

# VRMenu: A User Interface Toolkit for Immersive Applications

Ian Curington<sup>1</sup>

*Advanced Visual Systems Ltd.*

*Hanworth Lane, Chertsey, Surrey KT16 9JX, U.K.*

## Abstract

*A menu system for virtual reality and immersive stereo applications is described that allows for rapid prototyping, configuration changes, and 3D-layout behavior management. The VRMenu user-interface toolkit exploits 3D geometric graphics systems and allows interactive immersive data exploration without leaving the environment to modify control parameters.*

**Key words:** Virtual reality, immersive display, data visualization, user interface, application development.

## 1. Introduction

A primary goal of interactive data visualization in an immersive environment is to get inside the data space for exploration and discovery. Ideally, no other distractions should be introduced, allowing full attention to be directed to the detail in the visualization. With 3D perspective display systems in general, and stereographic display systems in particular, all visual representations should be available to the user in the same environment. Traditional menu and GUI systems break this concept, by presenting menu items in a flat 2D non-stereographic display mode. If these are mixed with immersive 3D stereo windows, they interrupt the immersive illusion, and detract from the visualization operations.

The VRMenu system is used to replace the 2D GUI with a 3D geometric equivalent, so that key parameters in the visualization can be controlled within the immersive environment without interruption. Just as in 2D GUI systems, layout behavior is important, for logical ordering and positioning of interactors. The VRMenu system includes tab panel grouping organization and 3D spatial layout control, without requiring any compiled code to be developed.

## 2. Related Systems

**2.1** The  $\mu$ USE software from MUSE Technologies Inc. [4] is an example of a large-scale application development system for perceptual, multi-sensory collaborative computing. As part of the immersive environment, a configurable menu system is available for custom application controls. The application programmer may directly control the menu system through an API while staying immersed in the data views.

**2.2** The CAVELib system from VRCO Inc. [6] also contains a menu system, again with application developer API to set up a custom configuration. The menu is a 2D element floating in 3D space, and interaction is generally controlled through a wand or stylus.

**2.3** The AVS VR Viewer, using the vrLib at NRL [2] creates 3D objects arranged in an array, which can be used to control selections and computational module parameters. The menu system is part of the vrLib, rather than AVS, so layout control behavior is not available in the same way as described in this paper.

**2.4** The SGI OpenInventor system [5] has interactors that are used to select and manipulate 3D objects using direct gestures. These are similar to the objects used for the menu system in VRMenu, but are not packaged as user interface objects in a rapid application development framework for immersive display.

**2.5** The existing user-interface development kit in AVS/Express [3] has a modular, object oriented structure, allowing for full rapid application development using visual programming. Each object is implemented as a 2D-window object as Motif or MFC widgets, and are normally managed in independent windows from the 3D visualization content windows. While the visualization content uses the viewer rendering pipeline, including the AVS/Express Multi-Pipe Edition [1] for multi-channel display, the existing menu system is not integrated into this path.

---

<sup>1</sup> Email: ianc@avs.com

### 3. VRMenu System Design

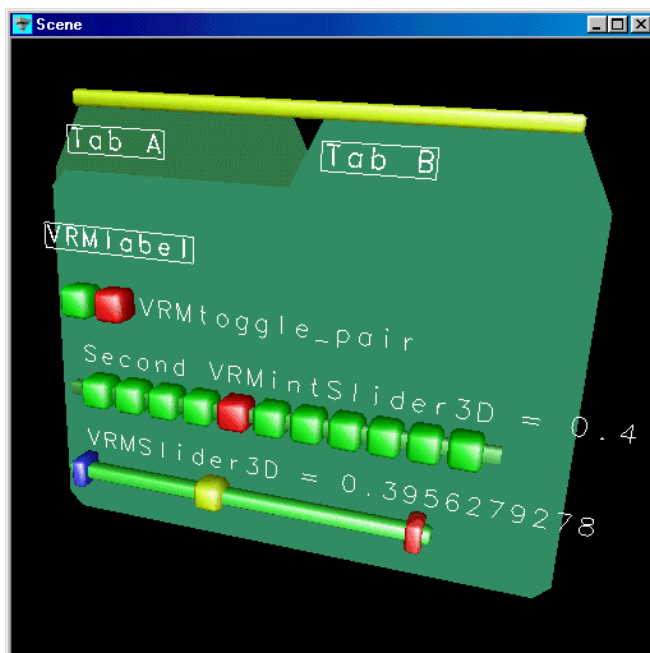


**Figure 1. Tab panel layout system with child menu interactive objects in 3D.**

The VRMenu system is designed to exploit high performance 3D graphics hardware. This is generally already available in immersive display systems due to the high demands of immersive data visualization. Because the VRMenu shares the same graphics pipeline, it operates in full stereo display modes, including head tracking unlike normal 2D user interface systems.

VRMenu objects co-exist with data in virtual 3D world, and are intended to be close-at-hand to the user, so switching attention between the data and menus is kept to a minimum.

One simple gesture allows movement placement of menu by user. As the user moves the viewpoint or the data within the environment, it is often necessary to move the menu system to a better place. The menu system floats a preset depth, normal to the viewer, however can be rotated and oriented as needed. The basic operational gesture uses a single point-select-drag operation. The act of selection operates toggles and choice buttons, while drag will move continuous valuator objects or translate the whole menu system in space, using a 6-DOF wand device. The device drivers

for input devices use the *Trackd* system from VRCO Inc. [6]

The VRMenu objects are designed as a user interface tool kit, with full application development support including a visual network editor for construction using the AVS/Express software system. Applications are assembled using drag and drop mouse operations, creating object-oriented relationships and attributes. The development environment also allows full C++ classes access to all VRMenu objects.

User interface layout formula is greatly simplified by a scheme using mouse connections between VRMenu elements. Each object passes layout information to the next object, so that layout formulas are made dynamic at runtime. The most typical layout is row-column behavior in two of the three dimensions.

The continuous valuator are derived from a base class that defines constrained transformations. The 4x4 transformation matrix of an object is filtered such that is locked onto a fixed 3D path, including end-stops for limited range movement. The continuous valuator objects such as sliders and range valuator are used for numeric parameter or scroll controls in visualization applications.

A sophisticated event handling system allows modelless menu operation, seamlessly updating the 3D display, user-interface objects and application state. The AVS/Express object manager event handler is used by the VRMenu system to propagate events to other objects.

#### **VRMenu Macro Descriptions**

##### **VRMbutton\_string**

A simple button object with a label string attached. The act of "selecting" this object triggers a "do" event. Deselecting it (or selecting anything else) has no effect. The effect is of a "one-shot" trigger event button. Styling is of a 2x wide yellow button with white text floating next to it.

##### **VRMmultiselect\_strings**

A Vertical list of cubes with a label per cube. The cubes may be picked and highlight in red. Any number of cubes may be selected, without radio behavior. The labels don't change color, just the cubes. The output is an array of integers, with the state of each list item set as 0 or 1.

## VRMradiolist\_strings

A Vertical list of cubes with a label per cube. The cubes may be picked, highlight in red, and the label string next to the cube also highlights in red. The output is the index number of the picked cube. The input is a list strings, so the number of cubes dynamically adjusts to the number of strings.

## VRMslider3Dsimple

Full interactive slider valuator widget. Select on the center slider object, then translate to move the slider between the stops. The value is available as an output parameter. The range of the output can be scaled by the min-max parameters. The objects made from very simple geometry. The end stops are not selectable.

## VRMslider3D

A Fancy Version of the simple slider, creating a full interactive slider valuator widget. Select on the center slider object then translate to move the slider between the stops. The value is available as an output parameter. The range of the output can be scaled by the min-max parameters. Fancy beveled cubes are used, center slider changes shape slightly when selected. Left and right end stop cubes will increment or decrement the output value when selected.

## VRMintSlider3D

A full interactive integer slider valuator widget. Click on one of the 10 blocks along the slider object. The value is available as an output parameter. No translation movement is required, just select one of the 10 preset positions.

## VRMtogggle\_string

A Simple toggle cube plus a label, where the label and the cube change to red if the toggle is set on. The second selection turns it off again.

## VRMtogggle\_pair

A simple two-cube toggle with a label. Clicking on either cube switches modes to the other one. One is always green, the other red. The label also changes color on selection.

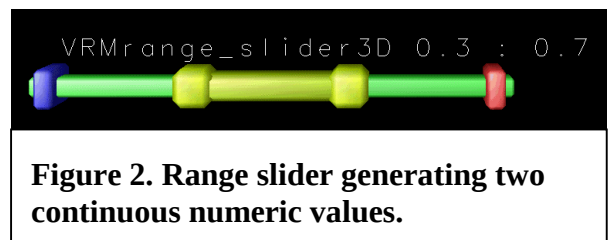
## VRMlabel

A simple stroke text 3D label, with layout parameter control. The label object is not selectable and is

designed for static labels only. The label appears in bold text with a box border by default.

## VRMtab\_group

A tab panel group visibility control system with two tab panels (fig. 1). The two input ports (in\_a, in\_b), take other widgets. Each list will then be part of one or the other tab groups. The scene is pruned such that only those groups on the visible tab panel are passed to the viewer. Clicking on either tab will swap tabs back and forth. Clicking on the top bar will toggle visibility of the entire tab panel system (hide/show).



## VRMrange\_slider3D

A range slider with variable center section. Two control cursors slide along the bar, yielding two floating point values, value\_a and value\_b (fig. 2). A bar connects the two cursors. The end stops if selected will increment both ends up or down as a pair.

## VRMslider3Dvert

A fancy version of the 3D slider with vertical orientation: a full interactive slider valuator widget. Select on the center slider object, then translate to move the slider up and down between the stops. The value is available as an output parameter. The range of the output can be scaled by the min-max parameters. Fancy beveled cubes are used; center slider changes shape slightly when selected. Top and bottom cubes increment or decrement the output value when selected, respectively. The label floats horizontally over the top.

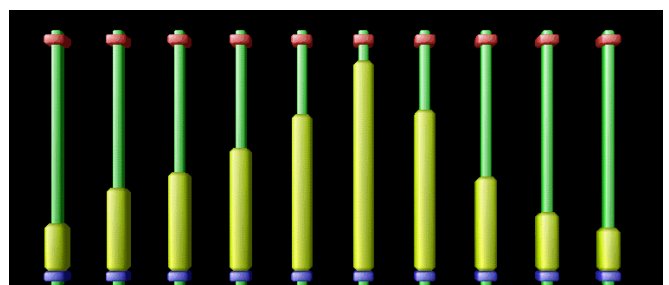


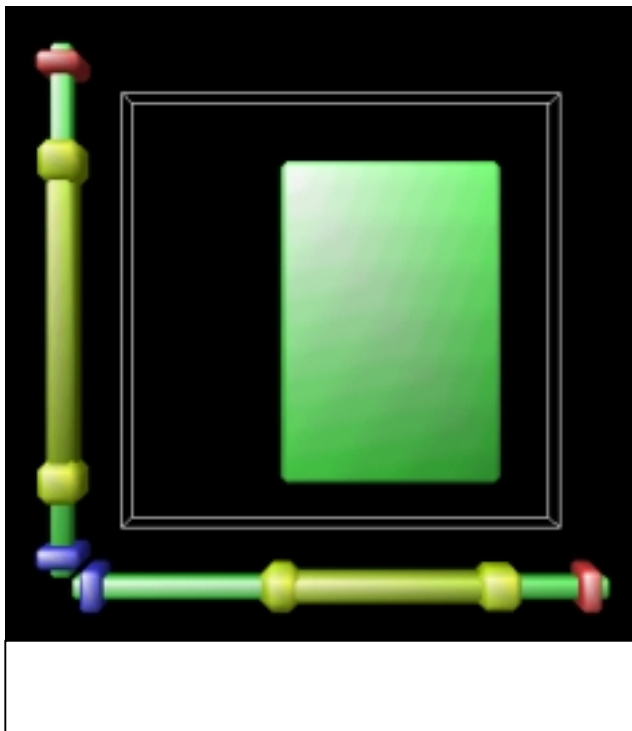
**Figure 3. Set of 10 vertical range bars to specify a range of values.**

## VRMslider3Dvertbar

A variation of a 3D vertical slider, with a bar that grows from the bottom, where the lower part of the center bar is fixed, and only the top end moves (fig. 3). This gives a quick visual cue to the current value. Select on the center slider object, then translate to move the slider between the stops. The value is available as an output parameter. The range of the output can be scaled by the min-max parameters. Fancy beveled cubes are used, center slider changes shape slightly when selected. Top and bottom cubes increment/decrement the value when selected. The vertical bar does not have a label object, so they can be stacked side-by-side, as in a VU meter, or audio graphic equalizer, or as an input control to a histogram equalization function.

## VRMrangeSlider3Dvert

A fancy range slider with vertical orientation without label. Includes two interactors for a range control. Select on one of the center slider objects, then translate to move the slider between the stops. The values are available as output parameters. The range of the outputs can be scaled by the min-max parameters. Fancy beveled cubes are used and the center slider cubes change shape slightly when selected. Top and bottom cubes increment or decrement both of the output values simultaneously when selected.



“Value\_a” and “value\_b” are independently available as output from this macro. No label is provided, as default layout is difficult with vertical placement, but can be attached externally.

## VRMrangeSlider3DZ

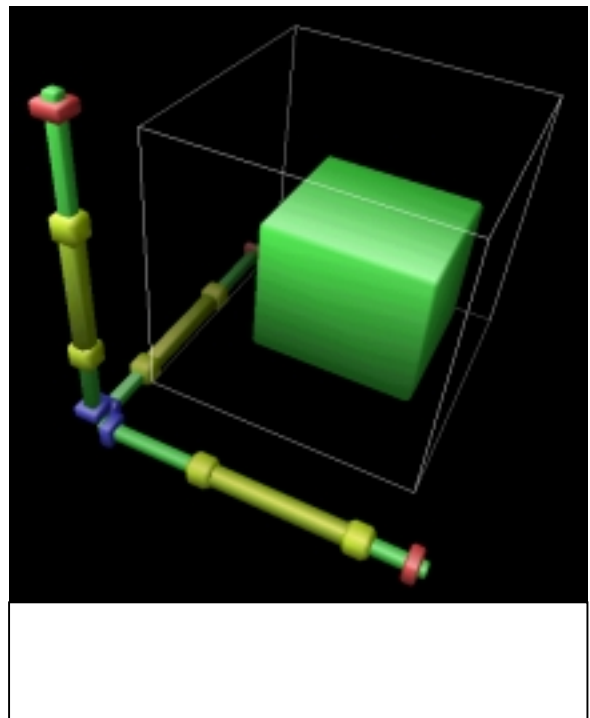
This is a range slider system with Z-axis orientation, similar to the VRMrangeSlider3Dvert, but with an orientation that comes out in depth toward the viewer.

## VRMrange2Dregion

Control device for a 2D crop region, with both vertical and horizontal scroll region interactors (fig. 4). The resulting region is shown as a thin square. Output values are xmin, xmax, ymin and ymax floats. This macro is constructed from vertical and horizontal range sliders described earlier.

## VRMrange3Dregion

Control device for a 3D crop region, with vertical, horizontal and depth scroll region interactors. The resulting region is shown as a rectangular 3D box. Output values are xmin, xmax, ymin, ymax, zmin and zmax floats. This macro is constructed from vertical, horizontal and Z-axis range sliders described earlier.



## 4. Conclusion

The VRMenu system is designed to give a flexible set of application construction objects to immersive 3D visualization applications. It overcomes the problem of 2D GUIs and allows full stereo display within a 3D exploration and discovery environment. The menu elements, valuator, choice systems, etc., are visual entities in a rapid application development system and include tab panel and layout behavior controls. The resulting menu system replaces normal user interface functions and operates in a full multi-pipe/multi-channel display environment.

## 5. References

- [1] Curington, I. "Multi-Pipe Rendering Framework for Visualization Applications" Proceedings of SIGRAD '99, Stockholm December 1999
- [2] Kuo, Lanzagorta, Rosenberg, Julier, Summers, Scannell "The AVS Virtual Reality Viewer" IEEE Visualization 1999 Late Breaking Hot Topics Proceedings, San Francisco, October 1999
- [3] "User Interface Kit" AVS/Express Developer Edition Reference Manual, Advanced Visual Systems Inc., Waltham Mass., June, 1996 (<http://www.avs.com>)
- [4] "Taking Collaboration into Orbit" Collaborative Design, Computer Graphics World, PennWell Publishers, September 1998 (<http://www.musetech.com>)
- [5] OpenInventor: <http://www.sgi.com/Technology/Inventor/>
- [6] CAVElib: <http://www.vrco.com>