A Low-Complexity Hyperspectral Anomaly Detection Algorithm and Its FPGA Implementation

Jie Lei , *Member, IEEE*, Geng Yang, Weiying Xie , *Member, IEEE*, Yunsong Li , *Member, IEEE*, and Xiuping Jia , *Senior Member, IEEE*

Abstract—On-board real-time anomaly detection has always been a challenging task in hyperspectral imaging analysis as it requires low computational complexity. Most of the existing anomaly detection algorithms inevitably trade off intensive computational complexity for high detection accuracy. This article presents a fast spectral–spatial anomaly detection algorithm with low complexity in hyperspectral images (HSIs) using morphological reconstruction and a simplified guided filter (Fast-MGD). Since the simple filtering techniques are applied, it is therefore feasible to achieve a field programmable gate array (FPGA)-based hardware implementation. More precisely, an effective deeply pipelined acceleration scheme is developed adopting high-level synthesis to support HSIs that are acquired over different scenes with different sizes and spectral bands. Experimental results show strong advantages of the proposed FPGA-based Fast-MGD in processing speed and resource consumption, while a high detection accuracy is remained. Its applicability in on-board real-time processing is demonstrated and verifie.

Index Terms—Anomaly detection, guided filtering, high-level synthesis (HLS), hyperspectral image (HSI), morphological reconstruction, real-time implementation.

I. INTRODUCTION

OWING to the richness and abundance in spectral–spatial information, hyperspectral images (HSIs) acquired by hyperspectral imaging have been widely used in various applications including classification [1], target or anomaly detection [2], etc. Benefiting from the fact that anomaly detection neither requires any prior information nor relies on the complex preprocessing like atmospheric and radiometric correction, hyperspectral anomaly detection shows its high applicability for real-time processing on satellites. It is expected as an effective cutting-edge technology for military and civilian tasks, such as precision agriculture [3] and civilian search–rescue operations [4].

Conducting real-time anomaly detection from satellites has always been a challenging task. On-ground computing platforms of high performance involving multicore processors and GPUs generally display their inabilities of being applied on satellites. In contrast, field programmable gate array (FPGA) is a better alternative to be adopted in the harsh environment of outer space for three reasons. First, FPGA can provide much more competent levels of performance while sustaining lower power consumption compared with GPUs [5]. Second, FPGA offers high-level flexibility due to the inherent ability to change functionality through partial or full reconfiguration. Third, the increasing characteristics of ionizing-radiation tolerance make FPGA the most extensive solution for on-board processing at earth observation satellites. Unfortunately, the hardware implementations on FPGA for on-board anomaly detection are few so far.

Aiming at solving problems of on-board real-time processing of hyperspectral anomaly detection, the motivation of our work is to develop an algorithm and hardware structure for both high accuracy and low complexity. In this article, we propose a fast spectral–spatial anomaly detection algorithm based on morphological reconstruction and the simplified guided filtering (Fast-MGD). The proposed Fast-MGD is a segmentation-based method that can sequentially actuate operations of average fusion, feature location, feature extraction, and feature clustering. The whole process can ensure high detection accuracy while avoiding complex matrix operations. Through studying the parallelism of Fast-MGD, FPGA-based hardware implementation is, therefore, proposed. In particular, a relatively mature high-level synthesis (HLS) [6] is especially utilized in obtaining better portability, higher flexibility, and shorter development period against the conventional register transfer level (RTL)-based

design method, contributing a minor proportion to the above method's overall superiority.

The key contributions of this article can be recapitulated as follows.

1) To satisfy low complexity while maintaining detection performance, a simple but effective combination of the advantageous morphological reconstruction and self-guided filtering is designed. Especially, self-guided filtering is further optimized by removing redundant mean operations on the intermediate coefficients to reduce memory requirements.

2) A novel and effective deep pipelined architecture is presented to accelerate our algorithm on FPGA by adopting HLS, which can be easily reconstructed by modifying several parameters for different scenes of various sizes featured with different numbers of spectral bands, offering strong feasibility for on-board real-time processing.

3) An efficient parametric configurable sharing architecture is designed for both morphological reconstruction and self-guided filtering.

The remainder of this article is organized as follows. Section II conducts an overall review of related work. Section III elaborates the principle of morphological filtering and guided filtering as well as the difficulties of implementing these two techniques. Section IV introduces the proposed Fast-MGD. Section V presents the FPGA implementation of Fast-MGD. Section VI provides experimental assessment on detection accuracy and processing performance of the proposed Fast-MGD and its hardware implementation on FPGA. Section VII concludes this article.

II. REVIEW OF RELATED WORK

Various anomaly detection algorithms have been proposed in recent decades. The Reed–Xiaoli (RX) algorithm [7] proposed by Reed and Yu is regarded as a milestone in hyperspectral anomaly detection. Under the assumption that the background follows a single Gaussian normal distribution, the RX detector detects anomalies by calculating the Mahalanobis distance between the pixel to be tested and the background obtained through estimating the sample mean and covariance matrix. Various variants of RX algorithms have been investigated to realize better detection performance [8], [9]. For instance, a concentric dual sliding window around each image pixel is adopted to estimate the background in the local RX (LRX) detector [10] proposed by Molero *et al.* However, the RX-based algorithm does not perform satisfactorily in complicated scenarios due to the linear assumption of the background.

Despite a kernel RX proposed by Kwon and Nasrabadi [11] further exploited the nonlinear characteristics of HSIs and mapped the spectral information of the original hyperspectral data to the high-dimensional feature space, yet the performance of this method is not, therefore, improved noticeably, in which the required computation of high-order Gram matrix and its inverse matrix is far beyond affordability.

In addition, diverse methods of non-RX algorithms in terms of representation-based, projection-based, and

segmentation-based have been proposed [12]–[16]. For instance, collaborative-representation-based detector (CRD) [13] employs the concept that each pixel in the background can be approximately represented by its spatial neighborhoods whereas anomalies cannot be. The collaboration of representation is reinforced by L2-norm minimization of the representation weight vector, in which a Euclidean distance-weighted regularization matrix is included to adjust the contribution of each neighboring pixel. However, the configuration of bi-window size in the algorithm is characterized with high sensitivity, posing detrimental impacts on its detection performance and processing speed.

To compensate the inadequate consideration of the spatial correlation in the above algorithms, a tensor-based method was thus proposed in [15] to describe the spectral information and spatial information equivalently. In recent decades, morphological filtering has been widely used as a novel HSI spatial descriptor for spatial feature extraction [16]–[18]. Based on morphological attribute filtering and domain transform recursive filtering, Kang *et al.* [16] proposed a segmentation-based detector (AED) to integrate spatial information throughout the feature extraction process. The area-based attribute filtering is developed to extract anomalous candidates with specific area properties, yet the gradient reversal cannot be thoroughly avoided by the domain transform filtering conducted in the postprocessing operation. To tackle with the aforementioned problems, Xie *et al.* [18] proposed a more efficient strategy based on structure tensor and guided filtering (STGF), but the preprocessing operation of structure tensor-based band selection is relatively complicated. It is worth emphasizing that the global tree-representation-based strategy for morphological attribute filtering in these methods is confronted with lots of obstacles in hardware implementation.

In recent years, theories of deep neural network attack detection have drawn much attention worldwide [19]–[23]. In [20], the adaptive weight deep belief network (DBN) with an autoencoder structure is used to acquire high-level features and reconstruction errors. Xie *et al.* [22] proposed an autoencoder and adversarial-learning-based semisupervised background estimation model, which leverages the training spectral vectors obtained by a specific searching method to learn the background spectral characteristics. In [23], the generative adversarial network is applied and developed to estimate the background by means of two effective constraints (a continuity representation constraint in the latent space and a discriminator-based authenticity constraint). However, these methods need to recover and store the reconstructed image by the same size as the original HSI, thereby imposing enormous pressure on storage. A spectral–spatial feature extraction method [21] was proposed by Lei *et al.* through performing a linear combination of joint spectral and spatial detection results. However, the designed DBN extracting reduced number out of spectral features cannot perform satisfactorily in all HSIs due to the shortage in training samples.

Obviously, most of the aforementioned anomaly detectors improve their detection accuracy by raising the intensity of computation complexity; in contrast, lowering the computational cost is indeed taken into account by certain researchers, which has so far met with limited success. A huge amount of complicated

computational operations regarding covariance matrix, inverse matrix, and eigenvalue decomposition are still unavoidable, posing negative encumbrance for real-time onboard detection.

Furthermore, on-board anomaly detection has stringent requirements on size, weight, power consumption, and radiation-hardened of the computing platform, to which FPGA stands out as a mainstream criterion for on-board hyperspectral data processing among the existing high-performance computation devices. For instance, Yang *et al.* [5] used a streaming background statistics approach for optimizing the constrained energy minimization and RX on FPGA. In [24], a lightweight network based on the quantization and the structured pruning (P-Q-AD) was proposed by utilizing the potential relationship between detection accuracy and throughput. However, most of the cutting-edge algorithms are not sufficiently implemented on FPGAs due to their intrinsic computational complexity and low parallelism.

To summarize, developing hyperspectral satellite remote sensing detection via algorithm and hardware implementation becomes an urgent must that focus on solving the key problem about opportune acquisition of abnormal targets.

III. BACKGROUND

A. Morphological Filtering

Morphological filtering is defined between a group of image points called object and kernel structuring element (SE) [25], which can be used as a powerful tool for extracting useful image components representing region shapes. Morphological operations are developed and further extended based on two basic operations, erosion, and dilation.

As one of the ingenious morphological operations, morphological reconstruction requires two input images, which consist of a marker image functioning as the starting point for transformation and a mask image constraining the transformation.

In this work, we consider rectangular flat SE **b** of size $r \times r$. Let two grayscale images **f** and **g** denote the marker image and the mask image, respectively, to which **f** is pointwise less than or equal to **g**. The morphological dilation reconstruction (R^D) of **g** from **f** is denoted by

$$R_{\mathbf{g}}^{D}(\mathbf{f}) = D_{\mathbf{g}}^{(k)}(\mathbf{f}) \tag{1}$$

where $D_{\mathbf{g}}^{(1)}(\mathbf{f}) = (\mathbf{f} \oplus \mathbf{b}) \wedge \mathbf{g}$, $D_{\mathbf{g}}^{(n)}(\mathbf{f}) = D_{\mathbf{g}}^{(1)}(D_{\mathbf{g}}^{(n-1)}(\mathbf{f}))$ for $2 \le n \le k$, $k, n \in N^+$ satisfies $D_{\mathbf{g}}^{(k)}(\mathbf{f}) = D_{\mathbf{g}}^{(k-1)}(\mathbf{f})$. The symbol $\oplus$ stands for the morphological dilation, and $\wedge$ denotes the pointwise minimum at each pixel of two images.

Similarly, if **f** is pointwise greater than or equal to **g**, the morphological erosion reconstruction (R^E) of **g** from **f** is expressed as

$$R_{\mathbf{g}}^{E}(\mathbf{f}) = E_{\mathbf{g}}^{(k)}(\mathbf{f}) \tag{2}$$

where $E_{\mathbf{g}}^{(1)}(\mathbf{f}) = (\mathbf{f} \ominus \mathbf{b}) \vee \mathbf{g}$, $E_{\mathbf{g}}^{(n)}(\mathbf{f}) = E_{\mathbf{g}}^{(1)}(E_{\mathbf{g}}^{(n-1)}(\mathbf{f}))$ for $2 \le n \le k$, $k, n \in N^+$ satisfies $E_{\mathbf{g}}^{(k)}(\mathbf{f}) = E_{\mathbf{g}}^{(k-1)}(\mathbf{f})$. The symbol $\ominus$ stands for the morphological erosion, and $\vee$ denotes the pointwise maximum at each pixel of two images.

The aforementioned operations can be combined into two more complex operations used in our work following certain sequences: opening and closing reconstructions, which show their better performance in extracting spatial features. The opening reconstruction can be conducted through a series of successive erosions followed by a dilation reconstruction. By duality, the result of the dilation of the input image followed by an erosion reconstruction is called closing reconstruction.

The erosion/dilation performed first in the opening/closing reconstruction is a window-based operation, which aims to find minimum/maximum among the pixels belonging to the SE that is characterized by its size and shape. A common method requires $l - 1$ comparisons if the SE is a local window containing l pixels, which results in redundancy of comparison, thereby exposing its extremely low computational efficiency. Van Herk [27] and Gil and Werman [28] proposed a recursive algorithm (HGW) whose complexity is size independent of the SE. However, we emphasize that the optimal comparison number does not always guarantee the best overall performance, to which latency and memory also matters significantly. The parallel architecture [29] proposed by Mukherjee *et al.* for rectangular elements reduces the processing time by simultaneously processing pixels within adjacent windows of SEs. The xfOpenCV library of Xilinx [30] provides dilation/erosion kernel being capable of processing SEs with various shapes, achieving good performance in terms of memory utilization and processing time. In our work, the data reuse structure based on SE decomposition is further designed, which not only reduces the number of comparisons, but also exhibits excellent performance in terms of latency and memory through keeping and reusing partial results generated during the computation process.

The subsequent dilation/erosion reconstruction in the opening/closing reconstruction can be essentially explained as constrained dilation and erosion operations. Vincent [31] proposed four effective strategies in 1993 involving standard technique, sequential reconstruction (SR), reconstruction using a queue of pixels, and hybrid reconstruction. Furthermore, a single pass reconstruction algorithm was proposed by Robinson and Whelan [32], i.e., downhill filtering strategy. Anacona-Mosquera *et al.* [33], [34] proposed effective hardware architectures based on SR and hybrid reconstruction, respectively. In [35], the morphological coprocessor unit is proposed for reconstruction operations based on the standard technique by using large SE pipelines and interconnective architecture of the pipelines. We choose the simplest standard technique that can be directly obtained according to the definitions of (1) and (2). However, this method is inapplicable for hardware implementation due to uncertainty of the iterations under idempotence [17]. Specifically, for the certain missions in hyperspectral anomaly detection, we adopt fixed iterations through conducting a large number of experiments to avoid the above problem.

B. Guided Filtering

The guided filter [36] computes the filtering output by considering the content of a guidance image. Here, we will briefly describe the rationale of guided filters.

Algorithm 1: Pseudocode of the Guided Filter.

Input: p : input image **O** : the guidance image
Parameters: r, ϵ
  $\text{mean}_{\mathbf{p}} = f_{\text{mean}}(\mathbf{p})$ (1)
  $\text{mean}_{\mathbf{O}} = f_{\text{mean}}(\mathbf{O})$ (2)
  $\text{Corr}_{\mathbf{O}} = f_{\text{mean}}(\mathbf{O}. * \mathbf{O})$ (3)
  $\text{Corr}_{\mathbf{pO}} = f_{\text{mean}}(\mathbf{p}. * \mathbf{O})$ (4)
  $\text{Var}_{\mathbf{O}} = \text{Corr}_{\mathbf{O}} - \text{mean}_{\mathbf{O}}. * \text{mean}_{\mathbf{O}}$ (5)
  $\text{Cov}_{\mathbf{pO}} = \text{Corr}_{\mathbf{pO}} - \text{mean}_{\mathbf{p}}. * \text{mean}_{\mathbf{O}}$ (6)
  $a = \text{Cov}_{\mathbf{pO}}./(\text{Var}_{\mathbf{O}} + \epsilon)$ (7)
  $b = \text{mean}_{\mathbf{p}} - a. * \text{mean}_{\mathbf{O}}$ (8)
  $\text{mean}_a = f_{\text{mean}}(a)$ (9)
  $\text{mean}_b = f_{\text{mean}}(b)$ (10)
  $\mathbf{q} = \text{mean}_a. * \mathbf{O} + \text{mean}_b$ (11)
Output: q

Fig. 1.   Mean filtering processing in [26].

Assuming that the output image **q** is a linear transformation of the guidance image **O** in a local window ω_j centered at the pixel j:

$$\mathbf{q_i} = a_j\mathbf{O_i} + b_j \quad \forall i \in \omega_j \tag{3}$$

where (a_j, b_j) are the linear coefficients and ω_j is a square window of a radius r. To determine the linear coefficients (a_j, b_j), we need to seek a solution that minimizes the difference between the input image **p** and the output image **q** while maintaining the linear model (3). Specifically, we minimize the following cost function in the window ω_j:

$$E(a_j, b_j) = \sum_{i \in \omega_j}((a_j\mathbf{O_i} + b_j - \mathbf{p_i})^2 + \epsilon a_j^2) \tag{4}$$

where ϵ is a regularization parameter penalizing large a_j. (4) is the linear ridge regression model, and its solution is given by

$$a_j = \frac{\frac{1}{|\omega|}\sum_{i \in \omega_j}\mathbf{O_i p_i} - \mu_j\overline{\mathbf{p_j}}}{\sigma_j^2 + \epsilon} \tag{5}$$

$$b_j = \overline{\mathbf{p_j}} - a_j\mu_j \tag{6}$$

where μ_j and σ_j^2 are the mean and variance of **O** in ω_j, and $|\omega|$ is the number of pixels in ω_j. $\overline{\mathbf{p_j}} = \frac{1}{|\omega|}\sum_{i \in \omega_j}\mathbf{p_i}$ is the mean of **p** in ω_j.

After computing (a_j, b_j) for all window ω_j in the image, we compute the filtering output by

$$\mathbf{q_i} = \overline{a_i}\mathbf{O_i} + \overline{b_i} \tag{7}$$

where $\overline{a_i} = \frac{1}{|\omega|}\sum_{j \in \omega_i}a_j$ and $\overline{b_i} = \frac{1}{|\omega|}\sum_{j \in \omega_i}b_j$ are the average coefficients of all windows overlapping i. We compute the filter output **q** from its definition (5)–(7) according to Algorithm 1.

In addition to the edge-preserving property without gradient reversal artifacts, guided filtering naturally has a fast and non-approximate linear time algorithm that can be decomposed into a series of mean filters (f_{mean} in Algorithm 1) with windows radius r.

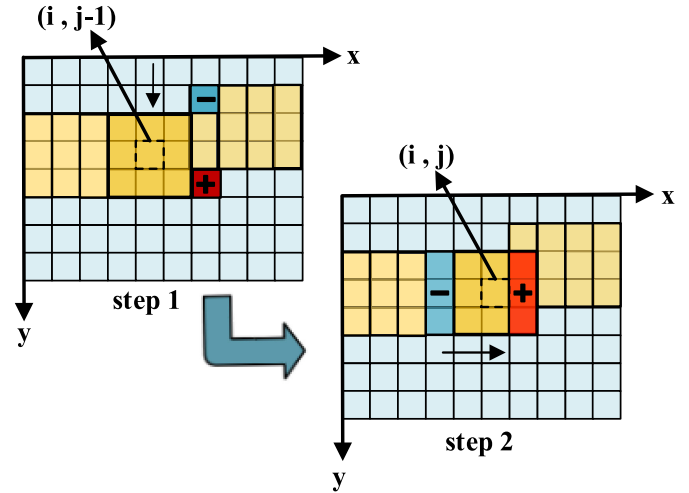It is obvious to note that the mean filter is also a window-based operation that is similar to the aforementioned erosion/dilation in the previous subsection. At present, one of the widely used techniques utilizes a custom mean filter derived from the approach in [26], [37], and [38], of whose main idea lies in maintaining a sum for each column in the image to be filtered. The detailed process of mean filtering is shown in Fig. 1. In specific, through subtracting its topmost old pixel and adding the bottom new pixel, the column sum is updated first. Then, through subtracting its leftmost column sum and adding the updated column sum, the window, therefore, moves to the right and its sum is updated. In our work, we further optimize the above architecture by studying the commonness between erosion/dilation and mean filter and design a general frame, which can be available for erosion/dilation and mean filter simultaneously.

According to Algorithm 1, it can be observed that there is strong data dependence in guided filtering, which limits the improvement of processing speed. Consequently, we design a high-throughput hardware architecture that eliminates data dependence, imposes reasonable simplification and segmentation for matrix operations.

IV. PROPOSED APPROACH

In this section, the proposed algorithm is described in detail. As shown in Fig. 2, the proposed Fast-MGD approach consists of four steps. First, average fusion is adopted to reduce dimension of the HSIs. Second, morphological opening and closing reconstruction are utilized for feature location. Third, a differential operation is conducted to extract anomalies in the HSIs. Finally, the final detection map is, therefore, obtained by means of the simplified self-guided filtering for feature clustering.

A. Average Fusion

The high dimensionality of the original HSIs will undoubtedly and inevitably incur extremely high computational complexity, throughout which massive storage capability is, therefore, required. The map of correlation coefficients of spectral bands in Fig. 3(a) shows that the adjacent bands are usually highly
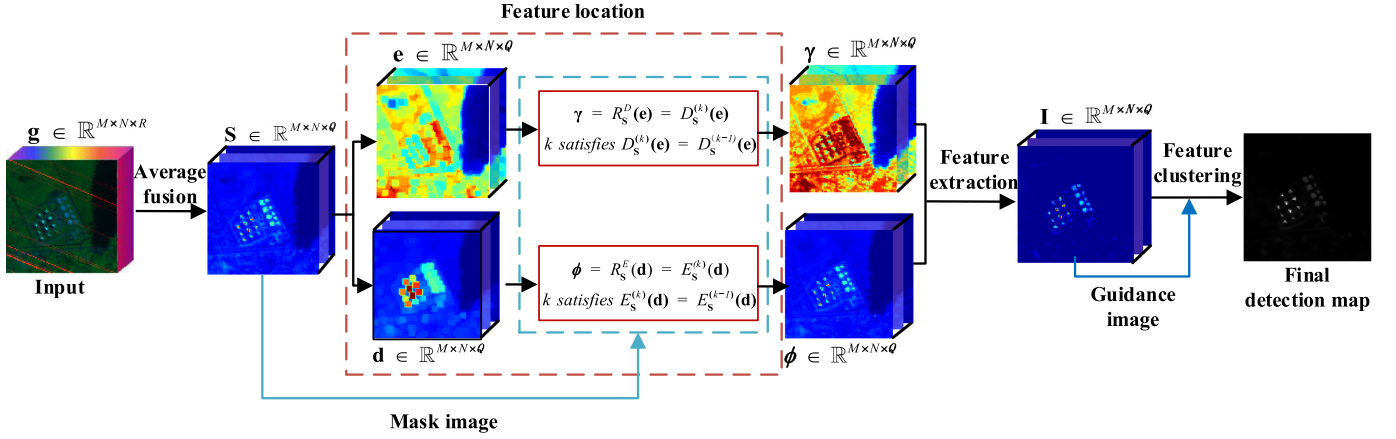
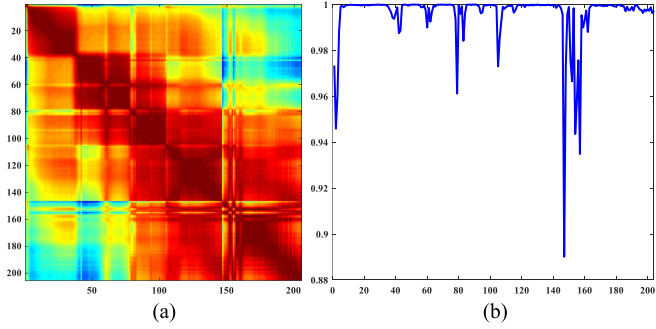Fig. 2. Schematic of the proposed Fast-MGD anomaly detection in HSI.

Fig. 3. (a) Maps of correlation coefficients of spectral bands. (b) Curve of correlation coefficients of adjacent spectral bands.

Fig. 4. Visualization of intermediate results on the San Diego dataset (using $r_a = 3$ and $k = 20$). (a) HSI. (b) First fused image $\mathbf{S}^1$. (c) and (d) are the intermediate results of the opening reconstruction. (e) and (f) are the intermediate results of the closing reconstruction. (g) and (h) are the result of $|\boldsymbol{\phi}^1 - \mathbf{S}^1|$ and $|\mathbf{S}^1 - \boldsymbol{\gamma}^1|$, respectively.

correlated with each other and contain redundant information. Besides, the correlation coefficient between adjacent bands can be calculated by the following equation:

$$\rho_{i,j} = \frac{\mathrm{Cov}(\mathbf{g}_i, \mathbf{g}_j)}{\sqrt{\mathrm{Var}(\mathbf{g}_i)\mathrm{Var}(\mathbf{g}_j)}} \tag{8}$$

where cov and var are the covariance and variance. $\mathbf{g}_i$ and $\mathbf{g}_j$ refer to the ith and jth hyperspectral bands. $i = 1, 2, \ldots, R - 1$. As shown in Fig. 3(b), the highest correlation coefficient is 0.99996 and the lowest is 0.89014. Therefore, it is preferable to exploit specific methods for feature reduction.

Based on this observation, the original HSI is, therefore, first partitioned into Q subsets of adjacent bands as follows:

$$\boldsymbol{C}^q = \begin{cases} \left(\mathbf{g}_{\lceil R/Q \rceil (q-1)+1}, \ldots, \mathbf{g}_{\lceil R/Q \rceil q}\right) & 1 \leq q \leq Q - 1 \\ \left(\mathbf{g}_{\lceil R/Q \rceil (q-1)+1}, \ldots, \mathbf{g}_R\right) & q = Q \end{cases}. \tag{9}$$

In the above expression, $\boldsymbol{C}^q$ ($q = 1, 2, \ldots, Q$) indicates the qth hyperspectral subset. $\mathbf{g} = (\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3, \ldots, \mathbf{g}_R)$ denotes an HSI with R spectral bands and $M \times N$ pixels in the spatial domain. $\lceil R/Q \rceil$ represents the smallest integer greater than or equal to R/Q.

Then, a simple and effective average fusion method [39] is adopted to reduce spectral dimension by combining the complementary information of adjacent bands in each subset. Specifically, the fused band $\mathbf{S}^q$ is obtained by the following equation:

$$\mathbf{S}^q = \frac{\sum_{i=1}^{N_q} \boldsymbol{C}_i^q}{N_q} \tag{10}$$

where $\boldsymbol{C}_i^q$ represents the ith band in the qth hyperspectral subset, and N_q is the total number of spectral bands in the qth subset. As shown in Fig. 4(b), the fused image retains the anomalies while effectively removing noise and redundant information to some extent.

B. Feature Location

In this stage, the opening and closing reconstruction are applied for feature localization, in which a basic operand, a rectangular SE $\mathbf{b}$, of size $r_a \times r_a$ is therefore defined. The erosion and dilation of $\mathbf{S}$ by $\mathbf{b}$ are first calculated as the marker images for dilation and erosion reconstruction, respectively. Let $\boldsymbol{\gamma}$ and $\boldsymbol{\phi}$ denote the opening and closing reconstruction, respectively.

Fig. 5. (a) Results of the differential operation ($\phi^1 - \gamma^1$). (b) Results of the self-guided filtering ($\mathbf{V}^1$). (c) Reference map.
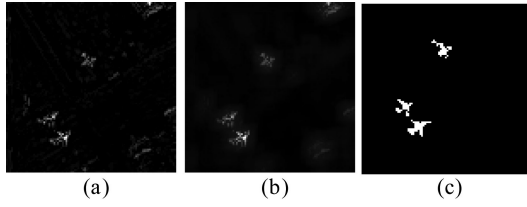

Algorithm 2: Pseudocode of the Self-Guided Filter.

Input: I
Parameters: r_b, ϵ
   mean$I = f$mean$(\mathbf{I})$ (1)
   Corr$I = f$mean$(\mathbf{I}.*\mathbf{I})$ (2)
   Var$I = $Corr$I - $mean$I.*$mean$I$ (3)
   $a = VarI./(VarI + \epsilon)$ (4)
   $b = $mean$I - a.*$mean$I$ (5)
   $\mathbf{V} = a.*\mathbf{I} + b$ (6)
Output: V

The operations performed at this stage can be expressed by the following equations:

$$\gamma^q = R_{\mathbf{S}^q}^D(\mathbf{e}^q) = D_{\mathbf{S}^q}^{(k)}(\mathbf{S}^q \ominus \mathbf{b})$$
$$= D_{\mathbf{S}^q}^{(1)}(D_{\mathbf{S}^q}^{(k-1)}(\mathbf{S}^q \ominus \mathbf{b}))$$
$$\phi^q = R_{\mathbf{S}^q}^E(\mathbf{d}^q) = E_{\mathbf{S}^q}^{(k)}(\mathbf{S}^q \oplus \mathbf{b}) \qquad (11)$$
$$= E_{\mathbf{S}^q}^{(1)}(E_{\mathbf{S}^q}^{(k-1)}(\mathbf{S}^q \oplus \mathbf{b}))$$

where $q = 1, \ldots, Q$ and SE $\mathbf{b}$ is also used to perform erosion and dilation reconstruction. However, the morphological reconstruction following its definition [see (1) and (2)] is an iterative procedure that is performed until reaching idempotence, which results in unpredictable time performance on hardware implementation. Fortunately, we experimentally proved that fixed iterations (k) can guarantee high accuracy in Section VI. Compared with the area-based attribute filtering applied in STGF [18] and AED [16], the morphological opening and closing reconstruction are easier to implement.

As shown in Fig. 4(c)–(f), the intermediate results on the San Diego dataset are visual, and the number of iterations in both opening reconstruction and closing reconstruction is set to 20. Taking the opening reconstruction operation as an example, the erosion operation [see Fig. 4(c)] is first performed to remove bright features smaller than the size of SE. However, the size of the dark feature is increased and the background is darkened. Subsequently, the reprocessing of the dilation reconstruction [see Fig. 4(d)] removes the bright features while reducing its impacts imposed on dark features and background. In other words, opening reconstruction only removes bright features on a constant background. Conversely, the closing reconstruction only removes dark features within a constant background.

C. Feature Extraction

Considering that anomaly objects in HSIs usually appear as small-area objects compared with the background [16], the anomaly objects [see Fig. 5(a)] are extracted through the differential operation, which merges the dark [see Fig. 4(g)] and light [see Fig. 4(h)] features from the above stage. For $q = 1, \ldots, Q$, the process can be expressed as the following equation:

$$\mathbf{I}^q = |\phi^q - \mathbf{S}^q| + |\mathbf{S}^q - \gamma^q| = \phi^q - \gamma^q. \qquad (12)$$

D. Feature Clustering

Although morphological reconstruction can effectively extract small-sized bright and dark objects in the image, it is difficult for the reconstruction to filter out some background interference and random noise, which may increase the false alarm rate of anomaly detection. However, as a local linear model that sufficiently utilizes the local spatial information of the image, guided filtering is a better alternative in solving the aforementioned problems.

The guided filter computes the filtering output by considering the content of a guidance image, which can be the input image itself or another different image. In this article, we consider the special case in which the input image is used as the guide. The results of self-guided filtering can be defined as follows:

$$\mathbf{V}_i^q = a_j^q \mathbf{I}_i^q + b_j^q \quad \forall i \in \omega_j^q. \qquad (13)$$

The coefficients can be calculated as

$$a_j^q = \frac{\frac{1}{|\omega^q|} \sum_{i \in \omega_j^q} (\mathbf{I}_i^q)^2 - (\mu_j^q)^2}{(\sigma_j^q)^2 + \epsilon} \qquad (14)$$
$$b_j^q = \mu_j^q - a_j^q \mu_j^q$$

where μ_j^q and σ_j^{q2} are the mean and variance of $\mathbf{I}^q$ in the local window ω_j^q of size $(2r_b + 1) \times (2r_b + 1)$, ϵ is a parameter that controls the smoothness of the filtering, and $q = 1, \ldots, Q$. Inspired by [40], we eliminate two redundant mean filters (steps (9) and (10) in Algorithm 1). In order to prove the effectiveness of this method, we conducted a lot of experiments in MATLAB using a and b instead of their mean. Our experimental results in Section VI showed that this modification imposes small impact on accuracy while greatly reducing the memory consumption. The self-guided filter applied in this article is described in Algorithm 2. As shown in Fig. 5(b), it can be observed that the local spatial strong intercorrelation amidst adjacent pixels is utilized efficiently in feature clustering. Moreover, the background noise is removed in contrast to Fig. 5(a).

Finally, we perform the following operation to obtain the detection map:

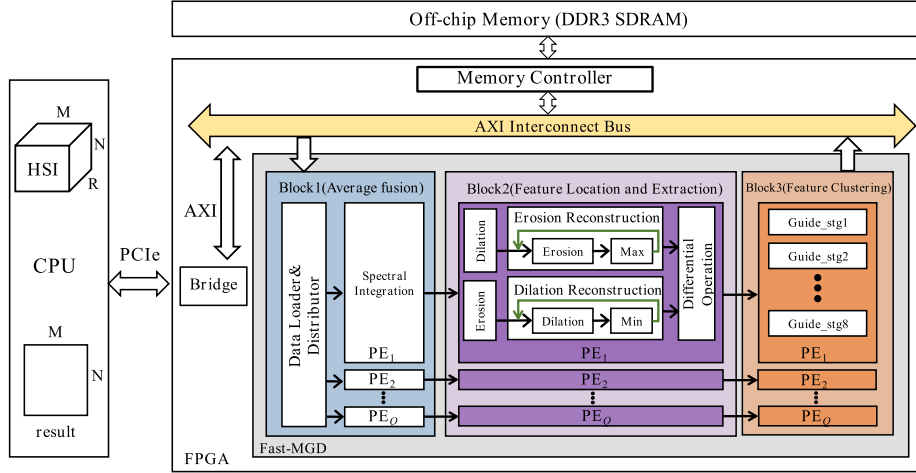$$\mathbf{O} = \sum_{q=1}^{Q} \omega_q \mathbf{V}^q \qquad (15)$$

Fig. 6. Overall hardware structure of Fast-MGD.

where $\omega_q = 1/Q\,(q = 1, \ldots Q)$ is the set, meaning that the processed result of each fused band contributes equally to finding anomaly objects.

V. FPGA IMPLEMENTATION

This section discusses the detailed implementation of our proposed algorithm. An overall hardware structure of Fast-MGD is given in Section V-A. Section V-B describes the microscopic hardware architecture of Fast-MGD in detail. Section V-C explains our design optimization strategies in FPGA implementation.

A. Overall Hardware Architecture of Fast-MGD

As shown in Fig. 6, the framework of Fast-MGD is mainly composed of two components including an off-chip memory (DDR3 SDRAM) for caching the HSI data and a processor core, which can be directly used for on-board real-time processing. The processor core that encompasses three modules is used for the major purpose of data processing. The first module is the average fusion unit (AFU), which is in charge of reducing the spectral dimensions of a given HSI. The second module is the feature location and extraction unit (FLEU), which is designed to extract anomalies based on morphological reconstruction. The last module is the feature clustering unit (FCU) being applied to further rectify the detection map through the self-guided filtering. Each module contains Q identical processing elements (PEs) that work in parallel. It is emphasized that only deep pipelined design highlighted by GRAY in the figure is concerned, while the rest of the figure only helps us to verify the function and performance of the proposed Fast-MGD hardware implementation.

B. Microscopic Hardware Architecture of Fast-MGD

1) AFU: The Data Loader and Distributor reads HSIs from the DDR3 SDRAM through the AXI interconnect bus and loads pixel vectors by bands for AFU. The data width of DDR3 SDRAM is set to 512 bit, which is the maximum available bit width of the device. Considering the case of an HSI with a pixel width by 16 bits, each address can store 32 successive bands of a pixel vector. The pixel derived from the Data Loader and Distributor is first stored in FIFO performing with 512 bit width, and then, the high-bit data are transferred to the Q Spectral Integration units, thereby reducing processing time through full use of the bandwidth of DDR3 SDRAM.

The Spectral Integration conducts sum and average operations along the spectral dimension during the data-width conversion (512-bit data are split into 16-bit data). The final result is stored in the FIFO and is delivered to the next module.

2) FLEU: As shown in Fig. 6, the opening and closing reconstruction (the erosion/dilation followed by dilation/erosion reconstruction) first work in parallel, and then, the differential operation is executed for feature extraction. The two stages transmit the data stream through a memory buffer, whereby a task-level pipeline architecture is, therefore, designed.

As can be seen from (11) in Section IV, the two basic image processing pipelines, referred to as erosion and dilation, are the core of the morphological reconstruction. Specifically, the two pipelines can be configured with registers and memory buffers to design more complex dilation/erosion reconstruction utilizing standard techniques.

The dilation/erosion computes the maximum/minimum for each pixel within a customizable rectangular SE. As depicted in Fig. 8, the dilation/erosion can be further subdivided into the row processing unit (RPU) and column processing unit (COPU).

The RPU reads the pixel in sequence from the FIFO as the dilation is initiated. Within each cycle, the new pixel and the pixels involved in the current shift registers of size $r_a - 1$ are first compared in the row diversity unit (RDU); then, the pixels in the registers are updated via reading the new pixel and removing the old one. We point out that the $array\_partition$ directive in HLS is utilized to completely decompose the array into individual elements. The comparison result is passed to COPU through the shift register.

When the COPU starts to work, the new result pixel from RPU is first compared with a column of pixels ($r_a - 1$ pixels)

```
using namespace hls;
#define iteration 20
#define fusion_band 2
void morph_close(stream<datatype_16nt> in[fusion_band],stream<datatype_16nt> ori[fusion_band]
                               ,stream<datatype_16nt> out[fusion_band]){
#pragma HLS DATAFLOW
        stream<datatype_16nt> mid1[iteration][fusion_band];
#pragma HLS STREAM variable=mid1 depth=1 dim=2
        stream<datatype_16nt> ori_1[iteration][fusion_band];
#pragma HLS STREAM variable=ori_1 depth=1 dim=2
        stream<datatype_16nt> mid2[iteration][fusion_band];
#pragma HLS STREAM variable=mid2 depth=1 dim=2
        stream<datatype_16nt> ori_2[iteration][fusion_band];
#pragma HLS STREAM variable=ori_2 depth=1 dim=2
        stream<datatype_16nt> mid3[fusion_band];
#pragma HLS STREAM variable=mid3 depth=1
        stream<datatype_16nt> ori_3[fusion_band];
#pragma HLS STREAM variable=ori_3 depth=1
        Dilation(in,ori,mid1[0],ori_1[0]);
        for(unsigned char i=0;i<iteration-1;i++){
#pragma HLS UNROLL
                Erosion(mid1[i],ori_1[i],mid2[i],ori_2[i]);
                Max(mid2[i],ori_2[i],mid1[i+1],ori_1[i+1]);
        }
        Erosion(mid1[iteration-1],ori_1[iteration-1],mid3,ori_3);
        Max_last(mid3,ori_3,out);
}
```

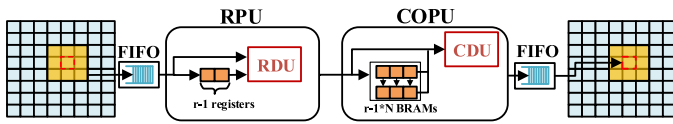Fig. 7.   Sample code used for implementing morphological closing operation.

Fig. 8.   Sharing architecture for dilation, erosion, and mean filter.

being read from the BRAMs of size $(r_a - 1) \times N$. The width of the image is represented by N. Then, the pixels of the current column in BRAM are updated via reading the new pixel and removing the old one. The $array\_reshape$ directive we use can reduce the consumption of BRAMs while accessing data in parallel. It should be emphasized that a result pixel can be obtained by $2 \times (r_a - 1)$ comparisons for the optimized structure of dilation/erosion, throughout which the computational efficiency is, therefore, improved.

With respect to its definition, the optimized dilation reconstruction consists of k identical stages, where k is set to 20. The i stage uses the output of the $i - 1$ stage as the marker input together with the buffered mask image, in which dilation and pointwise minimum (Min in Fig. 6) are performed at each stage. Similarly, erosion reconstruction is, therefore, designed with the same structure. Fig. 7 shows the sample code used to implement morphological closing operation in HLS. The ori, ori_1, ori_2, and ori_3 in the figure is mainly used to cache the mask image that is used for morphological closing operation. HLS_UNROLL directive is added to implement k instances of erosion reconstruction.

Finally, the subtraction is performed on the results of the opening and closing reconstruction, by which the final results are stored in the FIFO for the FCU.

3) FCU: This section describes how to effectively implement the self-guided filter used for feature clustering in hardware, of whose pseudocode is shown in Algorithm 2. As the fundamental operation of the self-guided filter, the mean filter is a windowed operation with radius r_b, which undertakes the main computational burden. Particularly, the sharing architecture of erosion and dilation is also applicable for the mean filter, in which only the RED color highlighted RDU and CDU need to
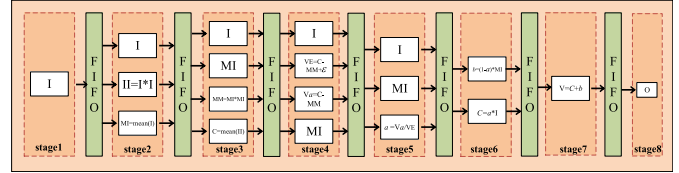

Fig. 9.   Hardware structure of self-guided filter (the mean filter is represented by **mean**).

be redesigned. Specifically, the RDU and CDU in mean filter perform sum operation instead of comparison. Consequently, the mean value is computed by multiplying the window sum (column sum) by the value of $1/(2r_b + 1)^2$.

As shown in Algorithm 2, the simplified self-guided filtering in each PE only includes two mean filters, which compute the values of $meanI$ and $CorrI$. The remaining values described in Algorithm 2 are computed using a set of arithmetic units, such as fixed-point multipliers, adders, subtractors, etc. We perform reasonable segmentation and sorting adjustments on complex coefficient operations and propose a deep pipelined design to make the entire module with higher throughput. As shown in Fig. 9, the entire module is divided into eight pipeline stages. Seven memory buffers of depth 1 are configured to resolve data dependencies amidst operations.

C. Highlight

First, it is worth noting that different optimization strategies concerning arrays, loops, latency, and throughput are utilized to achieve a tradeoff between area and processing speed through adding directives provided in HLS. For instance, the entire hardware implementation employs global deep pipelining, which takes advantage of the $dataflow$ directive to raise the concurrency of the RTL implementation and overall throughput. The three units are connected by FIFO memories with a depth of 1. The C++ template class $hls :: stream <>$ is adopted for mapping these data to FIFOs.

Second, we design a sharing architecture for the core units of FCU and FLEU (erosion, dilation, and mean filter), as shown in Fig. 8, which only requires reconfiguration of the RDU and CDU highlighted by RED. The general pipelined architecture based on row and column decomposition further simplifies the overall hardware implementation, thereby achieving high throughput performance.

Third, the type of the input data is 16-bit unsigned fixed-point (15-bit fractional part), and the type of intermediate data needs to be taken into account seriously due to its significant impact on detection accuracy and resource consumption. In particular, a large number of arithmetic units are designed in the FCU. As listed out in Table I, the appropriate data types provided by HLS are applied to different intermediate data by balancing detection accuracy and resource consumption. W and I in the table represent the word length in bits and the number of bits above the decimal point. ap_ufixed<> refers to the unsigned fixed point.

TABLE I
DATA STRUCTURE FOR FAST-MGD IN FPGA IMPLEMENTATIONS

Fixed-pointed data	ap_ufixed$<$W, I $>$
$meanI$	ap_ufixed$<18, 1>$
$CorrI/VarI$	ap_ufixed$<23, 1>$
a/b	ap_ufixed$<24, 1>$
V/O	ap_ufixed$<19, 1>$

Fourth, it needs to be emphasized that the proposed FPGA-based hardware architecture does not depend on any specific underlying physical devices of FPGA and vendor-provided IP cores. In addition, the framework supports HSIs with different bands and sizes. The parameters of each unit, the number of fused bands Q, the size of SE r_a, the radius of mean filter r_b, the iteration k, and the smoothness of the guided filtering ϵ in HLS can be adjusted to adapt to various complicated scenarios. This parameter-configurable architecture improves scalability and portability of the whole structure.

VI. EXPERIMENTAL RESULTS AND ANALYSIS

This section is organized as follows. The hyperspectral datasets used for different scenes in the experiments are first introduced in Section VI-A. Section VI-B describes the parameter settings of the detection algorithms used for comparison and the proposed Fast-MGD. Section VI-C evaluates the detection accuracy of the software version using MATLAB of the proposed Fast-MGD by comparing the widely used detectors mentioned above. The effects of morphological reconstruction on feature localization and self-guided filter for feature clustering in Fast-MGD are further analyzed in Section VI-D. Section VI-E shows a comparison of the proposed hardware implementation evaluated on a Xilinx Virtex7 FPGA with the corresponding software version.

A. HSI Dataset

1) San Diego Dataset: The first dataset used in the experiments has been widely applied for target detection, which is captured by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) sensor over the San Diego airport area, CA, USA. The original image contains 100×100 pixels with 224 bands, ranging from 400 to 2500 nm. Due to dense water vapor and atmospheric influence, 189 bands are retained after removing the noise bands (1–6, 33–35, 97, 107–113, 153–166, and 221–224). Three aircraft are treated as the anomaly objects that need to be detected in the image. The sample image and the reference detection map is shown in Fig. 10(a), respectively.

2) Urban-Beach Dataset: The Urban-Beach database collected by the AVIRIS Sensor was accessible on the website (http://xudongkang.weebly.com). The visualization images and corresponding maps are shown in Fig. 10(b)–(e), respectively. These scenes consist of 100×100 [see Fig. 10(b)–(d)] or 150×150 [see Fig. 10(e)] pixels with different bands, in which the noise bands have been removed. The different spatial resolutions of these scenes are described in Table IV.

3) EI Segundo DataSet: The third dataset collected by the AVIRIS Sensor describes an area that is composed of oil refinery, several residential areas, parks, and a school zone in EI Segundo, CA, USA, which includes 224 spectral channels in the range of 366–2469 nm. The picture contains 2048 anomalous pixels with a spatial size of 250×300 and a ground resolution of 7.1 m. The visualization images and corresponding maps are shown in Fig. 10(f), respectively.

B. Comparison of Methods and Parameter Settings

The detection accuracy of different algorithms can be evaluated by the receiver operating characteristic (ROC) through commonly used measurement [41]. Meanwhile, in order to further quantify the magnitude of the detectors' accuracy, the area under a ROC curve is adopted, being referred to as the area under the curve (AUC) [42]. As shown in Fig. 11, the true positive rate (TPR) on the vertical axis is defined as the proportion of correct positive results in all positive samples when a set of threshold (T) for the detection map is determined. Similarly, the false positive rate (FPR) on the horizontal axis indicates the proportion of false positive results in all negative samples. When the values of FPR are identical, the increase in the TPR value makes the AUC scores of TPR and FPR approaching the unity value, indicating that the detection accuracy is positively correlated with the performance of the algorithm. In this article, the AUC score of FPR and T also functions as a critical indicator to reflect the false alarm rate of the detection algorithm. Conversely, the AUC score of FPR and T approaching to zero implies lower false alarm rate, with which the detection accuracy is negatively correlated.

To verify the advantageous performance of Fast-MGD in hyperspectral anomaly detection, various detectors are involved concerning the AED [16], RX [7], LRX [10], and CRD [13] for comparison. In this case, we select the optimal parameters for the specific sample image with regard to the corresponding AUC scores of FPR and TPR. For the LRX algorithm, the proper size of the sliding window should be selected under scrutiny to achieve its optimal performance. For the CRD algorithm, Li and Du confirmed that the regularization parameter **λ** being set to 10^6 can achieve the optimal detection accuracy. For the AED algorithm, the specific parameter settings involved in the AED algorithm have been described by its authors in detail.

For our proposed algorithm, the parameters Q, r_a, k, and r_b used in the operations of feature location and feature clustering can be adjustably configured to adapt different scenes, thereby achieving optimal detection results. The AUC score of TPR and FPR is applied to evaluate the objective performance of the proposed algorithm by systematically varying the parameter settings one by one. Fig. 12 reports the effects of parameter settings on six real datasets, respectively. $Q = 2$ is adopted by finding a tradeoff between detection accuracy and resource consumption. While such a setting cannot achieve the best performance for individual sample images, it is designed for ensuring stable and acceptable performance for most of the tested HSIs. The size of SE r_a is set to 3 in Fig. 10(a) and (c), to 5 in Fig. 10(b), (d), and (e), and to 9 in Fig. 10(f). A reasonably large value of
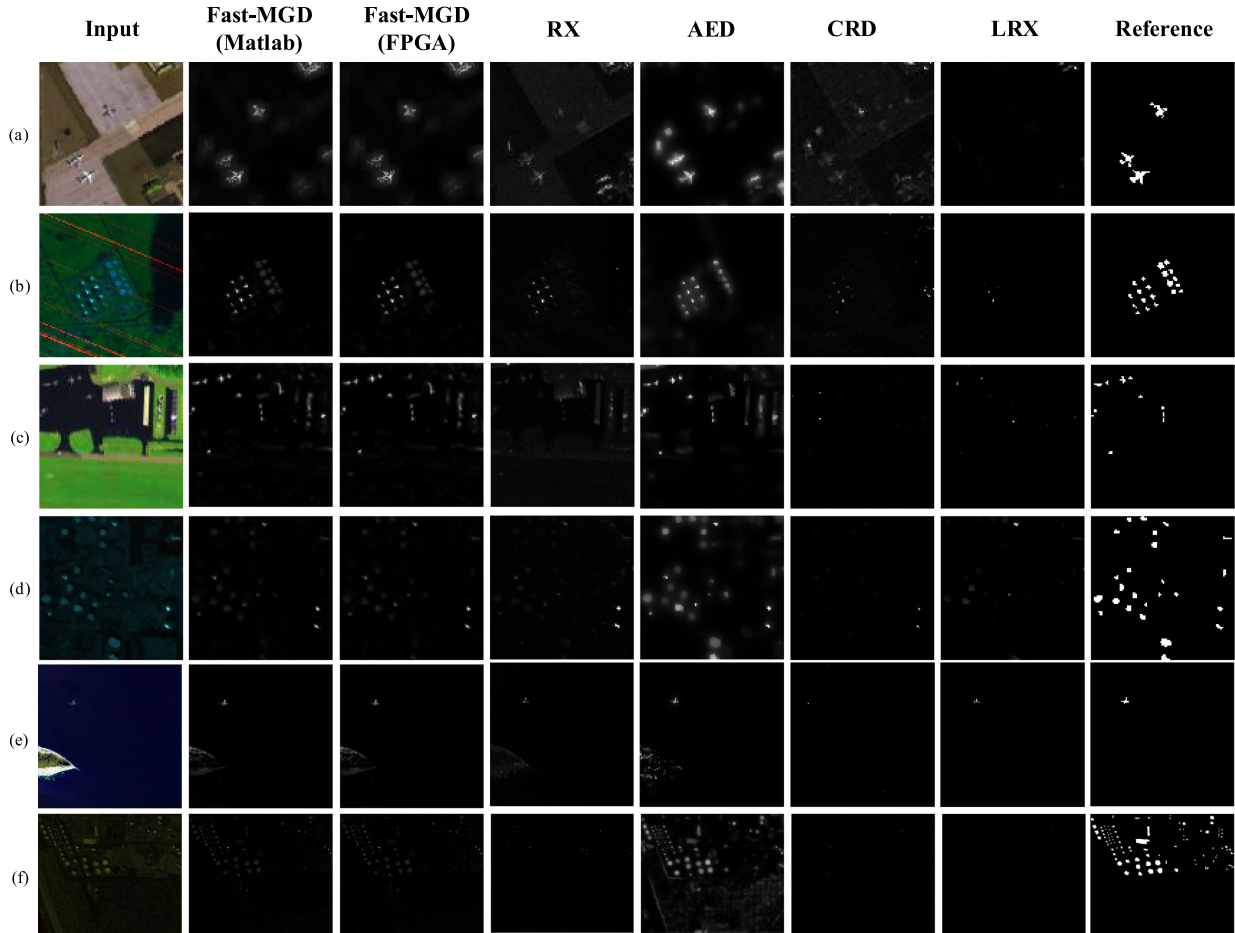
Fig. 10. (a)–(f) Color composites of HSIs and detection maps of the compared methods. The last column shows the reference detection maps, which are obtained by human labeling.

r_a may enable the detection of certain false anomalous objects. Similarly, a relatively small r_a may result in failures to detect a large area of anomalous information, imposing detrimental impacts on the performance of the detection accuracy. Both of the above cases attribute to the low detection accuracy and AUC scores. However, a default setting $k = 20$ is applicable for all sample images. The influence of k can be analyzed with more detailed descriptions in Section VI-D. $r_b = 1$ is set for most of the sample images. Exceptionally, r_b is adjusted to 5 for the San Diego dataset to achieve better performance [see Fig. 10(a)].

C. Detection Results

For the San Diego dataset, the AUC score of TPR and FPR is listed in Table II. The AUC score is 0.98432, which is much higher than RX (0.94094), CRD (0.96263), and LRX (0.91430). Compared with the AED method in which the AUC score of TPR and FPR is ranked the first, the detection map of the Fast-MGD method contains less background information in Fig. 10(a).

For Urban-Beach datasets, the proposed method in Table II shows the best AUC scores of TPR and FPR in all scenes. Compared with the RX, LRX, CRD, and AED, the detection results obtained by Fast-MGD achieve better visual inspection,

as shown in Fig. 10(b)–(e). Most of the anomalies in urban scenes [see Fig. 10(b)–(d)] are unable to be detected by the LRX and CRD methods, and the lousy AUC scores of TPR and FPR also support this observation. Nevertheless, these two methods display inadequate performance throughout with respect to computing time, not to mention the inconvenience induced in optimal settings of the inner and outer window. As shown in Fig. 11(e), LRX performs well on the ROC curve of this dataset, yet it is worth noting that this method cannot have satisfying detection results on all HSIs. The RX method is merely suitable for sample images with simple background distribution and for large area anomalies, which is unsuitable for complex HSIs, as shown in Fig. 10(c).

For the EI Segundo dataset, the detection maps are shown in Fig. 10(f). By visualizing the final detection maps, we notice that the proposed Fast-MGD shows excellent performance on anomaly detection compared to other methods, whereby most of the anomalies in the sample image can be clearly observed. The AUC score of TPR and FPR shown in Table II further demonstrates the superiority our approach. Although the AUC score of TPR and FPR of RX is considerable, it does not perform well on visual effects. In addition, we also compare the ROC curves of different methods on the EI Segundo dataset. As
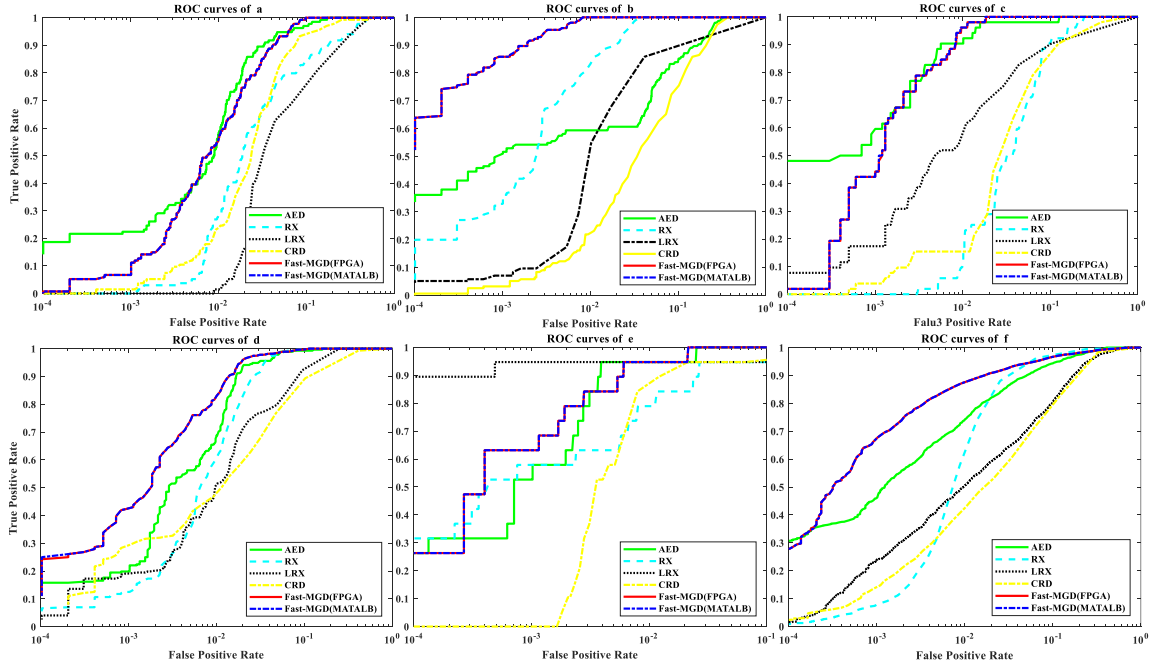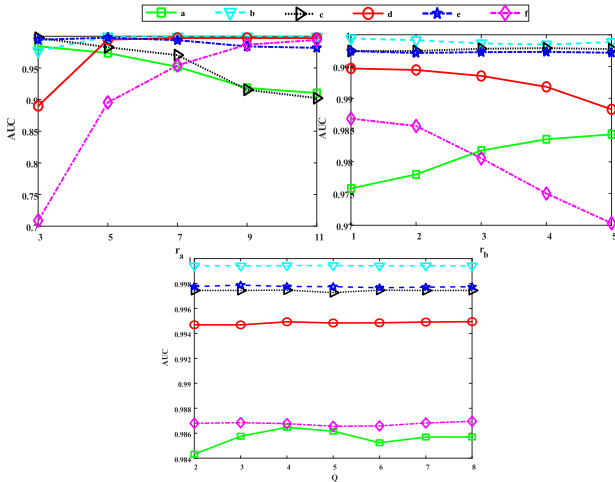
Fig. 11.   ROC curves of the compared methods.

Fig. 12.   Parameter sensitivity analysis for parameter r_a, r_b, and Q.

TABLE II
EVALUATION AUC SCORES OF (TPR AND FPR) OBTAINED BY THE FAST-MGD
(MATLAB), FAST-MGD (FPGA), RX, AED, CRD, AND LRX

HSIs	Fast-MGD (Matlab)	Fast-MGD (FPGA)	RX	AED	CRD	LRX
a	0.98432	0.98432	0.94041	**0.98499**	0.96263	0.91430
b	0.99942	**0.99942**	0.99463	0.95585	0.93042	0.91571
c	0.99745	**0.99745**	0.95119	0.99554	0.93970	0.93420
d	0.99470	**0.99470**	0.98870	0.99131	0.95156	0.97054
e	0.99776	**0.99776**	0.98281	0.99740	0.96720	0.97495
f	**0.98680**	0.98672	0.98153	0.98038	0.93429	0.94439
Average	**0.99341**	0.99340	0.97321	0.98425	0.94763	0.94235

TABLE III
EVALUATION AUC SCORES OF FPR AND T OBTAINED BY THE
FAST-MGD (MATLAB), FAST-MGD (FPGA), AND AED

HSIs	Fast-MGD (Matlab)	Fast-MGD (FPGA)	AED
a	0.0343	**0.0343**	0.0343
b	0.0059	**0.0059**	0.0228
c	0.0144	**0.0144**	0.0218
d	0.0062	**0.0062**	0.0311
e	0.0045	**0.0045**	0.0051
f	0.0063	**0.0063**	0.0223
Average	0.0119	**0.0119**	0.0229

shown in Fig. 11(f), the proposed method exhibits higher TPRs regardless of the FPR fluctuating between 0 and 1.

The AED method comes the second out of the proposed methods in terms of the average AUC scores of TPR and FPR. As shown in Fig. 11(a), (c), and (d), the ROC curves of AED are close to that of the Fast-MGD in these three scenarios. In order to further compare the performance of AED with that of Fast-MGD, AUC scores of FPR and T are shown in Table III. It can be noticed that our algorithm has a lower false alarm rate while remaining higher detection accuracy. Furthermore, we emphasize that the proposed Fast-MGD is a hardware-friendly algorithm featured with lower complexity.

D. Component Analysis

The opening/closing reconstruction performed in feature localization can effectively describe the spatial information of HSIs, which requires repeated combining of dilation/erosion and
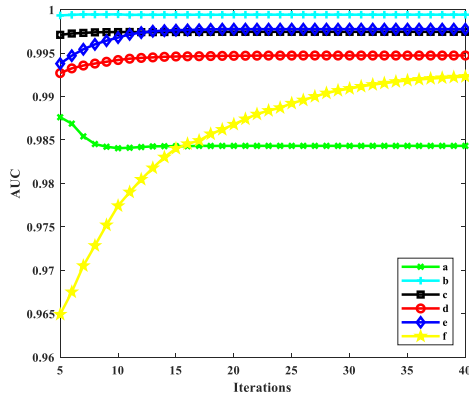
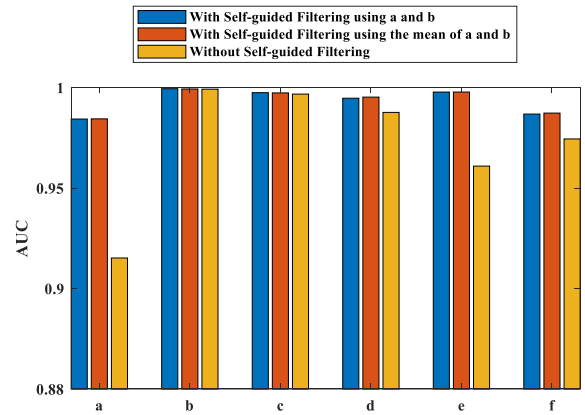Fig. 13. Effect of the number of iterations on each dataset.

Fig. 14. AUC scores of FPR and TPR on Fast-MGD with simplified self-guided filtering, with self-guided filtering, and without the self-guided filtering.

pointwise minimum/maximum operations until idempotency is reached. However, a great many of uncontrollable iterations sometimes may reach hundreds of times, of whose intricacies not only require a lot of time to tackle with, but also bring unbelievably high computational costs in hardware implementations.

As shown in Fig. 13, we analyze the impact of the number of iterations imposed on the AUC scores of TPR and FPR on each dataset while keeping other parameters remain optimal. It is surprising that the continuous increase in the number of iterations did not induce correlated rise in the curve of AUC; instead, it continued to rise first and then tended to stabilize. For example, the AUC score obtained through the proposed Fast-MGD is 0.99776 in Fig. 10(e) when the number of iterations increases to 20. Subsequently, the continuous increase in the iterations does not bring any evident improvements in the AUC scores, upon which the number of iterations can be set to a fixed value for all images without considering idempotency. This operation can significantly reduce the computational costs and memory consumption at the expense of a merely small loss in detection accuracy. However, we can observe that the AUC score of a dataset [see Fig. 10(a)] is the highest (0.98762) when the iterations is 5. It is worth noting that too small iterations will lead to a high false alarm rate. For example, when $k = 5$, the AUC score of FPR and T is as high as 0.0410 for a dataset. A large number of experiments have found that a default setting $k = 20$ can enable most tested HSIs to perform well in false alarm rate and detection accuracy.

We also analyze the influence of the simplified self-guided filtering over additional experiments on each dataset. As shown in Fig. 14, we calculated the AUC scores of TPR and FPR on Fast-MGD with simplified self-guided filtering, with self-guided filtering, and without self-guided filtering, respectively. The third column clearly shows that self-guided filtering significantly improves the detection accuracy. Moreover, the beneficial advantages of using the spatial correlation amidst adjacent pixels of HSIs for anomaly detection are, therefore, validated. By observing the first and second columns of the histogram, it can be concluded that the negative impact of the mean of a and b on detection performance can be almost ignored compared with the reduction of memory consumption.

E. FPGA Implementation

1) Comparison of Detection Accuracy Performance: It should be pointed out that the parameter settings in the FPGA implementation of Fast-MGD are identical with that in the software. As shown in Fig. 10, the proposed FPGA implementation shares almost the same visual effects with the software version. In Fig. 11, the ROC curves of hardware implementation are depicted. It is noticeable that the ROC curves of all scenes are very close to that of the software, which is attributed to the reasonably sound setting of intermediate data types in the FCU. Hence, the proposed deep pipelined hardware architecture is proven to be far more than satisfactory.

2) Performance Evaluation: A Virtex7 XC7VX690 T FPGA used to evaluate the proposed Fast-MGD hardware architecture contains 433 200 LUTs, 174 200 LUTRAMs, 866 400 FFs, 1470 Block RAMs, and 3600 DSPs. Table IV shows the resource utilization corresponding to Fast-MGD for diverse images with different sizes and resolutions using Vivado 2018.3. Optimal configurable parameters are set for different HSIs to ensure excellent detection accuracy.

As illustrated in Table IV, a constant number of DSPs are mainly used to implement the multiplication of Stage 2, Stage 3, and Stage 6 in FCU. Most FIFOs of depth 1 are inferred as LUTs and FFs. The consumption of BRAMs is mainly due to COPU in the sharing architecture. The number of LUTs, LUTRAMs, FFs, and BRAMs is related to the image size and the setting of configurable parameters. Also, Table V lists out the resource summary of dilation using Vivado HLS for fixed image size (100×100) and varied r_a to further quantitatively evaluate the designed sharing architecture. The estimation of the fastest achievable clock frequency is 3.310 ns. The number of LUTs, FFs, and BRAMs increases linearly with r_a. In summary, our cost-effective hardware designing philosophy using HLS aims at supporting HSIs with different sizes and spectral bands through adjustment of several configurable parameters, including Q, k, r_a, and r_b.

Three different platforms, including C++ using OpenMP, MATLAB, and FPGA for the above six datasets, are adopted in

TABLE IV
SUMMARY OF RESOURCE UTILIZATION FOR THE FPGA-BASED IMPLEMENTATION IN HSIS WITH DIFFERENT SPATIAL RESOLUTIONS AND SPECTRAL BANDS

HSIs	Size of image	Resolutions	LUTs (433200)	LUTRAMs (174200)	FFs (866400)	BRAMs (1470)	DSPs (3600)
a	$100 \times 100 \times 189$	3.5m	33969(7.84%)	836(0.48%)	49964(5.77%)	108(7.35%)	12(0.33%)
b	$100 \times 100 \times 207$	17.2m	43890(10.13%)	1001(0.57%)	57962(6.69%)	141(9.59%)	12(0.33%)
c	$100 \times 100 \times 191$	3.5m	31681(7.31%)	287(0.16%)	46370(5.35%)	99(6.73%)	12(0.33%)
d	$100 \times 100 \times 205$	7.1m	43874(10.13%)	1001(0.57%)	57929(6.69%)	141(9.59%)	12(0.33%)
e	$150 \times 150 \times 188$	17.2m	44906(10.37%)	1003(0.58%)	58593(6.76%)	141(9.59%)	12(0.33%)
f	$250 \times 300 \times 224$	7.1m	67591(15.60%)	963(0.55%)	83713(9.66%)	262.5(17.86%)	12(0.33%)

TABLE V
MEASURED PERFORMANCE OF DILATION USING 100×100 IMAGES

r_a	3	5	7	9	11
Latency	10808	11629	11640	12065	12496
BRAM_18Ks	1	2	3	4	5
FFs	298	469	765	1158	1601
LUTs	496	738	1014	1354	1726

TABLE VI
PROCESSING TIME MEASURED FOR FAST-MGD METHODS IN MATLAB, C++, AND FPGA IMPLEMENTATIONS

HSIs	MATLAB (ms)	C++(ms)	FPGA (ms)
a	224	45	0.528
b	232	55	0.553
c	212	45	0.513
d	232	54	0.553
e	408	122	1.176
f	1265	573	7.856

our experiments. The code of software version using MATLAB R2017b and Visual studio 2017 is executed on the Window 10 operation system equipped with the Intel Core (TM) quad CPU @2.50 GHz and 8-GB main memory. The hardware architecture is implemented by using Vivado HLS 2018.3. The clock frequency of the Virtex7 FPGA is set at 200 MHz. As shown in Table VI, the processing time of FPGA-based implementation achieved a speedup of more than 161 times higher than that of the MATLAB and more than 72 times higher than that of C++.

It must be emphasized that the processing time achieved by the FPGA implementation of Fast-MGD is strictly in real time for all the above scenarios. In specific, the scanning time of cross-track line in AVIRIS being referred to as a push-broom instrument is quite fast (8.3 ms to collect 512 full-pixel vectors) [43], indicating that the processing time of the scene must be limited within 0.16 s (in Fig. 10(e) and (f), the values are the exceptional 0.36 and 1.216) to achieve real-time performance thoroughly. From Table VI, it can be observed that the processing times are lower than 8 ms, including the load time and the data transfer time from the CPU to FPGA device.

In summary, the proposed hardware implementation characterized by low resource utilization and real-time processing speed displays its great advantages in on-board anomaly detection.

VII. CONCLUSION

Most of the existing hyperspectral anomaly detection methods have so far met with limited success due to high computational complexity, to which their inability in on-board implementation is, therefore, proven. In this article, a low-complexity anomaly detection algorithm and its corresponding cost-effective hardware architecture on FPGA are proposed. Experimental results on several real HSIs further verified the superiority of the Fast-MGD and its implementation with respect to computational complexity, processing speed, detection accuracy, as well as resource consumption, thereby validating its suitability for on-board application.

Despite the aforementioned advantages of our method, certain shortcomings should be pointed out concerning related parameter settings, which may impose encumbrance on its application. Our future research interests will focus on the adaptive parameter determination techniques, paving way for the improvement in the generality and robustness of our model.

REFERENCES

[1] C. Chang and S. Chiang, "Anomaly detection and classification for hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 40, no. 6, pp. 1314–1325, Jun. 2002.

[2] N. M. Nasrabadi, "Hyperspectral target detection: An overview of current and future challenges," *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 34–44, Jan. 2014.

[3] C. Mccann, K. S. Repasky, R. L. Lawrence, and S. L. Powell, "Multitemporal mesoscale hyperspectral data of mixed agricultural and grassland regions for anomaly detection," *ISPRS J. Photogramm. Remote Sens.*, vol. 131, pp. 121–133, 2017.

[4] M. T. Eismann, A. D. Stocker, and N. M. Nasrabadi, "Automated hyperspectral cueing for civilian search and rescue," *Proc. IEEE*, vol. 97, no. 6, pp. 1031–1055, Jun. 2009.

[5] B. Yang, M. Yang, A. Plaza, L. Gao, and B. Zhang, "Dual-mode FPGA implementation of target and anomaly detection algorithms for real-time hyperspectral imaging," *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 2950–2961, Jun. 2015.

[6] J. Cong, B. Liu, S. Neuendorffer, J. Noguera, K. Vissers, and Z. Zhang, "High-level synthesis for FPGAs: From prototyping to deployment," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 30, no. 4, pp. 473–491, Apr. 2011.

[7] I. S. Reed and X. Yu, "Adaptive multiple-band CFAR detection of an optical pattern with unknown spectral distribution," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 38, no. 10, pp. 1760–1770, Oct. 1990.

[8] Q. Guo, B. Zhang, Q. Ran, L. Gao, J. Li, and A. Plaza, "Weighted-RXD and linear filter-based RXD: Improving background statistics estimation for anomaly detection in hyperspectral imagery," *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2351–2366, Jun. 2014.

[9] J. E. Fowler and Q. Du, "Anomaly detection and reconstruction from random projections," *IEEE Trans. Image Process.*, vol. 21, no. 1, pp. 184–195, Jan. 2012.

[10] J. M. Molero, E. M. Garzn, I. Garcła, and A. Plaza, "Analysis and optimizations of global and local versions of the RX algorithm for anomaly detection in hyperspectral data," *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.*, vol. 6, no. 2, pp. 801–814, Apr. 2013.

[11] H. Kwon and N. M. Nasrabadi, "Kernel RX-algorithm: A nonlinear anomaly detector for hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 2, pp. 388–397, Feb. 2005.

[12] J. Li, H. Zhang, L. Zhang, and L. Ma, "Hyperspectral anomaly detection by the use of background joint sparse representation," *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 2523–2533, Jun. 2015.

[13] W. Li and Q. Du, "Collaborative representation for hyperspectral anomaly detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 3, pp. 1463–1474, Mar. 2014.

[14] R. J. Johnson, J. P. Williams, and K. W. Bauer, "AutoGAD: An improved ICA-based hyperspectral anomaly detection algorithm," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 6, pp. 3492–3503, Aug. 2012.

[15] X. Zhang, G. Wen, and W. Dai, "A tensor decomposition-based anomaly detection algorithm for hyperspectral image," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 5801–5820, Oct. 2016.

[16] X. Kang, X. Zhang, S. Li, K. Li, J. Li, and J. A. Benediktsson, "Hyperspectral anomaly detection with attribute and edge-preserving filters," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 10, pp. 5600–5611, Oct. 2017.

[17] M. Dalla Mura, J. A. Benediktsson, B. Waske, and L. Bruzzone, "Morphological attribute profiles for the analysis of very high resolution images," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 10, pp. 3747–3762, Oct. 2010.

[18] W. Xie, T. Jiang, Y. Li, X. Jia, and J. Lei, "Structure tensor and guided filtering-based algorithm for hyperspectral anomaly detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 7, pp. 4218–4230, Jul. 2019.

[19] S. Song, H. Zhou, Y. Yang, and J. Song, "Hyperspectral anomaly detection via convolutional neural network and low rank with density-based clustering," *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.*, vol. 12, no. 9, pp. 3637–3649, Sep. 2019.

[20] N. Ma, Y. Peng, S. Wang, and P. H. W. Leong, "An unsupervised deep hyperspectral anomaly detector," *Sensors*, vol. 18, no. 3, 2018, Art. no. 693.

[21] J. Lei, W. Xie, J. Yang, Y. Li, and C.-I. Chang, "Spectral–spatial feature extraction for hyperspectral anomaly detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 10, pp. 8131–8143, Oct. 2019.

[22] W. Xie, B. Liu, Y. Li, J. Lei, and Q. Du, "Autoencoder and adversarial-learning-based semisupervised background estimation for hyperspectral anomaly detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 8, pp. 5416–5427, Aug. 2020.

[23] K. Jiang, W. Xie, Y. Li, J. Lei, G. He, and Q. Du, "Semisupervised spectral learning with generative adversarial network for hyperspectral anomaly detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 7, pp. 5224–5236, Jul. 2020.

[24] N. Ma, X. Yu, Y. Peng, and S. Wang, "A lightweight hyperspectral image anomaly detector for real-time mission," *Remote Sens.*, vol. 11, no. 13, 2019, Art. no. 1622.

[25] P. Maragos, "Pattern spectrum and multiscale shape representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 701–716, Jul. 1989.

[26] C. K. Vala *et al.*, "High-speed low-complexity guided image filtering-based disparity estimation," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 2, pp. 606–617, Feb. 2017.

[27] M. Van Herk, "A fast algorithm for local minimum and maximum filters on rectangular and octagonal kernels," *Pattern Recognit. Lett.*, vol. 13, no. 7, pp. 517–521, Jul. 1992.

[28] J. Gil and M. Werman, "Computing 2-D min, median, and max filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 5, pp. 504–507, May 1993.

[29] D. Mukherjee, S. Mukhopadhyay, and G. Biswas, "FPGA-based parallel implementation of morphological operators for 2D gray-level images," *Arab. J. Sci. Eng.*, vol. 42, no. 8, pp. 3191–3206, 2017.

[30] Xilinx OpenCV User Guide (UG1233), Xilinx, San Jose, CA, USA, 2019.

[31] L. Vincent, "Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms," *IEEE Trans. Image Process.*, vol. 2, no. 2, pp. 176–201, Apr. 1993.

[32] K. Robinson and P. F. Whelan, "Efficient morphological reconstruction: A downhill filter," *Pattern Recognit. Lett.*, vol. 25, no. 15, pp. 1759–1767, 2004.

[33] O. Anacona-Mosquera, G. Teodoro, G. Vinhal, R. P. Jacobi, R. C. Sampaio, and C. H. Llanos, "Efficient hardware implementation of morphological reconstruction based on sequential reconstruction algorithm," in *Proc. 30th Symp. Integr. Circuits Syst. Des.*, Aug. 2017, pp. 162–167.

[34] O. Anacona-Mosquera, F. R. Cabral, R. C. Sampaio, G. Teodoro, R. P. Jacobi, and C. H. Llanos, "Efficient hardware implementation of the fast hybrid morphological reconstruction algorithm," in *Proc. 31th Symp. Integr. Circuits Syst. Des.*, 2018, pp. 1–6.

[35] J. Bartovskỳ, P. Dokládal, M. Faessel, E. Dokladalova, and M. Bilodeau, "Morphological co-processing unit for embedded devices," *J. Real Time Image Process.*, vol. 15, no. 4, pp. 775–786, 2018.

[36] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 6, pp. 1397–1409, Jun. 2013.

[37] C. Ttofis and T. Theocharides, "High-quality real-time hardware stereo matching based on guided image filtering," in *Proc. Des., Autom. Test Eur. Conf. Exhib.*, Mar. 2014, pp. 1–6.

[38] C. Ttofis, C. Kyrkou, and T. Theocharides, "A low-cost real-time embedded stereo vision system for accurate disparity estimation based on guided image filtering," *IEEE Trans. Comput.*, vol. 65, no. 9, pp. 2678–2693, Sep. 2016.

[39] Y. Zhang, X. Kang, S. Li, P. Duan, and J. A. Benediktsson, "Feature extraction from hyperspectral images using learned edge structures," *Remote Sens Lett.*, vol. 10, no. 3, pp. 244–253, 2019.

[40] P. Kareem, A. Khan, and C.-M. Kyung, "Memory efficient self guided image filtering," in *Proc. Int. SoC Des. Conf.*, 2017, pp. 308–309.

[41] J. P. Kerekes, "Receiver operating characteristic curve confidence intervals and regions," *IEEE Geosci. Remote Sens Lett.*, vol. 5, no. 2, pp. 251–255, Apr. 2008.

[42] C. E. Metz, "Basic principles of ROC analysis," *Semin. Nucl. Med.*, vol. 8, no. 4, pp. 283–298, 1978.

[43] J. Lei *et al.*, "A novel FPGA-based architecture for fast automatic target detection in hyperspectral images," *Remote Sens.*, vol. 11, no. 2, 2019, Art. no. 146.

Jie Lei (Member, IEEE) received the M.S. degree in telecommunication and information systems and the Ph.D. degree in signal and information processing from Xidian University, Xi'an, China, in 2006 and 2010, respectively.

From 2014 to 2015, he has been a Visiting Scholar with the Department of Computer Science, University of California, Los Angeles, CA, USA. He is currently an Associate Professor with the School of Telecommunications Engineering, Xidian University. His research interests include image and video processing, computer vision, and customized computing for big data applications.

Dr. Lei is a member of the Image Coding and Processing Center, State Key Laboratory of Integrated Services Networks. He is also with the Science and Technology on Electro-Optic Control Laboratory.

Geng Yang received the B.E. degree in telecommunications engineering, in 2019 from Xidian University, Xi'an, China, where she is currently working toward the M.S. degree with the Image Coding and Processing Center, State Key Laboratory of Integrated Services Networks.

Her research interests include hyperspectral image, anomaly detection, and unsupervised learning.

Weiying Xie (Member, IEEE) received the B.S. degree in electronic information science and technology from the University of Jinan, Jinan, China, in 2011, the M.S. degree in communication and information systems from Lanzhou University, Lanzhou, China, in 2014, and the Ph.D. degree in communication and information systems from Xidian University, Xi'an, China, in 2017.

She is currently an Associate Professor with the State Key Laboratory of Integrated Services Networks, Xidian University. She has authored or coauthored more than 30 papers in refereed journals. Her research interests include neural networks, machine learning, hyperspectral image processing, and high-performance computing.

Xiuping Jia (Senior Member, IEEE) received the B.Eng. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 1982, and the Ph.D. degree in electrical engineering from the University of New South Wales, Sydney, NSW, Australia, in 1996.

Since 1988, she has been with the School of Engineering and Information Technology, The University of New South Wales, Canberra, ACT, Australia, where she is currently an Associate Professor. She has authored or coauthored more than 190 referred papers, including 98 journal papers. She has coauthored the remote sensing textbook entitled *Remote Sensing Digital Image Analysis* (3rd ed. New York, NY, USA: Springer-Verlag, 1999 and 4th ed. New York, NY, USA: Springer-Verlag, 2006). Her research interests include remote sensing, image processing, and spatial data analysis.

Dr. Jia is a Subject Editor for the *Journal of Soils and Sediments* and an Associate Editor for the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING.

Yunsong Li (Member, IEEE) received the M.S. degree in telecommunication and information systems and the Ph.D. degree in signal and information processing from Xidian University, Xi'an, China, in 1999 and 2002, respectively.

In 1999, he joined the School of Tetecommunications Engineering, Xidian University, where he is currently a Professor. He is the Director of the Image Coding and Processing Center, State Key Laboratory of Integrated Services Networks. His research interests include image and video processing and high-performance computing.