Pixel-parallel feature detection on vision chip

Jie Yang, Cong Shi, Liyuan Liu, Jian Liu and Nanjian Wu

Pixel-parallel FAST and SIFT feature detection algorithms are proposed, which are suitable for the vision chip architecture. The proposed algorithms can execute the FAST and SIFT feature detections on the vision chip in a pixel-parallel manner. The Feature detection utilising the proposed algorithms can be realised by basic logic and arithmetic operations. Experiments and detailed analysis show that the proposed algorithms can perform feature detection on the vision chip at a higher speed compared with that on a general processor or an application-specific circuit.

Introduction: Feature detection in an image sequence is a prerequisite step for many image processing and computer vision algorithms such as object tracking and recognition. It labels local regions of a 'feature' of an image so that local appearance properties can be further exploited. The speed of feature detection is a key requirement for real-time or high-speed realisation of many algorithms. However, the performance of central processing unit (CPU)-based feature detection is inadequate to achieve high-speed processing. Recently, a massively parallel SOC called the vision chip [1–4] has been widely used in the field of high-speed image processing. It adopts a parallel SIMD processing elements (PEs) array to finish image processing algorithms in a pixel-parallel manner. However, current feature detection algorithms cannot be directly implemented on vision chips. In this Letter, we modify FAST and SIFT feature detections in a pixel-parallel way so that they can be implemented on a vision chip. Experiments and performance analysis have been carried out. The results indicate that our proposed methods can achieve higher performance than previous works.

Proposed FAST feature detection: The FAST [5] detector operates on a discretised circle around a candidate point $\boldsymbol{p}$, as shown in Fig. 1*a*. The circle usually contains 16 pixels. $\boldsymbol{p}$ is classified as a feature pixel if there exists a contiguous arc of at least $\boldsymbol{n}$ pixels on the circle that are all brighter or all darker than $\boldsymbol{p}$ by a threshold $\boldsymbol{t}$. The original FAST detector adopts a decision tree to improve performance on a CPU-based system. However, this method is not suitable for the vision chip with the pixel-parallel architecture, which is significantly different from the general-purpose CPU. Our proposed FAST-9 (FAST with $\boldsymbol{n}=9$) detector on the vision chip is shown in Fig. 1*b*.

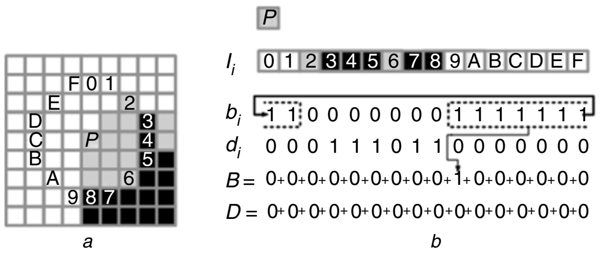

Fig. 1 *FAST feature detector and proposed FAST-9 feature detection procedure on vision chip*

a FAST feature detector
b Proposed FAST-9 feature detection procedure on vision chip

First, the grey value I_i of the *i*th pixel, that on the circle, is compared with the centre pixel value I_p with a threshold *t* as the following equation

$$b_i = s\big((I_p + t) - I_i\big), \quad d_i = s\big(I_i - (I_p - t)\big)$$

where

$$s(x) = \begin{cases} 1, & x < 0 \\ 0, & x \geq 0 \end{cases} \tag{1}$$

The addition and subtraction operations in (1) can be easily accomplished by cascading PE operations [2]; the b_i and d_i values are the sign bit of the subtraction results. The 1 bit variables b_i and d_i indicate whether the *i*th pixel on the circle is brighter or darker than the centre pixel $\boldsymbol{p}$ by the threshold $\boldsymbol{t}$, respectively. Equation (2) is used to test whether nine continuous pixels are all brighter or all darker than the centre pixel. Since b_i and d_i are 1 bit variables, the multiplication operation in (2) is substituted with 'and' operations. As shown in Fig. 1*b*, the nine consecutive b_i in the dashed rectangle yield a '1' with eight 'and' operations. If B or D is non-zero, the centre pixel can be classified as a feature point

$$B = \sum_{i=0}^{15} \prod_{j=0}^{8} U(b_{i,j}), \quad D = \sum_{i=0}^{15} \prod_{j=0}^{8} U(d_{i,j})$$

where

$$U(x_{i,j}) = \begin{cases} x_{i+j-16}, & \text{if } i+j > 15 \\ x_{i+j}, & \text{otherwise} \end{cases} \tag{2}$$

The feature point response function proposed in [5] can also be implemented as shown in the following equation

$$V = \max(V_b, V_d)$$

where

$$V_b = \sum_{i=0}^{15} b_i * \big(|I_i - I_p| - t\big), \quad V_d = \sum_{i=0}^{15} d_i * \big(|I_p - I_i| - t\big) \tag{3}$$

The multiplication operations in (3) are again substituted by 'and' operations. Lastly, non-maximal suppression is carried out to remove the feature point, which has an adjacent feature with higher V.

The above FAST implementation requires only simple logic and arithmetic operations and no flow control is needed. It is suitable for SIMD vision chip realisation. Since all PEs execute the same operations at the same time, the above operations only need to be carried out once to finish the FAST feature detection.

Proposed SIFT feature detection: SIFT [6] is widely accepted as one of the most popular and highest quality feature detectors currently available. It selects the local extrema of an image filtered with difference of Gaussians. The key step of SIFT feature detection is to obtain a Gaussian filtered image with different coefficients. As shown in Fig. 2*a*, the Gaussian images in the proposed method are generated by filtering the original image with a primary Gaussian filter of coefficient δ for 16 times. However, in the original SIFT algorithm, the Gaussian images are obtained by complex arithmetic calculation. The proposed method is based on the Gaussian cascading property [7], which indicates that filtering an image with Gaussian coefficient δ for n times is equal to filtering the image with Gaussian coefficient $\sqrt{n}\delta$. Therefore, the five scales in an octave of the SIFT algorithm are acquired by taking the 1, 2, 4, 8 and 16 times filtered image. Gaussian filtering can be finished with a vision chip in a pixel-parallel way. This ensures high performance of the proposed procedure. Furthermore, as shown in Fig. 2*b*, if a Gaussian filter is separable, then the kernel (first row) of the filter can be used to filter the image both vertically and horizontally. The resultant image is equal to the Gaussian filtered image. Separating the two-dimensional (2D) Gaussian filter into two 1D kernels helps to reduce the computation overhead of calculating the Gaussian image and further improves performance. Three kernels corresponding to Gaussian filters with different σ values are carefully selected. The divided 8, 16 and 32 operations of the kernels are completed by ignoring the last 3, 4 and 5 bits of the results, respectively. The remaining procedure of the SIFT algorithm, namely, the difference of Gaussian images and non-maximal suppression can be easily carried out in a PE array of the vision chip.

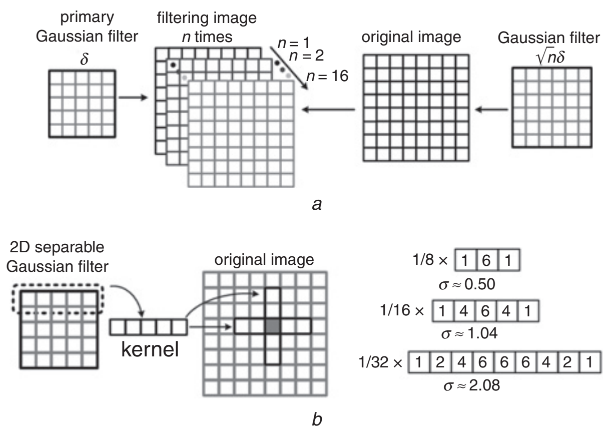

Fig. 2 *Procedure of generating Gaussian images on vision chip*

a Generate different Gaussian images with primary Gaussian filter
b Use kernels to implement primary Gaussian filters

Experimental results and evaluations: The proposed feature detectors are implemented on a field programmable gate array (FPGA). The FPGA runs at 50 MHz and a vision chip architecture that contains a 128 × 64 PE array is implemented. The PE array can carry out 'bit-and', 'bit-inversion', 'bit-or' and 'bit-addition' operations. Fig. 3 shows the operation breakdown of the proposed FAST and SIFT feature detections on the vision chip. The major parts of operations are 'bit-addition' and 'bit-and'. Since every kernel of the proposed SIFT algorithm has to run for many times, it takes many more operations than FAST. The results of performing FAST-9 and SIFT on the FPGA are shown in Fig. 4. The FAST-9 detection result is almost identical with its MATLAB counterpart shown in Fig. 4*d*, since they are mathematically equivalent. The SIFT feature detected on the FPGA is slightly different from its personal computer counterpart due to the integer approximation of Gaussian filters. However, most of the feature points are detected and our experiments indicate that these differences with the original SIFT algorithm do not effect the results of the final recognition process.

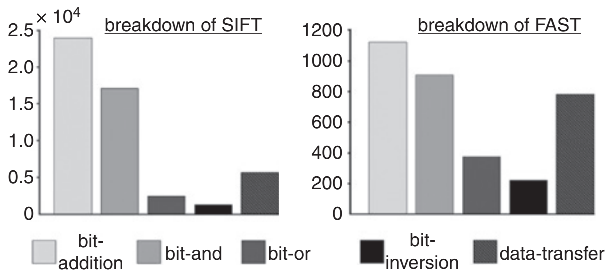

Fig. 3 *Operation breakdown of proposed feature detections*

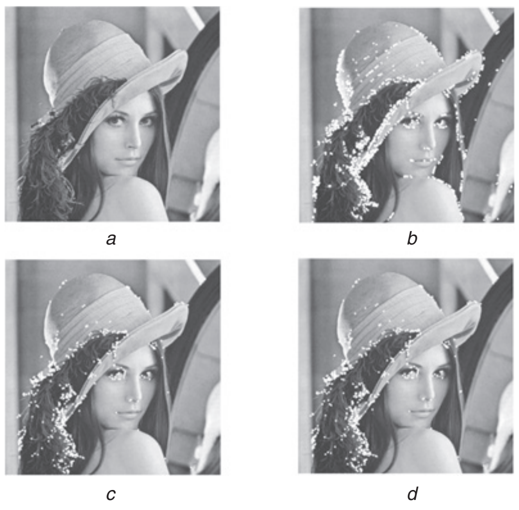

Fig. 4 *Example and comparison of feature detection results*

a Original image
b SIFT feature detected by FPGA
c FAST-9 feature detected by FPGA
d FAST-9 feature detected by MATLAB

Table 1: Comparison with other works

	This work	Moko *et al.* [8]	Clemons *et al.* [9]	Dohi *et al.* [10]
Algorithm	FAST SIFT	FAST	FAST SIFT	FAST
Implementation	FPGA	Embedded processor	Multicore processor	FPGA
Frequency	50 MHz	168 MHz	1 GHz	25 MHz
Resolution	512 × 512	512 × 512	1024 × 768	640 × 480
Processing speed (fps)	FAST: 460 SIFT: 30	27.3	~40	62.5

Table 1 shows a comparison with previously reported hardware implementations for feature detectors. Compared with [8, 10], our proposed FAST feature detection can process an image with approximately the same resolution and with much higher performance. When compared with [9], our experimented image resolution is three times smaller, but the processing speed of the FAST algorithm is more than 10 times faster. The SIFT implementation of [9] achieves a higher performance than our work; however, it runs at a twenty times faster clock frequency and its hardware is much more complex than ours.

Conclusion: We have introduced a pixel-parallel feature detection implementation for vision chips. The proposed feature detection can be finished on the vision chip with basic logic and arithmetic operations. Experimental results show that the pixel-parallel feature detection on the vision chip greatly improves feature detection performance.

Acknowledgments: This work was supported by the National Natural Science Foundation of China (nos. 60976023 and 61234003) and the Special Funds for the Major State Basic Research Project of China (no. 2011CB93-2902).

15 May 2014
doi: 10.1049/el.2014.1787

Jie Yang, Cong Shi, Liyuan Liu, Jian Liu and Nanjian Wu (*Institute of Semiconductors, Chinese Academy of Science, No. 35 QinHua East Road, Beijing 100083, People's Republic of China*)

E-mail: nanjian@red.semi.ac.cn

References

1 Lopich, A., and Dudek, P.: 'A SIMD cellular processor array vision chip with asynchronous processing capabilities', *IEEE Trans. Circuits Syst. I, Regul. Pap.*, 2011, **58**, (10), pp. 2420–2431, doi: 10.1109/TCSI.2011.2131370
2 Zhang, W., Fu, Q., and Wu, N.-J.: 'A programmable vision chip based on multiple levels of parallel processors', *IEEE J. Solid-State Circuits*, 2011, **46**, (9), pp. 2132–2147, doi: 10.1109/JSSC.2011.2158024
3 Shi, C., Yang, J., Han, Y., Cao, Z., Qin, Q., Liu, L., Wu, N.-J., and Wang, Z.: 'A 1000 fps vision chip based on a dynamically reconfigurable hybrid architecture comprising a PE array and self-organizing map neural network'. Int. Solid-State Circuits Conf., San Francisco, CA, USA, February 2014, pp. 128–129, doi: 10.1109/ISSCC.2014.6757367
4 Yang, J., Shi, C., Liu, L., and Wu, N.: 'Heterogeneous vision chip and LBP-based algorithm for high-speed tracking', *Electron. Lett.*, 2014, **50**, (6), pp. 438–439, doi: 10.1049/el.2014.0033
5 Rosten, E., and Drummond, T.: 'Machine learning for high-speed corner detection'. European Conf. Computer Vision, Graz, Austria, May 2006, pp. 430–443, doi: 10.1007/1174402334
6 Lowe, D.G.: 'Distinctive image features from scale-invariant key points', *Int. J. Comput. Vis.*, 2004, **60**, (2), pp. 91–110, doi: 10.102–3/B:VISI.0000029664.99615.94
7 Jain, R., Kasturi, R., and Schunck, B.G.: 'Machine vision' (McGraw-Hill, New York, 1995)
8 Moko, Y., Watanabe, Y., Komuro, T., Ishikawa, M., Nakajima, M., and Arimoto, K.: 'Implementation and evaluation of fast corner detection on the massively parallel embedded processor MX-G'. Computer Vision and Pattern Recognition Workshops, Colorado Springs, CO, USA, June 2011, pp. 157–162, doi: 10.1109/CVP-RW.2011.5981839
9 Clemons, J., Jones, A., Perricone, R., Savarese, S., and Austin, T.: 'EFFEX: an embedded processor for computer vision based feature extraction'. Design Automation Conf., New York, USA, June 2011, pp. 1020–1025
10 Dohi, K., Yorita, Y., Shibata, Y., and Oguri, K.: 'Pattern compression of fast corner detection for efficient hardware implementation'. Int. Conf. Field Programmable Logic and Applications, Chania, Greece, September 2011, pp. 478–481 doi: 10.1109/FPL.2011.94