Enhancing Visual Basic GUI applications using VRML Scenes

Bala Dhandayuthapani Veerasamy

Abstract— Rapid Application Development (RAD) enables ever expanding needs for speedy development of computer application programs that are sophisticated, reliable, and full-featured. Visual Basic was the first RAD tool for the Windows operating system, and too many people say still it is the best. To provide very good attraction in visual basic 6 applications, this paper directing to use VRML scenes over the visual basic environment.

***Keywords*—** Cortona Control, Interpolator, Route, Sensor, Visual Basic, VRML

I. Introduction

MICROSOFT'S Visual Basic product [1] is defined as a programming system. Simply put this programming system is used to write Windows-based computer programs; it includes the Visual Basic language as well as a number of tools that help you write these programs. By using Visual Basic to create your own customized programs are not bound by the limitations of a particular "off-the shelf" computer program; rather, you can design applications to meet your own specific needs. A good computer program should be flexible enough to fit the task at hand, rather than having to modify your needs to fit the program. Visual Basic 6 is Microsoft's prior and greatest version of the Visual Basic programming language. Although writing programs can be a tedious chore at times, Visual Basic reduces the effort required on your part and makes programming enjoyable. Visual Basic makes many aspects of programming as simple as dragging graphic objects onto the screen with your mouse.

The Virtual Reality Modelling Language (VRML) [2] can be seen as a 3-D visual extension with animations of the World Wide Web (WWW). Since, it can construct websites with very attractively to be used in e-learning process such as virtual class rooms. People can navigate through 3-D space and click on objects representing URLs. VRML inserts itself seamlessly in the Web's connectivity. VRML browsers can access other VRML files via an URL. They can access any other format that then is passed to another application. On the other hand HTML browsers can be configured to fire up VRML helper applications (or plug-ins). HTTP servers, finally, can be configured to tell the client that a VRML (*.wrl) document is transferred.

Department of Computing, Mekelle University, Mekelle, Ethiopia (Phone: +251 914740551; e-mail: dhanssoft@gmail.com).

VRML defines a file format that integrates 3D graphics [3] and multimedia, has a variety of powerful mechanisms that provide the content creators with almost unlimited capabilities of building animated and interactive 3D content. However, in certain cases, the implementation of the content creator's intent with the use of scene elements described in the VRML97 Specification can lead to a large file size, low performance of the VRML browser, or even can be impossible. The Specification does not specify advanced scene description elements and mechanisms that can enhance the capabilities of content creators and improve size and performance, such as advanced geometry representations, texturing techniques, sensors, interpolators, support for new multimedia file formats and inter-object collision.

ParallelGraphics Cortona SDK [4] provides an Application Programming Interface (API) that enables authors and developers to integrate ParallelGraphics 3D technology into other applications using Visual Basic 6. Cortona SDK is an API, which you build four simple applications in Microsoft Visual Basic 6.0 that incorporate 3D graphics and are based on ParallelGraphics' Cortona Software Development Kit.

II. Working with VRML

VRML2 [2] has about eight major node types. They are Grouping nodes, Special groups, Common nodes, Sensors, Geometry nodes, Appearance, Interpolators, Bindable Nodes. The features of important nodes have explained in detail with the concepts and the Node reference sections in the VRML 97 specification. The structure of a WRL File or VRML (*.wrl) files have 3 basic elements. They are header, commands and nodes. The header tells the browser that the file is VRML and which version also. A header line is mandatory field. Comments are preceded by #. Most everything else is nodes. Nodes generally are in Capital letters. It has set of curly braces {...}. It has number of fields, all or some of which are optional.

Fields with that can have multiple values require braces [...]. Fields always start with lowercase letters.

The logic of programming interactive worlds is more or less the same as in traditional GUI programming. You will have to define what Events the browser should generate who should handle them and how they should be handled. The difference is that all VRML nodes can generate and receive events. ROUTEs will define how different VRML are bound together. In order to understand events, you have to rethink the concept

of a VRML node. You might want to read the entry “Nodes, Fields and Events" in the specification also. Typically, a VRML node is composed of a set of fields. Some of these, i.e. the exposed fields can be altered by incoming events. Each VRML node also has a set of incoming and out coming events defined and can hook together nodes by defining a route that links together any out coming and incoming event of the same time. In order to connect a node generating a message (event) with another node receiving a message you must use a “ROUTE" statement. Usually Events are triggered by sensors. A variety of Sensor Nodes allow you to detect what and when something happens.

The TouchSensor generates events as the pointing device “passes over" any geometry nodes that are descendants of the TouchSensor's parent group. In the code below we will make use of the “isActive" field which will change its value from FALSE to TRUE whenever the user points to it through mouse. It generates events as time passes. It can be used for many purposes including driving continuous simulations and animations; controlling periodic activities; initiating single occurrence events such as an alarm clock. Interpolator nodes are designed for linear keyframed animation. The OrientationInterpolator node interpolates among a list of rotation values specified in the *keyValue* field. These rotations are absolute in object space and therefore are not cumulative. The *keyValue* field shall contain exactly as many rotations as there are keyframes in the *key* field. The PositionInterpolator node linearly interpolates among a list of 3D vectors. The *keyValue* field shall contain exactly as many values as in the *key* field. ColorInterpolator node interpolates among a list of MFColor key values to produce an SFColor (RGB) *value_changed* event. The number of colours in the *keyValue* field shall be equal to the number of keyframes in the *key* field. The *keyValue* field and *value_changed* events are defined in RGB colour space. The following program has color and orientation effect.

Program 1. Sample VRML Animations

```
#VRML V2.0 utf8
DEF myColor ColorInterpolator {
   key          [   0.0,    0.5,    1.0   ]
   keyValue   [ 1 0 0,  0 1 0,  0 0 1 ] # red, green, blue
}
DEF Ori OrientationInterpolator {
  key             [     0          0.5                1        ]
  keyValue     [1 0 1  0, 1 0 1  3.1416,1 0 1  6.2832]
}
DEF transform Transform {
  rotation  0 0 0  0
  children [   Shape {
      appearance Appearance {  material DEF myMaterial
   Material { } }
           geometry Sphere {radius 2 }
          }
          ]
}
DEF myClock TimeSensor {
   cycleInterval 10.0      # 10 second animation
   loop          TRUE      # infinitely cycling animation
}
#coloring
   ROUTE myClock.fraction_changed TO
   myColor.set_fraction
   ROUTE myColor.value_changed TO
   myMaterial.set_diffuseColor
#rotations
ROUTE myClock.fraction_changed TO Ori.set_fraction
ROUTE Ori.value_changed TO transform.rotation
```

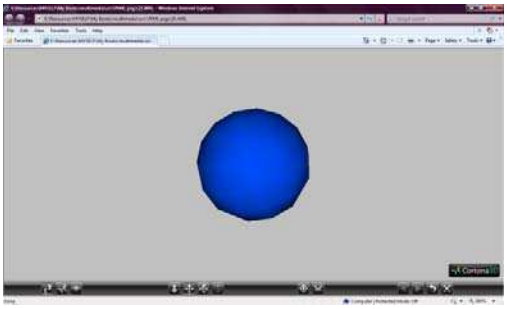

Fig. 1 VRML output using Internet Browser

III. Working with Cortona Control in VB

Cortona ActiveX control is a windowed control which enables an application with the functions of a VRML browser: interpretation of files in the VRML97 file format, presentation of their content to the user in the VB application window or in Internet browser window, and support of the user’s interaction with the scenes. This functionality and various setting of Cortona Control are provided by its properties, methods, events and related objects. To use Cortona Control and VRML Automation interfaces in your VB application, you need to reference them in this application. To add cortona control over VB choose the Components item from the Project menu, then in the Controls tab of the Components dialog window check the ParallelGraphics Cortona VRML Client 2.1 Type Library item. The Cortona VRML Control will available now in the Toolbox. The Cortona uses the methods, properties, events and object of VRML automation interface, create and use native scripting. Cortona Control includes a set of properties that customize its functionality and provide access to the related objects and Cortona engine. If the initial value of a property is not specified explicitly in the application, the default value of this property is used. Some of the read-write properties of Cortona Control have unchangeable default values. The default values of other properties can be modified by the user through the Cortona context menu. The following illustrates using cortona control in visual basic. (See Fig. 2.)

Scene loading: The Scene property specifies the VRML file to be loaded in Cortona; the baseURL property can be used for resolving of relative URLs. The ShowLogo and ShowProgress properties determine whether Cortona logo and the indicator of the loading progress are displayed in the Cortona 3D window when the scene is loaded in Cortona. To

delay scene displaying until all its resources are loaded, the WaitForAllResources should be used. The value of the LoadDroppedScene property specifies whether VRML scenes are loaded in Cortona if the user drops them in the Cortona Control window.

```
Private Sub Command1_Click()
CommonDialog1.Filter = "*.wrl"
CommonDialog1.ShowOpen
Cortona1.Scene = CommonDialog1.FileName
End Sub
```

Appearance of the Cortona Control window: The BackColor property determines the background color in the 3D window and the HeadLight property specifies the headlight state in Cortona window. To customize the user interface, the Skin and NavigationBar properties should be used to select the desired skin and whether or not it is displayed in the Cortona Control window respectively. The Mask property can specify an arbitrary clipping area in this window for VRML scenes rendering. The pixel buffer of the Cortona 3D window can be obtained through the Picture property.

```
Private Sub Command2_Click()
CommonDialog1.ShowColor
Cortona1.BackColor = CommonDialog1.Color
End Sub
Private Sub Command3_Click()
CommonDialog1.ShowColor
Cortona1.GradientBackColor = CommonDialog1.Color
End Sub
Private Sub Check1_Click()
If Check1.Value = 1 Then
Cortona1.NavigationBar = True
Else
Cortona1.NavigationBar = False
End If
End Sub
```

Navigation in scene: The NavigationType and NavigationStyle properties specify the current navigation type and style in Cortona respectively. To obtain the list of navigation types and styles which are currently available (supported), the SupportedNavigationTypes and SupportedNavigationStyles should be used. The ColliderMode and TravelSpeed properties determines the current collider mode (on, off, or set by the scene author) and speed of viewer's motion in the 3D scene (slowest, slower, normal, faster, or fastest).

```
Private Sub Form_Load()
Combo1.AddItem "Walk"
Combo1.AddItem "Fly"
Combo1.AddItem "Examine"
Combo1.AddItem "None"
Combo2.AddItem "Slowest"
Combo2.AddItem "Slower"
Combo2.AddItem "Normal"
Combo2.AddItem "Faster"
Combo2.AddItem "Fastest"
End Sub
Private Sub Combo1_Click()
If Combo1.ListIndex = 0 Then
   Cortona1.NavigationType = "WALK"
ElseIf Combo1.ListIndex = 1 Then
 Cortona1.NavigationType = "FLY"
ElseIf Combo1.ListIndex = 2 Then
 Cortona1.NavigationType = "EXAMINE"
ElseIf Combo1.ListIndex = 3 Then
 Cortona1.NavigationType = "NONE"
End If
End Sub
Private Sub Combo2_Click()
If Combo2.ListIndex = 0 Then
   Cortona1.TravelSpeed = 0
ElseIf Combo2.ListIndex = 1 Then
   Cortona1.TravelSpeed = 1
ElseIf Combo2.ListIndex = 2 Then
   Cortona1.TravelSpeed = 2
ElseIf Combo2.ListIndex = 3 Then
   Cortona1.TravelSpeed = 3
ElseIf Combo2.ListIndex = 4 Then
   Cortona1.TravelSpeed = 4
End If
End Sub
```

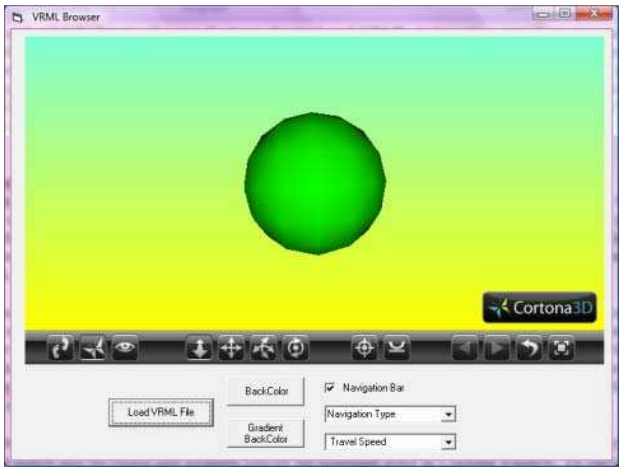

Fig. 2 VRML output using Cortona control in VB

IV. Conclusion

RAD systems provide a number of tools to help build Graphical User Interfaces that would normally take a large development effort to quickly build working programs. Visual Basic is one of the GUI applications on windows platforms. RAD systems have tended to emphasize reducing development time, produce extremely fast code. To provide very good attraction on visual basic 6 applications, this paper propose to use VRML scenes over the visual basic environment by using Cortona ActiveX control.

REFERENCES

[1] Francesco Balena, Programming Visual Basic 6.0, Microsoft press, 1999.
[2] Daniel K. Schneider, and Sylvere Martin-Michiellot, "VRML Primer and Tutorial", Available: http://tecfa.unige.ch
[3] Cortona3D software, Available: http://www.cortona3d.com
[4] ParallelGraphics Cortona SDK, ParallelGraphics, 2003, Available: http://www.parallelgraphics.com/products/sdk

Bala Dhandayuthapani Veerasamy was born in Tamil Nadu, India in the year 1979. The author was awarded his first masters degree M.S in Information Technology from Bharathidasan University in 2002 and his second masters degree M.Tech in Information Technology from Allahabad Agricultural Institute of Deemed University in 2005. He has published more than fifteen peer reviewed technical papers on various international journals and conferences. He has managed as technical chairperson of an international conference. He has an active participation as a program committee member as well as an editorial review board member in international conferences. He is also a member of an editorial review board in international journals.

He has offered courses to Computer Science and Engineering, Information Systems and Technology, since 8 years in the academic field. His academic career started in reputed engineering colleges in India. At present, he is working as a Lecturer in the Department of Computing, College of Engineering, Mekelle University, Ethiopia. His teaching interest focuses on Parallel and Distributed Computing, Object Oriented Programming, Web Technologies and Multimedia Systems. His research interest includes Parallel and Distributed Computing, Multimedia and Wireless Computing. He has prepared teaching material for various courses that he has handled. At present, his textbook on "An Introduction to Parallel and Distributed Computing through java" is under review and is expected to be published shortly. He has the life membership of ISTE (Indian Society of Technical Education).