Haptic Task Constraints for 3D Interaction

Rick Komerska and Colin Ware
Data Visualization Research Lab, Center for Coastal & Ocean Mapping (CCOM),
University of New Hampshire
{komerska | colinw}@ccom.unh.edu

Abstract

We have created a haptically enabled fish tank VR that we call Haptic-GeoZui3D that utilizes a set of haptic widget and data object elements to support rapid and intuitive interaction within a large geographical data space. We leverage the center of workspace navigation metaphor with a Phantom 1.0 haptic device situated in a fish tank VR arrangement to provide a synergistic environment for developing, demonstrating and evaluating these haptic elements. We have developed several principles to guide our effort, chief of which is the notion that haptic forces should be used to provide constraints on user tasks, rather than mimic physical object forces. This paper provides a detailed overview of Haptic-GeoZui3D and it's application in path planning for Autonomous Undersea Vehicles (AUVs), as well as some ideas for future development.

1. Introduction

In our laboratory, we are involved in creating highly interactive 3D visualizations of various oceanographic data as well as investigating issues related to monitoring and control of remotely operated and autonomous undersea vehicles [1]. We are interested in exploring the usefulness of haptics in providing a more intuitive and rapid interaction in this 3D domain.

It has long been recognized that for many user interface problems, adding task-related constraints can improve a user interface. For instance, computer-aided design systems employ sophisticated constraints based on concepts such as snap-dragging [2], forcing objects to line up or rotate about certain fixed axes. A related concept is the notion of "virtual fixtures", which employ force feedback to guide a user in carrying out manual and supervisory control tasks [3, 4]. There are of course many constraints inherent in real world interaction; e.g. physical objects do not in general interpenetrate each other when they come into contact.

Force feedback enables users to feel constraints embodied in a virtual element. Thus, for example, if a particular widget should only be allowed to rotate about a certain axis, then that constraint can be physically imposed to restrict the range of motion of the input device. To direct us in our development of haptic interface elements, we have developed a set of design principles that incorporate this notion of task constraints as a fundamental precept.

We have developed a haptically enabled fish tank VR that we call *Haptic-GeoZui3D* to provide a platform for the creation and evaluation of our haptic interaction elements. Fish tank VR refers to the creation of a small but high quality virtual reality that combines a number of technologies, such as head-tracking and stereo glasses, to their mutual advantage [5]. GeoZui3D, on which our system is based, stands for *Geographic Zoomable User Interface 3D* and it is a highly interactive system for visualization of a range of geographic data [6].

Our target application for Haptic-GeoZui3D is path planning for Autonomous Undersea Vehicles (AUVs). AUVs are an emerging technology, which utilize unmanned untethered robotic platforms to perform underwater operations in semi-autonomous or fully autonomous modes. An AUV mission profile is typically programmed and downloaded to the vehicle before the start of the mission. When laying out the path the vehicle should follow during it's mission, the planner in many cases uses 2D plan view maps (digital and paper) to plot the course. In many cases, verifying that the path selected is safe is done after the course is defined. Our application represents an attempt to integrate a typical mission-planning task, path planning, into an interactive 3D environment to provide a more intuitive means of defining transit paths as well as to help alleviate path safety concerns. The use of haptics provides immediate feedback on allowed interactions with both the tool widgets and data elements; i.e., the user is visually and haptically restricted from doing things that are not safe. The output from our program is a file containing transit path waypoints that the planner can incorporate into the vehicle mission profile.

Within the Haptic-GeoZui3D environment, we have developed a set of haptic data object and widget interface elements. Haptic object elements include representations of an undersea vehicle, it's trackline waypoints, and a sea bottom (or bathymetric) surface. Haptic widgets developed support scene navigation as well as vehicle and

waypoint placement using snap-to grids. We have also developed a haptic pie menu widget for in-situ 3D contextual menu selection and a haptic slider control widget for adjusting parameter values.

2. Development Environment

Our Haptic-GeoZui3D environment is essentially composed of a visualization system, a haptics device, and an integrated physical workspace.

The visualization component, GeoZui3D, utilizes OpenGL for graphics and Tcl/Tk for windowing and scripting capabilities across Windows, Linux and Irix operating systems. It also provides stereoscopic display with the appropriate hardware.

GeoZui3D has a design that makes it especially suitable for haptic enhancement. It uses center of workspace interaction as a unifying concept [6]. This concept refers to a set of interaction techniques defined with respect to a central fixed point in 3D space, conceptually within arm's length of the user. In GeoZui3D, objects in the environment are brought to the center of the workspace and operated on by contextually appropriate tools. This metaphor mimics typical physical workspaces that are commonly constructed, such as an office desk or technician's workbench. This mode of interaction is especially suited for fish tank VR because the goal is to create a small, high quality virtual reality environment that is localized near the user (unlike immersion VR which attempts to create a much larger virtual space).

A SensAble Technologies Phantom 1.0 haptic input device is used in our environment. The Phantom was chosen because its pen interface provides a simple and intuitive pointing device that is similar in function to a mouse in a 2D environment yet provides for 3D selection and application of fine force constraints. Programming of the haptic elements in our environment was done in C++ using SensAble's GHOST SDK. Our haptic workspace is constructed as a rectangular volume whose maximum size is set through the GHOST API to be 17 cm (width) by 14.5 cm (height) by 8 cm (depth). These dimensions are notably smaller than the advertised maximum device dimensions of 25 x 18 x 13 cm, but ensure that the entire rectangular volume is accessible. Although the working volume of the Phantom 1.0 device is small, it matches well with the size of the workspace in which we normally interact haptically with objects in our everyday environment. It is also highly compatible with fish tank VR and GeoZui3D because of the GeoZui3D interaction style which is to bring objects to the center of the workspace and give them an appropriate scale for haptic interaction. Collision detection between the Phantom proxy and the haptically enabled elements was performed using custom classes derived from the GHOST gstForceField and gstTriPolyMeshHaptic classes, while specific constraint forces were created in custom classes derived from the GHOST gstConstraintEffect class.

In Haptic-GeoZui3D, the visualization and haptic components are unified using a fish tank VR arrangement shown schematically in Figure 1. A horizontal mirror is used to superimpose virtual computer graphics imagery onto the Phantom workspace. The placement of the mirror also means that the Phantom and the user's hand are hidden from view. However, a proxy for the pen that the user holds is shown and, because the user's actual eye position is used to compute the CG imagery, visual and haptic imagery are co-registered at all times. To accomplish this, we use a 17-inch monitor set at a 45° angle above the mirror. Stereoscopic display is provided using NuVision Technologies stereo glasses with a monitor refresh rate of 100 Hz. We also provide head-tracking capability, through the use of a Polhemus FASTRAK system with a sensor mounted to the stereo glasses.

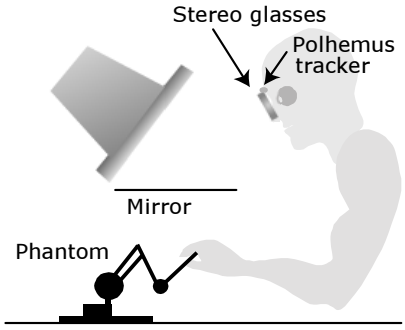

Figure 1. Fish tank VR setup.

3. Design Philosophy

In general, research suggests [7] that touching objects, especially with a single point of contact, provides little useful information about object shape. That task is best left to visual display. However, a number of studies have shown that considerable benefit can be gained by feeling constraints that are relevant to task performance [8]. For example, placing a peg in a hole is done faster if the force constraints are provided. Therefore, the focus of our work has been on finding ways of adding haptic constraints in such a way that they improve task performance.

With this concept in mind, we have evolved the following set of design principles to guide us in development of haptic 3D interaction elements:

- Haptically represent constraints rather than objects
- Display constraints both visually and haptically (constraints are possibilities for movement, limits on motion)

- Visually emphasize potential for interaction (manipulation hot spots)
- On contact, visually reveal additional constraints
- Make state information both haptically and visually accessible

An interesting way of combining constraints with a direct manipulation interface is to create haptic widgets [9]. The idea of a widget is to encapsulate both behavior and affordances in a single object. Thus if an object looks like a handle, and behaves like a handle when clicked on with a mouse, learning time will be minimized. We have extended this notion to our set of domain-specific haptic elements as well.

In our application, we have implemented a number of haptically enhanced data objects and interaction widgets. Widgets include elements designed to support object layout and scene navigation, as well as haptically enhanced menus and slider controls for mode selection and parameter adjustment, respectively.

We also differentiate between the notion of "passive" and "active" constraints. Passive constraints are force fields surrounding static elements. Active constraint forces guide a user in repositioning elements that have been selected.

Haptic-GeoZui3D leverages the center of workspace metaphor to let the user directly manipulate virtual objects and widgets in the environment. Elements that can be manipulated possess a visual and haptic hotspot by which the user can interact through the Phantom proxy (visually modeled as a pen). To select such an element, for example, the user moves the Phantom pen tip until its virtual proxy falls within the capture radius of the hotspot. At this point, the pen tip is subjected to a spring force that snaps the tip to the widget center. We have found that a radius of 3 mm and a spring force constant of 0.3 N/mm provide a nice combination of ease of selection and snap-to haptic feedback. This spring force constitutes a "passive" constraint and signals to the user that they can now manipulate the element, if they wish to. Additional visual cues, such as an element state numerical value and/or permissible manipulation track, are also displayed. To manipulate the element, the user presses the Phantom stylus switch while moving the pen. Appropriate haptic "active" constraints are imposed which properly guide the user during this interaction. The nature of these active constraints is based on the type of element being manipulated and its context with respect to other elements in the environment. Visually, the element changes to a standard color (we use green). Releasing the switch locks the haptic (and visual) position of the element and re-enables the passive constraints. The object or widget elemnt changes back to its default coloring. To detach, the user then simply pulls the pen away from the attached element, to beyond the 3mm radius, where the attractive force drops to zero.

While an element is being manipulated, the passive capture forces of certain other elements must be temporarily deactivated, while others must be left active. For example, while manipulating a trackline waypoint object, the user may want to use the grid widget to help place the object, but will not want to be captured by any of the scene navigation widgets.

In the next sections, we describe the implementation of our object and widget elements.

4. Haptic Data Objects

We have created several domain specific objects that are useful in our undersea vehicle path planning application that have been visually and haptically rendered. These include a bathymetric surface, a vehicle representation, and a vehicle trackline waypoint object. Figure 2 shows an example of each of these elements.

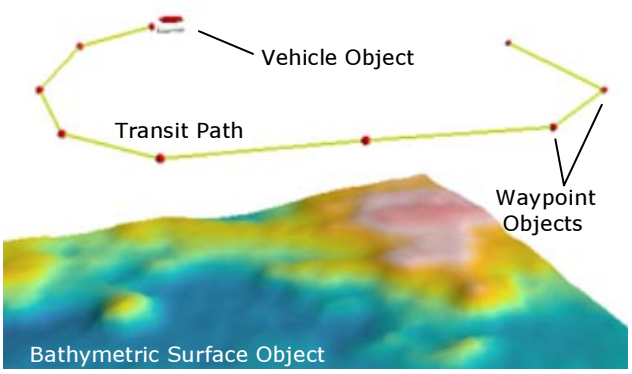

Figure 2. Haptically enabled data objects.

The vehicle object is designed to represent the location and orientation of an actual undersea vehicle, as well as provide a contextual anchor for menu options representing vehicle-specific tasks available to the planner when building a mission profile. Although not yet implemented in our application, examples include activating mission sensors, obtaining a GPS position fix, or initiating a communications link. Haptic-GeoZui3D supports the planning of a transit path through track lines represented as a set of connected waypoints. The user enters track-laying mode by selecting the *transit* option from the vehicle's menu (see Section 5.3). Two waypoint icons appear; one at the vehicle starting location and the other at the Phantom tip location, with a rubber band line between them. To place a waypoint, the user presses the Phantom switch. This causes the current waypoint to anchor in space and another to form at the tip of the Phantom location. In this manner, the user can lay down a connected path of arbitrary length. While in this mode, the

only force constraints present are imposed by the bathymetric surface and the placement grids (see Section 5.2). To exit track-laying mode, the user makes the appropriate menu selection from the attached waypoint object. The user can now go back and edit the path, as each waypoint is capable of being selected and manipulated. In addition, contextual menu options available while attached to the waypoint allow the user to add or delete waypoints at the selected location.

Visually, the vehicle and waypoint objects are shown as icons; the vehicle is represented by its geometry while trackline waypoint objects are represented as spheres. Both of these elements are capable of being manipulated, and thus are haptically represented for selection as described in Section 3, with their hotspot being the center of their visual icon. As the environment is scaled out (minified), these objects maintain their original visual and haptic size. This visual/haptic sizing model works when the interaction volume is much larger than the size of the objects, allowing us to essentially treat the objects as points. Since the primary task for operation on these objects is selection and relocation, the interaction typically is done at this minified scale and the model works well.

Both the vehicle and waypoint objects are capable of being manipulated freely in 3D space; there are no active constraints imposed on this motion. Movement limits are subject only to the haptic wall boundaries and the bathymetric surface object described next. The capture forces for the scene navigation widgets (see Section 5.1) are deactivated, preventing the user from selecting these elements while manipulating the vehicle or waypoint objects.

The bathymetric surface object is haptically rendered as a unidirectional constraint surface, which blocks the user from moving the Phantom from the topside down. This is appropriate to the intended task constraint of restricting the user from performing the undesirable action of placing trackline waypoints or vehicles below the ocean bottom surface. The haptic surface is coincident with the visual surface, and is built using the GHOST gstTriPolyMeshHaptic class. User motion is not constrained when moving from the underside of the surface upwards; this helps prevent the Phantom from becoming trapped under the surface. The bathymetric surface transforms appropriately as the scene is scaled, translated and reoriented. It cannot be manipulated.

5. Haptic Widgets

5.1 Scene Navigation

Scene navigation is performed using the widget set shown in Figure 3. It is designed to control the viewpoint by bringing a large virtual space into the range of the smaller haptic workspace and encapsulates the behaviors of yaw and pitch rotation, scaling and translation.

The yaw, pitch and scale widgets are anchored about the vertical axis of a set of crosshairs that are located at the center of the rectangular haptic workspace volume. Each widget is capable of being manipulated and are modeled as described in Section 3. These can be visually and haptically enabled and disabled through a "main menu" option selection. Our menus are context-sensitive based upon the currently selected widget or object, with the main menu available when no widget or object is selected.

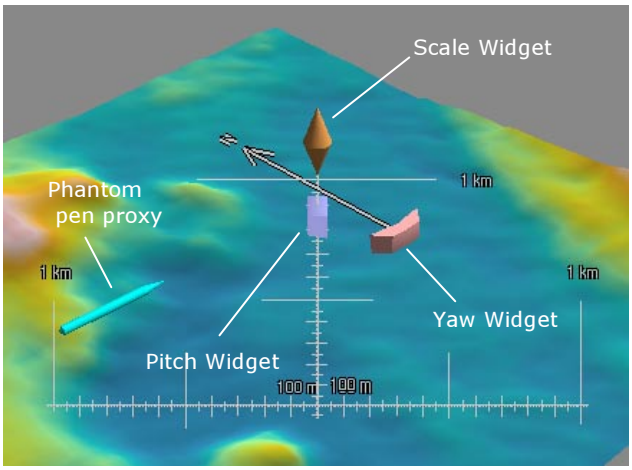

Figure 3. Scene navigation widgets with Phantom pen proxy.

The yaw widget is modeled as a tab handle anchored on a compass arrow passing through and perpendicular to the vertical scale axis. Its role is to allow rotation of the world about this axis. The widget hotspot is the tab center. Once attached to the handle, a circular band appears as shown in Figure 4, providing a visual cue as to how the handle will be actively constrained. When the user presses the Phantom switch, the active constraint forces restrict the Phantom tip movement along a 12mm radius ring co-registered with the surface of the band. Major and minor haptic detents are established at 10^{o} and 1^{o} increments, respectively, to provide additional position cues.

The pitch widget allows the user to rotate the world about the horizontal scale axis and is modeled as a tab handle, situated near the top of the vertical axis, underneath the scale widget. When the user attaches to the handle, a circular band appears as shown in Figure 4. This band has its center at the crosshairs and lies in the plane parallel to the vertical axis and perpendicular to the horizontal axis. In a similar fashion as with the yaw widget, the active constraint forces restrict the Phantom tip movement along a 27mm radius ring co-registered with the surface of this band. Haptic stops are imposed at $+90^{o}$ (toward the user) and -40^{o} to help prevent the

environment surface from hiding the widgets. Major and minor haptic detents are established at 15° and 0.5° increments, respectively. As the pitch changes, the orientation of the vertical axis, along with the location and orientation of the yaw, pitch and scale widgets, also changes.

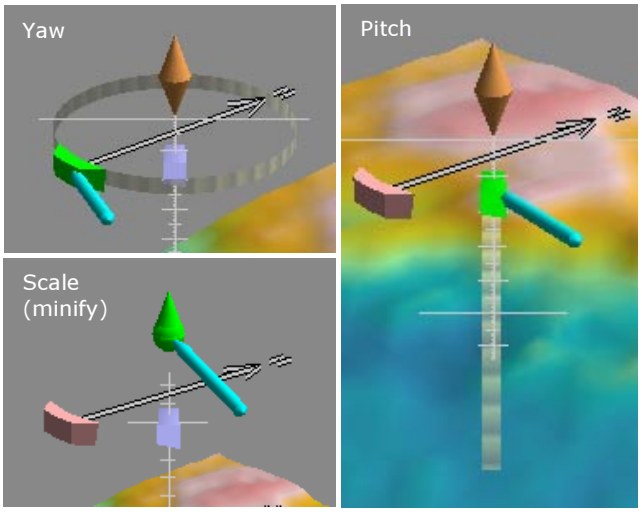

Figure 4. Scene navigation widgets activated.

Uniform scaling of the environment about the center of the workspace is implemented through use of the scale widget, shown visually as two opposing cones atop the vertical axis. Once attached to the widget center, the user presses the Phantom switch and pulls up or pushes down *along* the axis direction to zoom in or out, respectively. Visually, the cones alter shape to indicate the direction of scaling, as shown in Figure 4. Haptically, the hotspot remains fixed in space; in this case the dot product of the spring restoring force vector with the vertical axis controls the magnification and minification rate of the widget. We have normalized the control such that a 2 N force provides approximately a 4x per second scaling rate, which many users of our system have found to be a comfortable interaction rate. Note that scaling the environment does not alter the scene navigation widgets. They always maintain their fixed location and size at the center of the workspace.

Translating the environment within the workspace is handled in one of two ways. In the first method, the user, while not attached to any of the other widgets, simply presses the Phantom switch and directly drags the point of interest to the crosshairs. Visually, the Phantom pen proxy changes color to green, while maintaining a fixed position with respect to the dragged world. The frame of reference is the fixed haptic frame, which provides an intuitive frame for direct manipulation. A small amount of inertia is imposed while dragging to give the world a sense of "weight".

The second method is needed because our visual workspace size is a viewing frustum that is larger, most notably in depth, than the haptic space. This can lead to the case where elements are visible but beyond the touchable space. When the user wishes to select an element that lies outside of the haptic wall boundary, he or she simply reaches for the object until the Phantom tip encounters the haptic wall. The effect is to "push" the workspace boundary in the direction of the object. A vector is formed from the wall contact point to the center of the workspace that provides the direction of scene translation. The magnitude is proportional to the wall reaction force and is given by the equation:

Translation rate [m/s] = 750 * reaction force [N] / scale

We have found that the above relation provides a comfortable range of movement velocity given the default scene scale of 10:1 (visual meters / haptic mm). The wall spring constant used is 0.8 N/mm.

If the location of interest lies far outside the reachable haptic workspace, the user will typically employ the scaling widget to zoom out such that the location is reachable, then drag the location of interest to the workspace center. Scaling in on this new center permits more detailed study and manipulation.

5.2 Object Placement

We have developed a set of two haptic grid widgets to aid in the placement of data objects in the environment: the depth grid and the altitude grid. They are particularly suited for placing trackline waypoints, allowing for the easy creation of vehicle paths with a constant depth or height above the sea bottom. For both, the grid extents match the bathymetric surface extents, the logic being that the bathymetry defines the known working environment in the (x,y) plane. Figure 5 depicts the grids and associated bathymetry. The grids are visually and haptically enabled through a main menu selection. The grid gap spacing is adjustable by the user by means of an appropriate slider control widget (see Section 5.4). Attaching to a grid and pressing the Phantom switch allows the grid to be moved in the vertical (depth) direction; haptic constraints restrict the user motion to this axis.

The depth grid is visually represented as a flat transparent surface overlaid with square gridlines. The altitude grid is similar to the depth grid except that the surface contour is identical to the bathymetric surface contour.

The grid widgets implement force functions, co-registered with the visual grid, to provide a snap-to effect at each node. To achieve this, both grids use a GHOST gstTriPolyMeshHaptic object to provide a collision

detection surface. This allows the grid to transform appropriately as the scene is scaled, translated or rotated, and provides a simple means to query the Phantom for its local coordinates on the surface. When the Phantom proxy tip collides with the surface, passive across-grid constraint forces are enabled and provide an intuitive snap-to effect.

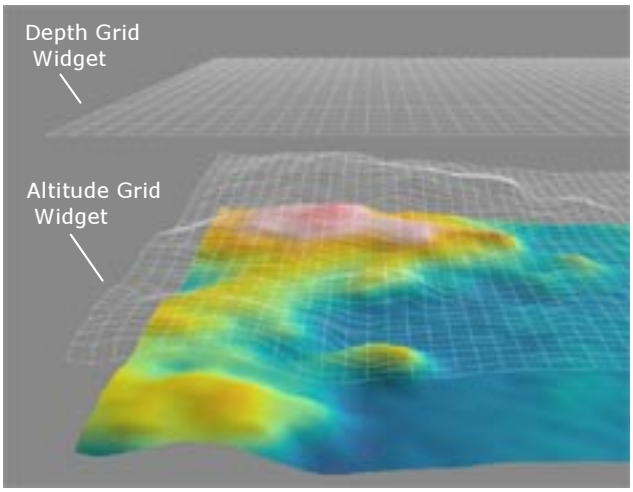

Figure 5. Depth and altitude grid widgets.

The constraint forces are calculated within custom gstConstraintEffect-derived classes using simple force functions. For example, the depth grid employs a force relation whose magnitude is given by the equation:

$$F\,[N] = k * \phi(dP)$$

ϕ : local scene coordinate space [m] $\rightarrow$ world haptic coordinate space [mm]

dP ≡ Phantom position offset from the closest node, in local scene coordinates
= Phantom local scene position –
{ (int (x / gap + 0.5)) * gap,
(int (y / gap + 0.5)) * gap, 0.0 }

k ≡ spring constant = 9.1 / (scale * gap)

and x and y are local Phantom scene offsets (in meters) measured from the grid corner origin, gap is the scene grid spacing (in meters), and k is a variable spring constant. The spring constant is inversely proportional to the product of the scene scale and gap spacing and provides for a consistent feel across different scales and gap sizes. It is bounded within the range of 0.13 to 0.52 N/mm to prevent the node force from becoming too "spongy" or exhibiting excessive vibration, respectively. The maximum snap force applied is 0.8 N; if the force magnitude is calculated to be greater than this, a force of zero is returned instead.

5.3 In-situ Pie Menus

We have developed a system that employs haptic pie menus to allow the user to perform mode selection using the pen interface. When activated, the pie menus are created centered in 3D space about the Phantom tip and positioned at right angles to the users view direction to account for the fish tank VR perspective. Forces are imposed to guide the user in making a selection. Figure 6a shows an example of a menu for a grid object.

To avoid overloading the functionality of the existing Phantom switch, we decided to add a second switch on the Phantom pen barrel. This switch would be used exclusively to pop-up a context-driven menu, in a similar fashion to how a right-button mouse click is handled in a typical 2D application. The second switch is mounted just behind and in line with the original Phantom switch. This switch was wired in parallel with the right-hand mouse button circuit from an old two-button PS/2 mouse. We then connected this mouse cable to a dual mouse port adapter that was in turn connected to the computer. This arrangement allowed the Phantom user (through this second switch) to generate mouse button events that could be caught in our application. We also added texture to the new switch surface to help the user haptically differentiate between the two buttons as otherwise they feel identical.

Visually, our menu layout is based on previous guidelines for designing pie menus [10]. We utilize a standard wedge size that subtends a 45° angle, with wedges aligned along the 8 ordinal compass points. A menu can display from 1 to 8 options. This, combined with our use of semi-transparent wedges, helps to reduce their tendency to visually obstruct the view of the environment. The menu layout has inner and outer radii of 5 mm and 16 mm, respectively.

When a menu is activated, we first disable all other environment forces. We then superimpose three assistive force components:

- 2D planar constraint
- Edge boundary constraints
- Wedge selection force

The haptic plane constraint is aligned in the same plane as the visual representation and acts to constrain the user to this 2D plane while making a selection. The edge boundary prevents the user from moving the Phantom tip outside of the valid menu widget region. This region includes the octagonal "home" space and visible wedge option spaces. Finally, the wedge selection force is activated when the user moves the Phantom tip more than 3 mm from the pie center. A spring force having a constant of 0.4 N/mm is created between the current Phantom location and a point whose radius is 8 mm and

centered on the closest wedge. This force acts to pull the Phantom toward this hotspot, at which time the wedge changes color from translucent white to red, indicating the option is ready to be selected (see Figure 6b). If the user decides not to select this but to choose another option, the user moves the Phantom until the tip crosses the wedge option boundary, at which point the selection force of the new wedge becomes active and pulls the Phantom to its center. These selection forces are capped at 0.6 N and 0.3 N for home-to-wedge and wedge-to-wedge movements, respectively.

Force is also used to indicate menu option selection as well as disengagement without selection. When the user is on an option wedge, they have the choice to push into the menu plane to make the selection. We employ a plane normal force threshold of 0.8 N to indicate selection. The user can also choose another option, as described above, or exit the menu completely by pulling away (toward their head location). We employ a plane normal force threshold of 1.1 N to indicate a menu exit without selection. In either cases where an option is selected or the menu exited, the menu will disappear, the normal environment object and widget forces are reactivated and, if appropriate, the selected option logic is executed.

5.4 In-situ Slider Controls

In addition to haptic pie menus, we have developed a haptically enabled slider control widget for adjusting parameter values. The slider control represents a type of very simple general purpose control widget, of which other examples include dials and buttons.

Our slider control is modeled visually as a thin rod with a sphere-shaped handle attached to the rod, located in 3D space. We orient the rod in a plane situated at right angles to the users view direction to account for the fish tank VR perspective. Within this plane, we orient the rod horizontally or vertically, although this orientation angle could be chosen such as to minimize visual obstruction with other data or widget elements. The length of the rod represents the available parameter range while the handle position on the rod represents the current parameter value, in a similar fashion to the traditional 2D slider control. The exact parameter value is also displayed next to the handle.

Haptically, our slider control draws upon the same passive/active force paradigm that we utilize with our other haptically enabled objects and widgets. The sphere handle exhibits the 0.9 N passive spring force to capture the Phantom pen tip. Once captured, the handle color changes to green and the user then has the option of adjusting the parameter value. She or he does this by pressing the Phantom switch, which triggers active force constraints to confine the pen tip to a 2 cm length haptic line. This also activates appropriately spaced detent forces along the line length. Releasing the switch locks the position of the haptic sphere handle, and parameter value, allowing the user to pull away and detach from the slider control.

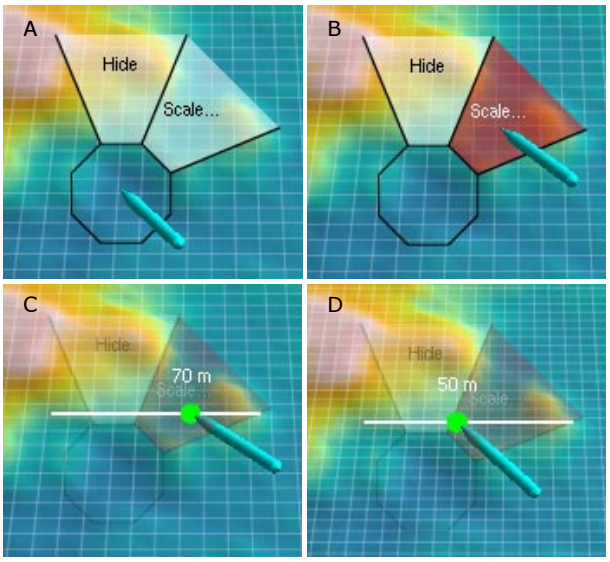

Figure 6. In-situ menu and slider control for adjusting the grid widget.

We have closely integrated the slider with our haptic pie menus, adapting ideas from marking menu and FlowMenu concepts for 2D menu/widget interaction [11]. Our design objective has been to make a smooth transition between the selection of a variable to be changed (using the pie menu) and the interactive adjustment of the value using the slider. To accomplish this, when a variable is selected, the pen tip is already on the slider cursor ready to make an adjustment. This is illustrated in Figure 6(c,d), where we show a slider control for adjusting the gap spacing of the depth grid widget. From the highlighted scale option position shown in Figure 6b, the user pushes through the wedge option to make the selection. At this point, the slider control is created with the Phantom tip initially attached to the control handle, slightly behind the menu. The menu forces are deactivated while the menu transparency is increased significantly. This provides a sense of context while not obscuring the slider control. The user manipulates the slider handle as described above, while direct feedback is provided through haptic detent forces, changing parameter value label, and immediate visual resizing of the depth grid widget gap spacing. After completing the parameter adjustment, the user detaches from the control, which removes both the pie menu and slider control and places the pen proxy and user interaction back into the scene space.

Note that sliders are general purpose control widgets that can be used independently from pie menus.

6. Future Concepts

We are presently exploring new concepts for haptically enabled elements to aid in the execution of specific tasks within our oceanographic domain. One such widget would allow us to easily create simple geometric force fields to keep the Phantom outside (or inside) the field region. An application of this widget could provide the ability for one user (or group) to demarcate a region so as to prevent another group tasked with planning a vehicle path from inadvertently plotting a course through the region identified by the first group.

We are also considering modifying the haptic representation of an object based on its scale. For example, when we wish to change the position of a vehicle or lay a trackline, as we describe earlier, it is appropriate to have a single haptic handle in which to interact. If we wanted to interact with a specific vehicle component however, we would naturally tend to zoom in on the vehicle to an appropriate visual scale where it filled a large percentage of our workspace. This could trigger a refinement of the number and location of haptic handles (and general interaction capabilities) that represent our ability to interact with that vehicle. We might be able to now add or remove a vehicle component, for instance. Zooming back out to a smaller scale would repartition our selection capability appropriately. This technique is similar to the idea of “critical zone analysis” for identifying and grouping logically related objects across different viewpoints [12].

7. Conclusion

We believe the approach we have taken in representing task constraints in the form of haptic data objects and widgets in a fish tank VR setup as described constitutes a powerful new way of interacting with 3D data environments. First, our approach provides a truly 3D interactive desktop environment. The Phantom provides a true 3D interaction device, with a familiar pen interface. GeoZui3D provides a VR environment optimized for use with georeferenced data objects as well as an interface metaphor that is well matched to the Phantom workspace. Stereoscopic vision combined with relatively high-resolution display and headtracking provide for accurate depth perception. The fish tank VR setup combines all of these elements in a synergistic manner to form a complete system. Secondly, the force feedback from the Phantom is used to augment our interactive experience in a meaningful way. Utilizing the forces to directly guide us in carrying out tasks, as well as following the supporting guidelines we mention in Section 3, appears to provide a very intuitive approach when interacting with data environments such as we have described with the AUV path planning application. These force and display techniques have the potential to minimize user learning time and interaction errors, and to speed up interaction time.

8. Acknowledgments

The authors gratefully acknowledge the support of NSF Grant 0081292, NOAA, and Matthew Plumlee and Roland Arsenault of the Data Visualization Lab.

9. References

[1] M. Plumlee, R. Komerska, R. Arsenault, C. Ware, L. Mayer, and S. G. Chappell, "Visualization Techniques to Support Monitoring and Control of Autonomous Platforms," *Proceedings 12th International Symposium on Unmanned Untethered Submersible Technology*, Durham, NH, 2001.

[2] E. A. Bier, "Snap-Dragging in Three Dimensions," *Proceedings Symposium on Interactive 3D Graphics*, Snowbird, UT, 1990, pp. 193-204.

[3] Z. Stanisic, E. Jackson, and S. Payandeh, "Virtual Fixtures as an Aid for Teleoperation," *Proceedings 9th Canadian Aeronautic and Space Institute Conference*, 1996.

[4] C. P. Sayers and R. P. Paul, "An Operator Interface for Teleprogramming Employing Synthetic Fixtures," *Presence*, vol. 3, pp. 309-320, 1994.

[5] C. Ware, K. Arthur, and K. S. Booth, "Fish Tank Virtual Reality," *Proceedings INTERCHI '93 Conference on Human Factors in Computing Systems*, Amsterdam, The Netherlands, 1993, pp. 37-42.

[6] C. Ware, M. Plumlee, R. Arsenault, L. A. Mayer, S. Smith, and D. House, "GeoZui3D: Data Fusion for Interpreting Oceanographic Data," *Proceedings Oceans 2001*, 2001.

[7] J. R. Flanagan and S. J. Lederman, "Neurobiology: Feeling bumps and holes," *Nature*, vol. 412, pp. 389-391, 2001.

[8] B. J. Unger, A. Nicolaidis, A. Thompson, R. L. Klatzky, R. L. Hollis, P. J. Berkelman, and S. Lederman, "Virtual Peg-in-Hole Performance Using a 6-DOF Magnetic Levitation Haptic Device: Comparison with Real Forces and with Visual Guidance Alone," *Proceedings 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, Orlando, FL, 2002.

[9] T. Miller and R. Zeleznik, "The Design of 3D Haptic Widgets," *Proceedings Symposium on Interactive 3D Graphics*, Atlanta, GA, 1999, pp. 97-102.

[10] D. Hopkins, "The Design and Implementation of Pie Menus," in *Dr. Dobb's Journal*, vol. 16, 1991, pp. 16-26.

[11] F. Guimbretière and T. Winograd, "FlowMenu: Combining Command, Text, and Data Entry," *Proceedings UIST '00*, San Diego, CA, 2000, pp. 213-216.

[12] S. Jul and G. W. Furnas, "Critical Zones in Desert Fog: Aids to Multiscale Navigation," *Proceedings UIST '98*, San Francisco, CA, 1998, pp. 97-106.