Multi-touch 3D Exploratory Analysis of Ocean Flow Models

Thomas J. Butkiewicz
University of New Hampshire, Durham, Thomas.Butkiewicz@unh.edu

Colin Ware
University of New Hampshire, Durham, colin.ware@unh.edu

Multi-touch 3D Exploratory Analysis of Ocean Flow Models

Thomas Butkiewicz and Colin Ware
Center for Coastal and Ocean Mapping
University of New Hampshire

***Abstract*—Modern ocean flow simulations are generating increasingly complex, multi-layer 3D ocean flow models. However, most researchers are still using traditional 2D visualizations to visualize these models one slice at a time. Properly designed 3D visualization tools can be highly effective for revealing the complex, dynamic flow patterns and structures present in these models. However, the transition from visualizing ocean flow patterns in 2D to 3D presents many challenges, including occlusion and depth ambiguity. Further complications arise from the interaction methods required to navigate, explore, and interact with these 3D datasets. We present a system that employs a combination of stereoscopic rendering, to best reveal and illustrate 3D structures and patterns, and multi-touch interaction, to allow for natural and efficient navigation and manipulation within the 3D environment. Exploratory visual analysis is facilitated through the use of a highly-interactive toolset which leverages a smart particle system. Multi-touch gestures allow users to quickly position dye emitting tools within the 3D model. Finally, we illustrate the potential applications of our system through examples of real world significance.**

Keywords - flow visualization; visual analysis; stereoscopic; multi-touch; particle system

I. INTRODUCTION

Ocean currents can change dramatically depending on depth, and modern ocean flow simulations can reveal these differences by outputting models with flow data at increasing numbers of depths. These models are inherently 3D, but all too often they are visualized one slice at time using traditional 2D visualizations. By focusing in and viewing flow patterns one depth at a time, the user loses the overall context as to how said patterns relate to those present above and below. By viewing the data in 3D, one can simultaneously focus on flow patterns at any depth range, while still preserving the overall context of how those patterns relate to the rest of the model. Furthermore, viewing in 3D naturally reveals the complexity of flow patterns with depth dependent structures.

Visualizing 3D flow data is inherently more challenging when compared to 2D. For example, depth ambiguity is an issue, as it can be hard to tell whether nearby objects are in front of, or behind each other. Our system utilizes stereoscopic rendering, in which each eye is presented with a slightly different image, generated from offset viewing positions, to produce a true 3D image which gives the viewer proper disparity-based depth cues. This ensures that the 3D positions of data items are correctly perceived and thus, it is easy to tell the relationships between them. This is especially relevant to our particular application, as ocean currents can often loop around or cross over at different depths, making it difficult (without stereoscopic 3D) to differentiate the paths of particles caught in visually overlapping currents.

Another major challenge when working with 3D datasets is selection within, and navigation through the environment. In 2D, navigation is as simple as scrolling around an image and zooming in and out. In 3D, the user can reposition their viewpoint with six degrees-of-freedom, looking at the model from any point, at any angle. Our system combines a focal-point based camera with efficient multi-touch gestures to allow the user to quickly move from viewpoint to viewpoint and examine areas of interest. This natural form of interaction is extended beyond navigation, as special multi-touch gestures must also be developed to enable selection of data and manipulation of visualization tools.

We should note that, while our system is optimized to take full advantage of a 3D compatible display with a multi-touch overlay, the design is general enough to be run on an ordinary desktop using a standard mouse and keyboard.

II. RELATED WORKS

In order to discuss prior work on flow visualization, it is useful to begin with some definitions. A *streamline* is line drawn into a steady flow (or an instant in time in an unsteady flow) that is always tangential to the flow direction. A *pathline* is the path (advection trajectory) of a particle dropped into an unsteady flow. A *streakline* is the contour formed by connecting particles emitted continuously from a point source, like a wisp of smoke from a cigarette. When a flow pattern does not change over time, streamlines, pathlines and streaklines are all the same.

Early systems created virtual equivalents of the smoke streams that are used in actual wind tunnels, or the dye that is streamed into flow tanks.[1] Other interactive methods for exploring modeled 3D flow fields include magic lenses, to increase the density of pathlines in an area of interest [2], and a "stream runner", that uses a slider to manipulate a time window, interactively revealing different subsections within a set of 3D streamlines.[3]

The techniques we present here have much in common with those described by Sobel et al. [4] They created a system using a CAVE immersive environment to visualize simulated blood flow through a fork in an artery. First, they pre-computed a large number of advection pathlines through the computational model space. Their interface included various interactive devices, including "sponges", devices that emit or absorb particle streams along the pathlines, and pathlets, short sections of trajectories that were animated along pathlines. These could be set to appear only near the walls of the blood vessels, or they could be randomly distributed in space and time.

Our system has also befitted from the concepts developed in the Center for Coastal and Ocean Mapping's (CCOM) GeoZui4D [5] software. This is a system designed to visualize heterogeneous geospatial data relating to oceanography. It supports visualization of flow model data, either using a technique similar to the stream-runner concept, or a grid of short, anchored pathlines to show tidal patterns in estuary or ocean flow models. One of the applications explored was the creation of mission plans for autonomous undersea vehicles. These vehicles have limited energy budgets, and taking current patterns into account is important.[6] The dye pot and dye pole devices in our system are based on these early prototypes developed at CCOM and applied in the visualization of ocean currents shown on the "Science on a Sphere" exhibit at the Smithsonian Museum of Natural History.

While systems similar to our have been developed, none utilize a combination of multi-touch and stereoscopic 3D in their interfaces. Schaeffer [7], for example, provides similar flow visualization tools for releasing dye particles as well as pathlet fields for general flow illustration, however his system's interface is entirely through a specialized haptic controller (a force feedback enabled pen-like device with buttons and a scrollwheel), and this haptic interface is the main focus of his work.

Our two-handed precision positioning gesture is similar to the "depth ray" and "lock ray" techniques proposed by Grossman and Balakrishnan [8] for selection in a 3D volumetric display. In the depth ray case, the user projected a ray into the volumetric display using a hand-held pointing device. A depth marker (cursor) was attached to the ray, and its position along the ray was determined by the distance from the display's outer surface to the users hand/pointing device using absolute mapping. This is similar to how we map the user's finger position along a vertical labeled scale in a depth panel to our cursor's position along a vertical pole through the model. Their lock ray technique expands upon the depth ray technique by having the user position the depth ray, press a button to lock it in place, and then move the device back and forth to move the depth marker along the locked ray. This is similar to the case in which the user performs our precision positioning gesture using only one finger at a time, which locks the vertical lat/long pole in place while selecting a depth along it.

This technique is also similar to the "Z-Technique" described by Martinet et al.[9] Their technique has the first finger position a ray, orthogonal to the screen plane, which selects the first object it intersects. This finger can then reposition the selected object parallel to the screen plane. Another finger then can be moved up or down on the screen to move the object further from, or closer to the user's viewpoint. Unlike the direct mapping in our labeled depth scale, they use indirect mapping based on the displacement of the second finger's movement, and no visual scale is provided.

Our one-handed pantograph selection technique is somewhat similar to the "dual finger midpoint" technique described by Benko et al. [10], in which a mouse cursor is positioned at the midpoint between two finger touch points. However, our technique differs in that we assume a pinching gesture with thumb and forefinger, and to avoid occlusion issues, we move the cursor outward from the midpoint in an orthogonal direction away from the hand generating the two touches. More significantly, our technique considers the distance between the two touches, and maps this distance to determine the desired depth of the cursor in 3D space.

Hancock et al. [11] seem to consider this inter-finger distance in their 3D "sticky fingers" translation gestures, where the user's two fingers "stick" to points on a 3D model. Moving the two fingers apart causes the model to move closer to the user's viewpoint, maintaining the illusion of the fingers being stuck to the same points on the model. Likewise, pinching the fingers together causes the model to move away from the user's viewpoint. This differs from our technique in that we do not necessarily have any object being manipulated; we are merely moving a cursor through potentially empty 3D space, which nothing to "stick" finger touch points to.

III. System Details

Our system employs a unique combination of off-the-shelf technologies to create an interface providing a powerful integration of high quality stereoscopic 3D graphics and natural touch-based interaction. While similar environments have been constructed, ours is remarkable due to its low cost, lack of wired devices encumbering the user, and low space requirements, which allow it to be deployed aboard research vessels or in standard office settings. An example of the system in use is shown in Figure 1.

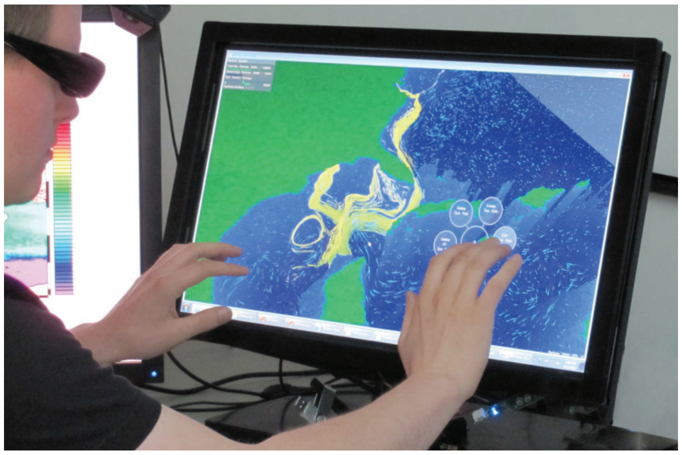


Figure 1. Our system in use in a typical office environment.

A. *Stereo Multi-touch Display*

The main component of our system's hardware setup is a custom multi-touch stereoscopic display. While many 3D capable monitors and multi-touch monitors are readily available, there are currently no suitable monitors successfully offering both technologies in an integrated fashion. Such a device is likely to be offered in the near future, but for now, we constructed our own. We begun by disassembling an Acer 24" 120Hz LCD 3D monitor, and extracting the LCD panel and control circuit boards. We mounted a PQ Labs 24" G3 Plus multi-touch screen directly to the LCD panel (to minimize parallax between the touch surface and the display surface). Finally, the setup is completed with NVIDIA's 3D Vision Kit, which consists of active LCD shutter glasses and an IR emitter to sync them to the monitor's refresh rate. These components are all readily available and cost roughly $1,000 USD, making this setup relatively affordable.

The display can either be mounted vertically, to preserve its functionality as an ordinary office monitor, or at an angle (~30 degrees), which has been shown to increase ease-of-use and decrease fatigue during prolonged touch-interaction. [12] [13]

B. *Depth Camera Integration*

A depth camera is an optional component that has the potential to enhance the usability of our setup. We have developed algorithms to utilize a low-cost ($150) Microsoft Kinect device, placed on the ceiling above a workstation, to perform a number of tracking duties, which were previously expensive or difficult to accomplish. This includes tracking the user's head so that the stereoscopic rendering can be updated to match the viewing direction, providing more realistic and less distorted 3D. Furthermore, by capturing the user's arms in front of the touch screen, it can also differentiate between whether the right or left hand was the source of touch points on the screen. This allows one to map different behaviors to each hand; e.g. the left hand is used for navigation of the virtual environment, while the right hand is used for selection/manipulation of objects in the environment.

C. *Model data*

Our system allows the user to import models generated by ocean flow simulations in the commonly used NetCDF format. Beyond just vectors for flow direction/speed, this format allows for the inclusion of many other variables, such as temperature and salinity. The methods used are general enough, however, that the system could be modified to import models in other grid-based formats, and even models based on irregular triangulated networks (through interpolation).

Upon loading a flow model, the system automatically generates geo-referenced terrain models of the land and seabed. This is done by examining what regions of the model do not have any water (zero depth), and the maximum depths encountered in all other areas. Automatically generating the terrain/bathometry allows the user to open flow models of any region, in any shape, and at any aspect ratio, without the need to provide the corresponding terrain files. The terrain model is colored to indicate the portions above (green) and below (dark blue) the water level, and has a subtle speckled texture applied to it. This texture helps provide better stereoscopic depth cues by giving more small details for the brain to match up, and allows for better perception of the shape and orientation of the terrain.

In addition to the flow model itself, the system can import vector data, in ESRI shapefile format. This data (lines, polygons, points) can then be integrated within the visualization for additional context and to aid navigation and selection. This can be seen later in Figure 12, where a polygonal coastal outline and the point locations of coastal nuclear power plants have been added for reference.

D. *Stereoscopic Rendering*

Stereoscopic rendering, providing each eye with an image rendered from a slightly offset position, is an important component of this system. Because we are rendering many small objects (pathlets, dye particles, etc) spread throughout a volume, it is critical that the user be able to perceive the relative distances between these objects. Without stereoscopic depth cues to resolve depth ambiguity issues, it would be very difficult to assess, for example, which particles are in front of other particles, or how deep a certain particle is relative to another.

Our system uses a focal point based camera system, meaning that the camera's location and movement is based around a focal point on the model. The user can navigate around the map by dragging/translating the focal point, can tilt the camera, changing the angle at which it views the focal point, and can zoom in and out, moving the camera away from, or closer to the focal point.

At all times, a focal point on the model is kept at the center of focus, both being physically located in the center of the screen, and at zero parallax (at the visual depth of the screen itself). Keeping the focal point at the zero parallax ensures it will be crisp, without any ghosting artifacts (as pixels here will display the same intensity to both eyes). Similarly, all interface elements (buttons, sliders, panels, etc.) are also kept at zero parallax. This is especially important since these will be directly interacted with by the user's fingers; if not at the depth of the physical screen, there would be conflicting depth cues.

Because of the sensitivity of human stereoscopic perception, we can achieve strong depth cues while only allowing objects to occupy a shallow depth range slightly in front of, and behind the screen. Indeed, this range has been shown to be fairly narrow, roughly 25% of the viewing distance in front of the screen and 60% behind it.[14] A diagram of our configuration is shown in Figure 2.

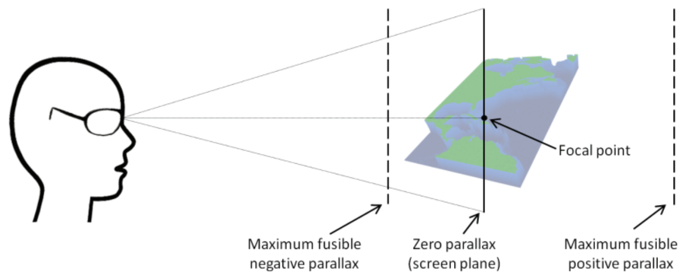

Focal point
Maximum fusible negative parallax
Zero parallax (screen plane)
Maximum fusible positive parallax

Figure 2. Diagram of how virtual models are positioned in terms of parallax.

Limiting the maximum parallax, especially the negative parallax (how far the virtual objects appear to come out of the screen), is crucial in avoiding diplopia (double vision) and visual fatigue. As such, when rendering each frame, the virtual eye separation and focal distance values used to calculate the viewing frustums for each eye are derived from the distance from the camera location to the current focal point on the terrain. This technique ensures that even as the user zooms in and out, the terrain model maintains a shallow depth range that will not produce too much parallax, while providing useful depth cues.

IV. Flow Visualization

Our system illustrates flow patterns through the use of a particle system and established visualization techniques that convert particle positions and movements into visual indicators of speed and direction. Tools are provided to allow the user to introduce dye particles in various ways to explore the flow patterns present.

A. Particle system details

The particle system used in our system is fairly basic, but provides the necessary functionality to illustrate flow patterns across a model. The particle system has three basic duties: introduce new particles, update existing particles, and remove unneeded particles.

Particles are added either randomly around the model as needed to maintain a user-specified number of particles for illustrating the general flow patterns throughout the model, or they are inserted in specific locations, tied to particle-emitting tools (detailed in Section V), at a rate specified by the emitter. If the total number of particles reaches the maximum allowed number (100,000+ depending on available processing power), new particles can either replace existing, older particles, or simply not be added. (We allow custom dye particles to overwrite "generic illustrative" particles, but not vice versa.)

Before each frame is drawn to screen, the particle system updates all existing particles. For each active particle, the flow vectors at its location (or interpolated from surrounding data points) are used (along with the time elapsed since the last frame was drawn) to translate the particle to its new location. If so desired, a particle can have additional vertical movement calculated based on comparing its current density value to that of the surrounding water. This can be done with a simple buoyancy value, providing constant vertical velocity, or it could be more rigorously calculated by providing salinity/temperature values which adjust to surrounding values over time. Once a particle is updated, its new location is stored, along with a timestamp, in a circular queue data structure. This forms a record of its previous locations and velocities, from which we derive its visual rendering.

After the update process, the particle system performs a cleanup operation by checking if any particles need to be deleted. This can be due either to a particle exceeding its user-specified lifetime, a particle flowing outside the confines of the current model, or a particle flowing inside a non-water area (or otherwise becoming stuck.)

B. Illustrating flow with pathlets

Each particle's advection trajectory, its pathline once inserted into the flow model, is traced out by a pathlet. A pathlet is a visual indicator of a particle's speed and direction over time. An example of how pathlets appear is shown in Figure 3.

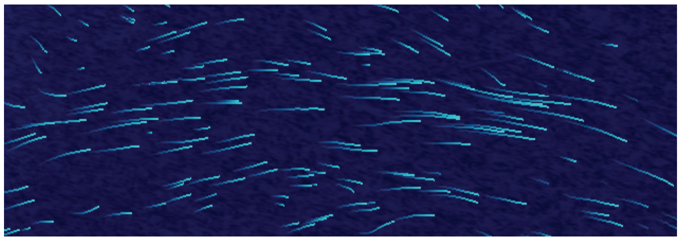


Figure 3. Pathlets illustrating a eastward flow.

Our pathlets are drawn as a series of line segments, starting at the current location of the particle, connecting each previous location in order until reaching a location past the elapsed time threshold. The user can specify the elapsed-time to draw, which changes the overall length of the pathlet, both giving it more or less visual weight, and also showing more or less historical trajectory data.

The opacity of the line segments at each positional vertex is modulated by the ratio of the time elapsed from the current frame over the maximum elapsed time of the pathlet. Thus, the most current position, at the front of the pathlet, is fully opaque; while the oldest position, at the end of the pathlet, is fully transparent. This gradual blending of the pathlet into the background as it "ages" has been shown to be a powerful perceptual cue as to the direction of the pathlet.[15] Indeed, even in static screenshots of our system, without the animation motion cues, it is easy to understand the direction and relative speed of different currents.

In addition to showing the current direction and historical path of a particle, pathlets also indicate the relative speeds within a field of particles. As a particle moves faster, its pathlet traces out a longer path, covering more pixels in total, and more importantly, more pixels with high-contrast, opaque coloring. This causes the faster major currents to have significantly higher visual weight than surrounding, slower moving waters. Even though they have the same density of particles per area, pathlets in faster currents form dense highly striated ribbons, while still waters manifest as sparse, shimmering regions of small, slowly drifting pathlets.

To provide the user with an overview of the flow patterns in all areas of the model, by default we automatically insert "illustrative particles" throughout the model. The pathlets tracing these particles' paths show the major currents, still waters, loop currents, etc. This provides a good reference to give the user context and aid in navigation of the model. At anytime the user can adjust the total number of particles as needed to either add emphasis or remove visual clutter. An example of the effectiveness of these illustrative particles is shown in Figure 4. While colored a non-distracting light blue (to blend with the dark blue background) by default, these illustrative particles can also be colored to indicate the values of particular variables across the model. For example, Figure 5 shows these particles colored to indicate different surface water temperatures, showing the Gulf Stream bringing warm water into the colder northern Atlantic Ocean. To provide an additional visual cue as to the depth of pathlets, their colors are modulated by depth, such that those in deeper areas appear darker and more saturated than those near the surface.

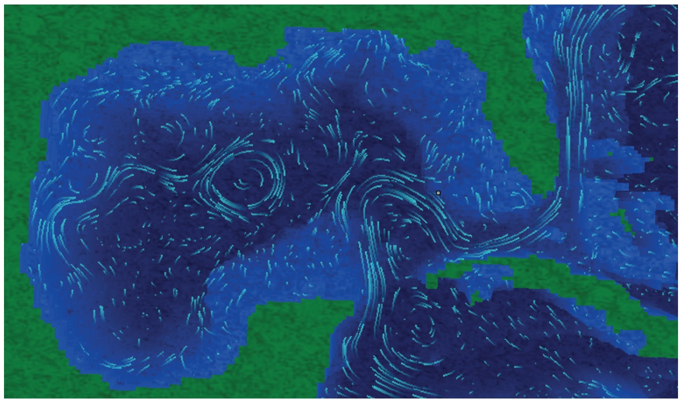


Figure 4. Randomly inserted illustrative particles in the Gulf of Mexico, showing the Gulf Loop Current (with ring) and the prominent Florida Current.

V. Dye Release Tools

For more focused analysis, we provide interactive tools that enable the user to insert dye particles into the model to explore the flow patterns in any particular regions of interest.

A. Direct insertion

For temporary exploration and experimentation, clusters of dye particles can inserted directly into the model at any point using the pantograph selection gesture described in the next section. Once inserted, they will move about in the surrounding flow and eventually fade away.

Once an area of interest is found, the user can affix dye emitters in place to continuously release dye particles as desired. We provide two types of emitters: dye pots and dye poles.

B. Dye pots

A dye pot is the most basic particle emitter. It is a single point location, from which particles are emitted continuously into the model, visually resulting in a constant stream of pathlets. When viewed from afar, the masses of pathlets blend together and resemble a streakline showing the general flow patterns away from the dye pot; upon closer inspection, the individual pathlets reveal details within the patterns and eddies.

Dye pots can be added anywhere in the model using either of the two positioning gestures detailed in the next section. Once placed, a control panel is spawned that allows the user to configure the emitter, both in terms of how it emits particles, and the properties of the particles emitted. In our system, a dye pot is actually treated as a dye pole with a single, point-sized emitter. This gives the user the flexibility, if desired, to expand a dye pot into a larger emitter or series of emitters.

C. Dye poles

In its simplest form, a dye pole is a vertical pole, extending from the seabed to the surface, along which any number of different dye particle emitters can be attached. An example of a dye pole with multiple emitters is shown in Figure 6. Dye poles can be added to the model at anytime by selecting the "add dye pole" menu option, and then specifying a lat/long location for the pole with a finger tap. Once a pole is added, a dye pole control panel is automatically spawned. This panel, shown in Figure 7, allows the user to add, remove, resize, and delete particle emitters along the pole's axis, control when and where the emitters release particles, and adjust the properties of the particles being emitted.

The left half of the panel shows an interactive diagram of the dye pole (with non-linear depth scaling) and any emitters currently attached to it. Emitters are drawn as color-coded cylinders, with relative radiuses representative of their actual sizes. The user can resize emitters by either touching directly their top or bottom and dragging, or doing the same to the arrow buttons at their side. (We found adding explicit buttons made it more clear how to resize emitters.) Wherever there is empty space available on the pole, buttons appear with a '+' symbol. When pressed, these add a new emitter, which is automatically sized to fill the empty gap in the pole. Touching an emitter selects it as the active emitter for editing.

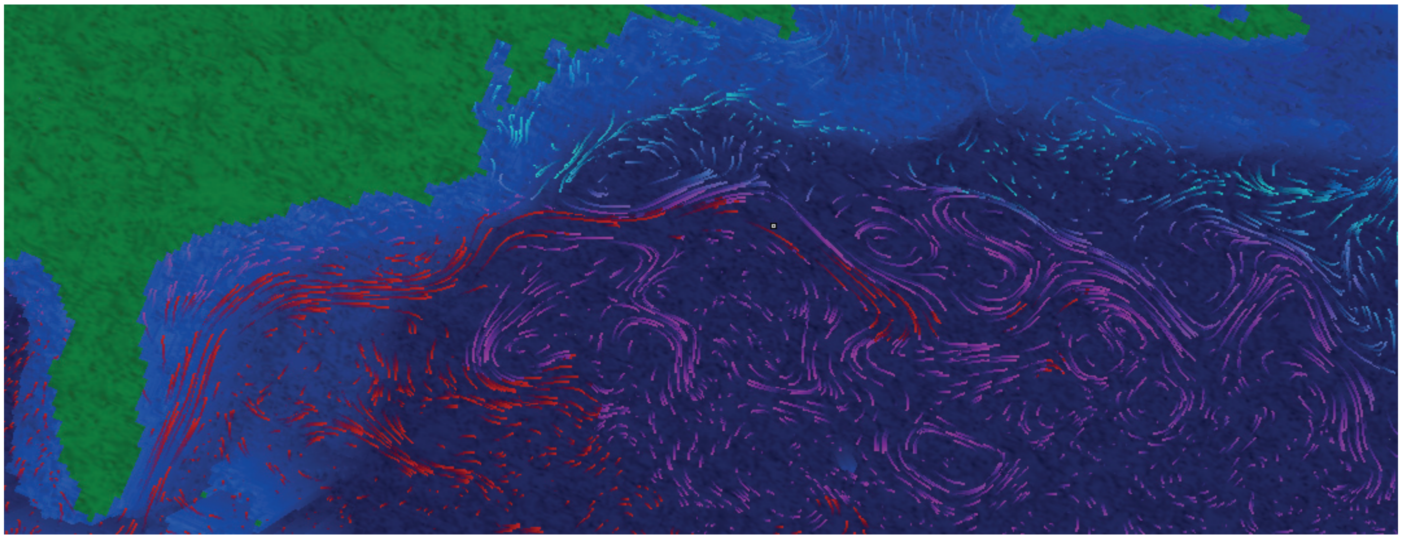


Figure 5. Illustrative pathlets colored to indicate water temperatures reveal the Gulf Stream current bringing warmer (red) water into the colder (blue) waters of the northern Atlanic Ocean.

On the right side of the panel is the emitter editor, where the user can control all the different properties of the selected emitter. This includes adjusting the size of the emitter, which is done via changing its depth bounds and its radius (the height and thickness of the cylinder), as well as setting the rate at which the emitter releases new particles. This editor also allows the user to set the properties of the particles to be emitted, including their color, the length of the pathlet trail drawn, the lifetime of the particle, and the relative buoyancy/density of the particle. For immediate feedback, the results of any adjustments are shown in real time in the main 3D view. In addition to these adjustments, commands are available to delete the emitter, and also to split the emitter into two separate emitters. When the user is done with the panel, it can either be collapsed into a small icon or closed entirely. It can be re-accessed at any time by selecting a dye pole with the 'edit dye pole' command.

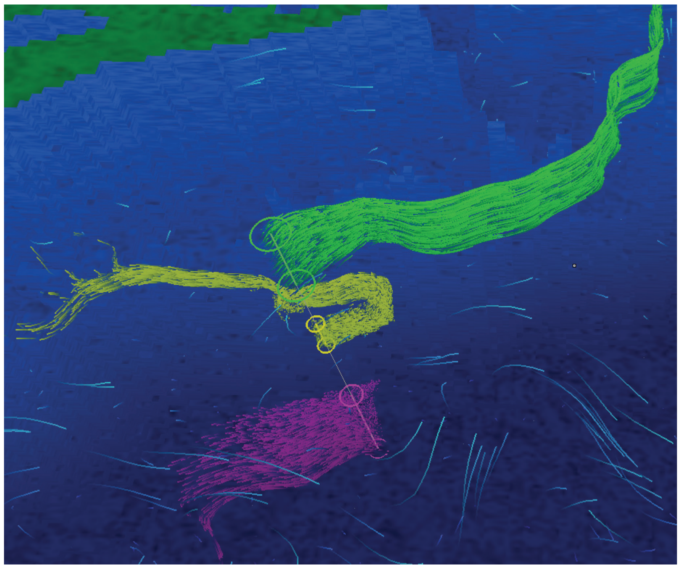


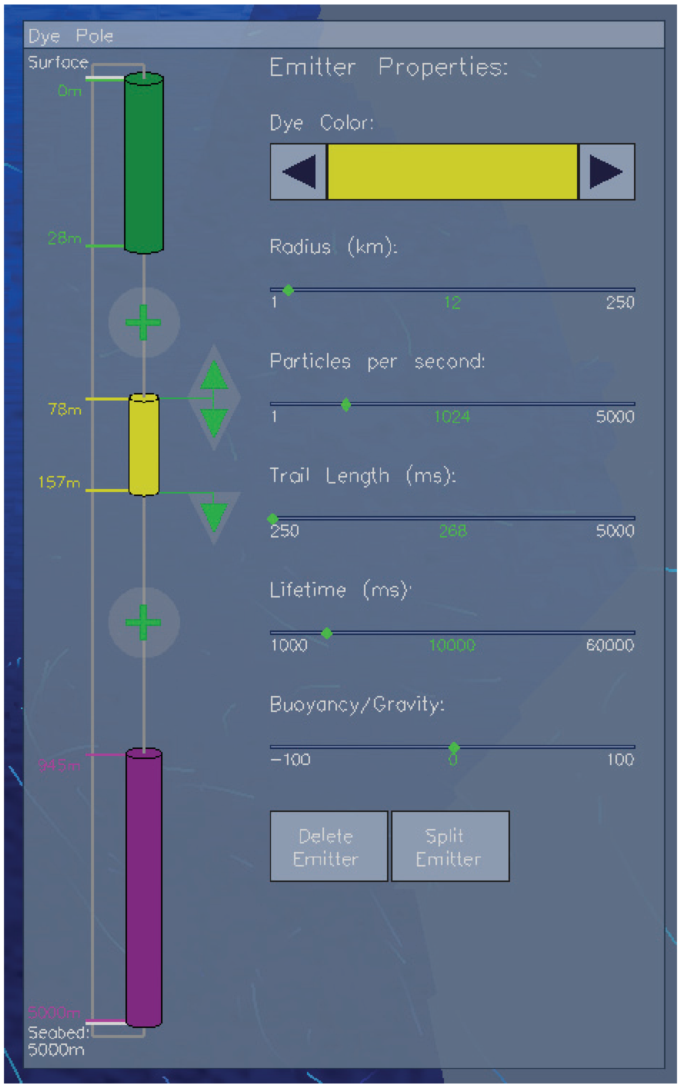
Figure 6. A dye pole (near Nova Scotia) with mutliple dye emitters along its axis to illustrate the significantly different flow patterns at various depths.

VI. Multi-Touch Interaction

Multi-touch displays allow for natural interaction and enable advanced direct selection and manipulation capabilities. However, in addition to the development of new gestures to complete these actions, moving from a traditional mouse/keyboard interface to multi-touch also requires redesign of some traditional interface elements.

A. *Interacting with onscreen entities*

A fingertip is simply far less precise than a mouse cursor, and thus the interactive interface elements in a touch screen application must be designed to accommodate this difference. In general, this leads to buttons (and other interface widgets) being much larger than traditional mouse-based interfaces, with more buffer room in between. Simply enlarging existing arrays of buttons to accommodate fingertips can quickly lead to running out of screen real-estate, which is better utilized for the visualization itself.



Emitter Properties:

Dye Color:

Radius (km):
1 12 250

Particles per second:
1 1024 5000

Trail Length (ms):
250 268 5000

Lifetime (ms):
1000 10000 60000

Buoyancy/Gravity:
−100 0 100

Figure 7. The dye pole editer panel for the dye pole shown in Figure 6. Here the user can add, delete, split, and resize emitters (color coded cylinders to the left), edit properties for each emitter, including the number of particles it will release and over what area, as well as properties of the particles themselves.

To replace static toolbars and other arrays of buttons, we implemented "on demand" pop-up menus. These radial menus (of which an example is shown in Figure 8) pop up whenever the user puts a single finger on the single and allows it to linger in the same spot for a short amount of time. Remarkably, the delay needed to differentiate a "linger" gesture without causing false-positive recognitions in place of other single-finger gestures (such as clicking a button or dragging a map) is quite short (~350ms). This style of menu is also known as a pie menu and has been shown to be faster than traditional linear menus in many circumstances [16], but its major drawback has always been the relatively large size of its menu items. However, in touch based interfaces the increased size is actually beneficial.

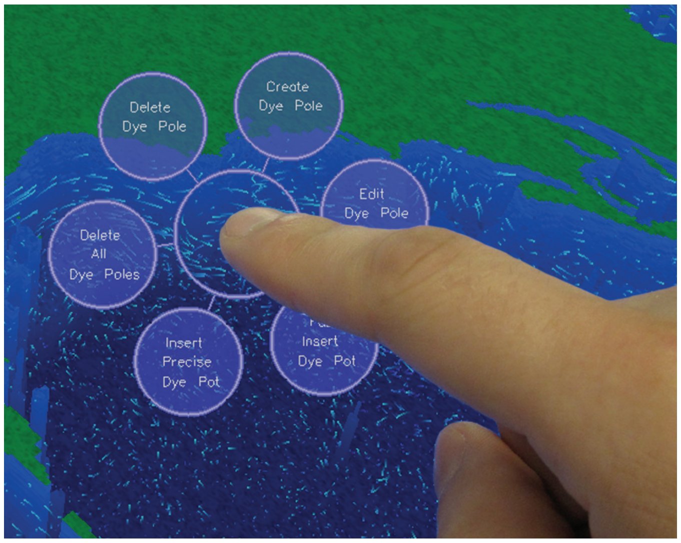


Figure 8. An example of a radial pop-up menu.

We attempt to use this pop-up menu technique to replace arrays of static buttons in as many places as possible. Lingering on different panels brings up specialized menus with the unique commands available for each panel, while lingering in the 3D environment spawns tools that can be used within that particular area. Reusing gestures throughout an application, while modifying their behavior based on the context in which they were made, reduces both the number of gestures the system must recognize and the user must memorize. [17]

As the user drags their finger over a menu item, it becomes highlighted to provide visual feedback. Menu items can also be hierarchical, such that upon finger-over, they expand sub menus, which branch off with addition commands. This reduces clutter and expands the total number of separate commands accessible at once.

B. *Navigation*

The default gestures enabled for touches not falling on interface elements are for navigation of the main 3D view of the model. We provide three simple gestures that allow the user to translate, zoom, and rotate the focal point based camera. The user can scroll around the model by dragging a single finger across the screen, which translates the camera's focal point, and thus the view of the model in the same direction. To zoom in and out, two fingers can be moved up or down vertically on the screen, which adjusts the viewing distance between the camera and the focal point. (The ubiquitous two-handed stretching gesture is also supported for zooming.) Finally, to rotate the view about the focal point, three or more fingers are placed on the screen and then moved about, the vertical and horizontal movements of which are mapped to heading and viewing angle values that control the position of the camera in a hemisphere above the model (with a radius of the current viewing distance.)

C. *Precise selection of 3D locations via 2D gestures*

The most challenging set of interactions in our system arise from the need to insert, select, reposition, and manipulate objects within a 3D volume using the 2D multi-touch interface. To accomplish these tasks we created two gestures that translate 2D touch points into 3D point locations within the model. One is designed for precision positioning, when accuracy matters. The other is designed for fast, fluid positioning, to lower the cost of exploration.

1) *Two-handed precise positioning*

The two-handed precision positioning gesture is the method used when the user wishes to put a cursor in a precise, known location and depth. It is optimized to be completed as a two handed command, but can also be performed with a single finger, albeit much slower. (This single finger functionality keeps this positioning method accessible for with mouse interaction for systems where no touch input devices are available).

Once this gesture mode is activated through the "Insert Precise Dye Pot" command, the first finger that is placed on the screen repositions a reticle (crosshairs with an open center) over the surface of the model with a vertical pole directly below it, stretching from the surface down to the seabed. Upon initial finger-down, a depth selection panel is spawned nearby. This depth selection panel shows the lat/long location of the pole (under the first finger), as well as a scale representing the water column along the pole, labeled with the depth of the seabed and the current depth of the cursor. (The scale is labeled on both sides to avoid finger occlusion issues.) Dragging a second finger along this scale repositions the cursor up and down along the pole (water column) under the reticle. When both fingers are down at the same time, moving either finger will reposition the cursor in either lat/long or vertical directions.

Releasing the finger on the scale while holding down the reticle finger will accept the current selection automatically. If the user lifts the reticle finger, the reticle will remain locked in place and the user can then interact with a single finger in either the depth scale panel (to adjust depth) or the main model view (to adjust lat/long position). Buttons to 'accept' or 'cancel' the selection are provided in the panel. An example of the depth selection panel being used to position a dye pot is shown in Figure 9.

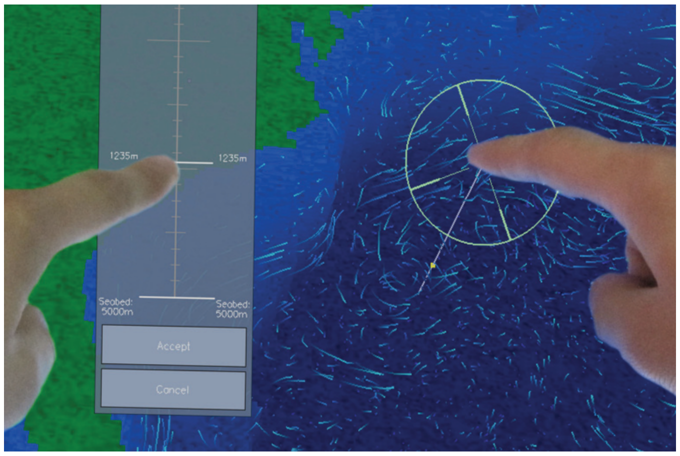


Figure 9. User placing a precision dye pot with the two-handed depth scale interface. The right hand finger controls the lat/long location of the vertical pole, while the left hand finger selects a depth along the pole from a labeled scale in the depth selection panel.

2) *Single-handed fast positioning*

The single-handed "pantograph" positioning gesture is used to quickly specify 3D coordinates in the model using only the thumb and index finger on a single hand. It is not as accurate as the technique in the previous section, since it does not provide a labeled scale, and the granularity of depth adjustments is limited by the relative distance between the users thumb and index finger versus the entire vertical height of the monitor.

Once this gesture mode has been activated, placing two fingers (intended to be the thumb and index finger) on the screen will begin positioning. Two circles will be drawn around the fingertips for visual feedback. A target point is determined by finding the vector between the two fingertips, then translating the midpoint orthogonally a short distance outward, away from the hand. (The user can specify right or left handedness, or this can be determined automatically via the Kinect integration). This target point, in screen coordinates, is projected onto the model, giving lat/long model coordinates for a vertical pole.

The distance between the two fingers is calculated, and used to determine the depth of the cursor along the vertical pole. A minimum separation distance (we use 2 cm) is specified based on fingertip sizes and the ability of the specific touch screen technology to resolve touches in close proximity. A maximum separation distance (we use 10 cm) is specified based on the comfortable maximum distance between thumb and index finger. (Automatic calibration could be done for each user by having them spread apart a calibration widget to measure their personal maximum inter-finger separation distance.) The system takes the distance between the two fingers, minus the minimum distance, and divides it by the maximum distance. This value is then multiplied by the total length of the vertical pole to get the depth value along it, where the cursor should be located.

As seen in Figure 10, a line connects the fingertips together, and two additional lines connect the fingertips to the cursor to provide direct visual feedback as to the effects of separating the fingers. If the users fingertips come too close together (below minimum distance), the line connecting them turns red to indicate this. Similarly, if they spread too far apart, the portion of the connecting line beyond the maximum separation distance turns red and the lines connecting to the cursor do not move past this point, indicating this is the maximum depth value.

A circular button follows the index finger, always a few cm to its upper right. This button allows the user to indicate they want to accept their selection, or otherwise initiate an action. This can be pressed either with the middle finger, or the index finger can be lifted (which locks the current cursor location) and used to press it. This can either end the gesture mode, as in the case of selecting an item or inserting a dye pot, or it can merely issue a command and allow positioning to continue, as in the case of placing dye particles directly. This latter case, for example, can be used to insert dye particles directly into the model whenever the button is held down, allowing the user to release particles continuously as they drag the cursor through the model.

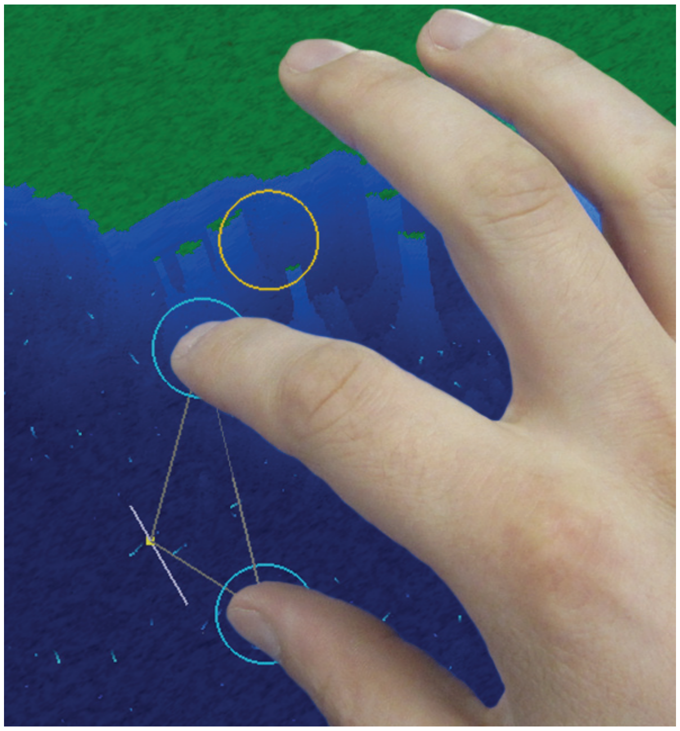


Figure 10. Placing a precision dye pot using the one-handed pantograph selection interface. The vertical pole is repositioned to correspond to a slightly offset midpoint of the two fingers, while the inter-finger distance adjusts the depth of the cursor (yellow dot) along the pole. Moving the fingers further apart drops the cursor toward the seabed, while bringing them closer together raises the cursor to the surface. The yellow circle accepts the current selection. Removing both fingers cancels the selection.

VII. Applications

Our system has the ability to provide a usable 3D visualization of the entire contents of a multi-depth ocean flow model. This provides a distinct advantage over traditional 2D methods for both those who are developing simulations, and the researchers consuming the final results of the simulations. Simulation developers can use our system to view intermediate results and gain an understanding of what parameters might need to be tweaked to produce their desired results. The end users of the generated flow models benefit from enhanced visualization capabilities, allowing them to quickly find and view the flow patterns they expect, while also revealing unexpected patterns through the use of exploratory analysis tools. The relatively easy to use interface also makes these flow models accessible to less technically trained researchers.

Beyond just presenting the contents of an ocean flow model, our system also provides limited simulation capabilities. As the physics-based particle behaviors increase in complexity, and aspects of other domain-specific simulations are incorporated, these capabilities will increase in usefulness and robustness. Even with the current system, experimentation shows the possibilities for powerful insight into how ocean flow patterns impact the marine environment. One particular research area in which there is obvious potential for this system is the prediction of impacted areas from pollutant releases.

Investigation of, and response to oil spills and other pollutant releases is an area in which examining flow patterns and the movement of particles within them is critical. The tools in our system can be used to reveal possible dispersal patterns from events such as the 2010 Deepwater Horizon oil spill in the Gulf of Mexico and the more recent release of radioactive coolant from Japan's damaged Fukushima nuclear reactors.

To roughly simulate a possible oil plume resulting from an offshore drilling accident, we can place a dye emitter on the seabed and configure it to release lower-density particles. These particles will rise up until they reach a depth at which the surrounding water is of similar density, where they will begin level off and spread outward. An example of a dye emitter configured this way can be seen in Figure 11.

Similarly, we can use the system get an idea of where coastal currents might take the radioactive coolant water released from the Fukushima reactors. The Naval Research Laboratory releases a daily high-resolution (1km) Navy Coastal Ocean Model for the region surrounding the disaster site. By releasing particles into this model from nested dye poles centered on the reactor site, we can get a sense of the most likely paths the contaminated waters will take any particular day. Figure 12 shows an example of this particular scenario.

VIII. Future Work

Perhaps the most important future work for this project is collaboration with actual flow modeling scientists to introduce more complex simulation aspects into the visualization. Currently, we can do basic buoyancy/gravity simulation based on relative density values of particles to surrounding water. However, more advanced particle behaviors, such as diffusion and weathering (e.g. oil into tarballs,) could be integrated to enhance the ability of the system to predict the potential paths and impact areas of pollutant releases.

Collaborating with actual simulation developers will also bring the ability to couple the visualization system with the simulation itself. An iterative experimentation cycle is then possible, in which visual analysis of the simulation's results leads to issuing changes that tweak the simulation's parameters. The model is then updated, and the changes are seen in the visualization for renewed analysis. If the visualization can show the differences between multiple iterations, then scenarios can be directly compared. For example, two particles could be released from the same point in two slightly different models, and the difference in the paths each follow could be drawn instead of individual pathlets.

While the system can currently handle time varying flow models by loading time steps one after another, we plan to incorporate more complex 4D visualization capabilities. An example of this would be pre-computing particle paths both forwards and backwards in time, and being able to track volumes that change shape over time as the particles defining them flow through the model.

Currently we provide dye pots, which are point-based, and dye poles, which can be either line-based or cylindrical-volume-based. However, many other useful types of dye emitters are possible. For example, releasing dye from a 2D plane, either a polygonal or freehand drawn shape, at a user defined orientation. Besides additional shapes, more advanced

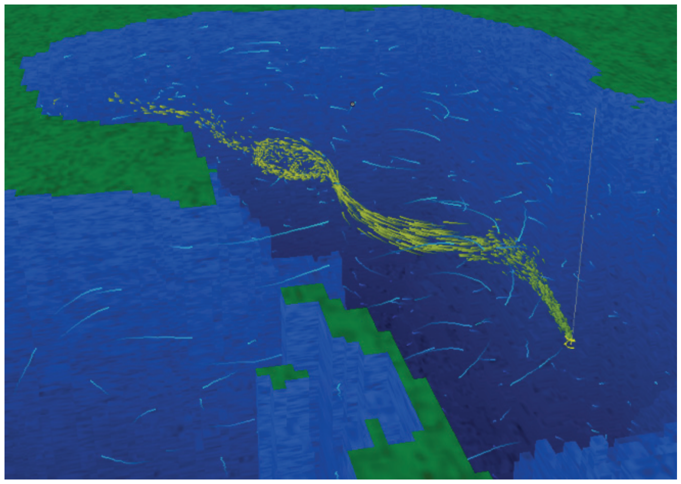


Figure 11. A dye emitter in the Gulf of Mexico releasing particles with density lower than that of the surrounding water. These particles rise quickly and then level off as they reach a depth where the water is of similar density.

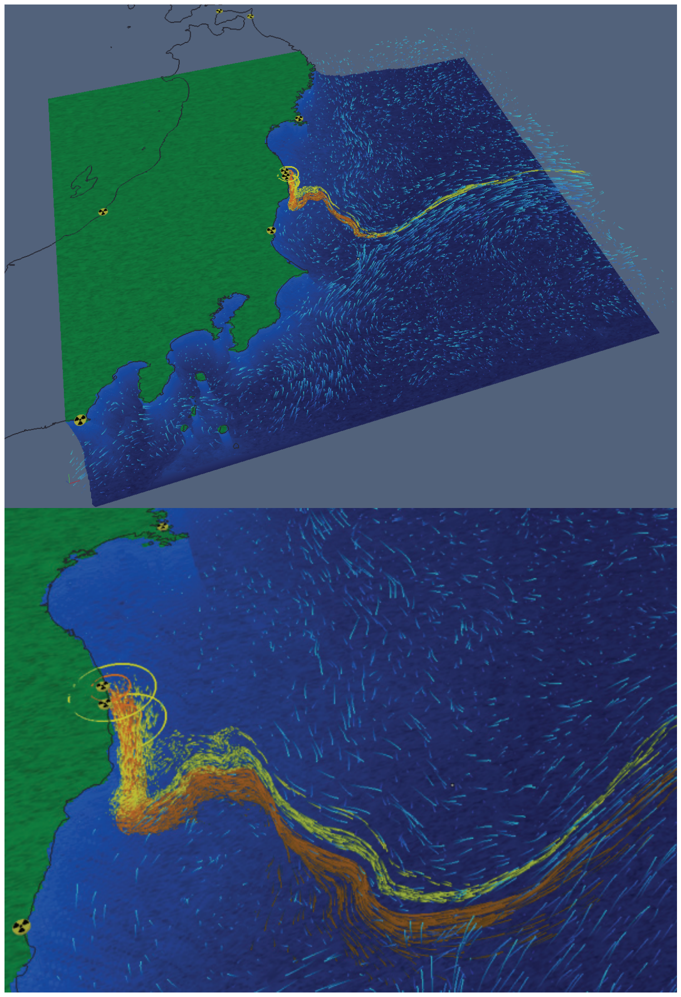


Figure 12. Example scenarios in which nested dye poles are releasing dye particles near Japan's Fukushima nuclear reactors to investigate possible dispersal patterns of radioactive coolant released from the disaster site. The particles are first carried south by coastal currents, and are then swept out into the Pacific Ocean via the Kuroshio current

volumetric dye emitters are possible, such as a dye emitter that conforms to the particle cloud boundaries generated from an existing emitter. Finally, a 4D emitter is possible, where dye emission varies over time. This could be useful for studying intermittent or cyclic phenomena such as runoff pollution entering estuaries after rainfall events.

Measurement tools are also an important feature currently in development. Flux measurement in particular should be particularly useful within this application. The user will be able to draw or otherwise define a 2D plane in the model, and see the rate at which particles pass through it. Beyond the number of particles, it should be quite useful to quantify the relative temperature and salinity of the particles passing through a region, as the deeper flow patterns in the ocean are driven by these differences.

Tracking the individual particles generated from an emitter, and then forming a volumetric cloud encapsulating them could provide useful regions of interest for probe-based analysis techniques.[18] This would allow for deeper analysis of, and comparisons between different currents. It might also provide useful impact assessments for pollutant release scenarios. Because the regions of interest in previous probe-based applications are generally static, the dynamic, fluid regions that could be generated within flow models would be a very interesting extension of that work.

Finally, we are currently evaluating the effectiveness of our multi-touch gestures. To conduct a classic Fitts's law study, we created a game-like environment, in which participants must move objects into specific positions using the various multi-touch gestures we created. By looking at the time to complete tasks while varying the object sizes, positions, and relative movements required, we can isolate the index of performance for each gesture. This should help us understand which gestures are most effective, what makes them effective, and how to refine future iterations of these gestures.

IX. Conclusions

While ocean flow simulations are currently outputting flow models with data at increasing numbers of depths, many oceanographers are still using traditional 2D interfaces to visualize these models. Viewing flow models one depth slice at a time causes the user to lose sense of the overall structure and how the patterns at one level relate to those above and below. However, 3D methods can provide this sense of context, as well as superior perception of the structures and patterns contained within these complex and dynamic models. Perhaps this disconnect between the software researchers use and the software researchers need is due to a lack of effective and easy to use visualization tools, driven by the challenging nature of designing easy-to-use interactive 3D visualizations.

We have attempted to address this need with our system, which combines stereoscopic and multi-touch technologies to provide an advanced, natural interface capable of presenting complex 3D flow visualizations in an understandable and accessible fashion. With only a limited amount of affordable hardware, oceanographers can replicate our setup and use our system to explore the contents of ocean flow models and even conduct limited simulations within them. We intend to continue developing this system, expanding upon its features, and refining our multi-touch gestures to build up a toolkit which will showcase the potential of the multi-touch stereoscopic display combination for effective visual analysis of 3D data, both oceanographic, and in general.

References

[1] K.-L. Ma, P. J. Smith. "Virtual smoke: an interactive 3D flow visualization technique", *Proc. of the 3rd conference on Visualization '92* (VIS '92), pp. 46-53, 1992.

[2] A. Fuhrmann, E. Gröller, "Real-time techniques for 3D flow visualization", *Proc. of the conference on visualization '98* (VIS '98), pp. 305-312, 1998.

[3] R. S. Laramee, "Interactive 3D Flow Visualization Using a Streamrunner", *Proc. Extended Abstracts ACM CHI '02*, pp. 804-805, 2002.

[4] J. S. Sobel, A. S. Forsberg, D. H. Laidlaw, R. C. Zeleznik, D. F. Keefe, I. Pivkin, G. E. Karniadakis, P. Richardson, S. Swartz, "Particle flurries", *Computer Graphics and Applications*, vol. 24, no. 2, pp. 76-85, 2004.

[5] R. Arsenault, C. Ware, M. Plumlee, S. Martin, L. Whitcomb, D. Wiley, T. Gross, A. Bilgili, "A system for visualizing time varying oceanographic 3D data", *Oceans '04 Techno-Ocean '04 (OTO'04) Conference Proc.*, pp. 743-747, 2004.

[6] R. Komerska, C. Ware, "Haptic GeoZui3D: exploring the use of haptics in AUV path planning", *13th Unmanned, Untethered Submersible Technology Symposium CD-ROM Proc.*, 2003.

[7] S. Schaeffer, "An augmented haptic interface as applied to flow visualization", M.S. dissertation, University of New Hampshire, 2007.

[8] T. Grossman, R. Balakrishnan, "The design and evaluation of selection techniques for 3D volumetric displays", *Proc. of the 19th annual ACM symposium on user interface software and technology* (UIST '06), pp. 3-12, 2006.

[9] A. Martinet, G. Casiez, L. Grisoni, "The design and evaluation of 3D positioning techniques for multi-touch displays", *Proc. of IEEE Symposium on 3D User Interfaces 2010* (3DUI), pp. 115-118, 2010.

[10] H. Benko, A. D. Wilson, P. Baudisch, "Precise selection techniques for multi-touch screens", *Proc. of the SIGCHI conference on Human Factors in computing systems* (CHI '06), pp. 1263-1272, 2006.

[11] M. Hancock, T. ten Cate, S. Carpendale, "Sticky tools: full 6DOF force-based interaction for multi-touch tables", *Proc. of the ACM International Conference on Interactive Tabletops and Surfaces* (ITS '09), pp. 133-140, 2009.

[12] A. Sears, "Improving touchscreen keyboards: design issues and a comparison with other devices", *Interacting with Computers*, vol. 3, issue 3, pp. 253-269, 1991.

[13] B. Ahlström, S. Lenman, and T. Marmolin, "Overcoming touchscreen user fatigue by workplace design", *Posters and short talks SIGCHI '92*, pp. 101-102, ACM, 1992.

[14] S. P. Williams, M. D. Parrish, "New computational control techniques and increased understanding for stereo 3-D displays", *Proc. SPIE Stereoscopic Display Applications*, pp. 73-82, 1990.

[15] D. Fowler, C. Ware, "Strokes for representing univariate vector field maps", *Proc. of Graphics Interface '89*, pp. 249-253, London, Ontario, 1989.

[16] J. Callahan, D. Hopkins, M. Weiser, B. Shneiderman, "An empirical comparison of pie vs. linear menus", *Proc. SIGCHI conference on Human factors in computing systems* (CHI '88), pp. 95-100, 1998.

[17] M. Wu, C. Shen, K. Ryall, C. Forlines, R. Balakrishnan, "Gesture registration, relaxation, and reuse for multi-point direct-touch surfaces", *Proc. of the First IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pp. 185-192, 2006.

[18] T. Butkiewicz, W. Dou, Z. Wartell, W. Ribarsky, R. Chang, "Multi-focused geospatial analysis using probes" *Proc IEEE Transactions on Visualization and Computer Graphics* (TVCG / InfoVis 2008), vol. 14, no 6, pp. 1165-1172, 2008.