Patch Descriptors

CSE P 576
Larry Zitnick (larryz@microsoft.com)
Many slides courtesy of Steve Sietz

How do we describe an image patch?

Patches with similar content should have similar descriptors.

What do human use?

Gabor filters...

... and many other things.
Encoding the gradients

Basic idea:
- Take 16x16 square window around detected feature
- Compute gradient orientation for each pixel
- Create histogram over edge orientations weighted by magnitude

Scale Invariant Feature Transform

- 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

SIFT descriptor

Adapted from slide by David Lowe
**SIFT descriptor**

Full version
- 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
- Threshold normalize the descriptor:

\[ \sum_i d_i^2 = 1 \text{ such that: } d_i < 0.2 \]

Why?

Adapted from slide by David Lowe

**Properties of SIFT**

Extraordinarily robust matching technique
- Can handle changes in viewpoint
  - Up to about 30 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


**When does SIFT fail?**

Patches SIFT thought were the same but aren’t:

**Other methods: Daisy**

Circular gradient binning

- 1 Ring 6 Segments
- 1 Ring 8 Segments
- 2 Rings 6 Segments
- 2 Rings 8 Segments

Daisy

Picking the best DAISY. S. Windor, G. Hua, M. Brown, CVPR 09
Other methods: SURF

For computational efficiency only compute gradient histogram with 4 bins:

![SURF descriptor example]

**SURF**: Speeded Up Robust Features
Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, ECCV 2006

Other methods: BRIEF

Randomly sample pair of pixels a and b.
1 if a > b, else 0. Store binary vector.

![BRIEF example]

**BRIEF**: binary robust independent elementary features,
Calonder, V Lepetit, C Strecha, ECCV 2010

**Feature distance**

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

- Simple approach is $SSD(f_1, f_2)$
  - sum of square differences between entries of the two descriptors
  - can give good scores to very ambiguous (bad) matches

**Feature distance**

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

- Better approach: ratio distance = $SSD(f_1, f_2) / SSD(f_1, f_2')$
  - $f_2$ 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 (~1) for ambiguous matches
Eliminating bad matches

Throw out features with distance > threshold
  • How to choose the threshold?

True/false positives

The distance threshold affects performance
  • True positives = # of detected matches that are correct
    – Suppose we want to maximize these—how to choose threshold?
  • False positives = # of detected matches that are incorrect
    – Suppose we want to minimize these—how to choose threshold?

Evaluating the results

How can we measure the performance of a feature matcher?

the matcher correctly found a match
  # true positives matched
  # true positives
features that really do have a match

false positive rate
# false positives matched
# false negatives
features that really don't have a match

the matcher said yes when the right answer was no

Evaluating the results

How can we measure the performance of a feature matcher?

ROC curves
  • Generated by counting # current/incorrect matches, for different thresholds
  • Want to maximize area under the curve (AUC)
  • Useful for comparing different feature matching methods
  • For more info: http://en.wikipedia.org/wiki/Receiver_operating_characteristic
Some actual ROC curves