Local Features Tutorial: Nov. 8, '04

Local Features Tutorial

References:

- Matlab SIFT tutorial (from course webpage)
- Lowe, David G. 'Distinctive Image Features from Scale Invariant Features', International Journal of Computer Vision, Vol. 60, No. 2, 2004, pp. 91-110

Local Features Tutorial

Previous week: View based models for object recognition

- The problem: Build a model that captures $general$ properties of eye appearance that we can use to identify eyes (though the approach is general, and does not depend on the particular object class).

- Generalized model of eye appearance based on PCA. Images taken from same pose and normalized for contrast.

- Demonstrated to be useful for classification, key property: the model can find instances of eyes it has never seen before.

Local Features Tutorial

Today: Local features for object recognition

- The problem: Obtain a representation that allows us to find a particular object we've encountered before (i.e. "find Paco's mug" as opposed to "find a mug").

- Local features based on the appearance of the object at particular interest points.

- Features should be reasonably invariant to illumination changes, and ideally, also to scaling, rotation, and minor changes in viewing direction.

- In addition, we can use local features for matching, this is useful for tracking and 3D scene reconstruction.

Local Features Tutorial

Key properties of a good local feature:

- Must be highly distinctive, a good feature should allow for correct object identification with low probability of mismatch. *Question: How to identify image locations that are distinctive enough?*.

- Should be easy to extract.

- Invariance, a good local feature should be tolerant to
◇ Image noise
◇ Changes in illumination
◇ Uniform scaling
◇ Rotation
◇ Minor changes in viewing direction
Question: How to construct the local feature to achieve invariance to the above?

- Should be easy to match against a (large) database of local features.

SIFT features

Scale Invariant Feature Transform (SIFT) is an approach for detecting and extracting local feature descriptors that are reasonably invariant to changes in illumination, image noise, rotation, scaling, and small changes in viewpoint.

Detection stages for SIFT features:

- Scale-space extrema detection

- Keypoint localization

- Orientation assignment

- Generation of keypoint descriptors.

In the following pages we'll examine these stages in detail.

Scale-space extrema detection

Interest points for SIFT features correspond to local extrema of difference-of-Gaussian filters at different scales.

Given a Gaussian-blurred image

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y),$$

where

$$G(x, y, \sigma) = 1/(2\pi\sigma^2) \exp^{-(x^2+y^2)/\sigma^2}$$

is a variable scale Gaussian, the result of convolving an image with a difference-of-Gaussian filter

$$G(x, y, k\sigma) - G(x, y, \sigma)$$

is given by

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \qquad (1)$$

Which is just the difference of the Gaussian-blurred images at scales σ and $k\sigma$.

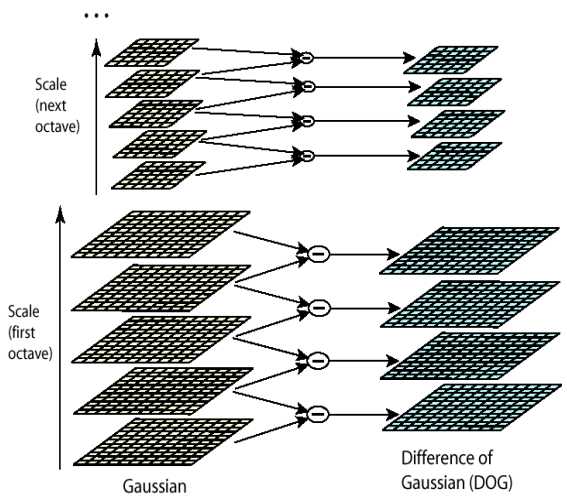


Figure 1: Diagram showing the blurred images at different scales, and the computation of the difference-of-Gaussian images (from Lowe, 2004, see ref. at the beginning of the tutorial)

The first step toward the detection of interest points is the convolution of the image with Gaussian filters at different scales, and the generation of difference-of-Gaussian images from the difference of adjacent blurred images.

Scale-space extrema detection

The convolved images are grouped by octave (an octave corresponds to doubling the value of σ), and the value of k is selected so that we obtain a fixed number of blurred images per octave. This also ensures that we obtain the same number of difference-of-Gaussian images per octave.

Note: The difference-of-Gaussian filter provides an approximation to the scale-normalized Laplacian of Gaussian $\sigma^2\nabla^2 G$. The difference-of-Gaussian filter is in effect a tunable bandpass filter.

Scale-space extrema detection

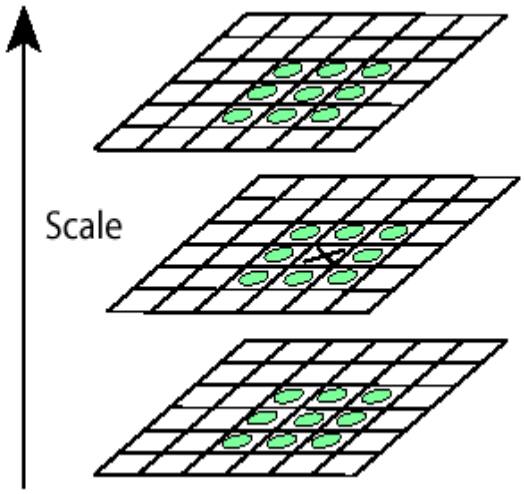


Figure 2: Local extrema detection, the pixel marked $\times$ is compared against its 26 neighbors in a $3 \times 3 \times 3$ neighborhood that spans adjacent DoG images (from Lowe, 2004)

Interest points (called keypoints in the SIFT framework) are identified as local maxima or minima of the DoG images across scales. Each pixel in the DoG images is compared to its 8 neighbors at the same scale, plus the 9 corresponding neighbors at neighboring scales. If the pixel is a local maximum or minimum, it is selected as a candidate keypoint.

Scale-space extrema detection

For each candidate keypoint:

- Interpolation of nearby data is used to accurately determine its position.

- Keypoints with low contrast are removed

- Responses along edges are eliminated

- The keypoint is assigned an orientation

To determine the keypoint orientation, a gradient orientation histogram is computed in the neighborhood of the keypoint (using the Gaussian image at the closest scale to the keypoint's scale). The contribution of each neighboring pixel is weighted by the gradient magnitude and a Gaussian window with a σ that is 1.5 times the scale of the keypoint.

Peaks in the histogram correspond to dominant orientations. A separate keypoint is created for the direction corresponding to the histogram maximum,

and any other direction within 80% of the maximum value.

All the properties of the keypoint are measured relative to the keypoint orientation, this provides invariance to rotation.

SIFT feature representation

Once a keypoint orientation has been selected, the feature descriptor is computed as a set of orientation histograms on 4×4 pixel neighborhoods. The orientation histograms are relative to the keypoint orientation, the orientation data comes from the Gaussian image closest in scale to the keypoint's scale.

Just like before, the contribution of each pixel is weighted by the gradient magnitude, and by a Gaussian with σ 1.5 times the scale of the keypoint.

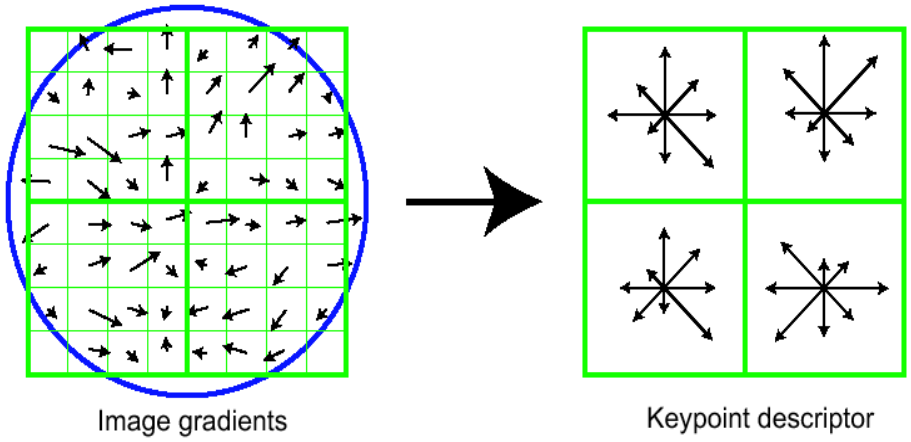


Figure 3: SIFT feature descriptor (from Lowe, 2004)

Histograms contain 8 bins each, and each descriptor contains an array of 4 histograms around the keypoint. This leads to a SIFT feature vector with $4 \times 4 \times 8 = 128$ elements. This vector is normalized to enhance invariance to changes in illumination.

SIFT feature matching

- Find nearest neighbor in a database of SIFT features from training images.

- For robustness, use ratio of nearest neighbor to ratio of second nearest neighbor.

- Neighbor with minimum Euclidean distance $\longrightarrow$ expensive search.

- Use an approximate, fast method to find nearest neighbor with high probability.

Recognition using SIFT features

- Compute SIFT features on the input image

- Match these features to the SIFT feature database

- Each keypoint specifies 4 parameters: 2D location, scale, and orientation.

- To increase recognition robustness: Hough transform to identify clusters of matches that vote for the same object pose.

- Each keypoint votes for the set of object poses that are consistent with the keypoint's location, scale, and orientation.

- Locations in the Hough accumulator that accumulate at least 3 votes are selected as candidate object/pose matches.

- A verification step matches the training image for the hypothesized object/pose to the image using a least-squares fit to the hypothesized location, scale, and orientation of the object.

SIFT matlab tutorial

Gaussian blurred images and Difference of Gaussian images

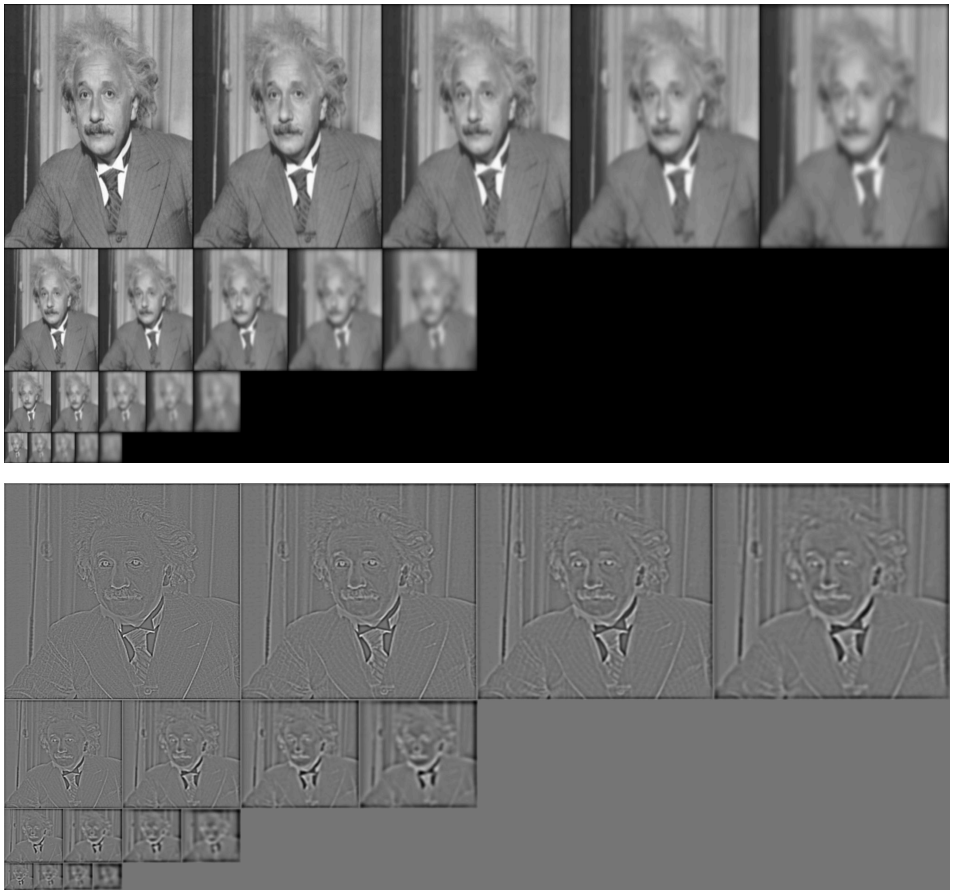
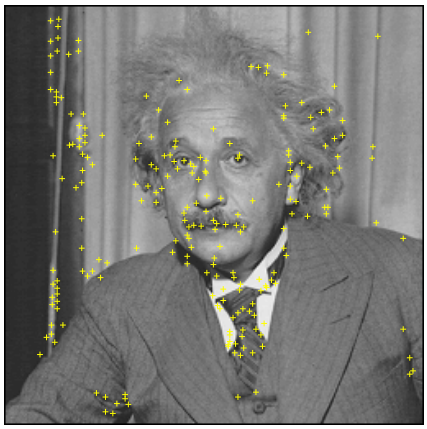


Figure 4: Gaussian and DoG images grouped by octave

SIFT matlab tutorial

Keypoint detection

a) b)



c)

Figure 5: a) Maxima of DoG across scales. b) Remaining keypoints after removal of low contrast points. C) Remaining keypoints after removal of edge responses (bottom).

SIFT matlab tutorial

Final keypoints with selected orientation and scale

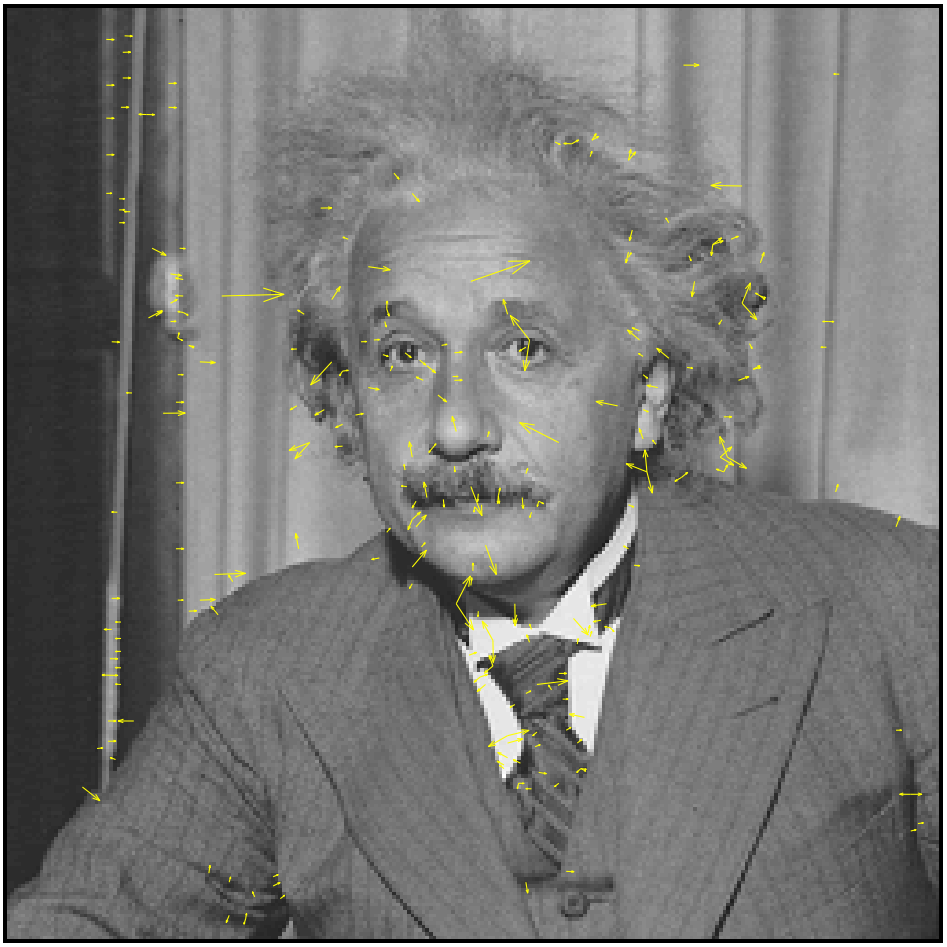


Figure 6: Extracted keypoints, arrows indicate scale and orientation.

SIFT matlab tutorial

Warped image and extracted keypoints

Figure 7: Warped image and extracted keypoints.

The hough transform of matched SIFT features yields

the transformation that aligns the original and warped images:

```
Computed affine transformation from rotated image to
   original image:
>> disp(aff);
    0.7060   -0.7052  128.4230
    0.7057    0.7100 -128.9491
         0         0    1.0000

Actual transformation from rotated image to
   original image:
>> disp(A);
    0.7071   -0.7071  128.6934
    0.7071    0.7071 -128.6934
         0         0    1.0000
```

SIFT matlab tutorial

Matching and alignment of different views using local features.

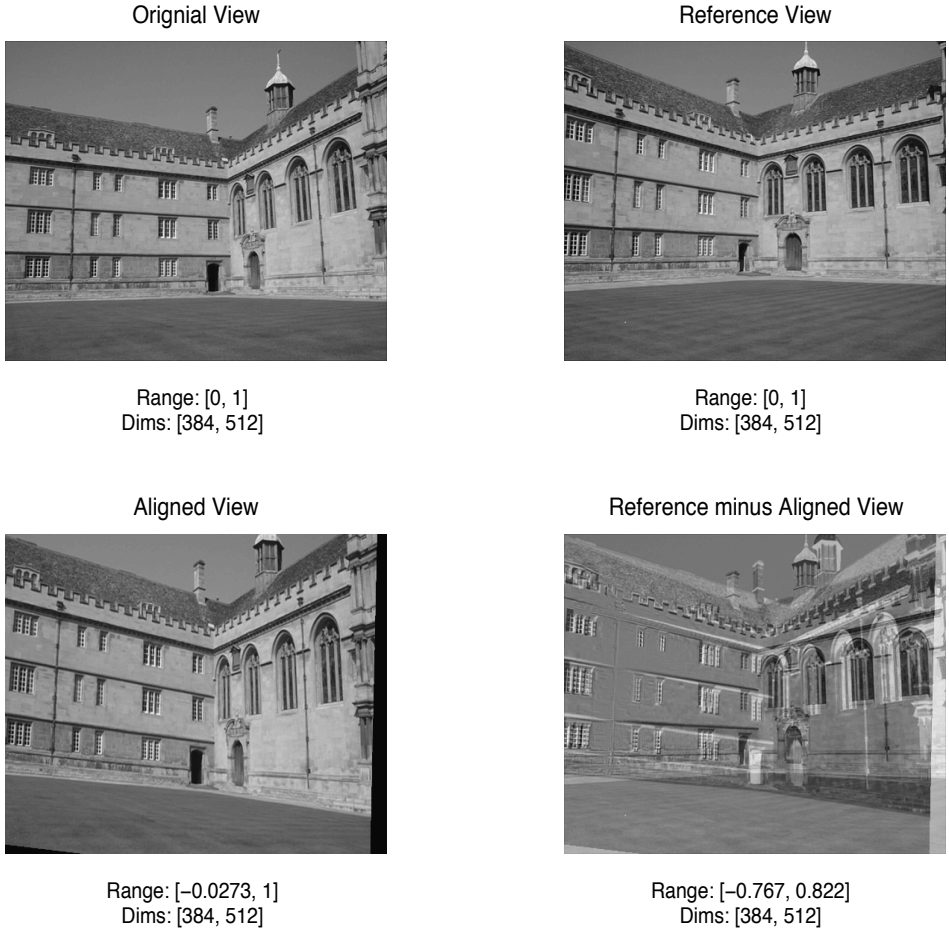


Figure 8: Two views of Wadham College and affine transformation for alignment.

SIFT matlab tutorial

Object recognition with SIFT

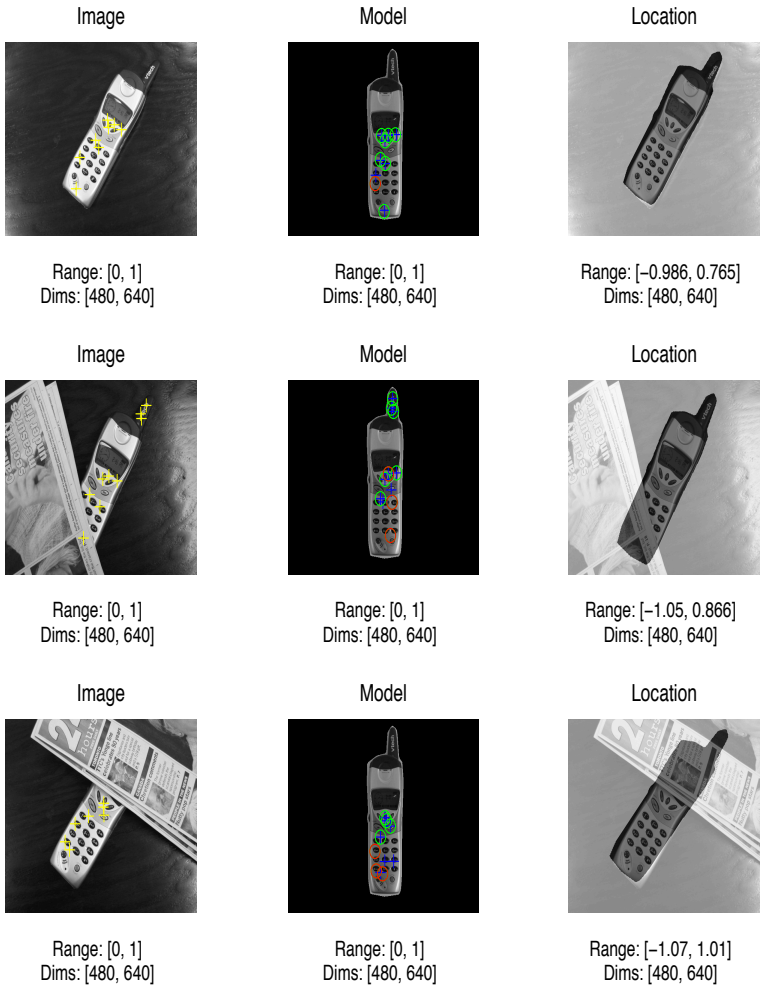


Figure 9: Cellphone examples with different poses and occlusion.

SIFT matlab tutorial

Object recognition with SIFT

Figure 10: Book example, what happens when we match similar features outside the object?

Closing Comments

- SIFT features are reasonably invariant to rotation, scaling, and illumination changes.

- We can use them for matching and object recognition among other things.

- Robust to occlusion, as long as we can see at least 3 features from the object we can compute the location and pose.

- Efficient on-line matching, recognition can be performed in close-to-real time (at least for small object databases).

Closing Comments

Questions:

- Do local features solve the object recognition problem?

- How about distinctiveness? how do we deal with false positives outside the object of interest? (see Figure 10).

- Can we learn new object models without photographing them under special conditions?

- How does this approach compare to the object recognition method proposed by Murase and Nayar? Recall that their model consists of a PCA basis for each object, generated from images taken under diverse illumination and viewing directions; and a representation of the manifold described by the training images in this eigenspace (see the tutorial on Eigen Eyes).