Analytical Study of Parallel and Distributed Image Processing

Harshad B. Prajapati
Information Technology Department
Dharmsinh Desai University
Nadiad-387001, Gujarat, INDIA
harshad.b.prajapati@gmail.com

Dr. Sanjay K. Vij
Director, Sardar Vallabhbhai Patel Institute of Technology
Sardar Vallabhbhai Patel Institute of Technology
Vasad-388306, Gujarat, INDIA
vijsanjay@gmail.com

Abstract— **The available literature on parallel and distributed image processing is scattered and not organized for use to beginners. Thus, there is a need of concise understanding of parallel and distributed image processing area. In this paper, we present analysis of parallel and distributed image processing with comprehensive details, so that it becomes very useful to beginners and to those who are new to parallel or distributed image processing field. We present the outcome of our study of parallel and distributed image processing with emphasis on mechanisms, tools/technology/API used, application domains, and ongoing research work. We examine the research issues in parallel and distributed image processing. We also identify some future research directions for distributed image processing. This study provides concise understanding of the parallel and distributed image processing area to the beginners.**

Keywords- ***Parallel Image Processing; Distributed Image Processing; Data Parallelism; Task Parallelism; Parallel Image Processing Tools; Distributed Image Processing Systems.***

I. Introduction

The concept of "*Using more than one computation resource for solving certain time consuming problems*" seems more interesting and valid. However, coordination among multiple computation resources and intelligent work distribution among them place major challenges on system designer. The literature on parallel and distributed image processing is available elsewhere, however, it is scattered and not organized for beginners. Thus, there is a need of concise understanding of parallel and distributed image processing area. We try to fulfill this need by providing analytical study of parallel and distributed image processing area in this paper.

Our objectives in this paper are as following. 1) Explore the parallel and distributed image processing in concise and simple way. 2) Present the study of parallel and distributed image processing with emphasis on mechanisms, tools/technology/API used, application domains, and ongoing research work. 3) Motivate and direct the readers for further research work in parallel and distributed image processing.

Our study discusses about how parallelism is possible in image processing. We provide brief details on "What is the need of parallel image processing". We discuss level of image processing operators and show that how parallelism is possible in these operators. We also briefly discuss about both parallel hardware and distributed system. We also provide study on tools/technology/API that are used in research and development of parallel and distributed image processing. Moreover, different application domains are also presented and related research is also explored in this analytical study.

The work in [1] presents overview of data parallel image processing routines. This work presents a number of low-level routines and their data parallel implementation. Our work differs from their work on data parallel image processing, as our focus is broad and concise and not just limited to data parallel image processing.

The paper is structured as follows. Section II discusses on what is the need of parallelism in image processing. Section III discusses on how parallelism can be applied in image processing. Section IV presents architecture for achieving parallelism. Section V provides analysis of applied parallel and distributed image processing. Section VI highlights our ongoing research work in distributed image processing. Finally, section VII concludes our work.

II. Need for Parallelism in Image Processing

In the last decade there has been tremendous growth of multimedia data attributed by various social networking sites. Thus, many large image-bases (database of images) exist in the Internet. Many search engines, e.g. Google, provide keyword based searching of images, however, the search results are based on only captions of images. In near future, we expect to have search facility based on image content. However, such kind of image processing, creating index on images based on their content, is really a challenging task. This search facility would have vast applicability, e.g., image search to detect unethical copy of image, which is at present not supported by search engines. Traditional approach of centralized computing does not work on such large image-bases.

There has been tremendous increase in *business through Internet*. To provide online view of product and sophisticated queries on product, it is required to have on the fly generation of image of the product with different preferences, e.g. car with red color and 360 degree view from inside car, supplied by

978-1-61284-861-7/11/$26.00 ©2011 IEEE

user. Response to such complicated queries can be generated immediately by parallel or distributed image processing.

High speed image processing on image sequences, recorded images of scientific experiments, and images/videos captured through surveillance system is really challenging and complex. Real-time image processing is employed in many applications such as multimedia applications, industrial inspection, medical, textile industries, etc. In real-time image processing, processing should be done with high speed on rapid sequence of images or on single image. Such high speed image processing can be handled using parallel or distributed image processing.

An image processing is not just limited to above mentioned applications. It has many applications in science and engineering such as bio-medical science, manufacturing and industry automation, space, agricultural and geological science, etc. On comparing ability of human and computer for processing of image, we find that a human brain can process images very fast because of its ability to generalize and infer knowledge from image data and large number of neurons working in parallel, on the other hand, the computerized image processing has many limitations compared to human brain. At present most of the applied image processing [2] works deal with enhancement, restoration, and segmentation of the image. While the image interpretation and knowledge inference still require human involvement. However, due to advancement in Artificial Intelligence [3], parallel computing, and network based computing [4], the solution of above is possible with intelligent image processing with parallel and distributed system architecture.

Looking at the architecture of standalone computer, it is found that standalone computer has its own limitation in speed and memory beyond which it cannot be extended in present architecture. Moreover, the computation demand is always found ahead of computational performance. The current computer hardware technology has reached at its limit as no substantial improvement in CPU speed has been observed since last few years. As more sophisticated applications are demanded by users, less processing time and faster response are required from applications. With e-information explosion, the computational load has increased manifold and possible solution is parallel and distributed processing.

III. Analysis of Parallelism in Image Processing

In this section we show how parallelism is possible in image processing. We first briefly discuss about image processing operators, then we briefly mention about what is parallel processing. We then show how parallelism is applied to image processing applications.

A. *Image Processing Operations*

The image processing [2] is done at following levels.

1. Low level image processing
2. Intermediate level image processing
3. High level image processing

1) Low Level Image Processing: The low level image processing operators operate at the pixel level. The input to low level image processing operators is an image and output is image or data. Few examples of low level image processing operators are contrast enhancement, noise reduction, and noise removal in image. They are also used for edge detection and various image transformations. In some applications, they are used to calculate characteristics of the input image (contours, histograms).

The low level image processing operations [5] can be divided into point operators, local neighborhood operators, recursive neighborhood operators and global operators. The point operator generates output pixel for corresponding input pixel based on that corresponding input pixel only. The point operators can be further divided into monadic operators, dyadic operators, and triadic operators. The monadic operators perform arithmetic and logical functions between pixels from image source and a constant value. The dyadic operators perform arithmetic and logical functions between two pixels from two image sources. The triadic image operators perform function on three pixels from three image sources. Local neighborhood operators create output pixel based on corresponding pixel and pixels surrounding corresponding pixel in input image. They are used for enhancing and changing appearance of images by sharpening, blurring, crispening the edges; and noise removal; Global operators generate output pixel based on all the pixels of the input image. Examples of global operators are Discrete Fourier Transform (DFT) and Histogram transform.

2) Intermediate level image processing: The intermediate level image processing operations operate on abstractions derived from the pixels of the image. These processing operations derive abstractions from the image pixels (region labelling, object tracking) so that it can help in further decision making about image. Examples are region labelling, object tracking, image measurement, perceptual grouping, etc.

3) High level image processing: The high level image processing operations operate in order to generate higher abstractions. They work on abstractions derived from intermediate-level image processing operators. They are used to interpret the image content. (e.g., classification and object recognition). These operations work on graphs, lists, relations among regions/objects to derive some decision.

B. *Parallelism*

In simplest words, the parallelism is simultaneous processing by two or more processing units. The parallelism can be achieved by one of two main approaches: (1) using parallel processing hardware and (2) using distributed computing/processing system. In the former approach, a computer system having multiple processors is used to accomplish computation task, while in latter approach, number of machines connected in network are used to accomplish computation task. Both approaches have their pros and cons. However, depending upon the requirement of the application and available budget, the selection of architecture is done.

C. *Applying Parallelism to Image Processing Applications*

The parallelism can be applied in image processing applications by three main ways: i) Data Parallel, ii) Task Parallel, and iii) Pipeline Parallel. We discuss them in this section.

1) Data Parallel: In data parallel [5] approach, the data is partitioned and distributed among the computing units. The major research challenge is efficient data decomposition and result composition. Two issues must be considered for efficient parallel execution: (i) data locality and (ii) load balancing. Image data should be distributed to processing units in such a way that there will be less unnecessary communication among processing units. Image data should be distributed among processing units in such a way that each processing unit gets approximately same load. At present the distribution of blocks of image is limited to parallel computers, which have multiple processing units. However, the classical approach of data parallelism is included here, as in future, the "network is a computer" may get conceived practically.

The data parallelism to image data can be applied using one of three basic ways: i) Pixel Parallel, ii) Row or Column Parallel and iii) Block Parallel. At present, the most of the parallel image processing applications use row/column parallel or block parallel approach rather than pixel parallel.

While performing parallel image processing, the image data can be distributed to available processing units using one of following distribution approaches. The main aim of data distribution approach is to achieve proportionally equal load distribution. These data distribution approaches are as following: (1) Row-stripe distribution, (2) Column-stripe distribution, (3) Block distribution (See Figure 1), (4) Row-cyclic distribution, (5) Column-cyclic distribution, (6) Window-wise distribution, and (7) Helicoidal distribution [5].

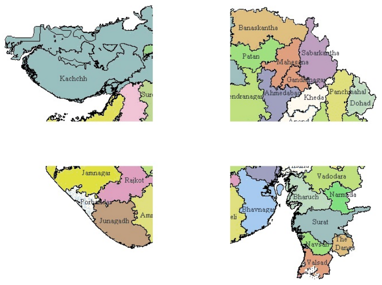
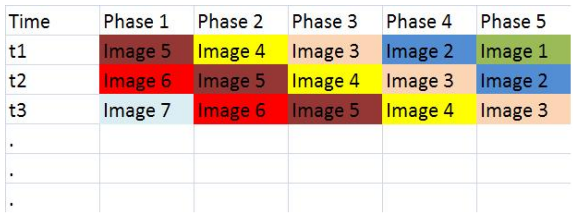


Figure 1. Division of map into four blocks for parallel processing using Block Parallel Approach

2) Task Parallel: In task parallel [5] approach, image processing instructions/low-level operations/sub-operations are grouped into tasks and each task is assigned to a different computing unit. An image processing application consists of many different operations. Some of these operations are independent, i.e., result of any operation does not depend on the result of other operations. The major research challenge in task parallel approach is efficient task decomposition and result composition.

3) Pipeline Parallel: Certain image processing algorithms consist of consecutive stages. If an image processing application requires multiple images to be processed, then pipeline processing of images can be done. In pipeline processing, images will be in different stages at the same time (See Figure 2.).



Time	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5
t1	Image 5	Image 4	Image 3	Image 2	Image 1
t2	Image 6	Image 5	Image 4	Image 3	Image 2
t3	Image 7	Image 6	Image 5	Image 4	Image 3
.					
.					
.					

Figure 2. Pipeline Parallel Image Processing

IV. Architecture for Achieving Parallelism

There are two major architectures of achieving parallelism: i) by using parallel processing hardware (tightly coupled system), and ii) by using distributed computing system (loosely coupled system). The tightly coupled parallel processing is architecture with common memory with shared bus which has many limitations [6]. With increase of network throughput (in terms of Gbps), it is possible to have parallel processing between networked computers. There are two basic mechanism of exchanging data among processing units: (i) through message passing and (ii) using shared memory. Here, we briefly discuss about the basic architecture of both parallel processing hardware and distributed system.

A. *Parallel Processing Hardware*

In parallel processing hardware, more than one compute resource solves a computation problem simultaneously. Based on the memory usage by different CPUs, the parallel processing hardware can be classified as i) Shared memory parallel computer architecture and ii) Distributed memory parallel computer architecture. A parallel solution of a problem can use shared memory, multi-threading, or message passing as programming model. Based on Flynn's classification of high-speed computers [7] [8], parallel processing hardware can be classified into four types namely i) Single Instruction, Single Data (SISD), ii) Single Instruction, Multiple Data (SIMD), iii) Multiple Instruction, Single Data (MIMD), and iv) Multiple Instruction, Multiple Data (MIMD) [9].

B. *Distributed System*

There are two main architecture of distributed system: i) Master slave, and ii) peer-to-peer [4]. These are discussed below.

1) Master Slave: The master-slave architecture approach uses the "Distribute Compute and Gather" philosophy for parallel image processing. In this architecture approach, the master processing unit splits and distributes the image data to the slave processing units. All slaves processing units work in parallel to achieve assigned task. Then master processing unit gathers and assembles the image back.

2) Peer-to-Peer: In peer-to-peer architecture, each participating entity has the same capabilities and either entity can initiate a communication. The participating entities make a portion of their resources directly available to other networked participating entities, without the need for central coordination (such as master or controller in Master slave architecture).

For each approach of parallelism in image processing, the suitable mechanism for efficient parallel processing are shown in Table I.

TABLE I. SUITABLE MECHANISMS FOR DIFFERENT APPROACHES OF PARALLELISM IN IMAGE PROCESSING

Approach of parallelism in Image processing	**Suitable mechanism for efficient Parallel Processing**
Data Parallel	Parallel computing on parallel hardware
Task Parallel	Distributed computing
Pipeline parallel	Distributed computing on low latency clusters

V. ANALYSIS OF APPLIED PARALLEL AND DISTRIBUTED IMAGE PROCESSING

In this study, we present the analysis of applied parallel and distributed image processing. We preset analysis of tools that are used for parallel and distributed processing. In this analysis, the readers can get idea about which tool is suitable for which kind of parallelism. We also present the application domains in which parallel and distributed image processing is possible. We also present analysis of research work in parallel and distributed image processing.

A. Tools and Technology

We present here most widely used tools/technology/API for parallel and distributed image processing.

Message Passing Interface (MPI) [10]: It is language independent specification of communication protocol for parallel programs running on a distributed memory system. The MPI emphasizes on portability, scalability and performances.

OpenMP [11]: It is de-facto standard API for developing shared memory parallel applications in C,C++, and Fortran languages. It is generally used to explicitly specify the parallelism in the program, e.g. when the compiler cannot find the parallelism. OpenMP is suitable for multi-core architectures. The reader is directed to [12] to know more about multi-core image processing using OpenMP.

Graphical Processing Unit Programming–CUDA [13]: Compute Unified Device Architecture (CUDA) is Nvidia's parallel computing architecture. The CUDA is the computing engine in Nvidia graphics processing units (GPUs). It can increase computing performance by harnessing the power of the GPU. Developers can access it through various standard programming languages. Scientists and researchers use CUDA for image [14] and video processing, simulation of fluid dynamics, computational biology, seismic analysis, medical analysis simulation, etc.

Cilk (Commercial version is available as Cilk++) [15]: Cilk is an ANSI C based general-purpose programming language designed for multithreaded parallel computing. The key design principle of Cilk is "programmers should be concerned with exposing parallelism, while the runtime environment should be concerned with dividing the parallel work among available processors."

Threading Building Blocks (TBB, Intel TBB) [16]: It is a C++ template library designed for parallel programming on multi-core processor architectures. It hides manual approach of thread creation, synchronization, and termination such as found in thread packages/library. In TBB, a program is described in terms of fine-grained tasks, which can have dependencies. The tasks are allocated to individual cores at runtime by library and they are executed according to dependencies. Refer [17] for application of Intel TBB.

MATLAB's Parallel Computing Toolbox (PCT) with MATLAB Distributed Computing Server (MDCS) [18]: The MATLAB has very good support for array and metric processing. Using PCT, the MATLAB can allow solving image processing problems using multi-core processors, GPUs, and computer clusters. To support this capability, high-level constructs, e.g. parallel for-loops, special array types, are supported, which allow us to parallelize MATLAB applications without CUDA or MPI programming.

The summary of all presented tools and their applicability for parallel and/or distributed processing are listed in Table II.

TABLE II. TOOLS AND TECHNOLOGIES USED IN PARALLEL AND DISTRIBUTED IMAGE PROCESSING

Tool and Technology	**Brief detail of Tool and Technology**	**Support for Parallel and/or Distributed Processing**
MPI	Specification of Communication Protocol. It is language and platform independent.	* Parallel processing * Distributed computing on low latency cluster
OpenMP	De-facto standard API for shared memory parallel applications	Parallel processing
CUDA	Computing engine in Nvidia graphics processing unit.	Parallel processing
Cilk	general-purpose programming language for multithreaded parallel computing	Parallel processing
TBB	C++ template library for parallel programming on multi-core processor architecture	Parallel processing
MATLAB's PCT with MDCS	Parallel and distributed environment for MATLAB programs. It has also support for scheduling.	* Parallel processing on *multi-core processors* and GPU * Distributed computing on clusters

B. *Application Domains*

We present list of few important application domains in Table III. In all these application domains, parallel and/or distributed image processing is applicable.

TABLE III. LIST OF FEW IMPORTANT APPLICATION DOMAINS

Application Domain	**Principal Image processing Operations Involved**	**Example Usecase**
Document Image Analysis	* Optical Character Recognition * Layout Analysis * Pre-processing of scanned documents (for skewed and tilted pages)	Automated document classification
Surveillance System	Object Identification	Security, Monitoring
Geographical Processing	Region matching and inference	Crop prediction, Transport Navigation
Bio-medical Image Analysis	* Template matching * feature extraction	Predication of a possible decease in future
Content based image retrieval	Template matching	search images that contain human being
Image Mining	* Template matching * association analysis	search residential areas near by river, lake
Optical Character Recognition	* Noise removal * image orientation * character separation	Image to text conversion

C. *Research work in Parallel Image Processing*

We studied research papers related to parallel image processing and present analysis of these papers in Table IV. The reader can get detailed idea about the research work under parallel image processing by referring those research papers.

TABLE IV. MAIN RESEARCH WORK TOPICS IN PARALLEL IMAGE PROCESSING

Topic	**Brief detail of work**
Image partitioning	Partition of image to achieve desired objectives (balance load, optimize processing time). E.g. see work in [19]
Parallel Processing Language	Design of a language/annotation elements to make parallelizable code. E.g. see work in [20] [21]
Parallel Image Processing using thread library	Exploiting available threading mechanisms for parallelism in image processing applications. E.g. see work in [22], [23]
Automatic parallelization of sequential code	Detection of automatic parallelization in available sequential source code. E.g. see work in [24]

D. *Research work in Distributed Image Processing*

We studied various related research papers and present analysis of these papers in Table V. The reader can get idea about the research work under distributed image processing by referring those research papers.

TABLE V. MAIN RESEARCH WORK TOPICS IN DISTRIBUTED IMAGE PROCESSING

Topic	**Brief detail of work**
Distributed Image Processing architecture	Design and implementation of a software architecture that supports transparent task distribution and execution.
Distributed Image Processing Library/Routines	Software library containing algorithmic skeletons for distributed image processing. Software library containing functions used in particular application area, e.g. Bio-medical. E.g. see work in [25]
Middleware/Mediator/Plugin/Toolkit	Creating wrapper on existing general purpose distributed system so that the distributed environment becomes suitable to a single node application. E.g. see work in [26], [27], [28], [29]
Visual Programming Environment	Design and implementation of GUI based environment to prepare application in form of task graph. Embedding intelligence in visual programming environment is research worthy. E.g. see work in [30]
Intelligent subtasking	Automatic subtasking of a bigger coarse-grained task into fine-grained subtasks that are available in form of services/functions. This area is still in infancy, however preliminary level work on automatic sub-tasking is presented in [31]

VI. BRIEF DISCUSSION OF OUR RESEARCH WORK IN DISTRIBUTED IMAGE PROCESSING

We are working on design and implementation of a distributed image processing system that uses concept of utility based distributed computing. The system would support user friendly GUI to generate, submit, and monitor Task Using this system, the users would be able to design image processing application in form of task graph through GUI by performing drag and drop. The system would transparently use contract net protocol for task assignment. The research work in this system would also explore algorithms to utilize resources of the system optimally and to satisfy user supplied scheduling criteria. Our research work would explore following algorithms: (1) Algorithm for manager to learn about "which are good contractors for particular task? (2) Algorithm for manager to estimate overall cost and time of whole application sub-tasks. (3) Algorithm for contractor to control prices of sub-tasks in order to grab more contracts in future. (4) Scheduling algorithm to schedule tasks of different applications satisfying data transfer cost, execution cost and execution time criteria.

VII. Conclusions

We presented analytical study of parallel and distributed image processing. In this study, we presented levels of image processing operations and approaches of applying parallelism in image processing. We also provided brief directions on the tools that are used in parallel and distributed image processing. We also identified broad research areas in parallel and distributed image processing and provided brief details and directions on them so as to reach at depth in those areas.

This analytical study, which has considered different aspects, would be helpful to get concise understanding of the parallel and distributed image processing area. Further research study can be explored in specific hardware devices/processors used for parallel image processing. In distributed image processing, analytical study on image mining and content based image retrieval can provide details on algorithms, techniques, system architectures, etc. covering vast applicability of image processing operations.

References

[1] T. Bräunl, "Tutorial in data parallel image processing," Australian Journal of Intelligent Information Processing Systems (AJIIPS), vol. 6, pp. 164–174, 2001.

[2] R. C. Gonzalez and R. E. Woods, Digital Image Processing (3rd Edition). Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006.

[3] S. Haykin, Neural Networks: A Comprehensive Foundation. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1998.

[4] Coulouris, J. Dollimore, and T. Kindberg, Distributed Systems: Concepts and Design (4th Edition) (International Computer Science). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2005.

[5] C. Soviany, "Embedding data and task parallelism in image processing applications," Ph.D. dissertation, Technische Univ. Delft, 2003.

[6] W. Burkhardt, "Limitations to parallel processing," in Computers and Communications, 1990. Conference Proceedings., Ninth Annual International Phoenix Conference on, Mar 1990, pp. 86–93.

[7] M. Flynn, "Very high-speed computing systems," Proceedings of the IEEE, vol. 54, no. 12, pp. 1901 – 1909, dec. 1966.

[8] M. J. Flynn, "Some computer organizations and their effectiveness," Computers, IEEE Transactions on, vol. C-21, no. 9, pp. 948 –960, sept. 1972.

[9] R. Duncan, "A survey of parallel computer architectures," Computer, vol. 23, no. 2, pp. 5 –16, feb 1990.

[10] W. Gropp, E. Lusk, and A. Skjellum, Using MPI (2nd ed.): portable parallel programming with the message-passing interface. Cambridge, MA, USA: MIT Press, 1999.

[11] OpenMP Web site. [Online]. Available: http://www.openmp.org

[12] G. Slabaugh, R. Boyes, and X. Yang, "Multicore image processing with openmp [applications corner]," Signal Processing Magazine, IEEE, vol. 27, no. 2, pp. 134 –138, march 2010.

[13] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with cuda," in ACM SIGGRAPH 2008 classes, ser. SIGGRAPH '08. New York, NY, USA: ACM, 2008, pp. 16:1–16:14.

[14] Z. Yang, Y. Zhu, and Y. Pu, "Parallel image processing based on cuda," in Computer Science and Software Engineering, 2008 International Conference on, vol. 3, dec. 2008, pp. 198 –201.

[15] R. D. Blumofe, C. F. Joerg, B. C. Kuszmaul, C. E. Leiserson, K. H. Randall, and Y. Zhou, "Cilk: An efficient multithreaded runtime system," Journal of Parallel and Distributed Computing, vol. 37, no. 1, pp. 55–69, Aug. 1996.

[16] Threading Building Blocks. [Online]. Available: http://www.threadingbuildingblocks.org/

[17] C. G. Kim, "Accelerating multimedia applications using Intel threading building blocks on multi-core processors," in Information Science and Applications (ICISA), 2011 International Conference on, april 2011, pp. 1 –7.

[18] G. Sharma and J. Martin, "Matlab: A language for parallel computing," International Journal of Parallel Programming, vol. 37, pp. 3–36, 2009.

[19] Q. Wu, X. He, and T. Hintz, "Distributed image processing on spiral architecture," in Algorithms and Architectures for Parallel Processing, 2002. Proceedings. Fifth International Conference on, 2002, pp. 84 – 91.

[20] J. Brown and D. Crookes, "A high level language for parallel image processing," Image and Vision Computing, vol. 12, no. 2, pp. 67 – 79, 1994.

[21] H. Matsuo, K. Nakada, and A. Iwata, "A distributed image processing environment vios iii and it's performance evaluation," in Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on, vol. 2, 16-20 1998, pp. 1538 –1542 vol.2.

[22] M. Lückenhaus, "A multi-agent system for parallelizing image analysis tasks," in in Proceedings of the International Conference on Intelligent Autonomous Systems 5 (IAS-5. IOS Press, 1998, pp. 579–586.

[23] M. Lückenhaus and W. Eckstein, "A multi-agent based system for parallel image processing," in in Proceedings of the International Conference on Parallel and Distributed Methods for Image Processing at SPIE's Annual Meeting, Proc. SPIE 3166, 1997, pp. 21–30.

[24] J. Baumstark, L. and L. Wills, "Exposing data-level parallelism in sequential image processing algorithms," in Reverse Engineering, 2002. Proceedings. Ninth Working Conference on, 2002, pp. 245 – 254.

[25] F. Seinstra, D. Koelma, J. Geusebroek, F. Verster, and A. Smeulders, "Efficient applications in user transparent parallel image processing," in Parallel and Distributed Processing Symposium., Proceedings International, IPDPS 2002, Abstracts and CD-ROM, 2002, pp. 127 – 134.

[26] E. Manolakos, D. Galatopoullos, and A. Funk, "Distributed matlab based signal and image processing using javaports," in Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on, vol. 5, 17-21 2004, pp. V – 217–20 vol.5.

[27] P. Czarnul, A. Ciereszko, and M. Fraczak, Towards Efficient Parallel Image Processing on Cluster Grids using GIMP. Springer, 2004, vol. 3037.

[28] T. Hemalatha, G. Athisha, and S. Jeyanthi, "Dynamic web service based image processing system," in Advanced Computing and Communications, 2008. ADCOM 2008. 16th International Conference on, dec. 2008, pp. 323 –328.

[29] S. Hastings, T. Kurc, S. Langella, U. Catalyurek, T. Pan, and J. Saltz, "Image processing for the grid: a toolkit for building grid-enabled image processing applications," in Cluster Computing and the Grid, 2003. Proceedings. CCGrid 2003. 3rd IEEE/ACM International Symposium on, 12-15 2003, pp. 36 – 43.

[30] Y.-S. Sim, C.-S. Lim, Y.-S. Moon, and S.-H. Park, "Design and implementation of the visual programming environment for the distributed image processing," in Image Processing, 1996. Proceedings., International Conference on, vol. 1, 16-19 1996, pp. 149 –152 vol.2.

[31] F. Niederl and A. Goller, "Method execution on a distributed image processing back-end," in Parallel and Distributed Processing, 1998. PDP '98. Proceedings of the Sixth Euromicro Workshop on, 21-23 1998, pp. 243 –249.