Scale Invariant Feature Transform - SIFT

Raul Queiroz Feitosa
Outline

- Overview
- Keypoint detector
- Descriptor
- Matching
- Alternative Approaches
The man behind the method

The Correspondence Problem

Example:
The Correspondence Problem

Introduction: difficult to locate an accurate match …
The Correspondence Problem

Promising image entities

Before starting the search for correspondences, it is convenient to locate entities, (interest points or keypoints) with good characteristics for an accurate match:

1. Distinctiveness: locally separable
2. Invariance: identifiable under the expected geometric and radiometric distortions
3. Stability: appears in both images
4. Seldomness: globally separable
Advantages

- **Invariance** to
  - Image scale,
  - Rotation on the plane.

- **Robustness** to
  - Affine distortion,
  - Change in 3D viewpoint,
  - Noise.

- **Locality**: feature are local, so robust to occlusion and clutter

- **Distinctiveness**: individual features can be matched to a large database of objects

- **Quantity**: many features can be generated for even small objects.
Basic Steps

1. Scale-space extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor construction
5. Matching keypoints

Interest-point detection

Description

Matching
Gaussian Image Pyramid

The scale-space representation:

The input image is successively smoothed with a Gaussian kernel and subsampled, forming an image pyramid.
DoG Pyramid

Overview:

Scale (next octave)

Scale (first octave)

Gaussian

Difference of Gaussian (DOG)
Building the Pyramids

Algorithm:

1. The image size is doubled

This implies in a smoothing corresponding to $\sigma=0.5$. Thus, the first smoothing should use $\sigma=0.5$ and from then on $\sigma=1.6$. 
Building the Pyramids

Algorithm:

2. Build the Gaussian pyramid

first octave

second octave

third octave
Building the Pyramids

Algorithm:

3. Build the DoG pyramid - $D(x,y,\sigma)$

first octave

second octave

third octave
Outline

- Overview
- Keypoint detector
- Descriptor
- Matching
- Alternative Approaches
SIFT Keypoint Detector

Detecting Extrema:
SIFT Keypoint Detector

Scale-space extrema detection - algorithm:

4. Extrema in 3D
Accurate Keypoint Localization

- From difference-of-Gaussian local extrema detection we obtain approximate values for keypoints.
- Originally these approximations were used directly.
- For an improvement in matching and stability, fitting to a 3D quadratic function is used.
- This is specially important to localize keypoints detected on higher octaves.
Filtering out weak keypoints

- **Keypoints with low contrast are discarded**
  - For image intensities normalized in [0 1], discard a keypoint if
    \[ |D(\hat{x})| < 0.03 \]

- **Keypoints along edges are discarded**
  - How?
SIFT Examples

The initial 832 keypoints locations at extrema of DoG. Vectors indicating scale, orientation, and location.

After applying a threshold on minimum contrast, 729 keypoints remain.

The final 536 keypoints after threshold on ratio of principal curvatures.
Outline

- Overview
- Keypoint detector
- Descriptor
- Matching
- Alternative Approaches
Assigning an Orientation

- Take the image $L$ at the scale closest to the keypoint.
- Calculate magnitude and orientations of gradient around the key point according to

\[
m(x, y) = \sqrt{(L(x + 1, y) - L(x - 1, y))^2 + (L(x, y + 1) - L(x, y - 1))^2}
\]

\[
\theta(x, y) = \tan^{-1} \left( \frac{(L(x, y + 1) - L(x, y - 1))}{(L(x + 1, y) - L(x - 1, y))} \right)
\]

- Compute the orientation histogram with 36 bins. Each sample is weighted by
  - gradient magnitude and
  - Gaussian circular window with a $\sigma$ equal to 1.5 times scale of keypoint
Assigning an Orientation

Gradients

Weights

Orientation bins

Frequency
Assigning an Orientation

- Highest peak in orientation histogram is found along with any other peaks within 80% of highest peak → more than one orientation may be assigned to a key point.

- A parabola is fit to the 3 closest histogram values to each peak and its maximum is taken → more accurate peak detection.

- Thus each keypoint has 4 dimensions:
  - \( x \) location,
  - \( y \) location,
  - \( \sigma \) scale, and
  - orientation
SIFT Descriptor

1. Compute the gradient magnitude and orientation at each image sample point around the keypoint. Magnitudes are weighted by a Gaussian window, with $\sigma = 1/2$ width of the descriptor window.

- to achieve orientation invariance, the coordinates of the descriptor and the gradient orientations are rotated relative to the keypoint orientation.
SIFT Descriptor

2. Form orientation histograms summarizing the contents over 4x4 subregions, with the length of each arrow given by the sum of the gradient magnitudes near that direction within the region.
SIFT Descriptor

3. Avoiding boundary effects between histograms
   - Weight equal to $1 - d$, for each of the 3 dimensions where $d$ is the distance of a sample to the center of a bin
SIFT Descriptor

4. Ensuring invariance to illumination
   - Vector is normalized to 1; its components are thresholded to no larger than 0.2 and then the vector is normalized again.

![Image of SIFT Descriptor with gradients and keypoint descriptor]
SIFT Descriptor

“This figure shows a $2 \times 2$ descriptor array computed from an $8 \times 8$ set of samples, whereas the experiments in this paper use $4 \times 4$ descriptors computed from a $16 \times 16$ sample array.” Lowe
SIFT Descriptor

Keypoint Descriptor provides invariance to

- Scale (by using the DoG pyramid)
- Illumination (by using the gradients + normalization)
- Rotation (by rotating the description relative to the main direction)
- 3D camera viewpoint (to a certain extent)
Outline

- Overview
- Keypoint detector
- Descriptor
- Matching
- Alternative Approaches
SIFT- Keypoint Matching

Up to this point we have:

- Found rough approximations for features by looking at the difference-of-Gaussians
- Localized the keypoint more accurately
- Thresholded poor keypoints
- Determined the orientation of a keypoint
- Calculated a 128 feature vector for each keypoint

How to find corresponding keypoints on a pair (or more) of images?
SIFT- Keypoint Matching

Keypoint Matching

- The *dissimilarity measure* is given by the Euclidean distance between descriptors.
- Independently match all keypoints in all octaves in one image with all keypoints in all octaves in other image.
- Take the closest neighbor.
- If the ratio of closest nearest neighbor with second closest nearest neighbor, is greater than 0.8, discard them.
SIFT- Keypoint Matching

Example of Matching
Outline

- Overview
- Keypoint detector
- Descriptor
- Matching
- Alternative Approaches
SIFT Related References

**SIFT**
- SUSAN, S., JAIN, A., SHARMA, A., VERMA, A., JAIN, S., Fuzzy match index for scale-invariant feature transform (SIFT) features with application to face recognition with wek supervision, IET Image Processing, v. 9, n. 11, pp. 951-958m 2915

**SURF**

**DAISY**

**HoG**
Exercise with SIFT

- Download the SIFTDemo.
- Read the README file.
- Try the MATLAB function *match* using as input arguments any image stereo pair contained in the package.
- Try the MATLAB functions *sift* and *showkeys*.
Recognition