American Journal of Engineering Research (AJER)
e-ISSN: 2320-0847 p-ISSN : 2320-0936
Volume-6, Issue-12, pp-209-219
www.ajer.org
Research Paper Open Access

Parallel Processing Implementation Using PVM for Image Processing Filtering

*AdelArfa[1],Riad Jernz[2]
[1](Department, of Computer Science/ Al-Jabal Al-GharbiUniversity, Libya)
Corresponding Author : *AdelArfa

ABSTRACT: *Parallelization is known as an optimization technique that is widely used in many fields with its main purpose of gaining a performance and reducing responding time. In other words, the phenomenon of parallelization is when something is parallel only when there is a definite point of independence for the order of operations and when operations can be performed in any order. On the other hand, one of the fields called the image processing is extremely demanding since it involves several operations. For instance, spatial filtering, image enhancement, image compression and image restoration are all types of image processing. Therefore, this project intends to implement a parallel processing using PVM (Parallel Virtual Machine) for image processing filtering using the Low Pass filter and evaluating its performance, speed-up and efficiency and time responding is based upon implementing filter on one processor through multiprocessors computers. This implementation involves partition of an image into sub-images to be processed in a remote location. After the image is sub-imaged, and is processed remotely in different processors (slaves in this case). The results will be sent back to the server (master), to reassemble all the sub-images to produce one image. A mixture of image sizes along with a set of distributed computing processors and a variety of filter masks will be used to analyse the performance of a parallel implementation.*

Keywords: *Image Processing, Parallel Processing, Parallel Virtual Machine,*

Date of Submission: 14-11-2017 Date of acceptance: 28-11-2017

I. INTRODUCTION

The parallel processing using Parallel Virtual Machine for image processing filtering using the Low Pass filter that is not the most commonly used network topology. This is because it can be extremely difficult to integrate computers and perform the parallel programming using parallel virtual machines for image processing. Therefore, a better solution must be acquired to resolve this for better and faster results. This project will look into the solutions which are available, and then it shows the performance of parallel programming using Parallel Virtue Machine for image processing using the Low Pass filter. For the parallel image processing, there should be a user friendly graphical user interface of any familiar application such as Adobe Photoshop or any other application. However, there is no such application which exists for the user in which a user can create a network and perform the image processing tasks to achieve significant result within no time.

1.1 Study Background

Image processing system requires consideration of its result by testing and checking through experiments before any solution can be imposed. The testing experimenting and prototyping, normally saves time and the cost to become a feasible solution. Small amount of information has been given in instructional material, to satisfy the difference between the application and the abstract part, to support the development environment. The main purpose of vision is to allow us to carry out three tasks: first: perception of our world; second: Better understanding of strategy to perform specific actions. Finally, perform that action. Filtering, segmentation and rotation are some of the image processing that involves local computation based upon the image array. This can be done in parallel [1].

Distributed systems and parallel processing are extremely powerful and fundamental process weighty task. Employing hardware for image processing is more expensive whereas a variety of distributed programming technologies such as JAVA RMI, Remote Procedure Call (RPC), MPI (Massage Passing Interface) and PVM (Parallel Virtual Machine). Hence, this project will use PVM for the design and

implementation of image filtering in parallel. The image processing filtering is based on the convolution that is the most common in this area. The convolution technique comprises of a Fourier transform relationship that represents a basic link between the space and frequency domains and it stands. Because of the corresponding of the image processing filtering to a convolution in the space domain between the image pixels and the convolution mask (image filter in the space domain), a Fourier transform relationship stands between them [2].In this project implementation of parallel processing using Parallel Virtual Machine is applied, and then implementing image processing techniques using Low Pass filter to remove the noise of the image. The project is organized in the form of chapters and presented in a document form. The main idea behind theory and software are explained to develop the required application and explain how the application was developed. However, most of the people are aware of image processing, Low Pass filter and Parallel virtual machine to some extent. This project will explain these concepts theoretically as well as practically by showing how it is managed to use them in the project.

Obtaining the efficient result and reducing response time can be shown in this project. The targeted audience for this project is the industry that is using image processing to get proper images, and, because of one computer system the response time is quite long, to deal with this problem, the system will implemented in parallel using workstation so that the problem can be divided into the network and get the result within a short time limit.

1.1.1 Parallel Computer Architecture

The architecture is the most valuable thing in the implementation of the study. There are two kinds of architectures, which are commonly used in the present days.

1-Von Neumann architecture
2-Harvard architecture

The original Von Neumann architecture is still commonly used in most computers. Consisting of a single processing unit, and a single memory unit for both the instruction and data, only a single memory location can be accessed at any one time. For instance, it contains only one bus, and both data and instructions, are scheduled and cannot be executed at the same time. Also, it uses only single integrated cache that contains not only instructions but data as well. Instructions and data have the variable portion in cache, which would help to have different data and instruction cache memory. Therefore, for execution of a sequence of operations, much of the processor is idle, [3]. There is another architecture named Harvard architecture which is used quite often. It splits the data and instruction buses and then lets the transfers be performed at the same time on all the buses. In addition, Harvard architecture can have more than one separate memory systems. However, if the data and instructions are treated at the same time then, the cache or memory is not checked. This means both can be used. But the problem arises when compiler set in the data inside the code. In the process of modifying the code, it is impossible because it has two different isolated memory systems and it requires a link among the memory systems to work on this. Integrated memory system with Harvard architecture is not recommended, it is only recommended if and only if there is a possibility to enter the data into both buses. Also, Harvard system always uses caches to achieve high performance. In result, it can be said that two different memory systems have the better ability to perform well but it is extremely difficult to implement them [3].

1.2 Study Objects

The main objectives of this study are:

- A functioning system implementation which is able to perform Low Pass Filter operation remotely.
- To evaluate its performance, speed-up, efficiency and time responding.
- Divide the image into sub-images to process them in a remote location.
- There will be a network working i.e., One master and one or more workers. Number of distributed processors and filters will be used to analyze the performance and efficiency.

II. RELATED WORKS

This part presents the previous work related to this study of parallel processing using Parallel Virtual Machine. It will show, who implemented the project and how and what is the study requirements.

2.1 Image Processing in Parallel Virtual Machines

The parallel versions of smoothing; edge detection and Hough transform algorithms in parallel which was WPVM (Windows Parallel Virtual Machine),[4]. They applied a smooth algorithm to a several image sizes (100X100, 400X400, 700X700, and 1000X1000) and tested them with filter masks 3X3, 5X5, 7X7, 9X9 and 15X15 dimensions. In parallel processing implementation, the problem should be divided into small ones which

is can be execute in different processors. In this case, they decided to divide the image rather than dividing the algorithm because of the functional dependencies in it sequence. Therefore, they considered three different methods to split the image into four blocks (sub-images): in horizontal slices; in vertical slices; both of them.
As result, dividing the image by using the method of the combination of both horizontal and vertical slices is chosen as a conclusion of the overlap region.

A cluster of eight PC's connected by a network is employed to test each algorithm with 1, 2, 4 and 8 PC's as well as different image sizes. The result was depending on the image size and the number of employed PC's. Filtering Image using WPVM with different masks dimensions, image sizes and number of employed computers [5]. It uses a master-slave process farm to implement image filter which means the image is divided into sub-images. More importantly, the master task is to control and create slave processes as well as sending each sub-image to salves to be filtered whereas the processed sub-image from each has to be sent back to the master to combine them to perform one output image. Thus the data about sub-image are also to be sent to the slave task including the extra pixels (overlap pixels) by the master. Their analysis of the result of testing the implementation was depending on three factors: the size of the message between the master and slaves; the size of different masks; and the number of processes.

2.2 Efficient Parallel Image Processing on Cluster Grids using GIMP

[5]used parallel image processing on cluster grid using GIMP. Now days, the experts of graphic mostly utilize low level parallel programs to boost up the image processing. They developed an efficient and highly praised GIMP, which puts so many filter operations in a queue in parallel to those images which are loaded by plug-in. The DAMPVM is an extension, which is developed to facilitate the end user with an elegant cluster shell. The Parallel Virtual Machine can be run on any operating system; it is not limited to specific operating system. In addition, DAMPVM can be modified to satisfy the needs of graphic software so it can use cluster quite well. The architecture they used for plug-in is given below: which contains three parts

- A Script-Fu Wrapper
- A decision-making GIMP deployed plug-in
- Worker nodes have parallel pipeline architecture.

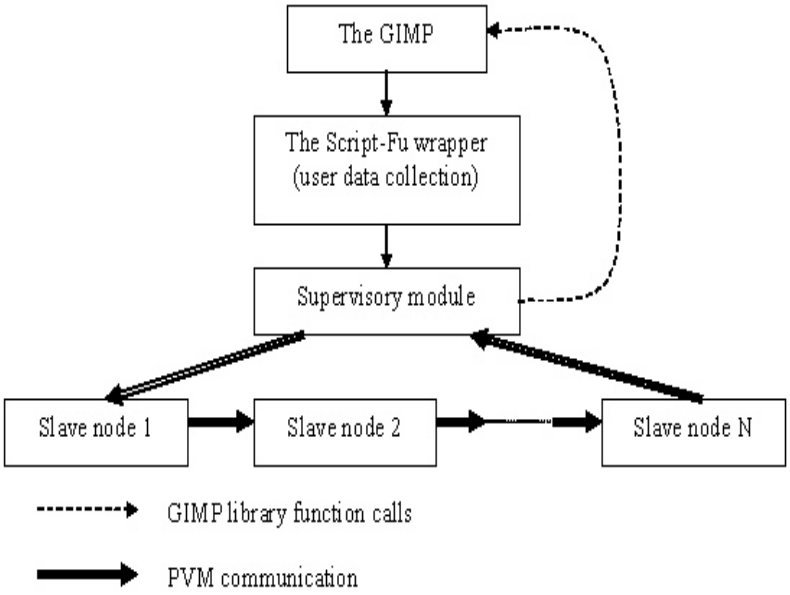


Figure 1: GIMP Plugin's Architecture [6].

According to that architecture, they applied the set of maximum ten filters to the images. These images were located on one computer with GIMP and connected to the cluster computers through the network and Parallel Virtual Machine is working properly.

They presented test scenarios and investigating results of cluster grids which contained most likely single processors and SMP nodes.For experimental results, they used 16 Intel Celeron 2GHz, 512 MB machines, which were connected with each other using 100Mbps Ethernet to show that the image processing can be successfully assisted by DAMPVM shell on cluster grids. They held an experiment to check the capability of

DAMPVM runtime, so that they can check the nodes which were least loaded or newly born tasks (Spawn tasks) and the tasks which were in pending state, so that these tasks can be submitted to processor when the process is done with its work.In other experiment, using remote shell they checked the static pipeline allocations and dynamic pipeline allocations. They distributed the work on the same network and the efficiency of plug-ins had to be alike. But they found out, that on simple network it is quite difficult to achieve these goals. The nodes are not only overloaded in academic examples but in inactive network as well. The plug-ins which is remote shell enabled, they should avoid loading those nodes which are already loaded with work. There are quite a few variables in pipeline simulation:

- The processes which are in pipeline
- The number of images to be processed
- Image sizes may vary
- Type of filter is decided while considering the processing time

In this given figure 5, it shows a filter result of 3 x 3 matrix, where processors p=10 for N= 25,800 x 600 and 1600 x 1200 bitmaps.In figure 6, there is filter result of 5 x 5 matrix, where processors are P= 10 for N=25 and N = 50,800 x 600 bitmaps. Almost 14 processors existed in inactive state. One of the processor had GIMP running over it i.e. host and statically 10 processors were selected. In second experiment, host was receiving GIMP images and sending them to the second stage itself, but in case of large images this becomes bottleneck because host also saves the images into the disk. Ultimately, remote shell was begun to open the 10 worker node processors.

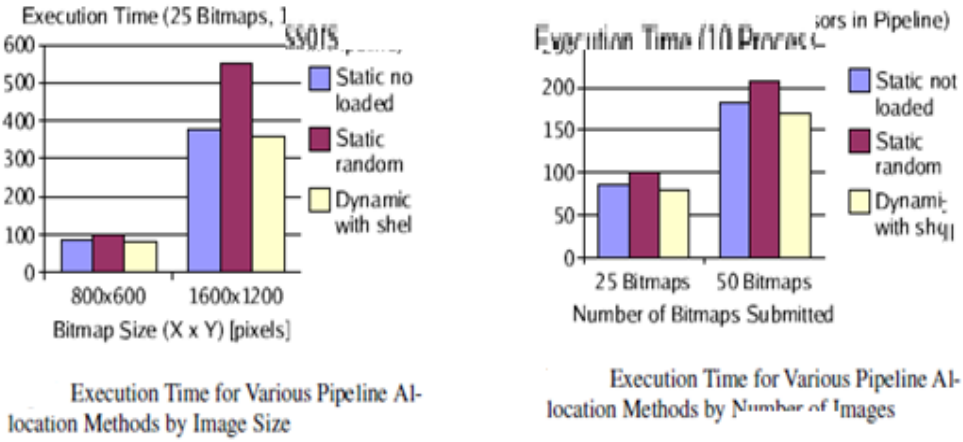


Figure 2: Two charts show different execution time [6].

From the result of their tests, they proved that, the parallel mapping is totally clear to the end user, and got the results of GIMP plug-in with using cluster grid shells. [6].

2.3 Parallel Image Processing and Computer Vision Architecture

In 2005, James Greco proposed this thesis and presented a narrative technique in hardware to use different image processing and computer vision algorithms. Image processing algorithms can be developed more efficiently and easily by using pipeline methods. He talked about some advantages and disadvantages of using specialized hardware and gave the architectural overview of it. The systems which are reusable are generally slower as compare to a system which is just working on one and only task. He proposed an architecture which was way more specific to image processing. Image processing tool replaced the ALU operations. Image processing tool has this ability to keep the speed above the lowest rate and generates similar inputs and outputs, which helps in module reusability and pipeline of successively coming modules.
Only two technologies can be used for the implementation of the architecture

1- The Application Specific Integrated Circuit (ASIC)
2- Field Programmable Gate Array (FPGA)

He used FPGA technology to design the architecture, because it does not propose a high cost solution for minute productions. The structure of Field Programmable Gate Array is given below:

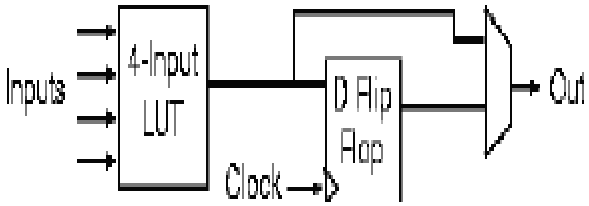


Figure 3 : The most basic implementation of a logic cell contains a LUT for combinatorial output and flip flop for registered output [7].

Then he performed some calculative analysis on the architecture. He performed these tests on 1.6 GHz Pentium 4 with 768 MB of RAM and got these results.

Table 1: P4 and FPGA timing comparison for three operators [6]

Operation	P4 (ms)	FPGA (ms)	Ratio
Vert. Sobel Filter	50.0	4.10	12.2
Highpass Threshold	10.0	4.10	2.44
3 x 3 Erode	30.0	4.10	7.32

Table : P4 and FPGA timing comparison for three operators

Then he performed some architecture experiments including object tracking using colour scheme.
He parameterized to distinguish the solid object colour. In the given example he distinguished blue colour from the other parts.

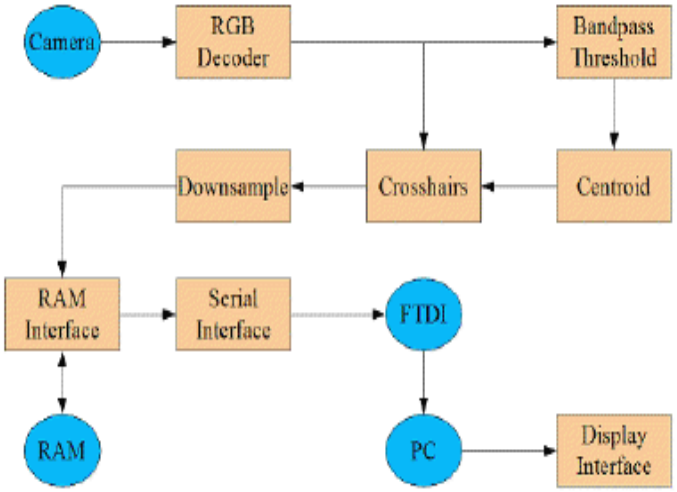


Figure 4: A simple implementation of our architecture is used to find the centre of a uniformly coloured object based on its colour properties.

The above Fig. 4 shows the implementation of the architecture which is used in the experiments, to discover the centre of coloured objects.Band pass threshold is also used to distinguish the colour objects. There is only two possibilities i.e. either balls are blue or either they are not blue. Then he found out the centroid of the image depending upon the result of band pass i.e. he considered all the pixels blue. The next point which he implemented was down sampling. He down sampled the pictures into 4 and stored it into the RAM.In his experiments, he proposed that architecture is based on hardware, which provides better presentation for some problems related to computer vision and image processing, while designing he used parallel approach to capitalize the throughput,[7].

III. METHODOLOGY

The method used is divided into three phases: the first phase is to implement the task on one processor (sequential implementation). Second one is a parallel implementation using PVM. Final phase is the analysis of performance.In the first phase, a sequential image filter is implemented and clarified by means of qualitative and quantitative analysis and comparing the result with parallel implementation and make sure the obtaining results are accurate and there is no error or fault in them.The parallel implementation that is the second phase is executed on four PC's also validated in the course of comparing its result with a single-processor. For this phase, the Parallel Virtual Machine has been used to make the processors interconnected with each other, in order to check the parallel processing i.e., There will be one host and four workers which will be linked, and host will manage all the parallel implementation activities.In the last phase, using different filter masks, image sizes and the number of computers are used to analysis the performance by using some measurements such as speed-up, efficiency and response time. The main purpose behind this study is to minimize the time response and improve the efficiency. The filters which used in this study are Low Pass filter and a cluster of computers are employed to perform parallel programming using image processing and images can have different sizes. Next, this will be distributed over the network using Parallel Virtual Machine, and then implemented Low Pass filter to remove the noise. Consequently, it will be able to see the result in lesser time and it will be more efficient and perfect.

3.1 Analysis of problem

Distributed Image Processing is a concept of using multiple computers to act in sync for processing images and there will be one master computer i.e. Host and all other computers will be Workers. By using multiple computers the processing task will take less time to complete.

3.1.1 Parallel processing using PVM

PVM (Parallel Virtual Machine) is an integrated set of tools and libraries that form a framework for building a single parallel computer from a collection of interconnected heterogeneous computers [8].
Parallel Virtual Machine (PVM) is also defined as a structure of code which allows the user or developer to use more than one processor into a single virtual machine in order to run parallel programming.

The principles on which Parallel virtual Machine works are given as:

1- User configured host group: All the application tasks execute on the user selected machines. The entire single machine or multiple processor machines are part of the host group,[9].
2- Transparent contact to the hardware: Either hardware setting maybe viewed as a characteristic less set of virtual processing fundamentals or either selects to utilize all the capabilities of particular machine in host group,[9].
3- Process based calculation: Parallelism in PVM has a unit which is known as task. It is not dependent and it is a control that changes among communication and calculation,[9].
4- Clear communication of passing model: The calculation tasks perform specific acts on application i.e. hybrid, functional breakdown of workload by transferring and accepting messages of one another,[9].
5- Heterogeneity Support: Parallel Virtual Machine allows messages to have more than one data types that can be exchanged among those machines which have diverse data types,[9].
6- Support of Multiprocessors: Parallel Virtual Machine always uses the local massage transfer services on Multiprocessors to get a benefit of the basic hardware,[9].
7- Parallel Virtual Machine on network consists of two machines i.e., Master and slave. There is always one Master "host" and others are Slaves "workers". Parallel Virtual Machine contains different set of programs i.e. A complete set of libraries and control or message passing program which is also known as PVMD. PVMD is an interface of massage transitory between workers and host application program. Basically, PVM systems include two parts,[7].

First part is a daemon resides in every node that is part of the virtual machine. A daemon is a program that constantly runs in background. In the case of PVM daemons monitor system's resources and exchange messages with other daemons. Second part is a library interface routines that contains a complete set of primitives to handle cooperation between tasks. PVM relies on the notion that an application is divided into tasks. Each task is responsible for a part of the computational workload of an application. Functional and data parallelism of tasks are supported, as well as a mixture between the two. The libraries which Parallel Virtual Machine contains are libfpvm.a and libpvm.a. libpvm passes the massage passing functionality to the worker so that it can easily correspond with the other hosts available on network. Host PVMD is triggered by these

libraries and deals with the information of transmitting the massage. Host PVMD catches the message and application part can view this message through the library call within the same program [9].

The primary goal of PVM is to build a flexible, cost-efficient solution to large computational problems relying on the aggregate power of many computers. Parallel Virtual Machine's design allows it to interconnect machines of different architecture - from laptops to Crays. Transparently to the user it handles message passing, data conversion, and task scheduling across such a network. Primary goal to PVM is to form a Single System Image or a "virtual machine" from a set of heterogeneous nodes connected by a network. Computers form a pool of resources and the user can specify which computers are to be used for executing a current set of tasks. PVM handles changes to the pool allowing machines to be added or deleted from the virtual machine at runtime.

A developer can handle the problems by distributing them on multiprocessors by using this set of programs and after the problem is solved, the output can be combined again into a required outcome. Parallel Virtual Machine can also be defined as a distributed memory system in which, problems are divided into quite a few parts and distributed all over the structure of processors. The parts of the problems are known as tasks. There is process known as control process which splits the problem into particular amount of parts and divide them to the processors and the recombining of the data parts can be done by using control program, [11].Computations are carried out by system processes. PVM API allows processes to be started or stopped according to different criteria, imposed also by external schedulers or resource managers. It supports also exchange of messages that check if a process is alive or informs when a process leaves the system. However, PVM allows only blocking send. Non-blocking send is available with MPI.

The Parallel Virtual Machine computing model is set up on the idea that an application may have quite a few tasks. Every task is responsible for the calculation of application workload. Most of the times, every task starts performing dissimilar functions and such process is known as functional parallelism, there is another type of parallelism and that is known as data parallelism. In data parallelism, all the tasks are similar and perform all the small parts related to it. The diagram which is ideal for Parallel Virtual Machine computing is given below that shows the heterogeneity supported by Parallel Virtual Machine.

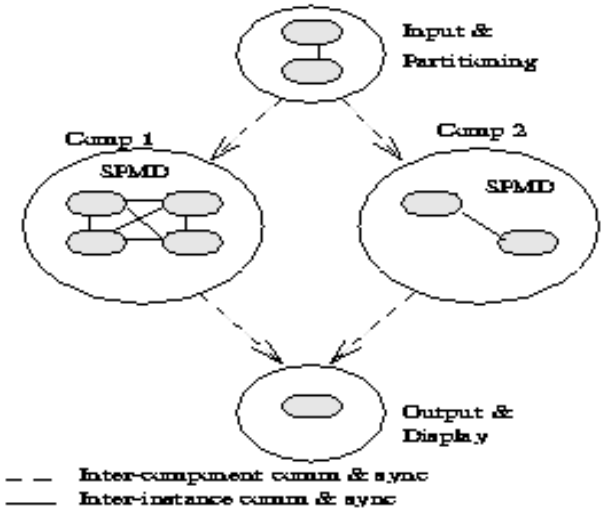
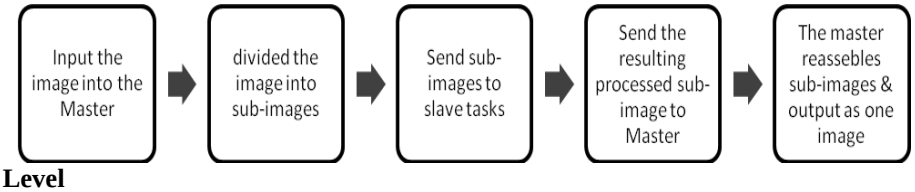


Figure 5: PVM Computation Model [7].

IV. DESIGN

This part illustrates and explains the system design and it architecture in primary level and detailed level.



4.1 Primary Level

Figure 6: Primary level of the system

In the primary design level, the steps are as following:
- Input the image into the Master.
- Divided the images into sub-images.
- Send sub-images to slave tasks.
- Send the result back to the master after sub-image processed.
- The master reassembles sub-image to perform one image and output the image.

4.2 Detailed Design

4.2.1 Master and its associated functions

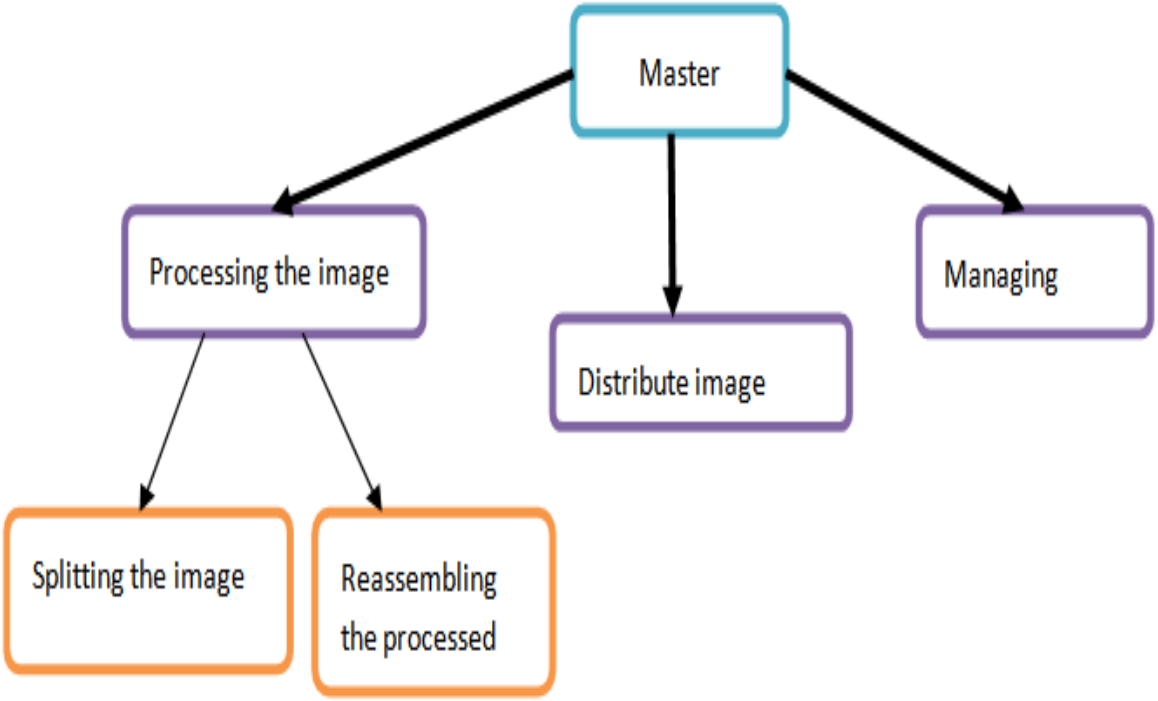


Figure 7: Master and its associated functions

- Processing the image

1. Dividing the input image into sub-images.
2. The sub-images have to be serialized before sending to the slaves.
3. The processed sub-images have to be reassembled to perform an output image.

- Distribute the image slices

1- Master (server) assigns each sub-image to the slaves.
2- The image can be divided depending on the number of slaves. Therefore, each slave can process one sub-image.

- Managing the slaves

1- Add the slaves in the virtual machine.
2- The entire slave should be assigned with the task.
3- Wait the slaves to finish their assigned tasks.

4.2.2 Slave and its associated functions

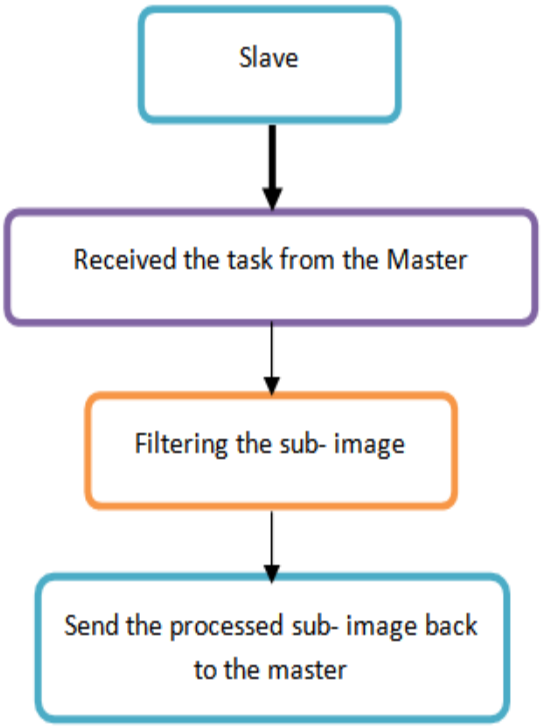


Figure 8: Slave and its associated functions

- Slave has three approaches which are as following:

1- The slave receives the sub-image which is sent from the master.
2- The slave applies the filter to the sub-image.
3- Finally, the slave sends back the processed sub-image to the master.

4.2.3 The master and the slaves interaction in remote system

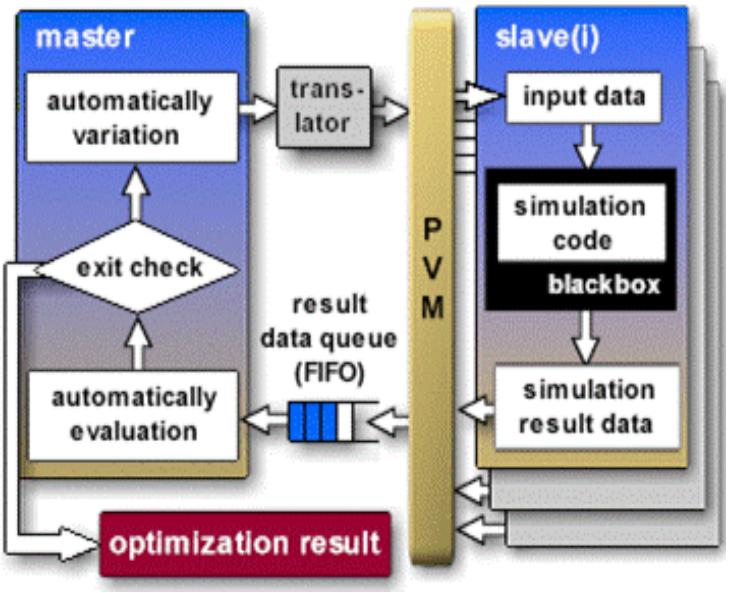


Figure 9 The master & the slaves interaction in remote system [9].

The master and the slaves are expected to intact as required in system design. Calling pvm_mytid( ) will enrolled the process in virtual machine. The master and slave interact by sending and receiving the message from each other.

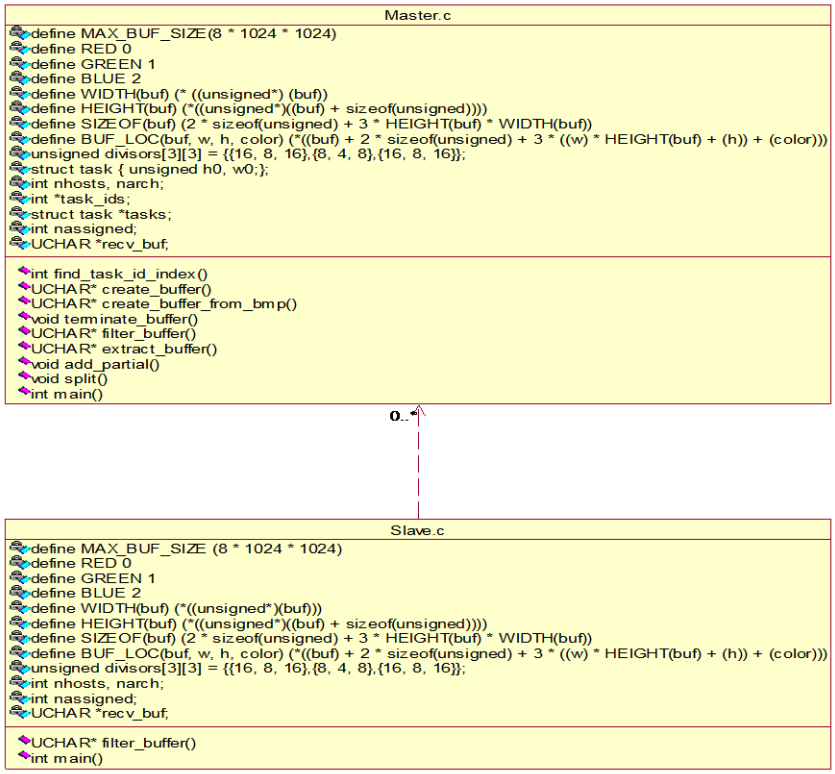


Figure 10 : Class diagram of master-slave program of in this system

V. IMPLEMENTATION

As it described in the design section, the implementation of image filter (Low Pass filter in this case) consists of master and slave process. The master generates slave processes and assigns each sub-image to the slave processes to be filtered. In addition, it is also inform the slave about which sub-image should be loaded for applying the filter by sending a message contains data about the sub-image. The master operates a splitting method on the input image which depends on the number of slaves. For instance, if the number of slaves = 8, then the image will be split into 8 slices. As the split method shows: split(intntasks, UCHAR *buf, unsigned h0, unsigned w0, unsigned height, unsigned width, UCHAR *buf_sol), where "ntasks" is the number of slaves. Then the master sends each sub-image to the slave processes by executing the following PVM method: nspawned = pvm_spawn(argv[3], (char**)0, 0, "", nhosts - 1, task_ids). Therefore, the master now has to process the Remaining sub-image. After all the sub-images processed, the master will perform the final stage which is reassemble them to one output image. The method for this task is: add_partial(UCHAR *buf, UCHAR *buf_sol, unsigned h0, unsigned w0). When the slave receives its assigned sub-image, it puts the sub-image into a buffer and also creates a new buffer from the input sub-image to operate the filter. The filter is applied to the sub-image by the method called filter_buffer(UCHAR *buf), then when it is finished. The slave sends back the result by executing this method: pvm_psend(parent_id, 1, send_buf, SIZEOF(send_buf), PVM_BYTE).

VI. CONCLUSION

PVM allows the implementation of parallel algorithms on inexpensive hardware. Using a collection of PC's connected together by network for parallel image processing, could be reasonably good solution rather than using a high-cost parallel architecture. Some of the considerations might be related to ones it will be faced when you attempting to obtain parallel processing implementation for image processing algorithms. There are some limitations to make such this applications run in parallel. The most important limitation is a communication via the network. This communication refers to data transformation between tasks. Therefore, it is essential to decide

a communication structure between tasks in a manner to be practically efficient. Generally, in image processing filtering, processing a large image with a large filtering mask tend to be more efficient than a small image with a small filtering mask.

REFERENCES

[1]. Z. Hussain, "Digital Image Processing: Practical Applications of Parallel Processing Techniques", E. Horwood,Publisher , 1991.
[2]. L. F. W. Góes, L. E. S. Ramos, C. A. P. S. Martins, "ParallelImage Filtering Using WPVM in a Windows Multicomputer",
[3]. International Conference on Computer Science, Software Engineering, Information Technology, e-Business, and Applications (CSITeA'02), Foz do Iguaçu, Brazil, 2002.
[4]. ARM (2010) ARM Information Centre. [Online]. http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.faqs/ka3839.html[18/08/ 2017]
[5]. R. Gonzales, and P. Winitz, "Digital Image Processing", 2nd edition, Addison-Wesley Publishing Company Inc, 1987.
[6]. Vaz C. El al (n.d) 'Image Processing in Parallel Virtual Machines Some Experiments and Conclusions'.
[7]. Dwayne, P. (2000). Image Processing in C. [Online]. April 26. 7 p. 76-87. http://homepages.inf.ed.ac.uk/rbf/BOOKS/PHILLIPS/cips2ed.pdf. [18/08/ 2017]
[8]. Pawel, C., Andrzej, C. & Marcin, F. (2006). Towards Efficient Parallel Image Processing on Cluster Grids using GIMP [Online]. http://wenku.baidu.com/view/7a787dd43186bceb19e8bbc5.html?from=related [19/08/ 2017]
[9]. Geist A., Beguelin A., Dongarra J., Jiang W. , Manchek R., Sunderam V., PVM:Parallel Virtual Machine: A Users' Guide and Tutorial for Networked Parallel Computing. MIT Press, 1994. [Online]. http://www.netlib.org/pvm3/book/pvm-book.html [05/09/2017]
[10]. G. James, Parallel image processing and computer vision architecture. BSc thesis in Computer Engineering, BSc diss., University of Florida. USA, BSc, 2005. (8)
[11]. Netlib (2000). *the PVM System* [Online]. Available from: http://www.netlib.org/pvm3/book/node17.html [17/08/2017]
[12]. Richard A, S. (1998). Parallel Processing using PVM. Linux Journal [Online]. http://www.linuxjournal.com/article/2258 [07/09/2017]
[13]. Nate, Stone. and Mike, Frank. (1998). parallel processing MCNP using PVM: experience and results on Pentium pro multiprocessor workstation and sun sparc ultra 1 network. Ne 255 term project [Online].
[14]. Acut, Software - CAOT Computer Aided Optimization Tool. [Online] http://www.acutgmbh.de/caot_english.html [01/09/2017]

*AdelArfa. "Parallel Processing Implementation Using PVM for Image Processing Filtering." American Journal of Engineering Research (AJER), vol. 06, no. 12, 2017, pp. 209-219.