EG UK Theory and Practice of Computer Graphics (2006)
M. McDerby, L. Lever (Editors)

Flexible Interaction with Large Point-Based Datasets

A. Rosiuta and G. Reina and T. Ertl

Visualization and Interactive Systems Institute, University of Stuttgart, Germany

Abstract

We present an application for interacting with large, point-based datasets built with commodity off-the-shelf hardware. Our system works on ordinary workstations with mouse and keyboard interaction as well as on immersive multi-head VR systems with tracked input devices. This approach allows thermodynamics and astrophysics researchers to interactively navigate, filter and inspect large datasets that result from particle-based simulations as used in those areas. We make use of an adaptive, hierarchical rendering approach and extend its data structures to optimize the interaction performance to be usable with datasets of hundreds of thousands of particles. We prove the validity of our concepts with informal user studies both in the application area and the computer science context.

Categories and Subject Descriptors (according to ACM CCS): J.2 [Physical Sciences and Engineering]: Astronomy/Physics I.3.7 [Three-Dimensional Graphics and Realism]: Virtual RealityI.3.6 [Methodology and Techniques]: Interaction Techniques

1. Introduction

Simulation is one of the most important research tools and gaining even more importance as commodity hardware clusters are becoming cheaper to purchase and employ. The resulting datasets are the grand challenge for current visualization methods, as on the one hand the large amounts of data itself are quite difficult to manage, process, and render interactively. On the other hand, even when the performance of the visualization systems is sufficient, datasets with millions of data points are impossible to grasp by the human researcher. To overcome this problem known as visual overload, several approaches can be considered, for example a sensible abstraction from the single data points, or filtering, as defined by the visualization pipeline. Another feature needed alongside the visualization is picking – to be able to inspect attributes that either cannot be efficiently visualized because of their type or because there simply are too many attributes to be shown simultaneously, like the full phase space for thermodynamics simulations.

A subset of this problem domain consists of particle-based simulation. Datasets of this type are found, for example, in astrophysics, computational physics and thermodynamics. Our approach focuses on this particular subject and extends an existing visualization framework by an intuitive user interface. We provide mechanisms for selecting, filtering and inspecting data both for workstation use (i.e. mouse and keyboard interaction) and immersive use with stereographic projection and a commodity marker-based optical tracking system with a freely movable interaction device.

The rest of this document is structured as follows: Section 2 gives a short overview of our usage scenario and some previous work, Section 3 explains the concrete application domain. Section 4 describes our user interface and interaction possibilities as well as outlining the implementation. Section 5 shows some usage examples and the results of our user studies. Section 6 concludes the work and gives an outlook on further possibilities.

2. Problem and Related Work

Our scenario consists of a relatively large space filled with tens of thousands, and up to millions, relatively small and in parts extremely densely packed particles. Usually there is floating matter with low density in a large part of the space and a smaller number of, in some way, coagulated particles, or clusters (droplets, galaxies, etc. depending on the simulation domain). Our requirement is that the user have tools which allow them to single out clusters or other interesting parts of the dataset as well as for being able to pick single particles, if needed. The hardest problem in this scenario is clutter, which is even worse for the galaxy simulation visualization because the particles are blended and thus the user cannot easily distinguish overlapping objects. Therefore we have an additional uncertainty when it comes to selecting because the user first has to find out which of these objects in a certain region he is really interested in. A comprehensive taxonomy on selection mechanisms and operations has been presented in [Wil96], which has been the base for deciding the features for our user interface. Mouse-and-keyboard-based interaction is probably the most common overall for workstation scenarios, however in VR and AR environments there has been considerable research not only about methods but also about suitable input devices.

The basic method is pointing-and-clicking which has the benefits of being widely known and accepted. The results are precise and it is very easy to switch between several operations quickly and easily using different buttons and hotkeys. The corresponding approach for immersive environments is the laser pointer and its many improvements like the spot light [LG94] or IntenSelect [dHKP05]. These approaches try to compensate the motorical deficiencies of human beings by adding fuzzyness and/or a ranking method to decide which of the possible candidates the user really wanted to select. Such improvements take into consideration small and far away objects as well as clutter, however comparing the datasets used for testing to the simulation scenarios we have to cope with, one will note that there are several orders of magnitude more objects in a scene in our application. Other common metaphors used for VR/AR interaction that have some correspondence in the real world are, for example, pen and tablet [BBMP97] or variations like pointer and clipping plane [dRFK*05] and real hand-held devices [WDC99]. An example for a virtual device controlled by a completely different commodity peripheral is the virtual tricorder/space mouse combination [WG95].

For our approach we wanted to stick to the commodity flystick device, if only for economical reasons, so our contribution to interface design consists mainly in the combination of methods – a radial menu and laserpointer combination plus a 3D cursor – and metaphors, i.e. a darkened room and a candle.

3. Dataset Types and Scenarios

For now we focus on two fields of research for our datasets. One is multibody simulation from astrophysics, more specifically the simulations carried out by the VIRGO supercomputing consortium, which yield point-based datasets with several millions of particles. The challenge to visualize such datasets is met by making use of the hierarchical rendering approach proposed in [HE03], however to improve performance even further and reduce visual overload, we wanted the option to interactively trim the dataset for easier focusing on the interesting parts. For this purpose the different selection operations are very important to allow a simple drilling down into smaller areas. The other research field is thermodynamics, where datasets are smaller (tens to hundreds of thousands molecules, see Figure 1 for an example), but the focus is on inspecting the attributes of particles and slicing open clusters of particles. In this area researchers are just starting to make use of the cheap performance boost commodity clusters are offering, so dataset size is not a constant problem for now. Due to the smaller size, the latter datasets can be sensibly streamed to have time-based visualization, and selection and highlighting is going to give us the ability to easily track interesting particles visually. The visualization in this case is handled by the framework extensions discussed in [RE05].

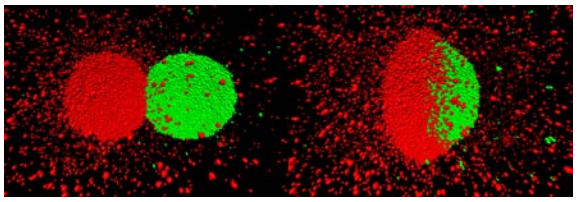


Figure 1: *An example of a thermodynamical simulation: Methane (red) and ethane (green) colliding at about 200m/s. The simulation takes place at 112K, so the methane is quickly evaporating.*

Since the framework we based our approach on renders the data adaptively, and tries to maintain a framerate of at least 10 fps, any filtering of the dataset results in the framework having more time to traverse, and render, the remaining points. Therefore such filtering effectively causes a refinement of the remaining regions if parts of the data had to be skipped as long as the whole dataset was visible.

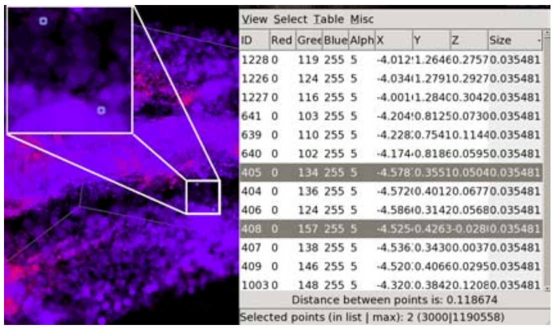
4. Features and Implementation

The user interface we have implemented has two principal modes, one for workplace use in conjunction with standard PC hardware and one for use in VR environments with stereo output and tracked interaction. The whole system is based on the concept of a two-tiered selection: a *temporary selection* and a *permanent selection*. This allows the user to create complex selection subspaces incrementally using the temporary set and to store satisfactory results in the permanent set without the risk of compromising the permanent selection with the next selection interaction. The storing of temporary sets is effected through boolean operations, so the user can add or subtract to or even intersect the temporary selection with the permanent selection. Other operations that can be performed on the data are:

- cropping to selection sets
- hiding of the temporary selection - the permanent selection cannot be hidden because it represents the result of all the filtering operations and thus must not disappear

- single-click toggling the hidden data to enable the user to re-check the context of the selected data without losing the selection subset.

Our user interface offers an additional window which contains the interaction menu and a table (see Figure 2). This table displays the attributes of all particles contained in the permanent selection, so the user can look up the exact values as they are output by the simulation, which is particularly helpful for particles that have more attributes than can readily be displayed by our currently available rendering modes, but are important for the understanding of the state of the simulation. The 3D view of the dataset is linked to the table, so the user can also select subsets of particles in the table which is reflected by bracketing the selected points in the 3D view by one bounding rectangle each.



ID	Red	Gree	Blue	Alph	X	Y	Z	Size
1228	0	119	255	5	-4.012	1.2646	0.2757	0.035481
1226	0	124	255	5	-4.034	1.2791	0.2927	0.035481
1227	0	116	255	5	-4.001	1.2840	0.3042	0.035481
641	0	103	255	5	-4.204	0.8125	0.0730	0.035481
639	0	110	255	5	-4.228	0.7541	0.1144	0.035481
640	0	102	255	5	-4.174	0.8186	0.0595	0.035481
405	0	134	255	5	-4.578	0.3551	0.0504	0.035481
404	0	136	255	5	-4.572	0.4012	0.0677	0.035481
406	0	124	255	5	-4.586	0.3142	0.0568	0.035481
408	0	157	255	5	-4.525	0.4263	-0.028	0.035481
407	0	138	255	5	-4.536	0.3430	0.0037	0.035481
409	0	146	255	5	-4.520	0.4066	0.0295	0.035481
1003	0	148	255	5	-4.320	0.3842	0.1208	0.035481

Distance between points is: 0.118674

Selected points (in list | max): 2 (3000|1190558)

Figure 2: *Selection in the table and linked points in the 3D view, highlighted by a small bounding box.*

Additionally, the user can interactively adjust the near and far planes of the GL context to coarsely select the spatial subvolume of the dataset he wants to interact with. Such cuboid 'slabs of interest' give the user another basic volumetric primitive by which he can specify the focus region he wants to interact with. An interesting slab can be "accepted"—marking all invisible points as invisible and resetting the planes to their default position for further interaction. The near and far planes' settings are visualized by rendering semi-transparent rectangles at the modified near and far planes. The rectangles have the size of the visible part of the old near plane. There is also a numerical display on which the z-axis positions of the viewport-parallel frustum planes are shown.

4.1. Workstation Interaction

When using keyboard and mouse for interaction, the keyboard can be used to gain access to almost all the functionality very quickly by using hotkeys. Less advanced users may choose to use the popup menu or the regular menu for triggering the interaction modes. The actual selection always happens by using the mouse. The user can perform a rectangular selection as well as paint with a brush with variable size as known from common graphics applications. Both selection types are projected indefinitely in depth based on the current view, so the result is always a frustum with an arbitrary profile. We chose not to limit the depth in this case because the astrophysics datasets have high overdraw and use blending, which makes it difficult to decide at which depth a point lives without rotating a dataset when no stereo output is available. The depth filtering can be easily performed by modifying the resulting frusta from different points of view. These actions create and modify the temporary selection. A virtual sphere metaphor is used to navigate in the dataset.

4.2. Tracked Interaction

Despite the several options from recent literature to implement interaction with VR environments, economical criteria made us search for a solution based on the optical tracking system already installed at our institute, which consists of four ARTtrack1 cameras. This infrared-based optical tracking system is capable of tracking the position and orientation of multiple targets, or *bodies*, which consist of several markers, and are configured to be worn on glasses or hands. Since these targets do not support any spatial reconfiguration, gestures or finger movements cannot be distinguished. To have some kind of triggering option, we used a so-called *flystick* (see Figure 3), which is basically a marker-equipped joystick handle with several buttons and radio-based communication with the tracking server. These triggers are used to actuate grabbing of the dataset as well as activation of UI elements, for example.

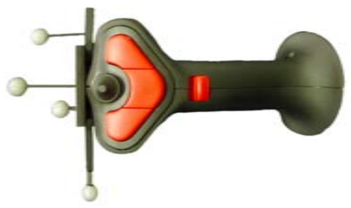


Figure 3: *The flystick used for interaction has several buttons for triggering interactions and reflecting markers for tracking with IR cameras.*

4.3. VR Interaction

When using the flystick, the main difference to mouse/keyboard is the visible 3D cursor. The user can press a button on the flystick to make the current cursor position a pivot point around which the dataset can be rotated or in relation to which the dataset can be translated. To decouple the selection and navigation precision from physiological precision of the users' movements and spare the user from having to run around in front of the powerwall, the cursor movement is not absolutely mapped from the flystick movement in front of the powerwall. Instead, the user can fixate the 3D cursor by pressing a button, and then reposition the flystick to a more comfortable position, thus emulating what for a desktop user is the lifting and repositioning of the mouse.

The position of the 3D cursor is visualized by a half-transparent sphere in the background and a real "cursor" in the foreground. Additionally, when the cursor intersects with points, these are colored differently to ease the spatial localization. In this mode, the brush selection is not projected, since the user can reposition the cursor in all three dimensions by direct interaction. Thus all the highlighted points are added to the temporary selection when the according button on the flystick is pressed.

The table with its detailed display of particle attributes is also available in the VR mode and projected onto the focus plane. However, since it is quite difficult to accurately grab the scrollbar or the corresponding buttons when interacting with the flystick and thus resorting to one's gross motor skills, the list of selected particles is scrolled differently: to allow easy relative and also absolute scrolling, a modifier key and pointing direction are used. When pointing into the list window, scrolling is relative, outside to the right (where the visible scrollbar lies) scrolling is absolute. Scrolling itself works by pointing up or down relatively to a "horizon" in the middle of the screen. Simple selection within the table is possible through pointing as well and "SHIFT" and "CTRL" are mapped to flystick buttons to allow for the selection of multiple items.

4.4. A 3D Candle

In this mode, the cursor disappears and a halo of the same size as the 3D brush is used to select points for rendering, as if the only source of light in the visualization were at the position of the cursor and everything beyond the halo were submerged in the dark. That way, all the processing power can be used for a small region of the dataset, which ensures maximum detail in the focus region while completely neglecting the context. This metaphor was chosen for two reasons: on the one hand, it allows for extremely high refinement and high performance since only a very limited part of the dataset has to be rendered, and on the other hand it reduces the high overdraw and the ensuing visual overload for the user drastically, especially in snapshots from cosmological simulations with several tens of millions of particles. Since this is only a metaphor we did not change the lighting conditions of the rendered scene at all, because (at least in case of the molecule rendering) back-lit primitives offer very low contrast and no sense of spatial extent.

4.5. Measures to Avoid Fatigue

A combination between pointing to the powerwall (for the selection list) and a 3D cursor is used to faciliate interaction. The scene can be "grabbed", allowing free movement and rotation. In a first attempt to provide a mode selection in this virtual reality scenario, a popup menu on the focus plane was used for user input. This proved to be unsuitable as mapping the flystick interaction to a relative mouse pointer movement on the focus plane was precise, but caused tremendous fatigue, causing the users' hands to slightly drift to the lower right after some menu interactions. Although it would be easily possible to recalibrate the hand's position to a less exertive position this feature would need to rely on some kind of fatigue-percepting algorithm. In a second attempt, the mouse cursor was positioned at the intersection point between a virtual ray cast along the flystick's z-axis and the focus plane. This removed the fatigue source as the user automatically relaxed his hand position when switching from the relative cursor movement manner to the pointing approach. However, this solution caused a loss of precision stemming from the combination of motor precision and tracking precision - small angle changes when standing two meters from a target cause considerable positional imprecision. The accidental selection of the wrong menu item was the most frequent result. To solve this problem, we used larger menu elements and a menu type that fits the directional pointing method far better: the radial menu.

4.6. The Radial Menu

The classical radial menu, which can be seen in applications with a mouse interface, e.g. Maya, was extended to a full-screen approach. Thus, the whole display area is used as a projection target allowing menu items to be equally larger in both dimensions. All menu items are shown at once. The amount of information may seem very large, however by highlighting only parts of the hierarchical structure, the user can concentrate on suitable amounts of information (5-9 highlighted sub menu items). All menu items are displayed as icons with a tooltip showing a short description of the item at the cursor's position when hovering over it. All submenu entries are only outlined by default, the corresponding icons are displayed when the corresponding menu is hovered over (see Figure 4). Another advantage of this approach is the that the user can more easily memorize the position of menu entries, and once a user learns where to find an entry he can instantly get there without having to follow a trail leading him there as a normal menu or even a cascading radial menu would require.

4.7. Implementation

We integrated our interaction component into the framework implemented at our institute by M. Hopf [HE03]. Its main advantages consist in the hierarchically kept data and

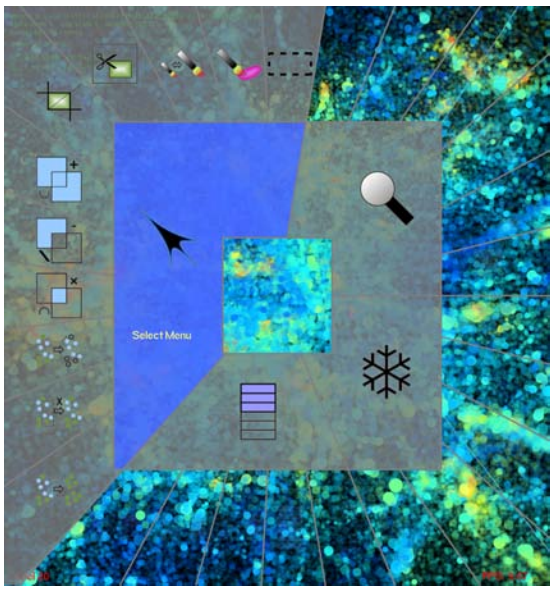


Figure 4: *The radial menu, blended over the dataset. Only submenu items in the selected menu are visible to reduce clutter.*

the possibility to adaptively adjust the recursion depth when rendering the points, thus guaranteeing a minimal framerate to allow interactive navigation of multi-million particle datasets. Furthermore it is also capable of using several machines for screen-space subdivided single-eye rendering, so it can drive a 4-projector passive stereo powerwall by using 4 output nodes and one head node for the interaction. Each node has access to the whole dataset and synchronization is achieved by only distributing the current rendering mode and view states, which is a sufficiently small amount of data for transmission over the ethernet. We use the head node to process the tracking system output and distribute the interaction with slight extensions to the existing communication protocol.

The data hierarchy consists of several recursive levels of spatial clusters, and the lowest level contains the actual particles. Each cluster also has an associated *representative* particle, which is rendered when no recursion into that cluster takes place. The recursion is steered by the screen size of the clusters.

If the selection were orders of magnitude slower, this would severely degrade the usability of the program and its acceptance by the users, so straight-forward traversal and flagging was not an option for our interaction extension. To enable correct hierarchical intersection and inclusion tests, we first had to tag all clusters with the bounding box extents of all their children. To support fast selection and visibility operations that avoid a complete traversal of the hierarchy when possible, we added 8 flags to each cluster/point:

- t the element is temporarily selected
- t_c the element has temporarily selected children
- s the element is selected
- s_c the element has selected children
- S_c all of the element's children are selected
- v the element is visible
- v_c the element has visible children
- V_c all of the element's children are visible.

To perform a temporary selection, the hierarchy needs to be traversed only where v_c holds. During the downstream recursion all points are tagged t if they are contained within the selection frustum. Upstream all clusters containing points $p|t$ are flagged t_c. To add or subtract the temporary selection from the permanent selection, only t_c are recursed, and the upstream is used for setting the flag s_c and removing t and t_c. For the intersection of both sets, both s_c and t_c need to be considered. To hide large numbers of points, the recursion can be stopped at the first cluster where the whole bounding box is inside the volume, since removing v, v_c, and V_c stops any further recursion and saves the effort of making all children consistently hidden (see also Figure 5). The reinstate consistency upon showing the points again, only those parts have to be traversed where V_c is not set.

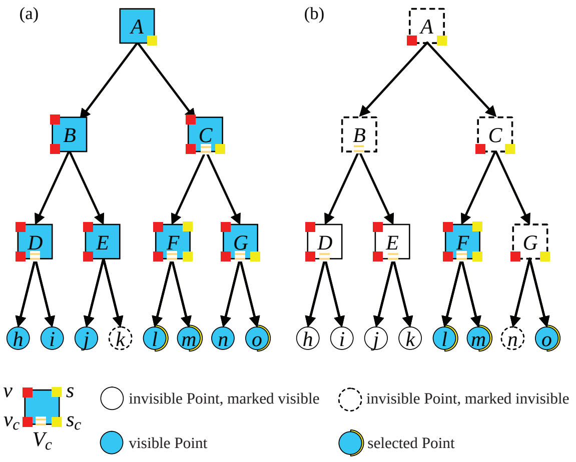

(a)
(b)
A
B
C
D
E
F
G
h
i
j
k
l
m
n
o
v
s
v_c
s_c
V_c
invisible Point, marked visible
invisible Point, marked invisible
visible Point
selected Point

Figure 5: *(a) Example for a consistent hierarchy before triggering an interaction (b) Example for an inconsistent hierarchy when quickly hiding A completely. Only selected points remain. B is temporarily inconsistent because its children are only hidden because the recursion stops at B and the data below need not be modified.*

One tricky part of the implementation was synchronizing the table interaction, because interaction basically has to be interpreted on the head node, but the user interface is displayed and used on the rendering nodes. The table was implemented with GTK [GTK], a free platform-independent toolkit. Although there are approaches which address multiple displays, like [dmx], we were unsure if its limitation regarding OpenGL extensions would not become a hindrance in the future and therefore refrained from using it, relying on our own implementation. Basically the relative mouse position inside the GTK window is transferred from the master node to the clients and all mouse clicks are synthesized from the flystick interaction. As off-screen clicks – which are common since the table will usually span several rendering nodes' displays when shown on the powerwall – are often misinterpreted, the window is temporarily moved to be completely visible (the viewport origin) and the click triggered only then. The window's position is reset to the earlier position after the click. The windows on all the screens have to be identical as pixel positions are transferred. As long as using the mouse on the master node and showing a replica on the client this works well and is almost trivial. When it comes to using this approach while using the flystick, however, this will not work. The windows can still contain the same information, but the horizontal and vertical screen resolution is different on the master compared to the clients. As a result, we have to transmit fractional window-relative coordinates from the master to the clients.

5. Results and User Studies

Figure 6 shows a closeup of a cosmological simulation consisting of 20 Million particles, where a freeform selection has been interactively drawn into the dataset. Figure 7 shows how the selection is perspectively extruded through the data as well as the holes caused by the spacing between letters. The adaptive rendering ensures about 10fps in both cases. Figure 2 finally shows a set of selected points and their data displayed in the table. Since the table and view are linked bidirectionally, the two selected points in the table are highlighted by bounding boxes in the 3D view.

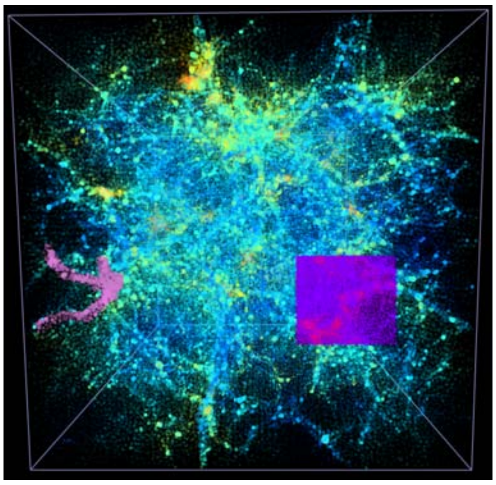


Figure 6: *Left: projected selection of a filament in the dataset, done with the brush tool. Right: projected rectangular selection. See also Fig. 7*

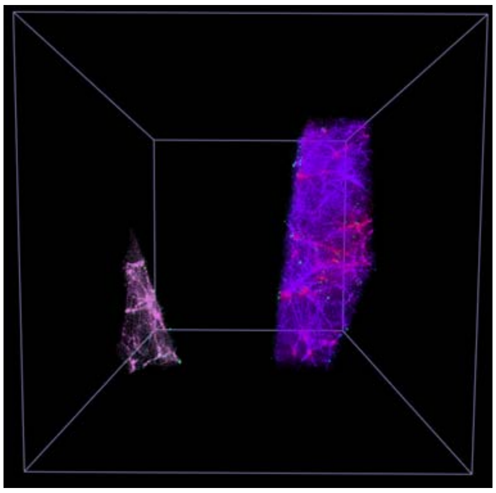


Figure 7: *Top-down view of the projected selections as in Fig. 6.*

We conducted several informal user studies with few participants as suggested in [KHI*03], on the one hand to improve the problematic menu interaction in the VR environment and on the other hand to test different hypotheses, i.e.:

- *Usability depends on the visualized dataset.* Contrary to our expectations, the complexity of the dataset has no negative impact on the usability. The users had to adjust the sensitivity of the positional mapping to better fit the amount of detail in the dataset, but they did not perceive that as a drawback.
- *Navigation using the flystick is more intuitive than using mouse and keyboard.* This was true, but only as long as the 3D cursor was inside the dataset and could easily be localized in space with stereo vision. Since the cursor is the interaction pivot, rotations around points outside the dataset put the users into difficulties at first. After some familiarization this effect disappeared.
- *The rectangular selection is better suited for large areas.* Even worse, the users almost exclusively preferred this mode over the brush selection when working in single-display mode. In the VR environment, however, their behavior was inverted as most users preferred the combined candle-and-brush mode.
- *A two-tier selection is not intuitive.* To our surprise, this feature was well accepted by the users.
- *Users with low to no experience using VR setups have more difficulty with the user interface than people that have already employed VR.* Neither was this hypothesis true, nor was it the contrary. Since the flystick is quite different from space mice or pointing rods, there was nearly no difference in the learning curve and execution speed between experienced and unexperienced users.

We then also conducted tests with one of the targeted users, i.e. a thermodynamics researcher with previous experience with VR environments. An aspect that was judged very positively was the navigation in the dataset by directly grabbing and rotating it using the flystick as a proxy for the interaction pivot. The filtering possibilities also received positive feedback with a particular emphasis on the slicing aspect which allows the user to look inside clusters of molecules to grasp their density, and, in case of mixtures, the composition of agglomerates and the coating of surfaces. The user also pointed out that such an interface would be very fitting for a future simulation steering application.

6. Conclusion and Future Work

We have described our approach for providing researchers working with particle-based simulations with a flexible, fast tool to interactively explore their simulation results. The acceptance from the users was very positive, which encouraged us to continue our work in this direction. As the next step we will integrate our project for efficiently streaming large time-based datasets based on the same rendering framework with these UI extensions. At that point, the selection will also give access to the larger number of attributes contained in those types of simulation (particle type, orientation, velocities, etc.). We plan to extend the system for greater flexibility and eventually turn it into a fully-fledged simulation steering application.

Acknowledgements

This work was partially supported by the DLR in the context of Project 688. We would like to thank Jadran Vrabec from the Institute of Thermodynamics and Thermal Process Engineering in Stuttgart for providing the molecular simulation results as well as for evaluating the system. The galaxy formation simulations were carried out by the Virgo Supercomputing Consortium using computers based at the Computing Centre of the Max-Planck Society in Garching and at the Institute for Computational Cosmology. The data is publicly available at www.mpa-garching.mpg.de/NumCos.

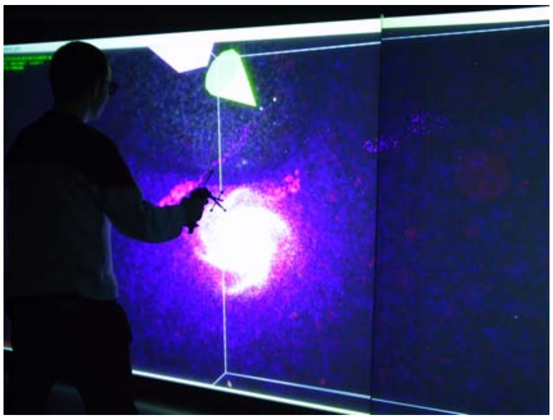


Figure 8: *One of the authors navigating a galaxy simulation with the VR interface.*

References

[BBMP97] Billinghurst M., Baldis S., Matheson L., Philips M.: 3d palette: a virtual reality content creation tool. In *VRST '97: Proceedings of the ACM symposium on Virtual reality software and technology* (1997), pp. 155–156.

[dHKP05] de Haan G., Koutek M., Post F. H.: IntenSelect: Using Dynamic Object Rating for Assisting 3D Object Selection. In *Eurographics Workshop on Virtual Environments (EGVE)* (2005), R. Blach E. K., (Ed.).

[dmx] Distributed Multihead X Project. http://dmx.sourceforge.net/.

[dRFK*05] del Río A., Fischer J., Köbele M., Bartz D., Strasser W.: Augmented Reality Interaction for Semiautomatic Volume Classification. In *Eurographics Workshop on Virtual Environments (EGVE)* (2005), R. Blach E. K., (Ed.), pp. 113–120.

[GTK] the GIMP Toolkit. http://www.gtk.org/.

[HE03] Hopf M., Ertl T.: Hierarchical Splatting of Scattered Data. In *Proceedings of IEEE Visualization '03* (2003), IEEE.

[KHI*03] Kosara R., Healey C. G., Interrante V., Laidlaw D. H., Ware C.: User studies: Why, how, and when. *Computer Graphics and Applications 23*, 4 (July/August 2003), 20–25.

[LG94] Liang J., Green M.: JDCAD: A highly interactive 3D modeling system. *Computers & Graphics 18*, 4 (1994), 499–506.

[RE05] Reina G., Ertl T.: Hardware-Accelerated Glyphs for Mono- and Dipoles in Molecular Dynamics Visualization. In *Procceedings of EUROGRAPHICS - IEEE VGTC Symposium on Visualization 2005* (2005), Brodlie K. W., Duke D. J.,, Joy K. I., (Eds.).

[WDC99] Watsen K., Darken R., Capps M.: A handheld computer as an interaction device to a virtual environment. In *Proceedings of the third Immersive Projection Technology Workshop* (1999).

[WG95] Wloka M. M., Greenfield E.: The virtual tricorder: A uniform interface for virtual reality. In *ACM Symposium on User Interface Software and Technology* (1995), pp. 39–40.

[Wil96] Wills G.: Selection: 524,288 ways to say "this is interesting". In *Proceedings InfoVis* (1996), pp. 54–60.