THE IMPROVEMENT OF SCALE-INVARIANT FEATURE TRANSFORM AND IMAGE TRACKING

[1]Chih-Wei Wu (吳治緯), [2]Jian-Jiun Ding (丁建均), [2]Chiou-Shann Fuh (傅楸善)

[1,2] Department of Electrical Engineering,
[3] Department of Computer Science and Information Engineering,
National Taiwan University,
Taipei, Taiwan, R.O.C
E-mail: r07942152@ntu.edu.tw [1], jjding@ntu.edu.tw [2], fuh@csie.ntu.edu.tw [3]

ABSTRACT

The Scale-Invariant Feature Transform (SIFT), has the good properties of extracting features from images. Because the Scale-Invariant Feature Transform can also find the feature points with the change of image size and rotation, in this research, we propose a new method to reduce the time of calculation of the Scale-Invariant Feature Transform. Traditional approaches on this method cannot achieve the real-time interaction. In order to address this problem, we will construct an improved model to deal with this work.

Keywords: Scale-Invariant Feature Transform, image tracking

1. INTRODUCTION

Recently, the technique of image tracking and recognition has been popular for a long time. The Scale-Invariant Feature Transform has the ability of handling different sizes of one sample image and this will help us to get the key-points. After we have the key-points, we can use them to conduct the image tracking and recognition. The original Scale-Invariant Feature Transform can be broadly categorized into four main approaches [5]-[8]:

The examination of extremums in scale space: In this step, we first pre-process images. We use the derivative functions to recognize the points which will not be affected by light and the change of image.
The position of key-points: Based on first step, we need to exclude some unstable key-points.
The direction of key-points: After finishing the second step, we assign several angles on key-points.
The descriptors of key-points: These descriptors are in the region of every key-point.

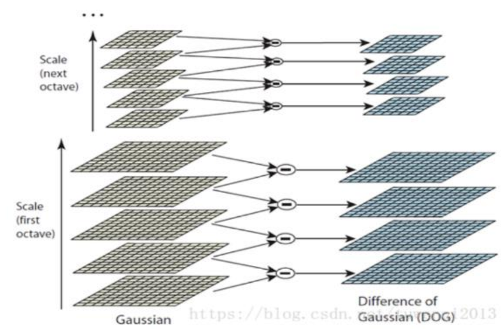

Fig. 1: Mutual subtraction on each Gaussian space.

The Scale-Invariant Feature Transform has worked with many computer vision methods for a long time. However, the calculation of getting the key-points will take much time [9]-[11]. We want to design a new method based on the original technique.

The real time of this approach will be the key problem for practice. In this paper, we propose an improvement approach. The Mutual subtraction on each Gaussian space is Fig. 1.

2. ORIGINAL MODEL

The original model of SIFT points detection has been worked for a long time. First, we want to discuss the previous model [2] [3] [5] [7] [15].
1: We construct the scale space by using the Gaussian function.

$$\frac{\partial G}{\partial \sigma} = \sigma \nabla^2 G \tag{1}$$

We do the differentiation on (1):

$$\sigma \nabla G \approx \frac{G(x,y,k\sigma)-G(x,y,\sigma)}{k\sigma-\sigma} \tag{2}$$

2: After we acquire the Gaussian space functions, we do the mutual subtraction on each Gaussian space function.

3: Find the maxima or minima in every point by comparing one pixel with the surrounding pixels (e.g. size = 3*3*3 pixels).
4: Assign every maximal and minimal point to be key-points and find the directions on every key-point.

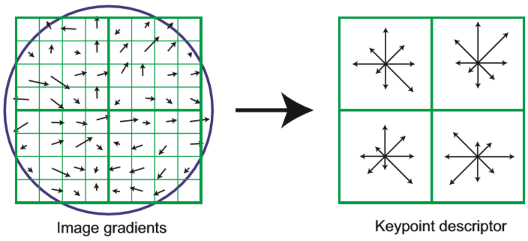

Fig. 2: A key-point descriptor and its eight direction angles

5: Exclude the unstable key-points by using the Harris edge detection.

After we set up the original model of SIFT, we will start constructing the improvement model.

In order to exclude the unstable key-points, we use the method of Harris corner detection. The following are the steps of Harris corner detection:

1. Calculate the first-order derivative of Gaussian function on *x* and *y* direction
2. According to first step, we get the I_x^2, I_y^2, and I_x*I_y $K = 0.04\sim0.06$

$$M = \sum_{x,y} w(x,y)\begin{bmatrix} Ix^2 & IxIy \\ IxIy & Iy^2 \end{bmatrix} \quad E(u,v) \approx [u\ v]M\begin{bmatrix} u \\ v \end{bmatrix} \quad (3)$$

$$R = \det(M) - k(\mathrm{trace}(M))^2 \quad (4)$$

3. Use the Gaussian function to get the S_{xx}, S_{yy}, S_{xy}
4. Get *R*, feature value, in every pixel.
5. Use *R* to find the key-points.

6: We set the size of searching block is 4x4, and find the direction of the key-points, then we can have eight directions of each key-point.
7: Yield the descriptor of the key-points, and we have 4x4x8 = 128 element feature vector of each key-point.
The key-points descriptor and its eight direction angles is Fig. 2.

3. RELATED RESEARCH

The edge detection of the image can be calculated by many algorithms. But the original algorithms can only get the curve of the object. We want to find the key-points. Starting from finding the edge point on the object and extend to get the key-points.

Feature matching is an important issue of object tracking and recognition [13] [14] [19] [20] [22] [28]. We can use the local and global features to set the key-points.

Fig. 3: The original image.

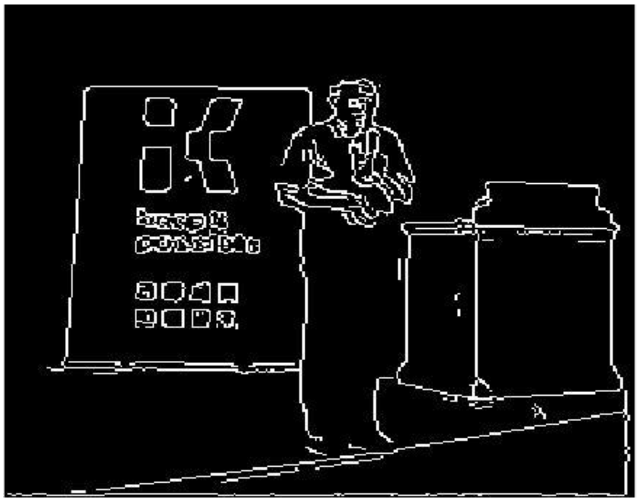

Fig. 4: The result of Canny edge detection.

The conventional way to find the features in an image is to use the edge detection. And then, using the result of edge detection to find the contour or outline from an image. There are several methods to get the edge detection. We want to introduce the relationship between edge detection and SIFT point.

The original image is Fig. 3. The result of Canny edge detection is Fig. 4.

The common way to acquire the result of edge is the Canny edge detection. The following are the steps of Canny edge detection [1] [4] [16].

1: Use the Gaussian function to smooth the image and reduce noise.

2: Compute the magnitude and orientation

$$G(j,k) = \sqrt{G_R^2(j,k) + G_C^2(j,k)} \quad (5)$$

$$\theta(j,k) = \tan^{-1}\left(\frac{G_C^2(j,k)}{G_R^2(j,k)}\right) \quad (6)$$

Fig. 5: The result of Harris detection points.

3: Search the nearest points along the edge normal

$$G_x(j,k) = \begin{cases} G(j,k & \text{if } G(j,k) > G(j',k') \text{and } G(j,k) > G(j'',k'') \\ 0 & \end{cases} \quad (7)$$

4: Label each pixel and set the two thresholds to get the local maximum and candidate pixel [20]

Label each pixels according to two threshold T_H, T_L

$G_N(x,y) \geq T_H$ Edge Pixel

$T_H > G_N(x,y) \geq T_L$ Candidate Pixel

$G_N(x,y) < T_L$ Non-edge Pixel (8)

5: Connect the Labeling points and get the result of edge detection. The result of Harris detection points is Fig. 5.

The advantage of Canny edge detection:

Good Detection: The optimal detector will minimize the probability of false positives as well as false negatives.

Good Localization: The edge detected has to be as close as possible to the true edge.

Single Response Constraint: The detector must return one result only from each edge point.

The Random Sample Consensus (RANSAC), is an algorithm which can exclude the unstable data from the data set we build.

But the original technique of edge detection needs to calculate whole pixels in one image, and this will take much time. Thus, we want to use SIFT points to reduce computation time.

A way to reduce useless SIFT points is the Harris corner detection. We use this algorithm to exclude unwanted key-points. This will help us to reduce many unstable points.

The red points in Fig. 3 are the result of Harris detection. We will take the result to reduce the useless SIFT points. This will make our model become more stable to every condition.


Fig. 6: The result of SIFT points before RANSAC.

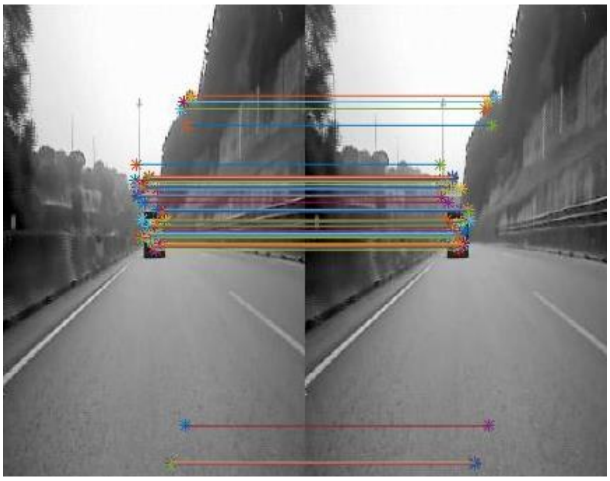
Fig. 7: The result of SIFT points after RANSAC.

The result before we conduct the RANSAC is shown in Fig. 6 and that after we conduct the RANSAC is shown in Fig. 7.

4. OUR PROPOSED MODEL

Based on the original SIFT points, we build a new method to reduce the computation time. After we detect SIFT points, we perform the following procedures which can reduce the unstable points:

1. **Remove the RANSAC-points**: We set up several thresholds to reduce the number of SIFT points. If SIFT points are stable, they will be the corresponding positions between frames. We use it to reduce the computation time of SIFT points and unstable SIFT points.

2. Select the specific regions to detect: In the original model, the algorithm of finding the SIFT points is to detect the whole image. We build a method to select specific regions of image to conduct the SIFT points detection. This will reduce computation time.

Fig. 8: The result of SIFT points.

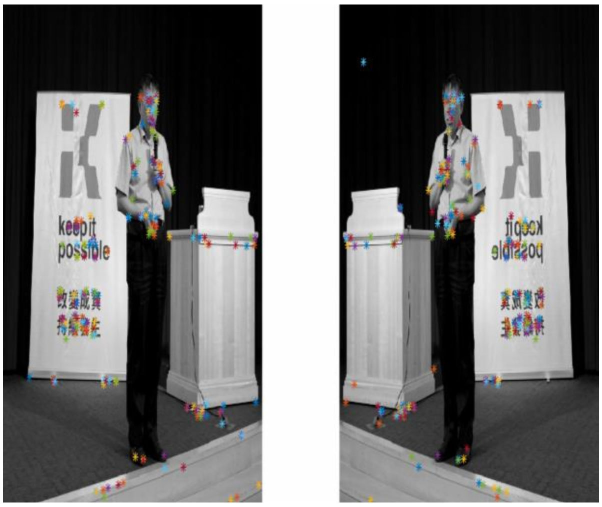

Fig. 9: The result of SIFT points between Original and Horizontal flipping.

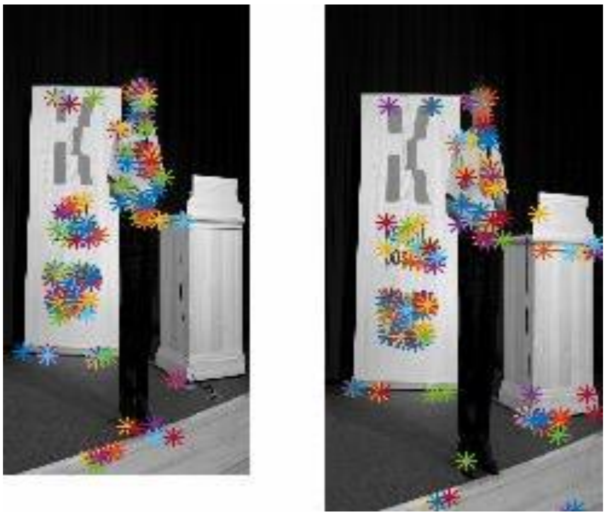

Fig. 10: The result of SIFT points between Original and Scaling.

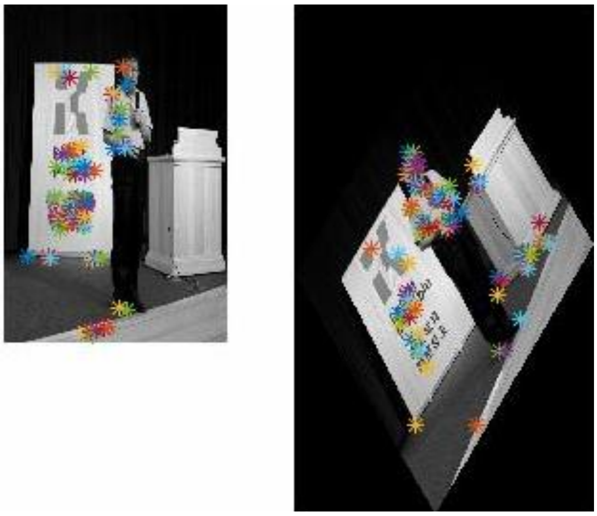

Fig. 11: The result of SIFT points between Original and Rotation.

5. RESULTS

The computation time in SIFT will reduce for real-time application. The new SIFT will reduce computation time and will find more precise key-points for one image. After working with this problem, we can find a more robust algorithm to acquire the feature points.

The new SIFT will reduce computation time and will find more precise key-points for one image [12] [18] [26]. After working with this problem, we can find a more robust algorithm to acquire the feature points.

The following we will conduct different situations. We take the experiments into two different parts and discuss the result of SIFT point detection:

First: Static picture

The image size is: 960*1080 pixels The result of original SIFT point detection is Fig. 8. The horizontal flipping vs. the original SIFT points is Fig. 9.

The Scaling vs. the original SIFT points is Fig. 10. The Rotation vs. the original SIFT points is Fig. 11.

The '*' is the representation of SIFT points. The number of SIFT between Original and Rotation images will be little different. The masks of filtering the image are too small.

We want to expand our model to dynamic images [17] [21] [23] [24] [25] [27]. The result of static images proves that our algorithm runs correctly. In the next section, we will test our model on video system. In order to expand our SIFT points model to be more flexible, we start working on dynamic image model. The dynamic we choose is Video system. After we finish the video system of SIFT points, we will make the algorithm become a real-time work.

Fig. 12: The result of SIFT points.

Fig. 13: The result of SIFT points after adding lines between two frames.

Second: Video
The frame size is: 800*800 pixels
The '*' represents SIFT points.

The result of original SIFT point detection is Fig. 12.
The SIFT point between two frames is Fig. 13.

The '*' is the representation of SIFT points. The lines between SIFT points are the corresponding positions. The corresponding positions are too much now if we want to do the image recognition or tracking. We need to reduce the number of SIFT points.

The number of SIFT points are smaller after we conduct the **selecting the specific region method** to reduce the SIFT points. The SIFT point between two frames is Fig. 14.

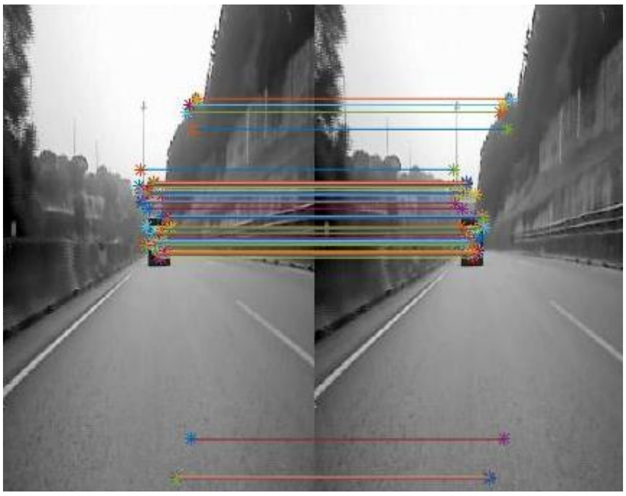

Fig. 14: The result of SIFT points after adding lines between two frames and selecting the specific region.

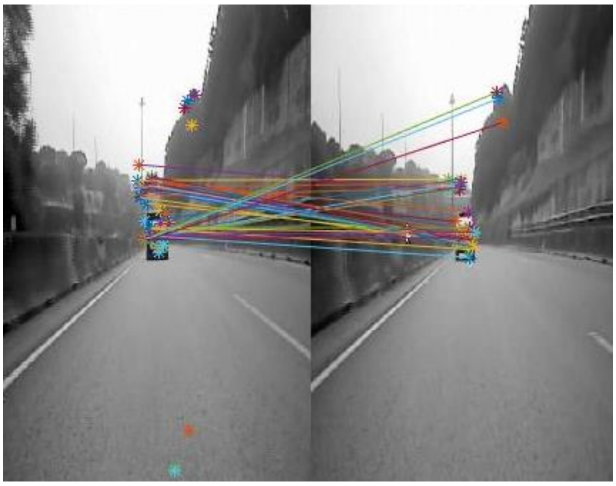

Fig. 15: Before using RANSAC method.

Fig. 16: After using RANSAC method

Fig. 17: Match the SIFT points between frames method

The number of SIFT points are smaller after we conduct the **RANSAC method** to reduce the SIFT points. Before we using the RANSAC is Fig. 15. After we using the RANSAC is Fig. 16.

The number of SIFT points become fewer after we conduct the **Match the SIFT points between frames method** to reduce the SIFT points. The result of using Match the SIFT points between frames method is Fig. 17.

We decide two thresholds to reduce SIFT points. From the part of position, we choose a specific region to find the SIFT points. From the part of pixel value, we delete the instance change of pixel level.

The result of the image has already deleted the unstable points. The lines between two images are to let us know that these points are the same key points. The time gap between two frames is 0.1sec.

We can use the smaller number of SIFT points to conduct image tracking and recognition. We fixed the detection block on specific region to make our detection model become easier to run the SIFT points detection.

The algorithm can track the same SIFT points between two frames and keep track the same object until we do not need it [28]. It is useful for us to conduct the object tracking and also our model can give real-time feedback when it is running. Also, we already reduce the number of SIFT points which can make our algorithm more efficient. We do not need the whole SIFT points, only few numbers. Based on smaller points, we still can conduct the image tracking normally.

In Table 1, we can reduce the number of SIFT points. After we reduce the points, the computation time will become smaller than before. This will help us to improve the original SIFT point algorithm.

Table 1: the number of SIFT points between Original and our proposed model.

(Image size = 800*800)	Number of SIFT points
Original	730 points
Select the specific regions to detect	70-80 points
Remove the RANSAC-points	30-50 points
Match the SIFT points between frames (Proposed)	6-15 points

6. FUTURE AND DISSCUSION

The original model about detecting the SIFT points in one picture will cost almost one minutes. It cannot be used for realistic works.

The number of SIFT points become fewer than before. Our model will reduce the computation time of SIFT points detection. When the computation time is smaller than before, the application of SIFT points will be applicable. We will use our new algorithm to conduct object tracking and recognition. Based on our method, the computation time of SIFT points detection will be real-time.

The machine learning and deep learning are very popular in these days. They also can make our model become more precisely. Our objective in the future is to combine our model with them. After we combine them, we will make our model more flexible.

ACKNOWLEDGE

The authors thank to the support of Alpha Network Inc.

REFERENCES

[1] C. Harris and M. Stephens. "A Combined Corner and Edge Detector," *Proceedings of Alvey Vision Conference*, Manchester, UK, **15**, 1988, pp. 147-151.

[2] U.S. Patent 6,711,293, "Method and apparatus for identifying scale invariant features in an image and use

of same for locating an object in an image", David Lowe's patent for the SIFT algorithm, March 23, 2004.

[3] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Key-points,” Vol. 60, No. 2, 2004, pp. 91-110.

[4] J. Canny, “A Computational Approach to Edge Detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 8, Issue 6, 1986, pp. 679-698.

[5] L. Rabiner ; M. Cheng ; A. Rosenberg ; C. McGonegal “A comparative performance study of several pitch detection algorithms, ” 1976, pp.399-418.

[6] Wenyu Chen ; Yanli Zhao ; Wenzhi Xie ; Nan Sang. "An improved SIFT algorithm for image feature-matching,” 2011.

[7] Huiyu Zhou, YuanYuan, ChunmeiShi "Object tracking using SIFT features and mean shift, ” 2009, pp.345-352.

[8] B. Schunck, B. Horn, Determining optical flow, in: DARPA81, 1981, pp. 144-156.

[9] D. Gavrila, L. Davis, 3D model-based tracking of humans in action: a multi-view approach, in: Proceedings of the Computer Vision and Pattern Recognition, 1996, pp. 73-80.

[10] D. Comaniciu, V. Ramesh, P. Meer, Real-time tracking of non-rigid objects using mean shift, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), South Carolina, 2000, pp. 142–149.

[11] P. Wunsch, G. Hirzinger, Real-time visual tracking of 3D objects with dynamic handling of occlusion, in: Proceedings of 97 International Conference on Robotics and Automation, 1997, pp. 2868–2879.

[12] J. MacCormick, M. Isard, Partitioned sampling, articulated objects, and interface-quality hand tracking, in: Proceedings of the European Conference on Computer Vision, 2000, pp. 390–395.

[13] J. Sullivan, A. Blake, M. Isard, J. MacCormick, Object localization by bayesian correlation, in: Proceedings of the Seventh International Conference on Computer Vision, 1999, pp. 1068–1075.

[14] G. Welch, G. Bishop, Scaat: incremental tracking with incomplete information, in: SIGGRAPH’97: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1997, pp. 333–344.

[15] Dianyuan Han, A standing tree image mosaic algorithm based on SIFT, International conference on Computer science and electronics engineering, Hangzhou, China, 2012, 1:353-356.

[16] C. Guo, X. Li, Linfeng Zhong and Xiang Luo. A Fast and Accurate Corner detection based on Harris Algorithm. Intelligent Information Technology Application 2009:2, 49-52.

[17] X. Dai and S. Khorram. A feature-based image registration algorithm using improved chain-code representation combined with invariant moments. IEEE Transactions on Geoscience and Remote Sensing 1999, 37: 2351-2362.

[18] Richard W. Pazzi, Azzedine Boukerche , Jing Feng and Ying Huang. A Novel Image Mosaicking Technique for Enlarging the Field of View of Images Transmitted over Wireless Image Sensor Networks, Mobile Netw Appl 2010 15:589-606.

[19] Oh-Seol Kwon and Yeong-Ho Ha. Panoramic video using Scale Invariant Feature Transform with embedded color-Invariant values. IEEE transactions on Consumer Electronics 2010, 56:792-798.

[20] Hernâni Gonçalves, José Alberto Gonçalves and Luís Corte-Real, HAIRIS: A Method for Automatic Image Registration Through Histogram-Based Image Segmentation, IEEE Transactions on Image Processing, 2011, 20:776-789.

[21] J. Flusser and T. Suk. Rotation Moment Invariants for Recognition of Symmetric Objects. IEEE Transactions on Image Processing 2006, 15:3784-3790.

[22] Lowe, D.G., 1999. Object recognition from local scale-invariant features. In: Proceedings of the International Conference on Computer Vision, Corfu, Greece, 1999, pp. 1150–1157.

[23] Cui, Y., Pagani, A., & Stricker, D. (2010, September). Sift in perception-based color space. In Image Processing (ICIP), 2010 17th IEEE International Conference on (pp. 3909–3912). IEEE.

[24] J. Matas, O. Chum, M. Urban, T. Pajdla, Robust wide baseline stereo from maximally stable extremal regions, in: Proceedings of BMVC, 2002, pp. 384–393.

[25] M. Pilu, A direct method for stereo correspondence based on singular value decomposition, in: Proceedings of CVPR, Puerto Rico, 1997, pp. 261–266.

[26] R. Zabih, J. Woodfill. Non-parametric local transforms for computing visual correspodence. In Proceedings of ECCV, pages 151–158, 1994.

[27] C. Schmid, A. Zissermann, Automatic line matching across views. in: Proceedings of CVPR, 1997, 666–671.

[28] Xiaoye Lu, R. Manduchi, “Wide-baseline feature matching using the cross-epipolar ordering constraint,” in: Proceedings of CVPR, 2004, pp. 16–23.