International Journal of Engineering Research and Development
ISSN: 2278-067X, Volume 1, Issue 7 (June 2012), PP.63-66
www.ijerd.com

Dynamic Graph Plotting with WPF

Pranali Dhete, Shirish Patil, Sameer Soni, Dr. B.B. Meshram, Mr. Manoj Kulkarni
Computer Department, VJTI Matunga, Mumbai-400019, India
Prime technologies, Andheri East, Mumbai

Abstract—*Windows Presentation Foundation (WPF) is widely used by developers and graphics designer to create modern user experiences without having to learn various technologies. Prior to WPF it was tiresome job for developers, who wanted to use 3D, video, speech and rich document viewing in addition to normal 2D graphics , to learn the several technologies and blend them together without much built in support. In this paper, with the help of WPF and the library Dynamic Data Display developed by Microsoft, we tried to generate graphics(chart plotting) dynamically which is not supported by existing WPF Feature.*

Keywords— *WPF, Dynamic Data Display, Chart Plotting, run time graph generation, XAML*

I. INTRODUCTION

Windows Presentation Foundation (WPF) developed by Microsoft is a computer-software graphical subsystem for rendering user interface in Windows-based applications. With various features like Direct3D, Data Binding, Media Services, Templates, Animations, Imaging, Effects it has become very easy to develop complex GUI with the simplest way. In this paper we are going to discuss graphics part of WPF. Microsoft has developed a library called 'Dynamic Data Display' which is used with WPF to generate various informative charts. Dynamic Data Display is a set of Silverlight controls for adding interactive visualization of dynamic data to Silverlight application. It can also be used with WPF application. It allows to create line graphs, bubble charts, heat maps and other complex 2D plots which are very common in scientific software. So, using these controls we tried to plot graphs dynamically.

II. WPF

Windows Presentation foundation (WPF) is a new graphical subsystem for rendering and display of Microsoft Windows applications [5]. It builds upon DirectX for drawing and rendering content which gives developers and designers lots of tools to create graphically pleasing user interface. WPF

introduces a common programming model and clearly separates presentation from logic. WPF provides graphics services (2D, 3D, vector graphics) which are rendered on the graphics card GPU leaving minimal impact to the CPU, provide powerful data binding mechanism, media services, layout engine so that the application can be easily resized and/or displayed on various screen sizes and does not use fixed point GUI widget placing as was the case before WPF,templates that are used to redefine how a GUI element looks (control template) or how data should be displayed (data template), animations, better text services, and so on. Thus,WPF is an extensive library full of features for creating very expressive GUI's and is used in .NET development environment. Creation of GUI in WPF is done using XAML[6], which is XML-like declarative language, where

GUI is created using XAML declarations and code logic behind GUI elements is created using one of the .NET languages such as C#. Sample XAML snippet can be seen in

Fig 1 below. WPF however does not force a developer to use XAML and all GUI can be created from code if one chooses to do so. However with XAML application designers can contribute to the software development lifecycle where their GUI design can be seamlessly and immediately used in the actual application developers do not need to develop a GUI per design as it was customary prior to WPF. Thus, software can be developed in parallel - designers develop GUI while developers create code behind such as business logic and data management modules.

III. FEATURES OF DYNAMIC DATA DISPLAY

Dynamic Data Display library provides the way by which we can plot the graph in WPF application. It's Microsoft Research group development. It includes various controls like chartplotter, cursorcoordinategraph, segment, line which are very useful in drawing charts.

This library can be downloaded from Microsoft's website and added in the reference of project solution.

Following are the steps for plotting the graphs using Dynamic Data Display.

1) Add Dynamic Data Display library to references of project.
2) Build the project.
3) Add namespace in the XAML file of project as shown below-

```
“xmlns:d3="http://research.microsoft.com/DynamicDataDisplay/1.0".
```

4) Add Chartplotter control to the “grid” of view.

```
<d3:ChartPlotter Name="plotter"/>
```

5) Add points to datasources X and Y.

```
dataSource = new EnumerableDataSource<Point>(dataCollection);
dataSource.SetXMapping(x => x.X);
dataSource.SetYMapping(y => y.Y);
```

6) Add this datasources to plotter by using the function AddLineGraph().

```
plotter.AddLineGraph(dataSource,new Pen(Brushes.Blue, 2),new CirclePointMarker { Size = 10, Fill = Brushes.Red
},new PenDescription("Data"));
```

7) And to see the points on the graph, use the function FitToView().

```
plotter.FitToView();
```

IV. DISADVANTAGE

Though Dynamic Data display provides powerful controls to plot the graphs in WPF, there are some disadvantages with this which are as follows.

A. Static plotting of points

In the current system we have to provide the list of points to plotter to plot the points. Thus restricting the friendly user interaction with graph plotting.

B. Cannot clear the points on the graph at run time

The points added on the graph cannot be deleted at runtime. To clear the specific point we have to delete it from the list. This is overhead as whenever we want to delete some point we have to delete it from list.

V. DEVELOPED SYSTEM

To overcome the above mentioned disadvantages, we have developed the program by which we can plot the points on graph at runtime. The initial steps to include Dynamic Data Display library to project are same as mentioned in the sectionII. Once the control ‘Chartplotter’ is added to xaml and project is builded, a blank plotter will be displayed in view as shown in below Fig. 1.

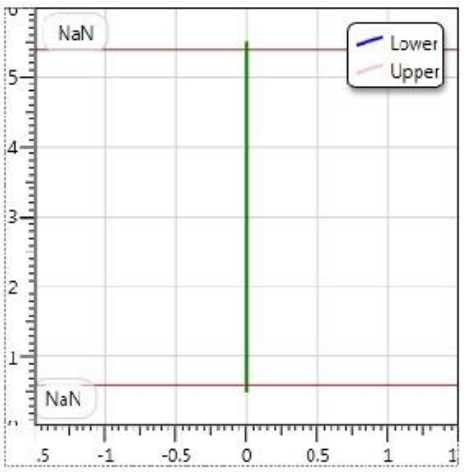


Fig. 1. ChartPlotter with no pints

The procedure to plot the points at runtime is explained with the following steps-

1) The first step is to capture the point on plotter by '*GotMouseCapture*' event generated by mouse click on plotter. The event added to xaml is shown below.

```
<d3:ChartPlotter Name="plotter" Grid.ColumnSpan="3" Mouse.GotMouseCapture="plotter_GotMouseCapture"/>
```

2) The code generated in code-behind file is as follow.

```
private void leadplotter_GotMouseCapture(object sender, MouseEventArgs e){}
```

3) Now the main task is to convert the mouse position which is obtained in parameter 'e' of the above method according to plotter's X-Y coordination. The logic behind this conversion is explained as below.

```
Point mousePosition = e.GetPosition(plotter);
var transform = plotter.Viewport.Transform;
Point mousePosInData = mousePosition.ScreenToData(transform);

double height = plotter.ActualHeight;

mousePosInData.X = mousePosInData.X - (height / 2000);
```

The mouse position is captured by '*e.GetPosition(plotter)*' method. This point is collected in mousePosition. Now the point held by mousePosition is in the screen resolution format i.e. it contains screen's co-ordinated where the plotter is situated on the screen. To get the actual plotter co-ordinate the function '*ScreenToData*' is used. To this function '*transform*' variable is passed which is obtained by plotter's viewport's '*Transform*' property. So the line "mousePosition.ScreenToData(transform)" will actually transform the screen coordinate to plotter's coordinate. This actual coordinate is collected in point *mousePosInData.* But the x-coordinate in *mousePosInData* is not the actual x-coordinate. To get the actual x-coordinate we have to first calculate plotter's height and then divide the height by constant. Sibtract this division from mousePosInData. The constant in our example is 2000. This constant value can be determined by the ticks given to the y-axix.

4) To show the points on the plotter following function is used.
```
dataSource.RaiseDataChanged();
```

5) Also the ticks of the axis can be adjusted by using following function
```
((NumericTicksProvider)((NumericAxis)plotter.HorizontalAxis).TicksProvider).MinStep = 0.5;
```

6) The output of the code is shown in the following Fig. 2.

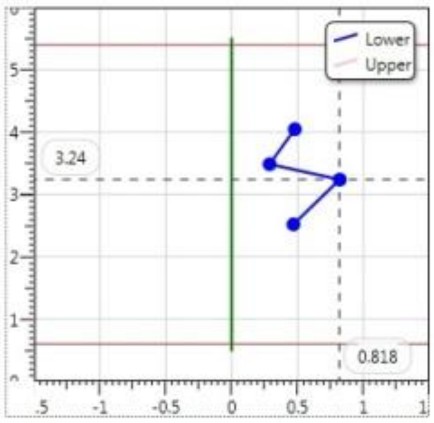
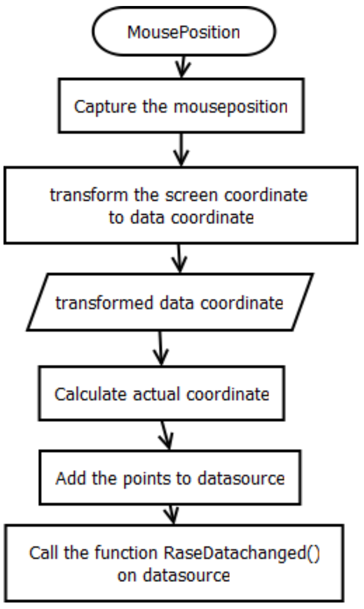


Fig. 2. ChartPlotter with points plotted at run time

The points are plotted at run time in above graph. Also the x and y coordinate are shown in rectangular boxes. This tooltip is displayed with the cursorcoordinategraph control. Thus by capturing the mouse position we can plot the points dynamically. The steps can be summarized in the following flowchart.


MousePosition
Capture the mouseposition
transform the screen coordinate to data coordinate
transformed data coordinate
Calculate actual coordinate
Add the points to datasource
Call the function RaseDatachanged() on datasource

Thus the two disadvantages which we came across are overcome with the help of above lines of codes. We can dynamically plot the points on graph and can also clear the points at run time. To clear the points we just have to maintain the list of all the points and in the clear command we have to clear that list.

Similarly, multiple graphs can be plotted on the same plotter by adding multiple datasources. Also we can restrict the x and y axes by using restriction method in the viewport pf plotter.

VI. FUTURE WORK

The above coding is done in code-behind. This can also be done using MVVM architecture. For this, instead of using code-behind file, separate .cs file should be used. Following the MVVM patter, the property binding should be done to properties of chartplotter. We are trying to access these control properties in the view-model i.e .cs file.

Another work in the list is to delete the individual point on the graph plotter.

VII. CONCLUSION

Thus with the combination of WPF and dynamic data display which is already proved as powerful way to design complex GUI, the proposed method has added an extra advantage in the field of design and graphics.

This will solve many problems while designing graphics part and also will prove more user friendly. Also it is flexible enough to add more features or information while displaying.

REFERENCES

[1]. Dynamic Data Display Homepage available on http://dynamicdatadisplay.codeplex.com

[2]. "Professional WPF Programming" by Chris Andrade, Shawn Livermore, Mike Meyers, Scott Van Vilet.

[3]. Dr. Samrat O. Khanna, Mijal A. Mistry, "STUDENT MONITORING SYSTEM USING WPF AND WEB SERVICES", IJARCSSE, vol 2, JAN 2012

[4]. Cao Shukun, Zhang Heng, Ze Xiangbo, Yang Qiujuan, Ai Changsheng, "Software and Hardware Platform Design for Open-CNC System," in IEEE 2008.

[5]. Fran Jarnjak, "Flexible GUI in Robotics Applications Using Windows Presentation Foundation Framework and Model View View Model Pattern," IEEE, 2010.

[6]. Microsoft Corporation "Introducing Windows Presentation Foundation" (undated). Available at: http://msdn.microsoft.com/en-Us/library/aa663364.aspx

[7]. Microsoft Corporation, "XAML overview"(undated). Available at: http://msdn.microsoft.com/en-us/library/ms752059.aspx

[8]. Arlen Feldman, Maxx Daymon, "WPF in Action with Visual Studio 2008". Manning Publications Co, 2009, pp 283-285.