KEYPOINT DETECTION BY CASCADED FAST

Takahiro Hasegawa† *Yuji Yamauchi*† *Mitsuru Ambai*‡ *Yuichi Yoshida*‡ *Hironobu Fujiyoshi*†

†Chubu University
{tkhr, yuu}@vision.cs.chubu.ac.jp, hf@cs.chubu.ac.jp
‡Denso IT Laboratory, Inc.
{manbai, yyoshida}@d-itlab.co.jp

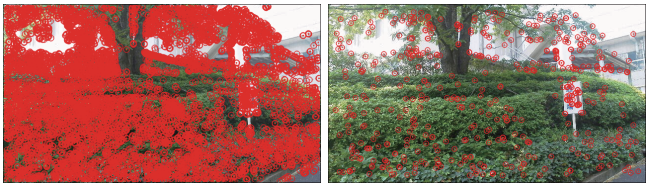
ABSTRACT

When the FAST method for detecting corner features at high speed is applied to images that include complex textures (regions that include foliage, shrubbery, etc.), many corners that are not needed for object recognition are detected because FAST defines corner features on the basis of a 16-pixel bounding circle. To overcome that problem, we propose the Cascaded FAST that defines corners on the basis of similarity in terms of intensity, continuity and orientation in a broader range of areas (20, 16, and 12 pixel bounding circles). Also, cascading three decision trees trained by the FAST approach enables high-speed corner detection in which non-corners are eliminated early in the process. Furthermore, Cascaded FAST determines scale by using an image pyramid and determines orientation at high speed by using a framework for referencing surrounding pixels.

Index Terms— Corner detector, Cascaded FAST, FAST, Keypoint matching

1. INTRODUCTION

Identifying the correspondence of features (keypoints) in different images is an important problem in the image recognition. The many keypoint detection methods that have been proposed to accomplish such detection can be classified into three approaches: corner detection methods (Harris[1], FAST[2]), blob detection methods (DoG[3], Harris-Affine[4]), and region detection methods (MESR[5])[6]. The method we propose here for high-speed keypoint detection is a corner detection method. The importance of the first proposal by Moravec in 1977[7]has resulted in the proposal of many corner detection methods since then[1] [4] [5] [8] [9] [10]. In recent years, the Features from Accelerated Segment Test (FAST) method has been attracting attention for high-speed corner detection [11][2]. FAST uses the relative brightness of the pixel of interest and an area bounded by 15 pixels centered on the pixel of interest to identify corners. High-speed detection of corners is then possible because of the efficiency of looking-up the surrounding pixels by using decision trees trained by machine learning based on this definition. FAST, however, has the problem of detecting a large number of corners in the complex textures of tree foliage and shrubbery, and so on that occur in natural images, such as shown in Fig. 1. That problem arises because corners are determined on the basis of intensity information for an area that is bounded by only 16 pixels. In assigning correspondences between two images on the basis of a large number of corner points, as shown in Fig. 1(a), two problems arise. One is that changes in viewpoint and disturbances such as the movement of leaves by wind in complex areas of natural images cause a change in the appearance of the image so that the same corner features cannot be detected in multiple images. The second problem is that one corner detected in the first image is compared to all of the corners detected in the second image in finding the correspondence of corners in two images. That is computationally expensive when many corners are detected.



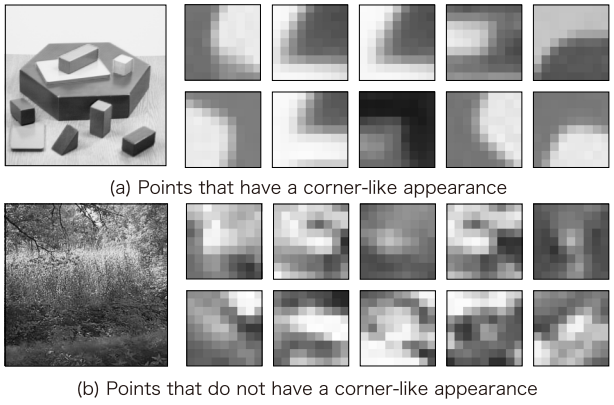
(a) FAST (b) Cascaded FAST

Fig. 1. Comparison of corner detection results for a natural image.

We therefore propose here the Cascaded FAST method for fast detection of only the points that are important in determining correspondences between images. Cascaded FAST uses decision trees that reference a wider range of pixels than does FAST, which uses only an area bounded by 16 pixels. Thus, corners can be detected quickly by cascading the decision trees. Cascaded FAST can suppress the detection of corner point in complex images of nature as shown in Fig. 1(b). In addition, Cascaded FAST determines scale by detecting corners in an image pyramid. Furthermore, the Cascaded FAST framework for referencing peripheral pixels can be used for simultaneous detection of corners and orientation.

2. PROBLEMS AND TENDENCIES WITH FAST

Although FAST can detect corners very quickly, it detects too many corners in areas of complex textures as shown in



(a) Points that have a corner-like appearance

(b) Points that do not have a corner-like appearance

Fig. 2. Appearance of corners detected by FAST.

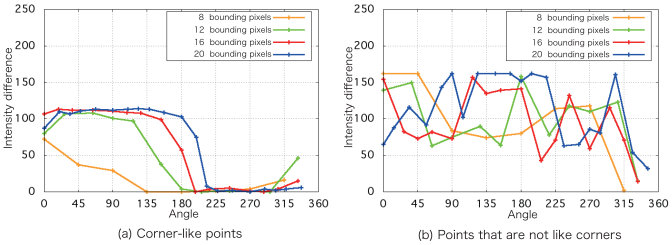


Fig. 3. Results of corner point analysis.

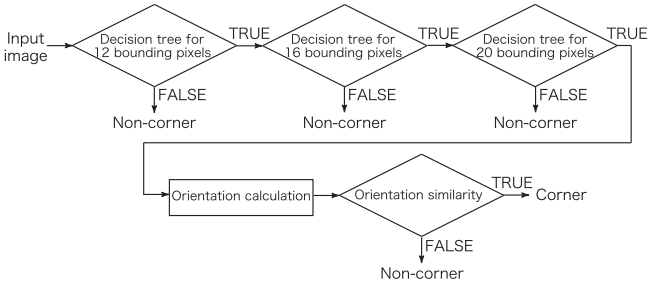


Fig. 4. Corner detection flow in Cascaded FAST.

Fig. 1. We investigated the tendencies in detection of corner-like points and points that are not corner-like by the FAST method. We first examined tendencies in the appearance of the region around corners in image patches that are centered around points that are detected as corners. We compared the appearance of those patch images as being corner-like and non-corner-like. Corner features detected in artificial images and natural images by FAST are shown in Fig. 2. The corner-like points shown in Fig. 2(a) tend to be similar in appearance for all of the patches. The non-corner-like points in Fig. 2(b), on the other hand, tend towards a large variance in appearance. From the tendencies in corner appearance shown in Fig. 2, we can expect the change in brightness of the pixels inside and outside of the 16-pixel bounding circle to change in the same way for corner-like points. We therefore quantitatively analyzed the tendency in the difference in brightness between the pixels in the surrounding region and the pixel of interest. The analysis used average difference values for 1,000 corners that were detected in images that contained only artificial objects and images that contained only natural regions. With n or more consecutive pixels that are all classified as either brighter or darker serving as a reference point, the pixel of interest and the reference point are connected by a straight line and a reference line is drawn (angle 0 degrees). The results of the analysis are presented in Fig. 3, where the vertical axis is the absolute value of the difference between the pixel of interest and a bounding pixel and the horizontal axis is the angle. The large difference values for the bounding pixels of the corner-like points (20, 16, and 12 pixels) are consecutive and the shapes of the graphs are similar. For the non-corner-like points, the difference values are dispersed for 20, 16, 12, and 8 bounding pixels. These results show that defining corners only on the basis of information for 20, 16, and 12 bounding pixels is effective for detecting only the corner-like points.

3. KEYPOINT DETECTION BY CASCADED FAST

The results of the investigation described in section 2 confirm that the changes in difference values for corner-like points are similar for the cases of 20, 16, and 12 bounding pixels. We therefore chose to base corner detection on information for that range of bounding pixels in the approach reported here as shown in Fig. 4. Cascaded FAST also obtains scale and orientation in addition to the corner coordinates, as is described in detail below.

3.1. Definition of corner

In the Cascaded FAST method, corner candidate points are first detected on the basis of conditions of continuous brighter or darker pixels in bounding circles of 20, 16, and 12 pixels. Then, orientation is calculated for the detected corner candidates, and if the orientations for the cases of 20, 16, and 12 bounding pixels are similar, the pixel of interest is detected as a corner. The processing is described in detail below.

Step1 : Continuous brighter or darker pixel condition

The pixels in bounding pixel sets of 20, 16, and 12 pixels are classified according to the ternary values of brighter, similar, and darker by

$$S_{p\to x} = \begin{cases} d, & I_{p\to x} \le I_p - t \quad \text{(darker)} \\ s, & I_p - t < I_{p\to x} < I_p + t \quad \text{(similar)} \\ b, & I_p + t \le I_{p\to x} \quad \text{(brighter)}, \end{cases} \tag{1}$$

where I_p is the brightness value of the pixel of interest, $x \in \{1, \cdots, 16\}$ are the positions of the bounding pixels, $I_{p\to x}$are the brightness values of the 16 bounding pixels, and t is a threshold value. FAST judges the pixel of interest to be a corner in the case that there are nine or more consecutive brighter or darker pixels in a set of 16 bounding pixels. Cascaded FAST, on the other hand, takes the pixel of interest to be a corner candidate point when a bounding pixel set of 20, 16 or 12 pixels has a continuous respective run of at least 11, 9 or 6 brighter or darker pixels. The number of consecutive brighter or darker pixels for sets of 20 or 12 bounding pixels

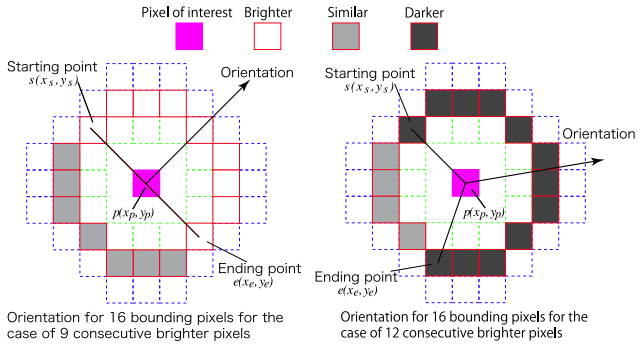


Fig. 5. Example of calculating orientation for 16 bounding pixels.

was determined in proportion to the 16 bounding pixels used by FAST.

Step2 : Calculation of orientation

For each corner candidate point obtained in Step1, the orientations are calculated for the bounding pixel sets of 20, 16 and 12 pixels. An example of calculating the orientation for 16 bounding pixels is presented in Fig. 5. First, the angle from the starting point to the ending point (sequence of all brighter or all darker pixels) is obtained. Denoting the angle of the pixel of interest $p(u_p, v_p)$ and the starting point pixel $x_s(u_s, v_s)$ with respect to the x axis as θ_s, and denoting the angle of the pixel of interest $p(u_p, v_p)$ and the ending point pixel $x_e(u_e, v_e)$ with respect to the x axis as θ_e, the angle of the starting point and ending point, $\theta_{s\to e}$, is obtained by

$$\theta_{s\to e} = \begin{cases} 360 - |\theta_s - \theta_e| & \text{If } \theta_s > \theta_e \\ |\theta_s - \theta_e| & \text{Otherwise,} \end{cases} \quad (2)$$

where $\theta_s = \mathrm{angle}(x_s, p)$ and $\theta_e = \mathrm{angle}(x_e, p)$. In the above equations, $\mathrm{angle}(\cdot)$ is a function that returns the angle of the starting point or ending point relative to the x axis. The orientation θ is then calculated as the direction that divides the angle of the starting point and ending point into two equal parts by

$$\theta = \frac{\theta_{s\to e}}{2} + \theta_s. \quad (3)$$

The orientations for the cases of 20 and 12 bounding pixels are calculated in the same way as in Fig. 5.

Step3 : Orientation similarity condition

Each corner candidate point is judged to be a corner or non-corner according to the similarity of the orientations for the bounding pixel sets of 20, 16 and 12 pixels. Denoting the orientation angle difference for the 16 pixel and 12 pixel sets as α, the orientation angle difference for the 16 pixel and 20 pixel sets is obtained as β. Then, the corner candidate pixel of interest $p(u_p, v_p)$ is detected as a corner by

$$p(u_p, v_p) = \begin{cases} c & \text{If } \alpha \le Th_1 \ \& \ \beta \le Th_2 \\ \bar{c} & \text{Otherwise,} \end{cases} \quad (4)$$

where Th_1 and Th_2 are threshold values. Because the orientations for 20 bounding pixels and 12 pixel bounding pixels have different resolutions, there are different threshold values for α and β.

3.2. Faster processing through cascading

For Cascaded FAST, the training sample is first divided into corner and non-corner features as described in section 3.1. The training sample is then used to train three decision trees by the ID3[12] algorithm with reference to bounding pixel sets of 20, 16 and 12 pixels. In Cascaded FAST, when all three of the decision trees decide that an input pixel is a corner, the pixel is output as a corner candidate point. If even one decision tree decides the point is a non-corner, the non-corner result is output. To expedite that processing, the decision trees are placed in a cascading arrangement (Fig. 4) so that non-corner pixels are rejected earlier in the process and corner candidate points are detected faster. Next, orientation is calculated for the candidate points as described in Step 2, and the final judgement is made on the basis of similarity as described in Step 3. The order of the decision trees for the bounding pixel sets of 20, 16 and 12 pixels does not affect the corner detection results at all, but preliminary experiments show that the order does affect detection time. As shown in Fig. 4, the decision tree order for referencing the bounding pixel sets in the order of 12, 16 and 20 pixels was confirmed to produce the fastest corner detection.

3.3. Obtaining orientation and scale

The corner detection methods described so far output only the corner point coordinates, but the proposed Cascaded FAST method can output scale and orientation in addition to corner point coordinates. Concerning scale, Cascaded FAST is applied to an image pyramid of multiple resolutions and the scale of the image resolution for which corner points were detected is used. If a corner is detected in same position of multiple scale, they are detected as different keypoints. For orientation, the value calculated according to Step 2 of section 3.1 is output.

The keypoints detected by Cascaded FAST are shown in Fig. 6. The centers of the red circles are the corner coordinates, the size of the circle indicates the scale, and the blue line indicates orientation. We can see from Fig. 6 that the orientations of the detected corners are rotated by the same number of degrees by which the image is rotated.

4. EVALUATION EXPERIMENTS

We performed evaluation experiments to test the effectiveness of Cascaded FAST. The hardware used for these experiments is a personal computer equipped with a 3.33-GHz Intel(R) X5470 CPU and 32 GB of memory.

4.1. Evaluation of corner detection

The corner detection results for each method are presented in Fig. 7. FAST detects a large number of corners in natural

Fig. 6. Examples of detected keypoints by Cascaded FAST.

regions, but Cascaded FAST detects a much smaller number. Also, we compared the Harris method, FAST, and Cascaded FAST regarding the time required for corner detection. The processing speeds are presented in Table 1. The processing time is 2.9 [ms] longer for Cascaded FAST than for FAST, but the speed is about 11 times as fast as with the Harris method. Also, Cascaded FAST is capable of operating at 135 [fps].

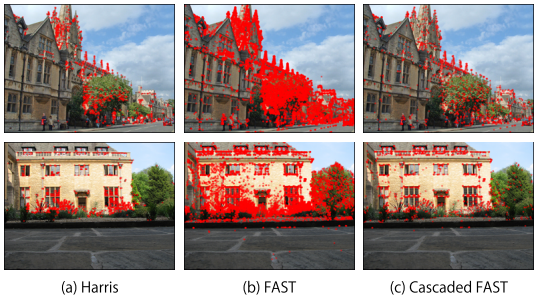


Fig. 7. Corner detection results for various methods.

Table 1. Comparison of detection times for various methods.

	Harris	FAST	Cascaded FAST
Detection time[ms]	81.1	4.5	7.4
Number of corners	3549	15913	2369

4.2. Evaluation of determining the correspondences in two images

Here, we evaluate the performance and processing time for matching keypoints in two images. The results for three methods, Harris, FAST, and Cascaded FAST, are compared. An image pyramid was used for the calculation of scale for all methods. The orientation calculations are performed by two methods, one based on the moment within a patch that is centered on a corner point [13] and the other is the one used by Cascaded FAST. The feature description is ORB, which is normalized to changes in rotation and scale [13]. ORB features represent the brightness relationship within a patch as a binary feature. In this experiment, we used 110 images (1024 $\times$ 768 pixels) applied by affine transforms.

For evaluation of keypoint correspondence, we used the matching rate calculated as the number of correct correspondences divided by the total number of correspondences. Next, we compare the matching rate and the frame rate (fps). The matching performance and speed results are presented in Fig. 8. The matching results are about the same for Cascaded FAST and the other methods, but Cascaded FAST has a higher frame rate. The computation times for the various processes are shown in Fig. 9. For the Harris method, the corner detection processing time accounts for a very high proportion of the total. FAST, on the other hand, does corner detection and orientation calculation at high speed, but requires a high proportion of processing time for feature description and distance calculation. The reason for that is the very large number of corners detected by FAST compared to the other methods. Cascaded FAST is faster than the other two methods for each type of processing. In particular, the orientation calculation in which the data for the 20 pixel bounded region is used can be eliminated, because the orientation values calculated for corner detection can be reused. Cascaded FAST can do image correspondence in about 43.7 [ms] when orientation is calculated by using the data for 20 bounding pixels.

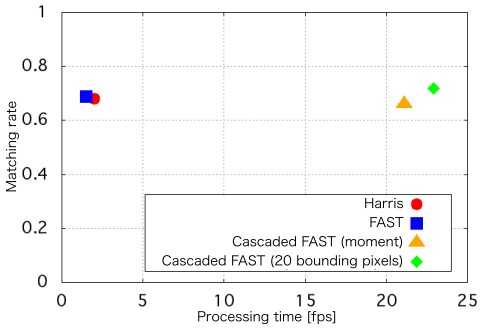


Fig. 8. Matching performance and speed.

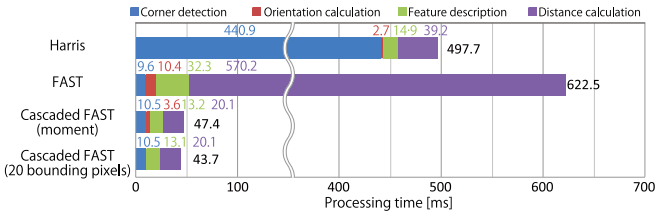


Fig. 9. Computation time for keypoint matching.

5. CONCLUSION

We have proposed Cascaded FAST as a method for suppressing unneeded points when identifying correspondences in natural regions between two images. Cascaded FAST suppresses corner detection by defining corner features in terms of continuous brightness values and orientation similarity in a wider range of bounding pixel sets that includes 20, 16 and 12 pixels. Cascaded FAST also achieves high-speed corner detection by cascading three decision trees trained by the FAST approach to eliminate non-corner candidates early in the process. Furthermore, this method uses image pyramids to determine scale. It can also determine orientation at high speed by using a framework for referencing the pixels in the bounding circle of a feature region and effectively match keypoints. Issues for future work include efficient acquisition of scale from an image pyramid.

6. REFERENCES

[1] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey Vision Conference*, 1988, pp. 147–151.

[2] E. Rosten, R. Porter, and T. Drummond, "FASTER and better: A machine learning approach to corner detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, pp. 105–119, 2010.

[3] David G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[4] Mikolajczyk and Schmid, "Scale & Affine Invariant Interest Point Detectors," *International Journal of Computer Vision*, vol. 60, no. 1, pp. 63–86, 2004.

[5] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust Wide Baseline Stereo from Maximally Stable Extremal Regions," in *British Machine Vision Conference*, 2002, pp. 36.1–36.10.

[6] Tinne Tuytelaars and Krystian Mikolajczyk, *Local Invariant Feature Detectors: A Survey*, Now Publishers Inc., Hanover, MA, USA, 2008.

[7] H. Moravec, "Towards Automatic Visual Obstacle Avoidance," in *International Joint Conference on Artificial Intelligence*, 1977, p. 584.

[8] J. Shi and C. Tomasi, "Good Features to Track," in *Conference on Computer Vision and Pattern Recognition*.

[9] S. M. Smith and J. M. Brady, "Susan—a new approach to low level image processing," *International Journal of Comput. Vision*, vol. 23, no. 1, pp. 45–78, 1997.

[10] J. P. Gravel, "Corner Detection," *Biological Cybernetic*, vol. 59, no. 4, pp. 139 – 153, 1988.

[11] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European Conference on Computer Vision*, 2006, pp. 430 – 443.

[12] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[13] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An Efficient Alternative to SIFT or SURF," in *International Conference on Computer Vision*, 2011, pp. 2564–2571.