Distributed Rendering and Collaborative User Navigation- and Scene Manipulation

Christian Kehl

University of Amsterdam,
Science Park 904, NL-1098XH Amsterdam
eMail: c.kehl@uva.nl | christian-kehl@web.de
URL: http://graphics.tudelft.nl/christian-kehl

Abstract: The broad availability of large datasets for decision-making, as well as current technological trends in large-area projection systems and intelligent interaction devices pose new challenges for designing Virtual Environments (VEs) for decision-making processes. This paper addresses three main challenges for modern VEs in the context of decision-making: the render organisation for large-scale projection systems, the scene navigation by a group of concurrent users, and the in-place modification of rendered datasets by concurrent users. Distributed Rendering, Collaborative User Navigation and Collaborative Scene Manipulation are my proposed approaches for extending VEs to work in modern decision-making setups.

1 Introduction

Visualising large-scale datasets is nowadays commonly applied in decision-making processes across many domains, such as GIS and medical diagnosis. In this environment, user groups observe and explore interactive 3D scenes showing their domain-specific data.

Although current visualisation systems are steered towards highlighting important information, target user groups continue to push demands on such systems. Our research group highlights three areas of improvement that emerged from recent studies with water management experts [1], which are discussed in this paper. First, larger projection- and screen setups and the support of visualisations on such devices create challenges for a distributed render organisation. Secondly, decision-making demands intuitive and scalable navigation interfaces that allow multiple users to participate in discussions. Lastly, interaction interfaces that allow concurrent user groups to modify (i.e. annotate) large models inside the visible scene is a major demand of experts across application domains.

The referred research on this topic is based on the work of de Haan [2], who developed a technical as well as cognitive framework for user interaction. The framework binds the tasks of “Rendering” and “Interaction”, so that rendering tasks relate to user actions. Technically, the framework is a conglomerate of StateStreams, StreamMachines and Application setups. The framework is well-applicable in small-scale Virtual Environments (VEs).

This approach is extended to facilitate large-scene rendering and smooth navigation and interaction by medium-sized user groups. Hence, I introduce the

following techniques: Projection-aware Distributed Rendering, Collaborative User Navigation and a new approach for Collaborative Scene Manipulation.

The approach and the developed techniques are validated using the AHN-2 point cloud dataset for large-scale rendering tests, and smaller, artificial point-sampled geometry for validating the navigation and interaction.

2 Related Work

Challenges in the design and development of decision-making support tools often arise from expert user demands, who are working with the tools on daily basis. One such user demand is the support for modern, large 2D displays, multi-display setups, as well as 2D-/3D stereo projector systems, across domains [3] [4] [5]. Rendering on screens at such scale and resolution results in more expensive computations. This can lead up to a point where the visualization becomes non-interactive, even when using modern Level-of-Detail and out-of-core rendering methods [6] [7]. Issues according to render speed can be commonly overcome via Distributed Rendering. The open issue with current Distributed Rendering approaches, such as Chromium [8] and Equalizer [9], is their limited applicability to non-planar projection surfaces, which we address in our approach.

Another challenge for decision-making support tools that emerges from user demands is the support for novel interaction devices. An increasing number of haptic- (joysticks, Wiimote [10]) and non-haptic (Kinect, LeapMotion) devices became available in recent years. Their importance for collaborative 3D VE's has already been proven in earlier research stages [11]. We focus on using these devices in a consistent manner for 3D scene, 6 Degree-of-Freedom (DoF) navigation. Early VE's for data exploration and decision-making, such as VRmeer [12] and i3D [13], assume a small number of users navigating the virtual world and being physically close to the projection device. A first approach for distant (i.e. *remote*) navigation is VRPN [14]. In this framework, device classes abstract a collection of input devices. Due to our focus on navigation, we provide consistent abstractions that allow 6-DoF navigation across the range of devices. We furthermore extend the approach by linking devices and function classes.

A last challenge we approach is the manipulation of objects in the visible scene by multiple users. Current decision-making support tools and former multiuser-VE's, such as DIVE [15], annotate and modify objects via texture images. This approach can be very memory-intensive for large models. Furthermore, it requires texture coordinates, which do not exist for all object representations (e.g. point-based datasets).

3 Projection-aware Distributed Rendering

Distributed, projection-aware, synchronised rendering is introduced to approach the diverse, possibly compute-intense demands of large-scale VEs. Hereby, each render *client* of the network, virtually connected to one output device (i.e. screen or projector), works on a dedicated part of the scene. Each client has a dedicated database connection to enable independent data requests. The *master* node distributes the interaction commands. It also composes the individual image into one framebuffer output, if required. The clients can be synchronized via timestamp

or frame number. The choice of the synchronisation method depends on the allowed lag between the render devices. When rendering stereoscopic projections, large lags lead to visual errors in the stereo-composition. The acceptable lag for composing a stereo image depends on projection-device specifics, as well as on the rendered scene.

For multi-projector systems, the master manages the view matrix update. This matrix is transferred to the clients, which multiply the incoming matrix with a local screen matrix. The local screen matrix incorporates the offset position (for planar projections) and radial offsets and shears (for cylindrical projections) for the connected projection device. The matrix composition is explained in Eq.1 and Eq.2.

Planar View Matrix and Projection:

$$M_{final} = M_{planar\_offset} \cdot M_{head\_position} \cdot M_{eye\_position} \cdot M_{global\_view} \quad \textbf{(1.1)}$$

$$left = C_{x_{screen}} - \frac{width}{2} \quad \textbf{(1.2)}$$

$$right = C_{x_{screen}} + \frac{width}{2} \quad \textbf{(1.3)}$$

$$top = C_{y_{screen}} + \frac{height}{2} \quad \textbf{(1.4)}$$

$$bottom = C_{y_{screen}} - \frac{height}{2} \quad \textbf{(1.5)}$$

$$P_{final} = projection(left, right, top, bottom, near, far) \cdot M_{eye\_offset} \quad \textbf{(1.6)}$$

Cylindrical View Matrix and Projection:

$$M_{final} = M_{cylindrical\_offset} \cdot M_{head\_position} \cdot M_{eye\_position} \cdot M_{global\_view} \quad \textbf{(2.1)}$$

$$left = C_{x_{screen}} - \frac{width \cdot sheer_x}{2} \quad \textbf{(2.2)}$$

$$right = C_{x_{screen}} + \frac{width \cdot sheer_x}{2} \quad \textbf{(2.3)}$$

$$top = C_{y_{screen}} + \frac{height \cdot sheer_y}{2} \quad \textbf{(2.4)}$$

$$bottom = C_{y_{screen}} - \frac{height \cdot sheer_y}{2} \quad \textbf{(2.5)}$$

$$P_{final} = projection(left, right, top, bottom, near, far) \cdot M_{eye\_offset} \quad \textbf{(2.6)}$$

The developed network protocol has a simple layout that consists of message length, message type (in this case: frame number) and the data. It can be send as byte format or plaintext to generic composition server applications.

4 Collaborative User Navigation

Next to the large-scale rendering, it should be possible for multiple, concurrent users to navigate inside the rendered scene. That commonly demands the presence of the users at the respective render device.

In order to overcome this restriction, the collaborative navigation approach allows the remote navigation of the scene via heterogeneous devices. The approach is

based on function abstraction and device abstraction. The navigation functions are abstracted according to the following class scheme:

- *Triggers*: event-based functions
- *Positioning*: camera positioning at random points
- *Movement*: camera positioning according to a continuous function
- *Orientation*: affine transformation-based model movement

This abstraction allows, for example, to implement- and switch easily between camera settings of different users. It continues at device level, where one of the following classes is tied to each physical interaction object:

- *Discrete Navigation*: movement and orientation via trigger-like objects (e.g. buttons, keys)
- *Continuous Navigation*: movement and orientation on axis-based objects (e.g. hats, throttles, sticks, gestures)
- *Discrete Trigger*: connecting trigger-like objects with switch-functions
- *Continuous Trigger*: connecting axis-based objects with scale-functions (e.g. transfer functions, colour range)

A mapping example of physical object, interaction class and function class is given in Figure 1, for a haptic- as well as non-haptic device.

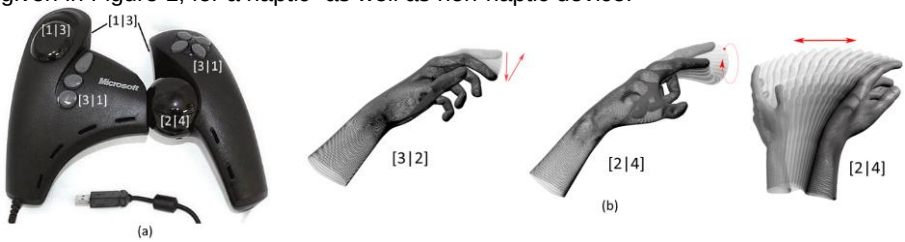

Figure 1 Interaction mapping examples with our framework, shown on haptic (a) and non-haptic (b) devices

The interaction clients, being connected to the particular device, are implemented in Python to simplify the framework's extension. It further allows platform-independent usage. The interaction server classes are available in native C++ and native Python, to be interfaced in generic render engines. The network protocol layout of section 3 was adapted to support the navigation commands. The synchronization of (dis)appearing devices during runtime is done via broadcasting the current view-matrix to each device. The server accumulates incoming, difference-coded view matrices of each device during rendering.

5 Collaborative Scene Manipulation

The collaborative scene manipulation is a key element for decentralized decision-

making support tools. It allows multiple users to annotate and adapt the scene on-the-fly. Each user performs his operations locally in independent views, while the render master collects all modifications and renders them in the global scene.

One challenge is the transmission of the model's geometry to each client. The aforementioned, large datasets are disseminated as simplified versions to reduce the stress on low-performance devices (e.g. laptops, tablets). For the experiments, we apply random point re-sampling as a pre-processing step to the rendering. We forward the interested reader to [16] [17] for more simplification methods.

In the next step, after marking and modifying the area, we need to adapt the global, non-simplified dataset accordingly. For this, we extend an initial approach for 2D area marking [7] that allows rendering exclusion of parts of the dataset, height adaptations and colouring. Generally, we combine a certain operation with a spatial constraint in order to modify the dataset. The first extension uses spherical markers for 3D annotation, for which their centres, radii and colour are stored. These data are organised as a texture, and subsequently evaluated as implicit function by a GPU shader to colour the dataset vertices. An example is given in Figure 2.

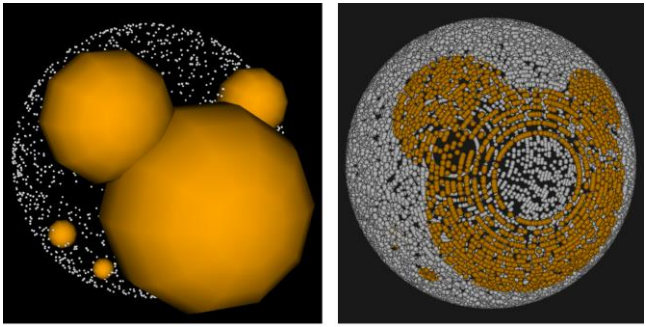

Figure 2 Example rendering for sphere-based area marking

A second step uses object-oriented bounding boxes (OOBBs) for marking. Hence, each box' centre as well as its world-to-object transformation matrix is stored, in addition to its colour and dimensions. The shader transforms the box centre and each scene vertex into local coordinates using the transposed transformation matrix. Then, the GPU evaluates the box' implicit function for colouring each vertex. The concept is shown in Figure 3, an example rendering is given in Figure 4.

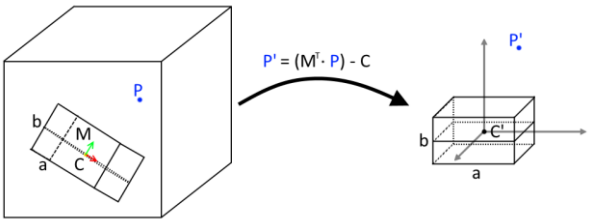

Figure 3 Sketch of box evaluation for area marking

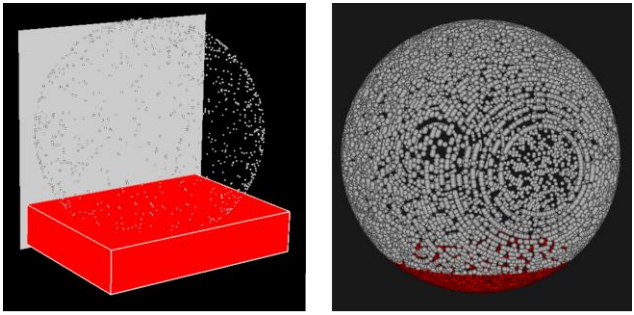

Figure 4 Example rendering for box-based area marking

A third, proposed extension handles 4-sided prisms. In this case, a single centre- and transformation matrix is not sufficient for the inside/outside-check. We therefore treat them as closed meshes, for which we store for each surface:

- 3 corner vertices
- their counter-clockwise-oriented indices
- their transformation matrix into a local coordinate system

Inside the GPU shader, each scene vertex is transformed with each plane's matrix and checked if the vertex is inside the volume of the combined implicit function.

6 Technical Results

The Distributed rendering is tested on parts of the Dutch AHN-2 (www.ahn.nl) multi-terabyte point dataset. The test configuration included 3 render clients, of which one was acting as master. The scene, an urban environment covering 5km by 12.5km (62.5 km², around 55 GB in total), was rendered in single-view and stereoscopic view, using planar- and cylindrical projections. In our test, each client was connected to an individual screen, without final frame composition. The render clients' hardware was as follows:

- Client 1/Master: Intel Xeon 6 x 3.2 GHz (12 processing elements (PE) in Hyperthreading (HT) -mode), 16 GB memory, NVIDIA GeForce GTX 680
- Client 2: Intel Xeon 6 x 3.2 GHz (12 PE's using HT), 16 GB memory, NVIDIA Quadro K4000

- Client 3: Intel Xeon 4 x 2 GHz, 8GB memory, NVIDIA GeForce GTX 480

The different device capabilities led to asynchronous render times between the clients. Hence, errors occurred in the frame matching when using a timestamp as mean of synchronisation. Using the frame number for synchronisation reduced the rendering speed from 40 frames per second on average to 16 frames per second, with an allowed lag of 3 frames. On the other hand, the visual result of the frames is homogeneous, which gives an acceptable frame matching, and also allows a stereo frame composition. An example rendering for cylindrical projections is given in Figure 5.

Figure 5 Example for Distributed Rendering of a cylindrical projection on a 3-display screen setup

The approach for collaborative user navigation received positive feedback in recent demonstrations. Two particular advantages of this interaction framework are the ability to steer the visual via multiple devices connected to one client, as well as the usage of several clients at the same time. The gained platform independence is important. As the render solutions commonly run on Linux-platforms, most users outside the informatics community were formerly hesitant to use in daily practice. Thus, decision-makers are more willing to navigate via Microsoft Windows while benefiting from advances in novel rendering algorithms.

The scene manipulation algorithms are currently only tested on a smaller, artificial dataset due to the necessary simplification pre-prepocessing. As the rendering is executed in screen space, the results can be generalized to larger datasets. Figure 6 (left) shows performance figures on the spherical area marking. As can be seen, we experience significant performance drops at around 30 sphere checks. This behaviour can be improved by ordering the markers in spatial hierarchies, as in Kehl et al. [7]. Figure 6 (right) shows performance figures for OOBB markers. Here, a significant performance drop is observed at 7 markers, due to the more complex inclusion check. The prism-based marking needs further refinement to be applied in real-time. In general, although majorly applying colour annotation as mean of scene modification in our tests, further operations such as object cutting and displacement can simply be implemented using our modification approach. The measurement systems was an Intel i7 920 x4 HT processor @ 2.66 GHz, 8 GB memory and an AMD HD 4850 X2 graphics adapter (only one GPU used).

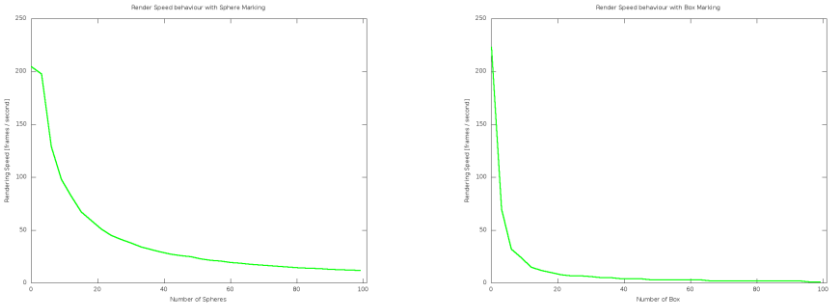

Figure 6 Measurements of the rendering speed [frames per second] for sphere-based (left) and box-based (right) area marking

7 Discussion

The techniques currently help water boards and hydrologists in the Netherlands to explore flood scenarios, occurring due to massive rainfall in urban environments and levee breaches in coastal areas. Projection setups that are currently used for this purpose include the 3D theatre in Groningen, a 5m-by-2.5m cylindrical wall project, large-scale multi-touch tables in offices and a PowerWall planar projection in the VRLab of the TU Delft. Furthermore, a large PowerWall setup for flood visualisation is used in the Delft Science Center and the “Watersnoodmuseum” in Zeeland for the purpose of public education.

The Collaborative User Navigation- and Scene Manipulation allow water experts to interactively communicate and plan flood protection measurements during routine meetings and workshops in the water-management domain. The 2.5D annotation version is already implemented in the 3di software package, and we are looking forward to see this annotation technique becomes common practice in the future. The approach has also further use in other domains of decision-making, such as oil- and gas reservoir modelling, where 3D data annotations form the basis of the reservoir segmentation, and pre-operative planning in hospitals, where medical diagnoses are based on scanned object segmentations.

8 Future Work

The presented techniques are part of a continuous research effort towards a consistent Visualisation- and Interaction framework for flexible communication settings (with respect to projection, render- and interaction devices).

The support for touch-based interaction devices is a demand emerging from recent studies, which we will integrate in our collaborative navigation. This is challenging to model because, in contrast to keyboard, joysticks and non-haptic devices, touch-screens commonly demand a visual reference surface to form navigation commands.

The technique of collaborative scene modification still offers various challenges. The data delivery of large-scale models as small data packages to client devices

still requires further attention. Although numerous techniques exist for data simplification, we will investigate the delivery of the data to the client devices via out-of-core data structures, such as they are delivered to the server application. As explained in detail in the technique discussions (section 6), we will devote further research in improving the prism-based annotation by simplifying the spatial constraint and avoiding performance bottlenecks. One further extension is management of the marker objects within texture-organised spatial hierarchies.

Acknowledgments

We thank the “Computer Graphics and Visualization Group” of Delft University of Technology (headed by Prof. Dr. Elmar Eisemann) for equipment support, as well as the “Donald Smits Institute” of the Rijksuniversiteit Groningen (headed by Dr. Frans van Hoesel) for their collaboration on the Collaborative Rendering approach experiments.

We further thank dr. Ir. Gerwin de Haan, for his explanations on his implemented Interaction framework in VRmeer, as well as Prof. Dr. rer. nat. Herbert Litschke and Simon J. Buckley, for their short-term review.

Literature

[1] J. G. Leskens, C. Kehl, T. Tutenel, T. Kol, G. de Haan, G. Stelling and E. Eisemann, “Interactive Flood-risk Analysis -- Case Studies using a High-performance Visualization System,” in *Special Issue on Decision Making on Adaptation to Climate Change*, Springer, (to appear 2014/2015).

[2] G. de Haan, Techniques and architectures for 3D interaction, 2009: Delft University of Technology, Delft.

[3] A. Majumder and B. Sajadi, “Large Area Displays: The Changing Face of Visualization,” *Computer*, vol. 46, no. 5, pp. 26-33, 2013.

[4] D. Mendes, M. Sousa, B. Araujo, A. Ferreira, H. Noronha, P. Campos, L. Soares, A. Raposo and J. Jorge, “Collaborative 3D Visualization on Large Screen Displays,” in *Powerwall-International Workshop on Interactive, Ultra-High-Resolution Displays - ACM CHI*, 2013.

[5] J. Kuchera-Morin, M. Wright, G. Wakefield, C. Roberts, D. Adderton, B. Sajadi, T. Höllerer and A. Majumder, “Immersive full-surround multi-user system design,” *Computers & Graphics*, 2014.

[6] C. Kehl and G. de Haan, “Interactive Simulation and Visualisation of Realistic Flooding Scenarios,” in *Intelligent Systems for Crisis Management*, Enschede, The Netherlands, Springer, 2012, pp. 79-93.

[7] C. Kehl, T. Tutenel and E. Eisemann, “Smooth, Interactive Rendering Techniques on Large-Scale, Geospatial Data in Flood Visualisation,” in

Information and Communication Technologie (ICT) Open, Eindhoven, The Netherlands, 2013.

[8] G. Humphreys, M. Houston, R. Ng, R. Frank, S. Ahern, P. D. Kirchner and J. T. Klosowski, "Chromium: a stream-processing framework for interactive rendering on clusters," *ACM Transactions on Graphics (TOG)*, vol. 3, no. 21, pp. 693-702, 2002.

[9] S. Eilemann, M. Makhinya and R. Pajarola, "Equalizer: A scalable parallel rendering framework," *IEEE Transaction on Visualization and Computer Graphics*, vol. 15, no. 3, pp. 436-452, 2009.

[10] C. Wingrave, B. Williamson, P. D. Varcholik, J. Rose, A. Miller, E. Charbonneau, J. Bolt and J. LaViola, "The Wiimote and Beyond: Spatially Convenient Devices for 3D User Interfaces," *IEEE Computer Graphics and Applications*, vol. 2, no. 30, pp. 71-85, 2010.

[11] A. Schmeil and M. Eppler, "Formalizing and Promoting Collaboration in 3D Virtual Environments - A Blueprint for the Creation of Group Interaction Patterns," in *Facets of Virtual Environments*, 33 ed., Springer Berlin Heidelberg, 2010, pp. 121-134.

[12] G. de Haan, E. J. Griffith, M. Koutek and F. H. Post, "Hybrid Interfaces in VEs: Intent and Interaction," in *Proceedings of the 12th Eurographics Conference on Virtual Environments*, Aire-la-Ville, Switzerland, Switzerland, 2006.

[13] E. Gobbetti and J.-F. Balaguer, "i3D: An interactive system for exploring annotated 3D environments," in *Scientific Visualization'95: proceedings of the International Symposium*, Cagliari, Italy, 1995.

[14] R. M. Taylor II, T. C. Hudson, A. Seeger, H. Weber, J. Juliano and A. T. Helser, "VRPN: A Device-independent, Network-transparent VR Peripheral System," in *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, New York, NY, USA, 2001.

[15] O. Hagsand, "Interactive multiuser VEs in the DIVE system," *IEEE MultiMedia*, vol. 1, no. 3, pp. 30-39, 1996.

[16] M. Pauly, M. Gross and L. P. Kobbelt, "Efficient simplification of point-sampled surfaces," *Proceedings of the conference on Visualization '02 (VIS '02)*, pp. 163-170, 2002.

[17] J. O. Talton, "A short survey of mesh simplification algorithms," *University of Illinois at Urbana-Champaign*, 2004.