Generalized Processing Application for Analysis of Large Scale Matrices of Human Motion Data

Satyanarayana Sundeep Nadendla

**Problem Report submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of**

**Master of Science
in
Electrical Engineering**

**Lane Department of Computer Science and Electrical Engineering
Morgantown, West Virginia
2009**

Keywords: Human Motion Data, WCF, WPF, MATLAB**.**

Abstract

Generalized Processing Application for Analysis of Large Scale Matrices of Human Motion Data

Satyanarayana Sundeep Nadendla

Measuring human performance, specifically with respect to individual capabilities and limitations, has long required exhaustive and subjective assessments. There are a number of existing software applications for visualizing Human Motion Data. There are many limitations to those applications. There is some functionality for the users to browse through and analyze the output data, although this is a costly and time-intensive process, and a prohibitive burden in large datasets. Also, none of these applications are distributed and hence users need to have a full set of software to analyze the data.

The Generalized processing application addresses these issues by intelligently processing the data files and allows the users to easily access information of their choice. The application has a remote server module which accepts requests from the client and processes those requests and sends the necessary information back to the client application. The output information, in most cases, is in the form of plots, which can be displayed, saved and printed.

Acknowledgements

I would like to express my gratitude to my advisor **Dr. Kevin A. Rider** for his supervision and valuable guidance throughout the project. I am very thankful for his constant support and encouragement. He always helped me to come up with ideas and implement them.

I would like to extend my thanks to **Dr. Ramana Reddy and Dr. James Mooney** for serving on my committee and supporting the project with their encouragement. I would like to thank all my friends for their support.

Finally, I am deeply thankful to **my beloved parents and brother** for their encouragement and support in all the things I do.

List of Figures

Table of Contents

Chapter 1. Introduction

Measuring human performance, specifically with respect to individual capabilities and limitations, has long required exhaustive and subjective assessments. More recently, technology has provided increasingly powerful and objective tools that can measure human kinematic and kinetic performance. Human motion capture has embedded itself between a seeming dichotomy of fields: animated motion pictures (e.g. Toy Story and Shrek), and scientific research. While the empirical concept is over 100 years old, the seminal work of Muybridge in the late 1890's was only useful for limited analysis of specific components of individual subjects. An example of his frame-by-frame analysis of kinematics of horse galloping is shown in Figure 1.

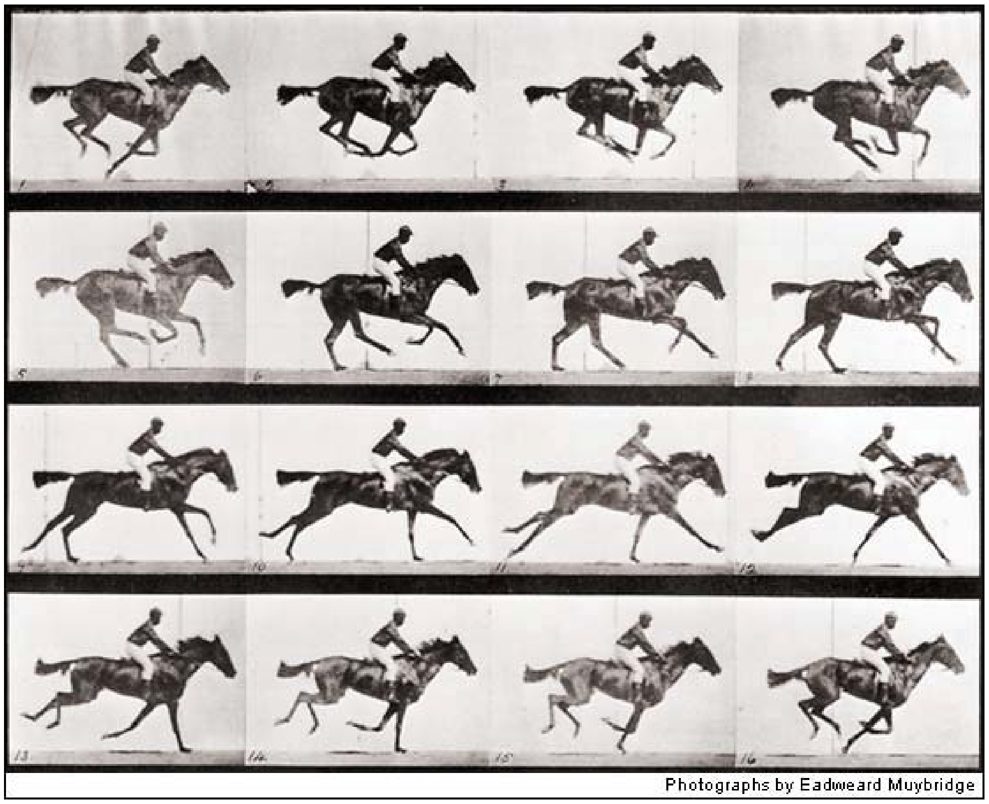


Figure 1 Kinematic snapshots of a galloping horse (Muybridge, 1978)

Despite the innovations of modern photography and video recording, both technologies are relegated to a two-dimensional (2D) view of the subject. The development of direct linear transform (DLT) provided the essential functionality of creating a three-dimensional (3D) perspective (ref?). Today, motion capture companies manufacture high-end computing systems and software that recreates a 3D digital representation from the 2D images from multiple cameras. The number and resolution of the cameras necessary to capture the desired information depends primarily on the nature of the movements being performed. In theory, each additional camera used will provide improved spatial resolution of the subject, but at a cost of increased data processing. The storage, mining, and analysis of these data can be prohibitive for individuals wishing to utilize the technology. Current practices for data analysis may require as many as 30 minutes of processing to analyze each minute of raw data captured. There are several time-intensive steps involved in this process, but one of the most significant is the time required to organize and prepare recorded data for statistical analysis and to create visual images based on gigabytes of collected data.

This proposal is intended to describe the procedure by which large scale matrices of human motion data can be rapidly searched, organized, and plotted, depending on user-specific parameters through a front end graphical user interface (GUI).

Chapter 2. Description of Collected Data (Ballast Study)

For the purpose of this project, data will be used from a previously conducted experiment evaluating the biomechanical effects of ground surfaces (i.e. ballast) on the hips, knees, and ankles of railroad workers. This study was conducted in the Ergonomics Laboratory under the initial development of Dr. Steven Wiker, and the execution and data processing by Dr. Kevin Rider. Eight MX13 (1.3 megapixel) VICON cameras were used to record the positions of 39 markers (14 mm spherical balls covered with reflective tape). These 39 markers constitute the marker set of anatomical landmarks, by which the VICON Nexus software can reconstruct the 3D positions of each marker, and “attach” a digital avatar, or manikin, for subsequent analyses.

Ten paid volunteer subjects participated in this experiment, requiring approximately two hours of their time. Subjects were asked to perform several motions onto three different surfaces: smooth ground, yard gravel, and mainline ballast (common to railyards). The activities performed included normal walking, sitting down and arising from a chair, stepping onto and down from an 8-inch curb, and climbing a multiple-rung ladder (simulating the side of a railroad box car). For the ladder climbing, subjects were required to always have three points of contact with the ladder supports and/or the ground, per regulations of the commercial railroad industry. Among other reasons, this regulation inhibits workers from jumping down from the ladder, increasing the probability of a lower limb or back injury. However, as can be demonstrated, workers are capable of descending from the third rung from the ground, without significant discomfort. As such, during the experiment, subjects were instructed to descend to the ground from the first (bottom), second, and third rungs. Ten subjects each performed three replications of eleven experimental tasks, resulting in 330 motion files, in an industry standard C3D file format. These files ranged in length from 5 seconds to 30 seconds, and collectively utilized four gigabytes of the hard disk drive's volume.

After the raw data was recorded, an initial processing routine was performed that attempted to the label each of the retro-reflective markers with the correct anatomical label. Upon completion of this automated batch processing, the first time-intensive step began, by which four graduate research assistants manually inspected and corrected each recorded motion file. A subsequent batch processing routine uses the data within corrected motion files to calculate the kinetic variables of interest, including the forces, moments, and powers of the hip, knee, and ankle joints.

It is a significant burden to effectively retrieve, investigate, and utilize specific portions of the dataset. Statistical analyses are useful for determining significant differences between means, maximum and minimum values, as well as many other discrete metrics. However, there are not effective methods to qualitatively compare the time-series information of human motion data. More importantly, before statistical tests can be performed, a visual inspection must be done on the data to determine the fidelity of the processed data. After the kinematic and kinetic post processing, as many as 139 variables with a total of nearly 400 degrees of freedom are created. There does not currently exist a way to search through the extensive dataset for variables of interest, and view the relevant time series data.

Chapter 3. Potential Solutions

3.1 Time Intensive Manual Efforts:

Human motion capture is widely used technique across various fields, including entertainment, sports, medical and research laboratories. Modern video and optoelectric motion capture systems are fast, accurate and reliable, and have applications extending from use in hospitals and clinics to the high-tech entertainment industry. While the entertainment industry is mostly concerned with the qualitative aspects of human movement, for example how bodies look when in motion, the research field's primary concern remains quantitative. Scientists and various government organizations use motion capture technique for analyzing various physical attributes of various people doing various jobs. Calculating the maximum Knee angle as part of a Ballast study is one example.

In the context of motion capturing for analysis of human motion based on their gait, various advanced motion capture systems are used. VICON is the leading motion capture system provider. It enables the users to produce desired data for a subject based on the sensor movement captured by the system. The sensors are placed on the subject’s body at various places of interest upon which the analysis is conducted.

A typical Motion capture experiment involves motion capture of at least a dozen sensors. The VICON system produces output data with the time based 3D spatial coordinates for each of the variable marker (sensor). The output can be analyzed by just viewing the output data.

The process of analyzing data directly, i.e. manually, without the use of any visualization technique is very time intensive and may take forever. Imagine analyzing an output data set with hundreds of variables and three spatial coordinates for each marker recorded along a time line.

Therefore, there is a definite need for a visualization technique which enables the user to easily select the variables needed and analyze the data in an efficient way. This saves a lot of time and also eases the operator’s work.

A computational technique will be a perfect choice for the problem at hand. This technique should automatically work on the selected data and display the output in a desired form will cater the requirement.

3.2 Existing Computational Applications and their Limitations:

There are a number of existing software applications for visualizing the Human Motion Data. These applications are used to measure single values like walking speed and stride length and also static values like range of motion of joints.

But, there are problems with those applications. First thing is there will be a huge amount of data, possibly over 100 graphs. The users have no choice of which graph he wants to view and there is no easy access for the user.

Secondly, the complexity of the human neuromuscular system makes it difficult to accurately pinpoint the underlying change seen in the data. For example, deviation of the knee angle from maximum knee angle.

Thirdly, the applications are supposed to be run on the local machine where the files are actually saved. There are no distributed applications which enable the user to just upload the data file / data directory into the application. Therefore these applications need them to be installed in all the computers where the data is saved for using them.

Chapter 4. Generalized Processing Application (GPA):

In the view of developing a more efficient application, the GPA is expected to be more user friendly, faster, and uses advanced programming techniques. The important advantage of our application is that our application just need not be installed on local computers. It can just be installed on the web server and the user just needs to upload the file onto the application from anywhere.

4.1 Data Handling:

Data collection for each subject will result in motion capture files. Then the markers in all the files are well labeled and gaps are properly filed by using VICON software. Data is then exported in an easily managed data structure (i.e. comma-delimited). These exported files contain the time-based trajectories (X, Y, Z) of each marker. The marker data files are then used in the project for further analysis.

4.2 Data Mining:

The Data Mining process has two steps. The analysis being done is of two types:

- Analysis of plots for various variables based on their trajectories.
- Biomechanical analysis involving statistical analysis of data

The problem report emphasizes on the analysis of variable trajectories.

4.3 Processing Operations:

This is the major part of analysis procedure in the project, which can be summarized by the processing operations,

- **Filtering** the raw data files.
- Processing the data files for **extracting** the variables.
- Time Series **plotting** of the selected variables.

All the three steps require various programming techniques to be utilized.

Filtering is particularly necessary for data to remove any sensor noise recorded as a part of motion capturing. A Low Pass Butterworth Filter (LPF) best caters our need. The VICON software has a batch processing technique which filters all the data files in a batch.

The Extracting step involves processing the filtered files to find out the different kind of variables recorded during the experiment and provide the user with a list of all the variables. This step can best be implemented using MATLAB tools.

A plotting routine is needed to plot the Time Series plots for each variable whenever the user wants to see the plot.

In order for the three routines to be made universal (i.e. use that for any data set, not just the Ballast study) an application is necessary for the user to be able to select / input a data set of his interest. A Graphical User Interface (GUI) is the one which best realizes the idea of universality of the above techniques.

4.4 Generalized User Functionality via GUI:

A web based application is expected to have a number of functionalities. These functionalities include,

1) Select a directory containing data files and perform a batch processing of filtering and additional processing.
2) Display a list of variables after the processing step.
3) Provide the user with the ability to select a particular variable.
4) Plot a Time Series graph for the selected variable.
5) The user should be able to select two or more variables and be able to see a combined plot of all the variables selected and compare the performance.
6) Once a file in a particular directory is filtered and processed the plots of variables extracted needs to be saved.
7) The user should be able to go back and forth through the steps involved in processing the data files.
8) The user should be able to readily print the plots.

4.5 Implementation of the GUI Functionality:

The idea of an application which is platform independent is best realized with the application installed on a remote server system and the users can access the application from where ever they want. So, a powerful server technology is needed, Windows servers are currently best of their kind. **WCF (**Windows Communication Foundation**)** is a .NET 3.5 framework which can be used to build distributed Server Client type of applications. The programming work is done in **C#**. A **Windows Service** is used for hosting the WCF. MATLAB is used for data processing steps like filtering and extracting of variables.

The particular advantages of having a distributed application instead of a desktop application is that there are no special configuration or changes needed on users PCs, lower costs, centralized data is secure and easy to backup, updates can be made quickly and easily, and access to data is always available from any networked computer. The following block diagram outlines the project.

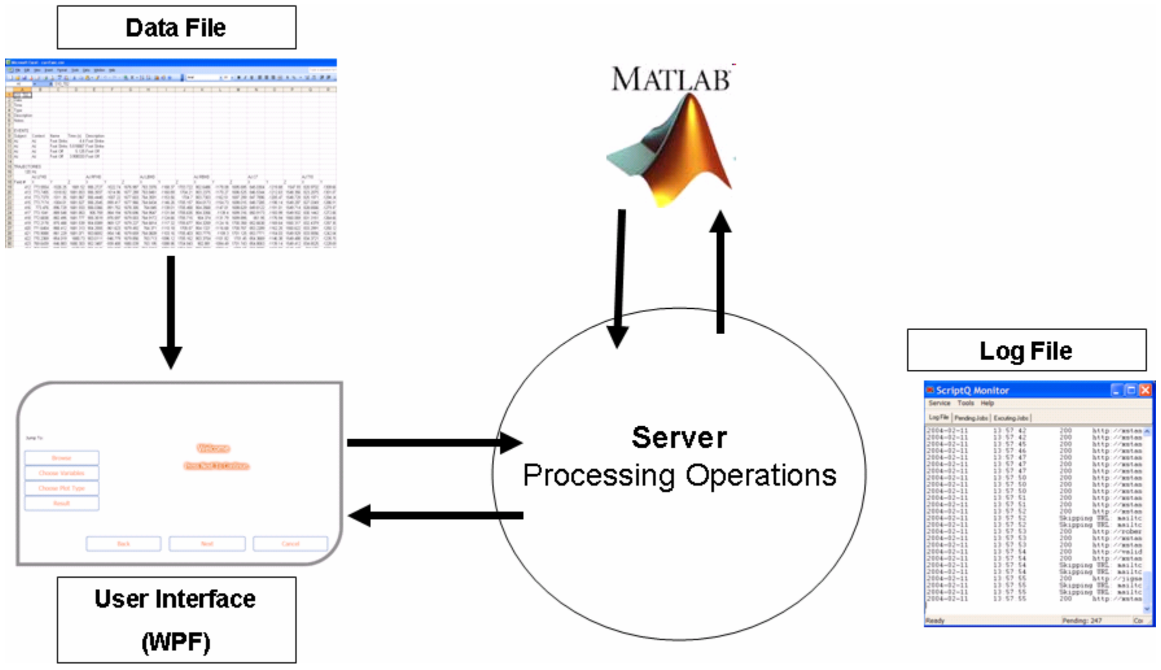


Figure 2 Project Block Diagram

Chapter 5. .NET Framework

5.1 Introduction

The .NET Framework [1] is an integral Windows component that supports building and running the next generation of applications and XML Web services.

The .NET Framework [1] has two main components: the common language runtime and the .NET Framework class library. The common language runtime [1] is the foundation of the .NET Framework. You can think [1] of the runtime as an agent that manages code at execution time, providing core services such as memory management, thread management, and remoting, while also enforcing strict type safety and other forms of code accuracy that promote security and robustness. In fact, [1] the concept of code management is a fundamental principle of the runtime. Code that targets [1] the runtime is known as managed code, while code that does not target the runtime is known as unmanaged code. The class library [1], the other main component of the .NET Framework [1], is a comprehensive, object-oriented collection of reusable types that you can use to develop applications ranging from traditional command-line or graphical user interface

(GUI) applications to applications based on the latest innovations provided by ASP.NET, such as Web Forms and XML Web services [1]. This infrastructure is depicted in Figure 3.

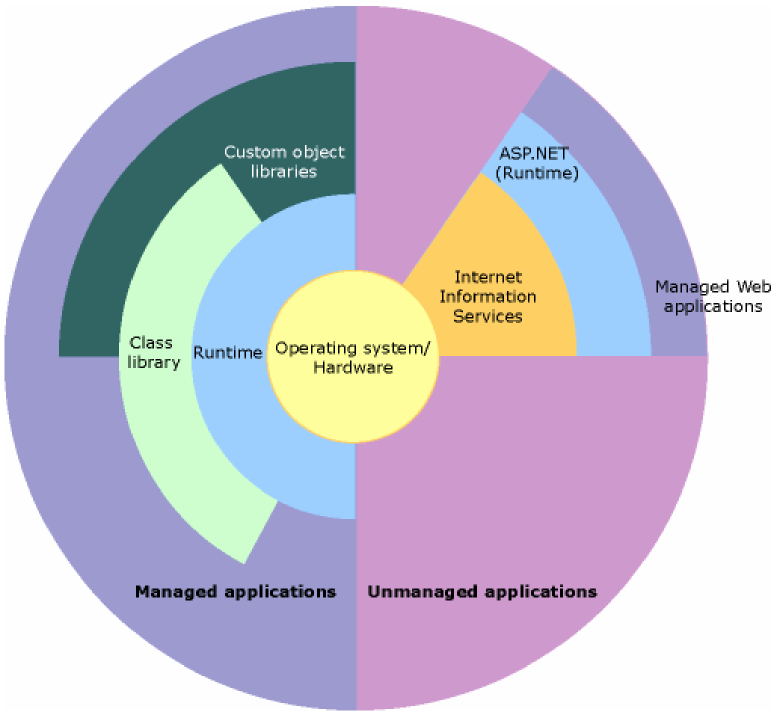


Figure 3 .NET framework

The common language runtime manages memory, thread execution, code execution, code safety verification, compilation, and other system services. These features [1] are intrinsic to the managed code that runs on the common language runtime.

The .NET Framework class library [1] is a collection of reusable types that tightly integrate with the common language runtime. The class library [1] is object-oriented, providing types from which your own managed code can derive functionality. This not only makes the .NET Framework types easy to use [1], but also reduces the time associated with learning new features of the .NET Framework.

5.2 .NET Framework Version 3.5

The .NET Framework 3.5 is the latest step in the evolution of Microsoft's flagship development platform, with each step building on what came before. This most recent release is a superset of the .NET Framework 3.0 [2], and it brings no breaking changes. Similarly, the .NET Framework 3.0 was a superset of the 2.0 release, and it also contained no breaking changes. To help make the stages in this evolution clear, Figure 4 below shows what's been added in the 3.0 and 3.5 releases.

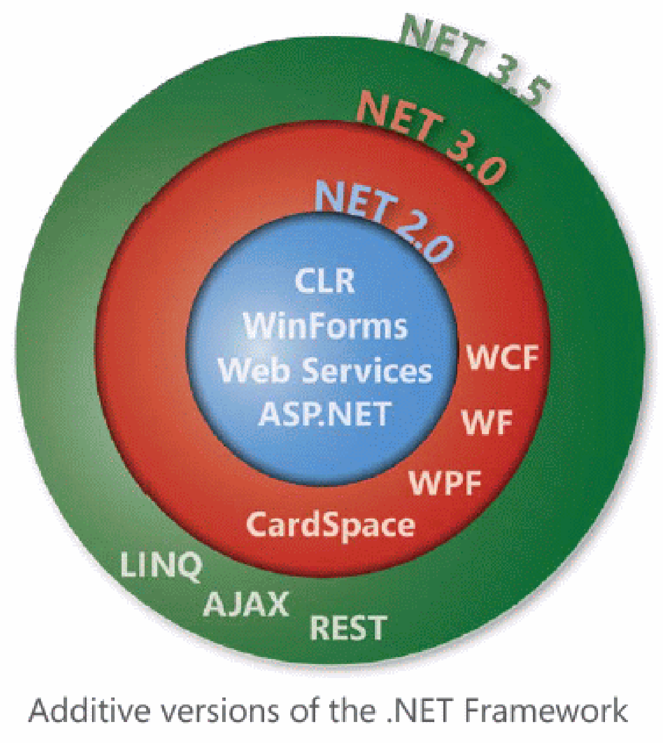


Figure 4 Additions to .NET framework

Everything in the .NET Framework depends on the Common Language Runtime (CLR). A large group of classes, known as the .NET Framework class library, is built on top of the CLR. This library has expanded with each release, as the figure shows. The .NET Framework 2.0 provided the fundamentals of a modern development environment, including the base class library, ASP.NET, ADO.NET, and much more. The .NET Framework 3.0 didn't change any of this, but it did add four important new technologies: WCF, WF, WPF, and Windows CardSpace.

5.2.1 Windows Presentation Foundation (WPF)

The goal [12] of WPF is to address the challenges of creating user interfaces for modern applications. As described next [12], WPF provides a range of capabilities to do this. A developer is free [12] to create a WPF application's interface entirely in C#, Visual Basic, or some other CLR-based language. WPF also allows specifying an interface using the XML-based XAML (Extensible Application Markup Language). XAML [3] simplifies creating a UI for the .NET Framework programming model [3]. You can create [3] visible UI elements in the declarative XAML markup, and then separate the UI definition from the run-time logic by using code-behind files, joined to the markup through partial class definitions. The ability [3] to mix code with markup in XAML is important because XML by itself is declarative, and does not really suggest a model for flow control. Elements and attributes [12] in XAML map directly to the classes and properties that WPF provides. For example, here's a simple button defined in XAML:

```
<Button Background="Red“>
 Sample Button
</Button>
```

This example creates a red button that contains the text “No”. The exact same result could also be produced with the following code:

```
Button btn = new Button();
btn.Background = Brushes.Red;
btn.Content = “No";
```

However it's defined, a WPF application can rely on *panels* for basic layout. Each panel typically contains *controls* [12], and those provided by WPF include Button, TextBox, ComboBox, Menu, and many more. How those controls [12] are positioned depends on what type of panel is chosen. A Grid, for instance [12], allows positioning controls on a specified grid, while a Canvas lets the developer place controls anywhere within its boundaries. And as usual in a GUI [12], events generated by the user are caught and handled by the various controls and other classes in the application. It's also possible [12] to apply styles and templates to groups of controls, which makes it easier for applications to have a uniform look.

WPF supports much more than these basic user interface functions, including the following:

Graphics: WPF includes support for creating two-dimensional and three-dimensional vector graphics. For 2D work [13], WPF provides standard abstractions such as shapes, brushes, and pens, while 3D graphics allowing defining a model that can be assigned lighting and camera position information. Unlike earlier technologies such as Windows Forms [13], which relies on GDI+ for graphics, WPF graphics aren't cabined off using a separate set of concepts that developers must understand. Instead, the XAML elements used for graphics can be combined naturally with those used for anything else in a user interface [12].

Images: Using XAML's Image tag [12], a WPF application can display images in various formats, including JPEG, GIF, and others.

Animation: WPF provides [12] built-in support for animating most parts of a user interface. A circle can grow and shrink [12], for example, or a button can smoothly change size. Applications [12] can also define storyboards containing timelines, allowing coordinated sequences of animations to occur.

Data Binding: Because many WPF applications display data [12], it's useful to have automatic support for mapping data to user interface elements. WPF provides [12] this kind of data binding for information contained in objects and other sources.

There are three [10] DLLs which make the Windows Presentation Foundation, that is WindowsBase (*WindowsBase.dll*), PresentationCore (*PresentationCore.dll*), and PresentationFoundation (*PresentationFoundation.dll*). This is graphically shown below.

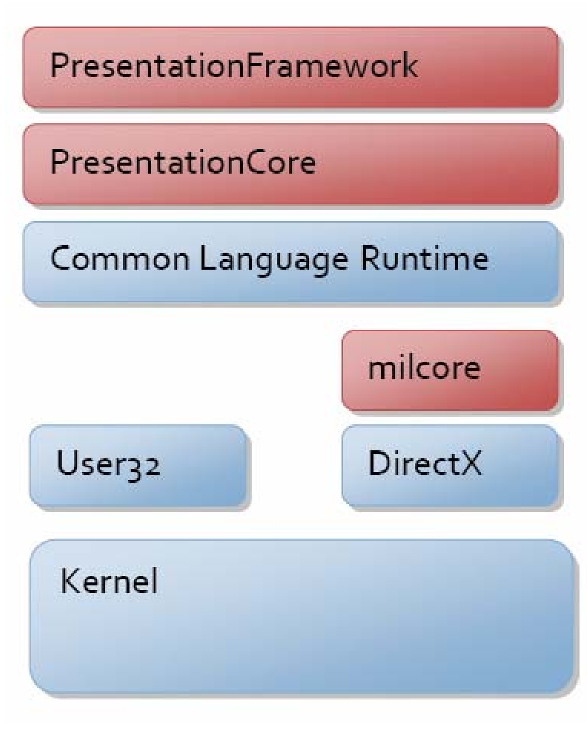


Figure 5 WPF Framework

WindowsBase defines most of the base types which you are going to use in WPF. For a WPF application, WindowsBase assembly must be included [10].

The second component is PresentationCore [10]. This DLL doesn't hold any UI component but it contains the base types which can be used in implementing the UI component (except the Window class) [10].

The third and last one [10] is the PresentationFoundation which contains all the WPF controls and other useful WPF functionality.

5.2.2 Windows Communication Foundation

The global acceptance of Web services, which includes standard protocols for application-to-application communication, has changed software development [4]. For example, the functions that Web services now provide include security, distributed transaction coordination, and reliable communication [4]. The benefits of the changes in Web services should be reflected in the tools and technologies that developers use [4].

Windows Communication Foundation (WCF) is designed to offer a manageable approach to distributed computing, broad interoperability, and direct support for service orientation [4].

WCF unifies the communication strategies. Previously, we used to have different technologies for different communication problem, so if you are doing TCP communication you will be using Sockets for example, and if you are providing online services over HTTP, you will be using Web Services, and if you are doing message queues, then you will be using MSMQ. So that was good, however it has some drawbacks.

1. We have to learn different models for every communication technique.
2. We can't change the communication technique being used in a certain application to another without massive changes in the code.

So what WCF provides is a **One Right Solution for all problems [11]**, which saves your time and effort, also it comes with a full fledged set of goodies such as Tracing, Logging which can be used regardless of the communication technique you are using.

One of WCF great benefits [11] that it provides a very handy extensible model, that you can add support for new protocols easily, or even slight changes to current protocols like adding a new encoding or encryption algorithm, or a new tracing mechanism [11]. The most elegant feature of WCF [11], that all the configuration of the services and protocol specific properties can be configured using XML config files outside the application, which is a great feature that allows administrators to be part of the game, so they can change stuff like the impersonation options, and security settings, enabling and disabling trace listeners and logging, without the need to ask developers to do it, which is somehow similar to ASP.net configuration.

WCF [4] is implemented primarily as a set of classes on top of the .NET Framework CLR. Because it extends their familiar environment [4], WCF enables developers who create object-oriented applications using the .NET Framework today to also build service-oriented applications in a familiar way [4].

The four fundamental concepts of WCF are,

1) Messages
2) Channels
3) Services
4) Behaviors

A message is a packet of data containing several pieces of important information being routed from a source to a destination. All messages are SOAP (Simple Object Access Protocol) messages, formatted in XML as SOAP envelopes.

Channels are the medium through which the messages are communicated.

WCF is made up of services and endpoints that communicate with clients, regardless of the platform. However, as you have learned, WCF builds on this to offer interoperability with other platforms and allows for the extension of WCF in many areas. A Windows Communication Foundation service is comprised of a service description and a collection of endpoints, as illustrated in Figure

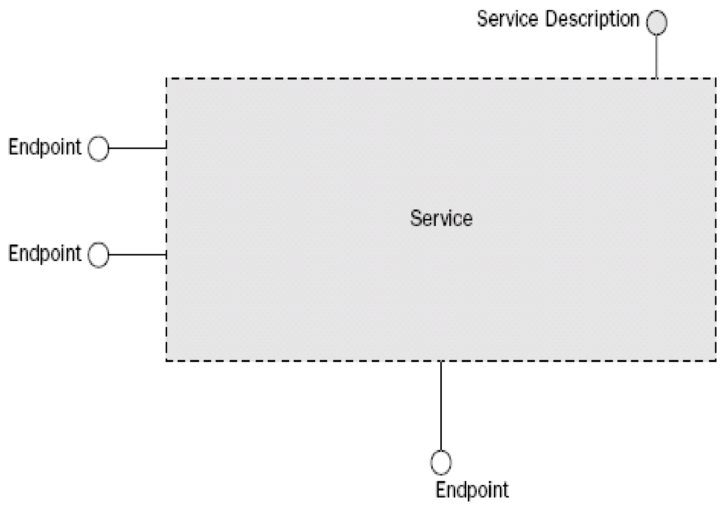


Figure 6 WCF Service

The service description is basically what broadcasts how the service can be accessed and what functionality the service provides. There are a small handful of standards in which to communicate and broadcast the service information, the most common being the WSDL (Web Service Description Language). Also, we can use an XSD Schema, a WS-

Policy, and WS-MEX (WS-Metadata Exchange). An endpoint is the component of the service that communicates
with the client and provides the service operations. Each endpoint has its very own address, which makes it indistinguishable from the other endpoints on the service.

Chapter 6. MATLAB

6.1 Introduction

MATLAB is a numerical computation and simulation tool that was developed into a commercial tool with a user friendly interface from the numerical function libraries LINPACK and EISPACK

MATLAB have functions which are similar to functions or methods or sub routines in traditional high level language. MATLAB functions have two possible parameter lists, one for input and one for output.

6.2 Plotting using MATLAB

MATLAB is very useful for making scientific, mathematical and engineering plots. It can create plots of known, analytical functions, it can plot data from other sources such as experimental measurements, it can analyze data, and then plot a comparison. The plot function takes different kinds of input and output parameters. An Example is shown below.

plot(M)

(plots the columns of M versus the
index of each value when M is a real
number.)

A sample plot generated by the above function is shown below:

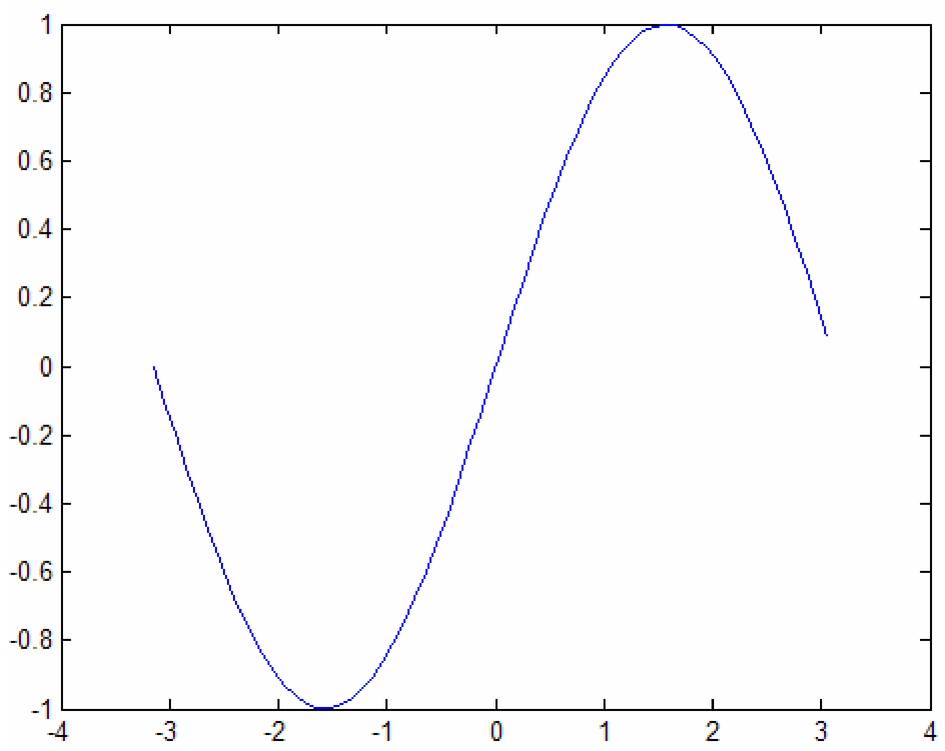


Figure 7 Sample MATLAB plot

6.3 Converting MATLAB code to .NET components

MATLAB Builder™ NE creates MATLAB® based .NET and COM components for royalty-free deployment on desktop machines, Web servers, or Web Services. As a result, users can integrate their MATLAB applications into .NET and COM applications. The builder encrypts MATLAB functions and generates either a .NET or COM wrapper around them to create component.

We can reference MATLAB based .NET and COM components as we would any other .NET assembly and COM object, for easy integration with existing and new applications. Those components which are created by the builder run against the MATLAB Compiler Runtime (MCR), the full set of shared libraries that support MATLAB. To run your .NET and COM components, you must distribute the MCR along with them. All Components created in MATLAB and the MCR can be deployed royalty free.

Together, MATLAB, MATLAB Compiler, and MATLAB Builder NE enable us to develop applications using MATLAB and then deploy them as components in .NET and COM environments. You can use the high-level, matrix-optimized language and the built-in math, and data analysis functions to rapidly prototype, develop, and test the applications. Once the applications are complete, we can use the builder to automatically package them as components and integrate them in .NET and COM applications. A sample is shown below.

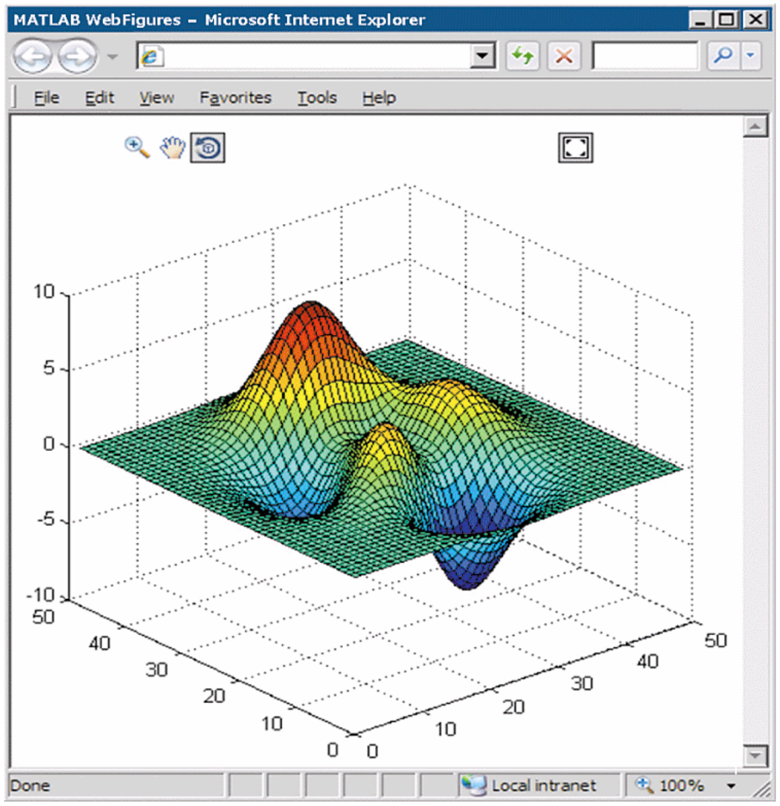


Figure 8 A .NET Web applications for visualizing a surface plot of the MATLAB peaks function.

Some of the key features include,

- MATLAB algorithms packaged into either .NET or COM components
- Automatic conversion between .NET or COM data types and MATLAB data types
- Ability to call a .NET assembly from a Common Language Specification (CLS), compliant language such as C#

Chapter 7. Implementation

The implementation of an application involves designing two major assemblies. The application from its out set looks like an n-tier architecture.

1) A Server Assembly
2) A User Interface

7.1 Server Assembly

The server assembly is the primary assembly which performs all the major tasks like extracting the variables, starting the MATLAB assembly, generating the plots and sending the plots over to the client for analysis. The server is hosted as a Windows Service on a remote machine. The communication between the server and the client is accomplished by using a WCF (Windows Communication Foundation) service.

Creating a WCF services involves configuring a service contract and creating a end point, where a method is exposed for the service users. So, each end point configured should have an address assigned to it.

Having created all the necessary services and their end points for all the methods exposed. The key end point methods (functions) are listed below

1) Input CSV Processor
2) Plotting using MatLab Assembly
3) Sending the plotted Images as byte stream
4) Log Handling
5) Service Installer

A detailed description of each of them is given below.

7.1.1 Input CSV Processor:

This function accepts a byte stream input and returns a Sorted List of variables. The byte stream should be the input CSV file that the user uploads using the client interface. The logic of the function reads through each line of the stream and intelligently extracts a list of variables and their corresponding coordinate values. A Custom Exception is thrown if the input file is not a legitimate VICON data file. *System.IO and System.Collection* assemblies are extensively used in this method for perfect extraction.

7.1.2 Plotting using MATLAB Assembly:

Once the user selects the desired variables and also the desired style of plotting, the client calls this function to get the plots. This function, when called, extracts the coordinate values of the selected variables into an array, then starts the MATLAB and runs the MATLAB assembly. Before it starts the MATLAB assembly, the function coverts the coordinate array of each variable into a MATLAB sensitive array. This is accomplished by a simple type casting. Once the MATLAB function is executed, the resulting plot images are converted to jpeg files and sent back to the client as byte streams. An ideal use of Enumerations can be observed in this method when deciding upon the plot style of the input variables.

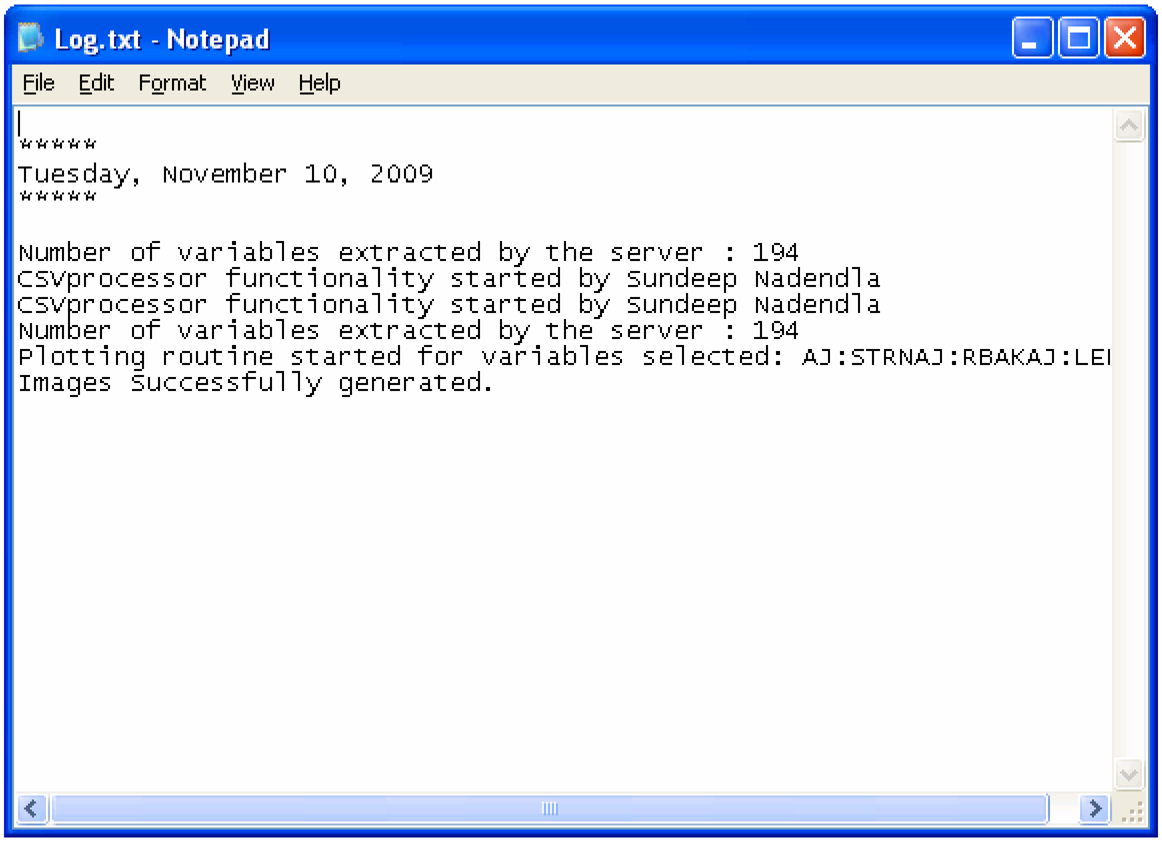
7.1.3 Sending Plotted Images as byte Stream:

Though there are so many methods to send image files over a network, I believe sending it as a byte stream is the efficient method of all. So, the plotted images are converted to byte stream and sent to client machine for the user.

7.1.4 Log Handling

This method keeps on updating the log file of the server machine. An intelligent coding is implemented here which separates the log lines based on the date when the message is written into the file. This makes the log file to be read easily and quickly.

A sample Log file content is shown below.



```
*****
Tuesday, November 10, 2009
*****

Number of variables extracted by the server : 194
CSVprocessor functionality started by Sundeep Nadendla
CSVprocessor functionality started by Sundeep Nadendla
Number of variables extracted by the server : 194
Plotting routine started for variables selected: AJ:STRNAJ:RBAKAJ:LEI
Images Successfully generated.
```

Figure 9.1 A Sample log file generated by the server based on user activity

7.1.5 Service Installer

The WCF service created needs to be hosted for the clients to use its services. There are multiple hosting options including hosting on an IIS Server, or hosting the WCF service as a Windows service. I chose the Windows Service technique. The windows implementations class is a WCF service. To qualify as a windows service, the class inherits from ServiceBase and implements onStart and On Stop methods. The host is also responsible for providing [14] a base address to the service host, which has been configured in application settings. The installer class [14], which inherits from Installer, allows the program to be installed as a Windows service by the Installutil.exe tool. Once the service is installed, the server is available for the client any time, as long as the Computer running the Service is switched on.

7.2. User Interface

Having known the capabilities of the Server assembly, the features of the User Interface are quite predictable. But, the user Interface is not just a loosely coupled UI application which does nothing apart from communicating with Server for desired information. It also implements more features. A complete list of features of the User Interface is shown below. The UI is styled to be slightly transparent for sleek look.

1) Browse through the current computer or any computer on the network to select the CSV file which needs to be processed.
2) Look at a list of all variables and select a list of variables for further processing
3) Select a type of plot
4) Select a co-ordinate index (if the plot type is Combined)
5) Look at the out put plots.
6) Save the plot(s) to a location or a folder
7) Left Panel Buttons for easy navigation.
8) Save the plots onto a word for easy printing.

Let us dive into the details.

7.2.1 Browse to select the File

As soon as the application is started, a welcome screen pops up (as shown below). The user must click Next button for next step.

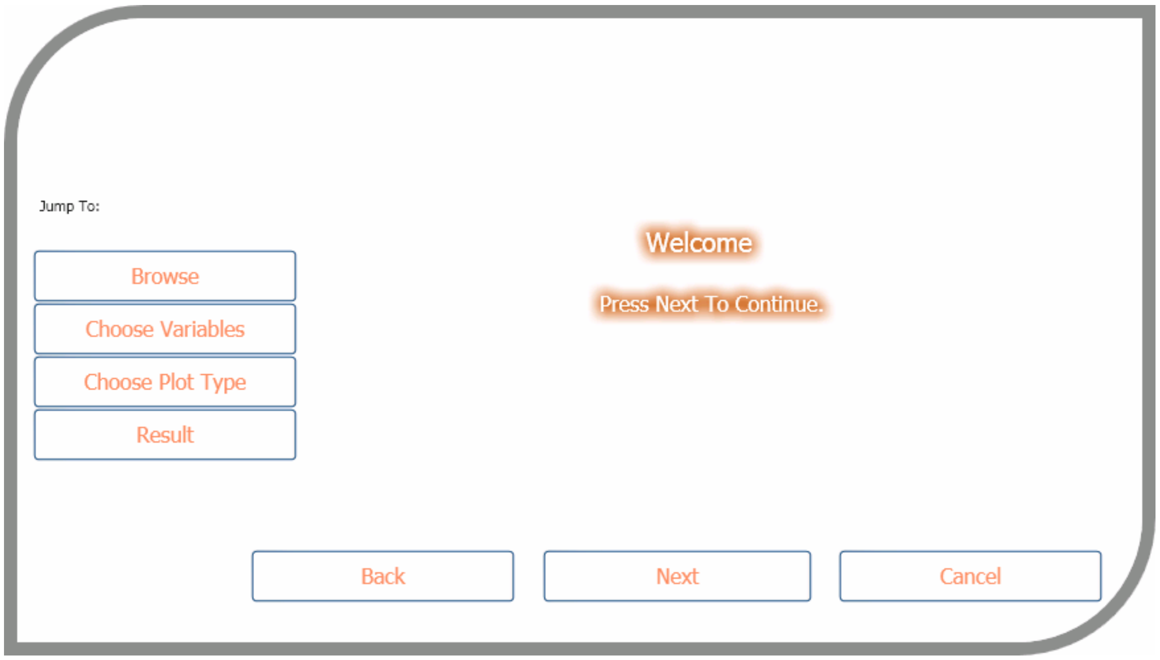


Figure 9 Welcome Screen

Once Next button is clicked, the user is prompted to browse through the computer and select the file (as shown below) and then he can click next.

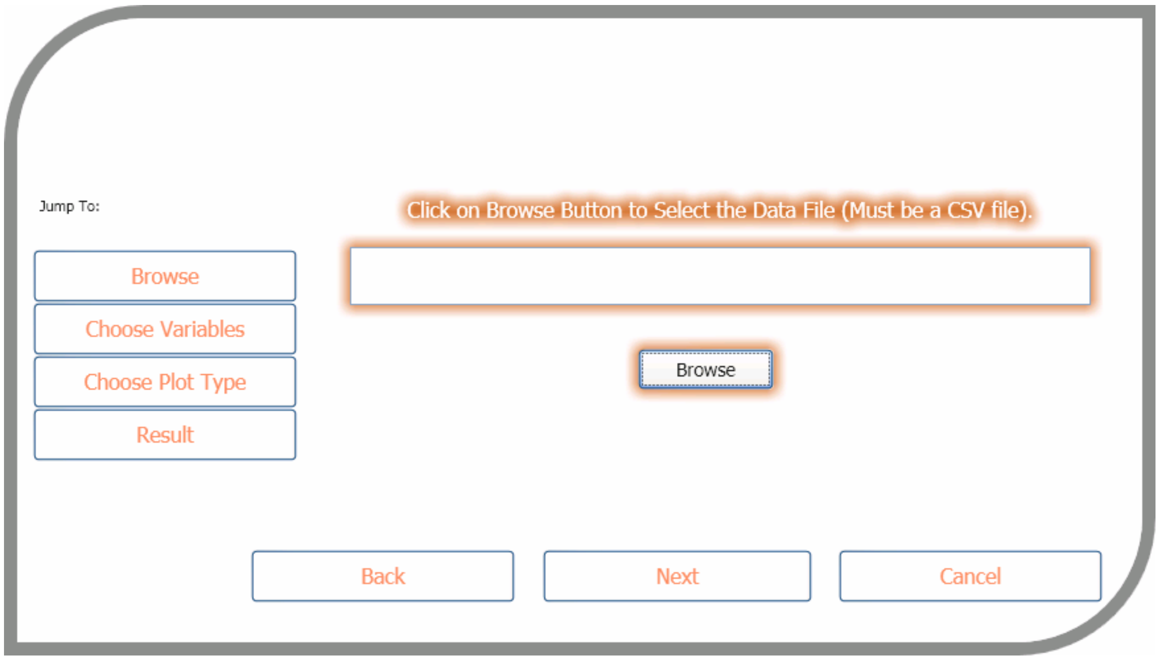


Figure 10 Browse Window

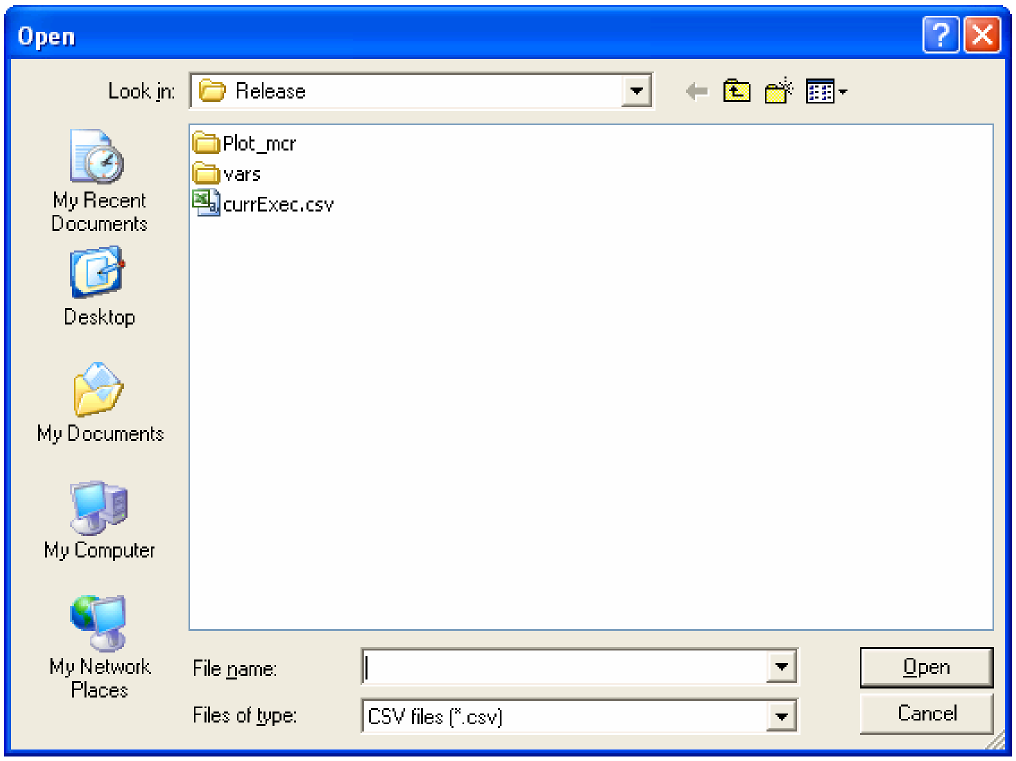


Figure 11 Browse File Dialog

7.2.2 Show variable List and enable user to select variables

The next panel shows the list of all variables extracted from the CSV file and also another empty List where we can add the selected variables which need to be further processed. A pictorial representation is shown below.

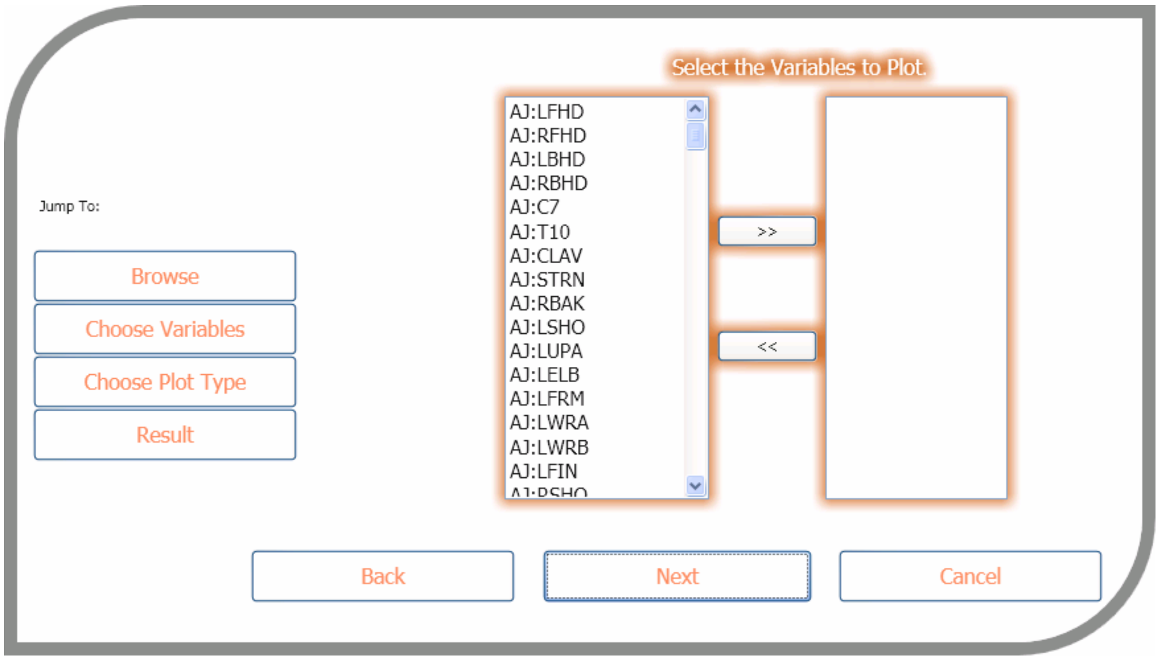


Figure 12 Variables Listed in a ListView

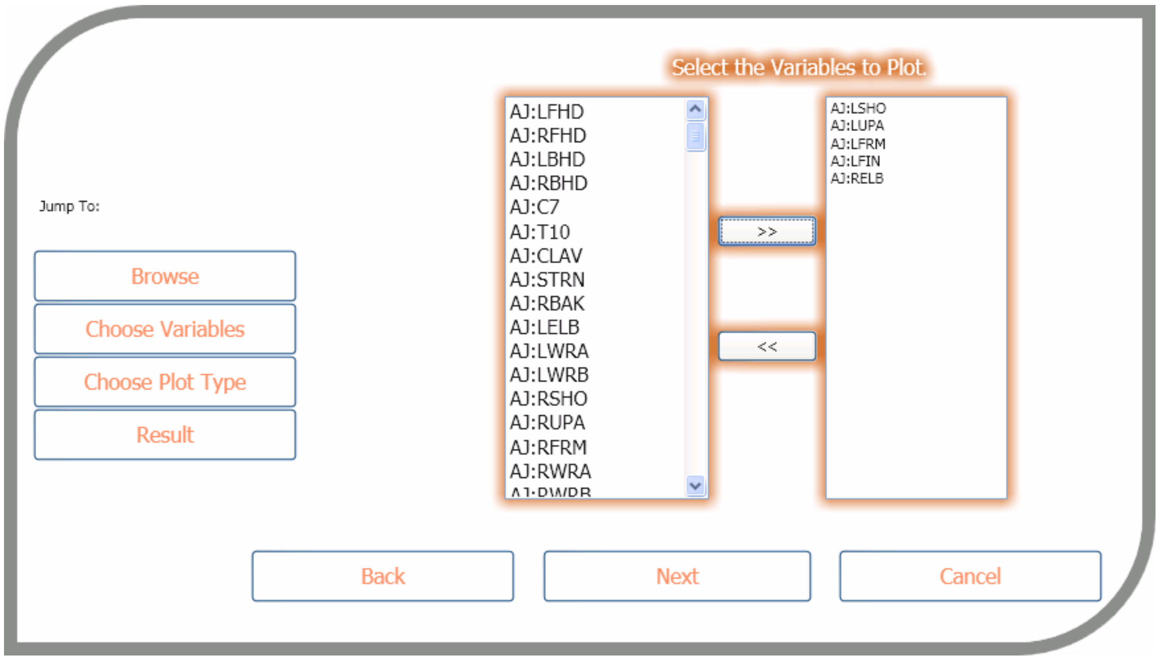


Figure 13 Selected Variables moved to Right Side ListView

<u>7.2.3 Plot Type Selection</u>

Once the user selects the variables that he wants to analyze, he is prompted to select a type of plot he wants for analyzing the variables (see images below).

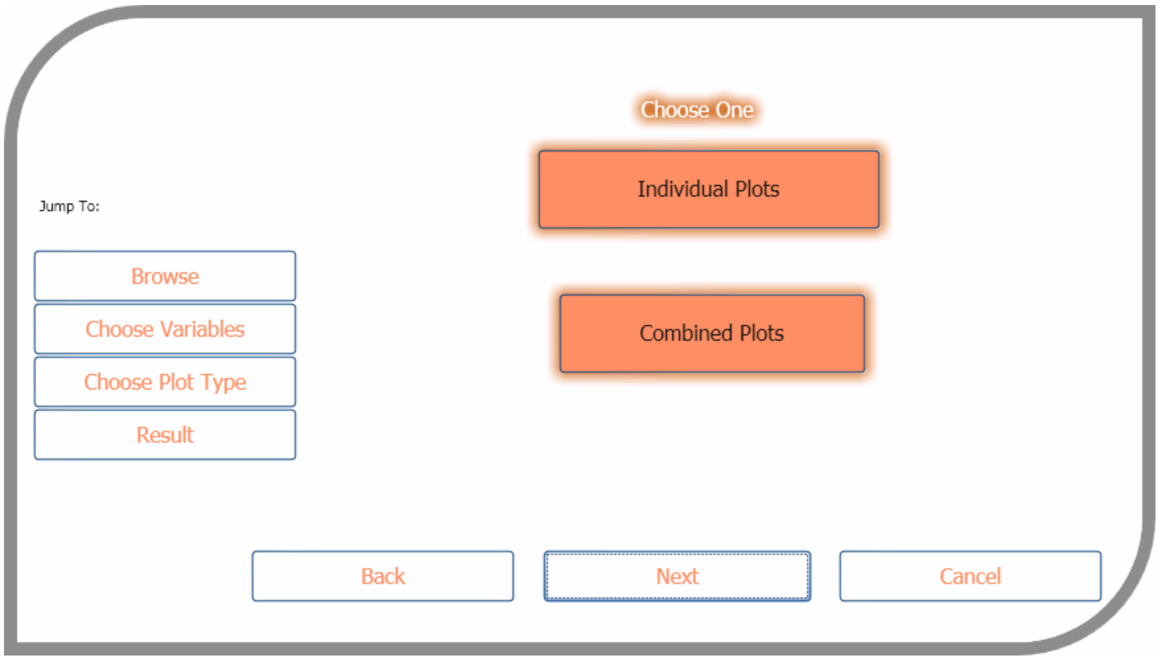


Figure 14 Select Plot Type

Individual plots generate one plot for each selected variable with three coordinates against time. While combined plot generates one plot for all the selected variables and the user will be prompted to select one of the three coordinates for plotting. As soon as a plot type is selected, it is the selected button color will be changed to White showing that the button is selected (see below).

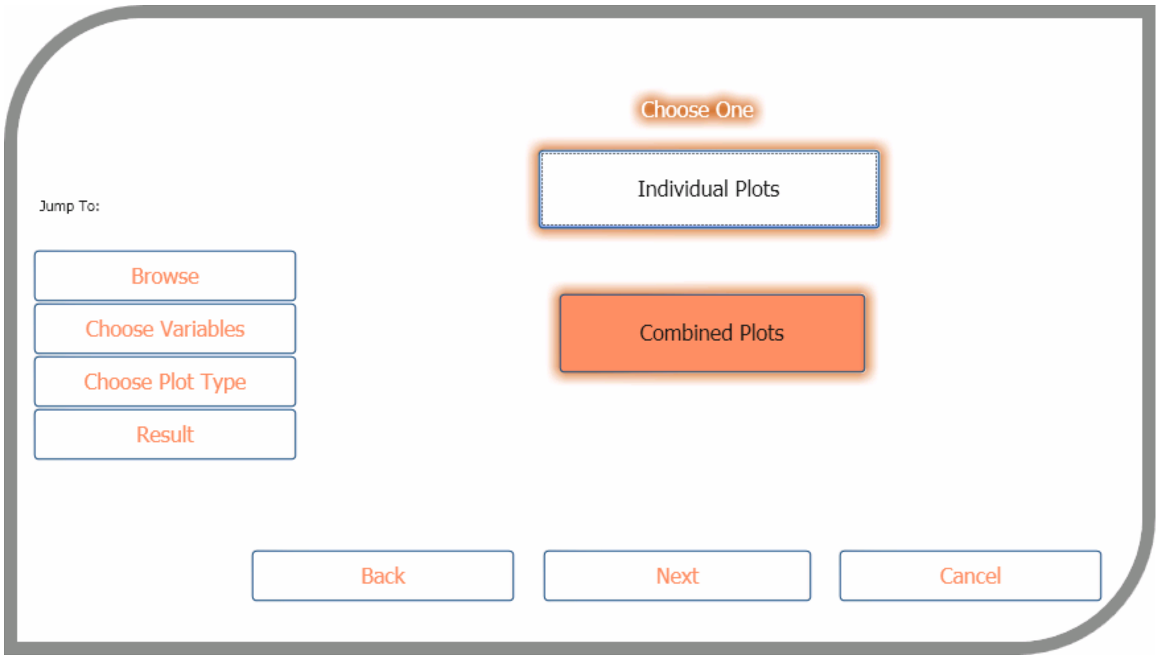


Figure 15 Select a plot type window after Individual plots is selected.

7.2.4 Coordinate type selection

If the selected plot style is combined, then the user will be prompted to select a coordinate from the given radio button list (as shown below).

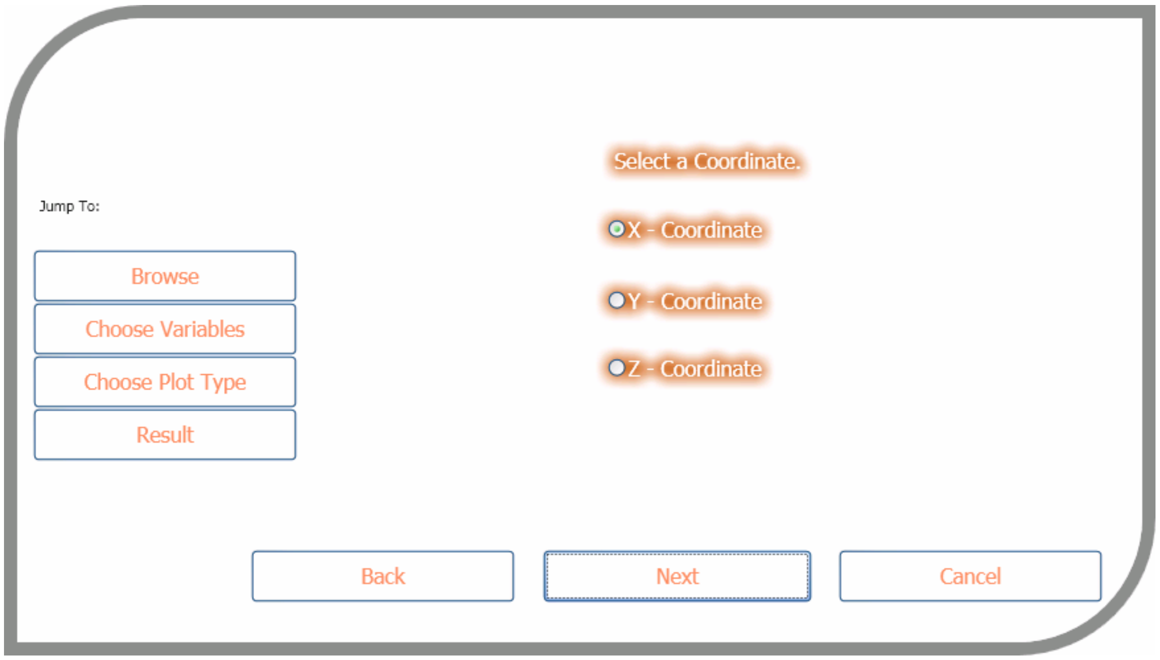


Figure 16 Coordinate selection

7.2.5 Output

Once the appropriate plot type and the coordinates (If required) are selected, the client sends a request to the server to get the plot images for a given set of variables and coordinates set. As explained previously, the server generates the plots and sends the images as byte stream to the client. The client then displays the plots accordingly (See below).

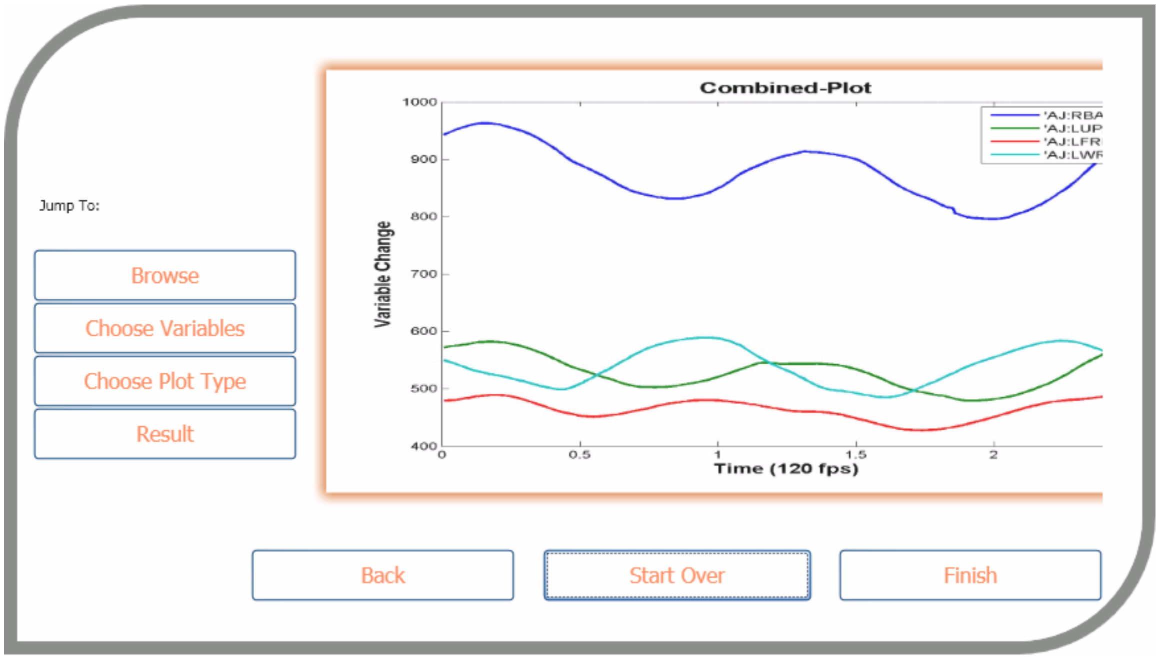


Figure 17 Output for combined plot style

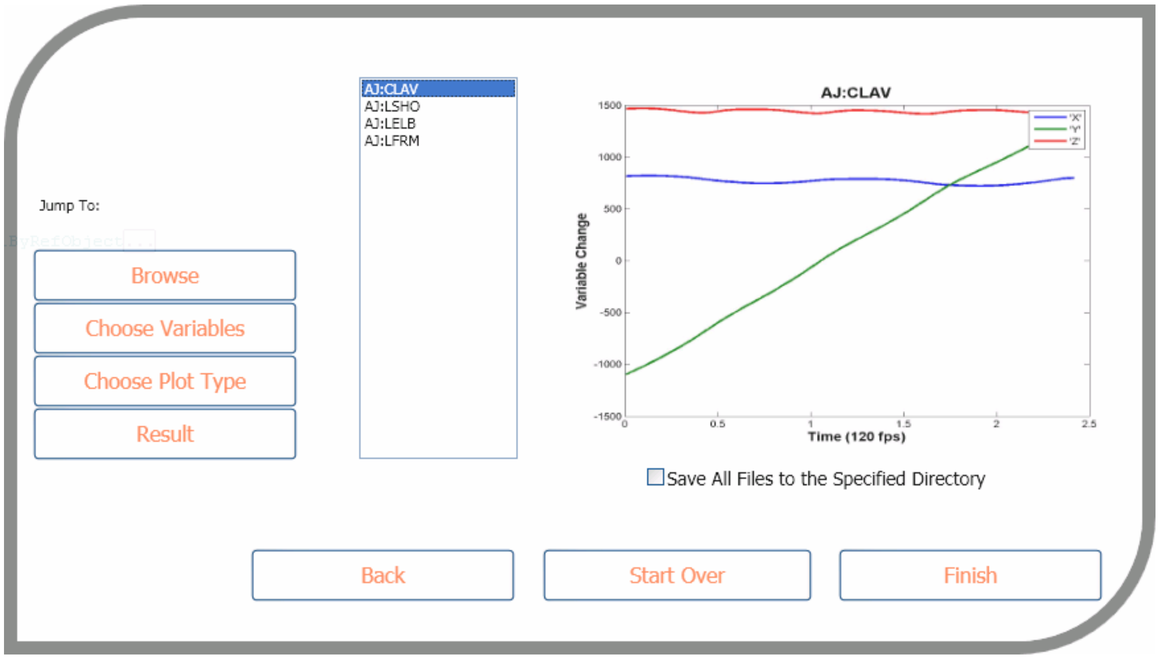
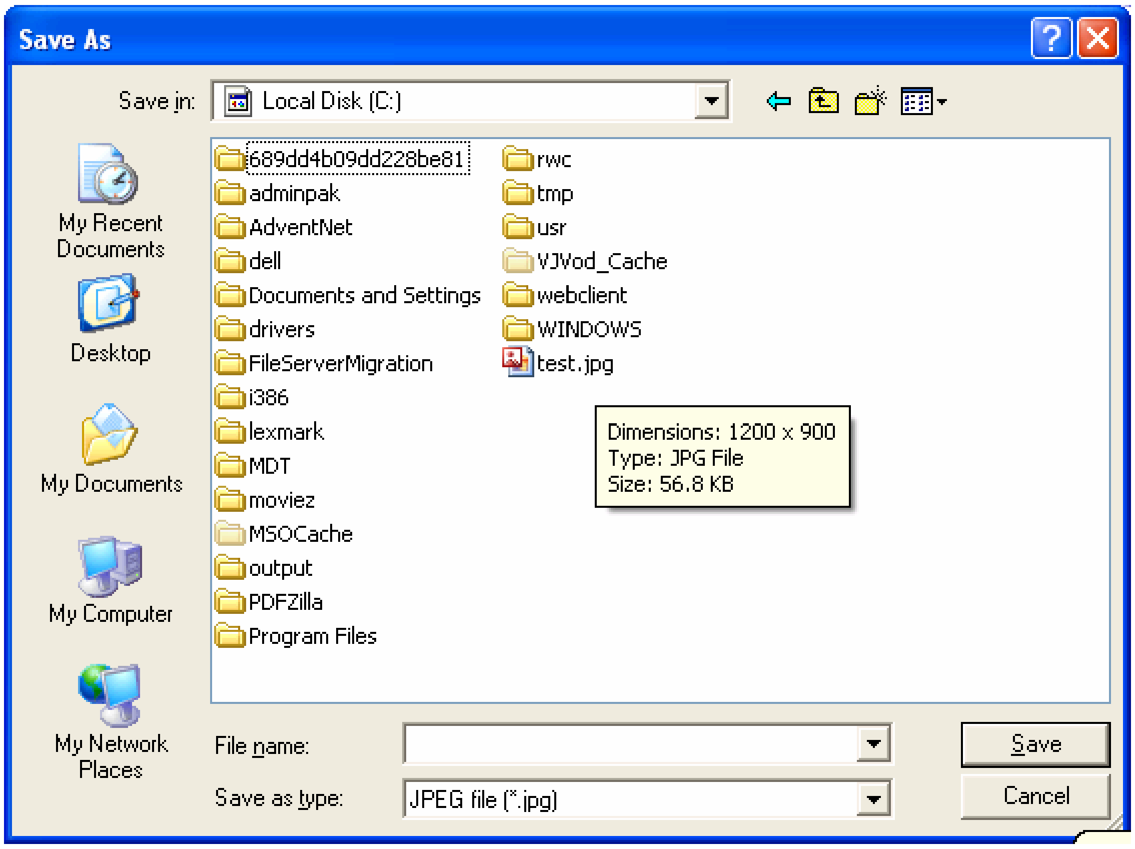


Figure 18 Output for individual plot style. (When the user selects a variable from the List, the plot for that particular variable shows up)

7.2.6 Saving the plots onto a location

To save one plot image on your computer, just click on the image and a save as dialog box pops up, using which you can save the file. In case of Individual plot style, you can skip saving each file into the computer by checking the Save All check box below the plot and then it prompts the user to select a folder where all the images needs to be saved (See below).

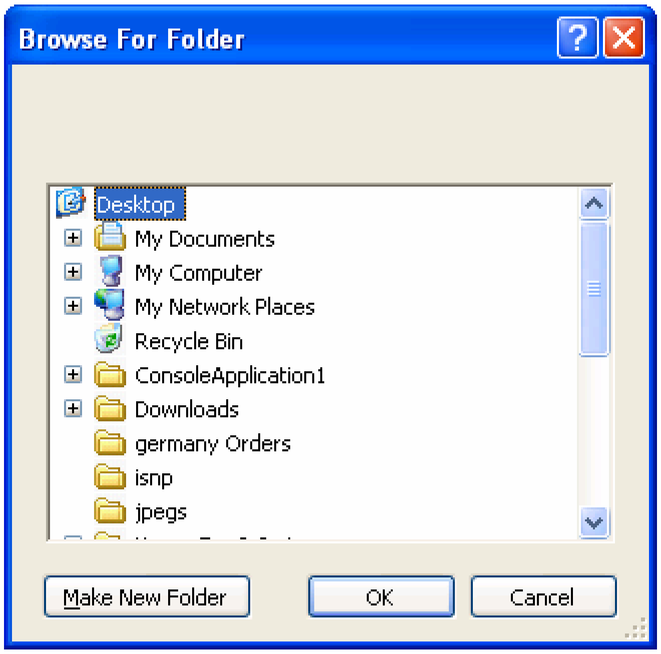


689dd4b09dd228be81
adminpak
AdventNet
dell
Documents and Settings
drivers
FileServerMigration
i386
lexmark
MDT
moviez
MSOCache
output
PDFZilla
Program Files
rwc
tmp
usr
VJVod_Cache
webclient
WINDOWS
test.jpg

Dimensions: 1200 x 900
Type: JPG File
Size: 56.8 KB

File name:
Save as type: JPEG file (*.jpg)

Figure 19 Saving each file (by clicking on the image)



Desktop
My Documents
My Computer
My Network Places
Recycle Bin
ConsoleApplication1
Downloads
germany Orders
isnp
jpegs

Figure 20 Saving all images (for individual plot types) into a folder

7.2.7 Left Panel Buttons

Apart from all the functionalities and features, there is also a left panel on the window which consists of four buttons. This panel helps the users to quickly navigate to the desired stage of the processing thus saves time in clicking the back and next buttons. For example, once the user looks at the output, he may want to change the variables. So, he can just click on the ‘Choose Variables’ button without having to going back. This saves a lot of time. These buttons are circled in the figure shown below.

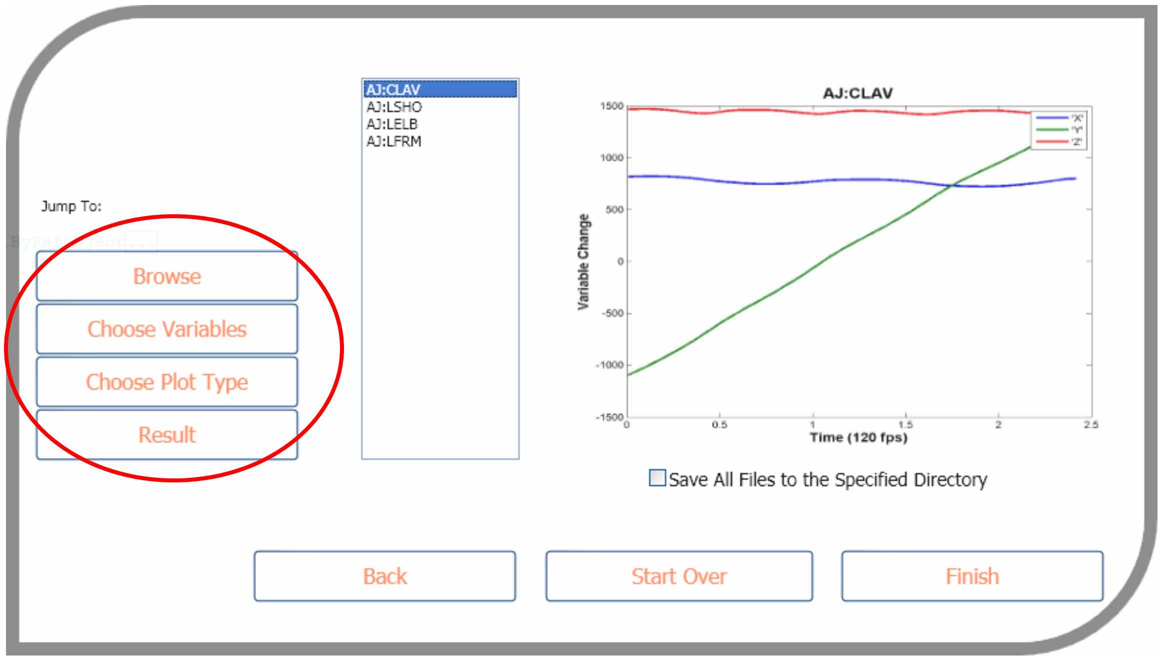


Figure 21 Left Panel for easy navigation

7.2.8 Print plots onto a Word Document for a hard copy

In case of individual plots, the might may want to see the images arranged in a word document and then can print a copy. .NET framework allows the use of Windows COM and COM+ objects in the applications that are built using the framework. Using this feature of the .NET framework, I referenced the object library of the Microsoft Word into my application and the underlying logic generates a Word document with all the plots arranged in a table form (See below). For the user to get this, he/she can just right click on an item in the variable list on the results panel and then click on the 'Print' item on the menu that pops up (See below).

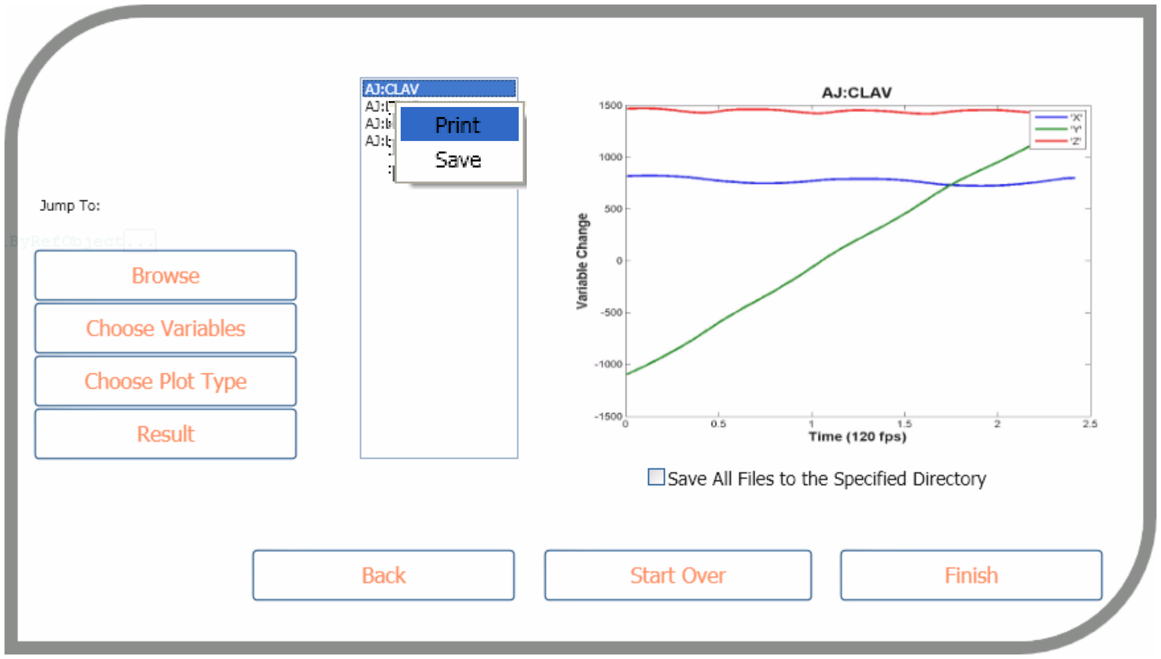


Figure 22 The Right Click Menu of a item in the variable list on the results panel showing the print menu item

.

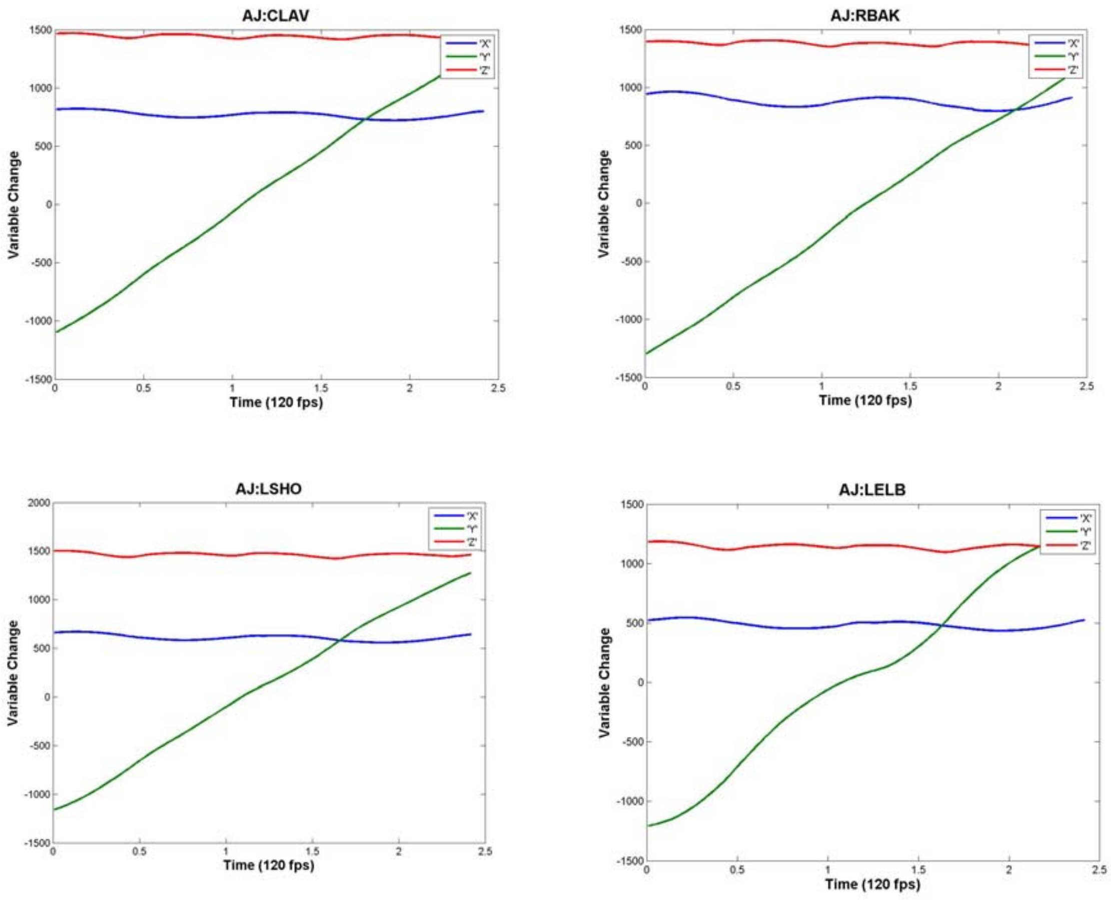


Figure 23 The above page is automatically generated as a table in a word document and the Page orientation is set to 'Landscape'

Chapter 8. Future Work:

The current implementation of the Generalized Processing Application gives the users an ability to analyze the human motion data from a remote location through the client application. The users are completely kept free from having to manually evaluate each variable and analyze individual data. Though the application provides all the necessary functionalities to the users, there are a few things that can be added to the application in the future which makes it more amicable to the users.

The best update that can be made to this application is the ability to change the names of the variables associated with the file being uploaded. The nomenclature for the variable names used the data files is a little confusing. An editable variable definition file saves some more time for the users. This functionality can be made to work in two modes. The first mode is the variable name updates can be made permanent or they can be either a particular user-based update. This can even be further drilled down to session based.

Since the application is built on a Server / Client distributed architecture, and the server works independent of the client, a wide variety of client applications can be designed to work with the server. With the possible advent of better User Interface designing technologies in the future, there is a lot of scope for UI in this application.

References

1) Conceptual Overview (.NET): http://msdn.microsoft.com/en-us/library/zw4w595w.aspx [online]

2) WCF, WF and .NET 3.5 Whitepapers by David Chappell: http://www.microsoft.com/downloads/details.aspx?FamilyID=2a8e06d9-188d-4ec8-ba2d-d3deb96fc06d&DisplayLang=en [online]

3) XAML Overview: http://msdn.microsoft.com/en-us/library/ms752059.aspx [online]

4) What is Windows Communication Foundation: http://msdn.microsoft.com/en-us/library/ms731082.aspx [online]

5) Professional WCF Programming .NET Development with the Windows® Communication Foundation by Scott Klein

6) INTRODUCTION TO MATLAB® & SIMULINK A Project Approach – Third Edition **O. BEUCHER and M. WEEKS**

7) Plotting with MATLAB: http://web.cecs.pdx.edu/~gerry/MATLAB/plotting/plotting.html [online]

8) Deploy MATLAB code as .NET and COM components: http://www.mathworks.com/products/netbuilder/ [online]

9) Working with MATLAB Builder NE: http://www.mathworks.in/products/netbuilder/description2.html [online]

10) WPF Framework Overview: http://www.codeproject.com/KB/WPF/WPFTutorials.aspx [online]

11) Service Oriented Architecture via Windows Communication Foundation: http://blogs.msdn.com/bashmohandes/archive/2007/09/17/soa-via-wcf-part-2.aspx

12) Introducing the .NET Framework 3.0: http://msdn.microsoft.com/en-us/library/aa479861.aspx [online]

13) Introducing .NET framework 3.5: http://download.microsoft.com/download/f/3/2/f32ff4c6-174f-4a2f-a58f-ed28437d7b1e/Introducing_NET_Framework_35_v1.doc [online]

14) Host a WCF Service in a Managed Windows http://msdn.microsoft.com/en-us/library/ms733069.aspx [online]

15) http://www.mathworks.com/access/helpdesk/help/techdoc/index.html?/access/helpdesk/help/techdoc/ref/plot.html&http://www.google.com/search?q=MATLAB+plot+&ie=utf-8&oe=utf-8&aq=t&rls=org.mozilla:en-US:official&client=firefox-a