IJARST
International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

A 135-frames/s 1080p 87.5-mWBinary-Descriptor-Based Image Feature Extraction Accelerator

[1]P.SHANKAR, M.Tech Assistant Professor, shankardvk48@gmail.com
[2]NENAVATH DASHARATH ,M.Tech Associate Professor , ndasharath86@gmail.com
Department-ECE
Pallavi Engineering College Hyderabad, Telangana 501505.

Abstract— **Binary picture descriptors, which derive image function description from the neighborhood photograph patches at once, are widely followed in the cellular and embedded applications dueto lower computational complexity and memory requirement. With the purpose of improving the computation efficiency with out degrading reputation overall performance, a light-weight binary robust descriptor is proposed based totally on the evaluation of the country-of-the artwork binary descriptors in this paper. A directional aspect detection and optimized keypoint score characteristic are developed to refine the keypoints. similarly, rotation invariance is accomplished by executing circular symmetric-based descriptor era and a rough-grained orientation calculation technique simultaneously. Them experimental results reveal that the proposed keypoint detector and binary descriptor reap more than instances speedup and at least 23.6% development in processing pace with similar performance, respectively. moreover, a completely huge scale integration structure is likewise designed based on in-depth exploration of bit-level and task-degree parallelism. based totally at the postlayout simulation in a TSMC sixty five-nm CMOS process, the accelerator can obtain one hundred thirty five frames/s on 1080p photo at the same time as best eating 87.5 mW at a 200 MHz running frequency.**

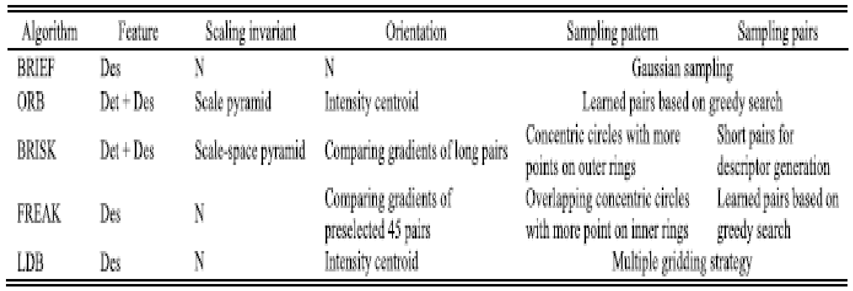
Index Terms—**Binary descriptor, feature extraction, key point detector, rotation invariance (RI).**

I. INTRODUCTION

With the deployment of high-quality image sensor on the embedded platforms (wearable devices, smart mobile devices, mobile robots, unmanned aerial vehicles, etc.), a set of visual applications, such as face recognition, augmented reality, visual tracking, and 3D reconstruction, is realized on these platforms. As one of the most computational intensive functions to support these visual applications, feature extraction, including keypoint detection and descriptor generation, aims to obtain distinctive and robust image representations. The resulted representations are used for further matching across different images, which suffer from certain variations in scale, brightness, viewpoints, and rotation. Traditional histogram-of-gradients-based descriptors, such as scale-invariant feature transform (SIFT) [1] and speeded up robust features (SURF) [2], can provide the best recognition performance while keeping invariant to scaling, rotation, andillumination problems. However, they are too computationally expensive and pose a huge burden to memory access and storage. Although several methods, such as integral image, dimension reduction, and locality sensitive hashing, are applied in the later variants, such as principle component analysis SIFT [3] and gradient location and orientation histogram [4], these algorithms are

difficult to achieve real-time processing on the embedded platforms, which are characterized as relatively weaker processing capability and lower memory capacity. In the last decades, several hardware solutions based on these algorithms are proposed to meet the real-time processing requirement. Although some works [5]–[7] achieve real-time processing, they mostly focus on images with low resolutions, such as video graphics array (VGA) (640 × 480), 512 × 512, or even lower. With the increasing demand of higher resolution image from visual applications, faster feature extraction accelerators with lower memory load are required. In recent years, binary descriptors have been deployed on the embedded platforms successfully. These binary descriptors are more computation efficient since these descriptors are directly computed from the local image patches by pairwise intensity comparison. Meanwhile, these binary strings require less memory storage and support quicker matching by calculating
the hamming distance through XOR and bit count operation. They have been widely applied in various embedded visual applications, such as 3D reconstruction [8], mobile visual search [9], mobile augmented reality [10], and so on. With higher quality images deployed on embedded devices and higher speed processing required from emerging visual applications, feature extraction accelerators based on these binary descriptors are required.In this paper, a hardware-oriented optimized algorithm, named lightweight binary robust descriptor (L-BIRD), is proposed to enable high-speed local feature extraction on embedded devices. With higher priority posed on the processing speed improvement, several bottlenecks are first located based on the complete analysis of the binary feature extraction algorithms. Then these computationally expensive components are optimized with other approaches under lower hardware budget. Compared with the state-of-the-art binary descriptors, the L-BIRD algorithm achieves at least 23.6% speedup, while keeping enough robustness to various image deformations. Furthermore, an L-BIRD-based very large scale

COMPARISON OF THE CURRENT BINARY DESCRIPTORS



Algorithm	Feature	Scaling invariant	Orientation	Sampling pattern	Sampling pairs
BRIEF	Des	N	N	Gaussian sampling	
ORB	Det + Des	Scale pyramid	Intensity centroid	Learned pairs based on greedy search	
BRISK	Det + Des	Scale-space pyramid	Comparing gradients of long pairs	Concentric circles with more points on outer rings	Short pairs for descriptor generation
FREAK	Des	N	Comparing gradients of preselected 45 pairs	Overlapping concentric circles with more point on inner rings	Learned pairs based on greedy search
LDB	Des	N	Intensity centroid	Multiple gridding strategy	

Note: Det: Detector, Des: Descriptor.

TABLE I

integration (VLSI) accelerator is also proposed via exploiting the bit-level and task-level parallelism. It achieves at least 135 frames/s for full high definition (HD) image's binary feature extraction, while only consuming 87.5 mW at 200 MHz. It occupies 5.76 mm2 in a 65-nm CMOS process with an equivalent gate count of 127k and 205-kB on-chip SRAM. In general, the contributions of this paper can be summarized as follows. 1) A directional pairwise edge detection approach and a more distinctive keypoint score function are added to features from accelerated segment test (FAST) to detect the keypoints with higher quality in a more computationally efficient manner. 2) A binary descriptor, which executes the circularsymmetric sampling-pattern-based descriptor generation and a coarse-grained orientation calculation in parallel rather than sequentially, is proposed to achieve rotation invariance (RI). 3) Based on the hardware-friendly optimization on the feature extraction algorithm, bit-level and task-level parallelisms are further exploited in the VLSI design process. To the best of our knowledge, this is the first binary feature-based hardware solution, which considers both processing speed and recognition performance under various image deformations. The rest of this paper is organized as follows. Section II reviews the related algorithms. Section III presents the motivations of this paper. The hardware-oriented optimization and performance evaluation are then presented in Section IV. The hardware architecture details are introduced in Section V. Section VI concludes this paper.

II. RELATED WORK

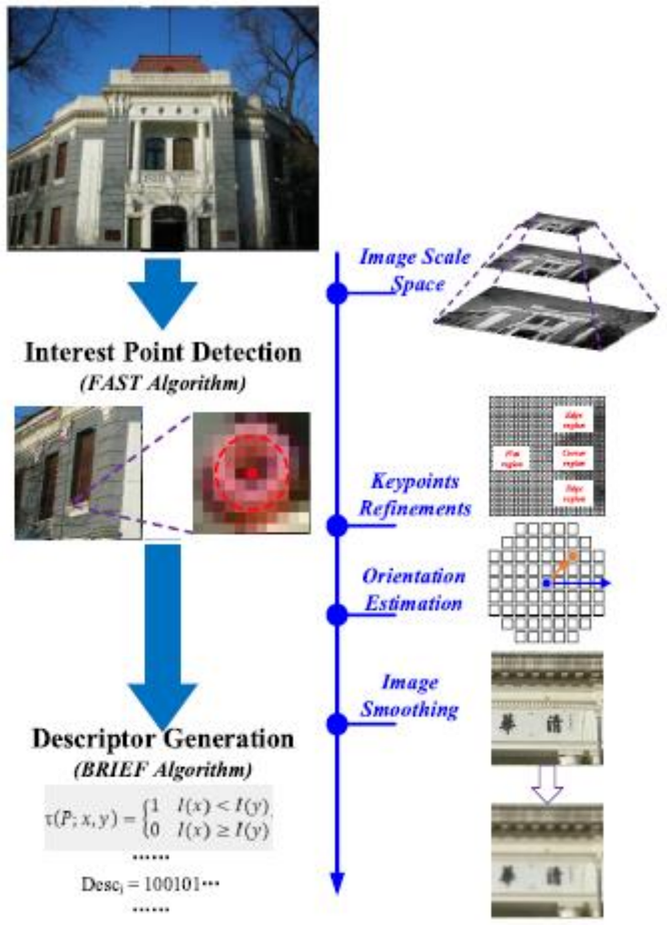
In this section, current related algorithms, including interest point detectors and binary image descriptors, are reviewed.

A. *Interest Point Detectors*

Lots of methods were proposed to implement interest point detection in the last several decades. There are some wellknown algorithms for corner detection, such as small univalue segment assimilating nucleus [11] and Harris corner detector [12], but they are too computationally expensive for embedded platforms, which suffer from low processing capability and limited memory capacity. The FAST algorithm acts as the most computationally efficient corner detection algorithm, which is always combined with binary robust independent elementary features (BRIEF) to support high-throughput real-time visual applications. An adaptive and generic accelerated segment test (AGAST) [13] is then proposed to be more generic and adaptive by deriving the optimal decision tree during segment test stage. The oriented FAST (oFAST) is proposed to achieve RI as well as keypoint refinement in the oFAST and rotated BRIEF (ORB). Scale-space pyramid is enabled in the binary robust invariant scalable keypoints (BRISK) algorithm's keypoint detector to achieve scale invariance based on the AGAST algorithm.

B. *Binary Descriptors*

A range of lightweight binary descriptors were proposed as the proliferation of the mobile-platform-based visual applications. ORB [14], BRISK [15], BRIEF [16], local difference binary (LDB) [17], and fast retina keypoint (FREAK) [18] all belong to binary descriptors. The BRIEF descriptor, which directly calculates binary strings via simple pairwise pixel intensity comparison, acts as the common principle followed by the later binary descriptors. However, BRIEF is sensitive to image scaling and rotation. To overcome these limitations, the later binary descriptors are further optimized by the different choice of orientation calculation method, sampling pattern, and sampling pairs for comparison (as shown in Table I). The oFAST, which incorporates an orientation estimation module and a scale pyramid, is proposed in ORB algorithm to make it rotation invariant and robust to scaling change. Besides, rather than randomly selecting comparison pairs in BRIEF, ORB utilizes a learning method to select highly variant and lower correlated binary tests to obtain better performance. BRISK constructs a scale-space pyramid with multioctaves and multiintra octaves to detect keypoints. A concentric circle-based sampling pattern is adopted to sample 60 points, which are equally located on the keypoint-centered circles. The sampling pairs of these points are divided into two subsets according to the threshold distance. The long-distance pairs are used to estimate the patch orientation, while the short-distance pairs are used to calculate the binary descriptor. Alahi *et al.* [18] further enhanced BRISK by leveraging a sampling strategy that resembles the retinal ganglion cells distribution. Inspired by the human visual system, FREAK also uses a circularsymmetric sampling pattern, which poses higher priority on inner points. To further improve the distinctiveness of binary descriptors, LDB computes the binary string not only using simple intensity comparison but also taking the first-order gradient difference into consideration



These binary descriptors are seeing their roles in fitting the goals of feature extraction on the embedded platforms. For example, BRIEF is deployed on the embedded system on chip in [19], and the processing speed on 1280 × 720 image achieves 60 frames/s. 94.3 frames/s in 1080p full HD resolution at a 200-MHz operating frequency is achieved in [20] by implementing a heterogeneous many-core system based on FAST [21] and BRIEF. Although faster processing is enabled on these platforms, they are implemented at the cost of lower recognition performance under image distortions (such as image rotation and scaling). In addition, they cannot meet the requirement of higher processing speed on higher quality images from emerging applications.

III. MOTIVATIONS

Current binary descriptors are always combined with FAST-like detectors to implement binary feature extraction collaboratively. As shown in Fig. 1, the FAST and BRIEF algorithms are the primary components for keypoint detection and descriptor generation, respectively. To make this combination more robust and reliable, additional functions are brought in to make it robust enough to various image distortions. First, an image scale space is required to support image matching across images with different resolutions. Second, a large number of keypoints would be detected according to the segment test of the FAST, which would result in unaffordable memory space requirement and decreased descriptor distinctiveness; therefore, additional function should be used to refine the

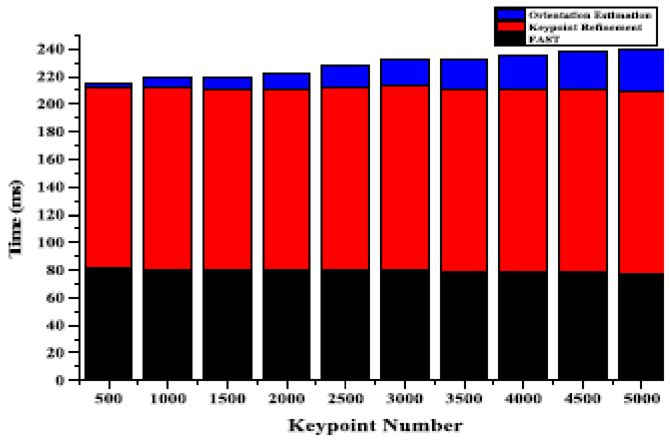


Fig. 2. O-FAST run-time distribution.

best corners with higher quality. Besides, the orientation ofthe image patch should be calculated to avoid performance degradation under image rotation. Finally, the single pixel, which is involved in the pairwise intensity comparison, is noise sensitive. Therefore, certain image smoothing method must be applied to make the resulted descriptor resistant to noise. Recently, several works [22]–[24] have been presented to evaluate the performance of the state-of-the-art keypoint detectors and binary descriptors. The experimental results demonstrate that binary descriptors achieve several times speedup in descriptor generation and matching compared with nonbinary descriptors, while providing comparable precision. Besides, the test results show that the performance varies with the different choice of keypoint detectors. Since the oFAST algorithm acts the counterpart detector of the proposed detector, the keypoints of the binary descriptors are all provided by the oFAST algorithm. In this work, to further analyze the bottleneck and exploit the potential optimization space, the runtime analysis of the oFAST detector and latest binary descriptors are conducted on a desktop with an Intel E7200 2.53-GHz processor, 4 GB of RAM, and

Microsoft Windows 7. LDB is the original implementation from the authors, while the others are all provided by the OpenCV 2.4.9 package. This experiment environment is also used in the later part of this paper. Simulations are executed on the Oxford dataset (resolution: 1000×700) to get the detailed time breakdown for each module. In the oFAST keypoint detector, two additional functions, orientation estimation and keypoint refinement, are enabled. A three-layer image pyramid is built in the simulation, while the threshold is set as 20. As shown in Fig. 2, the keypoint refinement module takes the most computing time during the keypoint detection process, while the FAST module takes almost a third of the total time. For the given images with fixed resolution, the time cost of the FAST detection is nearly the same, since the segment test is executed on each pixel. The same keypoints are detected by the segment test once the threshold is designated. These keypoints' corner score is first calculated and then sorted in descending order according to their score value. The time cost of the segment test, score calculation, and keypoint reordering are the same in all the test cases. The only difference resides in the time consumed

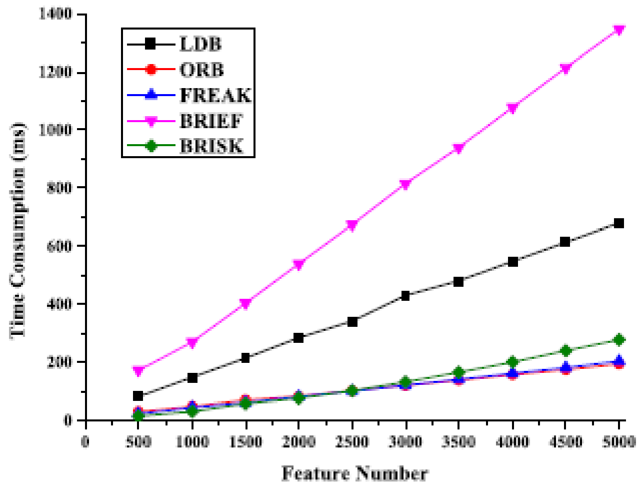


Fig. 3. Current binary descriptors' run-time consumption.

by desired keypoints storage in each test case, which takes only a tiny fraction of the total time cost. Therefore, the time consumption of the keypoint refinement also keeps constant. However, the time consumption of orientation estimation keeps linearly increasing along with the number of keypoints. The run-time consumption of the current-of-the-state binary descriptors is shown in Fig. 3. The oFAST detector (one layer, scale factor = 1, and threshold 20) is used to provide the same keypoints for the binary descriptors. The time consumption of these binary descriptors keeps linearly increasing while the keypoint number becomes larger. The image smoothing module consumes the most time of the descriptor generation. For instance, Gaussian image filtering consumes nearly 125 times larger than the ORB descriptor. 94.19% of the time is spend on the image smoothing in the BRIEF descriptor. To sum up, in order to satisfy the requirement of higher speed processing on higher quality images on mobile and embedded platform, the main objectives of this paper can be summarized as follows.

1) How to refine the keypoints detected by the FAST algorithm in a more computation-efficient manner.

2) How to achieve binary descriptor generation in faster speed than current binary feature extraction hardware implementation while keeping enough robustness to various image distortions.

3) How to make use of the bit-level and task-level parallelism hidden in feature extraction process without severely degrading image recognition performance.

IV. PROPOSED ALGORITHM

In this section, the L-BIRD algorithm is proposed to achieve binary feature extraction with lower computational complexity and memory requirement. It consists of an enhanced FAST (En-FAST) keypoint detector and a rotation-invariant binary descriptor. The performance comparison and hardware resource usage requirement are also presented in this section.

A. En-FAST Keypoint Detector

In the En-FAST keypoint detector, a directional edge detector and an optimized corner score function are utilized together to replace the Harris detector in the oFAST detector, which

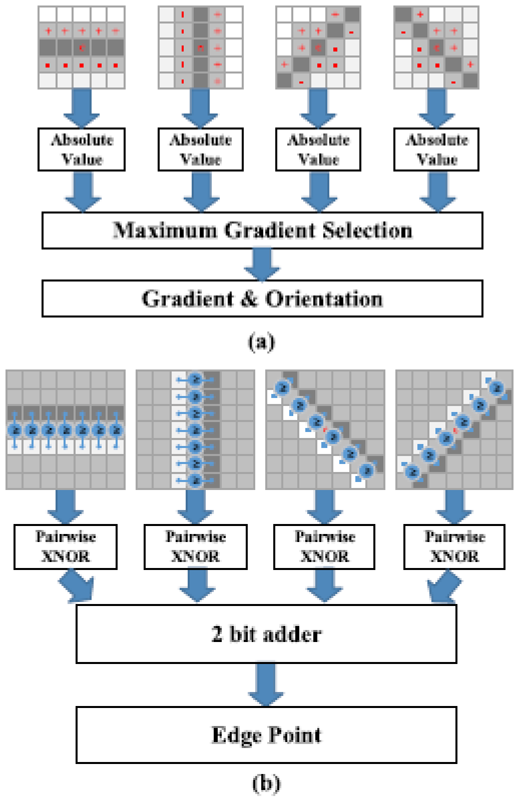


Fig. 4. Directional pairwise edge detection. (a) Edge detection method in [25]. (b) Edge point detection method in this work.

aims to refine the keypoints detected by the segment test method. The resource requirement is substantially reduced compared with the oFAST algorithm. The image pyramid, which is the same as the approach used in the oFAST algorithm, is also adopted to support multiscale features in the En-FAST.

1) Directional Edge Detector: Inspired by the directional edge detection approach used in [25], the directional pairwise edge detector (DPED) is proposed to remove the detected keypoint on edges. In [25] [shown in Fig. 4(a)], a candidate pixel centered in a 5 × 5 image block is convoluted with four-directional filtering kernels (horizontal, +45° and vertical, −45°), respectively. Then, the absolute values of these

four convolution results are compared, the direction with the maximum value is selected as the edge direction. In this paper, the aim of the DPED is determining whether or not the candidate pixel is an edge point. In order to improve the algorithm parallelism, the intensities of the corresponding pairs on the four directions are simultaneously compared rather than accumulation of their difference [as shown in Fig. 4(b)]. The candidate pixel is considered to be an edge point only if all the comparisons along this direction share the same result. If one of the directional test detects the candidate pixel as an edge point, this point is removed from the keypoint pool. Take a 7 × 7 image block as an example, the DPED requires only 26 comparison operations and 26 inclusive OR operations to get the final result. Although higher processing speed is obtained in the DPED, less edge points will be detected due to stricter constraints.

2) Score Function:

In the original FAST algorithm, non maximum suppression is applied to detected keypoints

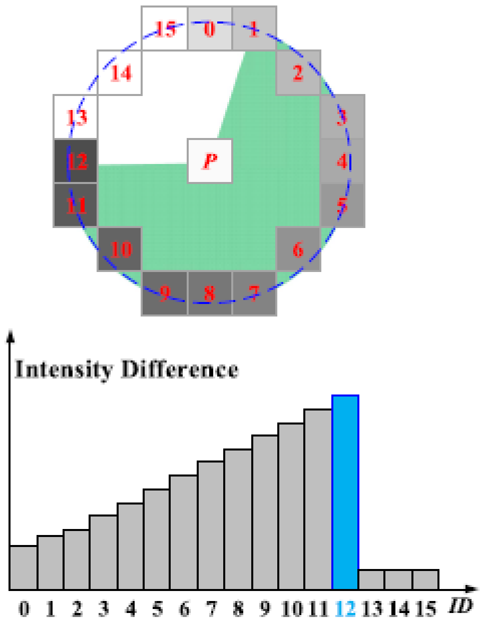


Fig. 5. Score and orientation estimation methods.

Comparison of Keypoint Refinement's Resource Usage

Resource	Harris	En-FAST
Addition/Subtraction	641	37
Multiplication	150 (fix), 4 (float)	None
Shift Operation	98	None
Comparison	None	26
Inclusive OR	None	26

TABLE II

according to their score, which is evaluated by a scorefunction *V* . As shown in the following, the function *V* is defined as the sum of the absolute difference between the pixels in the contiguous set and the center pixel:

$$V = \max\left(\sum_{x\in S_b} |I_x - I_p| - \text{th}, \sum_{x\in S_d} |I_p - I_x| - \text{th}\right) \quad (1)$$

where *Sb/Sd* are the pixels belonging to brighter or darker set. As shown in (1), the number of pixels involved in the accumulation ranges from 9 to 16. For thousands of keypoints, many of them share the same score value, which makes the nonmaximum suppression results unsuccessful. Considering the local spatial correlation of the pixels, the pixels located in the fan-shaped region are all taken into the corner response calculation. As shown in the green colored area in Fig. 5, the fan area is formed by the candidate pixel and the continuous arc. Therefore, the optimized score function *V* _ is formulated as

$$V' = \sum_{x\in \text{Area}_s} (|I_x - I_p| - \text{th}). \quad (2)$$

In the oFAST algorithm, the keypoint score function and edge point removal are implemented by the Harris detector. In the En-FAST, these functions are realized by the optimized score function and DPED, respectively. The operations required per pixel (window size: 7×7) are shown in Table II. The multiplier is not required in the En FAST, which are replaced by a comparator and inclusive OR operation.

Comparison of Resource Requirements

Resource	oFAST	Proposed
Addition/Subtraction	1906	16
Multiplication	511	None
Comparison	None	15 (max)
Trigonometric Function	1	None

TABLE III

B. Rotation-Invariant Binary Descriptor

In the descriptor generation process, the feature quality of state-of-the-art binary descriptors varies depending on the different choices of sampling pattern, orientation estimation, and comparison points' selection. Sampling pattern determines where to sample points in the patch around the keypoint. Orientation estimation measures the orientation of the keypoint and rotate the image patch to generate descriptors invariant to in-plane rotations. Comparison points are used to build the final descriptor by comparison. However, in current binary descriptors, RI is realized by rotating the keypoint-centered image patch according to the estimated orientation, which means the coordinates of all the involved sampling points should be revised before executing intensity comparison.To be worse, the descriptor generation cannot be conducted until the orientation result is available. In this work, a simple orientation estimation approach is proposed. Together with a circular symmetric sampling pattern and comparison approach, the descriptor generation can be executed with the orientation estimation simultaneously. RI is easily achieved by shifting the binary descriptors according to the result of orientation estimation.

1) Orientation Estimation:

Different from most of the binary descriptors, whose orientations are estimated in the range of image patch, only 16 pixels on the Bresenham circle around the keypoint are involved in the orientation estimation process. First, the absolute value of difference between each pair of adjacent pixels on the circle is calculated and assigned an ID number from 0 to 15. Then the ID number with the maximum value is defined as the final orientation. The proposed method is shown in Fig. 5. Consequently, the circle is divided into 16 uneven sectors. The operations required to obtain the orientation for each pixel are compared between the intensity centroid approach and the proposed method in Table III.

2) Rotation-Invariant Sampling Pattern: In this paper, a circular symmetric sampling pattern is utilized. The derived binary descriptor achieves rotation invariant easily by just shifting certain bits according to the result of the aforementioned orientation estimation approach. As shown in Fig. 6(a), the sample points of the ORB algorithm are irregularly distributed in the image patch (patch size is 31 × 31). These pixel coordinates are created by executing an offline learning process to find the uncorrelated point pairs with high variance and mean at 0.5. Although high distinctive descriptor is obtained, the large number of registers for sampling points imposes a huge burden to hardware implementation. A circular sampling pattern concentric with the keypoint is built in the BRISK and FREAK algorithms.The

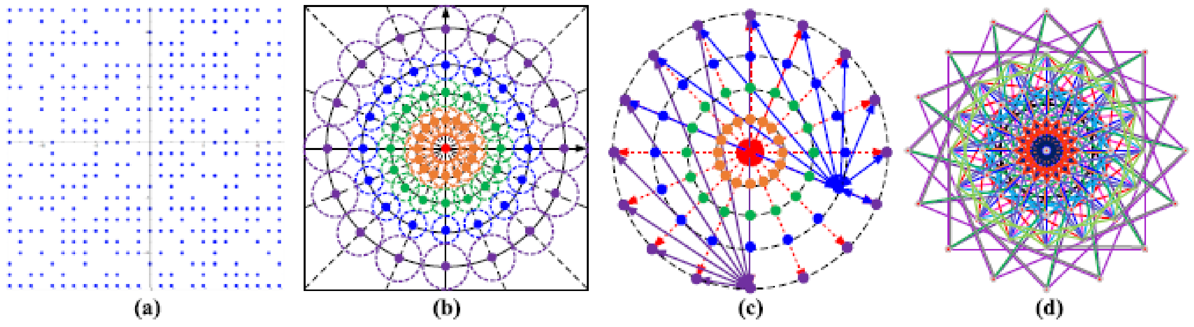


Fig. 6. Different sampling patterns and L-BIRD sampling pairs. (a) ORB. (b) L-BIRD. (c) Sampling pair selection approaches and constraints. (d) Sampling pattern of a 256-bit L-BIRD descriptor.

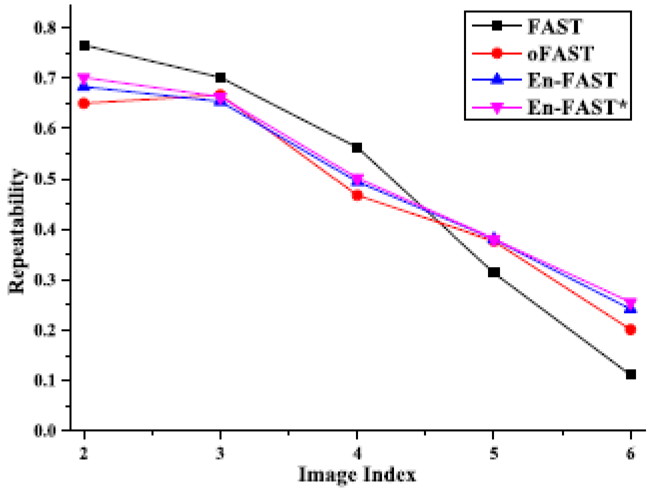


Fig. 7. Repeatability comparison (En-FAST∗ with DPED enabled only).

BRISK defines 60 points located on circles equally, while the sampling number increases from inner circles to outer circles. A human retina-inspired circular sampling grid in decreasing density order from the center to outer circle is utilized in the FREAK algorithm. A circular symmetric sampling pattern, whose sampling points are equally distributed on each circle, is utilized in this paper [shown in Fig. 6(b)]. Taken an image patch with predefined patch size 31×31, five concentric circles, whose radius are increasing from 1 to 13, are formed to get the location of desired sampling points in this work. Each intersection point between the 16 directions and concentric circles denotes a sampling point. Therefore, 64 sampling points are derived by the outer four circles. Taking the central point into consideration, there are totally 65 sampling points in the proposed sampling pattern. As proved in [26], the box filter, which is faster to compute, can be used as the filter engine instead of Gaussian filter without any matching performance loss. As shown in Fig. 6(b), a hybrid box filter structure composed of two different kernel sizes (3 × 3 and 5 × 5) is used to smooth the intensity value at the sampling points. Compared with Gaussian filtering per pixel in the ORB algorithm, 26× speedup is achieved since only 65 points are filtered by the box filters in this work. Any 2 pixels out of the 65 pixels can constitute a sampling pair to perform an intensity comparison, which would denote 1 bit of the binary descriptor. The selection methods of sampling pairs are shown in Fig. 6(c). The sampling pairs can be chosen in only one circle or across different circles, as indicated by the purple arrows and blue arrows, respectively. In order to support RI proposed in this work, 15 additional sampling pairs are determined once a pair of pixels is chosen. As illustrated by the red arrows in Fig. 6(c), the 15 pairs of pixels identified by the red dotted arrows must also be taken in the sampling pattern if the pixels identified by the red solid arrow are considered as a sampling pair. We define these circular-symmetric 16 pairs of pixels as a set. Sixteen bits of the descriptor are obtained once a set is confirmed. Consequently, it ends up with 130 possible choices to find a desirable set. To get an L-BIRD-16/32/64 descriptor, 8/16/32 sets are required to be chosen. In this paper, the learning process similar to the ORB algorithm is adopted to select the best sampling pairs with low correlation and high variance. The learning process is as follows. 1) A total of 100k keypoint-centered image patches (size: 31 × 31) are selected from the PASCAL 2012 data set [27]. All the potential sets in each image patch are also been enumerated. 2) Performing test on each set by iterating over all thepatches. 3) Sequencing all the sets according to their variance. It is preferable to have a mean of 0.5. 4) Choosing the desired number of sets with highest variance for the final sampling pattern by a greedy search. Fig. 6(d) shows the distribution of 256-bit descriptor's samplingmpattern. It is obvious that the sampling density in the central region is higher than those in the peripheral circles. Five inner circle sets are selected, while the other 11 sets are chosen across different circles.

C. Algorithm Performance Evaluation

In this section, the performance of the L-BIRD algorithm is evaluated. The repeatability and processing time of the En-FAST are compared with those of the raw FAST and oFAST algorithm on public benchmarks. The construction efficiency and recognition performance under various image distortions are also compared between the proposed work and the current-of-the-art binary descriptors

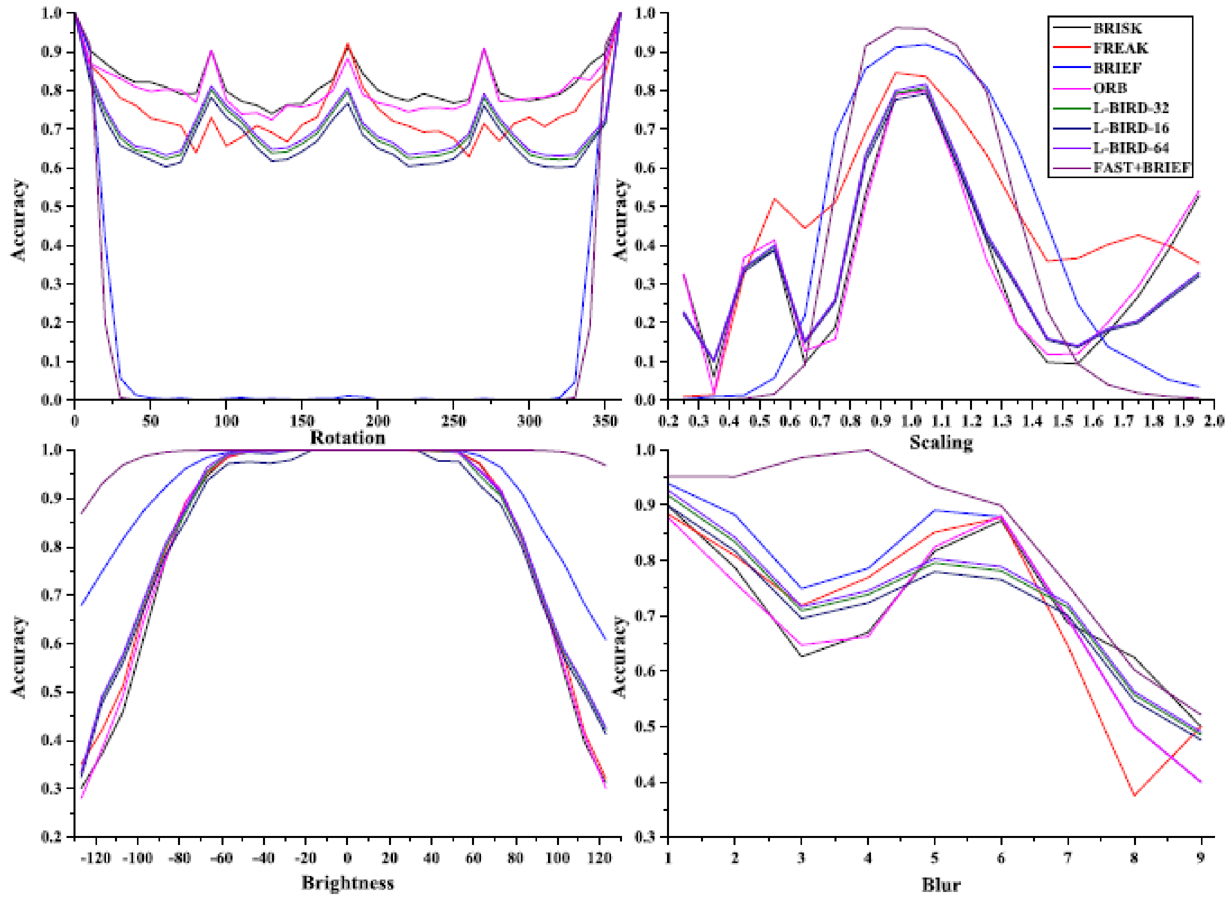


Fig. 8. Recognition accuracy comparison.

DETECTION EFFICIENCY COMPARISON

Algorithms	Time/keypoint (us)	Keypoint number	Frame/Sec
FAST	1.47	39568	30.49
FAST(nonmax)	4.74	11888	31.44
oFAST	29.12	2000	20.93
En-FAST	13.29	2524	29.81

TABLE IV

1) Keypoint Detector Performance Comparison: The evaluation uses six image sequences from the widely used Oxford dataset to test the average runtime consumption and repeatability. The timing metrics, including time cost to detect each keypoint, keypoint number, and processing speed, are all shown in Table IV. In order to derive a fair comparison, multiscale space support is disabled in the En-FAST and oFAST. The En-FAST takes 13.29 μs to detect a keypoint, which achieves more than two times speedup improvement compared with the oFAST algorithm. Meanwhile, the En-FAST achieves a comparable processing speed as the raw FAST algorithm. Although only 4.74 and 1.47 μs are required to detect a keypoint in the raw FAST algorithm with and without nonmax suppression, the large number of keypoint detected by the FAST poses a huge burden on the requirement of memory space for hardware-oriented implementation. Fig. 7

DESCRIPTOR CONSTRUCTION EFFICIENCY COMPARISON

ALGORITHMS		Time (us)
BRIEF-32		281.1
ORB-32		60.97
BRISK-32		43.82
FREAK-32		59.95
LDB-32		273.21
Proposed	16 bytes	30.81
	32 bytes	33.46
	64 bytes	37.25

TABLE V

shows the repeatability comparison among the raw FAST, oFAST, and En-FAST algorithms. The repeatability data are the average value of the six image sequences (*Bikes*, *Trees*, *Leuven*, *University of British Columbia*, *Wall*, and *Boat*) of the Oxford dataset. The En-FAST provides a comparable repeatability as the oFAST algorithm. The En FAST∗ in the Fig. 7 is used to demonstrate the impact on the detector's repeatability with the DPED integrated into the FAST algorithm. Its repeatability seems to be better than the that of the final En- FAST implementation, which is due to the larger number of keypoints. It should be noted that the high repeatability of the raw FAST comes at the expense of detecting several times larger than the keypoint number of all the other three keypoint detectors.

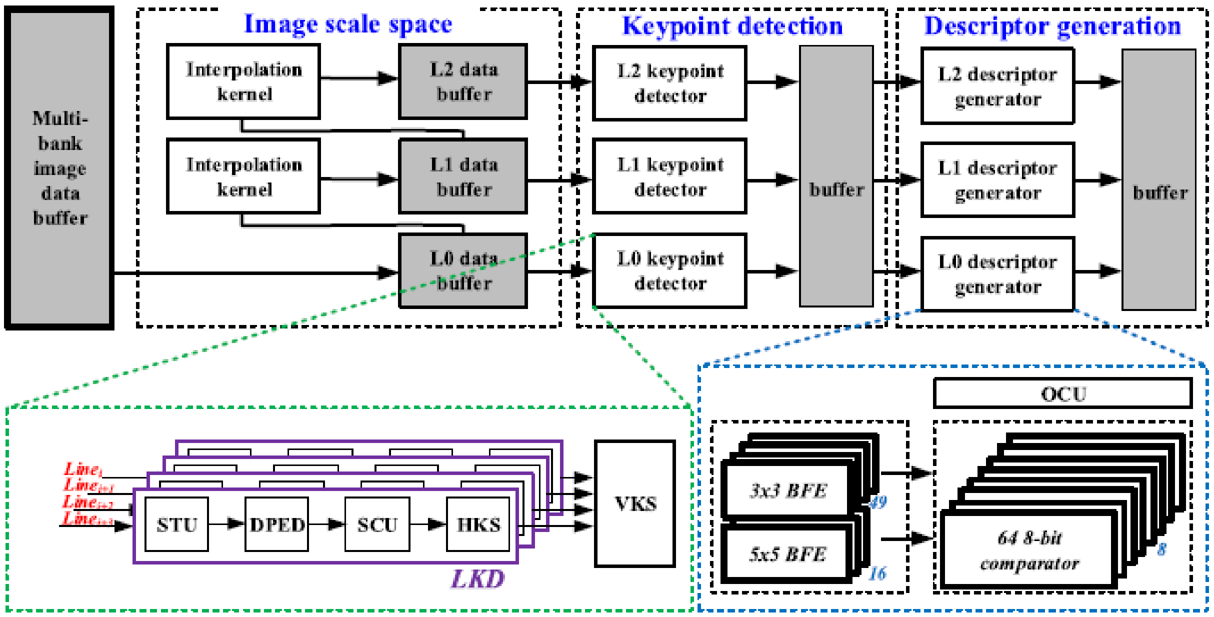


Fig. 9. System architecture of the accelerator.

2) Binary Descriptor Performance Evaluation: The time cost to build a descriptor and recognition accuracy are considered in this section to evaluate the processing speed and recognition performance of the proposed descriptor. Since the En-FAST keypoint detector and proposed descriptor are tightly coupled, the En-FAST is used to provide 1000 keypoints for the L-BIRD descriptor. The oFAST algorithm (layer number = 3 and scale factor = 2/3) is used as the keypoint detector to provide the same number of keypoints for other descriptors. Table V shows the average time cost for constructing a single descriptor of each binary descriptor algorithm. The experimental result shows that it requires 30.81, 33.46, and 37.25 μs to construct the three versions of the L-BIRD descriptor. At least 23.6% speedup is achieved compared with the BRISK descriptor. Recognition accuracy comparison is conducted to quantify the robustness in variation to different image deformations in the same simulation environment. The combination of the raw FAST detector (threshold = 40) and BRIEF descriptor is also considered in the evaluation. As observed in Fig. 8, the L-BIRD provides a comparable accuracy as the other algorithms in variance to brightness, blur, and scale change. Obviously, the L-BIRD-64 outperforms the L-BIRD-32 and L-BIRD-16 in all cases while dissipating the largest memory overhead. Due to the usage of coarse-grained orientation estimation, certain performance degradation under in-plane rotation is brought in. The BRIEF descriptor performs better under brightness and blur distortions, but it is not invariant to image rotation and scaling. It should be noted that the high performance of the FAST and BRIEF comes at the expense of detecting nearly an order of magnitude larger than those of the others in the number of keypoints.

V. HARDWARE ARCHITECTURE AND IMPLEMENTATION

A. Hardware Architecture

The overall architecture of the proposed feature extraction accelerator consists of three main components: 1) multiscale space construction; 2) En-FAST-based keypoint detection module; and 3) L-BIRD descriptor generation module. As shown in Fig. 9, three feature extraction channels are built for each layer of a three-layer image pyramid (scale factor is 2/3). In the keypoint detection stage, up to four line keypoint detectors (LKDs) are supported to achieve parallel keypoint detection across lines. The LKD consists of a segment test unit (STU), DPED, score calculation unit (SCU), and horizontal keypoint selector (HKS), which are executed sequentially. In the LKD, the EPED removes the edge point among the keypoints detected by the STU. The SCU calculates the score value of each keypoint so as to support later keypoint selection. HKS achieves keypoint selection by choosing keypoint with the highest corner score out of M keypoints in the horizontal direction. A vertical keypoint selector chooses the best keypoint from the keypoints derived by former four LKDs. Therefore, $4 \times M$ ($M = 4$ in the final implementation of this work) nonmax suppression is implemented to achieve 2D keypoint refinement. A hybrid box filter engine (BFE) and comparator array are utilized to generate the descriptor. The final descriptor is output for further processing according to the result of orientation calculation unit.

1) String Searching Based Corner Detector:

In the image recognition accelerator proposed in [20], a novel segment test method, which is optimized based on the string searching algorithm, is utilized to implement the interest point detection. Following the same principle, a new string searching algorithm-based corner detector is customized according to the requirements of the L-BIRD algorithm. Furthermore, the corresponding hardware architecture is also proposed to accomplish segment test faster. In the segment test of the original FAST algorithm, the relationship between each pixel on the Bresenham circle and candidate pixel is classified into three classes: 1) brighter; 2) darker; and 3) similar. A 32-bit binary string is derived after the segment test, while each binary test results in 2 bits. However, the information that whether the consecutive set is darker or brighter is not useful after the segment test. Basedon this observation, the ternary test is split into two parallel binary tests as

$$\mathrm{Rel}(P,x)_{\mathrm{dark}} = \begin{cases} 1 & I_x \le I_p - \mathrm{th} \\ 0 & I_x > I_p - \mathrm{th} \end{cases} \qquad (3)$$

$$\mathrm{Rel}(P,x)_{\mathrm{bright}} = \begin{cases} 1 & I_x \ge I_p + \mathrm{th} \\ 0 & I_x < I_p + \mathrm{th}. \end{cases} \qquad (4)$$

In other words, the original segment test is divided into two separate segment tests to judge the candidate whether a dark or bright keypoint, respectively. There are two 16-bit binary strings generated simultaneously. The segment test is converted into a problem of searching N ($8 < N <= 16$) continuous ones in a binary string. The special conditions of the segment test are summarized as follows.

1) Only two binary characters (0 and 1) are involved.

2) The searching pattern consists of N bits of 1.

3) The length of text is fixed (24 bits) and small for the given pattern length (9 bits in this work).

4) The start position and the end position of the consecutive strings in the text should be pointed out once the candidate pixel is considered as a keypoint.

The optimized segment test approach is proposed based on the modification of the Sunday algorithm [28]. The data processing flow of the proposed approach is described in Fig. 10. Since the test is launched on the circle, the first 8 bits of the result should be appended to the end of the text to make sure all the possible conditions are covered. The length of the *pixels* is (15 + length of the pattern) for the given searching pattern. The position of the consecutive string is described by *start_id* and *end_id*. Different from the solution in [20], the next bit adjacent to the window end is also taken into consideration in shift distance calculation. Since the position information of the continuous string is required to compute the score and estimate the orientation of each keypoint in the L-BIRD algorithm, this value should also be considered in the proposed segment test. Therefore, there are three subfunctions in the segment test: 1) window equality check; 2) shift distance calculation; and 3) continuous string position estimation. In the window equality check, each bit of the text located in the window is 1 from right to left. The comparison test is terminated once 0 appeared in the window. The window equality check is continuously executed until the window end reach the endpoint of the text. The candidate pixel is considered as nonkeypoint if the window equality check is not successful during the entire process. Temporal shift distance is figured out once the window equality check fails. The final shift distance is assigned as (pattern length +1)

Procedure of the proposed segment test

```
Segment test(pixels, start_id, end_id)
  text_end = 15;
  window_head = 0;
  start_id = 0;
  end_id = 0;
  shift = 0;
  //Segment test
  while(window_head <= text_end)
  begin
    text = pixel[window_head:window_head + 8];
    //@ window equality check
    for(scan_posi = 8; scan_posi >= 0;scan_posi--)
    begin
      if(text[scan_posi] == 0)
      begin
        //@shift distance calculation
        if(pixel[window_head+9] == 0)
          shift=10;
        else
          shift=scan_posi + 1;
        //@end shift distance calculation
        break;
      end
    end //@ window equality check
    window_head = window_head + shift;
    //@ Continuous string position estimation
    if(scan_posi == 0)
      begin
        start_id = window_head;
        end_id = start_id + 8;
        for(end_posi=start_id + 8; end_posi <= 23; end_posi++)
          if(pixel[end_posi] == 1)
            end_id = end_id + 1;
        return Keypoint;
      end
    //@ Continuous string position estimation
  end
return NOT_Keypoint;
```

Fig. 10. Processing flow of the segment test method.

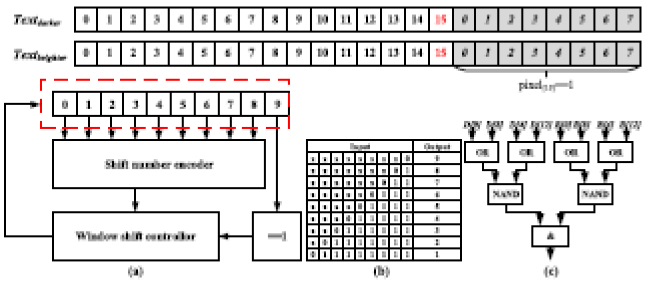


Fig. 11. (a) STU. (b) Truth table of shift number encoder. (c) Early rejection. if the next bit is 0. Otherwise, the window head is shifted to the last bit of 1 appeared in the window equality check. The start position is the window head position if the window equality check is successful. The end position is decided until the first 0 appeared in the following bits of the text. As proved in [20], the case of keypoint test is always ended earlier than the cases of nonkeypoint test. However, most of the pixels in regular images are not keypoints. Therefore, some early termination methodologies should be developed to end the segment test as soon as possible if the candidate pixel is

not a keypoint. In this paper, a similar early rejection approach, which considers four pixel values at 0, 4, 8, and 12, is also proposed to terminate the segment test simultaneously. Considering the special scenario of the proposed segment test, the candidate pixel is rejected only when two zeroes exist in both the two 16-bit strings. The detailed implementation of the STU is shown in Fig. 11. The encoder derived by Fig. 11(b) and the value adjacent to the window are both involved in the shifting distance calculation. The average number of cycles to finish segment test is 1.96 cycles, while 3 cycles are required in the worst case. With the early rejection signal generated in Fig. 11(c), 1.38 cycles is required to detect a keypoint on average

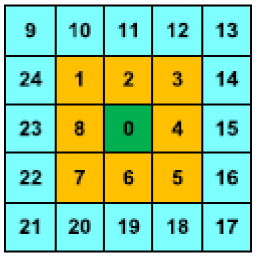


Fig. 12. Pixels partition for the optimized box filter.

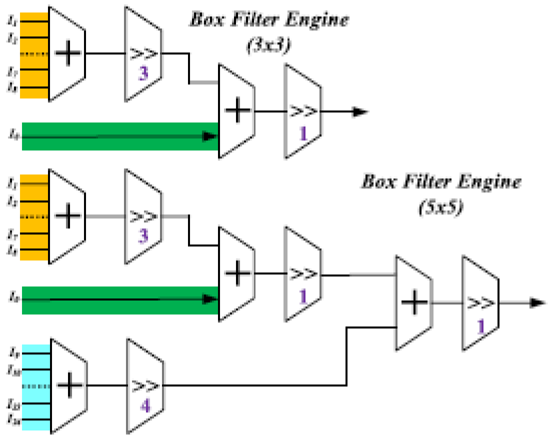


Fig. 13. BFE implementation.

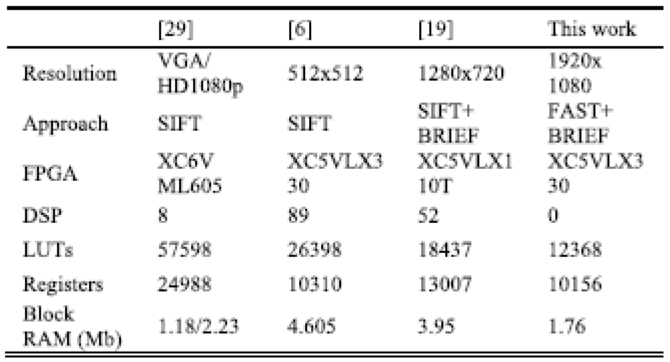
.

2) Hybrid Box Filter Engine: Two box filters with different kernel sizes (5×5 and 3×3) are involved in the image smoothing process in the proposed algorithm. To further reduce the hardware usage and accelerating the filtering process, these two filters are optimized to achieve image blurring just by addition and shifting operations. For the convenience of description, each pixel in the 5 × 5 block is given an ID number and highlighted with different colors in Fig. 12. Take the 3 × 3 box filter as an example, the average value of the peripheral 8 pixels (Nos. 1–8, yellow) is calculated by seven adders and a 3-bit shifter. The final result of the 3×3 box filter is the average value of the central pixel and the average value of outer 8 pixels. The filter value of 5 × 5 box filter is derived in the similar approach. The average value of the outer 16 pixels (Nos. 9–24, blue) is obtained to calculate the final result with the result obtained from the inner 9 pixels. The corresponding mathematical formula are defined in (5) and (6). The final gate level implementation is shown in Fig. 13. This hybrid BFE design not only simplifies the filtering process but also enhances the contributions near the center pixel to improve the smoothing result

$$Dst_{3\times3} = \left(\left(\sum_{n=1}^{8} I_n\right) \gg 3 + I_{\text{center}}\right) \gg 1 \qquad (5)$$

$$Dst_{5\times5} = \left(\left(\sum_{n=9}^{24} I_n\right) \gg 4 + Dst_{3\times3}\right) \gg 1. \qquad (6)$$

Comparison of FPGA Resource Usage



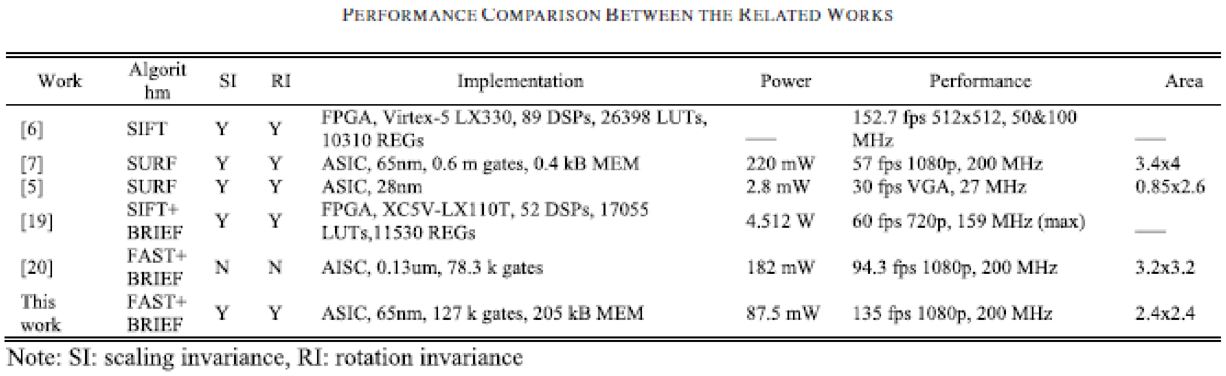
	[29]	[6]	[19]	This work
Resolution	VGA/ HD1080p	512x512	1280x720	1920x 1080
Approach	SIFT	SIFT	SIFT+ BRIEF	FAST+ BRIEF
FPGA	XC6V ML605	XC5VLX3 30	XC5VLX1 10T	XC5VLX3 30
DSP	8	89	52	0
LUTs	57598	26398	18437	12368
Registers	24988	10310	13007	10156
Block RAM (Mb)	1.18/2.23	4.605	3.95	1.76

TABLE VI

B. Hardware Implementation

The proposed accelerator is implemented in the Verilog hardware description language and verified on a FPGA platform (Xilinx Veirtex-5 LX330). Since different FPGA platforms are utilized, the number of the typical components provided by the FPGA is compared with the related work. The related key features, such as image resolution, key algorithms, and FPGA platform, are also listed in Table VI. The digital signal processor intellectual property cores are not utilized in the proposed feature extraction accelerator. Thanks to the lightweight hardware oriented optimization, which tries to achieve the same function by avoiding unnecessary complex multiplications and division operations, the proposed accelerator consumes the least lookup tables (LUTs) and registers. The proposed accelerator is fabricated by the TSMC 65-nm CMOS technology. The detailed hardware features, which are derived by the postlayout simulation, are compared with those of the most recent works in Table VII. The accelerator requires 11.96 cycles to detect a keypoint while scanning 16 pixels in the worst case, which means 0.75 cycles/pixel is achieved in the keypoint detection stage. 0.7 cycles/pixel is achieved while taking the early rejection module into consideration. It requires 6 cycles to generate a binary descriptor. The accelerator achieves at least 135 frames/s in 1080p resolution while operating at 200 MHz. It is difficult to evaluate the performance of the related solutions quantitatively due to serious differences in implementation environments, algorithms, and circuit configurations. However, it is obvious that the binary descriptor-based solutions perform better than the SIFT/SURF-based accelerators in terms of area, processing speed, and power consumption, while providing a comparable performance. Due to the lack of binary descriptorbased implementations, we target on comparing this work with [20]. Additional 48.7k gates logics are brought in to support RI and scaling invariance (SI) while keeping a higher processing speed.

Fig. 14 shows the power efficiency and area efficiency improvement of the proposed feature accelerator. It achieves 1542 frames/s/W in the power efficiency, while nearly six times improvement is achieved with respect to the SURF based accelerators proposed in [7]. Besides, the area efficiency is 23.4 frames/s/mm2, while nearly 10% improvement is achieved compared with that in [20]. Although,



PERFORMANCE COMPARISON BETWEEN THE RELATED WORKS

Work	Algorithm	SI	RI	Implementation	Power	Performance	Area
[6]	SIFT	Y	Y	FPGA, Virtex-5 LX330, 89 DSPs, 26398 LUTs, 10310 REGs	——	152.7 fps 512x512, 50&100 MHz	——
[7]	SURF	Y	Y	ASIC, 65nm, 0.6 m gates, 0.4 kB MEM	220 mW	57 fps 1080p, 200 MHz	3.4x4
[5]	SURF	Y	Y	ASIC, 28nm	2.8 mW	30 fps VGA, 27 MHz	0.85x2.6
[19]	SIFT+ BRIEF	Y	Y	FPGA, XC5V-LX110T, 52 DSPs, 17055 LUTs,11530 REGs	4.512 W	60 fps 720p, 159 MHz (max)	——
[20]	FAST+ BRIEF	N	N	AISC, 0.13um, 78.3 k gates	182 mW	94.3 fps 1080p, 200 MHz	3.2x3.2
This work	FAST+ BRIEF	Y	Y	ASIC, 65nm, 127 k gates, 205 kB MEM	87.5 mW	135 fps 1080p, 200 MHz	2.4x2.4

Note: SI: scaling invariance, RI: rotation invariance

TABLE VII

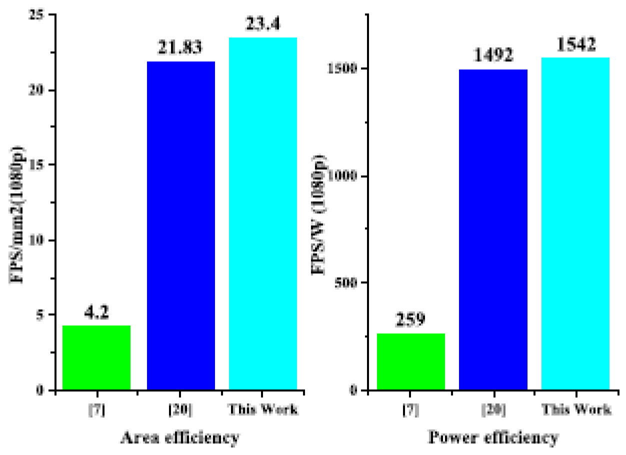


Fig. 14. Performance comparison.

certain improvements are achieved in both area efficiency and power efficiency, the unbalanced workload in the proposed architecture brings in area-efficiency loss and power efficiency loss while comparing with the unified data path architecture proposed in [20].

VI. CONCLUSION

With the aim of implementing hardware feature extraction accelerator tailored for mobile and embedded platforms, a lightweight feature extraction algorithm, L-BIRD, is proposed in this paper. A directional edge point detector and optimized corner score function are built on FAST to derive keypoints with higher quality. A coarse-grain orientation estimation approach is proposed to calculate the orientation of the image patch with less hardware resource usage. Furthermore, a circular-symmetric sampling pattern is adopted to generate the final binary descriptor. The L-BIRD algorithm outperforms the state-of-the-art descriptors in terms of processing speed without serious performance degradation by replacing the most time-consuming functions with more computationally efficient modules. Finally, the bit-level and task-level parallelism are further exploited during the hardware architecture design. The VLSI proposed in this paper will be taped out in a TSMC 65-nm CMOS technology. Our future work lies in constructing more compact binary descriptors with higher distinctiveness and developing a more efficient matching algorithm to improve the recognition performance.

REFERENCES

[1] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Proc. 7th IEEE Int. Conf. Comput. Vis.*, vol. 2. Sep. 1999, pp. 1150 1157.

[2] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded up robust features,” in *Proc. 9th Eur. Conf. Comput. Vis.*, 2006, pp. 404–417.

[3] Y. Ke and R. Sukthankar, “PCA-SIFT: A more distinctive representation for local image descriptors,” in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2. Jun./Jul. 2004, pp. II-506–II-513.

[4] K. Mikolajczyk and C. Schmid, “A performance evaluation of local descriptors,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, Oct. 2005.

[5] D. Jeon *et al.*, “An energy efficient full-frame feature extraction accelerator with shift-latch FIFO in 28 nm CMOS,” *IEEE J. Solid-State Circuits*, vol. 49, no. 5, pp. 1271–1284, May 2014.

[6] J. Jiang, X. Li, and G. Zhang, “SIFT hardware implementation for real-time image feature extraction,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 7, pp. 1209–1220, Jul. 2014.

[7] L. Liu, W. Zhang, C. Deng, S. Yin, S. Cai, and S. Wei, “SURFEX: A 57 fps 1080P resolution 220 mW silicon implementation for simplified speeded-up robust feature with 65 nm process,” in *Proc. IEEE Custom Integr. Circuits Conf.*, Sep. 2013, pp. 1–4.

[8] P. Tanskanen, K. Kolev, L. Meier, F. Camposeco, O. Saurer, and M. Pollefeys, “Live metric 3D reconstruction on mobile phones,” in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 65–72.

[9] H. Li, Y. Wang, T. Mei, J. Wang, and S. Li, “Interactive multimodalvisual search on mobile device,” *IEEE Trans. Multimedia*, vol. 15, no. 3,
pp. 594–607, Apr. 2013.

[10] X. Yang and K.-T. Cheng, “Learning optimized local difference binaries for scalable augmented reality on mobile devices,” *IEEE Trans. Vis. Comput. Graphics*, vol. 20, no. 6, pp. 852–865, Jun. 2014.

[11] S. M. Smith and J. M. Brady, “SUSAN—A new approach to low level image processing,” *Int. J. Comput. Vis.*, vol. 23, no. 1, pp. 45–78, May 1997.

[12] C. Harris and M. Stephens, “A combined corner and edge detector,” in *Proc. 4th Alvey Vis. Conf.*, 1988, pp. 147–151.

[13] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger, “Adaptive and generic corner detection based on the accelerated segment test,” in *Proc. 11th Eur. Conf. Comput. Vis.*, 2010, pp. 183–196.

[14] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2564–2571.

[15] S. Leutenegger, M. Chli, and R. Y. Siegwart, “BRISK: Binary robust invariant scalable keypoints,” in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2548–2555.

[16] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “BRIEF: Binary robust independent elementary features,” in *Proc. 11th Eur. Conf. Comput. Vis.*, 2010, pp. 778–792.

[17] X. Yang and K.-T. Cheng, “LDB: An ultra-fast feature for scalable augmented reality on mobile devices,” in *Proc. IEEE Int. Symp. Mixed Augmented Reality (ISMAR)*, Nov. 2012, pp. 49–57.

[18] A. Alahi, R. Ortiz, and P. Vandergheynst, “FREAK: Fast retina keypoint,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012,
pp. 510–517.

[19] J. Wang, S. Zhong, L. Yan, and Z. Cao, “An embedded system-on-chip architecture for real-time visual detection and matching,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 3, pp. 525–538, Mar. 2014. [20] J.-S. Park, H.-E. Kim, and L.-S. Kim, “A 182 mW 94.3 f/s in full HD pattern-matching based image recognition accelerator for an embedded vision system in 0.13-μm CMOS technology,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 5, pp. 832–845, May 2013.

[21] E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” in *Proc. 9th Eur. Conf. Comput. Vis.*, 2006, pp. 430–443.

[22] A. Canclini, M. Cesana, A. Redondi, M. Tagliasacchi, J. Ascenso, and R. Cilla, “Evaluation of low-complexity visual feature detectors and descriptors,” in *Proc. 18th Int. Conf. Digit. Signal Process. (DSP)*, Jul. 2013, pp. 1–7.

[23] O. Miksik and K. Mikolajczyk, “Evaluation of local detectors and descriptors for fast feature matching,” in *Proc. 21st Int. Conf. Pattern Recognit. (ICPR)*, Nov. 2012, pp. 2681–2684.

[24] J. Heinly, E. Dunn, and J.-M. Frahm, “Comparative evaluation of binary features,” in *Proc. 12th Eur. Conf. Comput. Vis.*, 2012, pp. 759–773.

[25] P. Zhao, H. Zhu, H. Li, and T. Shibata, “A directional-edge-based real-time object tracking system employing multiple candidate-location generation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 3, pp. 503–517, Mar. 2013.

[26] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua, “BRIEF: Computing a local binary descriptor very fast,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1281–1298, Jul. 2012.

[27] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*. [Online]. Available: http://pascallin.ecs.soton.ac.uk/challenges/VOC/ voc2012/workshop/index.html, accessed Feb. 2014.

[28] A. Hume and D. Sunday, “Fast string searching,” *Softw., Pract. Exper.*, vol. 21, no. 11, pp. 1221–1248, Nov. 1991.

[29] L.-C. Chiu, T.-S. Chang, J.-Y. Chen, and N. Y.-C. Chang, “Fast SIFT design for real-time visual feature extraction,” *IEEE Trans. Image*
Process., vol. 22, no. 8, pp. 3158–3167, Aug. 2013.