Master research project
Direct-Touch Interaction for 3D Flow Visualization

Tijmen Klein

Contents

1 Introduction

Direct-touch devices are a popular way to intuitively interact with a computer. Touch-based interfaces are easy to understand and can give a feeling of being in control of the underlying data [IHC09]. Furthermore, touch-based interactions can perform better than traditional mouse input, for example in the selection of targets on a screen [KAD09] and by facilitating awareness in collaborative settings [HMDR08]. However, scientific visualization commonly deals with data that is defined in 3D space, whereas the input of a touchscreen is only two-dimensional. Consequently, an intuitive mapping from 2D touch input to 3D manipulations is required in order to control 3D scientific data in an interactive visualization.

This report presents a design study that deals with both the hardware and software design for an interactive application for the exploration of scientific flow visualizations (see Figure 1). It deals with the combination of 7 DOF navigation, 3 DOF cutting plane placement, 2 DOF drilling exploration, 5 DOF positioning of seeding particles in the dataset, the exploration of temporal aspects, and volumetric & isosurface based visualizations. By choosing smart postures and bi-manual actions, we can assure that these interactions do not interfere with each other. As testing data for these interactions we used 3D fluid flow simulations consisting of a scalar and vector field. The developed application is evaluated with fluid flow experts using an observational evaluation.

The combination of interaction techniques and the chosen example provide evidence that the direct manipulation of data using fluid interactions can be beneficial for the touch-based exploration of scientific data. The provided 2D view brings our application

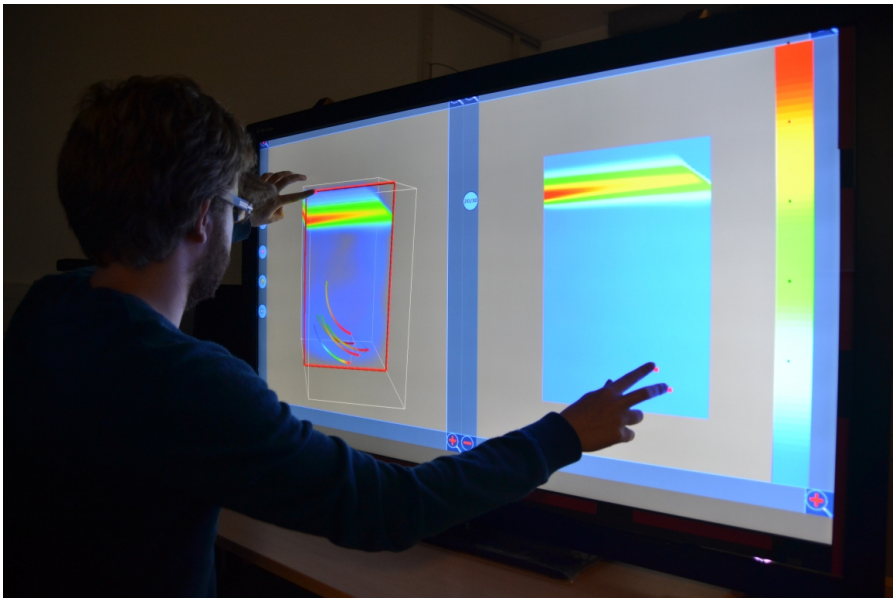

Figure 1: A person using the final application.

closer to traditional exploration tools. The evaluation showed that cooperation comes naturally with this hardware setup and interface design, which can stimulate researchers to collaborate on new data.

The remainder of this report starts with a brief evaluation of related work in Section 2. This is followed by an explanation of the problems and our design process in Section 3. Section 4 discusses the results of the projects and talks about the evaluation with the fluid flow experts. Finally, the report is concluded in Section 5, that also presents some suggestions for possible future work.

2 Related Work

Both interacting with scientific datasets and touch-based interactions have a broad history of scientific research. In this chapter we briefly discuss some interaction and visualization techniques that are related to the work that is presented in this report.

2.1 2D Touch-based Interaction

Touch-based interactions can perform better than traditional mouse-based input in the selection of elements on a screen [KAD09] and by facilitating awareness in collaborative settings [HMDR08]. This makes touch-based interactions very suitable for the exploration of scientific data, which is what happens in our application.

In the real world, the location and direction of a touch also affect the resulting interaction. Moving your hand while touching a 2D object (for example, a piece of paper on a table) near the center of its mass will result in a translation of the object (in either direction). The direction of the movement defines the interaction when a touch happens near the edge of a 2D object; moving parallel to the center of mass results in a translation, while a perpendicular movement will result in a rotation. This intuitive mapping can also be used on touch enabled devices to interact with 2D scenes. The Rotate'N Translate (RNT) [KCST05] technique implements this mapping and is the *de facto* standard for manipulating 2D objects with a single touch; it maps 2 degrees of freedom (x- and y-location of the touch) to 3 degrees of freedom in the interaction (x- and y-translation, rotation). This is not the only one-finger interaction-technique for 2D objects; TNT shows a promising technique to interact with 2D objects that is based on the techniques that people use to reorient sheets of paper on actual tabletops [LPS*06]. A user study showed that this TNT technique can perform faster than RNT and that TNT was the preferred technique of the participants. The downside of this approach is that it requires a dedicated sensor, either in the form of a finger sleeve or a block. This renders the TNT technique useless for projects that rely on only a touch-sensitive screen for the input (such as our own project).

These rotation and translation techniques do not need to be limited to a single touch. Translation and reorientation can be combined with zooming when there are two interaction points, which results in the the popular rotate-scale-translate (RST) technique that is the *de facto* standard for 2D manipulation with two touches [HCV*06,

RDH09]. The translation can be determined based on only the translation of the first point, while the object can be rotated by an angle formed by T_2, T_1' and T_2'; the scaling factor is determined by $|T_1'T_2'| : |T_1T_2|$ (see Figure 2). The popularity of RST has led to the situation that users are expecting RST-like behavior of all touch-based interfaces, this is an important reason to incorporate RST (or a similar technique) in an application.

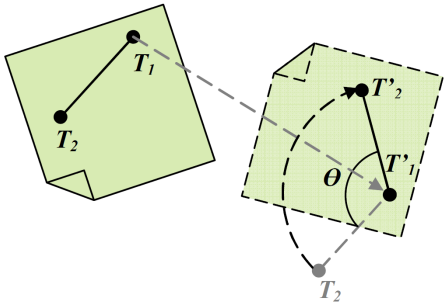

Figure 2: Two point rotation and translation, image from [HCV*06].

2.2 3D interaction

All previously mentioned techniques are designed to work with 2D data. However, the data used by scientific visualizations is commonly defined in 3D space, which means that a mapping from 2D touch input to 3D interactions is required.

It is possible to extend the 2D RST-manipulation into three dimensional space, while preserving its original semantics [RDH09]. This method allows users to directly manipulate 3D objects with three touch points, a method particularly well suited for manipulating planes in a 3D scene. This approach has it limitations, since not all fingers can be "sticky" at all times, this is a direct result of manipulating 3D objects on a 2D screen. This method also requires objects that can be touched and that constrain the interaction. When this is not possible, e. g. in the case of volumetric data; the displayed space can be manipulated by widgets such as Cohe et al.'s tBox [CDH11] (that uses physically plausible gestures for rotations) or Yu et al.'s FI3D [YSI*10] (that utilizes border widgets as spring-loaded modes in order to provide 7 DOF with two interaction points).

The traditional 2D touch interaction can be extended with the use of virtual reality. For example, a "World in Miniature" (WIM) can be projected above a touch-enabled table. A horizontal slice of this 3D object can be shown on the touch-enabled table where all the interactions happen [CML*11]. These interactions can then affect the projected WIM above the table. Showing a WIM is useful when most interactions happen at a detailed (zoomed in) level while requiring an overview of the whole data.

Both *Touching Floating Objects in Projection-based Virtual Reality Environments* [VSB*10] and *2D Touching of 3D Stereoscopic Objects* [VSBH11] explain there is a problem to get haptic feedback when touching objects in stereoscopic 3D. Objects with a zero parallax can be touched, but objects with a positive parallax (behind the screen)

and negative parallax (in front of the screen) pose a problem. This problem is the worst for objects in front of the screen, since the user has to reach through the objects in order to touch the screen, which distorts the stereoscopic projection. Valkov et al. [VSB*10] propose a method where floating objects are moved towards the interactive surface when the user walks around, a human is usually not able to perceive this. Furthermore, humans do not detect small inconsistencies between the location of the object that they see and feel. The user is able to perceive passive haptic feedback if the objects are moved towards the zero parallax. Our project does not utilize stereoscopic projections, which limits the usefulness of using the zero parallax for passive haptic feedback. However, our hardware setup does support stereoscopic projections, which could enhance the project even further. Using passive haptic feedback in this situation could help ease the interactions of the user.

3 Concept and Realization

We have developed a system that allows the user to interactively explore a time dependent scientific dataset of flow simulations. In order to effectively interact with such a dataset, several interaction techniques need to be combined. One of the challenges is to find interaction techniques that work well for these specific tasks. Another problem is to effectively combine these techniques so that they work together without interfering each other.

3.1 Hardware

Our hardware design is based around a 1920×1080, 55 inch, tiltable display, that can easily accommodate two people working at the same time. A PQLabs Multi-Touch G^3 Plus overlay is used to provide the touch events. This overlay consists of a glass plate and a frame containing an array of IR emitters and receivers that is capable of detecting up to 32 simultaneous touches. The overlay produces TUIO events for the touches on the screen [KBBC05]. The open TUIO framework provides a protocol and API for touch-based surfaces, and uses the Open Sound Control (OSC) format to encode its data and send it over UDP. Using TUIO allows for relative easy porting of our application to other hardware setups. The combination of the screen and the touch overlay is mounted in a stand that allows for easy adjustments in both height and angle (see Figure 3).

There were some issues with the hardware setup that we used. First of all, the display itself can be used to show 3D content using IR-based shutter glasses. This, however, posed problems with the rest of the setup. The IR emitter for these glasses is located on the frame of the screen, which resulted in two complications. If we would properly mount the PQLabs overlay on the screen, then the overlay would block the emitter. Lowering the overlay slightly, so that the emitter would be visible again, results in the emitter being behind the glass over the overlay; this resulted in serious problems with the synchronization of the shutter glasses. The use of stereoscopy could possibly enhance the application even more, but due to these hardware constraints we have not been

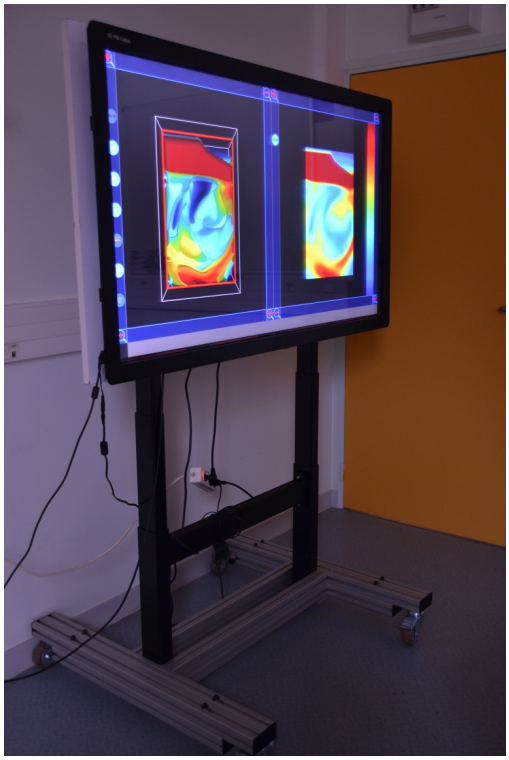

Figure 3: The final setup of the physical display.

able to test this thoroughly. There are a number of ways to solve this synchronization problem, but all of them require other hardware. One option would be to use a screen that employs radio frequency (RF)-based glasses. Another possibility would be a screen that accepts and displays a 120Hz (60Hz for both eyes) signal, and use a pair of external glasses that can directly be synchronized with the video card of the computer that is running the application. Even another option would be to use a screen that can somehow expose its synchronization signal, so that an external device can be utilized for sending the signal for the glasses. At the time of buying the screen, this problem with the screen was not yet known. Therefore, screens that accept 120Hz signals or allow external synchronization were not considered advantageous over screens with native 3D support and internal synchronization.

The second problem of the setup had to do with the processing lag of the touch overlay. This problem needs to be solved in order to make the final system usable. We were able to partially reduce the lag by changing the parameters of the driver of the touch overlay. This driver uses a running average of the touches of some time ago, which results in more stable (less jumpy) output points. However, these average points do not need to be close to the last known point, there can be a distance between the current touch points and the average points. This distance can be reduced by using fewer history points for the average, which means that some accuracy is traded for speed. This

parameter can be modified in a configuration file that is provided by PQ labs. The main solution for this problem, however, was to always display the points on the screen that the system currently detects. This provides the user with visual feedback, so that they are aware of how the input is being processed. This issue of lag will diminish over time when the quality of touch overlays will increase, and the price of the setup will decrease.

3.2 Flow Simulation and Visualization

The temporal flow data consists of vector fields and scalar fields that are calculated at discrete time steps.

The scalar field (Figure 4) in this particular dataset is a Finite Time Lyapunov Exponent (FTLE) field, which represents the speed of divergence of the flow [HS11]. These FTLE fields can help to enlighten certain aspects of physical flow, the peaks of a FTLE field represent some Lagrangian Coherent Structures (LCS). A LCS acts as a frontier in the flow, separating areas of particles with different behavior, and these particles are not able to cross the LCS frontier [HS11]. This can, for example, help to provide information about how the mixing in a physical flow happens; which is a relevant problem in industrial configuration. It helps to determine if all components of a product are well mixed.

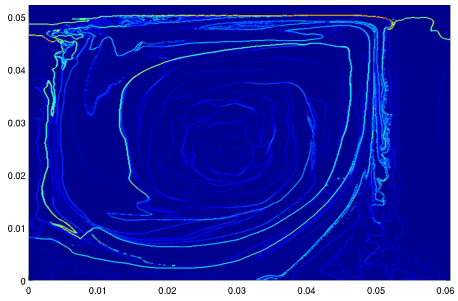

Figure 4: Slice of the scalar FTLE field in a traditional visualization tool.

We consulted experts in the field in order to find out what they need to do to explore and understand such a dataset. In order to do basic exploration, some traditional navigations actions (rotation, translation, zooming) are required. Furthermore, an arbitrary cutting plane is required in order to limit the dataset. Such a cutting plane can, at the same time, be used to provide a 2D view (or slice) of the data, which can be useful to compare against traditional flow simulations. An important technique to explore the vector part of flow simulations is to insert particles and trace these particles in the stream. This can be useful in a number of ways. First, inserting a group of particles close together can help to understand the divergence of the flow, which is useful when searching for the LCS frontiers (see Figure 5). Second, inserting particles in a larger region can aid to perceive the global characteristics of the flow and finding interesting subregions for more detailed investigations.

Isosurfaces are frequently used to inspect the FTLE scalar field of flow simulations.

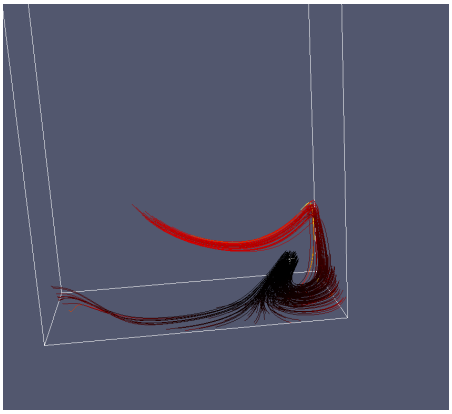

Figure 5: Tracing particles in order to discover the divergence in fluid simulation.

Such an isosurface visualization can be paired with a semi-transparent volumetric scalar visualization in order to get the best of both worlds (see Figure 6). Some kind of probing can help when the user wants to examine scalar values at specific locations in the 3D space to further understand the data. Finally, the temporal aspect of the data needs to be explored since the vector data and scalar data are time dependent.

Before working further on the project, we chose a framework to use for the scientific visualization. Our collaborating experts were used to ParaView [SKL], but ParaView's interface heavily relies on standard mouse-based input in combination with traditional menu structures. Therefore, we decided to base our development on the Visualization Toolkit (VTK), the underlying visualization framework of ParaView [SAH00]. VTK allows us to directly access the basic visualization techniques that have the same "look and feel" as ParaView without forcing us to use ParaView's interaction paradigms.

The fact that ParaView uses VTK for its underlying structures and visualizations gives us another advantage, it allows us to couple our VTK-based application with external ParaView instances. ParaView is a commonly used visualization tool in Cave [Bry96] environments. Our tool sends scene and interaction information using OSC. This information can be picked up by a ParaView plug-in, allowing our VTK-application to synchronize with the ParaView instance(s) (in a one-way fashion). Currently, not all available information is send, since no suitable ParaView plug-in has been implemented yet.

The explorations of scientific datasets are often not done alone, but in a group of two to three persons. Therefore, it is useful to provide a split view interface, so that two (or more) views can be interacted with independently. One view can show the isosurfaces while the other focuses on the trajectories of the injected particles. One view could also be utilized as a 2D view of the cutting plane, since these 2D views can provide relevant information to the domain experts. Using the split views also allows us to combine different visualization techniques without getting too much clutter on the screen.

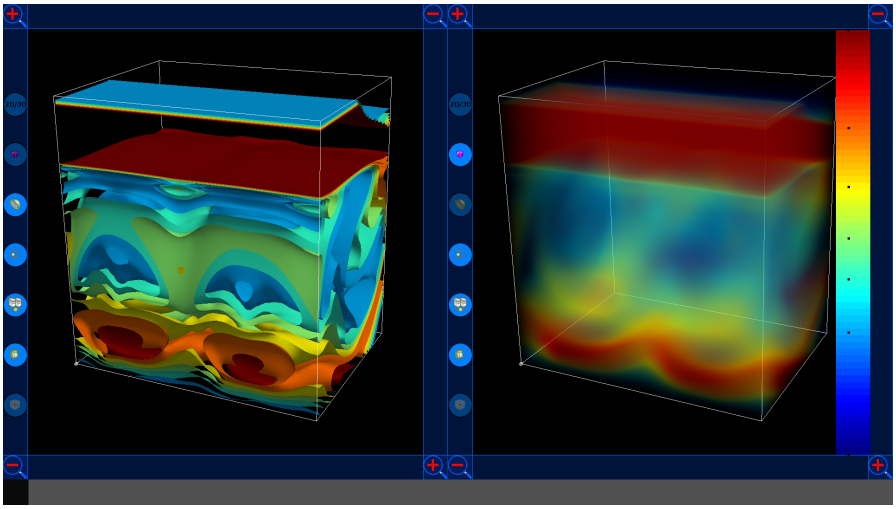

Figure 6: A screenshot of the final interface with two 3D views, showing both the isosurface and volumetric visualizations.

3.3 Interaction Techniques

Several interaction techniques are needed in order to fulfill the exploration requirements mentioned earlier. One of the challenges here is to effectively combine multiple intuitive interactions that do not interfere with each other, while minimizing the amount of modes.

Our basic rotation, translation, and zooming interactions are inspired by FI3D [YSI*10], which uses a frame on the side for spring-loaded actions. The behavior of our FI3D widget is equal to their second case study; a single touch results in a trackball rotation around the x- and y-axes, while a single touch that starts in the frame and move inwards results in a translations. Touching the frame and moving parallel to the frame will result in a rotation around the z-axis. One-finger zooming can happen using the corners of the frame, where two of the corners provide zoom-in functionality and the other two corners provide zoom-out functionality. These simplistic one-finger interactions are reinforced by two-finger RST interaction, that offers translation, rotation around the z-axis, and zooming; a total of 4 DOF output with 2 DOF input.

Another important interaction is the modification of the cutting plane. Modifying a cutting plane with an arbitrary normal can be a cumbersome task with a mouse. We chose to incorporate bi-manual three finger control (inspired by Reisman et al.'s screen-space technique [RDH09]) for these interactions (see Figure 7), so that they can remain modeless and do not interfere with the one- and two-finger interactions that are described above. In order to rotate the cutting plane we need a rotation axis and an angle. The first two touches define the rotation axis, and have to lie on the displayed plane in order to switch to the plane-rotation mode from RST-mode. The two 2D points on the screen are mapped onto two 3D points that lie on the plane that is currently displayed. The line between these 2 points serves as our axis of rotation. The third finger is used to move across the screen, the traveled distance is used as the angle for the rotation. The

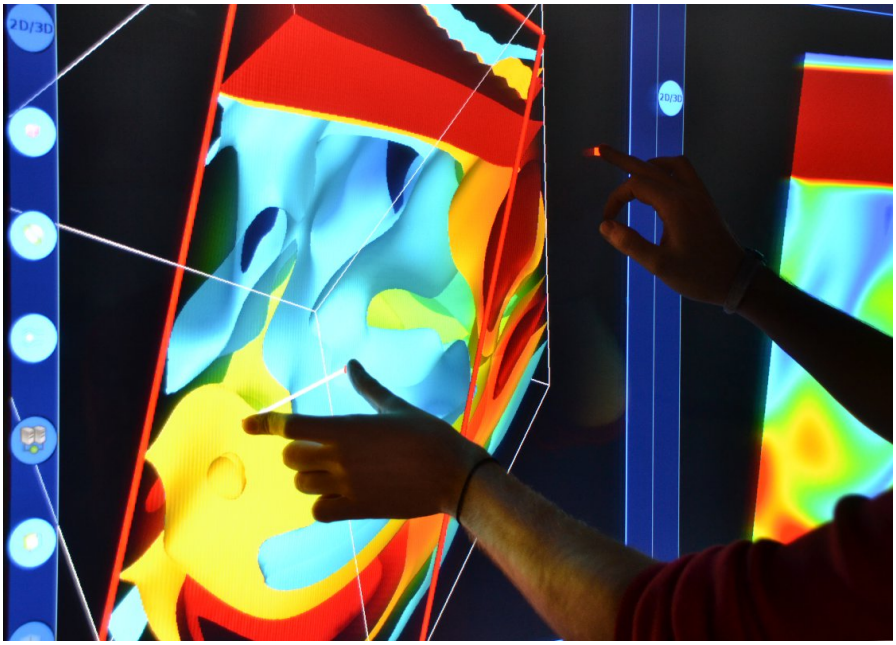

Figure 7: Cutting plane interaction with bi-manual control.

second interaction that can happen with the cutting plane is moving it along its normal. In order to do this, the user places the first two touches on the cutting plane, and used a third touch to make a movement that starts in one of the segments of the frame. The length of this movement is used as the length of the translation along the normal of the plane.

A cutting plane can be used to create an arbitrary 2D slice of the 3D dataset. This 2D slice is normally shown in the second view of the application. While this 2D slice can help to understand the dataset, it also serves a different purpose within our application; it is used for the insertion of particles that are traced throughout the vector field. Positioning a particle in 3D space with a 2D touch input is non-trivial and often unintuitive. Using the 2D slice as a seeding plane makes the positioning of particles more intuitive. When the user touches the 2D slice, the touch position is mapped to a 3D position that lies on the plane. This 3D position is then used as the seeding position for the particle tracer. Several particles are inserted in a very small sphere that is centered on this picked position.

The picking of the particle position happens in one view, while the resulting traces are visible in the other; this ensures that the fingers do not occlude the view of the result. Extending this particle position technique to multiple touches increases the exploration possibilities. When the user places two touches onto the 2D view, a line is positioned between the two mapped 3D positions. Seeding points are then placed along this line, which makes it easier to understand the differences of particle behavior along a specific axis (see Figure 8). A larger sphere is used to seed particles when the user places three

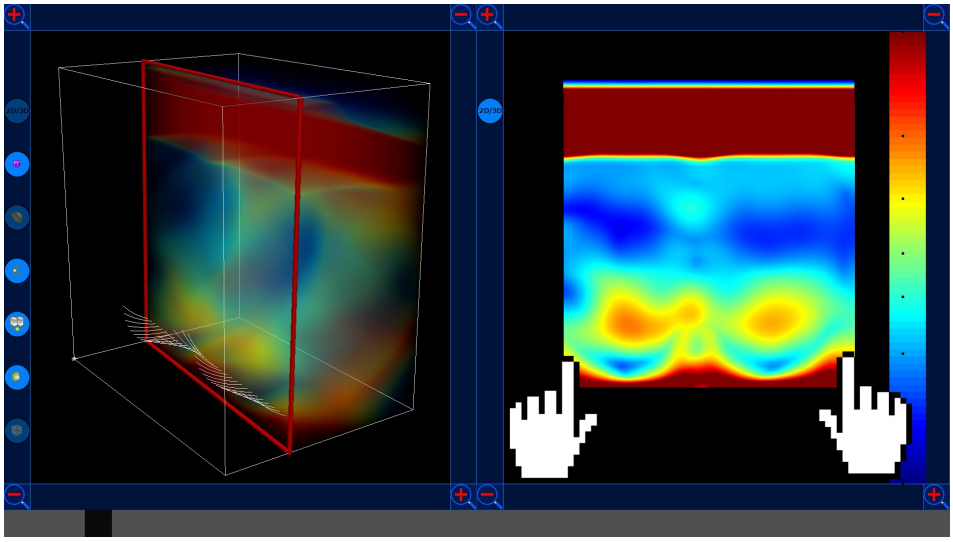

Figure 8: Seeding particles using two hands in the 2D view.

or more touches on the 2D view. The distance between the different touches define the size of the sphere, so that the user is able to interactively increase and decrease the sphere. Seeding points are then randomly positioned in this sphere, which makes it easy to explore larger parts of the dataset at once. These three different techniques for positioning particles allow the user to explore the data in a broad scale.

As mentioned above, our application has the ability to seed points on a line or inside a sphere, which means that the number of used seeding points is not fixed. In order to modify this number of points, we use the frame that is supplied by the FI3D technique. When the user moves along the frame in the 2D view (the same interaction that is used for the z-axis rotation in the 3D view), the number of seeding points is increased or decreased. This action not only adjust the amount of seeding particles, but also modifies the visualization of the traced lines. When the amount of seeding points goes below a threshold, the lines are changed into very thin ribbons. Ribbons can provide more context than lines, e. g. by encoding the vorticity of a fluid flow, something that is not possible with lines. However, it is hard to see this extra context where the number of ribbons becomes very large; lines will provide a better overview in this case. Decreasing the amount of seeding points even further will result in a linear increase of the thickness of the ribbons, until one thick ribbon remains.

Adjusting the isovalues happens through a dedicated widget on the right of the screen, that is based on a scalar bar. Positioning a finger in this widget will either place a new isosurface at this position, or pick up an existing isosurface in case one is close to the touched position. While the finger is down, it can be moved to (re)adjust the corresponding isovalue. This results in live updates of the rendered isosurfaces, so that the user can immediately see the results of the interaction.

A probing interaction can be started using one of the dedicated spring-loaded buttons

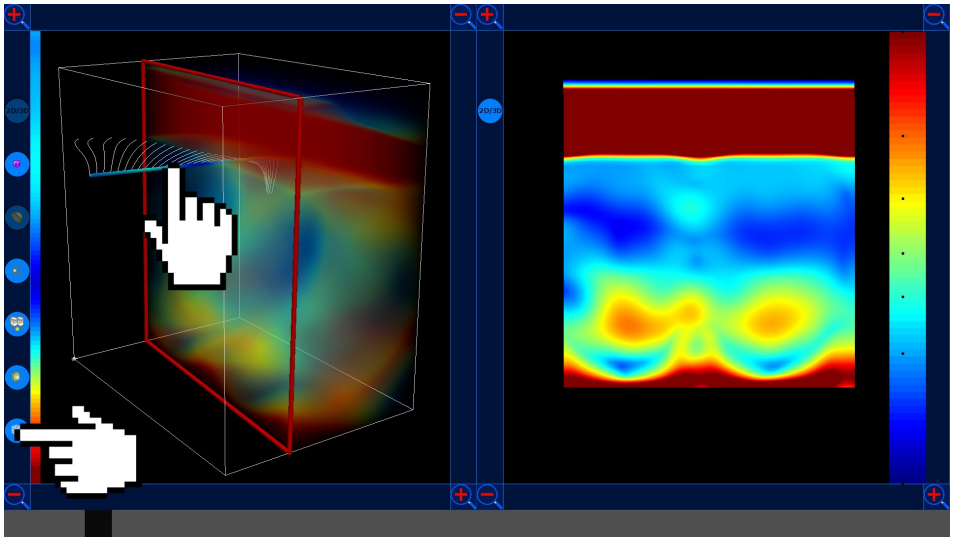

Figure 9: The probing interaction, using two hands.

in the frame on the left. One option is to start this interaction with one hand, and then move this touch into the data; when the touch is released the interaction stops. Another possibility is to touch the probing button with the non-dominant hand, and do the actual probing with the other hand (see Figure 9). While the dominant hand is probing, the non-dominant hand can release the probing button. During this interaction, a scalar bar (on the left) and a tube (in the data) appear. These show the scalar values that are perpendicular to the cutting plane on the position of the probing touch, like a drilling core going through the earth. This interaction helps to quickly discover specific scalar aspects of the data. During the development, the probing initially happened along the view line. However, due to feedback of our collaborating experts this was changed into a probing line perpendicular to the cutting plane. This has the advantage that the probed values can not only be shown in the scalar bar on the left, but also can be rendered as a texture on the probing tube in the 3D view.

4 Results and Evaluation

In addition to the results that we got throughout the process of the participatory design of our application, we also liked to get a better understanding of the quality of the application towards the end of the project. For this we conducted an observational study using five domain experts in the field of fluid mechanics. This evaluation session provided interesting results on our application and some possible ideas for expanding and improving it even further.

All results are acquired using the dataset that is used for the images throughout this report (unless specified otherwise). The application is able to run at interactive rates (60 FPS or higher) with this specific data on a modern workstation computer.

4.1 Evaluation study

The observational evaluation session happened with a total five fluid mechanics experts (all male, 27-57 years, median 43 years), of which two are the collaborating experts of the participatory design process. The other three experts were completely new to the application. The experts were split in a total of three groups, where two groups (G1, G2) contained two persons and the last group (G3) contained only a single person. The collaborating experts were in G1.

All sessions took between one and two hours in total and followed the same format. First, the participants were informed about the project and the application. Then, the application was started and a simple dataset was loaded into it. This dataset served as an example in order for the participants to get used to the application. A small tutorial followed that briefly explained all individual features and interactions of the application, the participants could try all interactions and ask questions about it. When the tutorial was done, the real dataset (that is used in the examples throughout this report) was loaded, all participants were familiar with this kind of data. The participants were asked to look for unexpected or otherwise interesting aspects of the data and were encouraged to 'think-out loud', so that the experimenter could know what their intentions were. The experimenter would sometimes remind the participants about certain functionalities in order to minimize the learning effects of the interface. This was followed by a semi-structured interview that discussed their findings with the interface, and where we tried to compare our touch-based application with more traditional fluid flow exploration and analysis tools. This resulted in very helpful feedback that is discussed in Section 4.2. All sessions ended with a small questionnaire to find out more information about the demographics, background, experiences with touch-based devices and exploratory visualizations of fluid flow data; and to acquire numeric feedback (using 5-point Likert scales) about certain specific aspects about the presented application.

4.2 Results

The evaluation sessions described in Section 4.1 gave interesting feedback and results for our developed application. These results can be grouped in four categories: small bugs and usability issues which should be relatively easy to fix, more severe issues that are related to the (software) design of our application, suggestions for new features that would further improve the application, and implications for other similar tools. The found results are discussed in this order.

4.2.1 Smaller usability issues

Some other small issues that we found had to do with the current hardware setup of our application. Some of the participants had troubles with the touch sensing overlay, when they were hovering just above the screen with their fingers, it would sometimes be detected as touches. This is an inherent consequence of using an IR-based touch sensing approach. Since the prices of more advanced touch sensing overlays are continuously

decreasing, this should not be a problem in the near future.

4.2.2 Design issues

We also found a number of more severe usability issues that have to do with the design of the application. The interactions that are required to adjust the cutting plane resulted in the most problems, one of the participants (from G2) found the interactions to be unintuitive, and therefore also had severe problems to precisely position the cutting plane in a way that he desired. Two other participants (other member of G2, G3) indicated that the cutting plane interaction would take some time to learn in order to use it efficiently, these two participants were however capable of interacting with the cutting plane to get the desired configurations. The two participants that also cooperated in the participatory design process (G1) were able to manipulate the cutting plane in a better way, which helps to confirm the indications that learning might help to improve the cutting plane interactions. However, these two participants also still indicated that the cutting plane interaction was the most severe usability issue of the current implementation.

Some interactions were thought of as not being precise enough, this particularly applies to the cutting plane interaction again. The participants found it to be very hard to align the cutting plane to the bounding box that contains the data; while some form of "snapping" could help in this specific case, it would not make the whole interaction more precise. On the other hand, some interactions were considered to be very precise, especially the streamline seeding. Also, the distance between two data points was larger than the size of a pixel on the screen, and these kind of exploratory visualizations have a more "qualitative" character according to the participants which requires less precision. The combination of this positive and negative feedback led to the conclusion that the application overall did not feel imprecise.

One way to improve the precision of the interactions, would be to add an option to isolate certain interactions. Some of the postures trigger multiple interactions simultaneously, e.g. initiating a cutting plane interaction starts by placing two fingers, which also results in a RST-interaction. Such a cutting plane interaction will therefore always also lead to some small other changes in the view, this led to annoyance with some of the participants. One participant (G3) suggested adding system-controlled modes in order to isolate interactions, this could be explained with the participants experience with traditional single-user tools that rely on keyboard and mouse input. While this indeed is a way to isolate interactions, it would also collide with the collaborative aspect of the application. Therefore this is not a viable solution in the case of our application. Using spring-loaded modes to isolate certain interactions would be solution that does not collide with the collaborative aspect.

4.2.3 New features

Next to the mentioned issues, the participants also mentioned some missing features that are required to make the application more useful in practice. Some of these features are

related to numerical measurements and other mathematical functionality, while others are related to the interface and visualization aspect.

When our application is compared to the traditional tools that are used for these visualizations and explorations (ParaView, Matlab, Tecplot), the participants mention that these tools have better support for the mathematical exploration aspects, precise interactions, and the possibility to reproduce quantitative specification of placements in space and time (e.g. the views, seeding positions). It was even suggested to add plug-in support to add specific (mathematical) functionality. While this may be a very good idea to make the application more practical, it is beyond the scope of the current project.

In order to use the application in a more practical way it also required to have more specific data information in the form of numerical read-outs at specific locations, some form of axis labels, small 3D coordinate system axes. This should be accompanied with the ability to load (and switch between) different datasets and toggle different color scales, linear vs. logarithmic scaling, adjustable transparency values for both the volumetric and isovalue visualization. It was even suggested to keep a history of a whole interaction session, so that researchers would be able to fully reproduce an entire exploration session. This could be used in order to retrieve images of specific views that were encountered during the exploration. At the same time this functionality could be used to replay a list of interactions on a other set of data, in order to compare two data sets. A slightly similar request was to add certain default views and/or explorations configurations, so that these can be used on different sets of data.

Some new features would expand functionalities that are already embraced in the current design. The drilling interaction that currently happens in the 3D view could also happen in the 2D view, a functionality that would be desired by the participants. The selection of isovalues currently happens in a dedicated widget, it was suggested to add the ability to use a probed value from the 2D view as a new isovalue. Another requested feature concerning the 2D view, was the ability to apply transformations on it: zooming, panning and flipping; this could help the user with the mental coordination in respect to the 3D view of the data. This coordination problem could perhaps also be solved by applying better default transformations on the 2D view, these default transformations would depend on the current transformations that are applied on the 3D view.

Quite a few of the requested features are related to the streamlines and particle seeding. Showing the flow direction of the streamlines could help the users to get an even better understanding of the underlying data. Letting the user adjust the length of the streamlines falls in the same category, and the same applies for the request of tracing the streamlines backward in time. The participants suggested that being able to place locations that continuously emit particles/seeds would improve the exploration. This would be accompanied by some form of grouping or labeling of the particles, so that particles from different origins (either in space or time) can be distinguished. One thing that was noted during the evaluation was that the large seeding regions were very rarely actively used.

4.2.4 Implications for other tools

During the evaluation session we found that our application has some major advantages over traditional exploration tools. The ability to directly manipulate the data using fluid interactions with quick feedback from the interface is essential for effectively exploring new data. A second advantage is the interwoven connection between the 3D and 2D view, that helps to narrow the gap between our application and more traditional exploration tools. The support for natural collaboration is the last major advantage of the designed tool.

The first mentioned advantage, direct manipulation, is necessary in order to intuitively discover interesting aspects in new and unknown data. Since our tool does not require the user to build dedicated visualization pipelines or define mathematical views (as in, for example, Matlab), it is easy to quickly load new data, which fits really well with the explorative aspect of the application. However, as new functionalities and features will be added it will become harder to keep the interface clean and simple. An implication of this is that the user will need to learn certain interactions in order to use the application and its interface to their full potential. The participants indicated that this should serve no problem, which can be explained by the fact that the traditional tools even have a relative steep learning curve. If a touch-based interface allows the user to intuitively do the basic interaction and stimulates the user to discover more advanced features during the exploration session, then the user will be able to learn about the application "on the go".

The provided 2D views relate well with the more traditional exploration tools that were used by the participants, since some of these tools heavily rely on 2D views. This make the integration of 2D views in a modern touch-based exploration tool essential, if it is to be used in practice. Since our application also utilizes the 2D view for specific interactions (e. g. the positioning of seeds points in the data) it serves more purposes than only a 2D slice of the 3D data.

The cooperative aspect of a touch-based interface is important during the exploration of new data, which is the reason that during the participatory design process of our tool we focused on accommodating at least two persons. The two views provided by the interface can be used by a single person to focus on different aspects of the data, but at the same time two users can both use a single view as their "personal" workspace. Both groups (G1, G2) found that collaborating with the interface was both natural and enjoyable, an example of two persons collaborating can be seen in Figure 10. The single person (G3) mentioned (without being asked about it) that he would really like to collaborate with other people using this application. All participants thought that collaborating with two persons would be ideal, but that it might also work with three persons. The 55 inch screen and two view interface lends itself naturally, changing this could perhaps favor a three person collaborative setup. During the evaluation session it was remarkable how the cooperation styles between G1 and G2 really differentiated. While the persons in G1 were really interacting at the same time, the persons in G2 basically took turns. This can be explained by the fact the G1 already had experience with the application, which made them better accustomed to the interactions. The

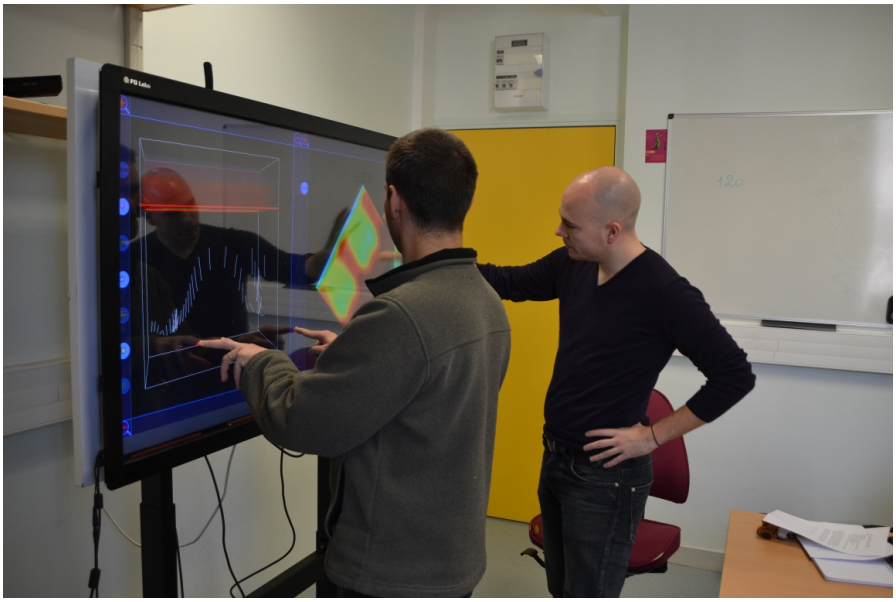

Figure 10: Two researchers collaborating using our application.

persons in G2 indicated that taking turns would ensure that their interaction did not interfere with each other.

These three mentioned advantages of a touch-based exploration tool for scientific data can serve as guidelines for other tools that work with scientific data.

5 Conclusion and Future Work

This report presented a direct-touch application for the exploration of 3D scientific vector and scalar data. It uses multiple non-overlapping gestures to provide a wide variety of possible interactions to the user. By providing a 2D view we narrow the gap with traditional tools, while at the same time the 2D slice is used in a new way to define positions in 3D space. In the 3D view the slice serves as a basis for the drilling interaction, the 2D view of this slice is used to precisely position seeding particles in the dataset.

During our evaluation session we found that the experts like the explorative aspects of the data, one participants even wants to use this tool to explore his own data. They indicated that the tool can be used to quickly get familiar with new and unknown data sets, which can help to effectively do mathematical analysis of the data in a more traditional tool afterwards.

While our application generally worked well, there is room for improvements. We found a number of smaller bugs, but also some issues that will require further research. The plane interaction was found to be the most problematic, and there is a demand for more precise and isolated interactions. New features will need to be added to make

the application more useful for real world situations. Nevertheless the participants of the evaluation were able to use the application in its current version without too much problems, and learning the more complex interactions such as the plane manipulation can help to use the application to its full potential.

We found three important results that can help other touch-based applications for exploration in their design. First of all, the direct manipulation of the data using fluid interactions with quick feedback from the interface is essential for effectively exploring new data. Providing a 2D view can help in situations where the traditional applications of a user heavily rely on 2D views, while at the same time it can function to precisely pick positions in 3D using only 2D input. And finally, the cooperative aspects of a touch-based application make it more suitable for the exploration of unknown data where experts need to collaborate.

This projects also opens the door for some other and new research. It would be interesting to look for precise and intuitive interactions to manipulate a plane in 3D space using only 2D input. While our hardware setup has support for stereoscopy, we have not actively exploited this during the current research. Interacting with 3D data that can be seen in stereoscopic 3D using only 2D input provides new interesting problems, for example concerning the parallax. The hardware could also be used for different setups, e. g. a table or drafting table setup, to see how this compares to a wall setup. Writing a ParaView plug-in that allows for synchronization with our application would mean that Cave environments can be controlled using our touch-based application. The application that has been developed for this project suits itself to be extended for more practical use of research of scientific data.

References

[Bry96] Bryson S.: Virtual reality in scientific visualization. *Commun. ACM 39*, 5 (May 1996), 62–71.

[CDH11] Cohé A., Dècle F., Hachet M.: tbox: a 3d transformation widget designed for touch-screens. In *Proceedings of the 2011 annual conference on Human factors in computing systems* (New York, NY, USA, 2011), CHI '11, ACM, pp. 3005–3008.

[CML*11] Coffey D., Malbraaten N., Le T., Borazjani I., Sotiropoulos F., Keefe D. F.: Slice wim: a multi-surface, multi-touch interface for overview+detail exploration of volume datasets in virtual reality. In *Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2011), I3D '11, ACM, pp. 191–198.

[HCV*06] Hancock M. S., Carpendale S., Vernier F. D., Wigdor D., Shen C.: Rotation and translation mechanisms for tabletop interaction. In *Proceedings of the First IEEE International Workshop on Horizontal Interactive Human-Computer Systems* (Washington, DC, USA, 2006), IEEE Computer Society, pp. 79–88.

[HMDR08] Hornecker E., Marshall P., Dalton N. S., Rogers Y.: Collaboration and interference: awareness with mice or touch input. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work* (New York, NY, USA, 2008), CSCW '08, ACM, pp. 167–176.

[HS11] Haller G., Sapsis T.: Lagrangian coherent structures and the smallest finite-time lyapunov exponent. *Chaos 21*, 2 (2011), 023115.

[IHC09] Isenberg T., Hinrichs U., Carpendale S.: Studying direct-touch interaction for 2d flow visualization. *Collaborative Visualization on Interactive Surfaces-CoVIS'09* (2009), 17–20.

[KAD09] Kin K., Agrawala M., DeRose T.: Determining the benefits of direct-touch, bimanual, and multifinger input on a multitouch workstation. In *Proceedings of Graphics Interface 2009* (Toronto, Ont., Canada, Canada, 2009), GI '09, Canadian Information Processing Society, pp. 119–124.

[KBBC05] Kaltenbrunner M., Bovermann T., Bencina R., Costanza E.: TUIO: A Protocol for Table-Top Tangible User Interfaces. In *Proc. Workshop on Gesture in Human-Computer Interaction and Simulation* (2005).

[KCST05] Kruger R., Carpendale S., Scott S. D., Tang A.: Fluid integration of rotation and translation. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 2005), CHI '05, ACM, pp. 601–610.

[LPS*06] Liu J., Pinelle D., Sallam S., Subramanian S., Gutwin C.: Tnt: improved rotation and translation on digital tables. In *Proceedings of Graphics Interface 2006* (Toronto, Ont., Canada, Canada, 2006), GI '06, Canadian Information Processing Society, pp. 25–32.

[RDH09] Reisman J. L., Davidson P. L., Han J. Y.: A screen-space formulation for 2d and 3d direct manipulation. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology* (New York, NY, USA, 2009), UIST '09, ACM, pp. 69–78.

[SAH00] Schroeder W. J., Avila L. S., Hoffman W.: Visualizing with VTK: A Tutorial. *IEEE Comput. Graph. Appl. 20*, 5 (Sept. 2000), 20–27.

[SKL] Sandia National Labs, Kitware Inc, Los Alamos National Labs: Paraview.

[VSB*10] Valkov D., Steinicke F., Bruder G., Hinrichs K. H., Schöning J., Daiber F., Krüger A.: Touching floating objects in projection-based virtual reality environments. In *Proceedings of Joint Virtual Reality Conference (JVRC 2010)* (2010), Eurographics, pp. 17–24.

[VSBH11] Valkov D., Steinicke F., Bruder G., Hinrichs K. H.: 2d touching of 3d stereoscopic objects. In *ACM Proceedings of CHI 2011 Conference on Human Factors in Computing Systems* (2011), ACM.

[YSI*10] Yu L., Svetachov P., Isenberg P., Everts M. H., Isenberg T.: Fi3d: Direct-touch interaction for the exploration of 3d scientific visualization spaces. *IEEE Transactions on Visualization and Computer Graphics 16* (2010), 1613–1622.