

Hyper-spectral Image Processing Applications on the SIMD Pixel Processor for the Digital Battlefield

Sek M. Chai¹, Antonio Gentile¹, Wilfredo E. Lugo-Beauchamp²,
José L. Cruz-Rivera², D. Scott Wills¹

¹Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332-0250
{sek, scott.wills}@ece.gatech.edu

²Electrical and Computer Engineering
University of Puerto Rico in Mayagüez
Mayagüez, Puerto Rico 00681-9042
jcruz@ece.uprm.edu

Abstract

Future military scenarios will rely on advanced imaging sensor technology beyond the visible spectrum to gain total battlefield awareness. Real-time processing of these data streams requires tremendous computational workloads and I/O throughputs. This paper presents three applications for hyper-spectral data streams, vector quantization, region autofocus, and K-means clustering, on the SIMD Pixel Processor (SIMPil). In SIMPil, an image sensor array (focal plane) is integrated on top of a SIMD computing layer to provide direct coupling between sensors and processors, alleviating I/O bandwidth bottlenecks while maintaining low power consumption and portability. Simulation results with sustained operation throughputs of 500-1500 Gops/sec support real-time performances and promote focal plane processing on SIMPil.

Keywords: Military applications, Hyper-spectral imaging, Object detection and tracking, Segmentation, Scene understanding, Focal plane image processing, Automated Target Recognition (ATR).

1.0 Introduction

Image processing systems in future battlefield scenarios will likely utilize hyper-spectral imaging for new levels of threat identification. These systems will be deployed in a variety of platforms including vehicles, foot soldiers, robots, and smart munitions. The key to successful deployment of these systems is the ability to extract and disseminate critical information from sensor data streams in a timely fashion [19]. Because these systems are to operate in a mobile and perhaps covert manner, stringent resource limitations (size, weight, and power) pose additional system design challenges.

This paper presents application developments to process hyper-spectral data streams on the SIMD Pixel processor (SIMPil) [4]. SIMPil is a fine-grain parallel architecture being developed at Georgia Tech for focal plane image processing. By employing a large array of stream processors with parallel interconnect between image sensors and processors, the architecture alleviates the I/O bandwidth requirements in image processing. Unlike traditional SIMD systems [1][14][20], SIMPil maintains high performance and modest generality in a low power and portable environment. Implementation details on SIMPil are presented elsewhere in [4][5][8].

Three important applications for compression, region autofocus, and scene segmentation have been implemented on SIMPil: a full-search vector quantization application to compress large hyper-spectral data streams; a region autofocus application to identify and reacquire a new field of attention; and a K-means clustering application to segment large images. Simulation results indicate sustained operation throughput in the range of 500-1500 Gops/sec. These results suggest the potential for real-time execution of multistage applications where additional automated target recognition (ATR) algorithms can be used to further analyze the image.

The rest of the paper is organized as follows. Section 2 presents motivation for this research. Section 3 describes the architecture of the SIMPil system. Section 4 presents the application development for hyper-spectral data streams. Section 5 discusses the performance results. Conclusions are offered in Section 6.

2.0 Motivation

Two apparent limits of future hyper-spectral image processing systems are data bandwidth and processing throughput. With rapid sensor advances in resolution, frame rate, and dynamic range, current systems capabilities will soon be exceeded. Consider the following table of existing and futuristic hyper-spectral sensor arrays.

Table 1. Data bandwidth and processing throughputs for a selection of sensor arrays

	AVIRIS	CASI	HYDICE	Future sensor array*
Image resolution (pixels)	614 x 512	578 x 578	320 x 240	1000 x 1000
Dynamic range (bits/pixel)	12	12	12	12
Spectral Bands	224	288	210	200
Data Bandwidth (Gbits/sec) **	25.4	34.6	5.8	72.0
Processing Throughput (Tops/sec) +	2.54	3.46	0.58	7.2

* Characteristics according to trends.

+ Processing throughput calculations for 100 ops/pixel.

** Data Bandwidth calculations assume 30 frames/sec.

References from [2] assuming appropriate pixel/line

Today's hyper-spectral sensor arrays require data bandwidth of 6 to 35 Gbits (0.75 to 4.4 GBytes) per second. This raw bandwidth requirement is sensitive to image resolution, as shown in the futuristic sensor array which will require a data bandwidth 72 Gbits (9 GBytes) per second. Conventional memory buffers and system busses are impractical for these arrays, providing neither the needed storage capacities nor the access speeds required for effective processing. For example, an Ultra 2 SCSI bus provides only a 7.6 MBytes/sec data bandwidth. Current wire and optoelectronic interconnect are nearing 4.0 Gbits/sec/channel for low channel counts.

For simple image processing algorithms such as edge detection, approximately 100 elementary operations (additions and subtractions) must be performed per pixel [7]. Hyper-spectral data streams incorporate an additional wavelength dimension in the data set, which elevate processing demands. Table 1 lists real-time processing demands ranging from 0.5 to 7 Tops/sec for the selected sensor arrays. More complex algorithms require more operations per pixel, and thus, higher processing throughput is necessary. Today's DSP microprocessors can deliver only 0.5 to 1 Gops/sec. Traditional parallel SIMD machines, such as the MASPARI II, provide higher throughput of 68 Gops/sec with 16K processing elements [17][18]. However, these parallel machines are built for scientific computation and are not suitable for stream based computation in hyper-spectral image processing. These machines are not portable, and they consume large amounts of power.

Portable hyper-spectral systems must balance performance and generality while maintaining low power consumption. To alleviate data bandwidth requirements, direct I/O coupling between the sensor arrays and processing elements (PEs) is necessary. Hyper-spectral image processing algorithms must be performed on many parallel PEs to maintain high throughputs. The following section discusses the SIMPiI system, a candidate architecture for hyper-spectral image processing.

3.0 SIMPiI system architecture

The SIMD Pixel Processor (SIMPiI) is a focal plane image processing system that employs area-array I/O to access the processors directly. The SIMPiI design explores the benefits of integrating an image sensor array with a high-performance multiprocessor-computing plane. This monolithic integration of image sensors and digital processing elements is the key-feature of the SIMPiI system. In SIMPiI, the image stream flows directly from the focal plane into the processing plane, retaining its spatial correlation, as depicted in Figure 1.

A block diagram for a 16-bit implementation is illustrated in Figure 2. Each processing element includes a RISC load/store datapath plus an interface to a 4x4 sensor subarray. A 16-bit datapath has been implemented which includes a 32-bit multiply-accumulator unit, a 16-word register file, and 64 words of local memory (the ISA allows for up to 256 words). The SIMD execution model allows for the entire image projected on many PEs to be acquired in a single cycle.

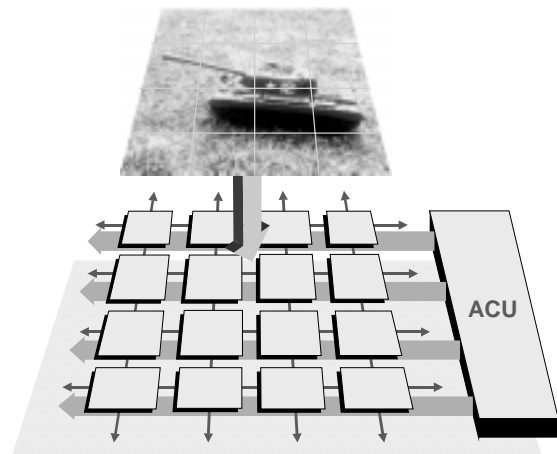


Figure 1: The SIMPiI system. Image streams are optically focussed into the sensor array and mapped onto the processing engine in a single operation.

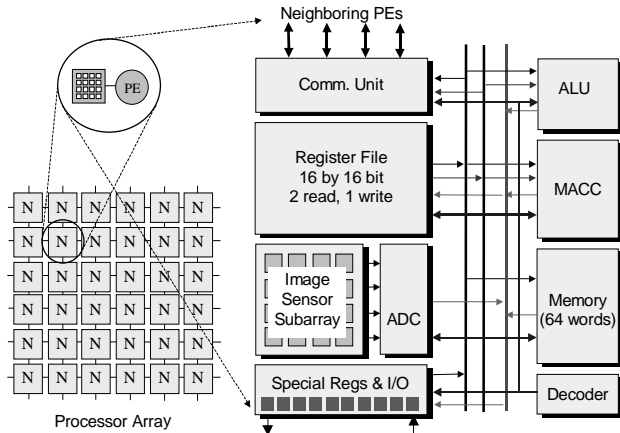


Figure 2. Block diagram of the SIMPil SIMD system

Early prototyping efforts have proved the feasibility of direct coupling of a processing core with sensor devices [4]. Silicon sensors is used to detect light in visible ranges while devices with other materials (InSb, HgCdTe) can be used to detect other spectral ranges. Spectral bands can be selected by placing a band pass filter with the sensors [23]. As the goal of this paper is to describe the SIMPil system capability, sensor integration is not described here but available from [4][24].

A 16-bit prototype of a SIMPil PE was designed in 0.8 μm CMOS process and fabricated through MOSIS. The prototypes were successfully tested and run at 25 MHz. The symbolic layout of the prototype PE is shown in Figure 3. A single PE is estimated to consume about 44.1 mW at 5 V, running at 25MHz over an application workload [5]. Large arrays of SIMPil PEs can be simulated using the SIMPil Simulator, an instruction level simulator, running under Windows95™[24].

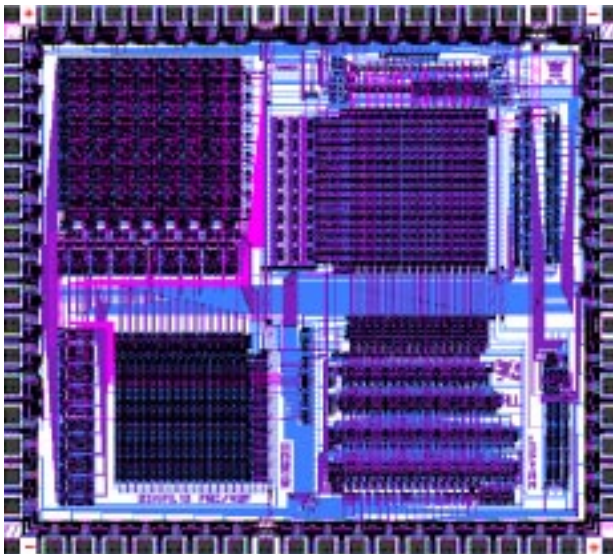


Figure 3. Symbolic layout of a SIMPil16 prototype. The chip measures 2.4x2.7 mm², and it packs about 35,000 transistors in 0.8 μm CMOS process

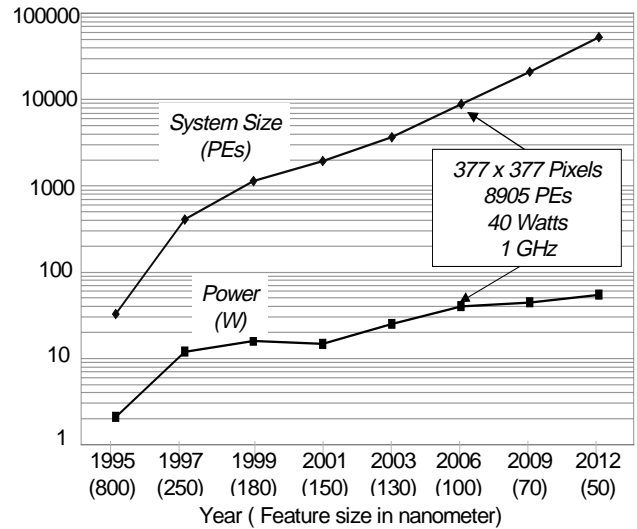


Figure 4. Projected system size and power consumption for a SIMPil Array

A power consumption model has been developed for the SIMPil system [5]. An image processing workload and technology parameters from NTRS [21] have been integrated with this model to project power consumption in different technology scenarios. Maximum system size in a single monolithic chip is determined from maximum die size, transistor density, and power density values for different years. Figure 4 shows the maximum allowable system size and power consumption of the SIMPil system before being limited by technology of the particular year.

Our future target system for performance analysis in this paper is given in Table 2. This SIMPil system is able to deliver an unparalleled performance with 4096 PEs integrated into a single monolithic device running at 500 MHz. For 0.1 μm VLSI technology, a larger system size is projected. However, a smaller, more reasonable system size is chosen because the power consumption model allocates power budget for the digital circuitry. An additional power budget for the interface circuitry must be included to obtain the complete power budget. Lower clock frequency is chosen to maintain low power consumption for the digital circuitry.

In the target system, a 256x256 array of image sensors is integrated with the system, and each PE is directly mapped to a 4x4 pixel sub-array. This system would deliver a peak throughput of about 1.5 Tops/sec in a monolithic device, enabling image and video processing applications that are currently unapproachable using today's portable DSP technology. The following section will present hyper-spectral image processing application development for SIMPil. For this target system, real time processing throughputs are obtained, thus promoting the parallel SIMPil architecture as a valid candidate architecture for hyper-spectral image processing.

Table 2. Technology and system parameters for the target SIMPil system

Target VLSI Technology	0.1 μm
Target Clock Rate	500 MHz
Transistor per PE	35 K
Image Sensor per PE	4x4
System Size	64x64 (4,096 PEs)
Image Sensor Array Size	256x256
Chip Size	800 mm^2
Interconnection Network	Torus
Total Number of Transistors (estimated)	143.4 M
Peak Operation Throughput	1.46 Tops/sec
Estimated Power	9 W

4.0 Hyper-spectral applications on SIMPil

This section describes three hyper-spectral image processing applications developed on the SIMPil system. These applications demonstrate the suitability of SIMPil for the digital battlefield and cover different aspects of hyper-spectral data stream processing. Compression of hyper-spectral data is necessary to reduce the demand on communication bandwidth between deployed platforms and the command center. Region autofocus is an important building block to reduce the amount of data processing in the field of regard. Scene segmentation is then needed to classify the different objects and act appropriately on them. These applications are crucial steps in automated target recognition (ATR) algorithms.

4.1 Image compression using full search vector quantization

Sensor technology, with increasing spectral range and data resolution, will continue to saturate available communication bandwidth. The demand for handling increasingly larger images at faster rates continually exceeds advances in technology and upgrades in the existing communications infrastructure. Efficient data storage and transmission through digital image compression therefore become important to alleviate memory storage limitations and transmission channel bottlenecks. Compression algorithms have been applied for hyper-spectral data stream [3]. The algorithm described in this section is a focal plane implementation on SIMPil.

Among compression algorithms, vector quantization (VQ) has become an attractive alternative to other transform-based techniques [10], mainly because of its computationally inexpensive decoding process. On the other hand, the larger computational requirements of the training and encoding processes are characterized by a very large amount of data parallelism, which makes VQ very suitable for parallel implementations [13][15][16][22].

In its basic scheme, vector quantization is a mapping function which associates vectors of input data to a representative vector (*codevector*), chosen within a previously learnt dictionary (*codebook*). A scalar index is then used to reference the chosen codevector in the dictionary and encode the input vector. The compression ratio depends on the cardinality of the codebook, usually much smaller than that of the input domain. In the 2D case, non-overlapping vectors are extracted from the input image by grouping a number of contiguous pixels, in order to retain available spatial correlation of the data. Hyper-spectral data streams require an extension of vectors into multiband cubes of spectral images. Non overlapping, multiband cubes are used to exploit both the spatial and the spectral correlation available in the data.

A full-search VQ encoding algorithm was implemented on the SIMPil architecture [9] for 2D image data. A 256-word codebook is used to achieve a 0.5 bit per pixel encoding of 8-bpp gray-level images. SIMPil offers a natural way to implement the encoding operation of a vector quantizer. Two levels of parallelism are exploited in this implementation. Distributing the codebook, where each processing element represents one codeword, provides one level. A second level of parallelism is provided by processing multiple input blocks in parallel.

The application moves input blocks through the system, so that each input block is compared against the entire codebook. When the process completes, each node contains the index of the best matching codeword for the original input block, and the corresponding distortion value. Through this process, the focal plane image is converted into a matrix of 8-bit values. The pseudo-code of the algorithm is given in Figure 5, along with a pictorial description of the parallel full search, for a 16-node SIMPil system. A key enabling role is played by the toroidal structure of the interconnection network, which enables the communications among the nodes in the opposite sides of the mesh. In this implementation, the input blocks are compared against the codebook in a systolic fashion, with a large number of them compared at any given time in parallel. This results in a large speedup and in a high degree of concurrency.

```

REPEAT for all rows
  REPEAT for all column
    Transfer input block, current_distortion, and current_index
    to the eastern node;
    Calculate local_distortion between input block and local
    codeword;
    IF local_distortion < current_distortion THEN
      current_index = local_index;
      current_distortion = local_distortion;
    END if
  END repeat
  Transfer input block, current_distortion, and current_index to the
  southern node;
END repeat
Output current indexes;

```

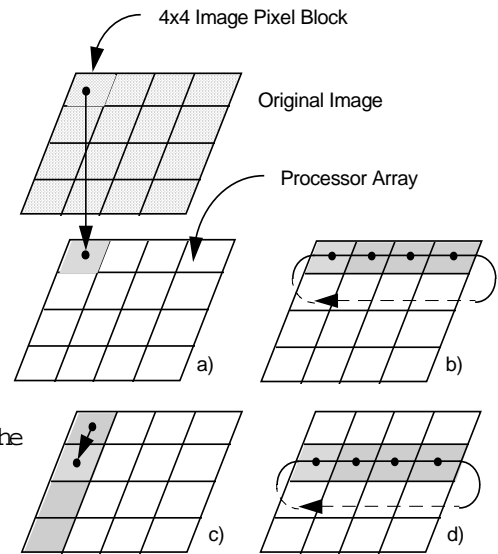


Figure 5: Parallel encoding algorithm, and communication patterns for the full search on the codebook.

4.2 Region autofocus

Region autofocus isolates regions of interest and provides a smaller image for further analysis. Irrelevant portions of the image are ignored, and system resources are dedicated to the proper region for threat identification. Consequently, workload and execution latency of Automated Target Recognition (ATR) algorithms are reduced by removing the need to process the entire hyper-spectral data stream.

The approach is illustrated in Figure 6. Hyper-spectral imaging sensors generate a data cube, containing a large number of images at different wavelengths. While threat visibility may be obstructed in the visible spectrum, other characteristics, such as heat or material signatures, can be detected at a different spectral wavelength, as in the sample image on the right in Figure 6.

Data fusion between the different spectral slices is first performed to detect threat presence. This is achieved by correlating image coordinates in different spectrums and producing a binary image. A pixel in the binary image is set if spectral signatures are detected at the given coordinates. This binarization (threshold) process is simplified for the SIMPil system because image pixels in the focal plane are spatially mapped to processors. Each PE handles the same spatial coordinates for each spectral image. In other words, each PE operates on a small tile of pixels with the same spatial coordinates. A stack of tiles with the same spatial coordinates, representing the different spectral slices, is stored in the same SIMPil PE.

More complex data fusion processes can be substituted to identify regions with the presence of desired spectral signatures [12]. The implemented data fusion processing stage uses the spectral signatures to determine the presence in a given spectral image. Shape and textural information can be included to further identify threat presence.

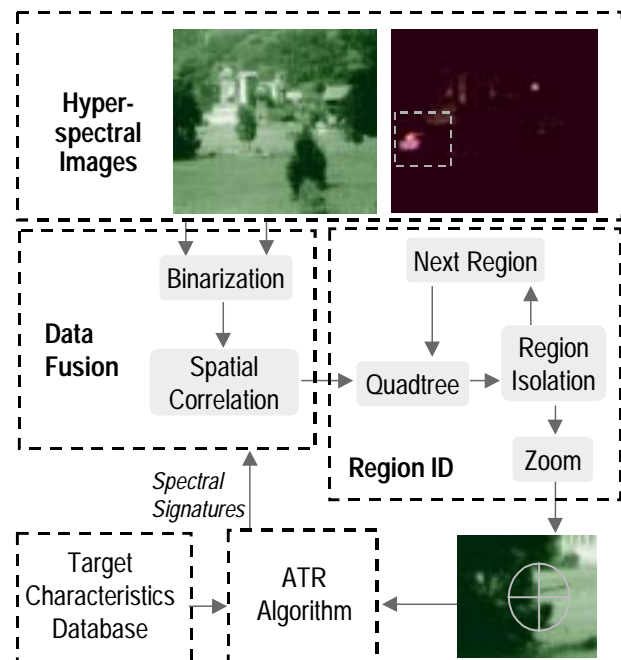


Figure 6. Processing steps for Region Autofocus to isolate regions of interest

A region identification processing stage is then utilized to locate a region of the view with the threat. A quadtree decomposition algorithm is used to subdivide the hyper-spectral image into smaller image quadrants. While quadtree can be used for image compression, it is used in this application context only to segment the original images into smaller isolated regions. Each region of interest can be identified by parsing the generated quadtree structure.

The region of interest is subsequently enlarged or zoomed to focus on the threat by redistributing pixel data of a single region among the SIMPil PEs. The enlarged image contains upsampled pixels from a single region of interest. This zoomed image is fed to ATR algorithms to further assess the effective presence of a threat.

These ATR algorithms take target information from a central database and inform the region autofocus algorithm of the proper spectral signatures. By redistributing pixel data, each SIMPil PE is ready to perform an ATR algorithm as if this smaller region was originally sampled from the sensor array. Therefore, no computational bandwidth is lost on regions not containing the threat. In addition, ATR algorithms can use information from the region autofocus application to refocus the sensor view towards the target.

The following paragraphs describe in detail the quadtree decomposition used in the region autofocus application. Performance numbers for region autofocus are obtained only for the data fusion and region identification processing stages. A complete ATR algorithm stage is being investigated and is not included in this analysis since the region autofocus application is a separate entity utilized to reduce image sizes for ATR algorithms.

Quadtree decomposition algorithm is used to subdivide the hyper-spectral image into smaller image quadrants. The recursive decomposition can be represented in a tree, as shown in Figure 7. At the top level of the tree (root), the decomposition starts. The root corresponds to the entire binary image. It is connected to four child nodes, which represent each quadrant from left to right and the NW, NE, SW, and SE quadrants respectively. If the child is gray, it is again connected to four other children. The tree will continue until the quadrants do not need any further subdivision. In the example, the NE, SE, and SW 2x2 pixel quadrants are regions of interest because the child nodes at the second tree level are either gray or black. The NW region is entirely white, which represents a region without the presence of the desired spectral signatures.

The essential component of this region autofocus application is the quadtree implementation. The processing stage maintains a high sustained utilization of 99.75% because of its communication scheme. The tree representation, shown in Figure 7, requires regular, many to one, communication patterns among PEs.

The implemented communication scheme, depicted in Figure 8, removes data broadcast and maintains minimal buffer storage. In the beginning of the processing steps, all

PEs are active to compute the tree representation for the subset of image pixels spatially mapped to them from the focal plane. Partial data is then collected in designated reference PE for each level of the quadtree, and the process is iterated until the complete tree description is aggregated. Notice that this quadtree decomposition begins at the leaf of the quadrant tree structure and continues towards the root. Other implementations will exhibit different activity patterns, but may not result in efficient PE utilizations.

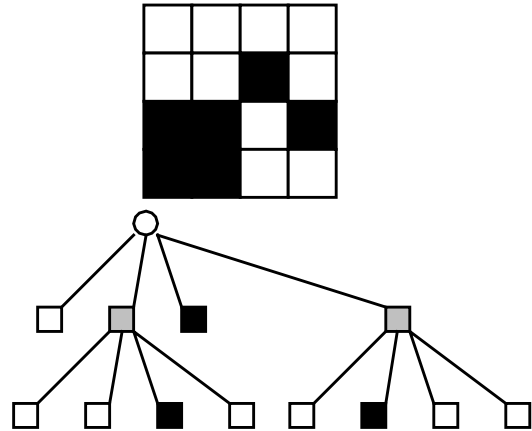


Figure 7. Example Binary 4x4 image and its quadtree representation

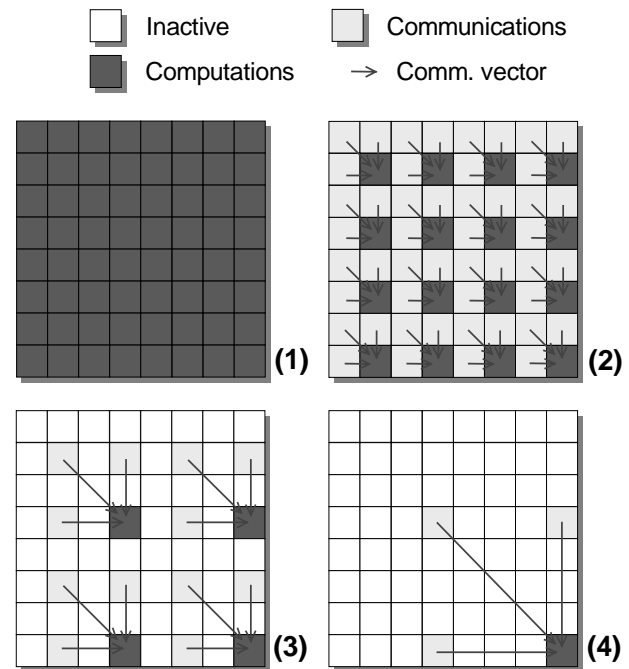


Figure 8. Communication patterns for the quadrant tree stages (1-4). Computation squares represent PEs operating on a quadtree node. The last computation square in (4) is the root of the quadrant tree.

4.3 K-Means clustering

K-means clustering is a commonly used technique to segment large image regions into specific objects or areas of interest. This sort of segmentation can, for example, be useful in search and rescue operations, where large areas must be scanned for specific objects. Hyper-spectral sensors can be used to obtain a clear signature of the objects of interest. K-means clustering is the spatial approach of the C-means algorithm used for hyper-spectral data classification.

The implemented algorithm takes as input the data-fused image to be analyzed and the number of clusters to be constructed (K). The process is illustrated in Figure 9. The algorithm first identifies all possible pixel clusters in an image according to a particular threshold metric. This is accomplished by having each PE compare its assigned pixel values to the specified threshold value. All pixels that satisfy the threshold condition (“live” pixels) are grouped into N clusters based on a connectivity criterion, and then the clusters are labeled accordingly. The toroidal interconnection network in SIMPil allows this process to be performed in a very efficient manner, as all PEs with live pixels can communicate with their neighbors in order to grow a cluster of pixels. The constructed clusters are labeled using a simple convention based on the identification numbers of all PEs with pixels belonging to a particular cluster. The PEs can then collaborate in each centroid computation of the N identified clusters.

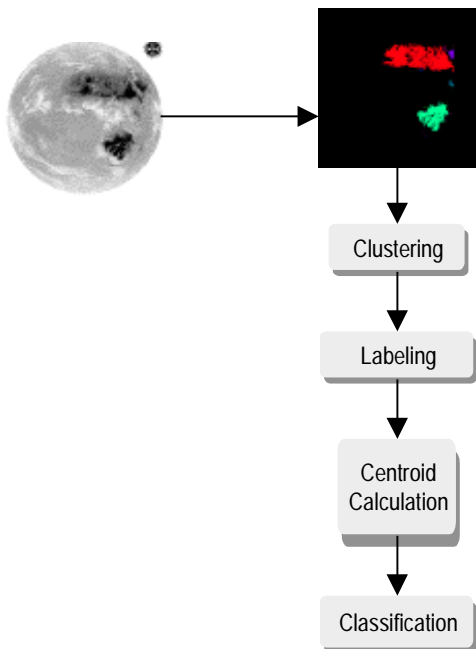


Figure 9. Processing steps for the K-means clustering algorithm.

Once the centroids for the N ($N > K$) clusters are determined, the algorithm must map these clusters into K new clusters. To do this, the algorithm chooses K clusters out of the N identified and maps the remaining $N-K$ clusters onto it. The centroids of these K clusters are denoted as seeds. The remaining $N-K$ clusters are mapped by minimizing the distance between their centroids and selected seeds. The implemented algorithm chooses the seeds randomly. After each one of the $N-K$ clusters is remapped, the centroid of the newly expanded cluster is calculated. This new centroid substitutes the original cluster’s centroid as one of the K seeds. The algorithm continues until no centroid changes cluster. At the end, there will be K main clusters that, taken together, will hold the N original clusters. Each of the K clusters will have a specific centroid calculated using all internal clusters. An example of this process is shown in Figure 10, for the cases where $N=7$ and $K=2$ and $K=3$. Note that the clusters generated will depend on the choice of the seed points.

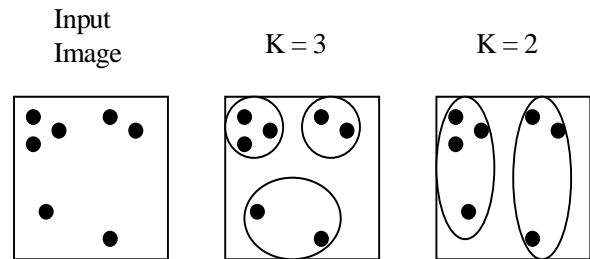


Figure 10. K-Means mapping example. The dots represent the N cluster centroids.

The K-Means clustering algorithm serves as the initial processing step in ATR algorithms to reduce large hyper-spectral images into manageable clusters. Like the region autofocus application, ATR algorithms take the K-Means cluster information along with target characteristics for further identification. To enhance this application, a new algorithm for choosing optimal seed points is currently being investigated. In addition, a C-means clustering algorithm using 220 spectral bands is also being developed.

5.0 Performance results

The applications described in the previous sections have been developed and simulated on the SIMPil Simulator [24]. Table 3 summarizes performances for all the implemented applications. The target system, described in Table 2 with 4096 PEs, was used for the simulations. This system size supports an image size of 256x256 pixels with a 4x4 pixel subarray per PE. For each application, the worst case scenario was used.

Table 3. Hyper-spectral application performances

Application	Image Size	Average Utilization (%)	Sustained Throughput (Gops/s)	Total Cycles	Execution Time (ms)
Vector Quantization	256x256	91.7	1,878	81,774	0.164
Region Autofocus	256x256	82.2	1,680	49,561	0.099
K-Means Clustering	256x256	66.2	1,354	79,827	0.160

System utilization is calculated as average concurrency, and it is relative to the total system size. Execution time and sustained throughput are computed with reference to the 500 MHz target platform described in Section 3.0. The simulations show that all applications execute in lower millisecond range, suggesting that they can be combined in stages to form complex applications, which are still able to execute at full frame rates (30-60 fps, or 15-30 ms). The high system utilization achieved further indicates that the underlying data bandwidth and parallel execution scheme in SIMPil are suitable for hyper-spectral image processing applications.

Traditional SIMD systems such as the MASPARI II delivers 68 Gops/s processing throughput with 16K PEs running at 12.5 MHz [17][18]. If a 40-fold increase in clock frequency alone is assumed, an estimated 2.7 Tops/s processing capability may be possible. However, data bandwidth in the global router (1.2Gbit/s) may easily limit the estimated performance. Furthermore, these systems typically rely on a “push-broom” computation strategy because of their limited I/O capability.

The SIMPil architecture benefits from direct coupling between sensor array and computing plane. The focal plane scheme allows stream-oriented mapping of image data directly with SIMPil PEs. Image processing applications and algorithms map well to the SIMPil architecture and its SIMD execution model [8]. With only 4096 PEs, SIMPil delivers competitive performance (0.5 to 1.5 Tops/s). This is achieved by keeping PEs busy since the focal plane I/O delivers spatially mapped data for each PE.

The focal plane I/O also alleviates the need to buffer the large amount of hyper-spectral image data. Computation is performed as the data arrives, rather than storing the entire data stream first. In comparison, traditional single processor architectures require large memory caches to hold stream data. These caches are not efficiently used because every stream element is read exactly once [6]. Instead of placing more caches in the chip, more PEs can be substituted to allow higher image resolution and processing throughput.

Other image processing applications such as JPEG and wavelet encoding are currently being developed. Standard building blocks for image processing algorithms, such as spatial filtering, DFT, morphological filtering, image

rotation and labeling, have been implemented [8]. More complex ATR algorithms are being pursued by integrating various components such as region autofocus with textural correlation.

Architecture models in [5] have shown that VLSI technology by the year 2012 supports a SIMPil system with approximately 850K-pixel (920x920). Power consumption for PEs is contained below 50 Watts for a 52,900 processor system in 50nm technology, with a projected performance in excess of 70 Tops/s. The above application simulations show the potential of processing hyper-spectral data streams on a parallel SIMD environment in a focal plane. The SIMPil architecture provides the suitable operating platform for current and future semiconductor technology.

6.0 Conclusions

Three applications to process hyper-spectral data streams have been presented in this paper: full search vector quantization, region autofocus, and K-means clustering. These applications have been implemented on SIMPil, a focal plane SIMD architecture being developed at Georgia Tech. Focal plane processing for hyper-spectral imaging systems meets the requirements for digital battlefield operations, providing both the computational and I/O capabilities, while maintaining low power consumption and portability. Simulated performance shows the potential for SIMPil to deliver real-time processing for hyper-spectral imaging.

7.0 Acknowledgements

The work was supported by the ARL (Contract: DAKF11-97-D-0001-013), AFOSR, Defense Advanced Research Projects Agency (Low Power Electronics Contract: FY1123-95-08217), and the National Science Foundation/Georgia Tech Packaging Research Center (Contract: EEC-9402723). We would like to thank the anonymous reviewers for their valuable comments on earlier drafts of the paper.

8.0 References

- [1] K. E. Batcher, "Design of the Massively Parallel Processor", *IEEE Transaction on Computer*, C9, vol. 9, pp.836-840, 1980
- [2] R. J. Birk, T. B. McCord, "Airborne Hyperspectral Sensor Systems," *IEEE Aerospace and Electronic Systems Magazine*, vol. 9, no 10, pp. 26-33, Oct 1994.
- [3] S. D. Briles, "Real time onboard hyperspectral image compression system for a parallel push broom sensor", *Algorithm for Multispectral and Hyperspectral Imagery III*, SPIE vol. 3071, A. Evan Iverson and Sylvia S. Shen, editors, pp. 182-190, April 1997.
- [4] H. H. Cat, et.al. "SIMPil: An OE Integrated SIMD Architecture for Focal Plane Processing Applications", *Massively Parallel Processing using Optical Interconnection (MPPOI-96)*, pp.44-52, 1996
- [5] S. M Chai, A. Gentile, D. S. Wills, "Impact of power density limitation in Gigascale Integration for the SIMD Pixel Processor", in *Proceedings of the 20th Anniversary Conference on Advanced Research in VLSI*, Atlanta, Georgia, pp. 57-71, 1999
- [6] K. Diefendorff and R. Dubey. "How Multimedia Workloads Will Change Processor Design", *IEEE Computer*, vol. 30, no. 9, pp.43-45, September 1997.
- [7] E.S. Eid and E. Fossum, "Real-time focal-plane array image processor", *Proceeding of The International Society for Optical Engineering*, vol.1197, pp.2-12, 1989
- [8] A. Gentile, et al. "Real-Time Image Processing on a Focal Plane SIMD Array", in *Parallel and Distributed Processing - Lecture Notes in Computer Science* (Eds. Jose' Rolim et al.), vol. 1586, pp. 400-405, Springer Verlag, New York, 1999
- [9] A. Gentile, H. H. Cat, F. Kossentini, F. Sorbello, and D. S. Wills, "Real-Time Vector Quantization-based Image Compression on the SIMPil Low Memory SIMD Architecture", in *Proc. of 1997 IEEE International Performance, Computing, and Communications Conference*, Phoenix/Tempe (AZ), pp. 10-16, 1997
- [10] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publisher, 1992
- [11] E. Gose, R. Johnsonbaugh, S. Jost, *Pattern Recognition and Image Analysis*, 1996 Prentice Hall PTR, Upper Saddle River, New Jersey 07458, Ch 5.
- [12] H. N. Gross, J. R. Schott, "Evaluating an image fusion algorithm with synthetic-image-generation tools", *Algorithms for Multispectral and Hyperspectral Imagery II*, SPIE vol. 2758, A. Evan Iverson, editor, pp. 136-147, April 1996.
- [13] T. B. Henderson, and K. S. Thyagarajan, "Implementation of VQ Algorithms on a Reconfigurable Array Processor", *Professional Paper AD-A236 483/4/XAB*, Naval Ocean Systems, San Diego (CA), May 1991
- [14] W. D. Hillis, "The Connection Machine", *The MIT Press*, 1985
- [15] H. J. Lee, J. C. Liu, A. K. Chan, and C. K. Chui, "A Parallel Vector Quantization Algorithm for Single-Instruction-Multiple-Data (SIMD) Multiprocessor Systems", in: *Proceedings of the 5th Data Compression Conference*, Snowbird (UT), pp. 479, March 1995
- [16] M. Manohar, J. C. Tilton, "Progressive Vector Quantization of Multispectral Image Data using a Massively Parallel SIMD Machine", *Data Compression Conference*, Snowbird (UT), pp. 181-186, 1992.
- [17] MasPar (MP-2) Users Guide, MasPar Corporation, Rev. A5, 1994
- [18] MasPar (MP-2) System Data Sheet, MasPar Corporation, 1993
- [19] J. Miller, "Sensor Technology for Army After Next (AAN)", Workshop on Image Processing Applications for the Digital Battlefield, Clark Atlanta University, September 14-15, 1998.
- [20] J. R. Nickolls, "The Design of the MasPar MP-1: A cost-effective Massively Parallel Computer", *IEEE Digest of Papers - ComCom*, pp.25-28, 1990
- [21] Semiconductor Industry Association, *The National Technology Roadmap for Semiconductors*, 1997.
- [22] K. S. Prashant, V. J. Matthews, "A Massively Parallel Algorithm for Vector Quantization", *Proceedings of 1995 NASA Space and Earth Sciences Workshop*, Salt Lake City, Utah, March 1995.
- [23] J. Schott, *Remote Sensing: The Image Chain Approach*, Oxford University Press, New York, pp.125-188, 1997.
- [24] PICA research group, SIMPil Home Page, <http://www.ee.gatech.edu/research/pica/simpil>