# High-Throughput and Area-Optimized Architecture for rBRIEF Feature Extraction

Thinh Hung Pham, *Member, IEEE*, Phong Tran, and Siew-Kei Lam, *Member, IEEE*

***Abstract*—Feature matching is a fundamental step in many real-time computer vision applications such as Simultaneous Localization And Mapping (SLAM), motion analysis and stereo correspondence. The performance of these applications depends on the distinctiveness of the visual feature descriptors used, and the speed at which they can be extracted from video frames. When combined with standard key-point detectors, the rotation-aware Binary Robust Independent Elementary Features (rBRIEF) descriptor has been shown to outperform its counterparts. In this paper, we present a deep-pipelined stream processing architecture that is capable of extracting rBRIEF features from high-throughput video frames. To achieve high processing rate and low complexity hardware, the proposed architecture incorporates an enhanced moving summation strategy to calculate the key-points' patch moments and employs approximate computations to achieve patch rotation. Multiplier-less circuitry is introduced throughout the architecture to avoid the use of costly multipliers. Implementation on the Altera Aria V device demonstrates that the proposed architecture leads to 53.3% reduction in hardware resources (adaptive logic modules) while achieving 50% higher accuracy (in terms of average Hamming distance) when compared to the state-of-the-art architecture. In addition, the proposed architecture is able to process high-resolution (1920×1080) images at 60 fps while consuming only 456.15 mW power.**

## I. Introduction

The pervasive adoption of cameras in various cutting-edge technologies today such as self-driving cars, robotics, virtual reality and augmented reality, have elevated the need for computer vision algorithms that are robust and can deliver high-speed solutions. A core application in these technologies is visual odometry or Simultaneous Localization And Mapping (SLAM) [1], which enables the system to localize itself in unknown environments. Feature matching plays a central role in this application to estimate the motion trajectory from one frame to the next. Evidently, the choice of visual feature descriptors plays an important role in feature matching. In addition, the feature extraction process must be achieved at high-speed to enable feature matching between each consecutive frames in order to improve robustness.

There are many recent advances in the development of feature descriptors to address the need for low complexity and robustness in feature matching systems. Scale-Invariant Feature Transform (SIFT) [2] and Speeded Up Robust Features (SURF) [3] were originally introduced as feature descriptors for feature matching systems. SIFT is robust to rotation and illumination changes. SIFT employs eight bin orientation histograms for each of the 16 sub-blocks to generate 128-dimensional vector to represent each description. The size of the required vector reduces the performance of feature matching. SURF is a faster successor of SIFT but requires 256 bytes to represent the descriptors. The descriptor size is still infeasible for real-time feature matching on high resolution images.

Local binary descriptors such as Binary Robust Independent Elementary Features (BRIEF) [4], Rotated BRIEF (ORB) [5], Binary Robust Invariant Scalable Keypoints (BRISK) [6], and Fast Retina Key-point (FREAK) [7] were introduced to supersede SIFT and SURF. They were developed to increase computational speed and reduce memory storage. The works in [5], [6], and [7] have demonstrated the superiority of the BRIEF descriptor and its variants, namely ORB and rotated BRIEF (rBRIEF), over the SIFT and SURF descriptors. In addition, the experimental results on visual SLAM [1] showed that BRIEF performs better than other binary descriptors, i.e. BRISK and FREAK. The BRIEF feature descriptor is represented by a 256-bit vector, which is obtained by comparing the intensity of 256 pairs of pixels in a patch centered at the key-point. The 256 pairs of pixels are often denoted as BRIEF point-pairs or sampling patterns. Each comparison is called a binary test, as it produces a '1' or '0' depending on the comparison result.

ORB descriptor integrates orientation and scale pyramid to compensate for the sensitivity to rotation and scale in BRIEF. In addition, a learning method is employed to choose the BRIEF point-pairs that have high variance and low correlation. This enhancement resulted in the rotation-aware Brief (rBRIEF) descriptor. ORB, which consists of a key-point detector and rBRIEF descriptor, quadruples FREAK and BRISK in terms of extraction time and leads to higher matching accuracy. The work in [8] provided a more comprehensive assessment for a combination of detectors and descriptors pairs in Unmanned Aerial Vehicles (UAV) guidance system. The result showed that ORB descriptor has an excellent performance. The authors in [9] evaluated the performance of BRIEF, ORB, and BRISK under various distortion and transformation images using five sets of metrics: precision, recall, matching score, entropy, and putative match ratio. BRIEF's fixed pattern produces better recall and putative match ratio than ORB or BRISK in non-geometric transforms but resulted in lower matching scores. Under the effects of rotation and scaling, ORB and BRISK outperform BRIEF in all the metrics considered. While BRIEF exhibits slightly higher recall and matching score over ORB and BRISK, the latter two descriptors result in higher quality matches. An extensive

Thinh Hung Pham, Phong Tran, and Siew-Kei Lam are with School of Computer Science and Engineering, Nanyang Technological University, Singapore (email:hung3@e.ntu.edu.sg)

study on 42 combination of detectors and descriptors under various image transformations is presented in [10]. The study revealed that ORB, which integrates rBRIEF, exhibited reliable performance and was deemed the most suitable descriptor for many vision applications.

The studies undertaken so far have concluded that the ORB descriptor not only provides excellent performance in many vision tasks but also has the advantage of requiring low computational complexity which makes it suitable for real-time applications. ORB feature extraction comprises of key-point detection (e.g. FAST [11] and Harris [12] corner detectors), and rBRIEF descriptor computation. The work in [13] reported that the conventional BRIEF architecture requires at least $2\times$ more hardware resources than the FAST architecture. Therefore, reducing the hardware resources for rBRIEF computation will lead to low complexity ORB feature extraction architecture that is well suited for realization on embedded systems with tight area constraints. As such, this paper focuses on developing an area-efficient architecture which is aimed at extracting rBRIEF features on high-rate pixel streams from the camera. The proposed architecture can be integrated into a feature extraction accelerator for visual odometry or SLAM system.

### A. Related Works

In this section, we review existing hardware designs for extracting BRIEF feature descriptors and its variants. The hardware accelerator in [14] computes a 128-bit length descriptor from BRIEF point-pairs that are sampled using Gaussian distribution in 9x9 image patches. The work in [15] utilizes a learning method [5] to build discriminative and uncorrelated binary tests. In addition, it employs an image data allocation scheme, wherein 4 pixels are clustered in a memory addressable location, so computing 256-bit descriptor only requires 15 clock cycles. To minimize the resource usage, [16] presented an architecture that compares the pixel intensities of a single BRIEF point-pair in each cycle. However, to generate a 256-bit descriptor, the BRIEF module can only process one key-point (i.e. corner) in 256 clock cycles.

As discussed in the previous section, rBRIEF has emerged to overcome the limitations of the BRIEF descriptor which suffers from poor invariance to rotation and scale changes. The feature extraction architecture in [17] and [18] combined oFAST (oriented FAST) and rBRIEF, while the work in [19] integrates Harris-Stephens with rBRIEF. The FPGA-based ORB implementation in [18], which integrates a key-point detector with rBRIEF, optimizes the word length by truncating the centroid to 8 bits based on prior studies on the distance error, which significantly reduces the number of registers and Look-up tables (LUTs). However, the authors only focused on improving memory allocation for real-time SLAM but did not provide detailed architecture for calculating the centroid, rotating the sampling patterns, and generating the key-points. In [17], data reuse is utilized by exploiting the incoming pixel streams and a memory scheme. The same patch is used for computing the descriptors of any two consecutively detected key-points whose distance is less than the width of the patch. This leads to over 17% reduction of the external memory bandwidth. Also, rather than using centroid-based orientation that leads to high computational complexity, the approach in [20] is adopted, which relies on the comparison of adjacent pixels on the Bresenham circle to determine the orientation. Their work mainly focused on the architecture for FAST and Non-Maximal Suppression (NMS). The description of the architecture for rotating the sampling patterns and computing the rBRIEF descriptors was not elaborated. The authors in [19] presented the implementation of ORB feature extraction using image pyramid to obtain scale invariance. However, the authors have not provided sufficient accuracy evaluation of the approximation methods used in their architecture.

The aforementioned works focus on non-streaming architectures for feature extraction in which the input image frame is first stored in memory. The pixels are then sequential accessed and processed by time multiplexing circuitries. This reduces computational resources, but results in increased hardware area for buffering and significantly lower throughput. On the other hand, stream processing architecture offers tremendous benefits. It not only leads to higher throughput, but also eliminates the need of large image frame buffers resulting in reduction in both power dissipation and hardware utilization [12].

### B. Contributions

This paper presents a low complexity stream processing architecture that is capable of extracting rBRIEF features from high-throughput video frames. The main contributions of this paper are:

- A novel deep-pipelined architecture that can extract rBRIEF feature descriptors on-the-fly from high-rate input pixel streams without utilizing any image frame buffers.
- An enhanced moving summation strategy is proposed to calculate the key-point's patch moments, which significantly reduces the hardware complexity. The proposed architecture does not use any conventional multipliers to perform rBRIEF feature computation. Instead multiplier-less circuitries leveraging on single constant multiplication are introduced throughout the architecture to avoid the use of costly multipliers.
- Approximate computation based on novel angle discretization methods is applied to compute the orientation of the patch. In addition, a folded architecture is proposed to rotate the BRIEF point-pairs by exploiting the sparsity of key-points.
- A comprehensive evaluation on the effect of approximation and bit-width truncation is undertaken to obtain the optimal trade-off between accuracy and complexity of the proposed architecture.

This paper is organized as follows: Section II provides a background on the BRIEF and rBRIEF feature extraction. Section III provides detailed description of the proposed feature extraction architecture. The implementation results in terms of efficiency and accuracy are discussed in Section IV. Finally, Section V concludes the paper.

## II. Feature Extraction

### A. Binary Robust Independent Elementary Features (BRIEF)

The BRIEF [4] binary feature descriptor is extracted from image patches centered at detected key-points (e.g. corners). The extraction process relies on a set of binary tests that compares the intensity of each BRIEF point-pair in the smoothed image patch to generate a bit-string. Specifically, the BRIEF descriptor is a $n_d$ dimensional bit-string where each bit corresponds to the comparison result in (1).

$$\tau(p; a, b) := \begin{cases} 1 \ if \ p(a) < p(b) \\ 0 \ otherwise, \end{cases} \tag{1}$$

where $p$ is the smoothed image patch, $p(a)$ and $p(b)$ are the intensity of a BRIEF point pair at location $a$ and $b$, respectively. The BRIEF descriptor can be expressed as follows.

$$f_{n_d}(p) := \sum_{1 \le i \le n_d} 2^{i-1} \ \tau(p; a_i, b_i). \tag{2}$$

It is essential to consider the number of binary tests as well as the distributions of samples in the hardware implementation as they could dramatically impact the speed, efficiency, memory storage, and recognition rate. Results in [4] suggest that a descriptor of 200 bits and beyond do not yield notable increase in recognition rate. BRIEF descriptors of 256 bits yield near-optimum result and descriptors of 512 bits only provide a small improvement. As our goal is to implement a high-throughput feature descriptor extraction which has low complexity, our architecture will be based on 256-bit descriptors.

### B. Rotation-Aware Brief (rBRIEF)

rBRIEF [5] has emerged to overcome the limitations of the BRIEF descriptor which suffers from poor invariance to rotation and scale changes. This is achieved by computing intensity centroid [21] of the image patch. The intensity centroid algorithm first calculates $m_{01}$ and $m_{10}$ which are the moments of the key-point's patch:

$$m_{01} = \sum_{x,y} y \times p_{x,y}, \quad m_{10} = \sum_{x,y} x \times p_{x,y}, \tag{3}$$

where $p_{x,y}$ is the intensity of pixel at the position (x,y). Next, it computes the arc tangent value of $m_{01}$ and $m_{10}$:

$$\theta = atan2(m_{01}, m_{10}). \tag{4}$$

Once the orientation is determined, the positions of the predetermined BRIEF point-pairs are steered by multiplying each pair of coordinates with the rotation matrix:

$$\begin{bmatrix} rx \\ ry \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{5}$$

Two characteristics that affect the distinctiveness of a feature descriptor are variance and correlation. Features with high variance are more discriminative, while those with low correlation imputes the distinctiveness of the features. Steered BRIEF exhibits neither low correlation (due to its high intrinsic eigenvalues), nor high variance (as once oriented, the distribution of the BRIEF point pairs becomes more scattered). To preserve the high variance of BRIEF and to minimize the correlation of binary tests, a learning method is applied with a greedy search among the training tests to determine the best point pair patterns. Our proposed rBRIEF extractor uses the learned point pair patterns presented in [5] which improves the rBRIEF descriptor in terms of correlation and variance.

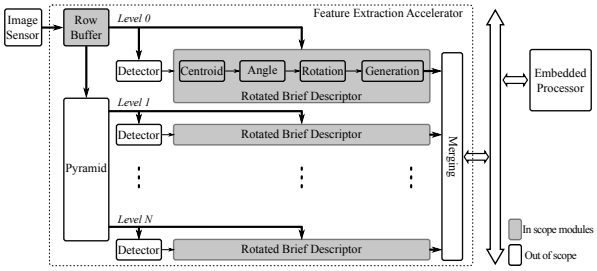



Fig. 1: Architecture of a typical multilevel Feature Extraction Accelerator.

## III. Proposed Architecture

In this section, we present the proposed rBRIEF architecture which can be integrated in a feature extraction accelerator for computer vision systems (e.g. visual SLAM system).

### A. Feature Extraction Accelerator

Fig. 1 illustrates the overview of a computer vision system which tightly couples an embedded processor and the feature extraction accelerator. The embedded processor offloads the compute intensive feature extraction process onto the accelerator, while it handles other menial tasks such as descriptor matching. The accelerator receives the pixel stream from an image sensor at the rate of one pixel per clock cycle. Under the assumption that the image is read sequentially using a raster scan mode, the incoming pixels need to be cached locally using a set of row buffers [12].

To enable the feature extraction for an image at multiple resolutions of scale invariance, an image pyramid computation creates multiple additional down-sampled copies, at successively lower resolutions. The image pyramid can be implemented using the approach described in [19], [22], [23]. The description of the image pyramid module is beyond the scope of the paper. In addition, the key-points at each pyramid level must be first determined by a detector. The detector can employ FAST or Harris corner detection. The implementation of the detector has been extensively studied in literature [11], [13], [24], [25]. The detector is also beyond the scope of this paper.

The rBRIEF feature extractor computes the 256-bit descriptors from image patches that are centered at the detected key-points. This paper presents a high-throughput and area optimized architecture for the rBRIEF feature extractor which requires intensive computation and consumes a large amount of resource utilization in feature extraction systems. The proposed architecture can be easily extended to multilevel by simply replicating the design in Fig. 2 for each pyramid

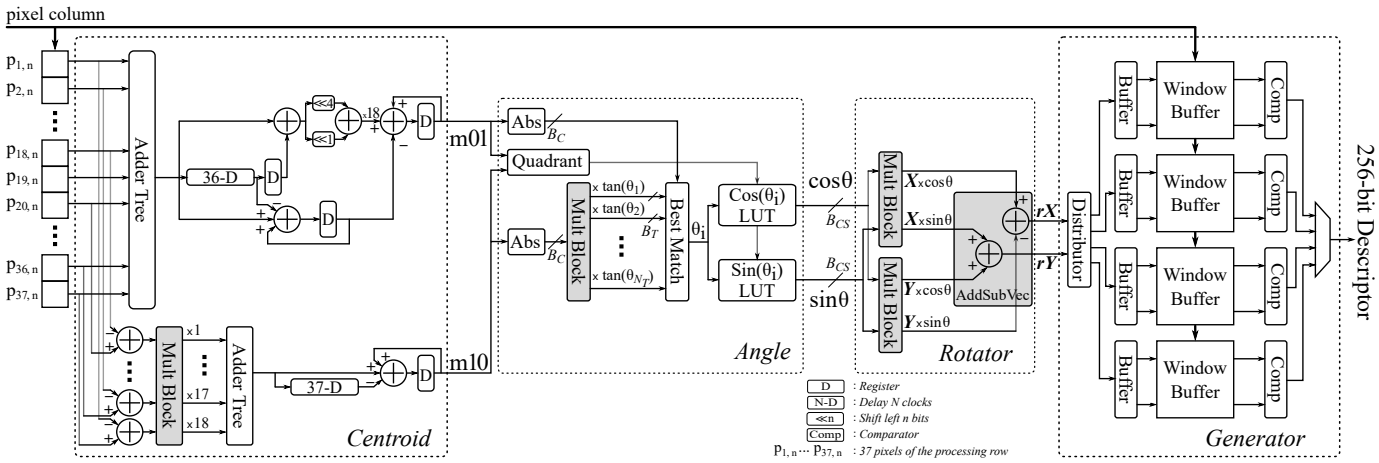



Fig. 2: Block diagram of rBRIEF feature extractor.

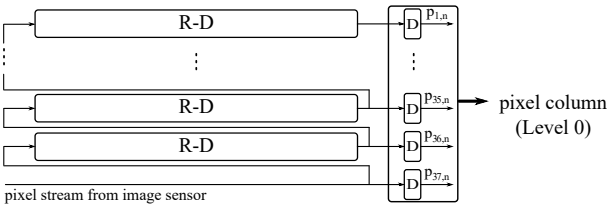



Fig. 3: Row buffer implementation.

level. As such, our work will lead to significant reduction in resource and power consumption, and higher throughput, which is important for deployment on embedded platforms such as UAVs. It is worth mentioning that compared to existing implementations e.g. [18], [19], the proposed stream processing architecture does not require any image frame buffers which significantly reduces the required hardware resources.

### *B. Row Buffering*

As an image from a sensor is sent to the feature extraction accelerator in a raster scan mode at a rate of one pixel per cycle, the incoming pixels are cached locally using a set of row buffers [12]. The row buffer's implementation in this paper is different from [12] as it is based on circular buffer architecture that is realized using block memory bits instead of shift registers. The length of each row buffer ($R$) is equivalent to the horizontal resolution of the image, and hence each row buffer effectively delays the input by $R$ clock cycles. Fig. 3 depicts the implementation of the row buffer. The pixels at the tail end of each row buffer are shifted into a set of registers to generate a pixel column, which will be processed by the pipeline feature extraction architecture. The rBRIEF implementation computes the descriptors from image patches, where each patch comprises of a $37 \times 37$ pixel window that is centered at a detected key-point. Since the row buffers are required to generate a 37 pixel column, 36 *Delay R* row buffers are employed.

### *C. rBRIEF Extraction Implementation*

The rBRIEF feature extractor consists of four units, i.e. *Centroid*, *Angle*, *Rotator*, and *Generator* as shown in Fig. 2. The *Centroid* unit is used to compute the moments (i.e. $m01$, $m10$) of a key-point's patch as in Eq. (3). The *Angle* unit calculates the orientation angle of the patch based on the values of the moments. The *Rotator* unit uses the orientation angle from the *Angle* unit to rotate the coordinates of the pre-determined BRIEF point-pairs as in Eq. (5). The rotated coordinates of the BRIEF point-pairs are used to access the corresponding pixel intensities in the window buffer. The *Generator* unit fetches the pixel intensities of the rotated BRIEF point-pairs from the window buffer to generate the description bits as in Eq. (1). The proposed rBRIEF feature extractor is designed with a low-complexity deep-pipelined architecture. The proposed hardware-efficient strategies enable additional pipeline stages to be introduced in the computation units (i.e. 5 stages for Centroid unit, 5 stages in Angle unit, and 9 stages in Rotation unit) to increase throughput, while still achieving significantly reduced area utilization.

In the following sub-sections, we propose hardware-efficient strategies to substantially reduce the hardware complexity of the rBRIEF extraction module without compromising on the processing throughput and descriptor accuracy. The proposed strategies rely on hardware-efficient moving summation, multiplier-less circuitry and folded architectures. Moving summation is a recursive mathematic function that has been previously presented to calculate auto-correlation in wireless system [26]. In this paper, we manipulate the calculation of image moments in the moving summation's formula such that it can be effectively computed in hardware. Multiplier-less circuitry performs multiplication with constants using shifters and adders instead of actual multipliers [27]. This has been shown to achieve hardware efficiency for convolution calculation in wireless systems [28]. In this work, we explore the accuracy effects of using multiplier-less circuitry for the computation of the *Centroid*, *Angle*, and *Rotator* units. Finally, folded architecture was presented in [29] for time multiplexing to reduce the hardware usage of an offset estimation in modulation systems. In this work, we investigate how folded architectures can be used in the *Rotator* unit to significantly reduce the resource utilization without compromising on the accuracy by exploiting sparsity of the key-points.

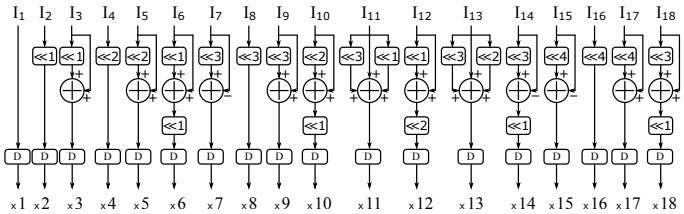



Fig. 4: Implementation of *Mult Block* in the *Centroid* unit.

*1) Centroid Unit:* We proposed a moving summation strategy to reduce the hardware complexity for computing the moments for $37 \times 37$ patches in a stream processing manner. The proposed strategy mathematically manipulates Eq. (3) to operate in a recursive manner by i) expressing the current value and next value with respect to input values, ii) subtracting the next value from the current value, and simplifying the subtraction, iii) rewriting the next value expression with respect to the current value and the subtraction. The derived formula is expressed as follows:

$$\begin{aligned} m01_{n+1} &= m01_n + 18 \times (C_n + C_{n-37}) - S_n, \\ m10_{n+1} &= m10_n + xC_n - xC_{n-37}, \end{aligned} \tag{6}$$

where

$$\begin{aligned} S_{n+1} &= S_n + C_n - C_{n-36}, \\ C_n &= \sum_{x=1}^{37} p_{x,n}, \\ xC_n &= \sum_{x=1}^{18} x \times (p_{38-x,n} - p_{x,n}) \end{aligned} \tag{7}$$

The $n+1$ subscript denotes the next value of the current value $n$. The $n-D$ subscript is the value after delay of $D$ clock cycles. $p_{x,n}$ denotes the pixel intensity of the $x^{th}$ element in the incoming pixel column of 37 pixels.

The implementation of the mathematical formulation for the *Centroid* unit is shown in Fig. 2. $C_n$ is calculated using an *adder tree* to accumulate the 37 values of an incoming pixel column. The computation of $xC_n$ utilizes a constant multiplier block to reduce the hardware resources and power consumption [28]. The constant multiplier block performs the addition, subtraction and shift operations to multiply an input with a constant. We use the *Hcub* algorithm in [27] to optimize the number of operations in constant multiplier block. The circuitry performing the multiplication in the *Centroid* unit is shown in Fig. 4.

*2) Angle Unit:* The processing of the *Angle* unit consists of two steps. The first step is to determine the orientation angle ($\theta$) and the second calculates the rotation terms ($cos\theta, sin\theta$). The orientation angle is computed based on $atan2(m01; m10)$, which is a costly operation if a direct implementation is adopted. We propose an approximate computation based on discretization of the angle $\theta$. Firstly, *Quadrant* determines the quadrant of the angle based on the signs of $m01$, $m10$. At the same time, the absolute values of $m01$, $m10$ are extracted by *Abs* so that the required angle falls within the quadrant $[0; \frac{\pi}{2})$. Then, the angle is discretized into $N_T$ values in the quadrant. Hence $\theta$ can only be one of $N_T$ possible values and their tangents can be precomputed. Instead of using division and $atan2$ operation, $\theta$ can be approximated as $\theta_i$ that best meets the condition in Eq. (8). The multiplication between $abs(m10)$ and $N_T$ possible tangent values is performed using a constant multiplier block.

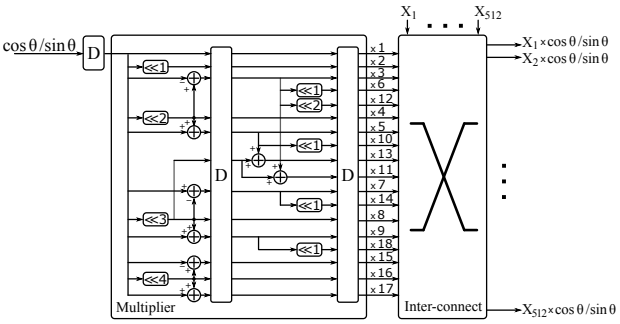



Fig. 5: Circuitry of *Mult Block* in the *Rotator* unit.

$$Abs(m01) = Abs(m10) \times tan(\theta_i) \tag{8}$$

After $\theta_i$ is found as one of $N_T$ possible values, the values of $cos\theta_i, sin\theta_i$ can be determined using Look-up Tables (LUT) instead of using iterative Coordinate Rotation Digital Computer (CORDIC) engines [30], [31]. Each $cos\theta_i/sin\theta_i$ LUT consists of $N_T$ values of $cos\theta_i/sin\theta_i$ based on the discretized $\theta$ angles. The values of $cos\theta_i/sin\theta_i$ are then adjusted using their sign values. The accuracy of this unit is dependent on $N_T$, the number of bits used to represent the discretized angle values ($B_T$), the number of bits for representing the centroid moment's absolute values ($B_C$), and the number of bits to represent the cos/sin values ($B_{CS}$). In Section IV-A, we will explore the bit representations that will lead to the best accuracy and hardware complexity trade-offs.

*3) Rotator Unit:* After the rotation terms of the orientation are calculated, the *Rotator* unit performs the rotation with the $\theta$ angle on the set of BRIEF point-pairs as in Eq. (5). The hardware implementation is depicted in the *Rotator* unit of Fig. 2. $\mathbf{X}, \mathbf{Y}$ are the coordinate vectors of the 512 samples. Performing rotation on a set of 512 samples for each key-point patch requires large amount of hardware resources. Theoretically, a total of 2048 multiplications and 1024 additions/subtractions are required. As such, we devised a hardware-efficient method to achieve this rotation as follows.

The calculation of the *Rotator* unit comprises of two steps. First, the coordinates of the sample set are multiplied with the $cos\theta$ and $sin\theta$ values using multiplier-less constant multiplier block. Then an *AddSubVector* block performs addition/subtraction of the multiplication's results to generate the rotated coordinates.

The proposed multiplier of the *Rotator* unit is shown in Fig. 5. To reduce the hardware complexity, we only use 18 constant multipliers instead of multiplying the $cos\theta$ and $sin\theta$ values with the 512 sample coordinates. This is possible as the coordinates of the sample set are within the range $[-18, 18]$ and can be precomputed. The multiplications of $cos\theta$ and $sin\theta$

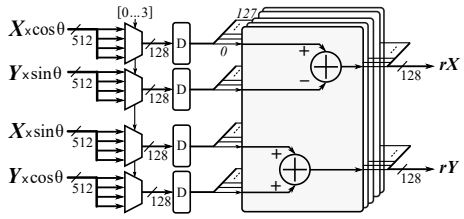



Fig. 6: 4-folded Circuitry of *AddSubVec* block in the *Rotator* unit.

with 18 constant values denoted $\times 1$ to $\times 18$ are performed using a multiplierless *Mult Block*. Based on the known value of each element in the coordinate vector, the corresponding result of the multiplication is either complemented (i.e negative value) or not (i.e positive value) to obtain $\mathbf{X} \times cos\theta$, $\mathbf{X} \times sin\theta$, $\mathbf{Y} \times cos\theta$, and $\mathbf{Y} \times sin\theta$. These are used to generate the rotated coordinates.

An efficient folded circuitry as illustrated in Fig. 6 is employed for the *AddSubVector*. This circuitry employs time multiplexing to reuse the hardware resources that results in significantly reduced hardware usage compared to a straightforward implementation. This approach leverages on the assumption that the Non-Maximal Suppression (NMS) operation using a $7 \times 7$ window [32] in the detector does not allow the detection of two key-points in 4 continuous pixels. We employ a 4-folded circuitry to perform the addition/subtraction of the rotation in 4 consecutive clock cycles, where 128 additions and 128 subtractions are simultaneously computed in each cycle. Two shift registers ($SR$) are needed to store the resulting 512 rotated coordinates ($\mathbf{rX}$ and $\mathbf{rY}$).

*4) Generator Unit:* The *Generator* unit receives the rotated samples that are calculated by the previous component and is responsible for computing the 256-bit descriptor from these samples. A possible approach is to cache the image patch in a window buffer and the rotated samples in a buffer that allows 2 samples to be read (i.e. using 2 ports memory blocks) per cycle for performing a binary test. In this case, 256 cycles are required to generate the 256-bit descriptor of a key-point. However, if a new key-point is identified within this duration of 256 cycles, its corresponding image patch cannot be loaded into the window buffer which is currently being accessed for computing the descriptor of the previous key-point. As such, this new key-point will have to be dropped if only one *Generator* unit is employed.

In order to maintain the processing throughput at the rate of the incoming pixels while minimizing the number of dropped key-points, we employ multiple components that can compute the descriptors of several key-point patches in parallel. It is worth mentioning that this approach is different from [19] as we only replicate the descriptor generator component (consisting of window buffers and binary tests) instead of the entire rBRIEF extraction module. This approach is possible because the previous units (i.e. Centroid, Angle and Rotator) still maintains a continuous calculation for the subsequent key-points. This results in a large reduction in hardware consumption compared to the work in [19].

A *Distributor* is responsible for selecting an available descriptor generator components for a new key-point. If at least one descriptor generator component is available when a new key-point is detected, the key-point is not dropped. While a sufficient number of descriptor generator components is required to guarantee an acceptable key-point dropping rate, a large number of components will result in increased resource usage. The optimal number of descriptor generator components ($N_D$) which leads to minimal key-point dropping rate is discussed in Section IV-A.

## IV. Results and Discussion

In this section, we first discuss the trade-off studies between accuracy and hardware complexity based on the following design parameters that was described in the previous sections (i.e. $N_T$, $B_C$, $B_T$, $B_{CS}$, and $N_D$). The design parameters that provides the best trade-offs will be adopted in our proposed architecture. The proposed rBRIEF descriptor extractor is implemented using Verilog hardware description language based on these design parameters and targeted on the Altera Arria V GX 5AGXFB3 device which is hosted on the Arria V GX FPGA Starter Kit. Finally, the implementation results are reported and compared with the state-of-the-art implementation in terms of performance, accuracy, and resource utilization.

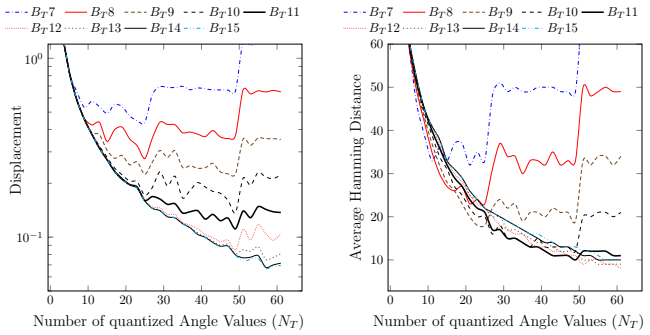
### A. Impact of Proposed Approximation Methods on Accuracy

In the *Angle* unit, an approximation is applied to calculate the centroid angle of a patch centered at a key-point. The centroid angle is discretized into $N_T$ possible values and the multiplication of discretized angle values are represented with $B_T$ bits. We have used *Displacement* and *Hamming distance* to evaluate the effects of our approximation methods for the proposed rBRIEF feature extractor. *Displacement* in rotated coordinates between the actual calculation and the approximate calculation is employed to quantitatively evaluate the accuracy of the approximation. The actual calculation employs double precision floating point that is costly and is not suitable for hardware implementation. The parameters that lead to the best design trade-offs are selected based on *Displacement* which measures the robustness of the approximation method to the precision loss due to angle discretization. *Hamming distance*, which is a metric that is often used in feature matching, is employed to validate the effect of the approximation method on the final output descriptors using the image dataset presented in Section IV-B. The *Displacement* is defined as follows:

$$Displacement = \sqrt{(rx - rx_{ac})^2 + (ry - ry_{ac})^2}, \quad (9)$$

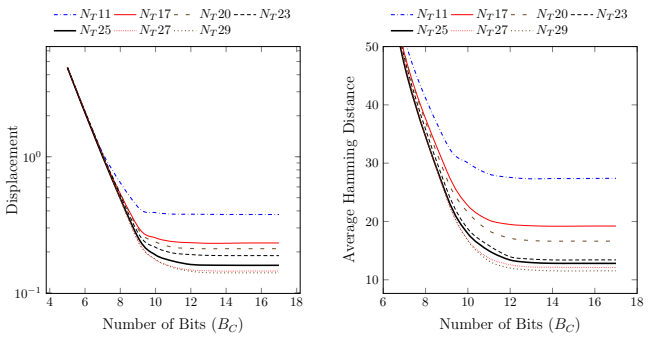
where ($rx_{ac}$, $ry_{ac}$) is the rotated coordinate obtained from the actual calculation while ($rx$, $ry$) is the corresponding coordinates that are computed using our approximated method in hardware. The image set "Graffiti" from "Affine Covariant Regions" Dataset [33] is employed in our investigations and the Harris-corner algorithm is used to detect a set of key-points.

Fig. 7 shows the accuracy evaluation of angle discretization with respect to $N_T$ in terms of (a) *Displacement* and (b)




(a) *Displacement* (b) *Hamming distance*

Fig. 7: The impact of rotation angle discretization on accuracy.




(a) *Displacement* (b) *Hamming distance*

Fig. 8: Investigating the number of bits for Centroid moments.

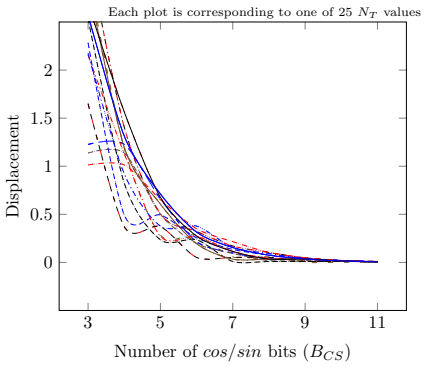



Fig. 9: Investigating the impact of number of bits for $cos/sin$'s values on accuracy.

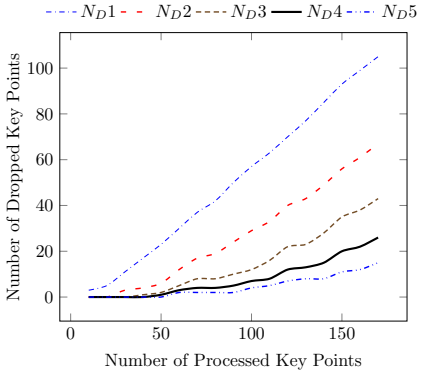



Fig. 10: Dropping rate for different number of replications of the descriptor generator components.

*Hamming distance*. $B_T\sharp n$ denotes $\sharp n$ bits that are used to represent the discretized angle values. It can be observed that *Displacement* is significantly reduced when $N_T$ increases to 25. *Hamming distance* has the same trend but presents small variation. Interestingly, increasing $N_T$ requires an increased number of bits ($B_T$) to represent the discretized angle's values so that the accuracy does not degrade. For example, if 9 bits are used to represent the discretized angle values (i.e. $B_T9$), the displacement progressively reduces till $N_T = 25$ but then exhibits an erratic increase in displacement when $N_T > 25$. However, if $B_T$ is set at 11 (i.e. $B_T11$), the displacement progressively decreases in the range $25 < N_T < 50$. Based on this investigation, we have chosen the design parameters $N_T = 25$ and $B_T = 11$ which provides the best trade-offs in accuracy and hardware complexity. Fig. 7b validates the selection with an acceptable loss in *Hamming distance*. Choosing larger design parameters yields a small gain in accuracy but incurs high hardware complexity.

The optimal number of bits ($B_C$) for representing centroid moments is also investigated. Fig. 8 shows (a) the *Displacement* and (b) the average *Hamming distance* of approximate calculation with respect to varying number of bits used for representing the absolute value of centroid moments (i.e. $m01, m10$). The displacement is also reported with different angle discretization options. $N_T\sharp n$ denotes that the angle is discretized with $N_T = \sharp n$ values. It can be observed that the average displacement sharply increases when $B_C$ reduces below 9. Interestingly, with the choice of $N_T = 25$ (i.e. $N_T25$) both *Displacement* and *Hamming distance* are further reduced when $B_C$ increases from 9 to 12 bits and the reduction starts to plateau at $B_C > 12$. Therefore, we have chosen $B_C = 12$ as the optimal bit-width for representing the absolute value of centroid moments in the proposed hardware implementation.

In addition, the optimal bit-width of $cos\theta, sin\theta$ values (i.e. $B_{CS}$) is investigated to further reduce the hardware complexity of the the *Rotator* unit. This study is based on the optimal discretization of the centroid angle $N_T = 25$ which we have determined earlier. Fig. 9 illustrates the average coordinate displacement with varying $B_{CS}$ with respect to each $N_T$. The displacement for each $N_T$ rapidly decreases as $B_{CS}$ increase to 7. The displacement of all the 25 cases is below 0.25 if $B_{CS} = 7$ bits is employed to represent the $cos\theta, sin\theta$ values. Further increase in $B_{CS}$ does not lead to any notable displacement reduction. Therefore, $B_{CS} = 7$ is selected for the proposed architecture.

In the *Generator* unit, multiple descriptor generator components are required to process the key-points in parallel. The number of component replications ($N_D$) is chosen with respect to the density of key-points (i.e. the number of detected key-points per image) and the acceptable rate of dropping key-points. The density is adjusted to a fixed number based on the corner scores of the Harris-corner detector. Fig. 10 shows the dropping rate with respect to key-point density for different number of component replications. $N_D\sharp n$ denotes


Fig. 11: Image dataset. From Top left: Bike, Boat, Graffiti, and UBC.

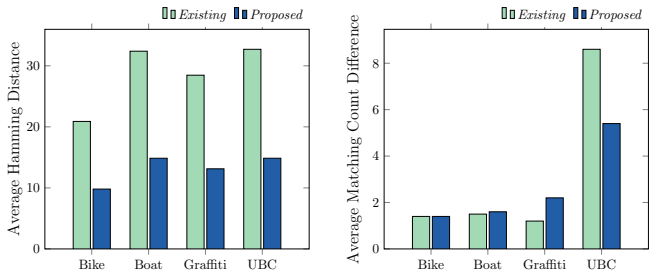
the replication of $\sharp n$ descriptor generator components. As can be seen, increasing the number of descriptor generator components from 1 to 4 achieves considerable reduction in dropping rate. The use of 4 descriptor generator components (i.e. $N_D4$) reduces the dropping rate to below 5 key-points for the density of 100 key-points/image. Therefore, in the implementation of the proposed rBRIEF descriptor, 4 descriptor generator components are used in the *Generator* unit.

### *B. Hardware Implementation Results*

This sub-section reports the FPGA implementation results of the proposed rBRIEF feature extractor based on the design parameters described in the previous sub-section. Table I summarizes the design parameters and compares the hardware utilization of the proposed implementation (denoted as *Proposed*) with the implementation in [19] (denoted as *Existing*). To obtain a fair comparison, both implementations targets the Arria V GX 5AGXFB3 device and employ the same number of descriptor generator components ($N_D$). We only report the hardware utilization of sub-modules in the existing implementation that have a corresponding function in Proposed, which excludes the detector and pyramid implementation. It can be observed that the proposed implementation consumes significantly reduced number of Logic Elements (LEs) and Adaptive Logic Modules (ALMs) compared to the existing implementation. The LE and ALM utilization

TABLE I: Comparison of Hardware Utilization

| | Existing [19] | Proposed |
|---|---|---|
| $N_D$ | 4 | 4 |
| $N_T$ | 16 | 25 |
| $B_C$ | 24 | 12 |
| $B_T$ | 16 | 11 |
| $B_{CS}$ | 9 | 7 |
| *Logic Elements* | 19720 | 12523 |
| *ALMs* | 21436 | 10019 |
| *DSP* | 92 | 0 |
| *Memory Bits* | 76000 | 117739 |
| $f_{max}$ (MHz) | 150 | 177 |




(a) *Hamming distance* (b) *Matching Count*

Fig. 12: Accuracy comparison between the existing and the proposed methods.

of *Proposed* is only 63.5% and 46.7%, respectively, of the existing architecture. The memory bits usage of *Proposed* is larger than that of *Existing*. However, the memory bits required in both implementations consume only a fraction (i.e. about 0.6%) of the total FPGA space. Interestingly, the proposed implementation avoids the use of power-hungry DSP blocks due to utilization of the constant multiplier blocks. In contrast, the existing implementation employs 92 DSP blocks. Furthermore, the hardware efficient strategies adopted by *Proposed* have resulted in a higher maximum clock frequency. In particular, *Proposed* achieves a higher operating frequency of 177 MHz compared to the clock frequency of 150 MHz for *Existing* on the same target FPGA platform.

In order to evaluate the accuracy of both hardware implementations, four image sets depicted in Fig. 11: Bike, Boat, Graffiti, and University of British Colombia (UBC), from the image dataset in [33] are employed. Fig. 11 also illustrates the key-points and their Hamming distance between the accurate rBRIEF descriptors and that of the proposed implementation. These results are obtained using Matlab to simulate the actual hardware computations based on the parameters reported in Table I.

The execution of both implementations based on the design parameters in Table I are simulated to evaluate the accuracy. The average Hamming distance of the 256 descriptors bits between each implementation and the actual computation which uses double precision floating points is employed as the accuracy metric. A lower average Hamming distance indicates that the corresponding implementation produces more accurate results.

Fig. 12 compares the accuracy between the existing and the proposed implementations with respect to the accurate calculation of rBRIEF descriptor in terms of (a) Hamming distance and (b) Matching Count. We choose a Hamming distance threshold of 60 for key-point matching. Due to the use of optimal design parameters that have been determined

TABLE II: Dynamic Power and Hardware Resource Utilization

| | Centroid | Angle | Rotator | Generator | Total |
|---|---|---|---|---|---|
| *LEs* | 1041 | 1233 | 8727 | 1502 | 12503 |
| *Registers* | 1212 | 1589 | 5817 | 1956 | 10574 |
| *Power (mW)* | 7.33 | 15.23 | 19.84 | 413.75 | 456.15 |

TABLE III: Performance Comparison

| | Clock Freq. (GHz) | Throughput (MPix/s) | Latency (ms) | Frame Rate (FPS) | Resolution |
|---|---|---|---|---|---|
| *Intel-i5* [19] | 4.3 | 28 | 74.1 | 13.5 | 1920×1080 |
| *GPU* [19] | 1.05 | 40 | 52.1 | 19.2 | 1920×1080 |
| *FPGA1* [19] | 0.150 | 150 | 13.8 | 72.3 | 1920×1080 |
| *FPGA2* [18] | 0.203 | - | 14.8 | 67 | 640×480 |
| *Proposed* | 0.175 | 175 | 11.9 | 84.4 | 1920×1080 |

from our rigorous design exploration, *Proposed* achieves about 50% reduced Hamming distance compared to *Existing* for all 4 image sets considered. For example, the average Hamming distance of the descriptor bits for *Proposed* is lower than 15 bits compared to the Hamming distance of the *Existing* implementation with over 30 bits for the image sets 'Boat' and 'UBC'. In Fig. 12b, due to image modification effect on rBRIEF matching performance the approximations of the proposed and the existing implementation have almost the same matching performance in 'Bike' and'Boat' datasets. *Existing* is slightly better in 'Graffiti' but suffers from considerable degradation in 'UBC' compared to *Proposed*.

A post-place-and-route simulation in ModelSim was used to obtain accurate signal transitions for analyzing the power consumption of the FPGA implementation using the PowerPlay Analyzer tool. Table II reports the dynamic power dissipation and hardware utilization of each hardware unit with an operating frequency of 175 MHz. The *Rotator* unit consumes the largest number of LEs and registers accounting for 70% and 55% respectively of the entire rBRIEF feature extractor to perform angle rotation for 512 samples. Due to the use of window buffers to store 37×37 key-point patches, which is replicated 4 times, the *Generator* unit consume 90.7% of the total dynamic power consumption of the rBRIEF extractor. The *Centroid* and the *Angle* units consume a small amount of resource usage due to the hardware efficient moving summation approach adopted for calculating the centroid moments and the approximate discretization approach for determining the angle. In particular, the *Centroid* and the *Angle* units consume only 8.3% and 9.8% of total LEs, and 11.5% and 15% total registers, respectively. In addition, these two units only dissipate a small amount of dynamic power, i.e. 1.6% and 3.3% respectively of the total power consumption of the rBRIEF extractor.

Finally, Table III compares the performance between the proposed implementation and the existing implementations. The performances of the FPGA implementations, denoted as *FPGA1* and *FPGA2*, are obtained from [18] and [19] respectively. The performances of the implementations on CPU and GPU platforms, denoted as *Intel-i5* and *GPU* respectively, are taken from [19]. The *Proposed* implementation offers the lowest latency ($<$ 12ms which enables real-time processing at 60fps) on video frames of 1920×1080 resolution at a clock operating frequency of 175 MHz.

## V. Conclusion

This paper presents a hardware-efficient pipelined architecture for rBRIEF feature extractor which is capable of processing a pixel stream at high throughput without the need of any image frame buffers. To lower the hardware complexity, we proposed a moving summation strategy for calculating the key-point patch moments. Approximate computation strategies have been devised to determine the patch's orientation angle and rotating the BRIEF point-pairs. In addition, multiplier-less circuitries are introduced to avoid the usage of costly multipliers throughout the pipelined architecture. Extensive design exploration have been undertaken to determine the design parameters that lead to the optimal accuracy and hardware complexity trade-offs. The proposed implementation significantly reduced the LEs by 62.42% while resulting in an increased accuracy by 50% when compared to the state-of-the-art implementation. The proposed architecture is able to process high-resolution (1920×1080) images at 60 fps while consuming only 239.34 mW power. In our future work, we plan to integrate the proposed feature extraction accelerator in a visual SLAM system for autonomous navigation of UAVs.

## Acknowledgment

This research project is partially funded by the National Research Foundation Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme.

## References

[1] J. Hartmann, J. H. Klüssendorff, and E. Maehle, "A comparison of feature descriptors for visual SLAM," in *European Conference on Mobile Robots (ECMR)*, Sept 2013, pp. 56–61.

[2] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[3] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," in *European Conference on Computer Vision (ECCV)*. Springer, 2006, pp. 404–417.

[4] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," in *European Conference on Computer Vision (ECCV)*. Springer, 2010, pp. 778–792.

[5] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *International Conference on Computer Vision (ICCV)*, Nov. 2011, pp. 2564–2571.

[6] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary Robust invariant scalable keypoints," in *International Conference on Computer Vision (ICCV)*, Nov. 2011, pp. 2548–2555.

[7] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast Retina Keypoint," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2012, pp. 510–517.

[8] B. Cowan, N. Imanberdiyev, C. Fu, Y. Dong, and E. Kayacan, "A performance evaluation of detectors and descriptors for UAV visual tracking," in *14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Nov. 2016, pp. 1–6.

[9] J. Heinly, E. Dunn, and J.-M. Frahm, "Comparative Evaluation of Binary Features," in *European Conference on Computer Vision (ECCV)*. Springer, 2012, pp. 759–773.

[10] D. Mukherjee, Q. M. Jonathan Wu, and G. Wang, "A comparative experimental study of image feature detectors and descriptors," *Machine Vision and Applications*, vol. 26, no. 4, pp. 443–466, May 2015.

[11] S. K. Lam, T. C. Lim, M. Wu, B. Cao, and B. A. Jasani, "Area-Time Efficient FAST Corner Detector using Data-path Transposition," *IEEE Transactions on Circuits and Systems II: Express Briefs*, pp. 1–1, 2017.

[12] S. K. Lam, R. K. Bijarniya, and M. Wu, "Lowering dynamic power in stream-based harris corner detection architecture," in *International Conference on Field Programmable Technology (ICFPT)*, Dec. 2017, pp. 176–182.

[13] O. Ulusel, C. Picardo, C. B. Harris, S. Reda, and R. I. Bahar, "Hardware acceleration of feature detection and description algorithms on low-power embedded platforms," in *26th International Conference on Field Programmable Logic and Applications (FPL)*, Aug. 2016.

[14] F. Brenot, J. Piat, and P. Fillatreau, "FPGA Based Hardware Acceleration of a BRIEF Correlator Module for a Monocular SLAM Application," in *Int. Conf. on Distributed Smart Camera (ICDSC)*, 2016, pp. 184–189.

[15] R. de Lima, J. Martinez-Carranza, A. Morales-Reyes, and R. Cumplido, "Accelerating the construction of BRIEF descriptors using an FPGA-based architecture," in *International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, Dec. 2015, pp. 1–6.

[16] H. Heo, J. y. Lee, K. y. Lee, and C. h. Lee, "FPGA based Implementation of FAST and BRIEF algorithm for object recognition," in *IEEE International Conference of IEEE Region 10 (TENCON)*, Oct. 2013.

[17] R. Sun, P. Liu, J. Wang, and Z. Zhou, "A low latency feature extraction accelerator with reduced internal memory," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2017.

[18] W. Fang, Y. Zhang, B. Yu, and S. Liu, "FPGA-based ORB Feature Extraction for Real-Time Visual SLAM," in *International Conference on Field Programmable Technology (ICFPT)*, Dec. 2017, pp. 275–278.

[19] J. Weberruss, L. Kleeman, D. Boland, and T. Drummond, "FPGA Acceleration of Multilevel ORB Feature Extraction for Computer Vision," in *27th International Conference on Field Programmable Logic and Applications (FPL)*, Sep. 2017.

[20] W. Zhu, L. Liu, G. Jiang, S. Yin, and S. Wei, "A 135-frames/s 1080p 87.5-mW Binary-Descriptor-Based Image Feature Extraction Accelerator," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 8, pp. 1532–1543, 2016.

[21] P. L. Rosin, "Measuring Corner Properties," *Computer Vision and Image Understanding*, vol. 73, no. 2, pp. 291–307, 1999.

[22] F. D. V. R. Oliveira, J. G. R. C. Gomes, J. Fernández-Berni, R. Carmona-Galán, R. del Río, and A. Rodríguez-Vázquez, "Gaussian Pyramid: Comparative Analysis of Hardware Architectures," *IEEE Trans. on Cir. and Sys. I: Regular Papers*, vol. 64, no. 9, pp. 2308–2321, Sep. 2017.

[23] G. Mishra, Y. L. Aung, M. Wu, S. K. Lam, and T. Srikanthan, "Real-Time Image Resizing Hardware Accelerator for Object Detection Algorithms," in *International Symposium on Electronic System Design (ISED)*, Dec. 2013, pp. 98–102.

[24] B. A. Jasani, S. K. Lam, P. K. Meher, and M. Wu, "Threshold-Guided Design and Optimization for Harris Corner Detector Architecture," *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[25] S.-K. Lam, T. C. Lim, M. Wu, B. Cao, and B. A. Jasani, "Data-path unrolling with logic folding for area-time-efficient FPGA-based FAST corner detector," *Journal of Real-Time Image Processing*, Oct 2017.

[26] T. H. Pham, V. A. Prasad, and A. S. Madhukumar, "A Hardware-Efficient Synchronization in L-DACS1 for Aeronautical Communications," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 5, pp. 924–932, May 2018.

[27] Y. Voronenko and M. Püschel, "Multiplierless Multiple Constant Multiplication," *ACM Transactions on Algorithms*, vol. 3, no. 2, 2007.

[28] T. H. Pham, S. A. Fahmy, and I. V. McLoughlin, "Low-Power Correlation for IEEE 802.16 OFDM Synchronization on FPGA," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 8, pp. 1549–1553, Aug 2013.

[29] ——, "Efficient Integer Frequency Offset Estimation Architecture for Enhanced OFDM Synchronization," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 4, pp. 1412–1420, April 2016.

[30] S. Suchitra, S. K. Lam, and T. Srikanthan, "Novel schemes for high-throughput image rotation," in *38th Asilomar Conference on Signals, Systems and Computers (ACSSC)*, vol. 2, Nov. 2004, pp. 1884–1888.

[31] S. Suchitra, S. k. Lam, C. T. Clarke, and T. Srikanthan, "Accelerating rotation of high-resolution images," *IEE Proceedings - Vision, Image and Signal Processing*, vol. 153, no. 6, pp. 815–824, Dec. 2006.

[32] A. Amaricai, C. E. Gavriliu, and O. Boncalo, "An FPGA sliding window-based architecture harris corner detector," in *International Conference on Field Programmable Logic and Applications (FPL)*, Sep. 2014, pp. 1–4.

[33] *Affine covariant regions datasets*, Oxford Visual Geometry Group, 2004. [Online]. Available: http://www.robots.ox.ac.uk/ vgg/data/data-aff.html


**Thinh Hung Pham** (M'14) received his B.S. degree in Electrical Electronic Engineering at Ho Chi Minh City, University of Technology, Viet Nam, and MSc. degree in Embedded Systems Engineering at the University of Leeds, UK, in 2007 and 2010, respectively. He completed his Ph.D. in the joint Nanyang Technological University and Technische Universitat Munchen program in Singapore in 2015. From 2015 to 2016, he was a Lecturer at Ho Chi Minh City, University of Technology. He is currently a Research Fellow at Nanyang Technological University.

**Phong Tran** is pursuing Bachelor of Engineering in Electrical and Electronics Engineering at Ho Chi Minh City University of Technology, Viet Nam. Since 2018, he has been an intern at School of Computer Science and Engineering (SCSE), Nanyang Technological University, Singapore. He has been working on low-power and hardware efficient computer vision algorithms based on FPGA platforms. His research interest currently focuses on computer vision, machine learning, and robotics.

**Siew-Kei Lam** (M'03) received his BASc, MEng and PhD from School of Computer Science and Engineering (SCSE), Nanyang Technological University, Singapore. He is currently an Assistant Professor in SCSE and his research investigates methods for realizing custom computing solutions in embedded systems. His current projects include developing architecture-aware algorithms for vision-enabled sensing, and design methodologies for secure and reliable embedded systems.