Scale Invariant Feature Transform for *n*-Dimensional Images (*n*-SIFT)

Release 1.00

Warren A. Cheung[1] and Ghassan Hamarneh[2]

December 11, 2007

[1]Bioinformatics Program,
Centre for Molecular Medicine and Therapeutics, Child and Family Research Institute,
University of British Columbia, Vancouver, Canada
E-mail: `wcheung@cmmt.ubc.ca`
[2]Medical Image Analysis Lab, Simon Fraser University, Burnaby, Canada
E-mail: `hamarneh@cs.sfu.ca`

Abstract

This document describes the implementation of several features previously developed[2], extending the 2D scale invariant feature transform (SIFT)[4, 5] for images of arbitrary dimensionality, such as 3D medical image volumes and time series, using ITK[1]. Specifically, we provide a scale invariant implementation of a weighted histogram of gradient feature, a rotationally invariant version of the histogram feature, and a SIFT-like feature, adapted to handle images of arbitrary dimensionality. This paper is accompanied with the source code, example input data, parameters and output data, allowing reproduction of the example results in this paper and the results previously reported[2]. Note that usage of SIFT in the United States is governed by US Patent 6,711,293.

Contents

[1]`http://www.itk.org`

Using filters from the Insight Toolkit, we have adapted techniques from the 2D scale invariant feature transform to handle images of arbitrary dimensionality. Determining point-to-point correspondence between medical images is crucial for comparison and analysis of medical analysis. For example, images of the same subject taken at multiple time points, using the same device, may require alignment due to the inability to exactly replicate the position and orientation of the subject between sessions.

Extending the previous success of using SIFT features for matching 2D medical images[6, 1], we demonstrate the effectiveness of generalising the techniques to images of arbitrary dimensionality. Towards this purpose, we provide three methods of generating scale invariant features, as previously presented[2] – a simple global histogram feature, a rotationally invariant version of the global histogram feature, and *n*-SIFT, the *n*-dimensional adaptation of the SIFT feature.

Currently, appropriate ITK filters and applicable input files exist for 3D and 4D (3D+time) images. Note that usage of SIFT in the United States is governed by US Patent 6,711,293, and a license must be obtained from the University of British Columbia for any commercial applications[2].

1 Scale Invariant Features

The following section provides an overview of the method. For further details, the reader is directed to our previous work[2][3].

The three features share the same basic framework – first, extrema of the difference of Gaussian scale space are identified. At these points, features are computed using the gradients around the points. Once features are computed for two images, we can compare feature vectors and match points that are sufficiently similar.

1.1 Salient Point Detection

To sample the difference of Gaussian space in a scale-invariant manner, Gaussian filter the image multiple times at sigma=$\sigma, k\sigma, k^2\sigma, \ldots$. For each pair of "adjacent" Gaussian filtered images (e.g. $k^j\sigma$ and $k^{j+1}\sigma$), we compute the difference of Gaussian by taking the difference of the images. This generates a series of difference of Gaussian images.

Salient points are extracted by locating local extrema in these images. A voxel is considered an extrema if it is the local maximum or minimum of all voxels orthogonally and diagonally adjacent in one of the difference of Gaussian images, as well as in the corresponding voxels in the preceding and successive images of the difference of Gaussian series.

Scale invariance is achieved by taking the Gaussian filtered image with sigma = σ and downsizing all dimensions by one half. Extrema in the difference of Gaussian space is then examined for this downsized image. To continue sampling scale space, we can repeat the process, testing the quarter-sized filtered version of the original image, and so on.

[2]http://www.cs.ubc.ca/~lowe/keypoints/

[3]http://www.cs.sfu.ca/~hamarneh/ecopy/isbi2007b.pdf or http://dx.doi.org/10.1109/ISBI.2007.356953

1.2 Feature Generation

At each of the extrema located, we examine the gradients near the point to compute a distinctive feature vector. The *global histogram feature* generates a single multi-dimensional histogram to summarise the local gradients. The bins of the histogram cover the hyperspherical coordinates of the gradients. For each gradient of the image, the value placed into the histogram is weighted by the magnitude of the gradient as well as a Gaussian centred on the feature point. The histogram bins form a feature vector, which is normalised.

The *reoriented global histogram feature* incorporates more robustness to rotation. After the histogram is computed, we rotate the bins such that the highest magnitude bin is always in the first position of the vector.

The *n-SIFT feature* uses 4^n multidimensional histograms to summarise hypercubic regions around the feature voxel. We divide the 16^n hypercubic region around the feature voxel into 4^n voxel subregions. Each region is summarised by a multidimensional histogram of the hyperspherical coordinates, weighted by magnitude and a Gaussian centred on the feature point. These vectors for these histograms are then concatenated to form the final n-SIFT feature.

1.3 Feature Matching

To match points from two images, we use the ℓ^2 distance between the feature vectors. To minimise the number of spurious matches, we only accept a match if the ratio of the distance for the best match and the second best match is below a threshold, thus ensuring the second best match is substantially worse than the best match. We also only accept the match if it is a recipricol best match – a feature x matches a feature y if and only if x is the best match for y and y is the best match to x.

2 Implementation Details

The images are loaded using `itk::ImageFileReader` and `itk::ImageSeriesReader` for 3D and 4D images respectively. Gaussian filtering is accomplished using `itk::DiscreteGaussianImageFilter`, and the difference between images is subsequently computed using `itk::SubtractImageFilter`. The keypoint-feature pairs are represented as a `itk::PointSet`, with the features represented using an `Array < float >`.

To generate synthetic test images, we manipulate the original image using `itk::ScalableAffineTransform` to generate scaled and rotated versions of the original image.

The code for this implementation conforms as much as possible to the coding style outlined in the Coding Style Guid by the Insight Software Consortium (June 2004)[4]. The code was documented using doxygen[5].

3 Using the Compiled Program

The included source compiles separate programs for 3D and 4D data, and for each of the feature types. `3Dhistkeys`,`3Drhistkeys` and `3Dsiftkeys` generate features for the histogram feature, the reoriented histogram feature and the n-SIFT feature respectively, for 3-dimensional images. All versions of the program

[4]http://www.insightsoftwareconsortium.org/documents/policies/Style.pdf
[5]http://www.cs.sfu.ca/~hamarneh/software/doxygen/nsift/output/html/

provide two basic modes: comparing two existing images (`--match`, which is default) and testing synthetic images (the option `--synthetic`).

3.1 Matching Two Images

When comparing existing images, the program takes as arguments the filenames of the two images to be compared. It will then compute and match features between the two images, returning a list of corresponding voxel locations. For example, the following command-line will match points from the BrainWeb[3][6] image `brainweb165a10f17.mha` to the image `brainweb1e1a10f20.mha`.

```
3Dsiftkeys --match brainweb165a10f17.mha  brainweb1e1a10f20.mha
```

When point $(x1,y1,z1)$ in the first image matches the point $(x2,y2,z2)$, the match is reported using the format:

```
[x1,y1,z1] => [x2,y2,z2]
```

For example, the above command would result in the following matches

```
[13, 55, 24] => [15, 47, 10]
[36, 32, 23] => [32, 34, 11]
[18, 58, 22] => [17, 55, 22]
[40, 46, 32] => [38, 46, 36]
[32, 56, 32] => [32, 54, 36]
[40, 40, 68] => [40, 40, 66]
[8, 22, 28] => [8, 10, 22]
[20, 56, 20] => [20, 56, 26]
[12, 74, 32] => [12, 72, 36]
[20, 54, 42] => [20, 54, 44]
[32, 36, 36] => [32, 34, 56]
[32, 70, 70] => [34, 70, 80]
```

3.2 Synthetic Performance Tests

When performing a synthetic image test, the user specifies the original test image, as well as scaling (`--scale`) and rotation (`--rotate`). Rotation is by default centred around the origin, but a flag can instead force rotation around the centre of the image(`--rotate-mid`). The program will then compute features for the original and the synthetically modified image, and return a list of corresponding voxel locations. As well, since the ground truth is known in this case, it comoputes the number of matches that are within 1.5, 3.0 and 7.5 of the corresponding location. For example, the following command will test rotation by 10 degrees about the centre of the image.

```
3Dsiftkeys --synthetic --rotate-mid --rotate 10 brainweb165a10f17.mha
```

The output reports the number of extrema found when testing each set of difference of Gaussian images. For example:

[6] `http://www.bic.mni.mcgill.ca/brainweb/`

```
Searching for Extrema in DoG Image 0-1 ( 90 108 90 ) Scale 0.5
Acc. Num Max: 29
Acc. Num Min: 25
Acc. Num Reject: 0
```

reports, for the initial pyramid level (0), and the first set of three difference of Gaussian images, the presence of 29 maxima, 25 minima, and no extrema were rejected due to the voxel intensity being too low. `( 90 108 90 )` reports the dimensions – $90 \times 108 \times 90$ voxels – of the image being tested.

At the end of the run, summary statistics are given to help evaluate the performance of the feature for keypoint matching. First, the total number of keypoints found in each image is given

```
Matching Keypoints
Keypoints1 Found: 449
Keypoints2 Found: 491
```

Here, 449 points were found in the original image (Keypoints 1), and 491 points were found in the rotated synthetic image (Keypoints 2). Then, a summary of how many potential matches at various error levels is provided.

```
Keypoints 2 Matching to Keypoints 1 (<1.5): 274
% of Keypoints 2 Matching (<1.5):  0.558045
```

Here, this describes that 274 of keypoints from the synthetic image are within 1.5 voxels of a keypoint in the original image, which is about 55.8% of all the keypoints generated in the synthetic image. Results are also provided for matches within 3.0 voxels and within 7.5 voxels, as well as for matches between the original image to the synthetic image.

Finally, a summary of the performance of the actual keypoint matches using the scale-invariant features is provided. For this example, the output is:

```
***Features 2 Matches Attempted: 219
Features 2 Matching to Features 1 (<1.5): 190
% of Features 2 Matching (<1.5):  0.86758
```

Here, we report 219 matches generated. Of these, 190 of the 219 match points that are within 1.5 voxels of the "true" match. Again, we also report the results at the error levels of 3.0 voxels and 7.5 voxels.

We can then examine the performance of the various feature types under angular rotation. We can continue to test the image file `brainweb165a10f17.mha`, and summarise the results after differing amounts of rotation and for different feature types. Table 1 reports the number of points correctly matched for the three feature types after 6, 10, 20 and 40 degree rotation, and Table 2 reports the fraction of points correctly matched, at 1.5 voxel error.

4 Software Requirements

The following open source packages were used:

Table 1: Number of points correctly matched for the global histogram feature, the reoriented global histogram feature and the *n*-SIFT feature, with error at most 1.5 voxels

Angle	Global	Reoriented	*n*-SIFT
6	72	31	232
10	19	34	190
20	9	40	72
40	0	31	0

Table 2: Fraction of points correctly matched for the global histogram feature, the reoriented global histogram feature and the *n*-SIFT feature, with error at most 1.5 voxels

Angle	Global	Reoriented	*n*-SIFT
6	0.55	0.37	0.86
10	0.33	0.66	0.86
20	0.31	0.56	0.25
40	0	0.44	0

- Insight Toolkit[7] 2.2.0
- CMake[8] 2.4
- gcc[9] 3.4.4

No further dependencies were used. The software was developed and compiled under the Cygwin[10] development environment for 32-bit Windows XP. It should be sufficient, assuming the presence of CMake and ITK, to invoke `cmake .; make` to build the example executables.

To link the code into your own program, simply include `itkScaleInvariantFeatureImageFilter.h`. You will need to load the images for analysis. Then, using an appropriate instance of `itk::ScaleInvariantFeatureImageFilter` for the image type and dimensionality, you can call the method `getSiftFeatures` to generate the set of points with the scale invariant features, and also call the `MatchKeypointsFeatures` to match two sets of points with features.

By default, the class will generate histogram features. To generate reoriented histogram features, define `REORIENT` when compiling. To generate *n*-SIFT features, define `SIFT_FEATURE` when compiling. See `CMakeLists.txt` for an example of how to rebuild the same program for each feature type using the preprocessor declarations.

5 Conclusion

We have provided a general implementation of the algorithms previously described[2] using ITK, and the example program shows how it can be leveraged to analyse the 3D and 4D (3D time series) images that can be loaded by ITK. The software can be applied to non medical *n*-dimensional images in general. Future work includes adapting the rotational invariance via reorientation to the *n*-SIFT feature, unifying the code to deal

[7] `http://www.itk.org`
[8] `http://www.cmake.org`
[9] `http://gcc.gnu.org`
[10] `http://www.cygwin.com`

more elegantly with dimensionality and feature types, optimising the code for speed by taking advantage of more specialised ITK filters, and providing a data structure for reporting matches.

References

[1] Jian Chen and Jie Tian. Rapid multi-modality preregistration based on SIFT descriptor. In *Proceedings of the 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2006 (EMBC 2006)*, pages 1437–1440, 2006. (document)

[2] Warren Cheung and Ghassan Hamarneh. n-SIFT: n-dimensional scale invariant feature transform for matching medical images. In *Proceedings of the Fourth IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 2007 (ISBI 2007)*, pages 720–723, 2007. (document), 1, 5

[3] Chris Cocosco, Vasken Kollokian, Remi K.-S. Kwan, and Alan Evans. Brainweb: Online interface to a 3d MRI simulated brain database. In *Proceedings of the 3rd International Conference on Functional Mapping of the Human Brain, NeuroImage*, volume 5, page S425, 1997. 3.1

[4] David Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision (ICCV 1999)*, volume 2, page 1150, 1999. (document)

[5] David Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004. (document)

[6] Mehdi Moradi and Purang Abolmaesumi. Medical image registration based on distinctive image features from scale-invariant (SIFT) keypoints. In *Proceedings of 19th Computer-Assisted Radiology and Surgery Conference (CARS 2005)*, page 1292, 2005. (document)