sum_{(i,j) \in V} \phi_u(d(i,j)) + \sum_{((i,j),(k,l)) \in E} \phi_b(d(i,j),d(k,l)) \]
Optimizing MRFs

• NP-Hard
• Approximate solutions
  • Message passing
  • Graph cut-based solutions
Dense correspondence with MRFs
Dense correspondence

• Goal: Assign disparity value to each pixel
• Problem: most pixels will be ambiguous
• Solution: propagate from unambiguous to ambiguous pixels
• Basic idea: nearby pixels likely to have same disparity (*smoothness*)
Dense correspondence

• Obtain disparity through optimization

\[ d^* = \arg\min_d \sum_{(i,j)} E_{data}(d(i, j)) + \sum_{(i,j),(k,l) \in \mathcal{N}} E_{smooth}(d(i, j), d(k, l)) \]

Based on e.g. NCC distance

\[(d(i,j) - d(k,l))^2\]
Detour: Graphical models

• Probabilistic models with graphs
• Nodes are variables
• Edges determine dependency structure
• independent of given all

It is 8 am

I wake up

Alarm rings

Naughty roommate

Roommate sets up prank
Markov Random Fields (MRFs)

- Probabilistic model
- Represented by graph
- Each node is random variable
- Edges represent *dependence structure*
  - independent of \textit{given all}
Markov Random Fields (MRFs)

- Hammersley-clifford theorem

\[ P(X = x) \propto e^{-E(x)} \]

\[ E(x) = \sum_{i \in \mathcal{V}} \phi_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \psi_{ij}(x_i, x_j) \]

\[ x^* = \arg \max_x P(X = x) = \arg \min_x E(x) \]
Dense correspondence as MRFs

- Obtain disparity through optimization
- Random variable: disparity
- Find most likely disparity

\[ d^* = \arg \min_d \sum_{(i,j)} E_{data}(d(i, j)) + \sum_{(i,j), (k,l) \in N} E_{smooth}(d(i, j), d(k, l)) \]

- Based on e.g. NCC distance
- \[(d(i,j) - d(k,l))^2\]
Aligning depth boundaries to image boundaries

• Some pairs more likely to have same disparity
• $w(i,j) \ (d(i,j) - d(k,l))^2$
• $w(i,j) = 0$ for edges
• *Conditional* Random Field (CRF)
Other applications of MRFs / CRFs

Optimizing MRFs

- NP-Hard
- Approximate solutions
  - Message passing
  - Mean field-based inference
  - Graph cut-based solutions

Dense correspondence with MRFs