Qt widget to manipulate plots for Octave Users

Ranasinghe Y.R.M.
Department of Computer Science & Engineering, University of Moratuwa
Sri Lanka
yrmranasinghe.10@cse.mrt.ac.lk

***Abstract*— Octave [1] is an open source mathematical analyzing & computing tool, similar to Matlab. This paper introduces a graphical interface to make plot manipulation in octave in more user-friendly way. Using the proposed widget, users can generate plots using a GUI, instead of using command line interface enhancing the real time interaction in Octave plotter module [2]. The widget is implemented using Qt user interface development framework and facilitates user with all "plot" functionalities provided by Octave command line interface. Further, the widget support to change the plots, there color, labels, etc. interactively. The module is developed adhering to Rational Unified Process, with 100% test coverage.**

***Index Terms*—Octave Plotting GUI, manipulate octave plots, Qt widget**

I. Introduction

A. Overview

Octave is a mathematical analyzing & computing tool, similar to Matlab, but it is an open source project. At initial stages of the project, user interaction happens via command line interface. Now the software is gradually introducing a graphical user interface to make it more user-friendly, and to make it popular among user community. The Qt UI framework is used to build GUI for Octave.

The graph plotting tool of Octave is very important feature for the software. But it still does not have dedicate, compatible Qt widget for user interaction when drawing graphs. Attempts to use external module for graph plotting tools such as FLTK [4] are there, yet does not follow original octave look and feel. Also it is hard to extend further as it is not fully compatible with Octave and other Octave GUIs implemented using Octave. So the requirement of a Qt widget for graph plotting was pertinent and it was included as a Google Summer of Code project this year [2].

So this project targets to provide a friendly interface for graph plotting for general Octave users and a compatible & extendable module for Octave developers.

B. Functionalities

1) Plot the Octave graphs using Qt

This Qt widget plots graphs communicating with Octave core, sending user data and retrieving the generated graphic object. The basic "plot" function [6] and other functions such as Fplot, ezplot, ezcontour [6].

2) Interactive plotting

The user can interact with the provided GUI to create various types of graphs from same data set, to select variables to plot directly from a workspace browser, to easily create and manipulate subplots in the figure and to set properties on graphics objects such as,

Axis and variables in each axis
The graph line width and color
The size of data set used
Scale of plot
Color codes used
Graph type (bar chat, pie chart, etc.)
The plot legend format & position

The graph changes dynamically when user applies the changes.

C. Quality attributes of the system

The usability of the widget is given high priority. It is designed in the way to comply with overall look & feel of Octave GUI and required training time is minimal.

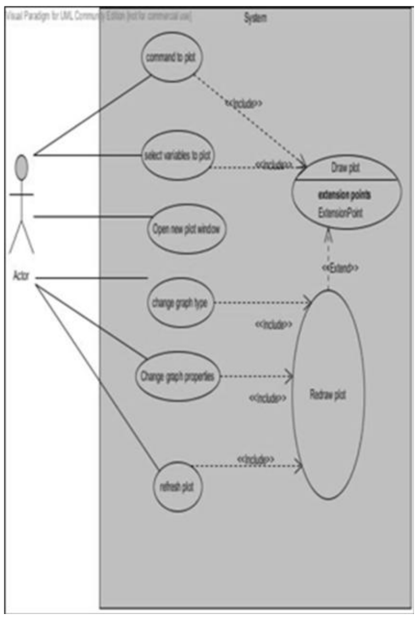

Fig. 1 The Use Case Diagram

The code quality, reusability and extendibility were also taken in to consideration.

The performance of the module was optimized very much, because this is real time application which is performing computationally demanding task. User requires fast response and high speed graphic processing.

D. Technology

The widget is developed using Qt framework, programmed using C++. Integrated development environment used was Eclipse.

“GNU Octave is a high-level language, primarily intended for numerical computations. It provides a convenient command line interface for solving linear and nonlinear problems numerically, and for performing other numerical experiments using a language that is mostly compatible with Matlab. It may also be used as a batch-oriented language.

Octave has extensive tools for solving common numerical linear algebra problems, finding the roots of nonlinear equations, integrating ordinary functions, manipulating polynomials, and integrating ordinary differential and differential-algebraic equations. It is easily extensible and customizable via user-defined functions written in Octave's own language, or using dynamically loaded modules written in C++, C, Fortran, or other languages.”

The Octave community emphasizes the requirement of this widget in following manner [2]:

“Octave has had for some time a native OpenGL plotter. The plotter requires some user interaction for manipulating the plots, and it's been using FLTK for quite some time. We want to replace this with Qt, so it fits better with the overall GUI look-and-feel and is easier to extend in the future.

QtHandles is a current work in progress integrating the octave OpenGL renderer plus good support for GUI elements (uicontrol, uimenu, uitoolbar...). This project may initially consist of integrating the existing QtHandles code base into Octave. Then if time permits, further improvements can be made to QtHandles."

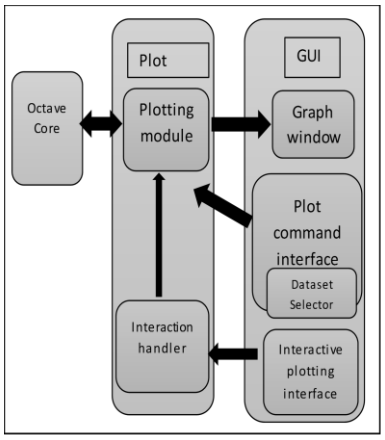

Fig. 2 The Logic View of system

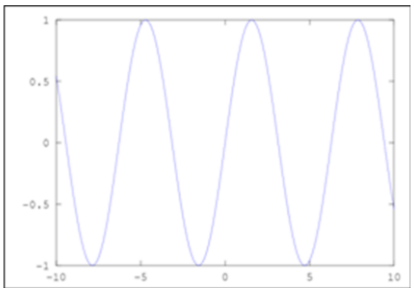

Fig. 3 plot(x, sin(x))

The “plot” function of Octave is used as the base for developing the plotting module.

The plot [5] function allows you to create simple x-y plots with linear axes. Consider following commands.

```
x = -10:0.1:10;
plot (x, sin (x));
```

These commands display a sine wave shown in figure 3. On most CLI based Octave systems, this command will open a separate plot window to display the graph.

The possible combinations of arguments are:

Function File: **plot** (*y*)
Function File: **plot** (*x*, *y*)
Function File: **plot** (*x*, *y*, *property*, *value*,..)

Function File: **plot** (*x*, *y*, *fmt*)
Function File: **plot** (*h*, ...)
Function File: *h* = **plot** (...)

The display settings of the graph can be set by passing other parameters. For example

```
plot (x, y, "@12", x, y2, x, y3, "4", x,
y4, "+")
```

will plot y with points of type 2 (displayed as ‘+’) and color 1 (red), y2 with lines, y3 with lines of color 4 (magenta) and y4 with points displayed as ‘+’.

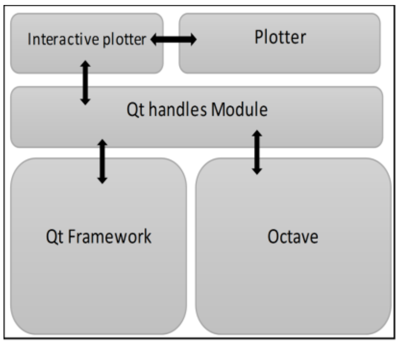

Fig. 4 the Implementation View

II. Design and Implementation

The Fig. 2 illustrates the layered architecture design on which the system components and system interactions are based.

There are separate GUI components to plot graphs including the "dataset selector", the graph is shown in another GUI component and it can be changed dynamically by interactive plotting component. The GUI communicates with the plot module. The 'data set selector' and interactive plotting component are implemented in this project. The widget communicates with the Octave Core as two separate threads communicating through standard input and output of Octave core.

Fig. 3 gives the implementation perspective of the system, its dependencies and technologies used.

The *Qt handles module* is the central communicator between Octave, Qt framework and the *plotter* and *interactive plotter* modules. The *plotter* and *interactive plotter* modules are developed from scratch. The *Qt handles module* [3] is available in the Octave project for the developers to use, but it is complex and hard to use as a whole and was not properly documented. So the separate *Qt Handles module* was developed reusing some components from original *Qt Handler*.

III. Results

The Qt widget is successfully completed with the full test coverage, unit tests written in C++. The basic "plot" function [5] is implemented for demonstration purposes and other functions are simply extendable from it.

The quality requirements such as usability, maintainability are also fulfilled. There is unexpected performance issue when drawing the graphs. The reason behind this is the overhead of communication between two threads, the octave core and the plotter module.

IV. Conclusion and Further development

The graphical user interface, though it has slight efficiency limitation, is much user friendly and will help Octave to reach a larger user community who are not comfortable with the command line interface. It is very important and useful because the other alternatives for Octave are not freely available. Octave is free and open source, so it should be further developed in to a level that user can customize it in to his/her own preference.

There are many avenues to be extended further on this project. The performance issue raised through communication among threads can be reduced by integrating the module directly to Octave core. Also other advanced plot functions should be implemented on the base build. The similar module can be built for Octave 3D plotting as well.

Acknowledgment

This work was guided and supported by academic staff, the Department of Computer Science & Engineering, University of Moratuwa.

References

[1] John.W.Eaton. *About GNU Octave* [Online]. Available: http://www.gnu.org/software/octave/about.html

[2] *Summer of Code Project Ideas* [Online]. Available: http://wiki.octave.org/Summer_of_Code_Project_Ideas#Implement_a_Qt_widget_for_manipulating_plots

[3] *QtHandles.*[Online] *Available:* https://github.com/goffioul/QtHandles

[4] Muthiah Annamalai. *Octave-FLTK project: Octave interface to FLTK library*. [Online] Available*:* http://octave-gtk.sourceforge.net/octave-fltk/

[5] *Octave Two-dimensional Function Plotting* [Online] Available: http://www.gnu.org/software/octave/doc/interpreter/Two_002ddimensional-Function-Plotting.html