

Video Enhancement using Grid Computing

by

Mohd Hazry Bin Sulaiman

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Technology (Hons)
(Information & Communication Technology)

JULY 2007

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Video Enhancement Using Grid Computing

By

Mohd Hazry Bin Sulaiman

A project dissertation submitted to the
Information & Communication Technology Programme
Universiti Teknologi PETRONAS
in partial fulfillment of the requirement for the
BACHELOR OF TECHNOLOGY (Hons)
(INFORMATION & COMMUNICATION TECHNOLOGY)

Approved by,

(Dr. M Nordin B. Zakaria)

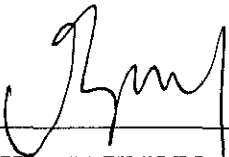
UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

July 2007

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.



MOHD HAZRY BIN SULAIMAN

Abstract

This paper describes the implementation of video enhancement program on grid computers. The implementation is focus on the enhancement of video brightness, contrast and hue of the video. Using the project, we can adjust the quality of the video manually as we like. We can have brighter video but not too bright as the color should soothes our sight. The video quality is important as people are happier and more satisfied with high video quality. To make this project a success, a research have to be conducted to collect the information needed for the project. Before begin the programming part, I have to make detailed planning. There will be a lot of research need to be done. Firstly, I need to do research on extracting AVI file to frames. Then I have to do research on grid computing. There are several grids computing environment which used different kind of programming languages. Then I have to integrate assemble the extraction coding and the grid computing coding together. Throughout the period of research and development, Spiral Development Methodology was use as the main methodology as it provides the flexibility needed for the project.

ACKNOWLEDGEMENT

Upon completing this final year project, first of all, I would like to express my gratitude to ALLAH the Almighty. I am indebted to the individuals who have contributed their ideas, view, encouragement and support within the length of this project. No substantial gratitude could ever measure up to assistance, guidance and drive to keep me striving towards accomplishing the goals set.

I would like to thank to my beloved parents, Sulaiman Md. Dom and Hamidah Ahmad, who have faith and give support that I can go distance. I would like to take this opportunity to thank to Dr M Nordin Zakaria, as the project supervisor for his encouragement, support, patient, help and guidance through out this project. Last but not least, thank you to everyone else that involve directly or indirectly in providing a big contribution to this project.

TABLE OF CONTENT

ABSTRACT		i
ACKNOWLEDGEMENT		ii
TABLE OF CONTENTS		iii
LIST OF FIGURE		iv
CHAPTER 1:	INTRODUCTION	
	1.1 Background of Study	5
	1.2 Problem Statement	6
	1.3 Objective & Scope of Study	8
CHAPTER 2:	LITERATURE REVIEW	
	2.1 Grid Computing	11
	2.2 MPICH	14
	2.3 Image Enhancement	17
	2.4 Developer's Image Library	19
	2.5 Audio Video Interleave	23
CHAPTER 3:	METHODOLOGY	
	3.1 Procedure Identification	24
	3.2 Tools Required	25
	3.3 System Flow	26
CHAPTER 4:	RESULT AND DISCUSSION	
	4.1 Extracting Video to frames	27
	4.2 Running Developer's Image Library	35
	4.3 MPICH2 Configurations	37
	4.4 Image Joiner	40
CHAPTER 5:	CONCLUSION	
	5.1 Conclusion	41
REFERENCES		42

LIST OF FIGURES

Figure 1.1 Final Year Project Part 1 Schedule

Figure 1.2 Final Year Project Part 2 Schedule

Figure 2.1 The First implementation of the MPICH-V1

Figure 2.2 The Newest Implementation of the MPICH-VCL

Figure 2.3 AVI Diagram

Figure 3.1 Spiral Development Model

Figure 4.1 Open Terminal

Figure 4.2 Open Directory

Figure 4.3 Explore Directory

Figure 4.4 File Browser

Figure 4.5 Command Line Descriptions

Figure 4.6 Execute extract file command

Figure 4.7 Extracting In Processes

Figure 4.8 Extracting process finished

Figure 4.9 Extracted File from the video

Figure 4.10 Explore extracted file using file browser

Figure 4.11 Frame of extracted image

Figure 4.3.1.1 Configuring the MPICH2

Figure 4.3.1.2 Gui of the Mpiexec.exe

Chapter 1

Introduction

1.1 Background Study

What is video? As describe in the wikipedia, the term video is from the Latin word and it means "I see". Video are commonly refers to several storage formats for moving pictures. Most videos will also have audio embedded in them. There are many digital video formats, including AVI, DVD, QuickTime, and MPEG-4. Quality of video essentially depends on the capturing method and storage used. Digital television (DTV) is a relatively recent format with higher quality than earlier television formats and has become a standard for television video. Video can bring many purposes and it like a very entertaining and effective way to deliver message. Nowadays there are many advertising using video. [1]

What is grid computing? Grid computing is also known as parallel computing. It enables the virtualization of distributed computing and data resources such as processing, network bandwidth and storage capacity to create a single system image, granting users and applications seamless access to vast IT capabilities. Just as an Internet user views a unified instance of content via the Web, a grid user essentially sees a single, large virtual computer. Grid computing brings a lot of benefit. Not only to the technology, has it also brought benefit to business and people. This technology enables communication across heterogeneous, geographically dispersed environments. With grid computing, organizations can optimize computing and data resources, pool them for large capacity workloads, share them across networks and enable collaboration. [2]

I have briefly described about the two elements that I will use for my final year project. These elements I need to master and understand because I need to make these two elements work in same platform. I have decided to work with video. I will experiment with the AVI format in my research and as for the grid computing, I will be using MPICH as has been installed in University Technology of Petronas.

1.2 Problem Statement

1.2.1 Problem Identification

Basically there are few problems that motivated this project development. Here are few issues that lead the author to propose this title:

- **The need to increase video quality**

Video quality is really important. A better quality video is more likely to have more viewers than the low quality video. A better quality also will deliver message clearer. This problem can be solved as this project main purpose is to enhance the quality of the video.

- **The need to shorten processing time**

Processing any video format which is in big size will take a lot of time. The conversion also will use a lot of the computer memory. The grid computing will help to solved the problem as it provide more high processor that can process many task and high usage task more faster. The grid computing is faster than any high end personal computer.

- **The need to reduce storage space**

Video processing use a lot of hard disk spaces. It is more like we have two same video file. This is because we cannot replace the existing video file as it is still been use to process the enhancement of the video. But we can delete the lower quality video and replace it with the one that we have finished convert. Grid computing provide a lot more spaces on the hard disk as it gather all the free hard disk spaces on the computers together so that user can have really big storage.

1.2.2 Significant of project

The project when implemented will provide many benefits. User can save a lot of time by doing video enhancement using the grid computing. Because of the short time taken to process it, user can do a lot of work in one day. This can improve productivity.

Today, the grid computing or the parallel computing has been implemented in our department. There is also a project where some labs at certain universities are link together to create a larger grid computing. Some of the universities are University Technology of Petronas, Multimedia University and University Tenaga Nasional. The main servers are located at the MIMOS. This project will expose myself to grid computing and take student or even lecturer to understand more about Information Technology.

By doing this project, there are so many things that will be learned. There is something that needs practice to be understood. So, the significant of the project is that I as a student can learn and experience the actual environment of grid computing. This kind of experience cannot be achieved in normal classes. Therefore this project brings a lot of benefit.

1.3 Objective and Scope of Study

1.3.1 The relevancy of the Project

The development of the video enhancement using grid computing has several objectives. The objectives of this project are:

- To develop video enhancement software that can work in grid computing environment.
- To prove that video enhancement can be done faster in grid computing environment.

1.3.2 Scope of Study

This project is mainly about the video enhancement that enable user:

- To enhance a video accordingly to their needs.
- To perform the enhancement in grid computing environment.

The scope of study which I need to do the research is about the grid computing. I have to learn the architecture of the grid computing. There are many implementations of message passing interface (MPI) today. But I will focus on using MPICH as the MPI. I also need to focus my study on how to enhance a video. As for the video I choose the AVI format video to be enhanced.

1.3.3 Feasibility of the Project Within the Scope and Time Frame

The development of this video enhancement using grid computing involves research work, interface design, coding, configuration and testing. Figure below show the scope of works which are represented in percentages.

- 30% in research work
- 5% in interface design
- 50% in coding
- 5% in configuration
- 10% in testing

The project consume a large amount of time especially in research work and coding because the research work need to be conduct in detail and the development of the application is not simple as it involved nontrivial coding. However, the implementation of this project is expected to be completed in the given period of time.

NO	DETAILS	WEEK NO													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Selection of project topic	■	■												
	-Propose topic		●												
2	Preliminary Research														
	-Project Planning			■	■										
	-Literature Review			■	■	■	■	■							
3	Submission of Preliminary Report								●						
4	Project research/work														
	-Literature Review								■	■	■	■	■		
	-Interface Design											■	■		
5	Submission of Interim Report												●		
6	Oral Presentation													●	

Figure 1.1 Final Year Project Part 1 Schedule

NO	DETAILS	WEEK NO													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Programming Research	■	■	■	■	■	■								
2	Development														
	- Prototype Development				■	■	■	■							
	- UTPEMS Development								■	■	■	■	■	■	
3	Submission of Progress Report			●											
4	Submission of Final Report Draft							●							
5	Submission of Final Report										●				
6	Oral Presentation													●	
7	Submission of Project Dissertation														●

Figure 1.2 Final Year Project Part 2 Schedule

Chapter 2

Literature Review and Theory

2.1 Introduction to Grid Computing

Today, there has been a lot of Message Passing Interface (MPI). MPI is language-independent computer communications descriptive application programming interface (API) for message-passing on a parallel computer. [3] As for the project, I was advised to use the MPICH software. MPICH is free software. It can be download and freely available in the website. It is portable implementation of MPI, a standard for message-passing for distributed-memory applications used in parallel computing. MPICH is available for most of the operating system of UNIX (Linux and Mac OS X) and Microsoft Windows. Moreover, MPICH is a developed program library.

The original implementation of MPICH is called MPICH1 and it implements the MPI-1.1 standard. As time goes there are updates on the software. As of 2006, the latest implementation is called MPICH2 and it implements the MPI-2.0 standard. There is still some bug in the software. The bug is that the MPICH2 does not yet support data translations between different hardware architectures.

There is also another version of the MPICH which called the MPICH-V. It is a research effort with theoretical studies, experimental evaluations and pragmatic implementations aiming to provide a MPI implementation based on MPICH. This MPICH-V features multiple fault tolerant protocols. MPICH-V provides automatic fault tolerant MPI library. As for example, it is totally unchanged application linked with the MPICH-V library is a fault tolerant application. [4] Fault Tolerance application is an application which has been designed to continue operates, possibly at a reduced level, rather than failing completely, when some part of the system fails. [5]

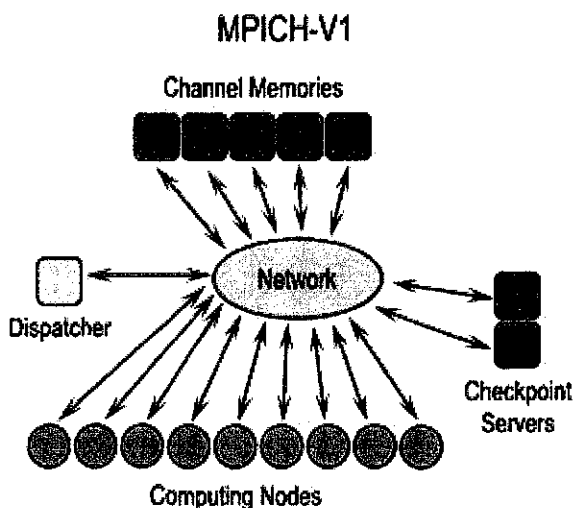
These features of the MPICH-V make it suitable for:

- Large clusters
- Cluster made from collection of nodes in a LAN environment (Desktop Grid)
- Grid deployments harnessing several clusters
- Campus/industry wide desktop Grids with volatile nodes

Currently, MPICH-V features four different protocols. Some researcher and developer are working on a new implementation of all these protocols inside a generic framework. Below are the implementations of the MPICH-V. There are 4 implementation already applied and used.

MPICH-V1

MPICH-V1 features a fault tolerant protocol designed for very large scale computing using heterogeneous networks. The protocol is designed to support transmission of messages across multiple nodes with in a self-healing topology to protect against recursive node and process failures. [6] Its fault tolerant protocol is well suited for Desktop Grids and Global computing as it can support a very high rate of faults, but requires a larger bandwidth for stable components to reach good performance.

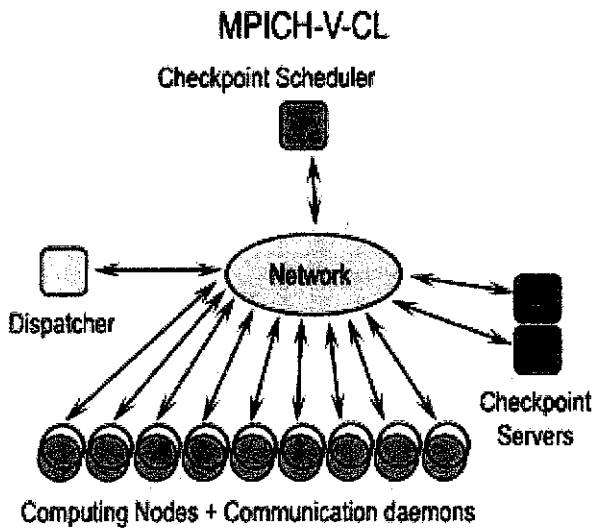


- uncoordinated checkpoint
- Remote pessimistic message logging (all messages are logged on reliable media and used for replay) through Channel Memories

Figure 2.1 The First implementation of the MPICH-V1

MPICH-VCL features a fault tolerant protocol designed for extra low latency dependent applications. The Chandy Lamport algorithm (an algorithm used in distributed systems for

recording a consistent global state of an asynchronous system) used in MPICH-VCL does not introduce any overhead during fault free execution. However, it requires restarting all nodes in the case of a single fault. As a consequence, it is less fault resilient than message logging protocols, and is only suited for medium scale clusters.



- coordinated checkpoint following Chandy-Lamport algorithm
- No overhead during fault free execution
- All nodes (even non faulty) have to be restarted from checkpoint when a crash occurs

Figure 2.2 The Newest Implementation of the MPICH-VCL

2.2 MPICH

MPI uses objects called communicators and groups to define which collection of processes may communicate with each other. Most MPI routines require you to specify a communicator as an argument.

#include "mpi.h" – header file, it is required for all programs which make MPI library call.

MPI_Init - Initializes the MPI execution environment. This function must be called in every MPI program, must be called before any other MPI functions and must be called only once in an MPI program. For C programs, *MPI_Init* may be used to pass the command line arguments to all processes, although this is not required by the standard and is implementation dependent.

Code : **MPI_Init(&argc,&argv)**

MPI_INIT (ierr)

MPI_Comm_size - Determines the number of processes in the group associated with a communicator. Generally used within the communicator *MPI_COMM_WORLD* to determine the number of processes being used by your application.

Code : **MPI_Comm_size(comm,&size)**

MPI_COMM_SIZE (comm,size,ierr)

MPI_Comm_rank - Determines the rank of the calling process within the communicator.

Code : **MPI_Comm_rank(comm,&rank)**

MPI_COMM_RANK (comm,rank,ierr)

MPI_Abort - Terminates all MPI processes associated with the communicator. In most MPI implementations it terminates ALL processes regardless of the communicator specified.

Code : **MPI_Abort(comm,errorcode)**

MPI_ABORT (comm,errorcode,ierr)

MPI_Get_processor_name - Returns the processor name. Also returns the length of the name. The buffer for "name" must be at least `MPI_MAX_PROCESSOR_NAME` characters in size. What is returned into "name" is implementation dependent - may not be the same as the output of the "hostname" or "host" shell commands.

Code: **`MPI_Get_processor_name(&name,&resultlength)`**
`MPI_GET_PROCESSOR_NAME (name,resultlength,ierr)`

MPI_Initialized - Indicates whether `MPI_Init` has been called - returns flag as either logical true (1) or false(0). MPI requires that `MPI_Init` be called once and only once by each process. This may pose a problem for modules that want to use MPI and are prepared to call `MPI_Init` if necessary. `MPI_Initialized` solves this problem.

Code: **`MPI_Initialized(&flag)`**
`MPI_INITIALIZED (flag,ierr)`

MPI_Wtime - Returns an elapsed wall clock time in seconds (double precision) on the calling processor.

Code : **`MPI_Wtime()`**
`MPI_WTIME ()`

MPI_Wtick - Returns the resolution in seconds (double precision) of `MPI_Wtime`.

Code : **`MPI_Wtick()`**
`MPI_WTICK ()`

MPI_Finalize - Terminates the MPI execution environment. This function should be the last MPI routine called in every MPI program - no other MPI routines may be called after it.

Code : **`MPI_Finalize()`**
`MPI_FINALIZE (ierr)`

Example of the MPI program coding:

```
#include <stdio.h>

#include <mpi.h>

int

main(int argc, char *argv[])

{

    int rank, size;

    MPI_Init(&argc, &argv);

    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

    MPI_Comm_size(MPI_COMM_WORLD, &size);

    printf("Hello world! I am %d of %d\n", rank, size);

    MPI_Finalize();

    return 0;

}
```

2.3 Images Enhancement

Enhancements are used to make it easier for visual interpretation and understanding of imagery. The advantage of digital imagery is that it allows us to manipulate the digital pixel values in an image. Although radiometric corrections for illumination, atmospheric influences, and sensor characteristics may be done prior to distribution of data to the user, the image may still not be optimized for visual interpretation. With large variations in spectral response from a diverse range of targets, no generic radiometric correction could optimally account for and display the optimum brightness range and contrast for all targets. Thus, for each application and each image, a custom adjustment of the range and distribution of brightness values is usually necessary.

Spatial filtering encompasses another set of digital processing functions which are used to enhance the appearance of an image. Spatial filters are designed to highlight or suppress specific features in an image based on their spatial frequency. "Rough" textured areas of an image, where the changes in tone are abrupt over a small area, have high spatial frequencies, while "smooth" areas with little variation in tone over several pixels, have low spatial frequencies. A common filtering procedure involves moving a 'window' of a few pixels in dimension over each pixel in the image, applying a mathematical calculation using the pixel values under that window, and replacing the central pixel with the new value. The window is moved along in both the row and column dimensions one pixel at a time and the calculation is repeated until the entire image has been filtered and a "new" image has been generated. By varying the calculation performed and the weightings of the individual pixels in the filter window, filters can be designed to enhance or suppress different types of features.

A low-pass filter is designed to emphasize larger, homogeneous areas of similar tone and reduce the smaller detail in an image. Thus, low-pass filters generally serve to smooth the appearance of an image. The average and median filters, often used for radar imagery. High-pass filters do the opposite and serve to sharpen the appearance of fine detail in an image. One implementation of a high-pass filter first applies a low-pass filter to an image and then subtracts the result from the original, leaving behind only the high spatial frequency information. Directional, or edge detection filters are designed to highlight linear features, such as roads or field boundaries. These filters can also be designed to enhance features which are oriented in specific directions. These filters are useful in applications such as geology, for the detection of linear geologic structures.

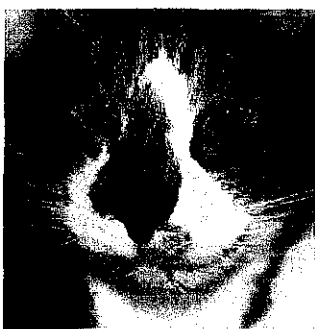
I will create the source code for the image enhancement part. The source code will be written in C language so that it can integrate or combine with the MPICH coding for task distribution.

Enhancement filters will bring out , or enhance the details of the image. The Laplacian filter is commonly used for edge detection. Convolution masks for Laplacian filters are:

-1	0	-1	0	-1	0	-1	-1	-1	-1
0	4	0	-1	4	-1	0	-1	8	-1
-1	0	-1	0	-1	0	-1	-1	-1	-1

In areas with great black to white change (larger slope), the resulting pixel will brighten. This works as edge enhancement for areas of great black to white change (larger slope).

Example:



Original Image



After Laplacian Filter



Contrast Enhanced

2.4 Developer's Image Library

As for my progress I have found a image library that can help me to enhance the image after extracting the images form a video. I will use the Developer's Image Library to make simple programming on how to load, enhance and save images. Image enhancement is an application that I need to show that the distributed application, MPICH, works. There are many ways to enhance images. But for my project, I chose to sharpen an image as an action to enhance the images.

This Developer's Image Library was known as Open Image Library (OpenIL), but the name was changed due to SGI's request. Developer's Image Library (DevIL) is a programming library that has a lot of image loading capabilities, yet is easy for a developer to learn and use. There are function on filter, display, save, delete, error detection and many other images processing function. It also support variety image formats. I have used some of the library function in my programming. Below are the explanations of my of some of the library function that I use:

2.4.1 Load Image

Firstly, I need to load the image that needs to be enhanced. There is a function that calls `iLoadImage` that make it easy to load image. But for most programs, a simple call to `iLoadImage` will suffice. IF the image was not loaded due to any of various reasons, `iLoadImage` returns false, else it returns true if the image was successfully loaded. This will be explained further in the error detection part.

Code for loading an image:

```
ILboolean iLoadImage(char * FileName );
```

FileName = Specifies which file to load an image from.

Example: `iLoadImage("soccer.jpg");`

2.4.2 Enhance Image

After the image has been load, I need to enhance the image. So I will use the `iluSharpen` function which has been included in the DevIL. `iluSharpen` can actually either sharpen or blur an image, depending on the value of `Factor`. There are also other functions which are `iluBlurAvg` and `iluBlurGaussian`. This function will give faster result for blurring. When the factor is set to 1.0, the image goes unchanged. When `Factor` is in the range 0.0 – 0.9, the current image is blurred. When `Factor` is in the range 1.1 - 2.5, the current image is sharpened. To achieve a more pronounced sharpening or blurring effect, simply increase the number of iterations by increasing the value passed in `Iter`.

```
ILboolean iluSharpen(ILfloat Factor, ILint Iter);
```

`Factor` = Factor to sharpen by.

`Iter` = Number of iterations to perform on the image.

2.4.3 Saving Image

After load the image and enhance it, the next step is to save the image. Saving the result image is also easy with the help of DevIL. I just need call `ilSaveImage` function with the desired filename as the only parameter. If DevIL could not save the image, `ilSaveImage` returns false, else it returns true. By default, DevIL will refuse to overwrite any images that already exist on the hard drive to prevent from overwriting important data. But, if we want to change this behavior to allow overwriting of files, I need to use the `ilEnable` function with the `IL_FILE_OVERWRITE` parameter.

Code for saving an image:

```
ILboolean ilSaveImage( char * FileName );
```

`FileName` = Specifies which file to save an image to

```
Example: ilEnable(IL_FILE_OVERWRITE);  
         ilSaveImage("socceredit.jpg");
```

2.4.4 Error Detection

This is an extra function in programming. It will not affect the programming if this function is not included in the program. But, if it is included it will make the program more users friendly. There will occasionally errors may occur in programming, same as in DevIL, such as an image not being loaded. If a DevIL function returns indicating an error an error code is set internally in DevIL and may be retrieved via `ilGetError`. Usually, the code is quite specific about what kind of error occurred. DevIL maintains an error stack (usually 32 errors deep) so that if more than one error is set, an error does not get "lost". When the function call `ilGetError` is call, the last error set is popped off of the stack. If no error has occurred, or all the errors have been popped off of the stack, `ilGetError` returns `IL_NO_ERROR`.

Code for saving an image :

```
ILenum ilGetError(ILenum Mode );
```

Mode = The mode value to be returned

Example : ILenum Error;

```
Error = ilGetError();
```


2.4.5 Display the Image

Another function that the library has is displaying images. So, I used this function to create a function to display the result image of my function. The DevIL has several several different APIs. It can be use for displaying an image through `ilut` function. Since `ilut` is separate from `il`, I can manually send data to the API just as `ilut` does. But before I can calll any `ilut` functions, I need to deal with OpenGL and after I have initialized OpenGL, I must call `ilutRenderer` with the `ILUT_OPENGL` parameter to initialize `ilut` correctly.

Most applications will then only need to call `ilutGLBindTexImage` to get a corresponding OpenGL texture from the DevIL image. If you only need to use the OpenGL texture and not the DevIL image after this, it is safe to delete the image.

Example of getting an OpenGL texture:

```
ILboolean ilutGLTexImage(GLuint Level);
```

Level = Texture level to place the image at. 0 is the base image level, and anything lower is a mipmap. Use `ilActiveMipmap` to access OpenIL's mipmaps.

```
GLuint Texture;
```

```
Texture = ilutGLBindTexImage().
```

2.5 Introduction to Audio Video Interleave

Audio Video Interleave also known as AVI, is a multimedia container format introduced by Microsoft in November 1992 as part of its Video for Windows technology. Video quality can be very good at smaller resolutions, but files tend to be rather large. These files can contain both audio and video data in a standard container that allows synchronous audio-with-video playback. It also supports multiple streaming audio and video, although these features are seldom used. These files are supported by Microsoft, and are unofficially called "AVI 2.0". [7]

Video clips on the World Wide Web are usually available in both AVI and QuickTime formats. In AVI, picture and sound elements are stored in the file as interleaved chunks of data. The first sub-chunk is identified by the "hdrl" tag. This sub-chunk is the file header and contains metadata about the video, such as its width, height and frame rate. The second sub-chunk is identified by the "movi" tag. This chunk contains the actual audio/visual data that make up the AVI movie. The third optional sub-chunk is identified by the "idx1" tag which indexes the physical addresses [within the file] of the data chunks. Upon creation of the file, the codec translates between raw data and the (compressed) data format used inside the chunk. [8]

A video file will typically store a movie clip. It's generally supported by many different platforms, although note that there are several different versions of AVI files in use, and not all players will play all versions.

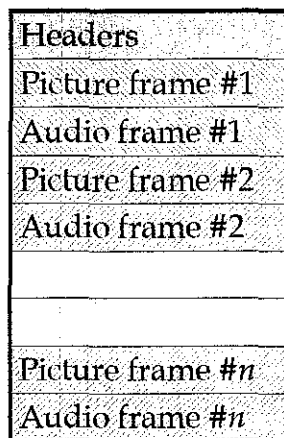


Figure 2.3 AVI Diagram

Chapter 3 Methodology

3.1 Procedure Identification

Throughout this project, Spiral Development Model methodology was chosen as the development methodology. This spiral methodology is a programming system that enables programmers to quickly build working programs. This model of development combines the features of the prototyping model and the waterfall model. This methodology really shows that the iteration is really important. This method also favored for complex project development.

There are some advantages of using Spiral methodology compared to the other traditional sequential development such as Rapid Application Development model. This model was not the only iterative development, but it was the first model to explain why the iteration matters. In spiral model, each phase starts with a design goal and ends with the supervisor or client reviewing the progress thus far. Analysis and engineering efforts are applied at each phase of the project, with an eye toward the end goal of the project. The iteration concept is perfectly ideal with the nature that human can't avoid mistakes and imperfect.

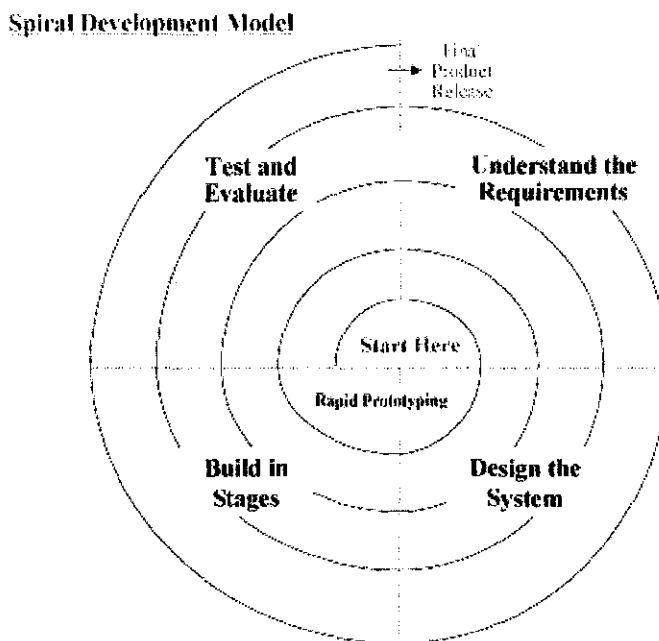


Figure 3.1 Spiral Development Model

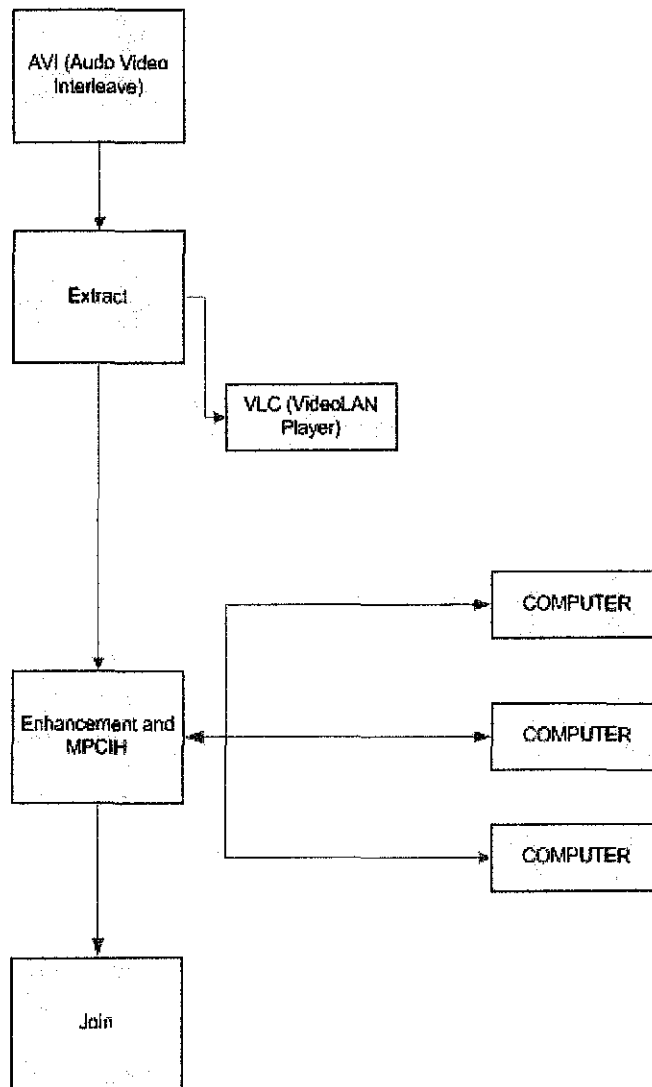
3.2 Tools Required

First and foremost, every software that is going to be developed, it needs a programming software which support the language that we going to use in order to create the program. As for my project, I will be using the GCC software for the programming part. This is likely to be one of the famous programming software that widely use by software developer in Linux based. It has many features that needs. It supports C++ language which I am going to use to develop my software.

For the grid computing, it is a must to use a lot of computer and Message Passing Interface. This computer is already available at the Multimedia Lab, University Technology of Petronas. I will be using MPICH as the MPI. The configuration has not yet been decided because before the configuration been done there are many process that and research to be done.

There are quite a few other tools available that I believed will be used in the development of this project. I would say I “believed” because I don’t really sure what the specific tools needed for the time being. But, I am sure it has something to do with all the tools stated above.

3.3 System Flow



Firstly I have to have an avi file format in order to proceed with my project. After I have the video in avi format, I need to extract the file to frames. The result then will be saved under different names. But the names are sort by number ascending. The result will be enhanced by frame or by batch. By batch mean that I can define frame 1 to frame 100 to be enhance by specific filter. The enhancement process will be done by some coding and will be process n the grid computing. The MPICH will be assigning the task to all nodes. After the enhancement task are done, the enhance images then will be collected and then join back together to form a video which in better quality from the original video without missing any important data or information. This is the system flow of my project.

Chapter 4

Result and Discussion

As for my project, there will be a lot of step included. Below are the main step that are required:

- Get an avi file
- Extract to frames(jpeg, BMP or gif file)
- Enhance the image by frames
- Configure MPICH to pass the job
- Joining the result back to video

4.1 Extracting Video to Frames

In my project, I require to extract the video (AVI) to frames as I need to enhance the video by frames. Below are the steps that I do to extract the video to frames that will be in jpeg format:

1. Firstly I boot the ubuntu Operating system and login. After logon to the ubuntu. I open one terminal.

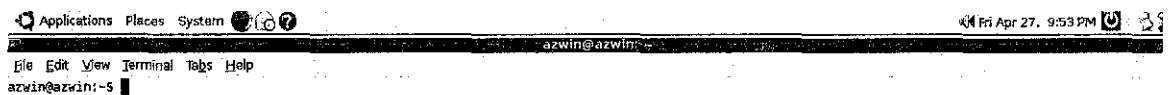


Figure 4.1 Open Terminal

2. Then I open the directory where I have samples video that is in avi format. I put the video in specific video which is in folder that named “extract”. Then I open the directory.
Command line : “cd extract”.



Figure 4.2 Open Directory

3. After entering the extract folder, to make sure that the video file is there, I use the ‘dir’ command to view any file that in the folder.

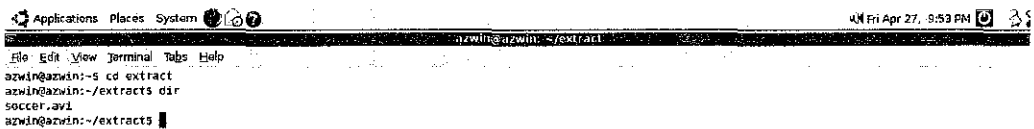


Figure 4.3 Explore Directory

This picture below is the view of file using the file browser. The file browser is another way to view file in directory. This is the easy way to view files in the folder because it allows click on the link.

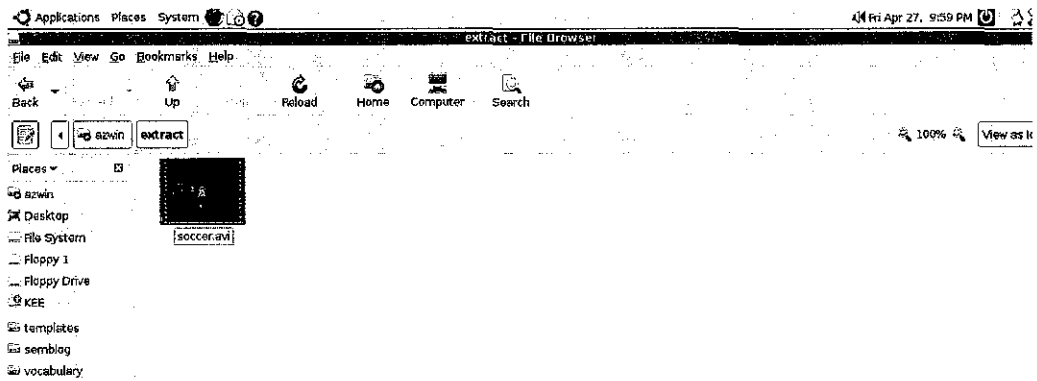


Figure 4.4 File Browser

4. To extract the video to frames and save it as a still image in jpeg format, I use the VideoLan Player (VLC). Any version of the VLC can extract the video. I also can use the ffmpeg software or mencoder software to extract the video, but as VLC is already embedded in Ubuntu, I just use it because it seems easier. The command is “`vlc -V image --image-out-format jpeg --image-out-ratio 3 --image-out-prefix xxxx /usr/file.avi`”.

Command	Description
Vlc	To open the software
--image-out-format	To set the format of the output file. For this I have chosen to save it to jpeg format. It also can be saved as any other picture file format.
--image-out-ratio	To set the parameter of the outputs file. It sets the video ratio. I choose to set it to 3. Means every 3 frames per second it will print the still image and save it to jpeg format.
--image-out-prefix	To set names for the output file. Prefer a simple file name to make easier to remember.
/usr/file.avi	To define the directory of the file that needs to be extracted.

Figure 4.5 Command Line Descriptions

Below is the figure of way that the extracting video been executed.

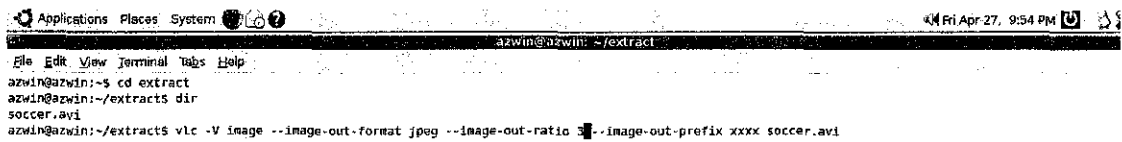


Figure 4.6 Execute extract file command

5. This figure shows while the extracting is in process. The VLC software is run and it show the length of time taken or time left.

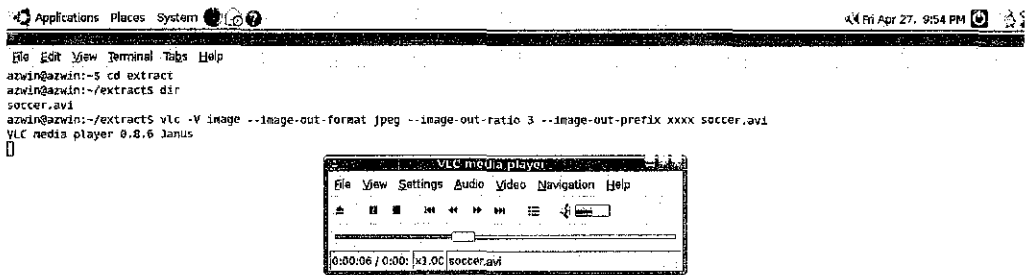


Figure 4.7 Extracting In Processes

6. This figure below shows that the extracting processes are done.

```

Applications Places System 10:20
azwin@azwin: ~/extract
File Edit View Terminal Tabs Help
azwin@azwin:~$ cd extract
azwin@azwin:~/extracts$ dir
soccer.avi
azwin@azwin:~/extracts$ vlc -V image --image-out-format jpeg --image-out-ratio 3 --image-out-prefix xxxx soccer.avi
VLC media player 0.8.6 Janus
[00000280] main playlist: nothing to play
[00000280] main playlist: stopping playback
azwin@azwin:~/extracts$

```

Figure 4.8 Extracting process finished

7. To view the result of the extracted video. I explore the directory of the output image. For this I just explore the directory of the video that is use to be the sample of the extracting part. We also can set the output file to be in the different folder.

```

Applications Places System 10:20
azwin@azwin: ~/extract
File Edit View Terminal Tabs Help
azwin@azwin:~$ cd extract
azwin@azwin:~/extracts$ dir
soccer.avi
azwin@azwin:~/extracts$ vlc -V image --image-out-format jpeg --image-out-ratio 3 --image-out-prefix xxxx soccer.avi
VLC media player 0.8.6 Janus
[00000280] main playlist: nothing to play
[00000280] main playlist: stopping playback
azwin@azwin:~/extracts$ dir
soccer.avi
xxxx000020.jpeg xxx0000041.jpeg xxx0000062.jpeg xxx0000063.jpeg xxx0000083.jpeg xxx000104.jpeg xxx000125.jpeg xxx000146.jpeg xxx000167.jpeg
xxxx000000.jpeg xxx0000021.jpeg xxx0000042.jpeg xxx0000064.jpeg xxx0000084.jpeg xxx000105.jpeg xxx000126.jpeg xxx000147.jpeg xxx000168.jpeg
xxxx000001.jpeg xxx0000022.jpeg xxx0000043.jpeg xxx0000065.jpeg xxx0000085.jpeg xxx000106.jpeg xxx000127.jpeg xxx000148.jpeg xxx000169.jpeg
xxxx000002.jpeg xxx0000023.jpeg xxx0000044.jpeg xxx0000066.jpeg xxx0000086.jpeg xxx000107.jpeg xxx000128.jpeg xxx000149.jpeg xxx000170.jpeg
xxxx000003.jpeg xxx0000024.jpeg xxx0000045.jpeg xxx0000067.jpeg xxx0000087.jpeg xxx000108.jpeg xxx000129.jpeg xxx000150.jpeg xxx000171.jpeg
xxxx000004.jpeg xxx0000025.jpeg xxx0000046.jpeg xxx0000068.jpeg xxx0000088.jpeg xxx000109.jpeg xxx000130.jpeg xxx000151.jpeg xxx000172.jpeg
xxxx000005.jpeg xxx0000026.jpeg xxx0000047.jpeg xxx0000069.jpeg xxx0000089.jpeg xxx000110.jpeg xxx000131.jpeg xxx000152.jpeg xxx000173.jpeg
xxxx000006.jpeg xxx0000027.jpeg xxx0000048.jpeg xxx0000070.jpeg xxx0000090.jpeg xxx000111.jpeg xxx000132.jpeg xxx000153.jpeg xxx000174.jpeg
xxxx000007.jpeg xxx0000028.jpeg xxx0000049.jpeg xxx0000071.jpeg xxx0000091.jpeg xxx000112.jpeg xxx000133.jpeg xxx000154.jpeg xxx000175.jpeg
xxxx000008.jpeg xxx0000029.jpeg xxx0000050.jpeg xxx0000072.jpeg xxx0000092.jpeg xxx000113.jpeg xxx000134.jpeg xxx000155.jpeg xxx000176.jpeg
xxxx000009.jpeg xxx0000030.jpeg xxx0000051.jpeg xxx0000073.jpeg xxx0000093.jpeg xxx000114.jpeg xxx000135.jpeg xxx000156.jpeg xxx000177.jpeg
xxxx000010.jpeg xxx0000031.jpeg xxx0000052.jpeg xxx0000074.jpeg xxx0000094.jpeg xxx000115.jpeg xxx000136.jpeg xxx000157.jpeg xxx000178.jpeg
xxxx000011.jpeg xxx0000032.jpeg xxx0000053.jpeg xxx0000075.jpeg xxx0000095.jpeg xxx000116.jpeg xxx000137.jpeg xxx000158.jpeg xxx000179.jpeg
xxxx000012.jpeg xxx0000033.jpeg xxx0000054.jpeg xxx0000076.jpeg xxx0000096.jpeg xxx000117.jpeg xxx000138.jpeg xxx000159.jpeg xxx000180.jpeg
xxxx000013.jpeg xxx0000034.jpeg xxx0000055.jpeg xxx0000077.jpeg xxx0000097.jpeg xxx000118.jpeg xxx000139.jpeg xxx000160.jpeg xxx000181.jpeg
xxxx000014.jpeg xxx0000035.jpeg xxx0000056.jpeg xxx0000078.jpeg xxx0000098.jpeg xxx000119.jpeg xxx000140.jpeg xxx000161.jpeg xxx000182.jpeg
xxxx000015.jpeg xxx0000036.jpeg xxx0000057.jpeg xxx0000079.jpeg xxx0000099.jpeg xxx000120.jpeg xxx000141.jpeg xxx000162.jpeg xxx000183.jpeg
xxxx000016.jpeg xxx0000037.jpeg xxx0000058.jpeg xxx0000080.jpeg xxx000100.jpeg xxx000121.jpeg xxx000142.jpeg xxx000163.jpeg
xxxx000017.jpeg xxx0000038.jpeg xxx0000059.jpeg xxx0000081.jpeg xxx000101.jpeg xxx000122.jpeg xxx000143.jpeg xxx000164.jpeg
xxxx000018.jpeg xxx0000039.jpeg xxx0000060.jpeg xxx0000082.jpeg xxx000102.jpeg xxx000123.jpeg xxx000144.jpeg xxx000165.jpeg
xxxx000019.jpeg xxx0000040.jpeg xxx0000061.jpeg xxx0000083.jpeg xxx000103.jpeg xxx000124.jpeg xxx000145.jpeg xxx000166.jpeg
azwin@azwin:~/extracts$

```

Figure 4.9 Extracted File from the video

This figure below is the preview of the results. I use the file browser to have better image of the result of the extracting process.



Figure 4.10 Explore extracted file using file browser

This figure below is the sample of one of the result that I get from the video extracting process.

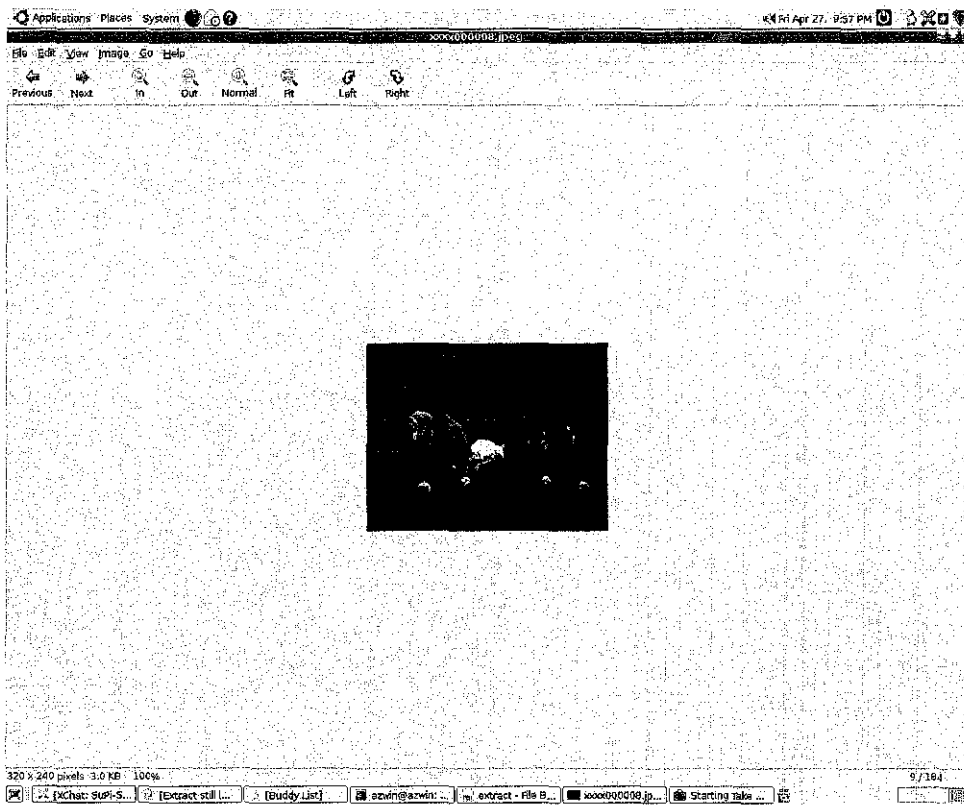


Figure 4.11 Frame of extracted image

4.2 Running Developer's Image Library

I have run the coding that I have created using Microsoft Visual Studio 2005. The codes that I have generated will accept the image as an input that the user declares. Then the program will take the image as input then apply the contrast filter. Below are my coding to apply the contrast filter to the images.

The Devil Code:

```
#include <il.h>
#include <ilu.h>
#include <stdio.h>

int main(int argc, char **argv)
{
    ILuint  ImgId;
    ILenum Error;

    // We use the filename specified in the first argument of the command-line.
    if (argc < 2) {
        fprintf(stderr, "DevIL_test : DevIL simple command line application.\n");
        fprintf(stderr, "Usage : DevIL_test <file> [output]\n");
        fprintf(stderr, "Default output is test.tga\n");
        return 1;
    }

    // Check if the shared lib's version matches the executable's version.
    if (ilGetInteger(IL_VERSION_NUM) < IL_VERSION ||
        iluGetInteger(ILU_VERSION_NUM) < ILU_VERSION) {
        printf("DevIL version is different...exiting!\n");
        return 2;
    }

    // Initialize DevIL.
    ilInit();

    // Generate the main image name to use.
    ilGenImages(1, &ImgId);
```

```

// Bind this image name.
ilBindImage(ImgId);

// Loads the image specified by File into the image named by ImgId.
if (!ilLoadImage(argv[1])) {
    printf("Could not open file...exiting.\n");
    return 3;
}

// Display the image's dimensions to the end user.
printf("Width: %d Height: %d Depth: %d Bpp: %d\n",
    ilGetInteger(IL_IMAGE_WIDTH),
    ilGetInteger(IL_IMAGE_HEIGHT),
    ilGetInteger(IL_IMAGE_DEPTH),
    ilGetInteger(IL_IMAGE_BITS_PER_PIXEL));
iluContrast(0.5);

// Enable this to let us overwrite the destination file if it already exists.
ilEnable(IL_FILE_OVERWRITE);

// If argv[2] is present, we save to this filename, else we save to test.tga.
if (argc > 2)
    ilSaveImage(argv[2]);
else
    ilSaveImage("test.tga");

// We're done with the image, so let's delete it.
ilDeleteImages(1, &ImgId);

// Simple Error detection loop that displays the Error to the user in a human-readable form.
while ((Error = ilGetError()) {
    printf("Error: %s\n", iluErrorString(Error));
}

return 0;
}

```

4.3 MPICH2 Configurations

4.3.1 Running MPICH2 Programs

Here I get to the configuring the MPICH2. As I use Windows as platform for my project, MPICH2 offer Graphical User Interface to configure the software. There are two methods. The first is on a dual processor machine. The second is across a network by specifying the IP addresses. Across a network is typically slow due to communication times. But I choose to use to use the MPICH2 across the network because these really show the purpose of my projects. Below are the steps to show the configurations:

ACROSS A NETWORK

1. To run without passwords several steps are required (mpich2-doc-windev.pdf).
DOMAIN Administrative rights are required.
 - a. On each node execute: "smpd -register_spn"
 - b. All jobs must be submitted with the -delegate command.
2. Copy the executable to each machine. This should be in the same directory structure as the MASTER node. For example, "C:\bspaul\helmholtz.mpi".
3. The Windows Firewall must be adjusted to allow MPICH2 to run.
 - a. Bring up the Windows Firewall from the Control Panel.
 - b. From the Exceptions Tab, select "Add Program" and make sure "C:\mpich2\bin\smpd.exe" and "C:\mpich2\bin\mpiexec.exe" are on the list.
 - c. From "Add Program" add the executable to the exceptions. N.B. If the Windows Security alert appears then make sure the executable has been added.
4. Log on to each machine you want to run on. This is not required, but prevents anyone else logging on and using those machines.
5. Run the program from a command prompt from the MASTER node by typing:
"mpiexec -hosts X hostname.1.com hostname.2.com hostname.X.com"
X is the number of hosts being used and hostname.X.com are the names of the machines running one. Verification can be made by checking the Windows Task Manager of each machine to verify they are at 100%.

Images of MPICH2 Graphical User Interface

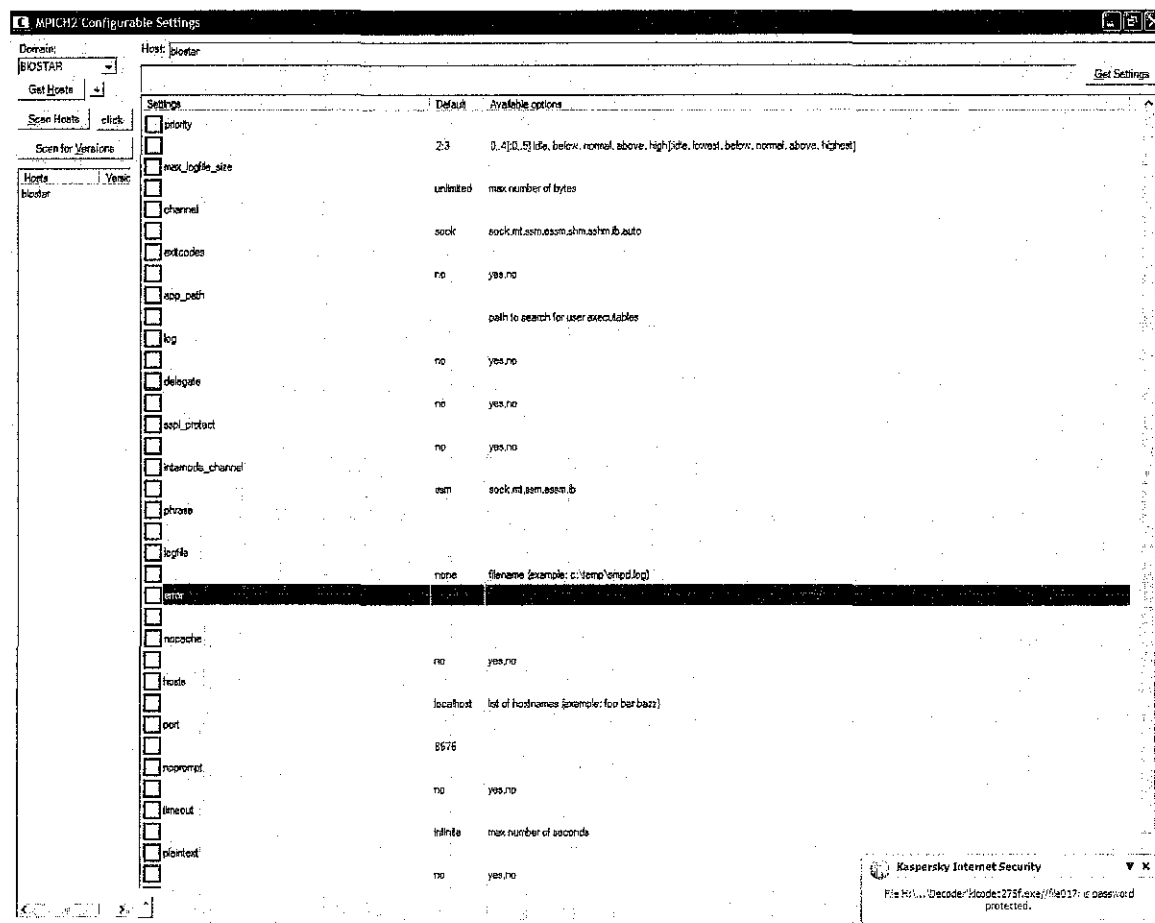


Figure 4.3.1.1 Configuring the MPICH2

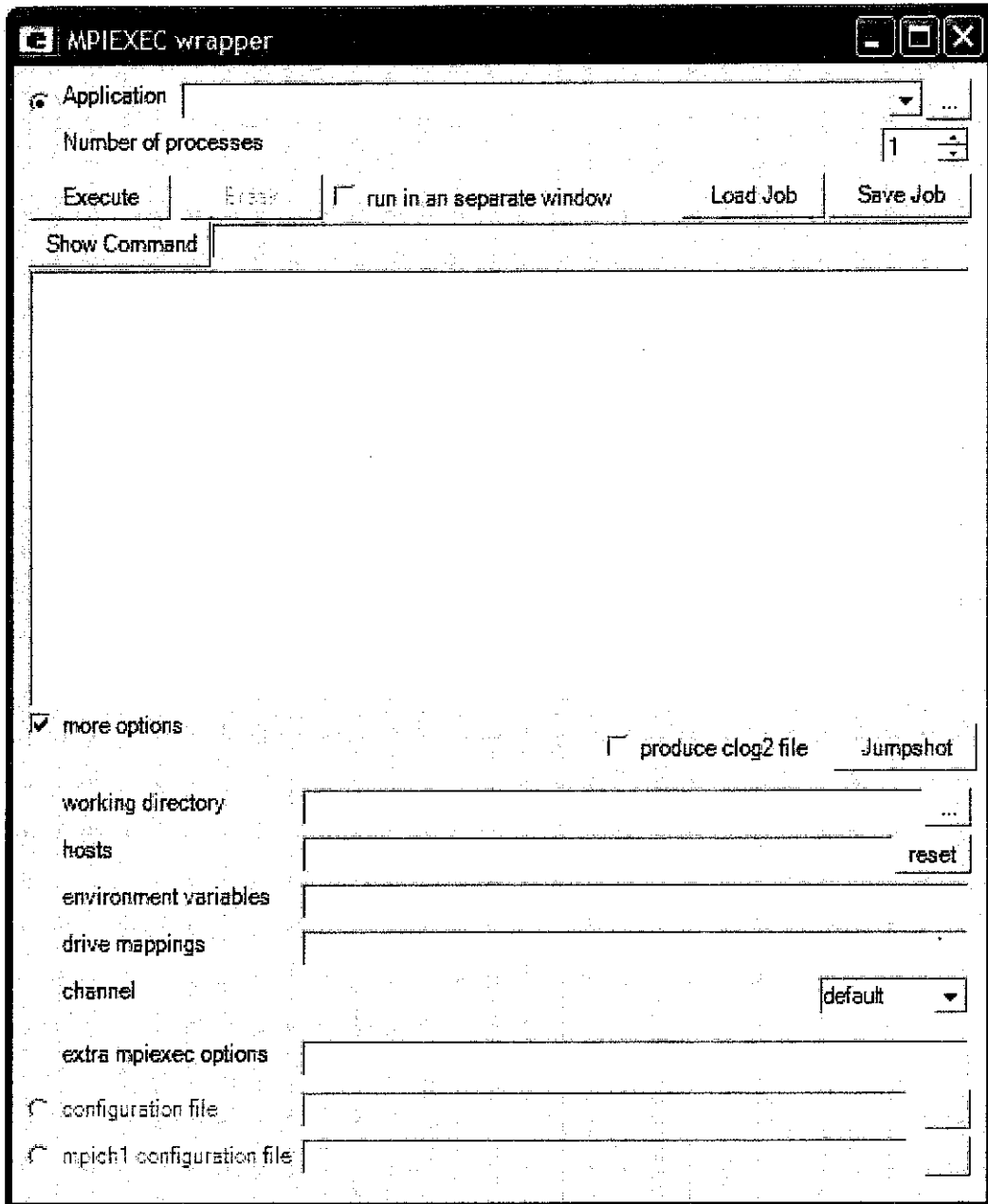
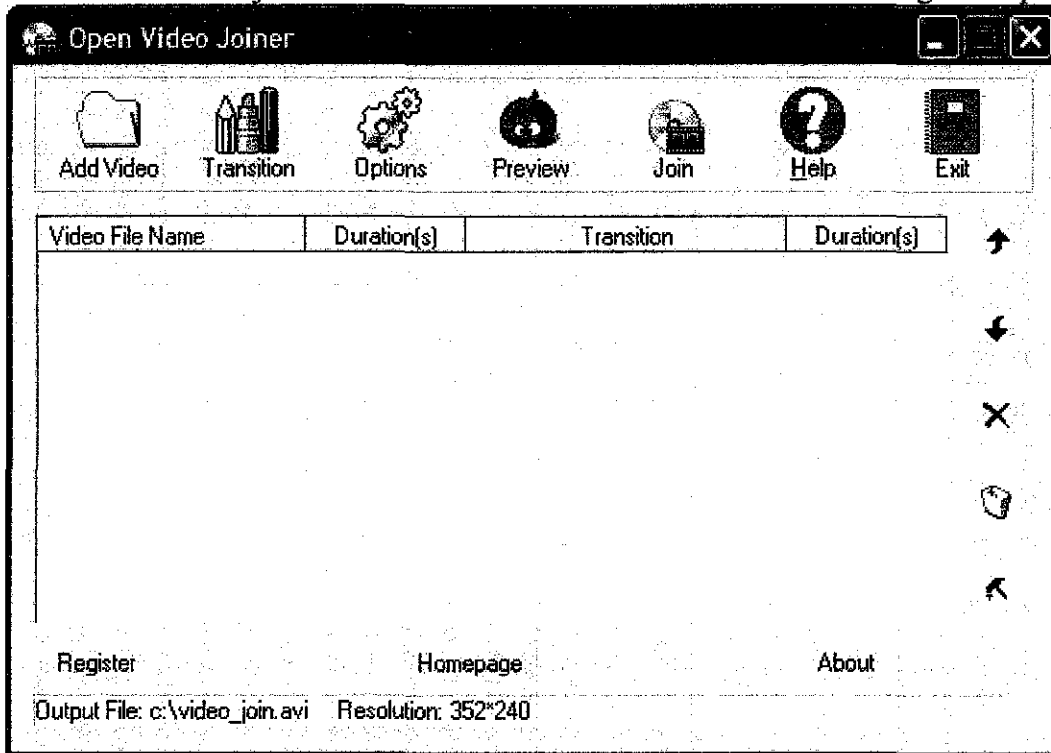


Figure 4.3.1.2 Gui of the Mpiexec.exe

4.4 Image Joiner

After I get the result from the MPICH2 (after apply contrast filter) I need to join the image to make it back to video format. I have decided to use the Open Video Joiner. It can join image to form it back to video. Main purpose of the tool is to make slideshow of image. But, it also can form a video in AVI format and even MPEG. It accepts the use of codec like DIVX and Xvid to form the video. Because of the codec, it also can set the video setting. It can set the resolution and bitrates. It also can join back the sound of the video. Below are the images samples of the tool.



1. Click the Add Video button, select the video files.

2. Click the Options button to change the output file name, frame rate, frame size, resize mode, audio compressor and video compressor.

To change each transition and the transition duration, click the Transition button or rightclick an item in the list.

To view each clip, doubleclick an item in the list, Window Media Player will play it automatically.

To preview the transition effect and the output video sequence, click the Preview button. The duration of each clip is limited in 10 seconds for your preview the output quickly.

3. At last click the Join button to Join all video files.

5.0 Conclusion

As the conclusion we can conclude that the implementation of video enhancement using grid computing can be very hard and challenging. We must understand the architecture thoroughly before we can apply the program. There is already a lot of software for video enhancement. But, it is not many that are compatible with the grid computing.

The research must not only been done to the tools that is stated above, it must cover more than one tools, this is to make sure that the chosen tools is the best and suitable for the projects. That is why we need a lot of time in the research part. As for this paper, there is some of the future work that I would like to explore in the near future. I certainly would like to do more study on the configuration of the MPICH and the architecture. I must also do more research in the coding about the software of video enhancement that I am going to develop. I have to make sure that the software is compatible with the grid computing architecture. I must do more study because what I only discuss these report are in general but not in depth. To make this project a success, I must do a lot of research before I proceed with the development.

References

1. <http://dret.net/glossary/avi>
2. http://en.wikipedia.org/wiki/Grid_computing
3. http://en.wikipedia.org/wiki/Message_Passing_Interface
4. <http://en.wikipedia.org/wiki/MPICH>
5. http://en.wikipedia.org/wiki/Fault-tolerant_design
6. http://icl.cs.utk.edu/news_pub/submissions/Scalable-ft-euro-pvmmmpi-2006.pdf
7. <http://faydoc.tripod.com/formats/avi.htm>
8. <http://www.videohelp.com/>
9. http://linux.about.com/library/cmd/blcmd11_ImageMagick.htm