EE 243 Advanced Computer Vision: Homework 3 Report

Tue-Cuong Dong-Si
tdongsi@ee.ucr.edu

1. Introduction

There are two main components in David Lowe's SIFT paper [1]: the SIFT detector and the SIFT descriptor. The detector, like any feature detector such as Harris corner detector, detects distinct features, or *keypoints*, from the image. SIFT descriptor, on the other hand, is a data structure that describes the appearance of the image patch corresponding to the keypoint found by the SIFT detector, for further application such as matching.

2. SIFT detector

2.1. Implementation

The pseudocode of our SIFT detection implementation is presented in Table 1.

Preprocessing: First, the image is loaded, and converted into grayscale color space if the image is in color. Then, the image is normalized into the value of range $[0, 1]$. The image is then upsampled to an image with double size, using linear interpolation. In his paper, Lowe claims that this will increases the number of keypoints. Before that, to avoid aliasing effects, the image is blurred by convoluting with a Gaussian mask with $\sigma_n = 0.5$.

Pyramids: The next step is to find the Gaussian and Difference-of-Gaussian (DoG) pyramids. According to the paper, to detect $s = 3$ intervals per octave, we generate $s + 3$ blurred images to build the Gaussian pyramids, then $s + 2$ DoG images, and finally s images can be used for extrema detection (removing the top and bottom images). The first image of the first octave is generated by blurring the upsampled image with a Gaussian mask with $\sigma = 1.6$, as specified by the paper. The number of octaves used is $O = 4$. The factor k is defined $k = 2^{\frac{1}{s}}$ as in the paper. At the end of each octave, the Gaussian image that is two from the top is subsampled to receive the first image of the next octave.

Local extrema detection & Keypoint localization: The local extrema points are detected by comparing each sample point with its eight neighbors and nine neighbors in the scale above and below. Then, those detected points are tested for sufficient contrast and for edge response removal, with $r = 10$ used for computing the curvature threshold.

Orientation assignment: After finding the keypoints, we have to assign the orientation to each keypoint. For this, we have to compute the gradient

Algorithm 1 SIFT feature detection algorithm

```
Load image, convert to gray scale, normalize.
Convolute the image with Gaussian mask, σ_n.
Double the image size (linear interpolation)

for each octave from 1 to O do
    for each interval from 1 to s + 3 do
        Compute the σ and size of the Gaussian mask.
        Convolute to build Gaussian pyramid.
        Subtract to build DoG pyramid.
    end for
    Subsampling, prepare for the next octave.
end for

for each octave from 1 to O do
    for each interval from 2 to s + 1 do
        Find max-min across space and scale
        for each detected point do
            Check for contrast
            if PASS contrast test then
                Compute the Hessian matrix
                Compute the ratio of principal curvature
                if the ratio ≤ threshold then
                    Save the keypoint.
                end if
            end if
        end for
    end for
end for

for each octave from 1 to O do
    for each interval from 2 to s + 1 do
      Compute the gradient magnitude & orientation
    end for
end for
for each octave from 1 to O do
    for each interval from 2 to s + 1 do
        Compute the Gaussian mask
        for each detected keypoint do
            Create the gradient orientation histogram
            Weight with magnitude and Gaussian mask

            Find the highest peak
            Find the local peaks within 80%.
            Parabola fitting to interpolate peak position.
            Save position, scale, and orientation.
        end for
    end for
end for
return Positions, scales, and orientations
```

orientation over a region about each keypoint and put into a histogram. This histogram will be weighted by the gradient magnitude and a Gaussian mask with a σ that is 1.5 times that of the keypoint's scale.

2.2. Results

Fig. 1 shows the output of our implementation, against the standard MATLAB implementation [2]. There may be a difference in parameters between the two implementation since many parameters are not concretely defined in the paper [1]. In summary, in our implementation, keypoints are more sparsely distributed, compared to Vedaldi's implementation where features are densely localized in some areas.

3. SIFT descriptor

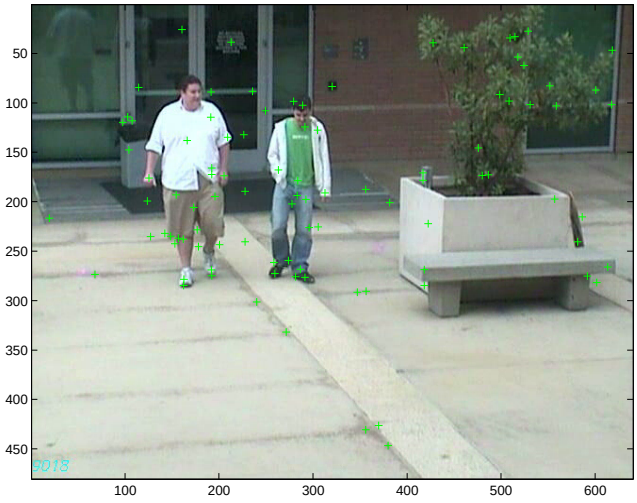
The SIFT descriptor algorithm is running right after SIFT detection algorithm. The pseudocode of our SIFT descriptor implementation is presented in Table 2.

Algorithm 2 SIFT descriptor algorithm

Require: SIFT detection algorithm's outputs

for each detected keypoint **do**
  Rotate the grid coorindates
  for each orientation sample **do**
    Interpolate the gradient
    Compute the weighting by x and y.
    {Trilinear interpolation to avoid boundary effects}
    Compute the Gaussian window
    Compute the weight value
    Accumulate the histogram bins after weighting
  end for
  Normalize descriptor to unit
  {for illumination invariance}
  Threshold the large components to 0.2, and normalize.
  Save the descriptor
end for
return 128×1 vectors of descriptors

Since our SIFT detection algorithm returns different keypoints, it is difficult to compare effectively with the standard MATLAB implementation [2]. Instead, we manually inspect the descriptors of those detected keypoints that are physically close in two implementations' outputs. Results show that they



(a) Our implementation

(b) Vedaldi's implementation

Figure 1. SIFT detector: comparison of Vedaldi's and our implementation.

are mostly close in general, with the norm value of the difference of the two descriptor vectors is less than 0.5 in most cases. It indicates that our SIFT implementation is functional, except that parameters need to be refined to match with the standard implementation.

References

[1] D. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[2] A. Vedaldi. An open implementation of the SIFT

detector and descriptor. Technical report, Citeseer, 2007.