Proceedings of SIGRAD 2011

Evaluations of Graphics and Visualization
— Efficiency, Usefulness, Accessibility, Usability

November 17–18, 2011

KTH, Stockholm, Sweden

Edited by

Thomas Larsson, Lars Kjelldahl & Kai-Mikael Jää-Aro

Linköping Electronic Conference Proceedings, No. 65
Linköping University Electronic Press
Linköping, Sweden, 2011
ISBN 978-91-7393-008-6
ISSN 1650-3686 (print)
ISSN 1650-3740 (online)
http://www.ep.liu.se/ecp_home/index.en.aspx?issue=065

Print: US-AB, KTH, 2011

Cover illustrations selected from the papers.

Preface

This year, we are happy to announce the 10th SIGRAD Conference Proceedings with peer-reviewed papers. The conference has been held annually since 1981, but in 2002 the SIGRAD conference shifted focus and become oriented towards publication of research papers. Since then, the conference has been hosted by universities located in Norrköping, Umeå, Gävle, Lund, Skövde, Uppsala, Stockholm, Göteborg, and Västerås.

This year the conference has returned back to KTH in Stockholm. As before, the conference aims at attracting researchers, engineers, developers, designer, teachers, and students across multiple disciplines, and to serve as a meeting place where you can learn and share ideas about some of the latest developments in computer graphics, visualization, and human-computer interaction. We are confident that the exciting presentations in this year's program conform to these goals.

Many have contributed to make the conference an enjoyable and beneficial experience. In particular, we would like to express our gratitude to the paper authors and the keynote speakers, Daniel Weiskopf, Achim Ebert, and Jan Gulliksen, without whom this conference would not have been possible. Also, we would like to thank the international program committee for their engagement and reviewing efforts. Finally, we would also like to thank the School of Computer Science and Communication (CSC) and VIC Sthlm for generously supporting the conference.

Welcome to SIGRAD 2011, the annual conference of the Swedish chapter of Eurographics! Be inspired, share experiences, and bring home new fresh ideas.

The editors

Conference co-chairs

Lars Kjelldahl, Royal Institute of Technology
Björn Thuresson, Royal Institute of Technology
Thomas Larsson, Mälardalen University
Kai-Mikael Jää-Aro, Vizrt Ardendo

International programme committee

Tomas Akenine-Möller, Lund University
Nils Andersson, Eon Reality
Ulf Assarsson, Chalmers University of Technology
Yifang Ban, Royal Institute of Technology
Mark E Dieckmann, Linköping University
Michael Dogget, Lund University
Thomas Ertl, University of Stuttgart
Morten Fjeld, Chalmers University of Technology
Jan Gulliksen, Royal Institute of Technology
Helwig Hauser, University of Bergen
Kai-Mikael Jää-Aro, Vizrt Ardendo
Andreas Kerren, Linnaeus University
Lars Kjelldahl, Royal Institute of Technology
Ivana Kolingerova, University of West Bohemia
Thomas Larsson, Mälardalen University
Erwin Laure, Royal Institute of Technology
Alex Olwal, Royal Institute of Technology
Holger Regenbrecht, University of Otago
Stefan Seipel, University of Gävle
Yngve Sundblad, Royal Institute of Technology
Rüdiger Westermann, Technical University of Munich
Anders Ynnerman, Linköping University

Table of Contents

Keynotes

Continuous Statistical Visualization — Where Information Visualization Meets Scientific Visualization

Daniel Weiskopf

Universität Stuttgart

Abstract

A rather recent trend in visualization is to combine traditional scientific visualization methods (such as volume rendering) with information and statistical visualization techniques (such as scatterplots). This combination of techniques is particularly useful for multi-variate data defined on spatial grids because the spatial relationships and the characteristics of the data attributes can be shown simultaneously. However, statistical and information visualization methods for multi-variate data, such as scatterplots and parallel-coordinates plots, have traditionally been applied to intrinsically discrete data points and, therefore, treat data as a collection of independent data samples. In contrast, this talk advocates the use of continuous data models for statistical visualization applied to continuous data. Main advantages are that visual artifacts from data sampling are avoided and that the visualization process becomes scalable with respect to data set size.

Biography: Daniel Weiskopf received the Diplom (MSc) degree in physics and the PhD degree in physics, both from Eberhard-Karls-Universität Tübingen, Germany, and he received the Habilitation degree in computer science at Universität Stuttgart, Germany. From 2005 to 2007, Dr. Weiskopf was an assistant professor of computing science at Simon Fraser University, Canada. Since 2007, he has been a professor of computer science at the Visualization Research Center, Universität Stuttgart (VISUS) and at the Visualization and Interactive Systems Institute (VIS), Universität Stuttgart. His research interests include visualization, visual analytics, GPU methods, real-time computer graphics, ubiquitous visualization, perception-oriented computer graphics, and special and general relativity. He is member of ACM SIGGRAPH, the Gesellschaft für Informatik, and the IEEE Computer Society.

Think big! — Usability of large screen environments

Achim Ebert

University of Kaiserslautern

Abstract

If you have ever seen a soccer game or a concert on a large screen TV, then you have been surely impressed with the captivating and amazing display. This is not limited to entertainment alone — the effect also is the same in more serious applications. In this respect, size matters. In most cases we cannot simply use the algorithms for standard displays on large screens — the size of the display significantly affects most visualizations and interaction modalities. Therefore, we need to understand the advantages and disadvantages of this medium, and the ways large screens are different or similar to smaller screens. Beside the need to develop special hardware and software, we always have to think about the user of such environments. Does he or she really need a wall-sized screen? Can he or she really make use of a gigapixel display wall? In the keynote we will give an overview of HCI and Visualization techniques dealing with large display environments. We will discuss how much size users really need and think about alternative approaches.

Biography: Achim Ebert holds a degree and a doctor in Computer Science. He is a professor and co-head of the Computer Graphics and HCI lab at the University of Kaiserslautern. He is also a member of the lead personnel of DFG's International Research Training Group (IRTG) "Visualization of Large and Unstructured Data Sets". His current research topics include information visualization, immersive scenarios, and human-computer interaction. He participated or led several national and international research projects in the area of visualization and HCI. He has published more than 50 refereed publications. Achim Ebert has founded and is heading the IFIP working group 13.7 on Human-Computer Interaction and Visualization. He acts as a member of many international program committees (e.g, ACM, IEEE, and IFIP) and as a reviewer for several journals and conferences.

Where do Visualization meet User-centred design? — Exploring the tension between innovative design ideas and user preferences and innovation

Jan Gulliksen

Royal Institute of Technology (KTH)

Abstract

One of the fundamentals of user-centred design and participatory design, in particular is the active involvement of users in the design and development process. However, users are often considered to lack capabilities of being innovative and to be able to see beyond what they currently have in front of them. On the other hand, when working with visualization, innovative ideas of new ways of displaying information may be brought up, far beyond what users could ever have imagined.

The purpose of this talk is to contrast the essential parts of user-centred design with the innovation capabilities required to visualize information in new ways and to explore the tension between these two fields. Could there be such a thing as user-centred information visualization, or are the two different fields pulling in completely different directions?

Biography: Jan Gulliksen is a professor of Human Computer Interaction at the Royal Institute of Technology (KTH) in Stockholm, Sweden. He is also the Dean of the School of Computer Science and Communication. Jan holds a Master of Science in Engineering Physics and a PhD in Systems Analysis and was promoted to full professor in Human Computer Interaction at Uppsala university, Sweden. Jan has conducted a number of practically oriented Action research projects with Public Authorities in Sweden with the purpose of introducing more user-centred design methodologies. Jan has also been active in writing ISO standards on Usability, Accessibility and Human-centred design. Jan has authored more than 100 publications. Jan was the founder of the NordiCHI conference series and is now the chairman of IFIP TC 13 on Human Computer Interaction, the organization behind the INTERACT conference series. Jan has also been active in the industry, working with putting the research into practice as a consultant.

Papers

Assisted Environment Map Probe Placement

M. G. Chajdas[1], A. Weis[1] and R. Westermann[1]

[1]Technische Universität München

Abstract

Placing probes for environment maps is a tedious and time-consuming task for current game developers. In particular, locating the significant places for reflection requires the artists to manually scan through a whole game level. Even then, probes may wind up being too similar and have to be cleaned up manually later on. Furthermore, the whole process needs to be repeated when the content changes.
We propose a novel algorithm which assists the artist when placing environment map probes. Based on the original game content, we produce a set of candidate locations for inspection by an artist. By setting a few parameters like sampling density and aggressiveness of probe elimination, the overall number of generated sample probes can be easily adjusted. All of this requires only a short pre-process which can be done using the available in-game renderer. From the generated sample set, the artist can decide which locations are important, add or remove probes manually and adjust the placement to account for supplemental, content-specific parameters. Our initial probe placement, together with additional information computed by our algorithm, allows the artist to place probes more efficiently. Early user testing indicates that the total time can be reduced by up to half a day for a single game level.

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

1. Introduction and related work

Environment mapping [BN76, Gre86] is a widely used technique in real-time rendering to simulate diffuse and specular reflections. Lately, games have also started to use light probes – which are essentially low-resolution environment maps – for image-based lighting. Environment maps are easy to generate, can be pre-computed, and render efficiently due to hardware support. However, they are still relatively expensive to create in real-time, especially on consoles with limited vertex throughput. Hence, a lot of games use fixed environment map probes, which are placed by an artist during the scene creation [McT04]. At runtime, the game engine selects one or multiple probes close to the affected object and uses them for reflection or illumination.

Figure 1: Environment map probes placed in the *Terminal* level in Crysis 2. Scene courtesy of Crytek from Crysis 2.

As content complexity has increased, placing environment map probes has become a time-consuming process. Typically, one artist is responsible for setting up all probes in a game scene. In order to do this, she manually places probes and checks all possible player locations using a special view mode which lets her examine the reflections [Cry11] (see also figure 1.) If a place has too few probes, resulting in wrong reflections, the artist has to create a new probe at an appropriate location.

A sample image showing the probe locations found by our algorithm on the *Alphalabs1* level. Our algorithm places probes at important locations, like the lit control room on the right or the warning lamp in the top left. The small glyphs indicate candidates, while the large glyphs correspond to actually placed probes. Scenes from Doom 3, © id Software

Typically, an engineer will have to re-check the probe placement later in the development process to ensure performance and memory targets are met. For instance, some areas of the level might already stress the memory budget and only allow for a few probes. Once the content changes, both the artist and engineer have to re-evaluate the placed probes to make sure they are still at valid locations.

It is clear that a significant amount of time can be saved by assisting the artist in probe placement, for instance, by guiding her to possible candidate locations. Moreover, by taking performance or gameplay-specific data into account, placing undesired probes can be avoided in areas which are unreachable or where performance issues might arise. The final probe selection and/or re-positioning, however, is always performed by the artist, since one can neither predict the way the game is actually played nor how the artist envisioned the experience for a specific scene.

1.0.0.1. Our contribution: The primary goal of our work is a robust approach which assists the game artist in accomplishing the task of environment map probe placement. Therefore, we present an algorithm that automatically computes a set of important locations. Similar to [War94], finding these locations is based on a search along the directions of major change in a 3D irradiance distribution. Contrarily, however, in our case this distribution is given by a precomputed irradiance volume [GSHG98]. Here, changes can be with respect to the overall incoming radiance or the distribution of this radiance.

Once our algorithm has determined candidate probe location, clusters consisting of environment maps with similar irradiance distribution are determined and replaced by a single representant. The similarity measure we use builds upon the earth-mover's distance [RTG00], a histogram based method for determining the similarity between images.

Together with a game artist we have performed a qualitative assessment of our results, by providing our sampling as an initial guess and letting the artist manually refine this sampling using her expert intuition. The evaluation has been done using game levels from Doom3. As Doom3 is known for its unusual darkness and contrast, we used global illumination for all of our rendering. In this way, our results are more representative for current-generation content.

Our approach is vastly different to previous approaches for automated placement of environment maps, which aimed for maximum visibility coverage, i.e. they tried to place a minimum number of probes such that any part of the surrounding scene is visible from at least one probe [ML03]. When dealing with games, however, maximizing coverage is neither feasible nor desirable. First of all because the budget for environment maps is strictly limited in modern game engines. For example, a typical Cryengine 2 outdoor area with 1 km^2 might use only a single probe [Kap11], meaning that complete coverage is usually impossible to achieve in the first place. Furthermore, coverage only considers the level geometry while completely ignoring lighting and shading effects. In modern game scenarios, lighting is an integral part of level-design and is used extensively to guide the player's attention towards specific spots in a level. When using an approach that is purely based on geometry, many of the implicit importance hints given by lighting are lost.

For instance, a single probe can be usually used for all dark rooms in a game.

2. Environment maps placement

Our algorithm revolves around the assumption that placement of a new probe becomes necessary once the variation in either intensity, hue or direction of incoming light exceeds a certain threshold. However, as long as there is no large, sudden change, we want to reuse the same probe as long as possible. In particular, we assume that a probe placement does not have to capture all of the scene geometry.

2.1. Irradiance field search

We start by constructing an irradiance volume, similar to those used in current games for lighting [GSHG98, Tat05]. In an irradiance volume, probes are placed at the vertices of a regular grid, and each probe captures the irradiance distribution function for all directions at the corresponding vertex. We are interested in specular reflections, which correspond to irradiance moment volumes with $n = \infty$. These can be easily computed by rendering a HDR spherical environment map for each grid vertex.

One notable consequence of using irradiance volumes in our approach is that the computation becomes scene and geometry independent as it only works on the probes, but the final placement is also partially restricted due to the fixed sampling. This is explained in more detail in section 2.3.

Once we have computed the irradiance volume, we want to discover the sinks of this field. We do so by analyzing the irradiance gradient field, which approximates the gradient of the irradiance field at each grid point of the irradiance volume. Instead of using a numerical solver to compute the derivatives of the irradiance distribution as proposed in [WH92], however, we use the irradiance probes directly to compute a single vector $\vec{f}$ that approximates the change in overall irradiance. Recall that each probe consists of a spherical environment map, where the pixel (u_i, v_i) captures the irradiance I in the direction $\vec{\omega}_i$. We compute the result vector by summing up all direction vectors for a single grid point, weighted by the luminance of the corresponding direction:

$$\vec{f} = \sum_{\omega_i \in \Omega} \vec{\omega}_i * \text{luminance}(I(\vec{\omega}_i))$$

An example of a so constructed gradient field is shown in figure 2.

Besides its efficiency, the proposed method for estimating the irradiance gradient has another desirable property. Since game levels often consist of tight hallways and unreachable space, the computation of derivatives becomes quite cumbersome due to the frequent occurrence of boundary cases. The vector $\vec{f}$, on the other hand, can be computed at every location where a probe can bee placed. Having these gradients at hand, we can now perform a search for sinks in the irradiance volume.

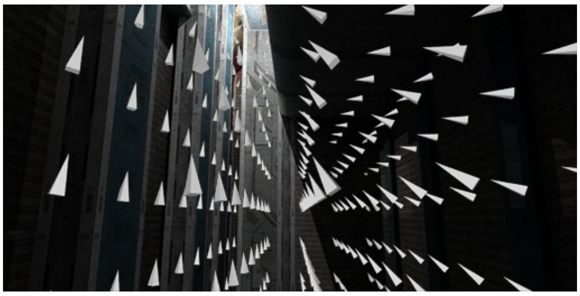
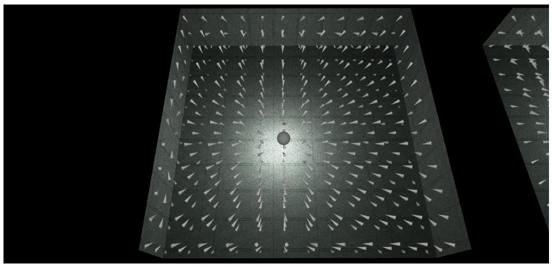
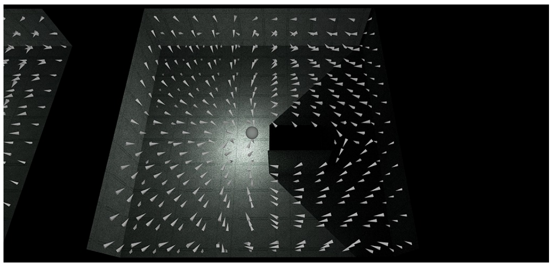


Figure 2: The irradiance gradient field in a dark area of *Alphalabs1*. Each glyph represents the direction of the gradient at a grid cell. The gradients are pointing towards the bright areas near the ceiling.



(a) Empty room with single light source



(b) Same room with a pillar at the center

Figure 3: An empty room with a single light source gets assigned a single probe near the floor in the middle of the room. An additional pillar in the room triggers an error case: While a probe in front of the pillar is placed correctly, no probe is set behind the pillar.

The reason why we search for the sinks can be explained with some intuition for what we are really searching for. Let us therefore assume a large, empty room with a single light in the middle, as can be seen in Figure 3. We want to place a single probe in this room, minimizing the visible error for all locations in the room. Clearly, if we place the probe in one of the corners, it will be wrong for most of the room as both the total reflected energy, as well as the direction change significantly. A search for sources of the gradient field would give this result. Recall that our gradient vectors point towards the light, so a source is local light minimum and hence darker than the surrounding area. In order to minimize the error, we need to place the probe somewhere close to the light – in the center of the room. Searching for the maxima, or in this case, equivalently the sinks of the gradient field, gives exactly the desired placement.

To find the sinks in the irradiance volume we use classical particle tracing in vector fields. From each grid point we start a single particle and integrate it through the gradient field using Runge-Kutta scheme of second order. As soon as we discover an area where multiple particle paths end – that is, the particles fall into a closed loop – we mark the probes as candidates. Conceptually, we perform a discretized gradient ascent which is robust and works well on incomplete volumes.

The tracing results in small pockets, or clusters, of probes. All probes of such a cluster are usually very similar, but some may still exhibit significant variance. For instance, probes around a corner can be partially in shadow and partially lit, so even though they are spatially close the contents will differ significantly.

While we could use each cluster as a probe location, but we can prune the candidate set further and make our approach more robust by using a post-process for cleaning up the clusters, which will be described in the next section.

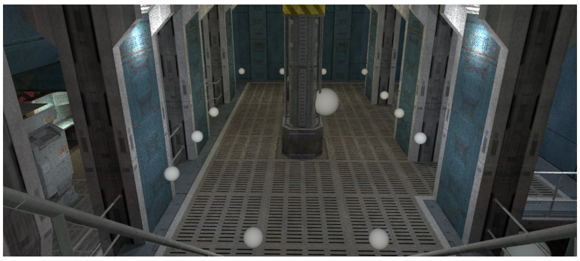


Figure 4: 1D sink manifold formed inside a room. The probes on the 1D line around the room are part of a single cluster, and our algorithm correctly generates a single probe in the center to represent this cluster.

Another common case of clustering is when the sources form 1D or 2D manifolds. For instance, rooms may end up with a line or plane of closely adjacent sources. An example for such a configuration can be seen in Figure 4. In this case, we need a post-process to determine whether the probes can indeed be merged and replaced by a single representative probe.

2.2. Probe matching

As our search process usually results in clusters of probes, we have devised an image-based metric to quickly prune such clusters and to identify those with low confidence. Our metric is based on two separate parts. First of all, we compute the histogram of each probe and use the earth-mover's distance to compare them. In order to reduce the dimensionality of our problem, we use only the luminance and hue instead of three histograms for RGB. By computing luminance and hue histograms upfront, we are able to quickly separate probes with significantly different colors, as well as probes containing bright highlights.

However, it is not enough to discern between probes where only the dominant light direction changes. For instance, the histograms in Figure 5 match each other very well, while the image content is clearly different.

Figure 5: An example case where the histogram metric finds a match. Both images have similar brightness and color, and hence the histogram distance considers them as similar. The FFT metric allows us to distinguish such cases.

We thus employ a second criterion, which is a FFT-based distance metric. For each probe, we compute a low-order FFT and compare the coefficients using

$$dist(\{f_1^0, f_1^1, \ldots\}, \{f_2^0, f_2^1, \ldots\}) = \sum_i \left((\|f_1^i\| - \|f_2^i\|)^2 * c^{M(i)} \right)$$

with the image frequencies $f_j^i \in \mathbb{C}$ of image j and the Manhattan norm $M(i)$ from the DC component of the 2D FFT. Typically, we use the first 25-100 frequencies only. Comparing probes based on their FFT is similar to comparing the spherical harmonic representation; we used the FFT as it is trivial to include additional frequencies and it can be efficiently computed on the GPU.

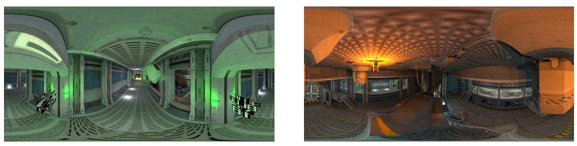


Figure 6: An example case where the FFT distance metric finds a match. Both images have a similar luminance distribution. The histogram metric however distinguishes this images due to the different colors.

2.3. Optimization

The probe matching procedure as described can be used to prune candidate probes, but it has several failure cases which make it unreliable. The main problem is that this method is very sensitive, so it produces many probes and moving a light source slightly can result in vastly different clusters.

On the other hand, perceptually different probes may wind up being classified as similar if there is low frequency variation in luminance and the colors are close under the earth-mover's distance.

Furthermore, probe matching is inherently order-dependent. This can easily be shown by considering a long row of probes: If a new probe is to be placed whenever the difference exceeds a threshold, the order in which the probes are traversed influences the result. For instance, starting from the middle might result in a single probe, while starting from one end could wind up with two probes being created. Finally, the probe matching procedure has no spatial knowledge and hence cannot guarantee good coverage everywhere. Even if spatial distance is taken into account, it is difficult to adapt it to the surrounding scenes – outdoor areas for instance can get away with far less probes than interior scenes.

The gradient field search alone, on the other hand, is not suitable either as it produces large clusters; in particular, it can produce a 1D or 2D manifold with potentially tens or hundreds of probes which are very similar. In such cases, using the gradient field alone is not sufficient to decide which probes can be safely merged and which require the formation of a new cluster.

To overcome these limitations we combine the gradient field search and probe matching process in the following way: We start by computing the probe set, which is an initial dense sampling of the accessible area with low-resolution probes. From these probes, we compute the gradient field and form the "initial cluster set", which contains all candidate locations. Typically, the initial cluster set is already extremely sparse compared to the sampling grid. We clean up the cluster set using our probe matching to form the "final candidate set", where each remaining probe has been assigned to one cluster. For each of these clusters, we eventually estimate a representative probe and store those. The data flow can be seen in Figure 7. Notice in particular that once the probes have been generated, the rest of the algorithm does no longer use the source scene geometry any more.

As we mentioned previously, the initial probe candidates are placed on a fixed grid. This limits the possible locations where a probe can be placed. In order to partially lift this limitation, we compute the average position for each cluster when we are about to place the final probe. As we have no geometric knowledge, an additional image based comparison is used to guarantee that the newly created probe is indeed a good representative for the cluster. This allows us to efficiently resolve cases like the manifold in Figure 4, which requires a new probe on a non-grid location in the middle of the room.

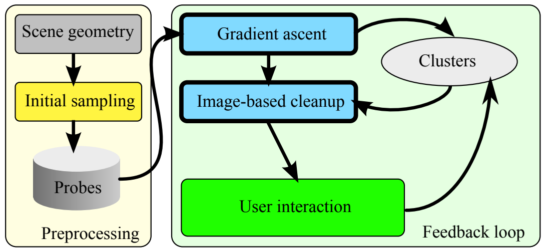


Figure 7: Data flow in our algorithm. After the preprocessing, we only work on the initial probe set. The gradient ascent generates a set of clusters, which is further refined by probe matching. User input is fed back by creating and/or deleting clusters.

In addition to the automatically placed probes, we also provide two additional hints to the artist. First of all, we show the irradiance gradient to the artist so she can quickly understand how the light flows through the scene and in which direction to move. Second, we also show the luminance distance to all surrounding probes. Typically, artists try to minimize the luminance change between adjacent probes. As we know the luminance throughout the whole volume, we can show the locations where the distance becomes too large. This allows artists to resolve the corner cases of our algorithm which are explained in section 3.

2.4. Cluster control

As the gradient field search produces clusters of probes, additional cluster constraints can be integrated into the process at this point quite easily. Clusters which lie inside "importance volumes" can be weighted higher, clusters outside reachable space can be directly pruned, and already existing probes can be compared to nearby clusters. This allows the artist to retain full control where the environment map probes are placed. We can also trivially guide the cluster size by game-data: For instance, if a player spends a significant amount of time in a certain location, we can reduce the spatial extents at which we form clusters and build more clusters to better cover the area. Similarly, we can increase the cluster size for parts where performance is critical, for example, areas where multiplayer profiling shows high player density.

3. Results

To validate our approach, we have tested our algorithm on actual game levels from Doom3 as well as specially crafted test geometry. Even though the Doom3 source data is not as complex as current-generation games, it is a good representative as our algorithm is independent of the actual scene complexity. On the other hand, the level size and the resulting number of probes in Doom3 is still representative for modern games.

We placed probes with 128×64 pixels resolution on a uniform grid throughout the reachable area in the scene. The probe density was set so high that even in tight areas like hallways we would still get 9-12 probes for each cross-section. As the probes are of very low resolution, they can be quickly computed using the in-game renderer or generated during the light baking. Since the time for sample generation highly depends on the game engine and content, we did not include it in our timings and instead focused on the particle tracing and clean-up times.

Figure 3a shows a simple synthetic test case consisting of a single room with one light source at the center. The glyph indicators point into the direction a particle would move during particle tracing. As expected in this case, all particles got drawn towards the light source, resulting in a single probe being placed in the middle of the room.

In Figure 3b we have a similar situation, but now a pillar in the middle of the room occludes parts of the level. In this case, our algorithm still places only one probe in the middle of the room, because particles starting in the pillar's shadow are immediately drawn to the light. This behavior is generally undesirable, as the reflections behind the pillar will be incorrect. Fortunately, such situations are very rare in real-world data, where there are usually at least some smaller ambient lights visible from any point in a level, resulting in additional probes being placed even in darker areas. Nonetheless, completely dark areas are currently not considered by our algorithm and will require the manual placement of additional probes by an artist.

A real-world example from *Alphalabs1* shows a more typical case. At one place in the game level a single bridge leads the player over a deep pit that is completely dark. As can be seen from Figure 2, the computed gradient field leads any particle that starts inside the pit directly to the bridge level. In this particular case, "turbulence" in the gradient field at the bottom of the pit cause a single probe to be placed in the center of the pit near the bottom, resulting in an optimal placement for this area.

Figure 8 shows how probe matching using the image-based metric prevents spatially close clusters from being merged. In this case, the probes from two adjacent rooms are spatially close together and thus considered for merging. However, as the brightness distribution of the probes is significantly different, the image-based metric prevents the clusters from merging.

Since most indoor scenes in games use doors to separate different areas, we have also investigated how the presence of doors affects the outcome of the algorithm and whether the level should be sampled with doors opened or closed. Figure 9 shows the result of this investigation, clearly indicating that the effect of a closed door is negligible for the final result. The influence of doors becomes noticeable only if the amount of light reflected on a door's surface is significantly higher than in its environment.

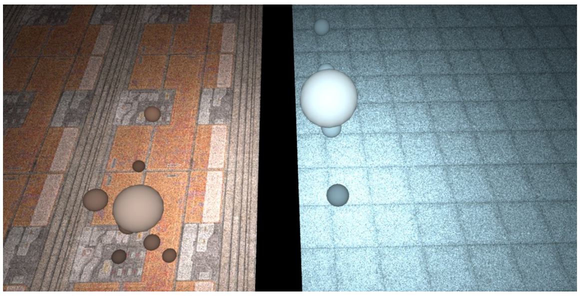


Figure 8: Finding clusters in a simple setup with two different rooms that are separated by walls. The light sources are placed close to the walls. Our algorithm is able to discern between each independent cluster, even though they are spatially close together.

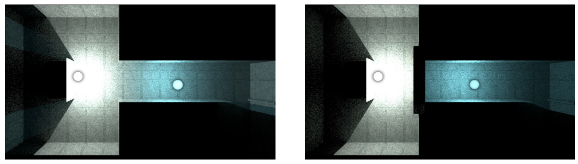


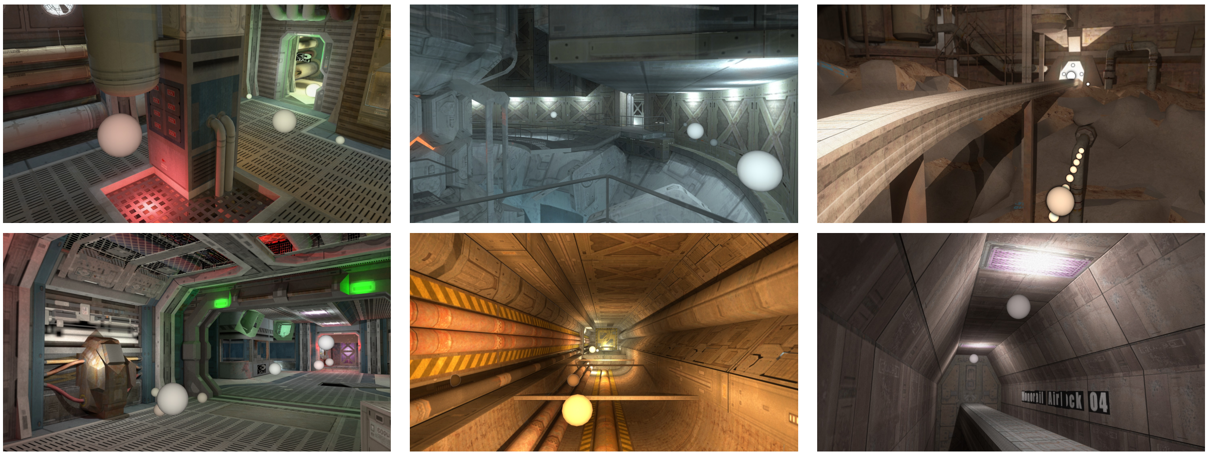
Figure 9: Open or closed doors do not have much impact on the placement, as it mostly depends on the reflected light. The two light sources on both sides of the wall trigger the placement of the probe in every case.

We have tested a representative subset of the Doom3 levels: *EnPro*, *Alphalabs1* and *Monorail*. *Alphalabs1* is a typical claustrophobic space station of tight hallways and corridors. *Monorail*, on the other hand, is dominated by two large outdoor areas, while *EnPro* is a mixture of both tight and open areas.

Some typical placements for these levels are shown in Figure 10. Note that game-specific information was not considered when generating the samples. In particular, probes are usually not generated at the player's eye level and might get placed in unreachable areas. However, this can be fixed easily in a post-processing step.

Figure 10a shows two typical indoor scenes from the *Alpalabs1* level. In the upper picture, three probes were placed in the scene: Two in the main room—one on each side of the central pillar—and a third one in the adjacent hallway. The bright red highlight to the left of the pillar is very prominent and thus triggers placement of a separate probe, while the central probe gives a good overall estimate of the reflection experienced in the rest of the room. The third probe is placed right behind the doorway. This is a typical situation, as the lighting situation changes drastically when stepping into the hallway. The algorithm successfully detects this from the irradiance field trace and adds a probe accordingly.

Similar behavior can be observed in the lower picture. The



(a) Alphalabs 1 (b) Energy processing and storage (c) Monorail

Figure 10: Typical probe placements in our sample levels. Pictures were generated with the same renderer that was used for generating the sample probes in our algorithm. Small glyphs indicate candidates for probes before clustering. Large glyphs indicate the final positions of environment probes as generated by our algorithm.

area is roughly divided into three rooms, with one probe placed at the center of each room. Additional probes get placed near the central cabin in the left part of the main room and under the red highlight at the back-door. These additional probes reflect the different lighting situations in the main room. Notice that the algorithm found a number of potential probe candidates for the airlock in the foreground, which get clustered and eventually result in a single probe being placed in the center of the room.

The next set of samples in Figure 10b shows some wider areas from the *EnPro* level. The upper picture was taken in a large cylindrical room. When moving along the outer wall of the cylinder, the generated environment maps change very slowly. This is because the reflected geometry is very homogeneous but the direction changes from which the light (e.g. the reddish light) at the center reflects. Our algorithm accounts for that, by periodically clustering probe candidates together, thus resulting in a sparse, regular placement of probes along the outer walls.

The lower picture shows a very long lift shaft spanning the whole height of the level. Our algorithm captures the very different light situations at both ends while it places only a single probe in the interior part.

The third sample set from the *Monorail* level is shown in Figure 10c. An importance volume has been placed around the tracks, as the player cannot freely move in this level. This restricts the number of probes significantly, as the level consists of huge empty volumes which are unreachable for the player.

The upper picture shows how our algorithm behaves for an open, outdoor environment. A large manifold of probe candidates is found parallel to the railway track. The image-based metric is able to merge those into a single probe. A second set of probes is found near the door in the back wall, which is flanked by additional light sources. This placement effectively models the reflection seen on the windows of a train moving along the railway tracks. The lower picture shows the placement of probes in a tunnel. Again, probes are placed close to areas of prominent changes in lighting, in this case due to the ceiling lights.

The most important parameter of the algorithm is the sampling density of the initial regular grid. Naturally, if the grid is too coarse, tight spaces will not get sampled correctly. On the other hand, sampling huge volumes of empty space with a high density is a careless waste of computational resources, so those should be sampled at lower densities.

In general, the grid sampling should be roughly in the same order as the character's size. Higher sampling densities will eventually result in more probes being generated, as more subtle features in the level will be detected. This can be a desirable effect if a detailed coverage of the scene is required, for instance, when using the environment maps as irradiance estimate for global illumination.

Table 1 shows some statistics of our algorithm for probe placement. The grid density was adjusted manually based on the overall layout of the level. For instance, the *EnPro* level required a relatively dense sampling to be able to capture the tight hallways accordingly, but the majority of probes was placed in two large vertical volumes in the middle of the level. An adaptive placement could resolve such cases.

As can be seen from the table, the irradiance field search is able to isolate a small subset of grid probes as potential

Scene	# Grid	# Candidate	# Final	t (s)
Alphalabs	8925	205 (2.3%)	117	2
EnPro	17640	208 (1.2%)	80	5
Monorail	11143	166 (1.5%)	29	2
AlphalabsHD	67980	1139 (1.7%)	508	121

Table 1: Statistics showing typical probe counts and processing times. From left to right: Number of sampled grid probes for irradiance approximation; Number of candidates found via particle tracing; Remaining number of probes after clustering; Computation time for particle tracing and clustering.

candidates. Since these candidates tend to form clusters of spatially close probes, the image-based metric is able to further reduce this set by a factor of 2-3. Note that although *Alphalabs* started with far fewer initial grid probes than *EnPro*, it ended up having more probes in the final set. This is mainly due to the fact that large empty volumes in *EnPro* tend to contain many grid probes that do not add any information to the irradiance field.

The computation times in the last column include the time for building the irradiance gradient field, perform the particle tracing and clustering the probe candidates using the image-based metric. They were measured on a 2.8 GHz Dual Xeon X5560 machine (8C/16T); as the algorithm is embarrassingly parallel, we used all cores of the machine. As a special case, we included the numbers for the *Alphalabs* level sampled at double density in each direction to show how the algorithm scales. The higher processing time is mostly related to longer particle tracing time through the denser grid.

Again, the initial sampling density turned out to be the key parameter of the algorithm. For scenarios in which a dense sampling is required, e.g. when generating probes for global illumination, an adaptive grid sampling will help reducing computation times considerably.

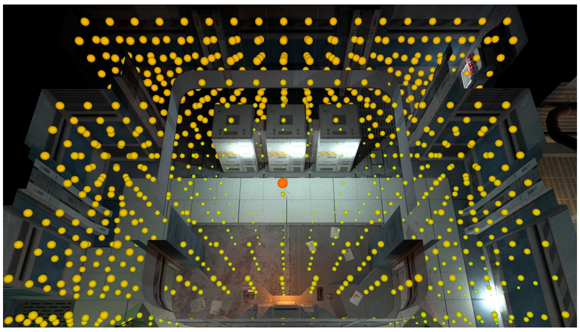


Figure 11: Luminance error field around a single probe in the center of the screen. Probes inside the control room have comparable brightness, while probes behind the boxes are much darker. The visualization helps artists to identify regions where noticeable changes would occur.

We have evaluated the quality of our placement by an artist who worked on Doom3, as well as an artist who was not particularly familiar with the setting. The vast majority of the changes involved moving the probes up to eye-level and away from walls, which we could not do automatically due to lack of player and collusion geometry. The artists indicated that the probe density is very similar to manually placed probes. In the future, we hope to get access to source data with already placed environment map probes for a more thorough validation.

Besides the initial placement, the artists found the gradient field visualization and the luminance distance helpful, which can be see in Figure 11. The latter allows an artist to quickly identify how much of the volume contains probes of similar brightness.

4. Conclusion

We have presented a novel approach to guide artists while placing environment reflection map probes. Our algorithm uses the irradiance gradient field which represents the reflected light as well as an adapted image-space metric which together provide a robust way to place environment map probes in complex game environments.

Our algorithm is an early exploratory approach to solve the problem. In the future, we would like to explore a more adaptive approach which adjusts the initial probe density based on the local geometric complexity. For instance, the sample rate should decrease in large outdoor areas and increase in tight places or situations with complex lighting.

We would also like to resolve the case where extremely dark areas may end up without any samples at all, as all samples have been moved towards the closest light. This failure case can be seen in Figure 3b. While it is possible to get an error indication for each probe by simply comparing the sample to the closest probe, we expect that our algorithm can be modified to directly resolve such cases.

Eventually, we expect that our algorithm could be modified to produce probe locations which are suitable for image-based lighting approaches. This requires a more rigorous treatment of the remaining error.

Acknowledgements

Thanks to Vicarious Visions for helpful comments and result validation. Thanks to Anton Kaplanyan for comments. Special thanks to Alexandru Adrian Radoiu for additional testing and feedback.

References

[BN76] BLINN J. F., NEWELL M. E.: Texture and reflection in computer generated images. *Commun. ACM 19* (October 1976), 542–547. 1

[Cry11] CRYTEK: CryEngine Documentation: Environment probes (http://freesdk.crydev.net/display/SDKDOC2/Environment+Probes), 2011. 1

[Gre86] GREENE N.: Environment mapping and other applications of world projections. *IEEE Comput. Graph. Appl. 6* (November 1986), 21–29. 1

[GSHG98] GREGER G., SHIRLEY P., HUBBARD P. M., GREENBERG D. P.: The irradiance volume. *IEEE Computer Graphics and Applications 18* (1998), 32–43. 2, 3

[Kap11] KAPLANYAN A.: Personal communication with Anton Kaplanyan, 2011. 2

[McT04] MCTAGGART G.: Half-life 2 / valve source shading. Game Developer's Conference, March 2004. 1

[ML03] MEYER A., LOSCOS C.: Real-time reflection on moving vehicles in urban environments. In *Proceedings of the ACM symposium on Virtual reality software and technology* (New York, NY, USA, 2003), VRST '03, ACM, pp. 32–40. 2

[RTG00] RUBNER Y., TOMASI C., GUIBAS L. J.: The earth moverâĂŹs distance as a metric for image retrieval. *International Journal of Computer Vision 40* (2000), 2000. 2

[Tat05] TATARCHUK N.: Irradiance volumes for games. Game Developer's Conference, 2005. 3

[War94] WARD G. J.: The radiance lighting simulation and rendering system. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1994), SIGGRAPH '94, ACM, pp. 459–472. 2

[WH92] WARD G. J., HECKBERT P. S.: Irradiance gradients, 1992. 3

Geometry Independent Surface Light Fields for Real Time Rendering of Precomputed Global Illumination

E. Miandji†[1] and J. Kronander[1] ‡ and J. Unger[1] §

[1]Linkoping University, Sweden

Abstract

We present a framework for generating, compressing and rendering of Surface Light Field (SLF) data. Our method is based on radiance data generated using physically based rendering methods. Thus the SLF data is generated directly instead of re-sampling digital photographs. Our SLF representation decouples spatial resolution from geometric complexity. We achieve this by uniform sampling of spatial dimension of the SLF function. For compression, we use Clustered Principal Component Analysis (CPCA). The SLF matrix is first clustered to low frequency groups of points across all directions. Then we apply PCA to each cluster. The clustering ensures that the within-cluster frequency of data is low, allowing for projection using a few principal components. Finally we reconstruct the CPCA encoded data using an efficient rendering algorithm. Our reconstruction technique ensures seamless reconstruction of discrete SLF data. We applied our rendering method for fast, high quality off-line rendering and real-time illumination of static scenes. The proposed framework is not limited to complexity of materials or light sources, enabling us to render high quality images describing the full global illumination in a scene.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

1. Introduction

The ongoing pursuit for virtual realism has incited many researchers for efficient and accurate modelling of the interaction of light between surfaces. The complexity of analytical models for spatially varying surface properties limit their usage for real-time rendering. Due to this limitation, many Image Based Rendering (IBR) techniques were introduced to directly acquire the appearance of a scene through captured images [Zha04]. A successful appearance description model, Surface Light Field (SLF), was introduced in [MRP98]. The SLF function is defined as $f(r,s,\theta,\phi)$; where r and s are parametric coordinates for addressing a point on a surface. θ and ϕ are used for representing a direction in spherical coordinates. Depending on the sampling density of this function, the data generated by this method easily exceeds the capabilities of modern hardware even for a moderately detailed scene. Therefore various compression methods has been widely used to reduce the SLF size for rendering.

In the context of radiometry in computer graphics, one can see a surface light field as a set of exitant radiance values for each point on a scene along every possible direction. The radiance can be resampled data from High Dynamic Range (HDR) images or computer generated radiance data based on physically based rendering techniques. In the case of computer generated radiance data, by placing a virtual camera anywhere in the scene we can simply look up the SLF data based on intersection point of the viewing ray with the scene and the direction of it. We utilize this observation in order to present a SLF-based framework for fast real-time rendering of static scenes with all-frequency view-dependent radiance distribution. Our method (Section 3) is divided into three stages: data generation (Section 3.1), compression (Section 3.2) and rendering (Section 3.3). We remove the resampling stage from [CBCG02] and instead compute the outgoing radiance of uniformly sampled points

† ehsmi345@student.liu.se
‡ joel.kronander@liu.se
§ jonas.unger@itn.liu.se

on each surface in the scene along different directions. We then compress the SLF data using Clustered Principal Component Analysis (CPCA), similar to [SHHS03]. The renderer can efficiently decompress and reconstruct the SLF data on the GPU (Section 4). We will present our results for a real-time and an offline renderer (Section 5)

Since our SLF approximation method is not based on surface primitives (vertices, faces and edges), having low tessellated geometries does not affect the rendering quality. Instead we use uniform sampling of surfaces which leads to decoupling of lighting from geometric complexity. For instance, a highly tessellated portion of a polygonal mesh may be a diffuse reflector having uniform radiance values. In this case, a lot of memory is dedicated for a very low frequency lighting data. Since lighting complexity is purely scene dependent and cannot be determined before rendering (specially for glossy and specular objects), we use dense uniform sampling to ensure that we do not under-sample SLF function. Then we cluster this data, taking advantage of the fact that for most scenes the within-cluster frequency is low [MSRB07], therefore allowing us to approximate them with low order of principal components.

The gap between the image quality of photo realistic offline renderers and the state of the art real-time renderers is our main motivation. Complexity of certain materials at micro-scale is beyond the capability of current hardware for real-time rendering using analytic solutions. Our proposed framework is not limited to complexity of materials or light sources. Our method supports first order lighting from any type light source, e.g. point, spot and area lights. The compression and rendering stages allow us to render all-frequency view-dependent lighting effects in real-time for static scenes. To summarize, we state our main contributions:

1. A flexible representation of SLF that decouples radiance data from geometric complexity.
2. Application of CPCA in efficient compression of SLF data while preserving view-dependent high frequency details.
3. An efficient real-time rendering method for CPCA generated data.
4. A fast power iteration method adapted for CPCA

2. Related Work

In computer graphics, the ultimate goal is to sample a dataset (3D world) and then reconstruct or equivalently render this data. There are two models for this purpose; *source description* and *appearance description* [Zha04]. The former requires mathematical models in order to describe the 3D world. Reflection models, geometric models such as polygonal meshes and light transport models are examples of source descriptors. The data for this model is computed using mathematical models and during rendering they are reconstructed. The latter is based on capturing data from a real environment using cameras or similar equipment. The plenoptic function [AB91], defined as $l^{(7)}(V_x, V_y, V_z, \theta, \phi, \lambda, t)$, is used for representing such model; the first three arguments define a point in space where a camera is placed, θ and ϕ define a direction, λ is the wavelength of the light rays towards the camera and t represents time. The goal of many Image Based Rendering (IBR) techniques is to simplify this function by applying certain assumptions for practical sampling and rendering [Zha04].

Surface light fields was first introduced in [MRP98] as an IBR technique for visualizing results of precomputed global illumination. They formulated this representation for closed parametric surfaces but used polygonal surfaces for practical sampling of surfaces. Spatial samples were placed on vertices and interpolated over the triangle. For directional samples they subdivide a polygon if it is more that eight pixels in screen space. By representing the SLF data as an array of images, block coding techniques were utilized for compression. Our method is similar to [MRP98] regarding the utilization of precomputed global illumination results, but differs in data generation, compression and rendering.

Chen et. al. [CBCG02] introduced a new approximation of SLF by using vertex-centered partitioning. Each part was projected into lower dimensional functions using PCA or NMF, resulting in a surface map and a view map. They tile and store surface and view maps in textures and compress the results further using Vector Quantization (VQ) and standard hardware accelerated texture compression methods. Unlike [MRP98], Chen et. al. used 3D photography for acquiring a set of images. This technique is based on using geometric models to assist re-sampling of captured images in order to evaluate the SLF function. Utilizing hardware accelerated interpolation between SLF partitions, they could achieve real-time performance. Similarly, in [WBD03] a bi-triangle or edge-based partitioning was introduced.

Lambert et. al. [LDH07] propose a sampling criterion in order to optimize the smoothness of outgoing radiance in the angular domain of SLF. This criterion eliminates the use of actual surface in the SLF definition, replacing the surface with a parametrization of SLF function. A seminal work in SLF rendering was introduced in [WAA*00]. They propose a framework for construction, compression, rendering and editing SLF data acquired through 3D photography. The compression was performed using a generalizations of VQ and PCA. The framework can achieve interactive performance on the CPU using a new view-dependent level-of-detail algorithm. The editing supports linear modification of surface geometry, changes in reflectance properties and transformation relative to the environment.

The study of compression methods is not limited to IBR literature. A machine learning compression method [KL97, TB99] was employed in [SHHS03] for compressing the precomputed radiance transfer (PRT) data. Referred to as CPCA, it is a combination of VQ and PCA. The signal is partitioned into clusters and transformed to an affine subspace. Compressing radiance transfer data is an active research field. In [MSRB07], an insightful study was performed to determine how light transport dimensionality increases with the cluster size. They show that the number of basis functions for glossy reflections is augmented linearly relative to cluster size. This study resulted in determining the optimal patch size for all-frequency relighting of 1024×1024 images. Ruiters and Klein applied tensor approximation to compress Bidirectional Texture Functions (BTF) with a factor of 3 to 4 better than PCA [RK09]. Tensor approximation was also used for compression of PRT data [TS06].

A thorough review of PRT can be found in [KSL05, Ram09]. Compared to these methods, our approach supports first order lighting from any type of light source such as point, spot and area lights. Although recent research such as [KAMJ05] add local lighting support to PRT, our method has this advantage inherently. Additionally, it can be used for all-frequency view-dependent effects that cannot be rendered faithfully with PRT because of projection on SH basis functions.

3. Method

In this section we present a detailed overview of our SLF compression and real-time rendering method. The technique can be outlined by the following steps:

1. We start by uniformly sampling the surface of a mesh in texture space, creating a set of points. For each point, we generate a number of directions on the unit sphere centered at the point. Then we evaluate outgoing radiance.
2. The data is clustered based on the difference between the radiance of points in all directions. We apply PCA on each cluster and store the results to disk
3. During rendering and for each ray, we fetch and decompress the radiance data corresponding to intersection point and the direction of ray

In the following subsections (3.1, 3.2 and 3.3) we will discuss each of the three main steps in more detail.

3.1. Data Generation

Sampling the SLF function requires discretization of spatial and directional parameters. We choose the two dimensional texture space for parameterizing the spatial coordinates. This is shown in Figure 1 (a) and (b). Given the number of samples for each coordinate of texture space, we use a rasterizer to determine world space points on a surface corresponding to sampled texture coordinates. The rasterizer uses barycentric coordinates to determine the world space coordinate of a point inside a triangle by interpolating vertex positions based on texture coordinates.

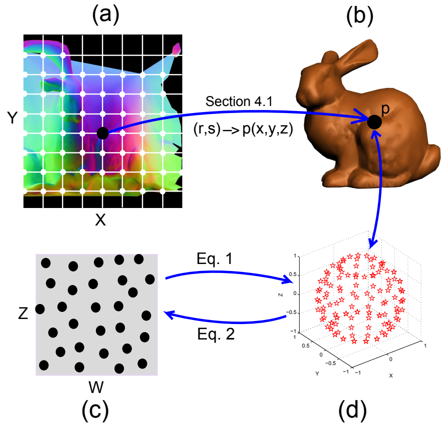


Figure 1: *A schematic illustration of data generation stage. (a) is texture space of the Stanford bunny model discretized with resolution X and Y, (b) illustrates a world space point on the mesh corresponding to a sample (r,s). The parameter space of directions sampled uniformly with resolution Z and W is shown in (c). And (d) illustrates a set of directions on the unit sphere.*

For sampling the unit sphere we need a two dimensional space of variables on the interval $[0,1]$ with respect to solid angle [PH04]. We define samples in this space and map them to 3D directions on the unit sphere. During rendering, the inverse mapping should be applied for SLF addressing. The forward and inverse mapping are shown in Figure 1 (c) and (d). For this purpose, we use the Latitude-Longitude formula in [RHD*10]. The forward mapping is expressed as:

$$\begin{aligned}(\theta,\phi) &= (\pi(\xi_1-1),\pi v),\\ (D_x,D_y,D_z) &= (sin\phi sin\theta, cos\phi, -sin\phi cos\theta),\end{aligned} \tag{1}$$

where $\xi_1 \in [0,2]$ and $\xi_2 \in [0,1]$ are uniform variables with constant spacing. ξ_1 and ξ_2 are mapped to azimuth and elevation angles, respectively. The bigger interval for ξ_1 is handled explicitly by multiplying a set of uniform variables on the interval $[0,1]$ by 2. Consequently, for backward mapping we have:

$$(\xi_1,\xi_2) = (1+\frac{1}{\pi}atan2(D_x,-D_z),\frac{1}{\pi}arccosD_y) \tag{2}$$

Also note that we sample a sphere instead of a hemisphere. This eliminates the need for converting every ray to local coordinate frame of a point during rendering with the drawback of having larger data. Since the compression method approximates all directions with a few principal components ($k << n$), this choice of sampling increases the rendering performance and does have a small impact on size of the data.

Having the position map and a set of outgoing directions for each point, we evaluate the outgoing radiance. We unwrap the discretized spatial (r and s) and directional dimensions (ϕ and θ) defining a matrix, F, where each row corresponds to a surface point and each column represents a direction. In other words, each row contains radiance for a point along all directions; and each column represents radiance along a specific direction for all surface points. Although many representation of our 4D data are possible (such as tensors), this type of representation facilitates our compression stage where we cluster this matrix and apply PCA on each cluster. To dicretize wavelength λ, we create three matrices F_R, F_G and F_B each storing radiance data for a color component. Compression is applied to each matrix separately. In the remainder of this paper we ignore wavelength dimension and denote the SLF matrix as F.

3.2. Compression

In this section, we discuss the compression of the generated SLF. The requirements for compression are as follows:

1. *High compression ratio*: Because of the large amount of data stored in the SLF data structure, an algorithm with high compression ratio is vital for fitting this data in the system memory and ultimately the GPU memory.
2. *Faithful decoding and reconstruction of data*: A scene may include diffuse surfaces (low frequency radiance variation along the surface) along with highly specular surfaces or even caustics (which exhibit very high frequency radiance variations). The SLF contains high frequency data in both spatial and angular domain. This puts high requirements on a compression method that can reconstruct high frequency data faithfully.
3. *Random access*: During the rendering stage it is required that the data be accessed randomly. This means that the decompression method should be able to locally decode the data and return the requested radiance value in an effective manner.

The widely used PCA method exhibits high compression ratio and enables random access to data; yet it cannot reproduce high frequency angular variations of radiance present in specular and glossy surfaces. For diffuse surfaces it can reproduce smooth surfaces with only one principal component, satisfying all three requirements to a great extent.

To this end, we use Clustered Principal Component Analysis (CPCA). A fast and versatile algorithm adapted from machine learning literature [KL97, TB99] to the field of computer graphics by Sloan et al. [SHHS03]. CPCA is a two stage method of clustering followed by a PCA for each cluster. When there is only one cluster, it is equivalent to PCA. This method has a high compression ratio and will provide random access of compressed data in real-time and on the GPU. The clustering part of CPCA will compensate for the reconstruction error of the compressed all-frequency SLF data by creating clusters of low frequency radiance data; allowing us to recover view-dependent details present in specular and glossy surfaces. The quality of reconstruction is highly dependent on the number of principal components and clusters; which increases the flexibility of the algorithm.

We calculate the mean of each row of F and store it in a *mean map*, denoted as μ. Then we calculate the matrix of residuals, G, via

$$G = [x_{p1} - \mu_{p1}, x_{p2} - \mu_{p2}, \ldots, x_{p_m} - \mu_{p_m}]^T, \tag{3}$$

The normalizing stage can also be done after clustering by subtracting the mean from each cluster. We cluster the SLF matrix using the K-Means method. Rows of G are treated as data items and columns as variables. The result of clustering is a set of cluster IDs with a size equal to the number of points. The outcome will be a set of matrices G_i of size $m_i \times n$ where $m_i \ll m$ is the number of points belonging to cluster i that have similar radiance distribution along all sampled directions. Note that m_i is not constant among all clusters. Each cluster may have different number of points.

Denoting a cluster's normalized SLF matrix as G_i, we write the Singular Value Decomposition (SVD) of it as $G_i = U_i D_i V_i^T$; where U_i is a $m_i \times k$ matrix of left singular vectors, V_i is a $n \times k$ matrix including right singular vectors and D_i is a diagonal matrix of singular values sorted in decreasing order. When $k < N$ we achieve a least-squares optimal linear approximation $\hat{G}_i = U_i D_i V_i^T$ of G_i where the approximation error, [SHHS03], is:

$$\sum_{j=1}^{m_i} \|x_{p_j} - \hat{x_{p_j}}\|^2 = \sum_{j=1}^{m_i} \|x_{p_j} - \mu_{p_j}\|^2 - \sum_{j=1}^{k} (D_i)_j^2 \tag{4}$$

This decomposition can be thought of as selecting a set of orthonormal eigenvectors representing a subset of directions and linearly approximating every other direction by calculating a weighted sum of eigenvectors. The weights are rows of $U_i D_i$ and principal components are rows of V_i. Since we are interested in the first few approximation terms ($k \ll n$), we use Power iteration method [CBCG02]. This method is well-suited for our technique for the following reasons:

- This method can iteratively calculate first k terms instead of performing a full SVD. This saves a lot of computation time
- The approximation is improved by adding more terms without the requirement for recalculating previous terms.
- The memory footprint is very low. A careful implementation of Algorithm 1 (see Appendix) will require extra memory allocation of size $n^2 + m + 2n$ when $m \gg n$. This is constant for all approximation terms.

In this method, left and right eigenvectors of the covariance matrix is calculated for each term, and iterated for all approximation terms k. The traditional power iteration method presented in [CBCG02] assumes that $m_i \gg n$, therefore the covariance matrix, $G_i^T G_i$, is of size $n \times n$. In the clustered PCA method [CPCA], it is very likely that each cluster will contain fewer points than directions. This will increase the PCA computation time dramatically since we have to apply PCA on each cluster. Instead, we modify power iteration according to [SHHS03], handling situations when a cluster has more variables rather than data items. The modified power iteration algorithm is presented in the Appendix.

We denote U_i as the surface map of cluster i and V_i as view map of cluster i. The number of points within clusters add to total number of points for a shape. Therefore we create one surface map U for each shape. On the other hand, each cluster has a unique view map V_i. In order to find the corresponding view map, we add a cluster index for each point in the surface map. Defining the size operator as Γ, the total memory consumption (γ) for this representation is

$$\begin{aligned}\gamma &= \Gamma(U)+\Gamma(V)+\Gamma(\mu)+\Gamma(\chi),\\ &= [\zeta mk]+\sum_{i=1}^{\eta}[\zeta nk]+[\zeta m]+[m],\end{aligned} \tag{5}$$

where χ is a vector containing cluster indices, ζ is number of color components and η is number of clusters.

3.3. Rendering

The rendering method presented here can be easily implemented for both CPU-based (offline) and GPU-based (real-time) renderers. In the offline case, a ray caster is used for shooting rays from the camera into the scene and extracting the corresponding radiance value of the screen sample being processed. The same procedure is applied to the real time renderer but with the difference that we fetch and reconstruct data in a pixel shader based on the view vector. The data is stored in volume textures for cache efficient access.

In order to get radiance of a point along a viewing ray, we need to reconstruct the SLF matrix, F, which is a discretization of the SLF function, $f(r,s,\theta,\phi)$:

$$f(r,s,\theta,\phi)\approx F[r,s,u,v]=\left(\sum_{i=1}^{\eta}U_iV_i^T\right)+\mu, \tag{6}$$

where U_i corresponds to the surface map of cluster i and V_i is the view map of cluster i. Here $[r,s]$ and $[u,v]$ are parametric space coordinates of a point and direction which are mapped to address the SLF matrix.

One advantage of this representation is that we can directly compute an element in F without the need for complete reconstruction of it. We define α and β to be a row in surface map and view map, respectively. In this way, α represents a surface map row index of a point (r,s) and β represents a view map row index of a direction (θ,ϕ). Then, we can compute the SLF matrix element $F[r,s,u,v]$ as

$$F[r,s,u,v]=\left(\sum_{j=1}^{k}U_i[\alpha,j]V_i[\beta,j]^T\right)+\mu[\alpha], \tag{7}$$

where $U_i[\alpha,j]$ is an element in the surface map of cluster i at row α and column j; similarly, $V_i[\beta,j]$ is an element in view map of cluster i at row β and column j. The summation in Equation (7) is an inner product between a row in U_i and V_i. Note that in practice we have one surface map U and several view maps V_i, $1<i<\eta$. Hence, $U_i[:,j]$ with $j=1\dots k$, represents all the principal vectors inside U that have cluster ID equal to i. Additionally, Equation 7 allows us to render a shape in one pass. It is not required to render each cluster separately (Section 4). As a result, the need for super-clustering is removed. This technique was introduced in [SHHS03] to reduce the overdraw of a triangles with vertices belonging to different clusters.

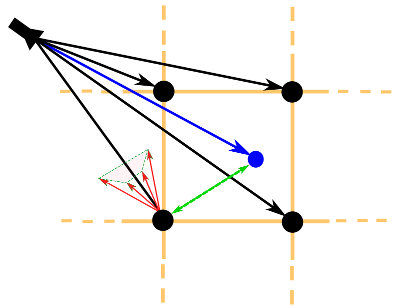


Figure 2: *Schematic view of SLF reconstruction during rendering*

To apply Equation (7) to a ray tracer or a GPU based scanline renderer, we need to convert a point $p(x,y,z)$ and a direction $d(x',y',z')$ to surface and view map row indices (α and β). To address surface map, we fetch texture coordinates of p (the intersection point). Then we find four nearest points in the surface map. Due to uniform sampling, this can be easily done by clamping or rounding ι_1X and ι_2Y, where ι_1 and ι_2 are interpolated texture coordinates of point p; X and Y are sampling resolution of texture space. The final surface map value for p is calculated by bi-linear interpolation of four neighboring points. As described in Figure 2, the blue arrow and circle correspond to the main view vector and its intersection point with the surface, respectively. The black arrows and circles are four nearest neighbors to the intersection point. The weights are calculated as the inverse of the Euclidean distance between (ι_1,ι_2) for main intersection point and the four nearest points. This is shown for a point in Figure 2 as the two-sided dashed green line.

To address the view map, we apply a similar procedure. We use the same neighboring points in order to calculate four direction vectors. For the main direction and each of the four additional direction vectors, we first calculate parameter space coordinates of them by applying Equation (2), getting ξ_1 and ξ_2. Then, each (ξ_1,ξ_2) coordinate set is discretized as (ξ_1Z,ξ_2W), again resulting in four view map indices for each. This is shown as red arrows for one of the nearest points in Figure 2. For simplicity of the illustration, we did not include red arrows for other neighboring points. In order to interpolate view map values of a point, the weights are calculated as the difference between the main direction vector and the four nearest direction vectors (red arrows in Figure 2). This is expressed as $w_i=e^{d.d_i}$, for $i=0\dots3$; where $|.|$ represents a dot product and d_i are nearest direction vectors.

The final view map value of the main direction is a weighted average with weights calculated before for spatial interpolation. This type of interpolation leads to seamless reconstruction of the SLF function (Section 5).

4. Implementation

The data generation stage was implemented as a renderer (C++ class) in PBRT [PH04]. We chose to implement a renderer rather than a surface integrator in order to experiment with different surface integrators. The renderer's input is a set of shapes flagged as SLF and non-SLF. For a SLF shape, we generate SLF matrix, compress it and then store both compressed and non-compressed data to disk. If a shape is flagged as non-SLF, we ignore it during data generation. Data generation parameters are divided in two parts. Parameters such as sphere sampling and compression methods that are unique for all shapes are provided by the renderer. On the other hand, $[X,Y,Z,W,k,\eta]$ are provided per shape. Therefore one can define various settings based on material complexity of a shape. We will use the same notation when presenting our rendering results. The renderer first computes the position map, a two dimensional array of points (Section 3), followed by a bucket of outgoing directions for each point. The point and the bucket of directions are given to a thread for computing outgoing radiance values using a surface integrator. Afterwards, the data is stored in F_R, F_G and F_B at a row corresponding to the point; then the thread terminates. This is iterated XY/q times, where q is the number of available processors.

The K-Means was performed by an optimized multi-processor implementation, provided by Wei-keng Liao (http://users.eecs.northwestern.edu/~wkliao/). We implemented the modified power iteration (Algorithm 1) using the Eigen 3 library (http://eigen.tuxfamily.org) for matrix and vector algebra. The error tolerance, ε, was set to 1e-12 and the maximum number of iterations, c, to 1000. The parameter c ensures finite loops and since we handle numerical fluctuations by monitoring the error, it is guaranteed that v_p for $p = c$, will have the least error.

We implemented a PBRT based renderer for offline rendering and a GPU based renderer using DirectX. For offline rendering, we do not use a surface integrator since the incoming radiance to the camera can be directly acquired from compressed SLF data. Providing a scene containing SLF and non-SLF shapes, we compute radiance for rays that intersect SLF shapes. To evaluate outgoing radiance for non-SLF shapes, we evoke the default integrator, e.g. path tracing. Our implementation does not support light transport between SLF and non-SLF shapes although they can exist in the same scene.

For real-time rendering, we store surface and view maps in 3D textures. Each slice of this texture contains an approximation term for a surface or view map. We can also store each approximation term in separate textures. But due to hardware limitations, this will limit the number of PCA terms, k, to a small value. Another advantage of using a volume texture is that we can change the number of PCA terms in real-time; that is, specifying a smaller value than k in Equation 7 or equivalently sampling fewer slices from the volume texture. As mentioned earlier, the compression stage generates one surface map and η view maps, where η is the number of clusters. Therefore, the size of the 3D surface map will be $X \times Y \times k$. For the 3D view map the size is $Z \times W \times \lceil\sqrt{\eta}\rceil$.

We tile individual view maps for each cluster in a sequential order. The same pattern is used for additional slices. There are two ways for addressing individual view maps inside the 3D view map during rendering. We can create a 2D texture, cluster map, of size $\lceil\sqrt{\eta}\rceil^2$ with two components where each element points to the top-left corner of a view map across all slices. Correspondingly, we can calculate the address in the pixel shader given η, Z and W. The first method is less expensive considering the fast texture lookup in modern hardware; it is also more straightforward to implement.

We implemented the real-time renderer utilizing DirectX 9.0. The surface, view and mean maps are 128-bit, 4-channel floating point textures. Cluster IDs are encoded in alpha channel of the mean map while the value for $\lceil\sqrt{\eta}\rceil$ is provided as a global parameter. Consequently, after fetching a mean map texel and extracting its alpha value, we can sample the cluster map. Returned values are texture addresses for top-left corner of a tile in view map. Having this base address, we fetch the exact texel inside a tile by adding the scaled ξ_1 and ξ_2 values in Equation 2. Because of HDR textures, the real time renderer requires a tone mapping operation as the final stage.

5. Results

Rendering results of our method is shown in Figure 3. The scene consists of a diffuse Cornell box, a diffuse torus and two specular spheres. For specular surfaces, the roughness parameter is set to 0.02. The mesh for the pink specular sphere is distorted intentionally. The left wall is textured with a relatively high frequency texture. As it can be seen in Figure 3, our uniform sampling method can reconstruct this texture with minimum aliasing or noise, despite the fact that it has four vertices only. Previous methods based on surface primitive partitioning of SLF fail on cases like this. Data generation and compression parameters for each shape is included in Table 1. In addition, we used a box filter of width 5 for interpolating radiance along super sampled sphere directions (Section 3.1). Sphere samples were generated using low-discrepancy sampler in PBRT. For the reference image we used the photon mapping integrator with 400000 photons and 64 final gathering samples. The resolution was set to 1024×1024 and we used 256 samples per pixel. Our CPU based renderer was configured accordingly but with the difference that we used only 8 samples per pixel. This value

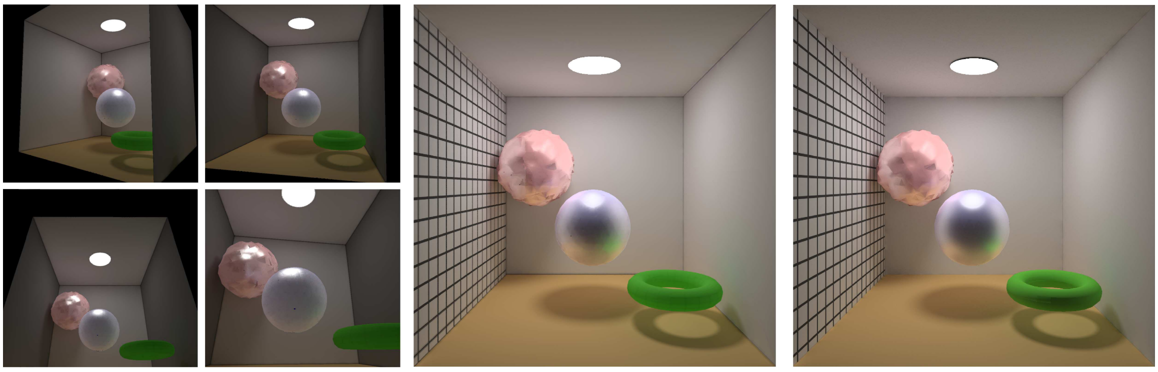


Figure 3: *Rendering results for our GPU based renderer (a), the CPU based renderer (b) and the reference image (c)*

Table 1: *Per-shape parameters and timing results for three stages of our method.*

Shape	X	Y	Z	W	k	η	Data Gen.	Comp.	CPU Rend.	GPU Rend.	Ref. Rend.
Walls (average)	256	256	32	16	1	1	144 min	1.7 sec	-	-	-
Torus	256	256	32	16	1	1	58 min	1.7 sec	-	-	-
Blue sphere	256	256	64	32	8	256	946 min	24 min	-	-	-
Pink Sphere	256	256	64	32	8	256	832 min	27 min	-	-	-
The scene	-	-	-	-	-	-	1980 min	51 min	16.7 sec	120 FPS	202 min

was enough since our rendering method internally interpolates radiance in spatial and angular dimensions. The image resolution for GPU renderer was set to 1920×1080.

Our performance results for three stages of the algorithm are illustrated in Table 1. The rendering parameters are the same to those mentioned earlier and the data generation stage uses the same photon mapping parameters. As illustrated, for diffuse surfaces we used the least amount of clusters and PCA terms ($k = 1$ and $\eta = 1$). For specular shapes, we set $k = 8$ and $\eta = 512$. Although we can increase k and reduce η while getting the same image quality, according to Equation 7 it will result in severe performance lost. Additionally, increasing k will affect the size of view map and surface map (Equation 5) while η only has impact on the view map. Comparing the rendering time for the reference renderer with our CPU renderer, we can conclude that our renderer can be used for fast realistic rendering of novel views of static scenes. We tested our results using a PC with a quad-core Intel Xeon processor and a NVIDIA GeForce 8800 Ultra. Using a PC with more cores will improve data generation and compression performance linearly due to the fact that all the stages utilize parallel processing.

6. Conclusions and Future Work

We presented a framework for creating, compressing and rendering SLF data that can be used for viewing static scenes with global illumination effects. Our SLF representation decouples radiance data from geometric complexity by using uniform sampling of spatial dimension. We also showed that the application of CPCA on SLF data can lead to relatively high compression ratios while preserving view-dependent high frequency details. Additionally, we presented an efficient rendering algorithm that can be used for real-time or fast off-line rendering of scenes with complex materials. Although we focused on computer-generated radiance data, the compression and rendering algorithm can be simply applied to re-sampled data of captured digital images.

Our future work is mainly concentrated on compression, rendering and interpolation. We seek to analyse various compression techniques on SLF data. Of course this is dependent on the representation of discretized SLF function. Whether we see it as a tensor or matrix, different compression techniques can be applied. Wavelet analysis, Tensor approximation [RK09, TS06] and sparse representations [RK09, BE08] have been successfully applied for compression. Using our presented representation, we can compress the surface map and the set of view maps further by utilizing aforementioned techniques. Having a smaller compressed data with minimal loss allows us to increase the size of uncompressed SLF matrix by sampling more densely, leading to better image quality with little rendering overhead.

References

[AB91] ADELSON E. H., BERGEN J. R.: The plenoptic function and the elements of early vision. In *Computational Models of Visual Processing*, Landy M. S., Movshon A. J., (Eds.). MIT Press, Cambridge, MA, 1991, pp. 3–20. 2

[BE08] BRYT O., ELAD M.: Compression of facial images using the k-svd algorithm. *J. Vis. Comun. Image Represent. 19* (May 2008), 270–282. 7

[CBCG02] CHEN W.-C., BOUGUET J.-Y., CHU M. H., GRZESZCZUK R.: Light field mapping: efficient representation and hardware rendering of surface light fields. *ACM Trans. Graph. 21* (July 2002), 447–456. 1, 2, 4

[KAMJ05] KRISTENSEN A. W., AKENINE-MÖLLER T., JENSEN H. W.: Precomputed local radiance transfer for real-time lighting design. *ACM Trans. Graph. 24* (July 2005), 1208–1215. 3

[KL97] KAMBHATLA N., LEEN T. K.: Dimension reduction by local principal component analysis. *Neural Comput. 9* (October 1997), 1493–1516. 2, 4

[KSL05] KAUTZ J., SLOAN P.-P., LEHTINEN J.: Precomputed radiance transfer: theory and practice. In *ACM SIGGRAPH 2005 Courses* (New York, NY, USA, 2005), SIGGRAPH '05, ACM. 3

[LDH07] LAMBERT P., DESCHENES J.-D., HEBERT P.: A sampling criterion for optimizing a surface light field. In *3-D Digital Imaging and Modeling, 2007. 3DIM '07. Sixth International Conference on* (aug. 2007), pp. 47 –54. 2

[MRP98] MILLER G. S. P., RUBIN S. M., PONCELEON D. B.: Lazy decompression of surface light fields for precomputed global illumination. In *Rendering Techniques* (1998), Drettakis G., Max N. L., (Eds.), Springer, pp. 281–292. 1, 2

[MSRB07] MAHAJAN D., SHLIZERMAN I. K., RAMAMOORTHI R., BELHUMEUR P.: A theory of locally low dimensional light transport. *ACM Trans. Graph. 26* (July 2007). 2, 3

[PH04] PHARR M., HUMPHREYS G.: *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. 3, 6

[Ram09] RAMAMOORTHI R.: Precomputation-based rendering. *Found. Trends. Comput. Graph. Vis. 3* (April 2009), 281–369. 3

[RHD*10] REINHARD E., HEIDRICH W., DEBEVEC P., PATTANAIK S., WARD G., MYSZKOWSKI K.: *High Dynamic Range Imaging, Second Edition: Acquisition, Display, and Image-Based Lighting*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2010. 3

[RK09] RUITERS R., KLEIN R.: Btf compression via sparse tensor decomposition. *Computer Graphics Forum 28*, 4 (jul. 2009), 1181–1188. 3, 7

[SHHS03] SLOAN P.-P., HALL J., HART J., SNYDER J.: Clustered principal components for precomputed radiance transfer. *ACM Trans. Graph. 22* (July 2003), 382–391. 2, 4, 5

[TB99] TIPPING M. E., BISHOP C. M.: Mixtures of probabilistic principal component analyzers. *Neural Comput. 11* (February 1999), 443–482. 2, 4

[TS06] TSAI Y.-T., SHIH Z.-C.: All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. *ACM Trans. Graph. 25* (July 2006), 967–976. 3, 7

[WAA*00] WOOD D. N., AZUMA D. I., ALDINGER K., CURLESS B., DUCHAMP T., SALESIN D. H., STUETZLE W.: Surface light fields for 3d photography. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., pp. 287–296. 2

[WBD03] WENDONG W., BAOCAI Y., DDEHUI K.: Non-uniform light field compression: a geometry-based method for image processing. In *Neural Networks and Signal Processing, 2003. Proceedings of the 2003 International Conference on* (dec. 2003), vol. 2, pp. 1058 – 1061 Vol.2. 2

[Zha04] ZHANG C.: A survey on image-based renderingâĂŤrepresentation, sampling and compression. *Signal Processing: Image Communication 19*, 1 (jan. 2004), 1–28. 1, 2

Appendix: Modified Power Iteration

Let F be a $m \times n$ matrix that represents SLF data of a cluster. Computing SVD of F yields $F = UDV^T$. Our goal to compute a $m \times k$ matrix UD and a $n \times k$ matrix V in a way that $\hat{F} = UV^T$ best approximates original matrix F. The power iteration method computes the first k eigenvectors of the matrix $A = F^T F$, the covariance matrix of size $n \times n$. When $m < n$, we compute the $m \times m$ matrix FF^T. Now the eigenvectors are F's left singular vectors and the right singular vectors can be computed as $V^T = U^T D^{-1} F$.

Algorithm 1: Calculate $\hat{F} = UV^T$ where F is $m \times n$, U is $m \times k$ and V is $n \times k$

Require: $m > 0$, $n > 0$, $k > 0$, c is maximum number of iterations for v_p or u_p convergence and ε is the error tolerance
if $M \geq N$ **then**
for $p = 1 \rightarrow k$ **do**
$A_p \leftarrow F^T F$
$\hat{\mathrm{v}}_p \leftarrow$ random $N \times 1$ non-zero values
$\hat{\mathrm{v}}_p \leftarrow \hat{\mathrm{v}}_p / \|\hat{\mathrm{v}}_p\|$
for $z = 1 \rightarrow c$ **do**
$\mathrm{v}_p \leftarrow A_p \hat{\mathrm{v}}_p$
$\lambda_p \leftarrow \|\mathrm{v}_p\|$
$\sigma \leftarrow \|\mathrm{v}_p - \hat{\mathrm{v}}_p\|$
if $\sigma < \varepsilon$ or $\sigma > \hat{\sigma}$ **then**
break
end if
$\hat{\sigma} \leftarrow \sigma$
$\hat{\mathrm{v}}_p \leftarrow \mathrm{v}_p$
end for
$\mathrm{u}_p \leftarrow F \mathrm{v}_p / \lambda_p$
$max_u \leftarrow max(abs(\mathrm{u}_p))$ and $max_v \leftarrow max(abs(\mathrm{v}_p))$
$\kappa \leftarrow \sqrt{max_u \lambda_p / max_v}$
$\mathrm{u}_p \leftarrow \lambda_p \mathrm{u}_p / \kappa$
$\mathrm{v}_p \leftarrow \kappa \mathrm{v}_p$
$F \leftarrow F - \mathrm{u}_p \mathrm{v}_p^T$
$U(:,p) \leftarrow \mathrm{u}_p$ *and* $V(:,p) \leftarrow \mathrm{v}_p$
end for
else $\{N < M\}$
for $p = 1 \rightarrow k$ **do**
$A_p \leftarrow FF^T$
$\hat{\mathrm{u}}_p \leftarrow$ random $M \times 1$ non-zero values
$\hat{\mathrm{u}}_p \leftarrow \hat{\mathrm{u}}_p / \|\hat{\mathrm{u}}_p\|$
for $z = 1 \rightarrow c$ **do**
$\mathrm{u}_p \leftarrow A_p \hat{\mathrm{u}}_p$
$\lambda_p \leftarrow \|\mathrm{u}_p\|$
$\sigma \leftarrow \|\mathrm{u}_p - \hat{\mathrm{u}}_p\|$
if $\sigma < \varepsilon$ or $\sigma > \hat{\sigma}$ **then**
break
end if
$\hat{\sigma} \leftarrow \sigma$
$\hat{\mathrm{u}}_p \leftarrow \mathrm{u}_p$
end for
$\mathrm{v}_p \leftarrow \mathrm{u}_p^T F / \lambda$
$max_u \leftarrow max(abs(\mathrm{u}_p))$ and $max_v \leftarrow max(abs(\mathrm{v}_p))$
$\kappa \leftarrow \sqrt{max_u \lambda_p / max_v}$
$\mathrm{v}_p \leftarrow \lambda_p \mathrm{v}_p / \kappa$
$\mathrm{u}_p \leftarrow \kappa \mathrm{u}_p$
$F \leftarrow F - \mathrm{u}_p \mathrm{v}_p$
$U(:,p) \leftarrow \mathrm{u}_p$ and $V(:,p) \leftarrow \mathrm{v}_p$
end for
end if

Accounting for Uncertainty in Medical Data: A CUDA Implementation of Normalized Convolution

S. Lindholm and J. Kronander

Department of Science and Technology, Linköping University

Abstract

The domain of medical imaging is naturally moving towards methods that can represent, and account for, local uncertainties in the image data. Even so, fast and efficient solutions that take uncertainty into account are not readily available even for common problems such as gradient estimation. In this work we present a CUDA implementation of Normalized Convolution, an uncertainty-aware image processing technique, well established in the signal processing domain. Our results show that up to 100X speedups are possible, which enables full resolution CT images to be processed at interactive processing speeds, fulfilling demands of both efficiency and interactivity that exist in the medical domain.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation

1. Introduction

An uncertainty-aware visualization pipeline has previously been acknowledged as a necessary development to provide reliable visualization tools in critical application areas such as medical diagnosis [Joh04, KS08]. Nevertheless, uncertainty-aware visualizations are still the exception rather than the norm. In visualization, uncertainty arise and propagate in each step of the pipeline, having significant impact on the final image [LLPY07]. A particularly pertinent sub-domain is low-dose medical imaging. Serious concerns about the dose levels in current clinical practice has recently been raised [BA10] and methods to retain diagnostic image quality at increasingly lower dose levels are highly requested. The work in this paper comprises one step towards the development of a fully uncertainty-aware pipeline.

Image processing frameworks for dealing with uncertainty have previously been presented in the Signal Processing community [Kay01], and the idea of separating values of the signal from the certainty of the measurements has a long history [Gra78, GK83, Knu89]. However, for achieving widespread medical use, algorithms must not only be robust and accurate but also fast. User evaluations [LP11] have shown that efficiency is one of the most challenging aspects of the medical work flow. In this work we focus on the Normalized Convolution (NC) [KW93a] framework for which we present a CUDA based implementation. Previous approaches using NC have been limited in practice by their exceedingly slow runtime speeds, an effect of the local adaptivity of the filters. In contrast, our parallelized CUDA implementation yields speedups in the order of 100X. To highlight the usefulness of our approach, we have chosen to exemplify the impact of uncertainty in medical image processing by focusing on estimating gradients from noisy data. Local image gradients are used in a multiple of common mid-level processing tasks such as edge-detection [Can86], shading [HKRs*06] and transfer function design [KKH02], and are therefore of high importance in any visualization pipeline. The main contributions of this work are

1. Based on the existing theory of Normalized Convolution, we present a framework for fast processing of uncertain medical image data, including discussions on hardware optimizations and numerical considerations.
2. Using a highly optimized CUDA implementation of Normalized Averaging and Normalized Convolution we achieve a 100X speedup compared to previous work.

It should be noted that although this paper focuses on

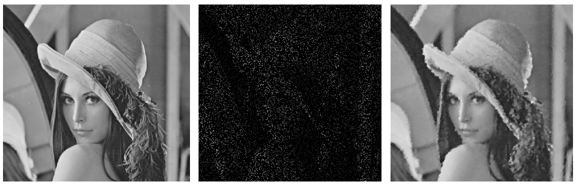


Figure 1: *Demonstration of the power of normalized averaging for restoring an image from only 10% of the original pixel values.* Left: *The original image.* Middle: *Image with 90% of the pixels removed.* Right: *The result of applying normalized averaging to the lossy image (a special case of Normalized Convolution with a single constant basis function).*

gradient estimation to highlight the effects of uncertainty, the method is independent with respect to the chosen basis/estiamtion desired. Other common convolution processes, such as curvature estimations, bilateral or anisotropic filtering would be affected in similar manner.

2. A Motivational Example

Consider a constant image [1 1 1] on which we apply a box filter [1 1 1]. The expected response is $(1+1+1)/3 = 1$ where the division by 3 is the natural filter normalization. Now consider the case where we have a dropped sample such that the same filter is now applied to the image [* 1 1] with a certainty of [0 1 1]. There are two naive ways to do this. Option one is to discard the uncertainty information (effectively interpreting * as 0) and filter the image as $(0+1+1)/3 = 0.67$. Option two is to discard the uncertain sample and calculate a new filter normalization $(1+1)/2 = 1$. This is called normalized averaging, and works well for the smoothing filters as demonstrated in Figure 1 where a decent image reconstruction has been achieved even after 90% of the pixels were dropped.

Now, lets see what happens in the case of a gradient filter, such as the central difference operator [-1 0 1]. Dropping a sample as in the example above leads to an effective adjustment of the filter from [-1 0 1] to [* 0 1]. It is easy to see that any constant offset in the image will now affect the filter response, which is not what was intended. It is important to note here that calculating a new filter normalization does not fix the problem. This is evident in Figure 2 where only 30% of the pixels have been removed but the gradient estimation is very poor.

In a more formal way, the operator [-1 0 1] can be interpreted as a basis vector used to analyze the signal. It is clear that getting a correct response is dependent upon its orthogonality to the constant vector [1 1 1]. Under the influence of uncertainty, this orthogonality cannot be guaranteed. As we shall see in the next section, normalized convolution provides a general framework to handle this problem. See Figure 3.

Figure 2: *Demonstration of the limitations when adapting normalized averaging for gradient estimation of lossy images (30%).* Left: *Original image.* Middle: *Lossy image.* Right: *Gradient estimation using filter weight normalization. It is evident that normalization of filter weights, which works well for smoothing filters, fails to produce sufficient results for gradient estimation of the lossy image.*

Figure 3: *Demonstration of the power of normalized convolution in gradient estimation of lossy images (30%).* Left: *Original image.* Middle: *Lossy image.* Right: *Gradient estimation using normalized convolution. Normalized convolution ensures the best possible estimations based on the given set of uncertainties for each pixel neighborhood in the image.*

3. Normalized Convolution Framework

The concept of *Normalized Convolution* (NC) was first introduced by Knutsson and Westin [KW93b,KWW93] to enable analysis of signals with locally varying sample certainty. Uncertainty in signal values can for example stem from known sensor dropouts (such as overexposed pixels in a camera), transmission errors, filtering of signal borders etc. Generally, the certainty of a signal element is modeled in the range [0..1], where 1 corresponds to a fully known element and 0 to a missing sample. In this section we will introduce the basic notions of normalized convolution as a local subspace approximation of a signal using a weighted metric, where weights are proportional to the signal certainty.

Normalized convolution have been used and extended from the original derivation in a number of works, notable is the thesis by Farnebäck [Far02], providing a comprehensive study on the use of NC for local polynomial expansions. The thesis also provides connections between NC and weighted least squares. Mühlich and Mester [MM04] also showed that NC can be interpreted in a statistical signal processing framework. More recently, connections has been found between NC and several modern filtering paradigms [Mil], such as non-local means [BCM05], the bilateral filter [TM98] and moving least squares [LS81]. Specifically, the difference between the methods can be shown to dependent only on the metric chosen in each local neighborhood.

3.1. Local Subspace Approximation Through Convolutions

In this section the basic signal processing framework used in NC is presented. Due to the limited scope of this paper, the reader is assumed to be familiar with basic concepts such as biorthogonal systems, dual coordinates, the metric tensor and inner product spaces (l^2). More details can be found in [Kre89, GK95].

Given a discrete signal $s(k)$ and a finite filter $f(l)$ with N taps, we can use convolution to compute a filter response,

$$h = s \star f \tag{1}$$

which can also be interpreted as an inner product,

$$h(k) = \langle s(k+l), f^*(-l) \rangle \tag{2}$$

This allows us to interpret convolution as a projection. Convolving $s(k)$ with a series of filters $f_m(l)$, where $m = 1...M$, gives M filter responses $h_m(k)$. These filter responses can be interpreted as the dual coordinates $\tilde{c}$ of the signal in the local *filter basis*. For M filters of length N we get a basis matrix B of size $N \times M$. This gives the local metric (or metric tensor) as

$$G = B^* G_0 B \tag{3}$$

where G_0 is the metric for the orthonormal cartesian coordinate frame. In the trivial case, this metric is defined by the identity matrix. From the dual dual coordinates and the metric we get

$$c = G^{-1} h_m(k) \tag{4}$$

with the coordinates c providing an expression for the local signal relative the basis given by the filters. For example, choosing the filters to be local polynomials $\{1, x, y, x^2, y^2\}$ we can approximate the Taylor expansion of a local signal region.

We can also interpret this subspace projection operation using a familiar least squares terminology. Consider a vector, $\mathbf{v} \in V$ and a subspace $U \subseteq V$ such that $\mathbf{v} = v_{||} + v_{\perp}$ with $v_{||} \in U$ and $v_{\perp} \notin U$. Least squares methods are then concerned with minimizing $|| \mathbf{v} - v_{||} ||$. If B is a base in U and c a set of coordinates describing $v_{||}$ in B, then this is the same as minimizing $|| \mathbf{v} - Bc ||= (\mathbf{v} - Bc)^* G_0 (\mathbf{v} - Bc)$. In classical linear algebra, the solution to this problem is given by the normal equations, stating that $c = (B^* G_0 B)^{-1} B^* G_0 \mathbf{v}$. Comparing this with Equation 4 we can identify the dual coordinates $\tilde{c}$ as $B^* G_0 \mathbf{v}$ and the metric G as $B^* G_0 B$.

So far, nothing apart from traditional image filtering has been introduced. All we have done is to interpret filtering at any given point in the image as a projection of a local patch of the image into a subspace spanned by a set of basis vectors.

3.2. Incorporating Filter Applicability

For our problem setting, we are interested in weighting the subspace projection (which can be seen as a least squares estimate) by a suitable *applicability function*, $a(l)$, describing the influence of neighbor pixels on the center pixel. To achieve this, instead of convolving the signal, $s(k)$, with the filters directly, we first multiply them with the applicability function to obtain a suitable *localization* of the filter basis. Thus we use filters defined as

$$g_m(l) = a(-l) f_m^*(-l) \tag{5}$$

where $a(l)$ is chosen to be a localizing function such that $a(l) > 0$, such as a Gaussian function.

Convolving $s(k)$ with $g_m(l)$ gives

$$\begin{aligned} h_m(k) &= \sum_l s(k+l) g_m(-l) && (6) \\ &= \sum_l s(k+l) a(l) f_m^*(l) && (7) \end{aligned}$$

which describes a generalized inner product between $s(k)$ and $f_m^*(l)$, where the weighting is provided by $a(l)$. This generalized inner product redefines the metric in the trivial basis as

$$G_0 = diag(a(l)) \tag{8}$$

which leads to an expression of the metric in B as

$$\begin{aligned} G &= B^* G_0 B && (9) \\ &= B^* diag(a(l)) B && (10) \end{aligned}$$

It is worth to note here that the matric G is still invariant as to where in the image the convolution is applied. In other words, the metric we use for our projection is always the same. With the introduction of uncertainty, this will no longer hold.

3.3. Incorporating Uncertainty

Given a signal $s(k)$ and accompanying signal certainty $r(k)$, we can allow the inner product to adapt to the local uncertainty of the signal. Incorporating the signal certainty in the convolution gives

$$h_m(k) = \sum_l s(k+l) r(k+l) a(l) f_m^*(l) \tag{11}$$

corresponding to a weighted inner product between $s(k)$ and $f_m^*(l)$, where the weights are set according to $r(k+l)a(l)$. This ensures that low certainty entries in the signal are given less weight when estimating neighborhood pixel regions. As

$r(k)$ varies with position, we thus get a position dependent metric $G_0(k)$.

$$G_0(k) = diag(a(l)r(k+l)) \tag{12}$$

In the filter basis, B, we can write the metric as

$$G(k) = B^* G_0(k) B \tag{13}$$
$$= B^* diag(a(l)r(k+l))B \tag{14}$$

and the local coordinates of the filter basis is given by inserting this expression into Equation (4)

$$c = (B^* diag(a(l)r(k+l))B)^{-1} h_m(k) \tag{15}$$

It is important to note that the metric is no longer spatially invariant, meaning we have to recompute G^{-1} at each point in the image. The spatial invariance also has the implication that the convolution will always be non-separable, independently of which filters that are used. Both of these things will impact the optimizations described in Section 4.

3.4. NC Optimization by Pre-computation

Computing the coefficients for every point using Equation (15) is a costly, as both a matrix inverse and several matrix products need to be evaluated. Explicitly expressing the computations with inner products and vector multiplications, denoting the columns of the base matrix B as b_m where $m = 1..M$, we can write Equation (15) as

$$c = \begin{pmatrix} \langle a \cdot r \cdot b_1, b_1 \rangle & \dots & \langle a \cdot r \cdot b_1, b_M \rangle \\ \vdots & \ddots & \vdots \\ \langle a \cdot r \cdot b_M, b_1 \rangle & \dots & \langle a \cdot r \cdot b_M, b_M \rangle \end{pmatrix}^{-1} \begin{pmatrix} \langle a \cdot r \cdot b_1, s \rangle \\ \vdots \\ \langle a \cdot r \cdot b_m, s \rangle \end{pmatrix} \tag{16}$$

Using the properties of the inner products, this can be rewritten as

$$c = \begin{pmatrix} \langle a \cdot b_1 \cdot b_1^*, r \rangle & \dots & \langle a \cdot b_1 \cdot b_M^*, r \rangle \\ \vdots & \ddots & \vdots \\ \langle a \cdot b_M \cdot b_1^*, r \rangle & \dots & \langle a \cdot b_M \cdot b_M^*, r \rangle \end{pmatrix}^{-1} \begin{pmatrix} \langle a \cdot b_1, r \cdot s \rangle \\ \vdots \\ \langle a \cdot b_m, r \cdot s \rangle \end{pmatrix} \tag{17}$$

Pre-computing the quantities $(a \cdot b_i \cdot b_j^*)$, $(a \cdot b_i)$ and $r \cdot s$ for all $i, j = 1...M$, significantly decreases the total number of multiplications necessary for each point/pixel in the signal. See [Far02] for more details.

4. Implementing Normalized Convolution in CUDA

This section will highlight and explain strengths and difficulties of performing normalized convolution on the Nvidia CUDA platform. In short, CUDA [Nvi09, Nvi10] is a parallel computing architecture developed to provide a C, or even C++, like interface to modern GPUs. Hierarchial layers of computational resources and memory provide opportunities to exploit parallel computation, coalesced access to global memory, utilization of fast shared memory, loop unrolling and latency hiding.

This section will begin by summarizing some of the key optimizations that can be used to speed up any non-separable convolution and will then proceed to difficulties related to normalized convolution. The impact of the optimizations on system performance is presented in Section 5.

A note on the word *kernel*: CUDA programs are called kernels, so to avoid the ambiguity between filter kernels and CUDA kernels we will henceforth simply refer to filters as filters while a kernel always refer to a CUDA program.

4.1. Optimizing Non-separable Convolution

Non-separable filtering is typically performed on patches of $4^2 - 16^2$ pixels on a version of the image that has been padded to the nearest multiple of 16. With one thread per pixel, this pattern allows fast coalesced reads over both image dimensions, copying one image patch to the shared memory of each available multiprocessor. Any additional boundary padding necessary for filtering (half the size of the filter) is typically copied as a secondary step or by additional threads. Highest performance is typically achieved when the patch size is as large as possible without overflowing the shared memory. Another important optimization used in this work is loop unrolling which, as explained later, is available 'for free' as the implementation already utilizes template functions. The next section will discuss optimizations specific for Normalized Convolution.

4.2. Optimizing Normalized Convolution

In Section 3.1, we saw that the main mathematical difference between standard and normalized convolution is that the subspace into which the signal is projected changes spatially in the image (due to the uncertainty). What this means in terms of computation is that we need to build, and invert, the metric matrix at each pixel.

There is plenty of documentation available for large matrix inversion using CUDA. There are, however, very few examples where matrices have be inverted on a per-thread basis (which is the case in normalized convolution). Applying standard, serial methods for matrix inversion is often problematic as these rely on recursion, a feature not present in the CUDA programming language.

The matrices that need to be inverted are $M \times M$ where M is the number of bases. These matrices are typically small, i.e. $M < 7$, but still require iterative solutions if $M > 3$. The iterative matrix inversion used in this work is based on the classic formula of scaling the adjoint matrix, Equation 18:

$$A^{-1} = \frac{1}{det(A)} * Adj(A) \tag{18}$$

Although this method is known to be computationally expensive and suffer from numerical inaccuracies for larger matrices it was deemed sufficient for the scope of this work. The computation of the determinant typically uses a recursive scheme with one dimensionality reduction per recursion. This is also true for the computation of the co-factor matrix as it relies on computation of determinants for a range of sub-matrices. This recursive process is performed through template instantiation using the following syntax.

```
template<int M> __device__ float Determinant(...)
    { ... Determinant<M-1>(...) ... }
template<> __device__ float Determinant<3>(...);
template<> __device__ float Determinant<2>(...);
template<> __device__ float Determinant<1>(...);
```

This effectively instantiates a unique function per recursion step, and thus circumvents the problem that no function may call itself. It is important to note, however, that the manual instantiation of $M = 1$ is necessary or an infinite number of functions will be instantiated, $[M, -\infty]$. The manual instantiations of $M = 2$ and $M = 3$ are provided for optimization purposes.

While function instantiation allows us to execute recursive processes, it also comes with a memory cost. The determinant function, for example, need to allocate a small array to hold a sub-matrix. If the 'recursive' call to the next determinant function is done within the scope of this allocation, the result is that subsequent allocations will be performed on each level of the recursive process. Now, remember that this is performed on a per-thread basis and that the amount of available registers and shared memory is small. The effect of this is that inversion of large matrices ($M \gtrsim 5$) will be significantly slowed as the shared memory is overflowed. Exactly when this happens depend on multiple factors, including filter and patch sizes.

4.3. Implementation

This section will provide a few additional notes on the CUDA implementation of normalized convolution. First, the allocation and utilization of the various memory types in CUDA will be detailed.

```
__shared__ float s_data[...];
__shared__ float r_data[...];
__constant__ float precomp_abb[...];
__constant__ float precomp_ab[...];
```

The two shared memory buffers, *s_data* and *r_data*, are allocated on a per-block basis and are initiated with the appropriate patches of the image and its uncertainty information during kernel launch. The two constant arrays are allocated on a per-grid basis and are initiated from the host code. *precomp_abb* and *precomp_ab* hold the applicability and precomputed tables described in Section 3.4.

As constant memory cannot be allocated dynamically, an upper boundary must be set for the number of bases and filter size. The simplest way to do so is to use pre-processor macros (i.e, *#define*). Similar to the template situation, a fairly straightforward way to achieve flexibility even when using 'fixed' values is to compile the same code multiple times with different macro values and then select the appropriate one during runtime. This is another example of a tradeoff between speed and flexibility in CUDA.

With the maximum number of bases, maximum kernel size and patch size denoted K_m, M_m and T respectively. Then, the two patches of shared memory requires $(T + 2 * K_m) * (T + 2 * K_m)$ floats each, and the two constant arrays ${M_m}^2 * {K_m}^2$ and $M_m * {K_m}^2$ floats respectively. In our implementation, patch size is set as a pre-processor macro but could potentially be passed as an argument as CUDA does support dynamic allocation of shared memory.

In the next section, the CUDA implementation is compared to a C++ implementation. The C++ implementation is to a large degree identical to the CUDA code, but use a series of *for*-loops to account for the lack of parallelism on the CPU.

5. Results

To evaluate how normalized convolution performs in CUDA, different implementation options were timed separately. These are presented here, followed by a general discussion in Section 6. All performance tests were performed at an image resolution of 512×512 pixels using a 2.67GHz CPU paired with a Nvidia GTX560Ti GPU.

First, the CUDA implementation is compared to a C++ implementation. Results can be found in Figure 4. As expected, the CUDA implementation is significantly faster over the entire parameter space. The CUDA implementation also displays little dependency on filter size thanks to the utilization of fast shared memory cache and registers. At the same time, however, a steeper increase in computation time is evident for cases with more than 4 bases. This steep performance increase is likely an effect of the increased memory consumption, forcing more samples to be taken outside of the GPUs fast local registers.

Second, the CUDA specific effects of patch size, memory layout and template usage were investigated. Figure 5 displays the performance impact when using template instantiations instead of relying on function parameters for configuring the number of bases and filter size. The template instantiation is nearly twice as fast as a result of loop unrolling and compiler optimizations. Figure 6 shows the benefit of utilizing shared memory in favor of the slower global memory. Naturally, utilizing shared memory is increasingly important the more memory that is read, such as for increasing filter sizes. Figure 7 shows the impact of computational speed when varying the patch size. As expected, larger patch sizes results in faster computation times. One reason for this is the increased occupancy of the various levels of computation on the GPU. Larger patch sizes also reduce the rela-

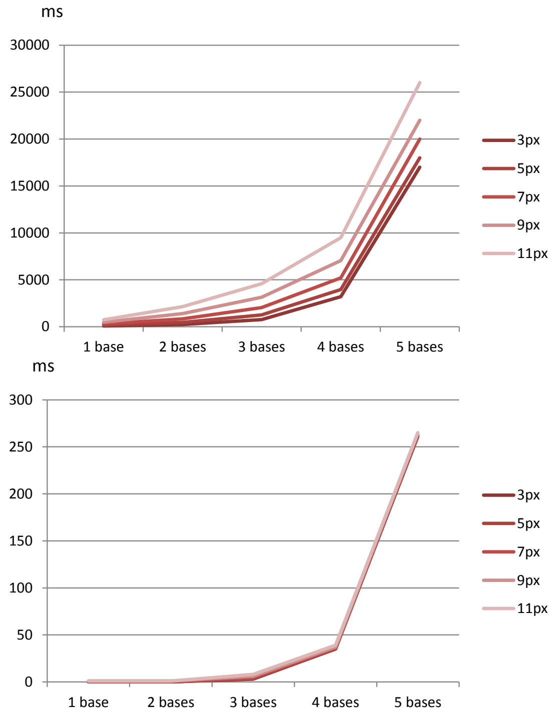


Figure 4: *C++ vs. CUDA.* Top: *C++ timings, in milliseconds, for varying number of bases and filter size.* Bottom: *CUDA timings, in milliseconds, for varying number of bases and filter size. The CUDA implementation is approximately 100 times faster but also displays more drastic increase in time consumption for higher number of bases, $M > 4$. It is also worth to note that the filter size has less impact in the CUDA implementation thanks to the utilization of shared memory.*

tive size of the filter skirt area (the pixels outside the patch but inside the filter radii). As a side note, this is a problem with non-separable filtering in general if the dimensionality of the images increases, as the relative area of the necessary boundary padding for each patch increases.

Figure 8 shows normalized gradient estimation on a real world data set where the *top row* contains (float left ro right); the reference image, lossy image and the uncertainty map. The *middle row* shows three types of gradient estimation on the reference image and the *bottom rows* shows the same gradient estimations on the lossy image. The three types of gradient estimation are (from left to right); no uncertainty compensation, re-weighted filters, and normalized convolution. The example is taken from a series of Computed Tomography images acquired at varying levels of dosage. the reference image was acquired at a dosage of 180mAs while the low-dose image was acquired at 12mAs. The uncertainty image was extracted by taking the absolute difference of the two images. It should be noted that the scans were taken

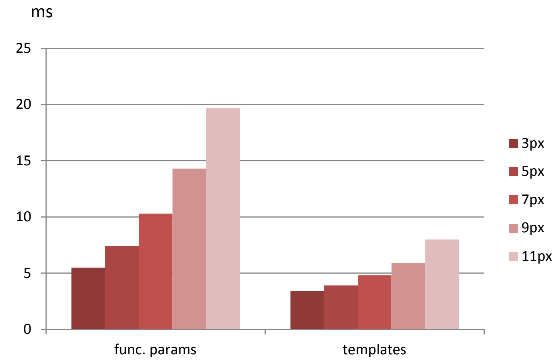


Figure 5: *Templates are used as they offer a decent trade-off between hardware optimization and program flexibility. Two variables are considered here; the number of bases and the filter size. Passing these variables as function arguments nearly doubles the computation time relative the case where the values are 'fixed' as template arguments. The downside to templates is that a large number of functions must be instantiated in order to cover all combinations of values.*

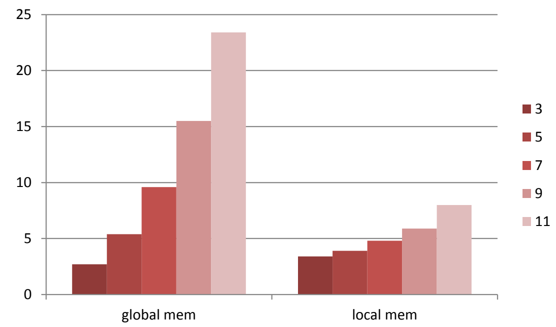


Figure 6: *Utilization of shared memory is essential in convolution as any given memory position is samples multiple times when applying a filter over a series of neighboring pixels. The larger the filter, the more important the shared memory becomes. The timings were performed using template functions with a three base setup.*

consecutively for the purpose of post-mortem imaging, so no registration was necessary to align the images. This is obviously a fabricated case, as we naturally would not have access to the high dose image when scanning live patients or would work directly on the high dose image in post mortem cases. It does, however, serve the purpose of highlighting how large of an effect uncertainty can have on gradient estimation in medical imaging.

6. Conclusions

In this work we have taken one step towards an uncertainty aware pipeline for medical imaging. Normalized con-

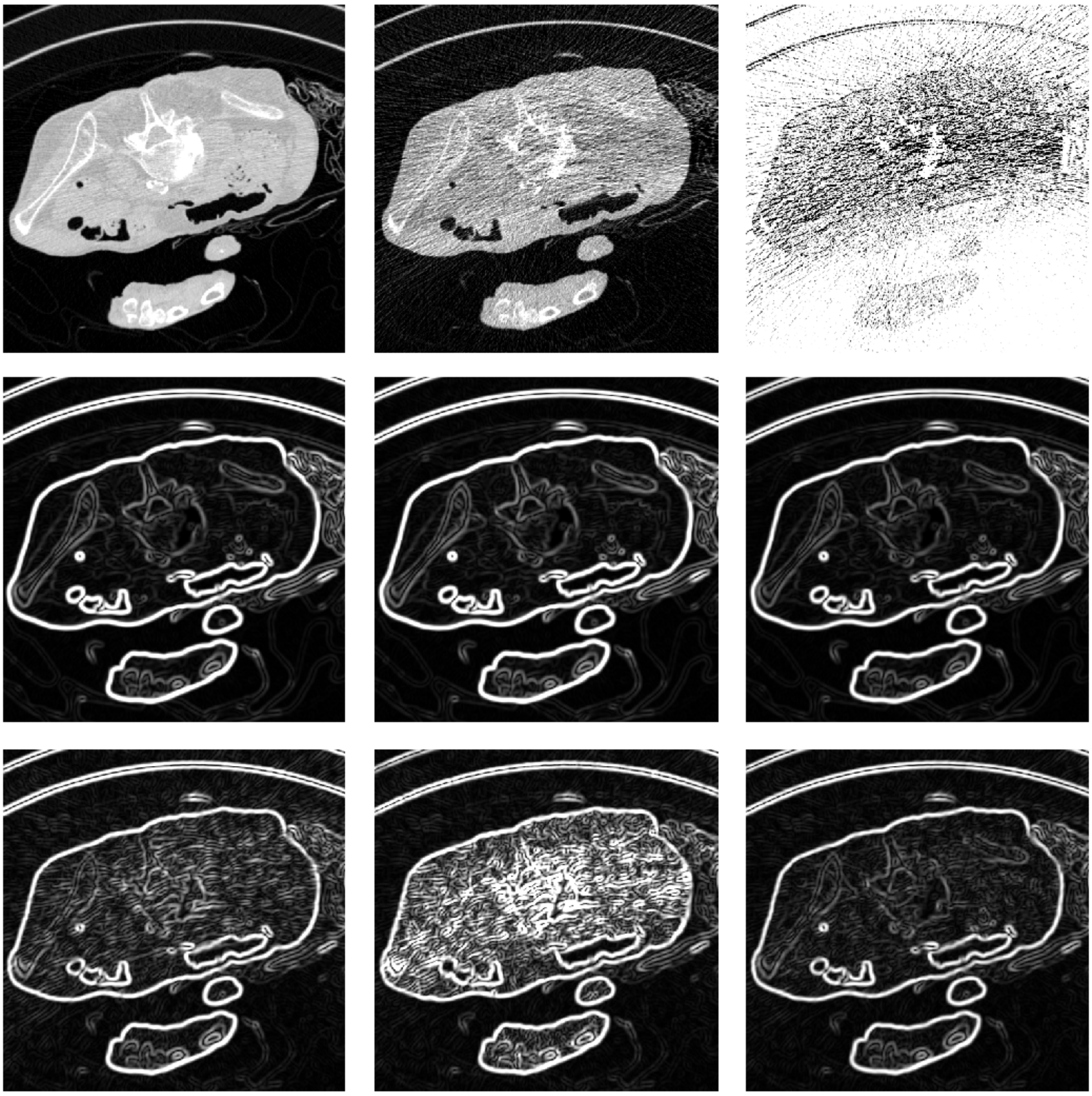


Figure 8: *Noise reduction in gradient computation on low-dose Compute Tomography (ldCT) images.* Top row: *High dose image (reference), low-dose image, difference image depicting noise in the* $[0,1]$ *range. The* middle *and* bottom *rows show gradient computation on the reference and low-dose image respectively.* Left: *Uncertainty has been discarded.* Center: *Using normalized averaging.* Right: *Using Normalized convolution. Normalized convolution clearly provides a better approximation of the gradients under the presence of uncertainty. The images were generated using a symmetric Gaussian applicability function with* $\sigma^2 = 2$ *and a filter size of* 11×11 *pixels. The CT data resolution is* 512×512 *pixels.*

volution, an established uncertainty aware image processing technique, has been implemented in the CUDA programming language to meet the high efficiency demands of the medical domain. We have demonstrated the importance of maintaining an uncertainty aware pipeline by showing the effect uncertain samples can have on gradient estimation.

However great potential CUDA has for speeding up computation of normalized convolution, it is not without limitations. As expected, computation time and storage requirements grow drastically with increased number of bases. This is, to a large degree, dependent on the chosen scheme to solve the linear system. Since the primary focus for this work was gradient estimation, it was deemed sufficient to use the scaled adjoint matrix and solve the system by using the inverse. For larger systems, however, more efficient and numerically robust approaches would be necessary. Memory consumption in particular will become a major concern as the number of bases grows and local/shared memory is over-

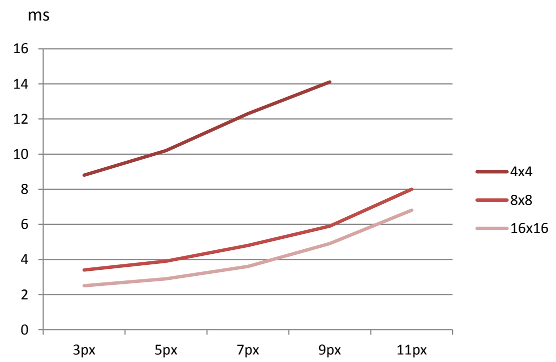


Figure 7: *The effect on CUDA performance when varying the patch size. While the patch size should be as large as possible it is limited by the memory consumption of the threads it triggers. The timings were performed using template functions with a three base setup.*

flowed. We did experience numerical errors in the matrix inversion process when dealing with larger matrices which further indicates that finding the inverse by the adjoint is only valid solution when dealing with low number of bases. Future work will naturally go towards alternative solutions to solve higher dimensional linear systems under the constraint of limited fast memory.

Acknowledgements

The authors would like to thank Hans Knutsson of the Department of Biomedical Engineering, Linköping University and Claes Lundström of the Department of Science and Technology, Linköping University.

References

[BA10] Brink J. A., Amis JR E. S.: Image Wisely: a campaign to increase awareness about adult radiation protection. *Radiology 257*, 3 (2010), 601–602. 1

[BCM05] Buades A., Coll B., Morel J. M.: A review of image denoising algorithms, with a new one. *Simul 4* (2005), 490–530. 2

[Can86] Canny J.: A computational approach to edge detection. *IEEE transactions on pattern analysis and machine intelligence* (1986). 1

[Far02] Farnebck G.: *Polynomial Expansion for Orientation and Motion Estimation*. PhD thesis, Linkping University, Sweden, SE-581 83 Linkoping, Sweden, 2002. Dissertation No 790, ISBN 91-7373-475-6. 2, 4

[GK83] Granlund G. H., Knutsson H.: Contrast of Structured and Homogenous Representations. In *Physical and Biological Processing of Images*, Braddick O. J., Sleigh A. C., (Eds.). Springer Verlag, Berlin, 1983, pp. 282–303. 1

[GK95] Granlund G. H., Knutsson H.: *Signal Processing for Computer Vision*. Kluwer Academic Publishers, 1995. ISBN 0-7923-9530-1. 3

[Gra78] Granlund G. H.: In search of a general picture processing operator. *Computer Graphics and Image Processing 8*, 2 (1978), 155–173. 1

[HKRs*06] Hadwiger M., Kniss J. M., Rezk-salama C., Weiskopf D., Engel K.: *Real-time Volume Graphics*. A. K. Peters, Ltd., Natick, MA, USA, 2006. 1

[Joh04] Johnson C.: Top Scientific Visualization Research Problems. *IEEE Computer Graphics and Applications 24*, 4 (2004), 13–17. 1

[Kay01] Kay S. M.: *Fundamentals o Statistical Signal Processing, Volume 1*. Prentice Hall, 2001, 2001. 1

[KKH02] Kniss J., Kindlmann G., Hansen C.: Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics 8* (July 2002), 270–285. 1

[Knu89] Knutsson H.: Representing local structure using tensors. In *The 6th Scandinavian Conference on Image Analysis* (Oulu, Finland, June 1989), -, pp. 244–251. Report LiTH-ISY-I-1019, Computer Vision Laboratory, Linköping University, Sweden, 1989. 1

[Kre89] Kreysig E.: *Introductory Functional Analysis with Applications*. Wiley, 1989. 3

[KS08] Kirby R. M., Silva C. T.: The need for verifiable visualization. *IEEE Comput. Graph. Appl. 28* (September 2008), 78–83. 1

[KW93a] Knutsson H., Westin C. F.: Normalized and differential convolution. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, x (1993), 515–523. 1

[KW93b] Knutsson H., Westin C.-F.: Normalized and differential convolution. In *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR '93., 1993 IEEE Computer Society Conference on* (jun. 1993), IEEE, pp. 515 –523. 2

[KWW93] Knutsson H., Westin C.-F., Westelius C.-J.: Filtering of Uncertain Irregularly Sampled Multidimensional Data. In *Twenty-seventh Asilomar Conf. on Signals, Systems & Computers* (Pacific Grove, California, USA, November 1993), IEEE, pp. 1301–1309. 2

[LLPY07] Lundström C., Ljung P., Persson A., Ynnerman A.: Uncertainty visualization in medical volume rendering using probabilistic animation. *IEEE transactions on visualization and computer graphics 13*, 6 (2007), 1648–1655. 1

[LP11] Lundström C., Persson A.: Characterizing visual analytics in diagnostic imaging. *International Workshop on Visual Analytics* (2011). 1

[LS81] Lancaster P., Salkauskas K.: Surfaces Generated by Moving Least Squares Methods. *Mathematics of Computation 37*, 155 (1981), 141–158. 3

[Mil] Milanfar P.: A tour of modern image filtering. To appear in IEEE Signal Processing Magazine. 2

[MM04] Mühlich M., Mester R.: A statistical extension of normalized convolution and its usage for image interpolation. *Proceedings of European Confereance on Signal Processing, (EuraSip)* (2004). 2

[Nvi09] Nvidia Corporation: NVIDIA CUDA C Programming Best Practices Guide CUDA Toolkit 2.3. 4

[Nvi10] Nvidia Corporation: Nvidia CUDA Programming Guide Version 2.3. 4

[TM98] Tomasi C., Manduchi R.: Bilateral filtering for gray and color images. *Computer Vision, IEEE International Conference on 0* (1998), 839. 3

Interactive Model Prototyping in Visualization Space

O. Daae Lampe[1,2] and H. Hauser[2]

[1]Christian Michelsen Research, Norway
[2]Department of Informatics, University of Bergen, Norway

Abstract

Researching formal models that explain selected natural phenomena of interest is a central aspect of most scientific work. A tested and confirmed model can be the key to classification, knowledge crystallization, and prediction. With this paper we propose a new approach to rapidly draft, fit and quantify model prototypes in visualization space. We also show that these models can provide important insights and accurate metrics about the original data. Using our technique, which is similar to the statistical concept of de-trending, data that behaves according to the model is de-emphasized, leaving only outliers and potential model flaws for further inspection. Moreover, we provide several techniques to assist the user in the process of prototyping such models. We demonstrate the usability of this approach in the context of the analysis of streaming process data from the Norwegian oil and gas industry, and on weather data, investigating the distribution of temperatures over the course of a year.

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: —Interaction techniques G.3 [Probability and Statistics]: —Time series analysis

1. Introduction

Modeling is an essential part of scientific work. To be able to learn from observations and to utilize the gained knowledge for subsequent analysis, such as prediction, the modeling of the observed phenomenon in some sort of a prototype is crucial. Also, central to science is that model hypotheses are tested, refined and validated, or possibly rejected after testing. In the following, we consider a *model* to be a physical, mathematical, or logical representation of a system entity, a natural phenomenon or process, and that *modeling* is the act of creating a model [Sil01]. In experiments or, as we will focus on, modern process logging, measurements and data is gathered. Establishing a model on measured data often starts by employing empirical / statistical tests with trial and error, finally ending up with a *model prototype* and the statistical confidence on the model's accuracy. When and if the model prototypes hold up to scrutiny, one can aim to generalize these, and create a *model template*. The model template can be thought of as a scale invariant model, something that would fit to data irrespective to influencing factors, and then used to quantify these factors. E.g., a model template of time over height squared would, if applied to Galileo's raw experimental data, establish the gravity constant, along with the statistical confidence of this value.

Considering the visualization of particle paths in a tokamak (fusion reactor) as another example, we first consider that the most obvious footprint of direct data visualization is the fact that the particles intensely rotate – an observed phenomenon that is principally important, but not really surprising. To see this in a visualization might be interesting, but shortly after confirming the expected rotation of particles, we want to proceed and look behind this phenomenom: is there any secondary motion characteristic to be seen? To actually check such a hypothesis, we can aim at abstracting already understood and accepted aspects of the investigated phenomenon from the data visualization. This abstraction leads to three results: (a) the finding itself, which will undergo an externalization, where the finding is pulled out of the visualization represented in a different form, and (b) a residual data visualization – where the finding has been subtracted from – which then allows for studying remaining aspects of the observed phenomenon that do not follow the model. In the case of the tokamak example, we can think of an abstracted visualization of the particles, e.g., by using a Poincaré map (the main feature, i.e., the rotation of the particles, is then no longer visible, but only off-rotation deviations of the particle paths). This clears the view and allows the user to gain a more thorough understanding of complex phenomena through iterated analysis, including modeling and abstraction. (c) since large scale movements or densities of data are subtracted after the abstraction, this enables the further study of features which might be a magnitude smaller than the overshadowing and perhaps obvious features.

With this paper we aim to introduce a novel iterative workflow of assisted modelling, abstraction and subtraction to completely map a dataset from the originally visualized view to an abstracted and quantified one. To achieve this goal, we first provide a novel technique to assist a user to sketch locally optimal models, and then how to represent these in an abstract and quantified manner. Our technique is mainly applicable to data that can be mapped to a 2D representation.

The remainder of this paper is organized as follows: next we discuss some related work. Then we elaborate on the theoretical part of this work in Sec. 3, before we go into detail with respect to our technique in Sec. 4. In Sec. 5 we present results from the application of our approach and demonstrate its usefulness in this context.

2. Related Work

Extracting well defined features is a related topic often studied in the field of flow visualization. Post et al. [PVH*03] provides a good overview of the current state of art and how the features are found, abstracted, and quantified. On other phenomenons where the basic models is understood, data for visualization can often be reduced or abstracted. Löffelmann et al. [LKG98] use Poincaré maps as such a technique to create abstractions of data, reducing the dimensionality, and thus allowing the visualization of secondary features. On data in which the model is not understood, Rheingans and desJardins showed that inductive learning techniques, such as self organizing maps (SOM), can construct explanatory models for large, high-dimensional data sets [dR99, Rd00]. Their technique employs an overlay of models and data visualization, and thus creates an implicit visual comparison of model vs. data. Models as such are used in all sorts of scientific work – it is therefore perhaps beyond the scope of this work to reasonably discuss the role of models in science and visualization. Examples reach as far as into model-based segmentation of medical data (for example research for cardiac diagnosis [ZSBH08]) or into model-based object recognition (such as for robot vision [CD86], for example). These approaches show how models are used for classification and segmentation. Often, they also utilize a difference view, in which they show the match of a model to the data (which here relates to our residual data visualization). By iteratively adopting our technique, we can find higher order features, which are related to the field of multi-resolution analysis and multi-scale modeling, such as wavelet-based approaches [Wal04], for example. However, instead of focusing on a decomposition in frequency domain, we capitalize on partial abstraction which is feature-based and local.

The obvious step after extracting features is to put them to good use. Liu and Stasko [LS10] investigate how internal representations (mental models) and external visualizations relate to each other. The authors state that such mental models are used during visual reasoning to "simulate" the behavior of the corresponding visualization system [LS10]. Our approach helps the analyst to externalize such mental models, and compare the data to it. Shrinivasan and van Wijk and Yang et al. have investigated how to effectively support an *externalization* of findings in visualization. Yang et al. [YXRW07] describe a system which allows users to externalize findings, or *nuggets*, while exploring a dataset. These nuggets are then added to a *Nugget Management System*, where clustering and meta-information, help the sense making process. They also describe how visualization in this nugget space prove useful as an abstraction of the original data. Shrinivasan and van Wijk present the *Knowledge View* [SvW08] in which not only the findings are externalized, but also the interaction path that lead to it. Their user study shows favorable results with respect to externalizing knowledge in mind maps. Shrinivasan and van Wijk and Yang et al. have investigated how to effectively support an *externalization* of findings in visualization. Yang et al. [YXRW07] describe a system which allows users to externalize findings, or *nuggets*, while exploring a dataset. These nuggets are then added to a *Nugget Management System*, where clustering and meta-information, help the sense making process. They also describe how visualization in this nugget space prove useful as an abstraction of the original data. Shrinivasan and van Wijk present the *Knowledge View* [SvW08] in which not only the findings are externalized, but also the interaction path that lead to it. Their user study shows favorable results with respect to externalizing knowledge in mind maps.

The visualization scheme utilized in this work, is highly dependent on a frequency based technique that also supports meaningful difference views. Daae Lampe and Hauser presented a technique [DH11b] that enables the continuous distribution of data, using kernel density estimates (KDE). This technique also extends to support the continuous distribution of data-samples that is temporally connected, in that it creates a line-kernel that connects these samples. Daae Lampe et al. also effectively utilized these 2D KDE techniques for difference views [DKH10] in an application that aimed to generalize how to create multiple views that highlight the differences in distributions between distinct categories.

3. The Basic Idea

One of the goals of this research is to effectively support practitioners and scientists to analyze process data that is streaming in or updated at considerable rates. We provide an approach that allows users to rapidly prototype models

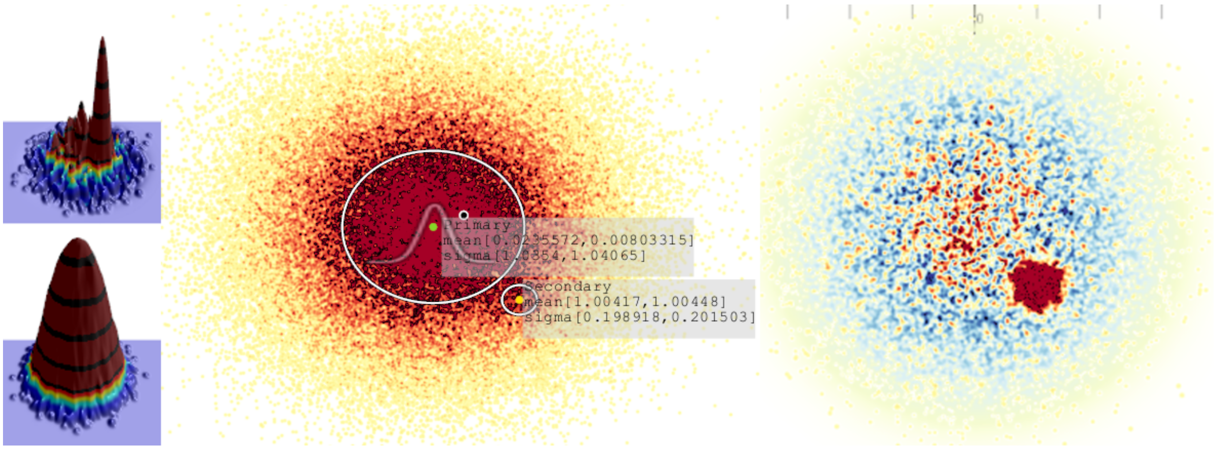


Figure 2: *A 2D version of the example in Fig. 3. The left images shows the (logarithmic) height-map after and before subtraction. The middle image shows the quantified measures as read out from both the primary and the secondary feature (after fitting two model prototypes). The image on the right shows the data, after abstracting the primary feature, clearly revealing the secondary feature, even though it was almost completely hidden.*

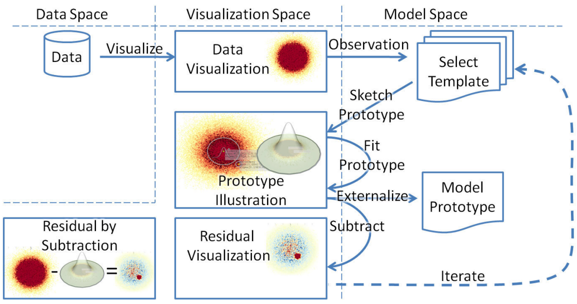


Figure 1: *Our proposed workflow: visualize and observe, sketch and fit, externalize and subtract, then iterate.*

for structures which the user identifies in a visualization of the data. These model prototypes act (1) as parts of the externalization of the user findings and (2) as means to quantify the structures for subsequent user tasks. Accordingly, we first focus on how to identify structures which lend themselves to model prototyping. We see two opportunities: either the user has a conceptual model of what to look for (analytic/confirmative setting), or she/he aims at creating one by looking at the data (explorative setting). In the first case, it is useful to integrate the anticipated model within the visualization, to get initial information on how well the data fits the model. In the second case, it is useful to have a visualization that supports the user in interactively prototyping the model to then subtract it from the visualization, and get immediate feedback on how well it fits.

From this description we extract our workflow: *visualize and observe, sketch and fit, externalize and subtract, then iterate*, as shown in Fig. 1. This figure is read from top left then right or down. The data is visualized, and by observation an interesting feature is detected. The user selects a suitable model template and by sketching onto the visualization, creates an initial model prototype. Through further sketching and automatic fitting, the prototype is finalized. This complete prototype is externalized to model space, and a residual visualization is created by subtracting the model prototype from the data visualization. At this point the procedure can be repeated by observing another feature in the residual visualization, selecting another template to prototype, and so on. To explain by example, we consider a test dataset consisting of serveral random samples adhering to two different normal distributions. One of these distributions are of a magnitude smaller than the other one, and thus occluded. This dataset is shown in Fig. 2 and a simplified 1D version of the same is shown in Fig. 3.

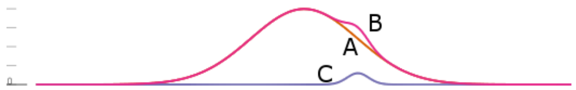


Figure 3: $A(x) = N(0,1)$, $C(x) = 0.05N(1,0.2)$ *and* $B(x) = A(x) + C(x)$

Visualize and observe: we first draw all the data and the middle figure of Fig. 2 is shown. In this figure we observe only a single normal distribution, and thus decide that this would be a suitable candidate model.

Sketch and fit: in this second step, we pick the selected candidate model, the normal distribution, and mark its center near the observed center directly in the visualization. Immediatly when the model is placed, an automatic fitting will initiate and the sketched model will be optimized to fit the data.

Externalize and subtract: the parameters from the, now optimized, model are extracted and displayed. In our example case, the normal distribution model yielded a mean $\mu = (0.02, 0.01)$ and variance $\sigma = (1.035, 1.04)$, which is close to the reference $N([0,0],[1,1])$. These parameters can then be externalized, e.g., to a simplified table of extracted results. After this, the density of this externalized model is subtracted from the view, yielding the visualization to the right in Fig. 2.

Iterate: Finally, now presented with the view where the main feature is subtracted, we can start over again by observing the second feature as another normal distribution. After sketching and fitting this second normal distribution, we can extract the parameters $N([1.004, 1.004], [0.1989, 0.2015])$, here compared to the reference $N([1,1],[0.2,0.2])$.

4. Interactive Model Prototyping

In this section, we explore the details related to the proposed workflow, which are, visualization, model sketching and fitting, externalization or quantification, and interactive convergence.

4.1. Visualization

To support the proposed workflow we separate our visual aspects into three parts, the data visualization, the model prototype illustration, and the residual visualization. The model prototype illustration serves the purpose of giving a non-occluding and condensed view of where all the previous and the current model prototypes are located. Additionally, the prototype illustrations act as handles for interaction. The residual visualization serves several purposes. The first purpose is to show how well the model fits the data, and the second one is to then utilize the *visual range* better (through a scale-up operation). Human perception, and thus visualization, has a limited tolerance for range, e.g., there is finite limit to how many colors we can distinguish, or a limited range in how we can perceive brightness. By subtracting low frequency, high amplitude, features, we can effectively and automatically create a new and optimized range.

Streaming process data requires a direct in situ visualization of the data, since it is constantly updated. The visualization technique utilized here, is based on work by Daae Lampe and Hauser. [DH11b], and 2D Kernel Density Estimates (KDE). This technique visualizes a large set of samples, but displaying the convolved sum of kernels, one per sample, resulting in an analytical density function, that supports meaningfull difference views [DKH10]. Additionally, the usage of scaled kernels will create a visualization that shows the distribution of time, independent of sampling rate.

4.2. Model Sketching and Fitting

In computer science terms, a model template would be a class, and a model prototype would be an instance of such a class. In the process of creating a prototype, *sketching* is considered the manual input, and *fitting* the automatic algorithm assisting the user. Model templates come with properties, that the prototype needs to instantiate, which we categorize below. We consider them to be *shape*, *distribution*, and *scale*.

Shape characterizes the form of the model *along* the sequence of samples (after visualization). A linear structure can be described by a line model, more complex forms would follow spline curves, for example. In our case, we are fine with a piecewise linear model template. However, more complex models are equally possible (as long as a fitting procedure, see below, is available, as well). We refer to this central characteristic of a model as the *shape construct*. Selecting a shape requires the selection of the following parameters, *shape construct* and *control points*. We will only consider the following subset of shapes for the remainder of this paper: the single point construct (see Fig. 2), and the piecewise linear construct, wich will fit data with some correlation.

Distribution determines the form of the model *across* the sequence of samples. Whether it is due to noise, weak measurements, or other natural phenomena, real world data rarely ever line up perfectly. We therefore consider a certain data distribution across the sequence of data samples, which we model accordingly. The definition of the distribution we will refer to as the *distribution construct*. Selecting a distribution requires the selection of the following parameters, *distribution construct* and *width*. In the following we will denote this width as $\mathbf{r}$, a vector separating the "radius" in the screen space coordinates u and v.

Scale, finally, is a measure of intensity. Depending on the visualization parameters, this parameter will have different meanings. E.g., for a box, the scale will be the average within it, for other, it will give a more complex measure of the intensity within the model.

Summing this up, we need to find a matching shape construct, a matching position, select a distribution construct, find the distribution width, and finally determine the scale, when we aim at fitting the model prototype to the data. To measure how well a model prototype fits the data, a problem not very different from image comparison, a correlation function like sum of squared differences has proven to be useful [GS99]. Other difference norms, such as the L1 norm, for example, also are possible and the choice of which norm to use is usually application-dependent. After choosing a squared differences norm (here L2), we investigate the opportunities to minimize it for fitting. To simplify the function to minimize we will consider the selection of shape construct and of distribution construct as selected manually by the user (according to his or her a priori assumptions about the data). In the following, we denote the discrete scalar field, which results from mapping the data to visualization space a $D(u,v)$, where u and v are the screen or canvas coordinates, and the models scalar field (also after mapping into visualization space) as $M(u,v)$. To generate this scalar field M, we first need to select the shape position $\mathbf{p}$, the distributions radius/extension $\mathbf{r}$, and the scale/height h. Based on these selections we have a function $L(\mathbf{p},\mathbf{r},h)$ which when mapped to the visualization space represents M

$$L(\mathbf{p},\mathbf{r},h) \rightarrow M(u,v) \tag{1}$$

Defining $\mathbb{UV}$ as the natural numbers from zero to canvas-width and canvas-height we get the difference measure (L2):

$$s = \sum_{(u,v)\in\mathbb{UV}} (D(u,v) - M(u,v))^2 \tag{2}$$

From eq. (1) and (2) we find that s, the sum of squared differences, is a function $s(\mathbf{p},\mathbf{r},h)$. From this we can extract our target variables through the following optimization:

$$\underset{\mathbf{p}\in\mathbb{UV},\ \mathbf{r}\in\mathbb{R}>0,\ h\in\mathbb{R}>0}{\operatorname{argmin}} s(\mathbf{p},\mathbf{r},h) \tag{3}$$

Optimizing this equation is not straightforward, but since the user inherently sketches close to the desired solution, we can avoid potential problems with locating the global optimum, and only aim for the local minima.

We experienced satisfying results with the traditional Newton's method for this optimization problem, due to its good convergence [NW99] in local problems. Using Newton's method requires a Hessian matrix, a function that is twice differentiable, and an initial estimate x_0 that is sufficiently close to the solution. Calculating the Hessian matrix (or inverse thereof) is an computationally expensive operation, and we separate the derivatives and to minimize the function s by individually minimizing the variables in order of their influence of the overall model field

$$\underset{\mathbf{p}\in\mathbb{UV}}{\operatorname{argmin}} s(\mathbf{p}), \quad \underset{\mathbf{r}\in\mathbb{R}>0}{\operatorname{argmin}} s(\mathbf{r}), \quad \underset{h\in\mathbb{R}>0}{\operatorname{argmin}} s(h) \tag{4}$$

We look into these optimizations in Sec. 4.4 with measured results of convergence.

4.3. Quantification and Model Prototyping

After completing a number of model prototypes in visualization space, we transfer quantitative information back from visualization space into *model space*, a technique called externalization. Model space can be thought of as a summary of the understood features found in the data, and thus a more holistic approach to modeling is possible; one that also takes model parameters into account, which haven't dealt with in visualization space. For example, when visualizing speed vs. height of an object in free fall, this only can lead to model prototypes correlating those two parameters, not (yet) considering other potentially influencing factors, such as aerodynamic drag, etc. As established in Sec. 4.2, the information available is the position $\mathbf{p}$, distribution extent/radius $\mathbf{r}$, and scale h. Transferring $\mathbf{p}$ to model-space is trivial, and so is also $\mathbf{r}$. If the selected distribution construct is *box* or *linear*, then the transformed $\mathbf{r}$ is directly the radius around our shape construct. When using other distribution constructs we must allow for other interpretations of $\mathbf{r}$, e.g., the normal distribution, where variance or σ is more informative.

We have now described how to extract positions of our abstraction, its distribution width and a scale, based on a given intensity. We will look into more detail on how to apply this information in synthetic and real life cases in the next two sections, but we can already now see the usefulness in cases as pure statistical measures, or quantitative readouts of mean and variance.

4.4. Interactive Convergence

When sketching model prototypes, it is inherently hard to accurately or optimally draw the model prototype, as this would require the user to locate a local minimum based on several parameters. As introduced in Sec. 4.2, this would require the user to set five parameters, $\mathbf{p}, \mathbf{r}$, and h, when she/he is modifying a single point model construct. In this work we suggest an assisted prototype fitting, in which the user gets feedback on whether the first suggestion will converge towards his/her desired solution, or not. In the interactive mode, the user moves one point, and when it is released at its new position, the fitting algorithm will initiate. The iterative fitting algorithm is configured such that it will slowly converge/diverge at its first steps. If the initial suggestion was not sufficiently close to the optimal position $\mathbf{x}*$, it will diverge, away from the users desired target position. Instead, we iterate the fitting only a few steps, with constant step size instead of using Newton's method, so that the user can click and redirect the point before it *runs* away. When the user sees that the point is converging towards the desired solution, she/he can initiate the fitting algorithm that will then converge with the speeds that Newton's method offers.

As discussed in Sec. 4.2, we established that we need to compare the least squares of the model vs. the data. To initiate the fitting, we first need a rasterized version of the data. This texture is created once per frame, or when the dataset is updated. Next, we need a rasterized version of the model, which we create using a *construct aware* rasterize function, specific for the different implemented models. To recall, we considered the data's rasterized texture as $D(u,v)$ and the models, $M(u,v)$. Next, we need to calculate the least squares, and then we need to sum all the calculations for u and v. These calculations, as specified by Eq. (2), are implemented on a shader, which first performs the least squares, and secondly performs the reduction sum. We discussed the separation of the optimizing previously (see Eq.(4)), and as our tests have shown good convergence, when we iterate stepwise in our previously implied order (ie. one step with position, one with extent, and then with height, before reiterating

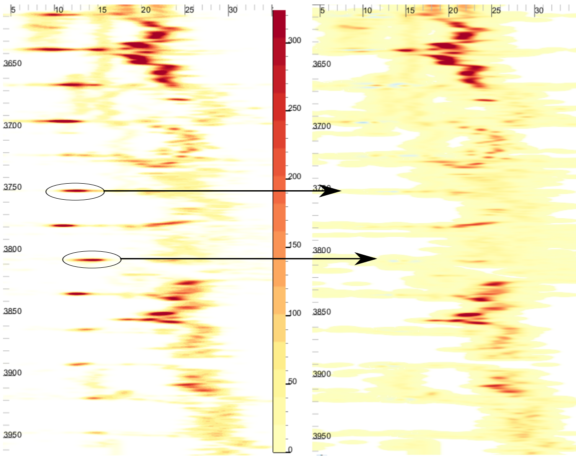


Figure 4: *Torque in kN.m over depth. The figure on the left shows the original data, containing some ROB tests we have identified, modeled and subtracted from the residual view to the right.*

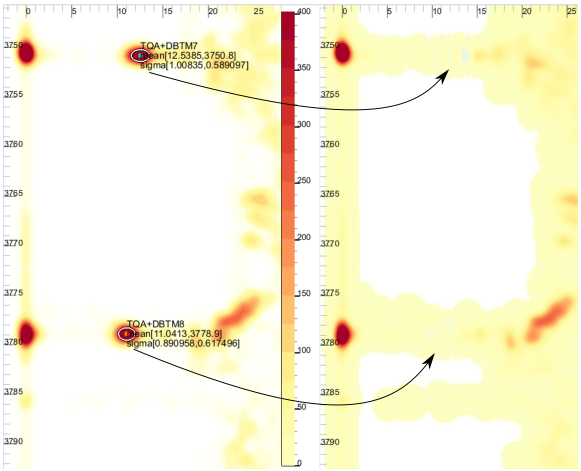


Figure 5: *A zoomed in view from Fig. 4 onto two rotation off bottom tests, showing how well this data is modeled and removed from the residual view. The residual view to the right has a larger area of yellow values due to a dynamic range color re-scale.*

next step). If we have a point construct, we repeat the process of calculating the sum of least squares, for our position $\mathbf{p}$ in the positions:

$$\mathbf{p}+\Delta u, \quad \mathbf{p}-\Delta u, \quad \mathbf{p}+\Delta v, \quad \mathbf{p}-\Delta v$$

Next, for Newton's method we need $f'(\mathbf{p}_n)$ and $f''(\mathbf{p}_n)$, now let (for both u and v):

$$f'(\mathbf{p}_n) = \big(s_n(\mathbf{p}_n+\Delta) - s_n(\mathbf{p}_n-\Delta)\big)/2|\Delta| \tag{5}$$

$$\begin{aligned} f''(\mathbf{p}_n) = \big(&(s_n(\mathbf{p}_n+\Delta) - s_n(\mathbf{p}_n)) \\ &-(s_n(\mathbf{p}_n) - s_n(\mathbf{p}_n-\Delta))\big)/|\Delta| \end{aligned} \tag{6}$$

And thus we are able, to calculate $\mathbf{p}_{n+1} = \mathbf{p}_n - \frac{f'(\mathbf{p}_n)}{f''(\mathbf{p}_n)}$, $n \geq 0$.

Next, we fit the extension of our distribution $\mathbf{r}_n$ (in the case of normal distribution, this would be σ), ending up with $\mathbf{r}_{n+1}$, finally, before calculating the scale, or height h_n of the distribution. This we implement in a similar manner, by calculating $s_n(h_n)$, $s_n(h_n+\Delta)$, and $s_n(h_n-\Delta)$, and creating the derivatives, using the same procedure as above, then by Newton's method, we calculate a new height h_{n+1}. To extend the above method, that was defined for point constructs, for piecewise lines, we repeat the first step, finding $\mathbf{p}_{n+1}$ for all $\mathbf{p_n}$ in the piecewise line segment.

5. Case Studies

We now present two case studies, one on process data from the Oil and Gas sector on real-time data generated under drilling operations, and one analyzing the temperature changes through several years.

5.1. Process Data

We will apply our approach to a dataset that contains 116,191 time-steps in total, spanning over a period of approx. 28 hours, with a varying sample rate from $1/30Hz$ to $30Hz$. It is a multivariate dataset containing 25 variables at each time-step in three major categories, *measured* data from the surface, measurement while drilling (*MWD*) equipment and *derived* data. MWD or down-hole measurements are measured from MWD tools down in the well and then transmitted to the surface via mud pulse.

A prominent usage of these data-streams are logging, early detection and warning in case of incidents or analysis to elaborate on a problem evolving or past. To give an early warning on potential incidents, a common strategy applied are friction tests. Increased friction can be a good indicator on amassed cuttings in the hole. To infer friction, two different techniques are applied: torque based tests, and weight based tests. When pulling the string up we can expect friction to act as a force against the movement, thus increasing the measured weight, and similarly when moving the string down we expect a lower weight. Rotating the drill string will give a response torque measured on the surface. This torque will increase with increasing friction. This test is called *rotation off bottom* or ROB.

Fig.4 shows abstracted and residual data visualization representing ROB. Notice how all the higher densities (representing time spent performing this ROB), are removed in the residual view to the right. Fig.5 shows a zoomed in version of Fig.4, where two of these tests and additionally their quantitative parameters are shown. From this model proto-

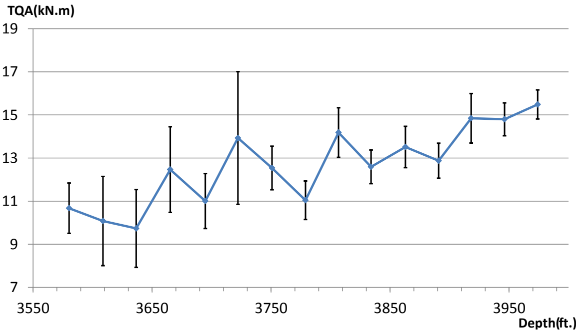


Figure 6: *Changing torque in kN.m over depth in feet, for a series of ROB tests. The abstracted results from Fig.4 are shown in a graph with error bars at 1σ for both depth and torque uncertainty. In this figure, the uncertainty for depth is negligable.*

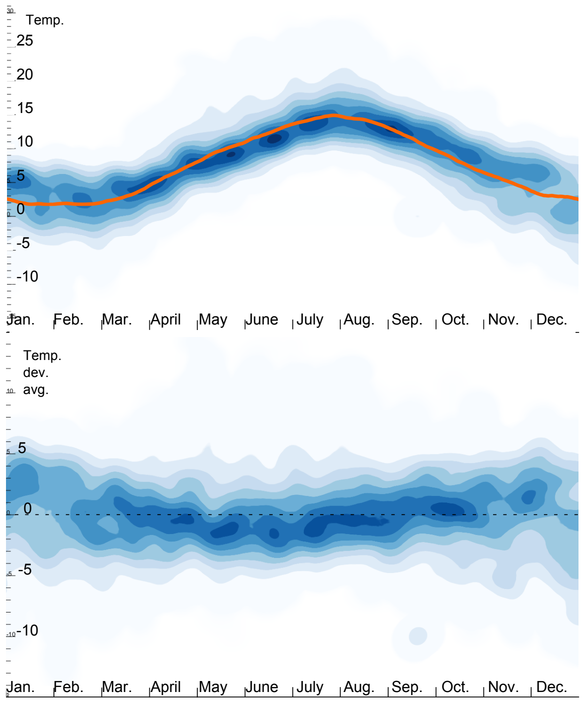


Figure 7: *On top, the distribution of measured temperature over the average year based on hourly data for ten years, by using the curve density estimate [DH11a]. The orange curve shows the cyclic moving average of the temperature over all ten years. On the bottom, the same data is shown, but here de-trended, by subtracting the moving average.*

type, one can now read out the average and mean during this ROB. A big advantage our technique has compared to the existing one, where the mean of all measured values during the ROB is taken, is that our technique would show a poor fit, if a poor ROB is performed. An example of a poor ROB is if the samples are increasing, or decreasing during the entire ROB period. Another poor ROB would have its samples clustered at two distinct torques, and a mean would then be misleadingly in middle between them. After matching a total of 15 of these ROB tests in Fig.4 we go to a more abstract view, where all the data is removed, and only the modeled statistcs are left. Fig.6 shows 15 ROB tests, with their standard deviations shown as error bars for both depth and torque. This abstraction illustrates quite well a problematic ROB friction that occurred at 3722 and at 3815 feet, but was put under control at larger depths with more moderate torque and a lower deviation (more certain measurements).

Quantifying torque is an important step in knowing how the down-hole conditions are developing, but that is only one step on the way of getting indications on what the friction is. There are no accurate algorithms that exist today, that can accurately calculate friction, even if all different conditions are taken care of. This is the reason why we look at torque, since it is an indication of friction even though many more variables affect the result.

5.2. Temperature

In this section we inspect hourly temperature readings from a single weather station, over the course of ten years, courtesy of eKlima [Nor]. The top of Fig. 7, displays the curve density estimate [DH11a] of these curves over an average year. In this display the most prominent feature is the seasonal change, with high temperatures during the summer, and lower temperatures during the winter. Fig. 7 also features an orange line overlaid to display the cyclic moving average of the yearly temperature for these ten years. Such a moving average is a good representation and abstraction for the yearly temperature, given that the data follows a normal distribution, but might provide false information if not [DH11a]. To investigate how well this average can abstract the data, we first create a new dataset containing the differences between the moving average and the measured temperature. This new dataset of deviations from the moving average, is shown as the bottom graph in Fig. 7. The higher peaks, at approximately zero, during the summer months in this dataset, indicate a more stable temperature, i.e., the average temperature is measured more often. To investigate how well the normal distribution fits the de-trended data, we apply a linear normal-distributed model to it. The resulting difference view between the applied model and the de-trended data is shown in figure 8. Now, as opposed to the previous two figures, the focus is placed on the deviations from the normal distribution. Our attention is drawn to the high intensity above the norm in January and December. Since these intensities are red, they present areas where the measured value is higher represented than the normal distribution. However, since the average is placed at zero, it indicates

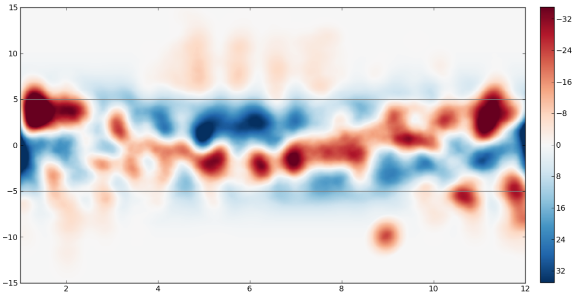


Figure 8: *The difference between a linear model and the detrended data from Fig. 7. The linear model applied has its mean μ on $y = 0$, a $\sigma = 1.85$ and its upper and lower quartile shown as grey lines. Note that due to the diffence view, the deviations shown here are of one magnitude greater than in Fig. 7.*

Name	Value	Reference	Pixel err.
Primary μ X	0.02	0	1.7
Primary μ Y	0.01	0	0.85
Primary σ X	1.035	1	2.98
Primary σ Y	1.04	1	3.4
Secondary μ X	1.004	1	0.34
Secondary μ Y	1.004	1	0.34
Secondary σ X	0.1989	0.2	0.0936
Secondary σ Y	0.2015	0.2	0.128

Table 1: *Error measurements in data space units and pixels on standard normal distribution with secondary feature, see Fig. 3 and Fig. 2. A rendering with 85 pixels per unit has been used for these calculations.*

a "tail" of low temperatures dragging the average down, i.e., a negatively skewed distribution. A second finding here is the anomaly placed mid September, where we find a peak of overrepresented values at an extreme ten degrees below the average. After closer inspection in the dataset, we found that this represents missing values which was defaulted to zero. As a third finding, we point to the light red areas above the grey quartile line in the summer months. These indicate that during these months the actual distribution has a positive skewness, leading to a bigger "tail" towards higher temperatures than the average and standard deviation would account for.

6. Discussion, Conclusions, and Future Work

Looking at the results from our synthetic test first, we see that not only primary features are properly detected on rasterized data, but also secondary features. From Fig. 2 we measure that one unit in data space corresponds to 85 pixels, which means that if we can detect, with this exact resolution, close to $1/85 \approx 0.012$ units accuracy, then we have sub pixel accuracy.

Looking at the results in table 1 we refer to the results on the secondary feature. These results estimate the original model with not only sub-pixel precision, but with at a tenth of a pixel accuracy, on the variance. Further we see that the secondary feature's mean is estimated to a level of a third of a pixel, also sub pixel precision. On the primary feature, we see that the results are within pixels with regards to the reference, with the estimates a little on the high side. The primary feature is the first one fitted in our data visualization, and thus it includes the secondary feature in its estimate, something that can explain the pixel offset in μ. The results on accuracy are very promising, which is very interesting considering that our approach achieves these results at $O(n)$ (in time complexity, as we rasterize n points once).

An important characteristic of our approach is the high degree of interactivity. When displaying streaming data, it is important to have a visualization scheme that is able to handle large time windows, i.e., if data is streamed one needs at some point to either omit "old" data from the visualization, or support a multi resolution scheme. We have implemented visualization mappings that allow fast rendering (> 60 fps), even if we show datasets spanning several days ($> 200k$ samples). The feedback on convergence (or divergence) is also an important aspect that facilitates interactive analysis.

With this work we have demonstrated how data visualization can benefit from interactive model prototyping, externalization and subtraction so that expert users can rapidly proceed through an in depth analysis of streaming process data, following the *visualize and observe, sketch and fit, externalize and subtract, then iterate* pattern. Subtracting identified features from the data visualization allows the user to reveal secondary features and additionally results in an externalized prototype giving quantification and overview.

We have shown that interactive model prototyping in visualization space can accurately quantify measured data. Moreover, we have shown that an analyst can quickly compare suggestions for formal models, by bringing them into the visualization, perform prototyping, and get quantitative results on how well they fit. Another important part of our work has been to move visualizations beyond the initial discovery, and to give the users a view into secondary features. A general conclusion from our work is that application processes usually do not stop after discoveries in visualization and that is therefore important for visualization research to more intensely think about what has to follow visualization, e.g., externalization, quantification and ultimately action.

In future work, we plan to look further into different reconstruction techniques, and also different distributions. An interesting aspect would be to investigate the support for distributions with rotations or shear, by enabling support for a full covariance matrix, instead of the vector **r**. Another plan is to extend the piecewise linear model to support higher-order templates, like spline curves. It would interesting to consider an extension to multivariate fields, or even to three dimensional fields, using 3D rasterizing functions.

7. Acknowledgements

The work presented here is a part of the project "e-Centre Laboratory for Automated Drilling Processes" (eLAD), participated by International Research Institute of Stavanger, Christian Michelsen Research and Institute for Energy Technology. The eLAD project is funded by grants from the Research Council of Norway (Petromaks Project 176018/S30, 2007-2011), Statoil ASA and ConocoPhillips Norway.

References

[CD86] CHIN R., DYER C.: Model-based recognition in robot vision. *ACM Comp. Surveys (CSUR) 18*, 1 (1986), 67–108. 2

[DH11a] DAAE LAMPE O., HAUSER H.: Curve density estimates. *Comp. Graphics Forum 30*, 3 (2011), 633–642. 7

[DH11b] DAAE LAMPE O., HAUSER H.: Interactive visualization of streaming data with kernel density estimation. In *Proceedings of the IEEE Pacific Visualization Symposium (PacificVis 2011)* (March 2011), pp. 171–178. 2, 4

[DKH10] DAAE LAMPE O., KEHRER J., HAUSER H.: Visual analysis of multivariate movement data using interactive difference views. In *Proceedings of Vision, Modeling, and Visualization (VMV 2010)* (2010), pp. 315–322. 2, 4

[dR99] DESJARDINS M., RHEINGANS P.: Visualization of high-dimensional model characteristics. In *Workshop on New Paradigms in Information Visualization and Manipulation* (1999), pp. 6–8. 2

[GS99] GESÙ V. D., STAROVOITOV V. V.: Distance-based functions for image comparison. *Pattern Recognition Letters 20*, 2 (1999), 207–214. 4

[LKG98] LÖFFELMANN H., KUČERA T., GRÖLLER E.: Visualizing Poincaré maps together with the underlying flow. In *Mathematical Visualization - Algorithms, Applications and numerics*. Springer, 1998, pp. 315–328. 2

[LS10] LIU Z., STASKO J.: Mental Models, Visual Reasoning and Interaction in Information Visualization: A Top-down Perspective. *IEEE Trans. Visualization and Computer Graphics 16*, 6 (2010), 999–1008. 2

[Nor] NORWEGIAN METEOROLOGICAL INSTITUTE: eKlima. eklima.met.no. [Online; accessed Nov-2010]. 7

[NW99] NOCEDAL J., WRIGHT S.: *Numerical Optimization*. Springer, 1999. 5

[PVH*03] POST F. H., VROLIJK B., HAUSER H., LARAMEE R. S., DOLEISCH H.: The State of the Art in Flow Visualization: Feature Extraction and Tracking. *Computer Graphics Forum 22* (2003), 775–792. 2

[Rd00] RHEINGANS P., DESJARDINS M.: Visualizing high-dimensional predicitive model quality. In *IEEE Visualization* (2000), pp. 493–496. 2

[Sil01] SILVERT W.: Modelling as a discipline. *International Journal of General Systems 30*, 3 (2001), 261–282. 1

[SvW08] SHRINIVASAN Y. B., VAN WIJK J. J.: Supporting the analytical reasoning process in information visualization. In *CHI '08: Proc. of SIGCHI on Human factors in computing systems* (2008), pp. 1237–1246. 2

[Wal04] WALNUT D. F.: *An Introduction to Wavelet Analysis*. Springer, 2004. 2

[YXRW07] YANG D., XIE Z., RUNDENSTEINER E. A., WARD M. O.: Managing discoveries in the visual analytics proc. *SIGKDD Explor. Newsl. 9*, 2 (2007). 2

[ZSBH08] ZAMBAL S., SCHÖLLHUBER A., BÜHLER K., HLADUVKA J.: Fast and robust localization of the heart in cardiac MRI series. *Proc. of Int. Conf. on Computer Vision Theory and Applications* (2008). 2

Detecting Insight and Emotion in Visualization Applications with a Commercial EEG Headset

Daniel Cernea[1,2], Andreas Kerren[2], and Achim Ebert[1]

[1]University of Kaiserslautern, Department of Computer Science, Computer Graphics and HCI Group, Germany
[2]Linnaeus University, DFM, Computer Science Department, ISOVIS Group, Sweden

Abstract

Insight represents a special element of knowledge building. From the beginning of their lives, humans experience moments of insight in which a certain idea or solution becomes as clear to them as never before. Especially in the field of visual representations, insight has the potential to be at the core of comprehension and pattern recognition. Still, one problem is that this moment of clarity is highly unpredictable and complex in nature, and many scientists have investigated different aspects of its generation process in the hope of capturing the essence of this eureka (Greek, for "I have found") moment.

In this paper, we look at insight from the spectrum of information visualization. In particular, we inspect the possible correlation between epiphanies and emotional responses subjects experience when having an insight. In order to check the existence of such a connection, we employ a set of initial tests involving the EPOC mobile electroencephalographic (EEG) headset for detecting emotional responses generated by insights. The insights are generated by open-ended tasks that take the form of visual riddles and visualization applications. Our results suggest that there is a strong connection between insight and emotions like frustration and excitement. Moreover, measuring emotional responses via EEG during an insight-related problem solving results in non-intrusive, nearly automatic detection of the major Aha! moments the user experiences. We argue that this indirect detection of insights opens the door for the objective evaluation and comparison of various visualizations techniques.

Categories and Subject Descriptors (according to ACM CCS): Information Systems [H.5.1]: Multimedia Information Systems—; User Interfaces [H.5.2]: Evaluation/methodology—.

1. Introduction

Insight, epiphany, eureka moment, Aha! effect [Leh08]—these are all names for one of the most intriguing and even mysterious process through which humans gain knowledge. But what is insight really and how does it enrich our capacity to gain and manage knowledge? There are many definitions, each capturing a slightly different aspect of the experience. The Merriam-Webster dictionary defines insight as "the act or result of apprehending the inner nature of things or of seeing intuitively". Encyclopedia Britannica exposes it as the "immediate and clear learning or understanding that takes place without overt trial-and-error testing". Whatever the definition, all suggest the presence of a moment of extreme clarity, a moment when a solution is found that satisfies all conditions for the problem that is inspected.

While this concept has been around for centuries, it has been only introduced in psychology at the beginning of the last century [Bue11], as the German term "Aha-Erlebnis". Since then, the process of insight has been investigated from the perspective of many fields, like medicine, cognitive neuroscience and computer science, to name just a few.

At the same time, some researchers dislike any reference to spontaneous Aha! moments because it suggests irrationality. Still, many of world's most famous discoveries have been achieved by people experiencing a moment of epiphany. Isaac Newton claimed having a moment of clarity when he observed an apple falling from a tree, insight that lead to the formulation of the theory of gravity. Similarly, Friedrich August Kekulé von Stradonitz experienced the ring-like structure of benzene in a daydream [MG90].

Besides the purely knowledge-related aspects of insight, particular experiences suggest that moments of epiphany are sometimes accompanied by extremely powerful emotions, like the joy of understanding a problem or the excitement of decoding a riddle after a timely process of analysis. These moments of triumph have in many instances shown their potential to shift the emotional states of a person. Still, "the shock of recognition" is not always a side effect of the Aha! experience [Par06], and further investigation is required to establish a possible correlation between insight and emotion on insight.

Furthermore, directly detecting moments of insight is difficult, and neuroscience has struggled to capture these events in real-time. While modern methods like fMRI scans support the identification of Aha! moments [CZGR09], these approaches are still very restrictive and even intrusive operations for the subjects. Nonetheless, adjacent processes like emotional reactions generated by the excitement and joy of insight might be more simply detected by mobile brain-computer interfaces (BCI) that do not influence the person's comfort and mobility to a large extent. These BCI devices can represent the key for a less intrusive, indirect identification and observation of periods of insight, as well as a migration of insight detection to wherever it takes place without limiting the environment of its existence, i.e., medical facility.

The following sections shortly highlight related work in the field of insight research as well as EEG-based detection of emotional states and corresponding brain activity. Next, a preliminary study is presented that involves the observation of brain signals by a commercial EEG headset and the translation of these signals into emotional reactions generated by moments of insight. We highlight the results of this study, as well as capture some advantages and limitations of indirect EEG-detection of insight-related patterns. Finally, we present possible future directions of this research and summarize our conclusions.

2. Insight and Visualization

Many scientific areas have taken it upon themselves to bring clarity to the concept of insight. As a result, various characteristics of insights have surfaced during the past years, some more relevant than others in comprehending the series of processes that converge to create an Aha! moment. Studies have found that insight can be seen as a two-phase process [QZ08]. During an initial step, a subject tries to systematically explore the space of possible solutions to the task. If this approach fails to give results in a certain time-frame, an impasse is achieved that can manifest itself in the form of frustration [SAM06]. People try to overcome this impasse in a subconscious manner that builds upon relaxing the constraints of the problem or approaching it in a non-conventional manner (thinking out of the box). If the change in the mental representation of the problem is successful, the second phase is reached, the impasse is overcome, and the subconscious process suddenly and unexpectedly provides the person with a piece of information—an insight.

Studies suggest that the presence of prior knowledge about the problem or tasks as well as knowledge of one or multiple possible solutions or patterns, can interfere with the unconscious processing that leads to an insight [AFS79, WSCT00]. The reduced prior knowledge only adds to the unpredictability of this concept, which is one of its essential characteristics derived from the complexity of mental activities. In [Mar90], insights are considered in terms of pattern matching, where the mind is trying to establish an approximate fit between the set of current patterns and previous experiences. Further, a categorization is highlighted involving the information content of the epiphany in terms of anticipation: to recognize (expected information) and to notice (unexpected information).

Besides the field of psychology, various studies from medicine and cognitive neuroscience have focused on pinning down the processes and brain activity in the moment of insight. Most of these employed brain-imaging technologies, like electroencephalography (EEG) and functional magnetic resonance imaging (fMRI) [BJBFK05], to observe the brain activation patterns of subjects while solving a wide range of insight-connected problems. Participants were asked to solve specific insight problems, visual and logical riddles [LN03], and anagrams [AZKI09]. Some of these problems, like anagrams, are used because their solution can be achieved in at least two ways: through a conscious, systematic search of the space of possible solution or through sudden insight that appears abruptly in the conscious mind [KFG*08, BJBFK05]. The experiments concluded that tasks that involve problem solving via insight activate certain parts of the human brain [JBBH*04, KFG*08], leading to the possibility of detecting when a subject experiences an Aha! moment, and distinguishing this from simply finding a solution based on a systematic search.

But what about fields like information visualization that have the concept of insight at their core? Over the years, researchers have focused on defining insight and its importance for visualization [SND05, Nor06, Pla04, CZGR09]. Most famously, insight is defined as the "purpose of visualization" [CMS99], the ultimate goal by which successful representations and interactions should be measured. But how can we measure something as unpredictable and multifaceted as insight?

Most approaches in the visualization community try to achieve this by including characterizations that are aimed at defining insight in an objective, quantitative manner [SND05, Nor06], with attributes like time, complexity, domain value, depth, uncertainty, unpredictability, correctness, expectedness, and others. Attention is sometimes focused to a particular topic, like cartography [MG90], to investigate the appearance of insight when working with a certain type of representation.

Still, in many publications, the focus quickly shifts towards the importance of insight for evaluating visualizations. If insight is the purpose of all visualization, then it should also be the measure by which the quality and functionality of visualizations is determined. Currently, this is achieved in most cases by performance and accuracy experiments on restrictive benchmark tasks. Sadly, such restrictive tasks often introduce bias or capture only the performance for a particular type of task without giving answers about the performance of another. While [SND05,Nor06] highlight viable alternatives to this by suggesting open-ended protocols together a set of quantitative measures for insight, such experiments could represent an intrusion in the analysis flow of the user by introducing interruptions or imposing the think aloud method.

In the following section, we highlight an approach that overcomes some limitations of the previously presented methods, by employing a mobile non-intrusive EEG-solution for detecting moments of insight during visual problem solving.

3. EEG Detection of Emotion and Insight

As moments of insight are accompanied by powerful emotions of joy and satisfaction on discovery or comprehension, the question arises if an objective connection can be established between the Aha! moment and the emotional explosion. In order to evaluate if insight generates emotional reactions that are detectable by means of EEG measurements, we devised a preliminary experiment that focuses on capturing the feelings of users while involved in visual problem solving.

This study is based on our previous work, where we investigated the capabilities of the Emotiv EPOC wireless neuroheadset to detect facial expressions and emotional states [COEK11]. After a validation of the EEG headsets functionality in simple tasks aimed at triggering certain emotional responses, the EPOC was used as a real-time evaluator of more complex applications, like spot-the-difference tasks and computer games. A set of emotions was considered during the tasks, including engagement, excitement, satisfaction and frustration. These emotions were computed by means of the Emotiv intelligent framework that interprets the signals from each electrode to offer a real-time summary of the user's feeling. The output of the EEG device was then compared to common evaluation methods, like video log analysis and post-task questionnaires. The results of this comparison are highlighted in Figure 1.

The average differences between the EPOC results and the questionnaire answers combined with the video log transcripts showed that the emotional states captured by the EEG headset were similar to the ones reported by the users themselves. On average, the distances between the responses were under one unit on a 5-point Likert scale (strongly agree, agree, neutral, disagree, and strongly disagree), with a standard deviation of again under a unit [COEK11]. Moreover, a

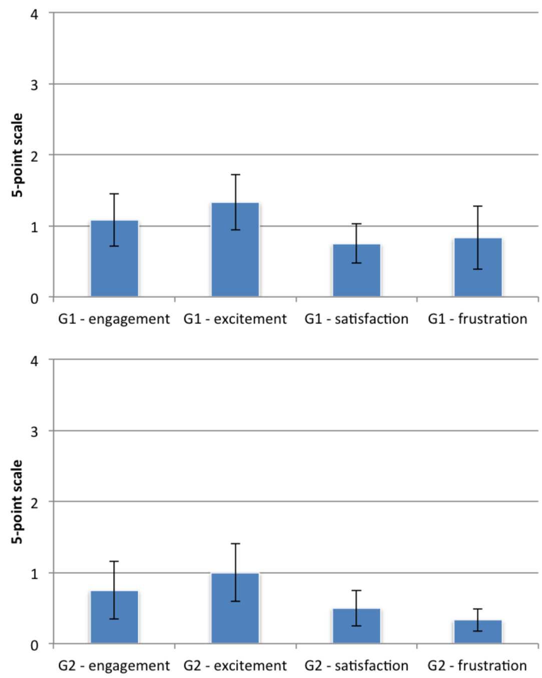


Figure 1: *Average difference between the EPOC device output and the questionnaire results for the two scenarios. Top figure, left to right: spot-the-difference task with engagement, excitement, satisfaction, frustration; Bottom figure, left to right: FPS game with engagement, excitement, satisfaction, and frustration. The distance of one unit in this 5-point scale is equivalent, for example, to the distance between "strongly agree" and "agree", or "disagree" and "neutral".*

paired sample t-test was computed in order to validate the results. Overall, the test suggested no significant difference between the subjectivity measurements and the questionnaire answer, except for one of the eight instances from Figure 1.

In this initial study, we built upon the validation of the EPOC device and its capacity to detect emotion states to explore the existence of a correlation between insight and emotion. More precisely, the spectrum of emotions that is considered in the following experiments involves only the excitement and frustration levels of the participants. The ultimate goal of this endeavor is the analysis of how well emotional states can reflect the presence of insight, and if capturing these states by EEG enables the detection of Aha! moments in information visualization techniques.

3.1. Pilot Study

The current study involved six participants with a good knowledge of visual representations and visualization techniques. The subjects were given a set of four tasks, two represented by visual insight problems and two information visualizations. For the visual riddles, the subjects had to find a unique solution, most likely resulting in a single fundamental insight. This allowed for a simple comparison of the moment of insight with the emotional states prior and during the discovery. At the same time, for the visualizations the participants were asked to find as many insights about the data as possible. For each tasks, every user had ten minutes to offer her/his insights.

Insights take time to process and clarify in the mind. Carl Friedrich Gauss said once after an epiphany: "I have the result, only I do not yet know how to get to it" [DGD04]. Therefore, once a user would report an insight, the EPOC output before this moment was inspected. More precisely, fluctuations in the level of frustration over a time period of two minutes before the insight, as well as changes in the excitement levels of the user ten seconds prior to the insight were explored.

3.1.1. Visual Insight Problems

For the visual riddles, all participants were initially subjected to a larger set of problems, of which only two were selected–Eight Coin Problem and Matchstick Arithmetic–that none of the subjects reported to know beforehand (Figure 2). For these two problems, only in 58% of all cases a solution was reached. In other words, the six participants reached an insight in 7 cases out of 12. Figure 3 highlights the correlation between insight and emotions in these cases.

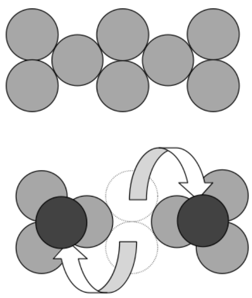


Figure 2: *Representation of the eight-coin problem. The top figure represents a possible initial configuration for the coins, while the bottom representation highlights the solution to the problem. The configuration of the coins has to be modified by moving only two coins, such that in the new grouping each coin touches exactly three others [OMC02].*

One can notice that over 80% of those who managed to solve the visual riddles have felt frustrated in the two minutes before the insights. This is also suggested by other publications, that cite frustration or deadlock as a prerequisite for the generation of an Aha! moment [MG90]. In a slightly lower percentage of cases, the subjects have also experienced excitement in the seconds prior to the insight.

While these results by themselves give us a reduced amount of information about the connection between insight and emotion, Figure 4 captures the percentage of emotional reactions for subjects that have not experienced insight at all. The lack of insight for these participants was suggested, on one hand, by their lack of a solution for the problems, but also by a post-task questionnaire that each of them filled out.

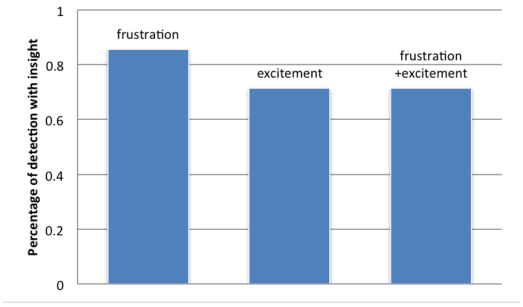


Figure 3: *Measured emotions with the EPOC headset in the presence of insight. The graph presents the average percentage of cases when frustration was detected before insight (Bar 1), excitement was detected during insight (Bar 2), and both frustration before and excitement during insight were measured (Bar 3).*

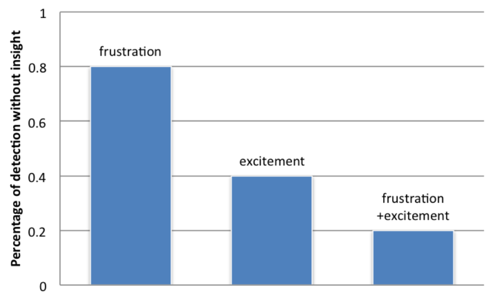


Figure 4: *Measured emotions with the EPOC headset in the absence of insight. The graph presents the average percentage of cases when frustration was detected and not followed by insight (Bar 1), excitement without insight (Bar 2) and the presence of both emotions when no insight was achieved (Bar 3).*

By inspecting both Figures 3 and 4, one notices that in both cases the frustration levels are around 80%, independent of the presence of an insight. But at the same time, the detection of excitement is much more likely in subjects that have experienced an insight. When looking at both of these emotional states, excitement and frustration were detected for 72% of the experienced insights. At the same time, the combination of excitement and frustration only appears in about 20% of the cases where no insight was gained by the subjects.

As frustration seems to be a recurring element in problem solving, the time of appearance for the feeling of frustration was analyzed. Our results suggest that states of frustration tend to be measured in participants more often during the later stages of an exploration process (second part of the ten minutes window). Also, emotional states of frustration that last longer (over one minute) are more likely to be followed by excitement, which we hypothesize might be correlated with insight. As this is a pilot study, further research involving more tasks and participants will be required to confirm these results.

The two visual problems (Figure 2) were followed by a questionnaires related to the emotional and mental states of the participants. After explaining what an Aha! moment is, we asked those that reported answers to the problems if they experienced an insight in this sense, or if it was a systematic search-and-find process that generated their answers. All the participants that experienced frustration and excitement, and that also reported the solution to the task, have confirmed that they experienced a moment of insight. On the other hand, in two instances, participants that supplied a solution and reported experiencing an epiphany were not reported by the EEG device as experiencing an increased frustration and excitement level.

3.1.2. ManyEyes Visualizations

For the information visualization tasks, we selected two visualizations from the ManyEyes website, as it harbors various datasets represented by widely accepted visualization techniques [VWvH*07]. More so, as the visualization are collaboratively explored and annotated, one can detect those that have a popular thematic and a high potential for revealing patterns and supporting hypotheses manipulation. The popularity of the visualizations was important in the selection process, as it could suggest the overall effort that users would invest in mining for insight in that representation. At the same time, a visualization that captures the tendencies of a topic that is highly relevant to the analyst has, in our view, a higher chance of generating an emotional reaction.

The two visualizations that were selected contained data about global demographics and social media, and were represented as a stacked graph and a cartographic visualization, respectively (Figure 5). The participants had an accommodation period with the ManyEyes website, during which the main supported interactions were highlighted to them. Before being the task, the participants were instructed to find all possible insights in the visualization. This approach is similar to the one in [Nor06], where insight was also observed by applying an open-ended protocol, without additional restrictions to the insights that were considered.

Furthermore, it was also suggested to the subjects to focus more towards deep insights that involve new hypotheses and multiple data types, to avoid noticing only simple facts about the data. Similarly to [SND05] and [Nor06], all spawned insights were grouped by depth into three levels: the first level refers to trivial insights that include direct observations of one data type; Level 2 insights that are generated by a combination of multiple data types or insights about a process; and Level 3 insights that refers to new hypotheses about the underlying information. The EPOC EEG headset was used to inspect the levels of emotional frustration and excitement during the interaction of the users with the visualizations.

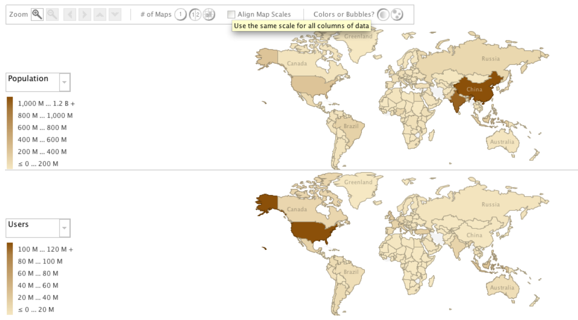


Figure 5: *Map visualization from the ManyEyes website employed in our experiments.*

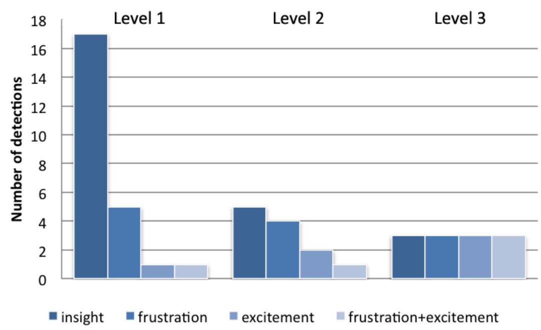


Figure 6: *Correlation between the number of insights and the instances where frustration, excitement and both frustration and excitement were detected. The results are grouped by depth of insight: the four leftmost bars represent the values for depth Level 1, the next four for depth Level 2, and the last four bar encode the number of insights and corresponding detected emotions for the deepest insights.*

Figure 6 presents the correlation between the number of generated insights and the various emotional states the users experienced. The bar chart is divided into three sections, representing the different depths of the captured insights and their corresponding emotions. The number of simple insights seems to dominate the representation, as deeper insights were more rarely detected. This fact is even more visible in Figure 7, where every single eureka moment and emotional state was plotted along a time axis.

Although the number of deeper insights is lower than the one of trivial observations, one notices the fact that deeper insights have a higher probability of generating an emotional response, especially a higher probability for excitement during the Aha! moment. This culminates in our tests for the deepest insights with a detection accuracy of 100%, via the EEG measurements of emotions, when considering both the prior experience of frustration and the excitement on discovery. Note that in Figures 6 and 7 the results of the two visualizations are convoluted, as no significant differences could be detected between the results for the two visualizations.

By using the temporal dimension, we also notice in Figure 7 that users more quickly detect the simpler insights than deep ones, and that the deep ones take more time and are less likely to be detected. Moreover, Level 3 insights are more probable to generate an emotional reaction that combines frustration and excitement, while easily noticeable facts are less likely to be accompanied by excitement. Therefore, the probability of accurately capturing an insight by measuring the emotional response of a subject via EEG seems higher when the insight is deeper, more complex, and it occurs later in the analysis process.

As previously, the participants have been asked to complete a questionnaire after employing the visualizations. Questions that were posed involved the interaction and visualizations, as well as the relevance and accessibility of the data presented in them. Many participants suggested that they did not experience the Aha! effect. Reasons given for this included the fact that the information they discovered had a low complexity and was "easy to find". Furthermore, while they were interested in the presented topic, they were not involved with it to the extent that any newly discovered insight would influence their way of thinking ("I don't think this can surprise me"). When inquired about the moments of insight, participants mentioned that they reached some answers by a simple search process. As suggested by [AZKI09], logical search for new information is a process different for gathering knowledge than the one of epiphany. Based on the questionnaire results, an even stronger correlation was noticed between the instantaneous insights that would not involve a systematic search process and the emotional responses; but as these investigations were subjective–based on open-end questions and the verbal narration of the participants' insights during the task–no quantitative values are currently available.

These answers, together with the unpredictability of insights, could represent a partial explanation for the limited number of deep insights generated by the participants. Our hope is that further experiments can generate a larger set of insights in diverse visualizations, and thus offer a more complete view of the possibilities and limitations of mobile EEG detection of insight.

Another relevant aspect for the validation of EEG measurements for detecting insight moments is given by the false positives. In our scenario, false positives are represented by moments in time when no insight is generated, but the emotional states of frustration and excitement are detected inside the time-frame described at the beginning of this section. In the second row of visualization tasks, only nine such instances were recorded by the EEG system. As the possibility exists that these were generated by elements external to the software system (real-world or mental distractions of the user), further investigation is required to deduce their impact on a larger scale. Note that an insight is implicitly considered true by the issuer, at least in the initial stager of the Aha! moment. Usually, in knowledge tasks a generated insight later undergoes a validation process that implies the systematic analysis of all facts and the insight information. This can result in an erosion of confidence, but even insights that contain false information will most likely have the potential to generate an emotional response. As a result, the EEG measurement of emotional states generated by insights should not be considered as a validation of the provided information, but as a sign of for the presence of insight.

4. Future Work

One can hypothesize about the potential of EEG measurements—and in a wider sense of emotion detection—to accurately capture the presence of moments of epiphany or insight in subjects during various tasks, like problem solving, pattern recognition and extraction, concept manipulation, etc. Although the nature of insight and emotion is clearly subjective, the presence of a mechanism for connecting these elements and reliably detecting one of them through mobile brain-imaging [COEK11] opens an entire set of possible research directions for the future.

A major opportunity in this sense is represented by the quantitative detection of insights in the process of evaluating visual applications and information visualization techniques. Especially for information visualization methods, the capacity to generate insights is the essence of a powerful representation [Nor06]. While emotional response does not quantify the power of an insight, it is capable of suggesting the presence of a reaction generated on insight. Additionally, this can suggest the relative value of the insight to the person, as our tests revealed that insights generate a detectable emotional reaction mostly if they are sufficiently deep, take a longer amount of time and effort to achieve and the topic of the problem is relevant to the subject. Therefore, in the future we plan to further investigate non-intrusive, mobile detection of emotional states of users during interaction and analysis of visualizations. Our hope is the development of new EEG-based tests for evaluating and comparing different visualization techniques, by simply looking at the number of insights that they enable. Such an approach could enable the detection of good visualization techniques and even foresee how easily users with a particular background would adopt these visualizations.

Besides evaluation of visualization techniques, the capacity to detect the moment of insight can be used for automatic

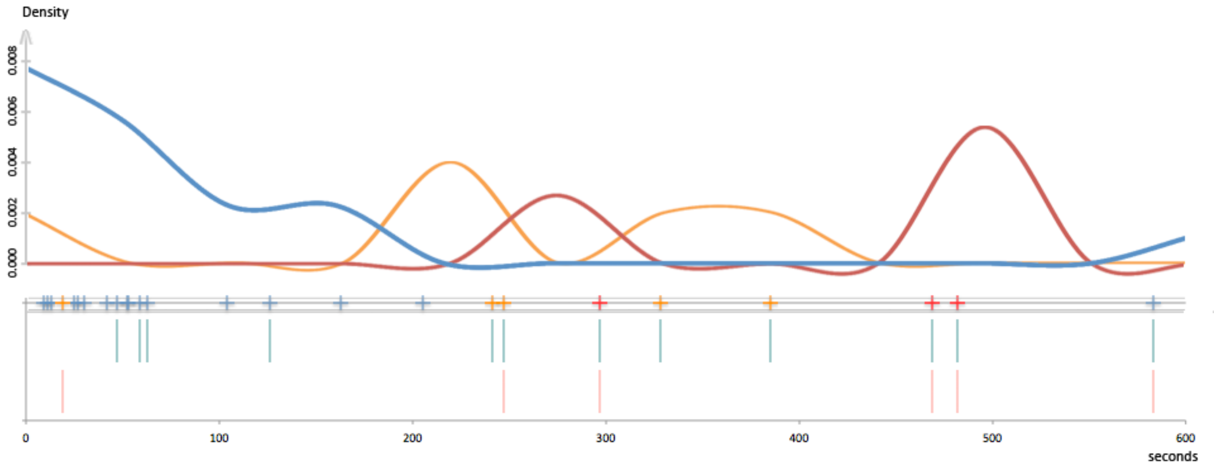


Figure 7: *Correlation between the insights generated by the participants and the emotional responses detected by the EEG headset. The insight moments are marked by 25 plus (+) signs and are plotted along the horizontal timeline. Note that all + signs are visible and there is no perfect overlap between two moments of insight. The colors of the insights represent the depth of the generated insight: blue is Level 1 (simple), orange is Level 2, and red is Level 3 (deepest). The green and red vertical lines beneath the insight + signs indicate the presence of an emotional response. A green line under a certain + sign indicates the presence of frustration previously to the insight generation. Similarly, the red line under a particular plus sign indicated the presence of excitement in the moment of insight generation. The three colored lines above the + signs represent the kernel density estimates for individual Gaussian kernels constructed around the three types of data points from the horizontal axis.*

operations like data tagging and binding based on the interactions the user executed shortly prior and during the moment of insight, highlighting of information involved in the Aha! moment and capturing layout screenshots that are relevant to a particular insight. Certainly, these methods would have to be implemented in a visualization solution that is more flexible, offering a wide range of closely coupled interaction possibilities (e.g., focus+context, InfoVis dashboards, etc.) and including a dataset that is more complex than the one highlighted in the previous section of this paper.

5. Conclusions

Insight plays a vital role in knowledge and understanding, especially in the field of visual applications. In the paper at hand, we narrowed our attention to the field of information visualization in the hope of exploring if insights in visual tasks have the potential of generating emotional responses. We measured the emotional responses of a set of subjects with the mobile EPOC EEG headset while trying to complete visual tasks: solve visual riddles and extract insight from visualizations. The obtained results have suggested not only a strong correlation between insights and feelings like frustration and excitement, but also that EEG measurements have the potential of detecting emotions corresponding to Aha! moments in an non-intrusive way. Further, it seems that the most accurate detection can be achieved if the generated insights required more thinking time, had already generated frustration in the subject, and contained potentially complex and unexpected information.

References

[AFS79] Auble P., Franks J., Soraci S.: Effort toward comprehension: Elaboration or aha!? *Memory & Cognition 7* (1979), 426–434. 2

[AZKI09] Aziz-Zadeh L., Kaplan J., Iacoboni M.: "aha!": The neural correlates of verbal insight solutions. *Hum Brain Mapp, 30(3)* (2009), 908–916. 2, 6

[BJBFK05] Bowden E., Jung-Beeman M., Fleck J., Kounios J.: New approaches to demystifying insight. *Trends in Cognitive Sciences, 9(7)* (2005), 322–328. 2

[Bue11] Buehler K.: Tatsachen und probleme zu einer psychologie der denkvorgänge. über gedanken. *Archiv für Psychologie, 9* (2011), 297–365. 1

[CMS99] Card S., Mackinlay J., Shneiderman B.: Readings in information visualization - using visualization to think. *San Francisco: Morgan Kaufmann* (1999). 2

[COEK11] Cernea D., Olech P., Ebert A., Kerren A.: Eeg-based measurement of subjective parameters in evaluations. *Proc. of the 14th International Conference on Human-Computer Interaction (HCII)* (2011). 3, 6

[CZGR09] Chang R., Ziemkiewicz C., Green T., Ribarsky W.: Defining insight for visual analytics. *IEEE Comput. Graph. Appl. 29, 2* (2009), 14–17. 2

[DGD04] Dunnington G., Gray J., Dohse F.: Carl friedrich gauss: Titan of science. *The Mathematical Association of America* (2004), 418. 4

[JBBH*04] JUNG-BEEMAN M., BOWDEN E., HABERMAN J., FRYMIARE J., ARAMBEL-LIU S., GREENBLATT R., REBER P., KOUNIOS J.: Neural activity when people solve verbal problems with insight. *PLoS Biol 2(4): e97* (2004). 2

[KFG*08] KOUNIOS J., FLECK J., GREEN D., PAYNE L., STEVENSON J., BOWDEN E., JUNG-BEEMAN M.: The origins of insight in resting-state brain activity. *Neuropsychologia. 46* (2008), 281–291. 2

[Leh08] LEHRER J.: The eureka hunt: Why do good ideas come to us when they do? *The New Yorker* (July 28, 2008), 40–45. 1

[LN03] LUO J., NIKI K.: Function of hippocampus in “insight” of problem solving. *Hippocampus 13* (2003), 316–323. 2

[Mar90] MARGOLIS H.: Patterns, thinking, and cognition: A theory of judgment. *University Of Chicago Press; 1st Edition* (1990). 2

[MG90] MACEACHREN A., GANTER J.: A pattern identification approach to cartographic visualization. *Cartographica: The International Journal for Geographic Information and Geovisualization, Volume 27, Number 2* (1990), 64–81. 1, 2, 4

[Nor06] NORTH C.: Toward measuring visualization insight. *IEEE Comput. Graph. Appl. 26, 3* (2006), 6–9. 2, 3, 5, 6

[OMC02] ORMEROD T., MACGREGOR J., CHRONICLE E.: Dynamics and constraints in insight problem solving. *Journal of Experimental Psychology: Learning, Memory, and Cognition, Vol. 28* (2002), 791–799. 4

[Par06] PARKER B.: Aha! *Trafford Publishing* (2006). 2

[Pla04] PLAISANT C.: The challenge of information visualization evaluation. *Proc. Working Conf. Advanced Visual Interfaces (AVI), ACM Press* (2004), 109–116. 2

[QZ08] QUI J., ZHANG Q.: Aha! effects in a guessing chinese logogriph task: An event-related potential study. *Chinese Science Bulletin. 53(3)* (2008), 384–391. 2

[SAM06] SAM: Scientific american mind, 2006. Oct. 2

[SND05] SARAIYA P., NORTH C., DUCA K.: An insight-based methodology for evaluating bioinformatics visualizations. *IEEE Trans. Visualizations and Computer Graphics, Volume 11, No. 4* (2005), 443–456. 2, 3, 5

[VWvH*07] VIEGAS F., WATTENBERG M., VAN HAM F., KRISS J., MCKEON M.: Manyeyes: a site for visualization at internet scale. *IEEE Transactions on Visualization and Computer Graphics* (2007), 1121–1128. 5

[WSCT00] WILLS T., SORACI S., CHECHILE R., TAYLOR H.: "aha" effects in the generation of pictures. *Memory & Cognition 28 (6)* (2000), 939–948. 2

Towards an Integrated Web-based Visualization Tool

A Comparative Survey of Visualization Techniques for Enhancing Stakeholders' Participation in Planning

P. Parsanezhad[1], U. Ranhagen[2] and Y. Ban[1]

[1] Geoinformatics division, Department of Urban Planning and Environment, KTH, Sweden
[2] Urban and Regional Studies division, Department of Urban Planning and Environment, KTH

Abstract

Digital visualization tools are widely used in planning nowadays around the world by various contributors to the field and in different planning scales. Visualization facilitates perception of underlying thoughts and objectives of planning alternatives and consequently assists with communication of the plan to different stakeholders. This, in turn, enables them to actively and efficiently participate in the procedure from the very initial stages to the implementation phase thanks to the insight provided by user-friendly visualization tools. Available visualization tools for planning, however, are either not integrated and efficient enough or too resource- or expertise-demanding and thus not entirely fulfilling the qualities mentioned above. This study is a search for a conceptual framework for an integrated web-based visualization tool. Web-accessibility diminishes temporal and spatial distance among the users and planning agents and provides the possibility for more participation in and interaction with planning projects. Within this study, major characteristics of an integrated tool have been investigated through literature, online resources, contacts with experts and practitioners, a survey over off-the-shelf products and comparative analysis of the outcomes. An evaluation cube was initially developed and used as the basis for provision of a set of dual criteria. A selection of visualization tools were examined against those criteria and results were demonstrated visually. Eventually, findings were used to provide a back-casted example of the integrated tool.

Keywords: visualization, participatory planning, web-based, CAD, GIS, urban planning

Categories and Subject Descriptors (according to ACM CCS): I.3.8 [Computer Graphics]: Applications

1. Visualization as a means for participatory planning

Demise of rationalism in planning due to social movements and widespread activities of non-governmental organizations in recent decades has paved the way for a more participatory approach towards planning [Rut85]. Terms such as *communicative planning*, *collaborative planning* and *participatory planning* swiftly found their way into the planning literature [Hea03] representing the fact that the borderline between providers and users of the plans is now blurred – if not fully lifted. Communicative planning helps strengthening social sustainability within the society by propelling planning processes towards micro levels and grassroots [MW08]. How to promote a participatory approach towards planning in practice, however, has always been a ground for discussion specifically when it comes to methods, practicalities and tools to be deployed [Bra09][CR04][Wat03][Hea03][Tew98].

Contemporary planning tools and techniques should facilitate collaboration among stakeholders during all planning phases. The more comprehensive, informative and interactive these tools are, the more equipped the planner will be to attain her/his facilitating role [CR04]. A shift in planning paradigms has thus occurred in favor of more visual approaches. Visualization is considered as a *Public Participation Spatial Decision Support System* (PP-SDSS) which is aimed at consensus-making [SK09]. The need for perceivable visualizations as a means for a widespread inclusion is perpetually emphasized by scholars [Sie06] [Sim01]. Yao, Tawfik, and Fernando contend that visual models are required in different stages of planning within collaborative urban planning support systems [YTF06]. Considering tremendous technological advances in visualization tools, it is deemed necessary to reformulate planning procedures based on these potentials aiming at more democratic planning routines.

2. Contemporary visualization tools: technological advances and empirical shortcomings

Planning instruments of our age should be capable of bidirectional transaction of ideas among planners and stakeholders and provide maximum authority for all stakeholders to participate in the process regardless of temporal or spatial distance [BCW08]. On the other hand, due to incredible technological improvements in display quality, realistic effects and Internet technology, 3D environments are now fairly available and workable similarly for experts and non-experts. It is no longer unrealistic to envision planning environments where all stakeholders can easily interact with planning proposals, submit their contributions in real-time [Ban11] and create own ideas while the simplicity and feasibility of the approach largely relies on the powerfulness of the methods for visual communication of planning proposals.

Figure 1: *The spectrum of contemporary visualization tools according to their efficiency, availability and ease of use.*

The term *Visualization in planning* refers to the optimal data representation framework designed for both planners and public within a planning procedure which uses Internet as an infrastructure for rapid, cheap and efficient dissemination of data [HJH01]. Such a combined use of visualization tools and Internet is more than essential for an efficient communicative planning practice [Bra09][See08][SDD98]. Visualization environments are largely diverse in magnitude of use, usability and efficiency. Some techniques are extensively resource-demanding such as Cubes. The most efficient and integrated available visualization systems are too complex and resource-demanding and require high level of expertise and experience to handle. This is not intrinsically a pitfall; but in practice, this has proved to notably hinder public participation i.e. in the form of digital divide [Bra09]. On the opposite end are applications which operate on ordinary PCs and laptops through the web which, in turn, are not as productive, integrated and planning-oriented as the tools of the first group. Figure 1 depicts the spectrum of contemporary visualization tools based on parameters such as efficiency, availability and ease of use.

Visualization environment for a web-based participatory planning procedure should ideally be workable for different groups of users and fairly integrated so as to bring all diverse planning issues together within a simplified and unified planning tool [Sta00]. In addition, the final product should be flexible enough to feature customized interfaces for different planning scales.

3. Digital visualization and modelling tools

The two major families of digital applications deployed in modelling and visualization for planning purposes are CAD and GIS applications. CAD applications have traditionally been common to small-scale planning projects (e.g. architecture) while GIS tools have governed the realm of territorial and regional planning. CAD solutions are usually more accurate due to their geometric base. They are often used to create three-dimensional models of built environment while GIS normally deals with two-dimensional maps. GIS applications are generally known as analytic tools compared with CAD tools which are mainly used based on their visual representation capabilities. A summarized comparison of generic characteristics of the applications within the two categories is provided in Table 1: ***Analytic comparison of capabilities and limitations of CAD and GIS applications.***

.

Table 1: *Analytic comparison of capabilities and limitations of CAD and GIS applications.*

CRITERIA

	CAD	**GIS**
Common Use in Planning and Design	Architecture, Urban Design	Urban Planning, Community Planning, Regional Planning
Scaling to Needs	Scales Not (Too Geometric)	Scales Well (Geography to Geometry)
Planning And Design Capabilities	Limited Flexibility And Possibility For Design	Suitable Instruments For Planning
Dominant Visualization Mode	High Realistic Visualizing Capabilities	Schematic Visualizing Capabilities
Analytic Strengths	Few Analytic Capabilities	Analytic Functions for Modelling Systems
Dominant Content	Physical Form	Natural and Socio-Economic Phenomena
Dominant	High Virtual	Thematic

Presentation Capabilities	Reality Capabilities	Representation Capabilities
Number of Alternatives	Increase in Number of Alternatives	Increase in Number of Alternatives/Scenarios
Automated Modelling	Operator-Demanding Modelling	Semi-Automated Modelling
Accuracy	High Accuracy Due to Geometric Base	Limited Accuracy
3D Visualization	Workable 3D Environment	Mainly 2D, limited 3D

This is now however something of the past. The two families are moving closer over time. This converging attitude is actually manifested in new products of the pioneer firms within the two categories e.g. Revit by Autodesk and 3D Analyst by ESRI. Besides, a closer interaction with the Internet and a variety of web-based products and functionalities can be monitored in both groups.

Innovative technological features such as VRML (in the past) and initiatives of pioneer companies e.g. Autodesk, ESRI and Google (quite recently) have been substantial to this scenario of integration. Widespread use of Google Sketchup and Google Earth by public and planners for creating a shared digital globe is an articulate example. Digital models of the buildings can now be created in Sketchup, located on their exact place on Google Earth and linked to their corresponding web-pages. This is nevertheless more appropriate for visual representation of already existing built features rather than planning practices. Some recent efforts have been done to use the same routine for planning purposes as well such as *City Planner* initiative by *Agency 9* [Age10]. Geometric models of Sketchup are nontheless not constructed according to informative hierarchical patterns. This hampers linking the models to databases; something which is required for integrated planning purposes.

4. Aims, objectives and scope of the research

This study is a search for a conceptual framework for an integrated web-based participatory planning tool. Based on the literature, it is hypothesized that provision of agile, distributed and astute routines for visualization is the key to a better communication with stakeholders during planning period. The sought-after integrated visualization tool is primarily aimed to provide solutions at urban scale.

This study, in particular, examines the capacities of available visualization tools and techniques to be deployed for maintaining a collaborative planning procedure. This has been realized through a thorough search for theories and literature on use of visualization in planning, a comprehensive survey over a set of state-of-the-art visualization applications followed by evaluation and analysis of the findings. Finally, generalities of a back-casted visualization interface are presented.

5. Background activities and institutional frameworks

The point of departure for this study was the project *KTH Classroom Search Engine & 3D KTH Virtual Campus* initiated by the *Division of Geo-informatics* in *School of Architecture and the Built Environment of KTH* in 2009. Within the project, digital exterior and interior models of the buildings throughout KTH main campus were constructed through footprint extrusion and enhanced with photorealistic mapping of the façades using Google SketchUp. Challenges confronted during presentation of digital models of the buildings in the ensuing seminar partly inspired the idea behind this study. Later on, getting in touch with ViSuCity (*Visual Sustainable City*) project helped to consolidate the subject as an interdisciplinary master thesis and, at the same time, a part of working package 8 (WP8) of the *ViSuCity* project [Ban11].

6. Methodology

Current study is a qualitative research initiating with a descriptive and exploratory survey followed by an evaluative phase and a brief prescriptive supplement at the end. Within the first phase, the overall design of the research was formulated and narrowed down to its most essential components.

Interviews and discussions with experts of visualizations helped developing an overall insight to the complicated and multi-disciplinary area of visualization. Choice of experts within the focus group (professionals in relevant fields) was based on their being competent and well-acknowledged within their fields of expertise and their availability. Utmost effort was also made to select people with various backgrounds to cover diverse aspects. At this stage, generalities of the field were discussed through face-to-face meetings and without any questionnaire or detailed agenda. The intention was to facilitate the exploratory approach of the initial phase and help discovering new areas of concern which were deemed necessary for developing a holistic insight.

Online search was carried on in two different ways: searching through websites, weblogs and web-catalogues of companies active in visualization and planning authorities as well as acquiring *Google Alerts*. The latter is a service which is activated in Gmail and periodically sends a set of relevant links to user's inbox based on the keywords provided. *Visualization* and *planning* were the main keywords used which were over time combined with other words and phrases such as *participatory* and *urban planning* so as to enhance the results. Proposed web pages were regularly monitored and filtered based on their relevance and importance and findings were incorporated into the research process. This approach helped access the state-of-the-art actors, activities, findings, trends and knowledge in the field which were later on used for choosing case studies.

Study of bibliographical resources helped discover history and basics of visualization in planning and also introduce a set of relevant disciplines e.g. collaborative planning, city modelling, virtual reality, augmented reality, Geographic Information Systems, Public Participatory GIS, Web 2.0, neogeography, e-government and decision support systems. These contributed to a more comprehensive understanding of the context over which the research question was to be examined and provided with a holistic solution.

Simultaneous use of the aforementioned sources of information contributed to accurate delimitation of the problem to be studied. Web-accessibility was also identified as a momentous quality for the looked-after visualization tool. The research question was consequently developed to: ***What are the main characteristics of an integrated web-based visualization tool for enhancing stakeholders' participation in planning procedures?***

In search of the integrated web-based visualization tool and based on the knowledge and insight already gained, an evaluative survey over a set of case studies was then conducted. Case studies were selected through online search. The main criteria for choice of visualization tools were being technically avant-garde and relatively outstanding in capabilities and possibilities. The visualization demonstrator of ViSuCity, Neo4 UrbanPlanner, was also considered as one of case studies. Examples were intended to possess favoured characteristics discovered through previous phase of study. Selected visualization tools were studied and generally acknowledged through a survey over their websites, weblogs, catalogues, manuals, user feedbacks, demo versions and available literature also contacts with developers and marketing agents through e-mail, phone call, net-meeting and meeting sessions.

An evaluation cube was then developed to introduce a method based on five parameters for evaluation of the whole range of visualization tools. The idea of the cube was basically taken from Prof. Ulf Ranhagen's notion (cited in [Ban11]). Here, however, the cube was used in a slightly different sense: as a method for evaluation of existing planning/visualization tools which will, in turn, be used to clearly outline characteristics of the integrated web-based planning tool. This evaluation method demonstrated in Figure 2 is based on locating existing visualization and planning tools onto the units within the cube. Major criteria which have been proposed for evaluation of each package are *efficiency* and *workability*. SWOT analysis method is proposed for a more comprehensive survey over each case (strengths, weaknesses, opportunities and threats). The three factors of *user groups, planning aspects and planning scales* and criteria such as *efficiency* and *workability* which had been proposed within the cubic model were then expanded to a set of dual criteria aimed at better elaboration of the optimal tool. Functioning in 2D or 3D environments, determining an integrated or specific approach, being user-friendly or expert-oriented, more interactive or more informative, suitable for planning or mere visualizing purposes, possessing visual or information-rich architecture, being workable or resource-demanding, participatory or excluding, using realistic or schematic representations and being interoperable or self-sufficient were the criteria set for further evaluation of case studies.

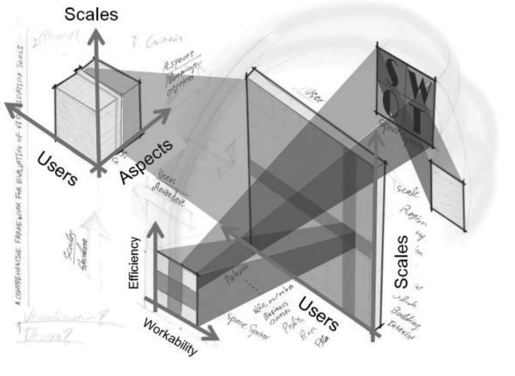


Figure 2: *The evaluation cube.*

A number of prominent visualization applications were then selected, studied, categorized and examined against dual criteria. Eventually, findings were applied to devise a back-casted proposal as a concrete example of an integrated web-based visualization tool for participatory planning.

7. Case studies

Selected case studies to be examined against dual criteria are: City Engine, City Maker, City Planner, Urban Circus, Neo4 UrbanPlanner and Symbiocity Scenarios.

City Maker™ is a multidisciplinary 3D visualization platform which is used in variety of fields including urban planning, management, administration, surveying, architecture, transportation, emergency, power and utilities. The product operates in close relationship with GIS applications and exchanges a variety of file formats such as DXF and DWG with planning packages. Mass data processing, delicate visual effects and interoperability are the most prominent features of City Maker. It has nonetheless an expert-oriented configuration and is not workable and user-friendly enough so as to be efficiently used by lay users. CityMaker has been developed jointly by Digital City Research Center of Beijing Tsinghua Urban Planning & Design Institute and Gvitech Technologies. [GVI10].

City Engine operates in real-time and possesses an interactive interface [Cie10]. Using *procedural* methods for rapid creation of urban fabric ruled by generic geometry-creation grammars is the main principle in City Engine [PM01]. This real-time creation of photorealistic representations of a fictive city or district based on a set of parameters which are set by users can be efficiently used in web-based visualization of the future. Nevertheless, City Engine is primarily focused on physical manifestation of built environment and should also be linked to dynamics of the city to be developed into a visualization tool for planning. Moreover, it is currently a stand-alone application. City Engine is the flagship product of the Zurich-based company, *Procedural Inc.*

City Planner is a user-friendly, web-based visualization tool for creating, sharing and communicating future urban plans. Inputs to the program are 3D models of urban entities constructed in SketchUp, Maya or 3D Studio Max. Digital models of planning proposals are placed into their global coordinate space and made available online for being observed, visually analyzed and evaluated by stakeholders. The product provides the infrastructure required for adding geo-referenced feedback to a developing urban plan [Cip10]. The tool is however mainly intended for visualization of planning alternatives rather than actual urban planning practice and does not generally include analytic functions. City Planner is developed by *Agency 9 AB* [Age10].

Among available interactive visualization products, *Neo4 Urban Planner* is one of the pioneers in creating realistic, static and animated outputs integrated with analytic planning tools. The product supports a variety file formats such as COLLADA, CityGML and those of ESRI ArcGIS. Neo4 Urban Planner is developed by Sightline Vision AB, Stockholm-based company which has been founded in 2000. It has undergone some modifications over time so as to expand its area of functionality e.g. in order to also perform social visualization [Sig10]. More web-based functionalities have been recently added to the product e.g. interactive multi-criteria evaluation for planning. Models have been linked to Google Maps and it has been made possible to add comments and sketches to the model and get feedback in real-time. Neo4 Urban Planner is one part of the *demonstrator* of the ViSuCity project.

Symbiocity is not basically a planning product but a trademark bringing together thousands of Swedish companies so as to provide a multidisciplinary and framework for marketing Swedish sustainable planning products and services around the globe. Symbiocity was established in 2008 and is administered by *Swedish Trade Council*. Symbiocity Scenarios is an online game within Symbiocity website which visualizes consequences of a set of planning strategies on a virtual city in real-time. The principals behind the user-friendly interface of the application can be deployed for developing a professional planning package with a participatory approach [Sym10]. In other words, *SymbioCity*'s most impressive features are realistic representation of urban features and dynamisms and availability through the Internet which facilitates collaboration and offers a fairly interactive and workable interface.

Urban Circus visualizes planning proposals with lots of realistic details in real-time and within a four-dimensional environment. It covers a variety of planning issues in different phases and is highly interoperable. Nonetheless, Urban Circus is not very participatory and its interactivity is mostly limited to navigation tools and presentation modes rather than decision-making and alteration possibilities. This visualization software takes input from 3DSMax, Maya and ArchiCAD among all. Outputs range from 2D rendered scenes, 3D panoramic view, 3D videos, 4D planning environments and interactive web pages. Urban Circus Company is based in Brisbane, Australia and has been founded by an urban planner, Dr. Ben Guy in 2004 [Urb10].

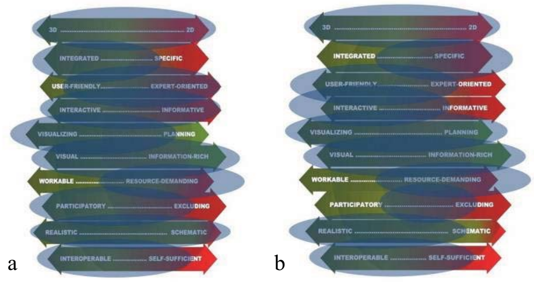


Figure 3: *(a) City Maker, (b) City Engine.*

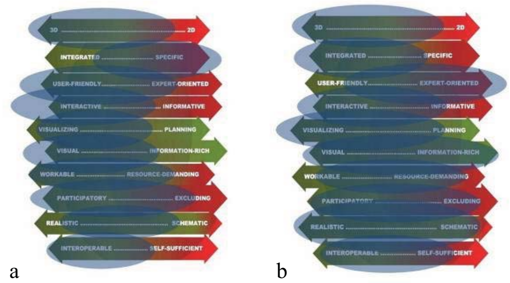


Figure 4: *(a) City Planner, (b) Neo4 Urban Planner.*

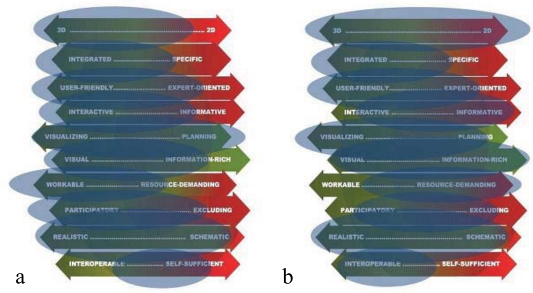


Figure 5: *(a) Symbiocity Scenarios, (b) Urban Circus.*

Figures 3-5 depict a summary of evaluation of the six case studies against dual criteria developed earlier within this research. Blue spots and their proximity to one end of each pair of dualities represent that the product possesses those characteristics. Hence, most-favored situations are where green areas and blue spots coincide the most.

8. Conclusion

The purpose of the evaluation phase described above is not ranking or categorization of the cases. Judgments are carried out in an approximate way and based on available sources of information. Diagrams convey merits and pitfalls of selected packages in a clear and concise visual setting. Findings through this approach are complementary to specifications for the integrated visualization tool acquired previously through literature review and online search.

Based on evaluation diagrams the following conclusions can be drawn:

- All studied cases possess three-dimensional visual interfaces which are deemed necessary for visualization tools.
- Almost all case studies provide realistic visualizations and they often also use schematic representations in combination for specific purposes.
- A majority of selected products are fairly interoperable and interactive. Only SymbioCity Scenarios is not efficient at exchanging inputs and outputs with other packages which is not a deficiency in essence since it has not been design as a planning tool.
- Most cases are user-friendly and information-rich. This is in line with the fact that the to-be-developed web-based visualization tool of tomorrow should be easy-to-use and also better connected to databases so that it fulfills the requirements for an integrated visualization media.

It goes without saying that there will still be a need for sophisticated expert-oriented visualization tools with numerous menus, functions and parameters to be accurately set by experts of relevant disciplines; but these will no more contribute to the participatory approach which has been of the main focus for this study. The same argument holds regarding an integrated approach towards web-based visualization tools: there will still be a need for specific visualization tools in the fields of e. g. lighting, noise, landscape, infrastructure, transportation, security, safety, circulation, line of sight, pollution, energy efficiency, etc. Whilst the communicative visualization tool should be capable of providing a holistic insight to lay users and offer an integrated yet workable solution.

Workability is the quality that a few examples possess. Even though technological developments enhance workability of digital tools on a daily basis, it is still quite necessary for the integrated tool to avoid resource-demanding solutions so as not to dampen the collaborative attitude by triggering digital divide [Bra09]. The evaluation cube, dual criteria, evaluation diagrams and this concluding part altogether envision the overall structure of the conceptual framework for the integrated web-based visualization tool. In the following, a further step is taken to provide a concrete back-casted example of the realized visualization tool which could satisfy requirements of the already-developed conceptual framework.

9. Proposal for the Integrated Web-Based Visualization Tool

Figure 6: *Back-casted interface of the proposed integrated web-based visualization tool.*

Figure 6 features the back-casted interface on a normal PC or laptop screen. The consequences of the changes made by the user to planning parameters through adjustment bars on the right side can be viewed within the window on the left side in real-time. The user can navigate freely throughout the scene to examine the outcome from different points of view. The change in values is only allowed within specific ranges. Figure 7 demonstrates the output of moving the *Green Area* slider to the right. New green areas are optimally located in various spots around the district.

Figure 7: *Real-time visualization of a change in the parameter Green Area by the user.*

Change of a parameter by the user can evidently be visualized in numerous ways. The optimal alternative should thus be produced and presented by the tool through pre-designed algorithms which are in turn based on local planning regulations and routines and implications of available spatial data. When all values are set, the visual consequences are carefully studied and examined from various points of view and the optimal configuration is envisioned, the user can publish her/his planning alternative by pressing the *Vote* button. **All users are thus free to compose their own planning alternatives through an intelligent and fully customized visualization/planning tool instead of choosing among a limited set of planning alternatives.** Based on the area of responsibility and scientific qualification of the user, a Multi-Criteria Evaluation (MCE) tool which is incorporated in the system assigns corresponding weights to each category of voters before putting the results together and developing an overall alternative or a set of collaborative choices.

Customization to different user groups and planning scales is an essential quality for a visualization tool. Adjust controls for planning within district level, for instance, will be totally different from those in the case of planning procedure of an urban block. Besides, the degree of realistic visualization should correspond to the planning stage. This example should only be considered as one of numerous possibilities for the integrated web-based visualization tool provided as a complement to theoretical findings.

10. Discussion

A major challenge in this study was the perpetual risk of using outdated data (literature, reports, software tutorials, technical recommendations, etc). New technological features in the realm of computer graphics in general, and web-based visualization tools in particular, are being developed and immediately presented to planners and public on a daily basis. Reliable scientific materials of a couple of months ago can thus easily be part of the history at the time being. Low band-width of the Internet, for example, was once considered as a serious impediment to transfer of photorealistic visualizations to the public [SDD98] which is no longer pertinent in the field of web infrastructure. Of the four types of virtual cities that Smith, Dodge, and Doyle introduce and describe (Web Listing Virtual Cities, Flat Virtual Cities, 3D Virtual Cities and True Virtual Cities) only True Virtual Cities can now be considered as a pertinent case within urban modelling discourse [SDD98]. The three other definitions refer to some abstract or symbolic models which do not comply with contemporary virtual environments' requirements. Nevertheless, utmost care has been taken to recognize and eliminate outdated materials based on findings of the complementary online part of the study.

Another area of uncertainty was differentiating between visualization, planning, presentation, drafting and enhancement tools. According to definitions of the term *visualization* the concept is closely intertwined with data representation and thus goes far beyond the mechanical act of virtual construction of an urban element in a schematic or realistic manner [BAS00][SDD98][OHD99]. On the other hand, a visualization tool is a medium which visually interprets planning ideas and also helps examine, alter and introduce planning proposals; but it should not necessarily act as the core of planning procedure. The interrelationship between the two concepts of planning and visualization makes it difficult to define the boundaries of the research and seems to cause problems particularly in choice of examples of visualization tools. To better comply with the definitions of the term, however, ubiquitous tools such as Auto Cad, Micro Station, Revit, 3D Studio Max, Maya and Rhyno were not selected as case studies.

Another challenging ground was avoiding contradictions when defining criteria for an efficient visualization tool. A fully integrated package, for example, which has taken a variety of disciplines into account, normally becomes so complicated that can no longer be realized and used by non-expert stakeholders. The fact that there is always a limit to the extent drastically diverse requirements can be met by the sought-after visualization tool should always be taken into account.

References

[Age10] (n.d.). *3D Maps*. Retrieved June 4, 2010, from Agency 9: http://www.agency9.com/about/

[Ban11] Ban, Y. (n.d.). *ViSuCity - A Visual Sustainable City Planning Tool*. Retrieved May 23, 2010, from: http://www.infra.kth.se/GIS/research/projects/ViSuCity/.

[BAS00] Blaser, A. D., Sester, M., & Egenhofer, M. J. (2000). Visualization in an Early Stage of the Problem Solving Process. *GIS Computer & Geosciences* , 26(1), pp. 57-66.

[BCW08] Ball, J., Capanni, N., & Watt, S. (2008). Virtual Reality for Mutual Understanding in Landscape Planning. *International Journal of Social Sciences* , Volume 2 Number 2 2008 ISSN 1306-973X.

[Bra09] Brabham, D. C. (2009). Crowdsourcing the public participation process for planning projects. *Planning Theory* , 8(3), 242-262.

[Cie10] *CITY ENGINE*. Retrieved 08 20, 1010, from http://www.procedural.com/cityengine/features.html

[Cip10] *CITY PLANNER*. Retrieved 08 20, 2010, from http://cityplanner.agency9.com/cityplanner/

[CR04] Connelly, S., & Richardson, T. (2004). Exclusion: The Necessary Difference between Ideal and Practical Consensus. *Journal of Environmental Planning and Management* , Vol. 47, No.1, January 2004.

[GVI10] *GVITECH*. Retrieved 08 20, 2010 from http://www.gvitech.com/en/simulation.asp

[Hea03] Healey, H. (2003). The Communicative Turn in Planning Theory and its Implications for Spatial Strategy Formation. In S. Campbell, & S. S. Fainstein, *Readings in Planning Theory. Second Edition* (pp. 237-255). Blackwell Publishers.

[HJH01] Huang, B., Jiang, B., & Hui, L. (2001). An integration of GIS, virtual reality and the Internet for visualization, analysis and exploration of spatial data. *International Journal of Geographical Information Science* , 15: 439-456.

[MW08] Mannberg, M., & Wihlborg, E. (2008). Communicative Planning – Friend or Foe? Obstacles and Opportunities for Implementing Sustainable Development Locally. *Sustainable Development, Vol. 16* , 35-43.

[OHD99] Orford, S., Harris, R., & Dorling, D. (1999). Geography: information visualization in the social sciencesa state of the art review. *Social Science Computer Review* , 17, 289-304.

[PM01] Parish, Y. I. H., Müller, P., 2001. Procedural Modeling of Cities. In Proceedings of ACM SIGGRAPH 2001, ACM Press / ACM SIGGRAPH, New York. E. Fiume (ed), COMPUTER GRAPHICS, Annual Conference Series, ACM, pages 301-308.

[Rut85] Rutland, P. (1985). *The Myth of the Plan : Lessons of Soviet Planning.* Chicago, IL, U.S.A.: Open Court: 1985 0812690052 / 9780812690057 First Edition Hard Cover Near.

[SDD98] Smith., A., Dodge, M., & Doyle, S. (1998). *Visual communication in urban planning and urban design.* Centre for Advanced Spatial Analysis. Case studies of visualization in the social sciences. ISSN: 1467-1298. http://www.agocg.ac.uk/reports/visual/casestud/contents.htm.

[See08] Seeger, C. (2008). The role of facilitated volunteered geographic information in the landscape planning and site design process. *Geo Journal* , 72 (3): 199 – 213.

[Sie06] Sieber, R. (2006). Public Participation and Geographic Information Systems: A Literature Review and Framework. *Annals of the American Association of Geographers* , 96/3:491-507.

[Sig10] *SIGHTLINE VISION*. Retrieved 08 20, 2010, from http://sightline.verkstad.org/se/wwwtmse/neo/

[Sim01] Simpson, D. M. (2001). Virtual reality and urban simulation in planning: A literature review and topical bibliography. *Journal of Planning Literature* , Vo. 15, No. 3, Feb. 2001: 359-376.

[SK09] Shen, M., & Kawakami, M. (2009). An online visualization tool for Internet-based local townscape design,Computer, environment and urban systems. *Computers, Environment and Urban Systems* , 34 (2010) 104–116.

[Sta00] Stabinsky, D. (2000). Bringing Social Analysis into a Multilateral Environmental Agreement: Social Impact Assessment and the Biosafety Protocol. *9(3): 260–283*.
SYMBIOCITY. (2010, Decedmber 6). Retrieved from http://www.symbiocity.org

[Sym10] *SYMBIOCITY*. (2010, Decedmber 6). Retrieved from http://www.symbiocity.org

[Tew98] Tewdwr-Jones, M. (1998). Deconstructing communicative rationality: a critique of Habermasian collaborative planning. *Environment and Planning* , A 1998, volume 30, pages 1975-1989.

[Urb10] *URBAN CIRCUS*. (n.d.). Retrieved 08 20, 2010, from http://www.circus3d.com/index.php

[Wat03] Watson, V. (2003). Conflicting Rationalities: Implications for Planning Theory and Ethics. *Journal of Planning Theory & Practice* , Vol. 4, No. 4, December 2003. p. 395–407. Seminar 3 The planner and the planning process.

[YTF06] Yao, J., Tawfik, H., & Fernando, T. (2006). A GIS based virtual urban simulation environment. In *Computational Science, Lecture Notes in Computer Science* (pp. 60-68). Verlag: Springer, LNCS 3993, Part 3.

A versatile material reflectance measurement system for use in production

Gabriel Eilertsen, Per Larsson and Jonas Unger

The Division of Media and Information Technology
Linköping University

Abstract

In this paper we present our developed bidirectional reflectance distribution capturing pipeline. It includes a constructed gonioreflectometer for reflectance measurements, as well as extensive software for operation, data visualization and parameter fitting of analytic models. Our focus is on the flexible user interface, aimed at material appearance creation for computer graphics, and targeted both for production and research employment.
Key challenges have been in providing a user friendly and effective software for functioning in a production environment, abstracting the details of the calculations involved in the reflectance capturing and fitting. We show how a combination of well-tuned tools can make complex processes such as reflectance calibration, measurement and fitting highly automated in a fast and easy work-flow, from material scanning to model parameters optimized for use in rendering. At the same time, the developed software provides a modifiable interface for detailed control. The importance of having good reflectance visualizations is also demonstrated, where the software plotting tools are able to show vital details of a reflectance distribution, giving valuable insight in to a materials properties and a models accuracy of fit to measured data, on both a local and global level.

1. Introduction

Visual realism and predictive rendering results are the central challenges in many computer graphics applications. These aspects are especially important in applications such as product visualization and rendering for commercials, where the synthesized images cannot in any way deviate from the corresponding real world objects. The key factors in the creation of high fidelity renderings are accurate modeling of the scene light transport and scattering events. This has led to research and development of advanced material models, so called bi-directional reflectance distribution functions (BRDF). A BRDF is a 4D function describing the reflectance and spectral characteristics of a material at a point on a surface with only a small number of parameters, typically three to five.

BRDFs are traditionally modeled by hand by an artist who adjust the parameters such that the model mimics the properties of the desired real world material. This is, however, a time consuming and difficult task, and the result is also often not fully reliable. A major difficulty for the artist during BRDF modeling is to understand which visual effects of the material are introduced by the BDRF and which effects that are caused by the virtual light setup used during modeling.

This has led to the introduction of methods for measuring real world materials [WLL*08] and using this information for parameter fitting of BRDF models have been proposed. Using measured material data enables highly accurate numerical modeling of real world BRDFs in a fast and convenient way. This, however, comes with the drawback that the measuring and parameter fitting procedures are very complex, and requires deep understanding of material modeling, color science, and numerical optimization techniques. It is therefore highly important to develop tools that make such systems viable for use by artists in fast paced production environments.

In this paper, we present an overview of our pipeline for measuring, processing and fitting of BRDF parameters to real world material properties. Our pipeline consists of a custom built high accuracy camera based gonioreflectometer, and a software pipeline for data processing and BDRF parameter fitting. Our software framework has been specifically developed to support artistic requirements of non-technical experts and includes a set of carefully selected techniques for visualization of the data at each stage in the process, and interaction tools which gives the user full controll over the parameter fitting process.

2. Background

Having a good estimation of a materials reflectance properties plays an important role in many different areas, such as in the paper, textile and color industry; and to a large extent in computer graphics where the rendering equation used for creating synthetic images is governed by the bidirectional reflectance distribution function [Kaj86]. The BRDF specifies for each light direction incident at a surface point on a material, how it is distributed over the hemisphere above the point when reflected. It is usually formulated in spherical coordinates with four dimensions for the incoming and outgoing polar and azimuthal angles, $\rho(\varphi_i, \theta_i, \varphi_o, \theta_o) = \rho(\vec{\omega}_i, \vec{\omega}_o)$.

A number of models for describing BRDFs have been proposed – both empirical (such as the Blinn-Phong [Bli77] and Ward [War92] models) and physically based (*e.g.* the Cook-Torrance model [CT81]). For many material modeling purposes such models can describe the reflectance well. However, given a real-world material which should be reproduced in rendering it is not straightforward to set the parameters of a BRDF model by hand cause of the high dimensionality; although the modeling may be perceived as accurate from a certain viewing direction and under certain lighting conditions, it can behave differently in other circumstances. Having parameters that are fitted to a reflectance measurement of the material instead, ensures that the model is used globally optimized to the sought material.

Constructions built for reflectance measurements are named gonioreflectometers and classically utilizes a four degrees of freedom setup, where a light source and a detector can be placed at any angle in the hemisphere above a material sample point. This usually provides a high precision and well-controlled measurement environment where one measurement is done for each light source/detector positioning. Examples of such high precision and multi-spectral measurement devices are *e.g.* the Spectral Tri-function Automated Reference Reflectometer (STARR), developed at National Institute of Standards and Technology (NIST) [PB96], or the examples from the Physikalisch-Technische Bundesanstalt [Erb80, HGH06]. While these represent expensive high standard equipment, a simpler gonioreflectometer with commercially available components and a straight-forward design was presented by [WSB*98].

With image-based techniques it is possible to use information from different pixels on the sensor for simultaneous capturing of a set of reflectance samples, resulting in a significant increase in speed of a BRDF scanning. However, with measurements on pixel level it is difficult to match the high precision of classic single detector gonioreflectometers. If the data should be used for optimizing an analytic model, this is not a significant problem, since small inaccuracies have little effect on the final outcome.

Different methods have been presented for performing multiple angle readings. One is by having a curved mirror reflecting different angles onto different parts of the image plane. In [War92], Ward captures the reflectance for all excitant reflectance angles in the same image using a hemispherical mirror and a fish-eye lens, yielding fast measurements. The concept with an ellipsoidal mirror, but in a different setup, was also brought to a small hand-held device in [DR97, MDL*98], aimed at use in the industry together with a graphical user interface (GUI) for operation. An alternate approach is to have a parabolic mirror focus on a measurement point on a material sample, with light incident on the mirror, so that it shows an image of the reflectance for a set of angles [DW04].

Instead of having mirrors to enable multiple reflectance readings, a material geometry other than planar can be used to create an angular image. In [MWL*99, MWLT00] spherical samples were used together with a light source to cover a large part of the outgoing reflectance angles in one image. The technique was also extended to include general concave objects by having their geometry known, *e.g.* by scanning. A similar approach for fast measurements was taken in [MPBM03], and extended in [NDM05] to incorporate anisotropic materials by performing measurements on a set of material strips on a rotating cylinder, where the strip orientations represent different azimuthal light angles.

3. System overview

The reflectance measurement setup is shown in Figure 1, and is related to the one described in [WSB*98], with four stepper motors controlling the four axes of incident and excitant light directions, but it utilizes a CCD camera as reflectance detector. The 4-axis construction offers greater flexibility and improved quality compared to image-based multi-angular capturing aproaches, with the possibility to average a measurement over a region. Furthermore, the setup supports continuous movement of the rig while the camera is reading, where capturing can be done at over 20 positions per second.

The built device utilizes 0.002° resolution stepper motor rotary tables, a 14-bit 1388x1038 pixels CCD sensor firewire camera, and a halogen light source with equalizer for a temporally consistent output. With a rig construction like ours, where the camera and the light source are kept on arms rotating above the material sample, the angles close to retro-reflection are occluded. In our case the camera is placed closer to the material than the light source, which result in an occluded region of about $\pm 5^\circ$. Except for these angles, all directions in the hemisphere can be captured at high angular precision and accuracy.

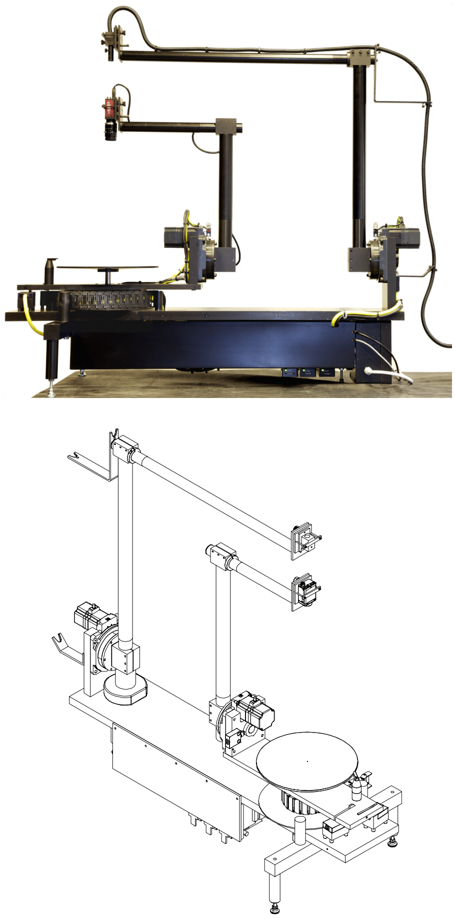


Figure 1: *The capturing device with stepper motors controlling rotation of the arms. Instead of having the light arm rotating for different incident azimuthal angles, the material plate can be rotated.*

A reflectance scanning is performed by having the camera doing continuous sweeps over the material, capturing the reflectance at predefined positions as the average over a user specified pixel radius. Denoting the incoming light direction (φ_i, θ_i) and the outgoing direction (φ_o, θ_o), where φ and θ are the azimuthal and polar angles respectively, this means that the camera sweeps are accomplished by having φ_i, θ_i and φ_o fixed while θ_o is continuously changed.

To be able to cover the dynamic range of reflectance, multiple exposures are needed [DM97]. Having the camera moving continuously over the material sample, it would be inefficient to stop at every saturated sample position and capture with shorter exposure times. Instead, the camera completes a sweep over the material ignoring over-exposed samples. Subsequently, the camera is swept over these sample positions using a shorter exposure time. The procedure is repeated until there are no saturated samples left to scan.

To enable simultaneous threaded calculations of reflectance, the captured images in a material scan are directly stored in a queue which is processed in a second thread, so that the reflectance evaluation processing does not become a bottleneck. The processed samples are then stored as unstructured data in four dimensional space, where each sample is assigned an incoming and outgoing light direction. The unstructured storing is to have an extendable approach for data storage, *e.g.* enabling adaptive sampling schemes. When used in visualizations the data is extrapolated onto a 4D uniform grid for simple and fast look-up.

4. Visualization and interaction

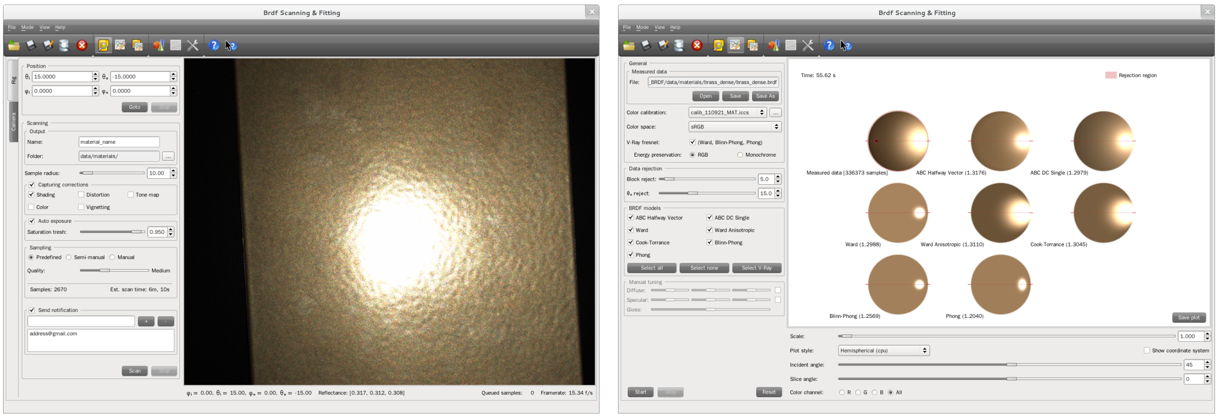
In providing a versatile system for physically based material reflectance modeling, we have developed comprehensive tools for measuring, viewing and fitting model parameters to BRDF data. With the software we try to address a number of general requirements we put on our system, which can be summarized as follows:

1. Users should be able to perform calibration of the system, *e.g.* for correct color output.
2. Performing a material scan should be on a selected level of abstraction; that is, selected from predefined settings or in a more manual approach.
3. Visualization of BRDF data should be able to show detailed reflectance properties in an informative manner, and make viewing of the high dimensionality intuitive.
4. Optimization of analytic BRDF model parameters to the measured data should be possible, and automated to the extent it is possible. It should be easy to change the fitting conditions for finding alternative parameter solutions.
5. Interaction with the software should be intuitive, and suiting usage both in production and more advanced areas.

4.1. User interface

To be able to use complex measurement equipment such as the gonioreflectometer in a production environment, a well-deployed user interface is of great importance. We propose an interface aimed at providing a fast and highly automated work-flow, from the measurement of material reflectance to exported BRDF model fitted to the gathered data. The software, for which the interface is demonstrated in Figure 2, enables automated and direct procedures for easy integration in a production pipeline at the same time as supporting a large number of user inputs, and an "advanced mode" for research oriented usage.

The process of measuring and fitting a material is depicted in Figure 3. A general calibration file stores the state of the GUI, while the camera calibration and color profile are created through a calibration procedure in a wizard. The output of a measurement is binary unstructured reflectance data, onto which a color profile can be applied and a fitting



(a) Scanning interface

(b) Visualization/fitting interface

Figure 2: *Overview of the interface, which is divided into a scanning and a visualization/fitting interface.*

started. The final outcome is optimized BRDF model parameters output to useful file formats for use in rendering.

Since the terminology used in relation to reflectance capturing and fitting not always is self-explanatory for those not familiar with the details of BRDFs, through-out the interface informative help is provided on all constituent tools by means of offering mouse triggered help on widgets.

4.2. Visualization modes

Having four dimensional data, careful thought is needed for visualizations, to show the details of a BRDF. The plotting and rendering tools available in the interface illustrates a measured materials reflectance in an informative manner from different abstraction levels, for insight into the material properties, and for comparison to parametric models in the fitting procedure. There are four different ways of visualizing the reflectance as function of incoming/outgoing angles, shown in Figure 4:

- **Polar plot, Figure 4(a)**: For a specified incident light direction, the plot shows the reflectance along a selected φ_o, or slice angle, plotted in the direction of the reflection. It illustrates a slice of the reflectance distribution in the hemisphere above the measured point, where the plotted data shows the magnitude as the distance to the origin of the plot. The direct correspondence to the reflectance distribution makes the visualization informative and easy to interpret. The restriction to one slice angle, however, makes it difficult to get an overview of the global BRDF shape, but for a local – in depth – comparison of measured data and fitted reflectance model small differences are easy to distinguish using the polar plot.
- **Cartesian plot, Figure 4(b)**: The visualization is similar to the polar plot, but here the slice of the hemisphere has been transformed to a cartesian coordinate system so that the vertical distance represents the reflectance for the different angles. While the polar plot gives an intuitive view of the reflectance distribution, the cartesian version shows an abstracted representation that is easy to read and use for comparison with BRDF models.
- **Hemispherical plot, Figure 4(c)**: The reflectance is drawn as colors for the entire hemisphere, viewed from above, for the specified incident light direction. In this way a large amount of data can be visualized simultaneously, for an easy overview of the reflectance distribution over all excitant directions. While the rendering shows a global image of the reflectance, it is difficult to see small local differences when comparing to a fitted BRDF model.
- **Geometric plot, Figure 4(d)**: Here, the hemispherical plot is rendered with GPU acceleration and interaction can take place, turning the plot for different views. Furthermore, there is an option which adds a geometric scaling according to the reflectance, creating a 3D representation of the distribution. This "extrusion" can be modified to get more or less effect, in visualization purposes, and can be seen as the 3D equivalent of the polar plot. Having the distribution rendered as a geometry provides very intuitive information on the material reflectance on a global

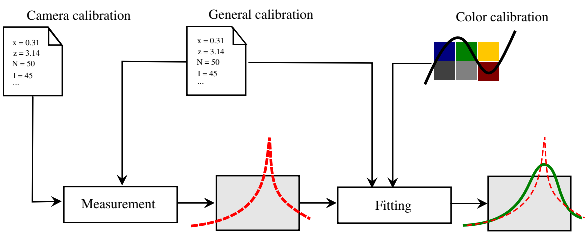
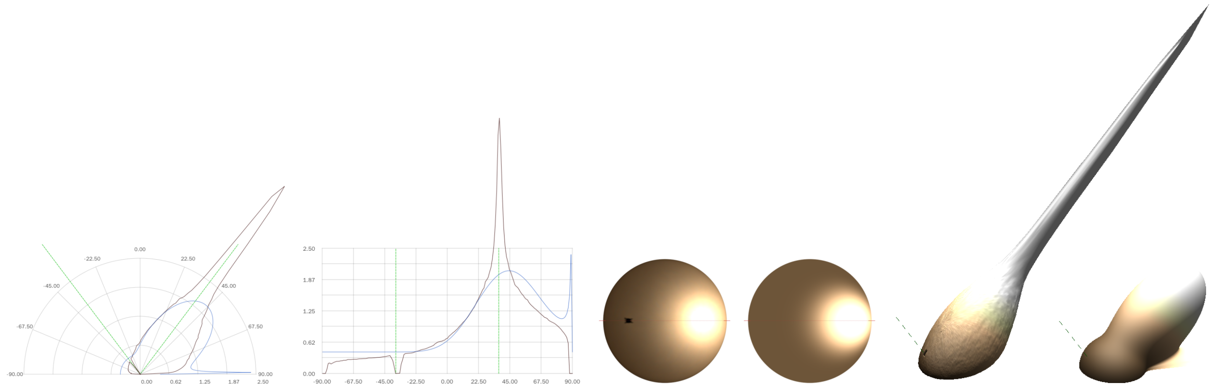


Figure 3: *Material measurement and fitting pipeline.*



(a) Polar (b) Cartesian (c) Hemispherical (d) Geometric

Figure 4: *Examples of the different visualization modes. Plotted together with the measured data is a fitted Ward BRDF.*

level, and comparing to a fitted BRDF model differences are easy to spot.

To summarize, the different rendering options are complementing each other. They all play important parts in an informative system for visualization of a measured materials reflectance properties and comparing it to a parameter fitting, on both a local and global level.

4.3. Parameter optimization

The fitting of an analytic model to the measured data is done in a non-linear least square manner using the Levenberg-Marquardt method, in a C/C++ implementation [Lou04]. During the optimization the visualizations are updated with the new parameters, and error measurements formulated as the root mean square difference are displayed, for visual feedback of the fitting progress and comparison by means of mathematical similarity.

For a fitting process, default start parameters, possibly scaled by the maximum input reflectance value, yields good results with most materials. However, since the parameter space, especially at high dimensions, can have a number of local minima, changing the parameters starting positions can give a different final outcome. To enable tuning of the parameters for a fitting, the interface provides interactive tools. The tuning is performed with sliders for diffuse and specular parts of the RGB-channels for a BRDF model, and with a slider for the general gloss, or width of the specular peak. With this classification of the tuning tools, the process becomes intuitive for an artist familiar with simple material properties.

Since the construction of the measurement stage makes samples around the retro-reflection direction occluded, there will be a set of faulty reflectance measurements. To avoid having this data affecting the fitting, the interface provides an option where the user can input an angle specifying the radius of a circular area around the retro-reflection direction where data will be ignored. Furthermore, a user specified amount of positions near gracing angles can also be specified for rejection; since polar angles near $\theta = 90^\circ$ are sensitive to calibration, and measured from a smaller material projection area on the sensor, this is useful for removing unreliable data.

One difficulty when fitting to highly specular materials is the high contrast between diffuse and specular values in the BRDF, which can make the parameter optimization overemphasize the large specular peak, resulting in lost accuracy of the diffuse parts. To overcome this problem the fitting interface provides an option for fitting to the logarithmic BRDF, decreasing the diffuse/specular contrast. To have a

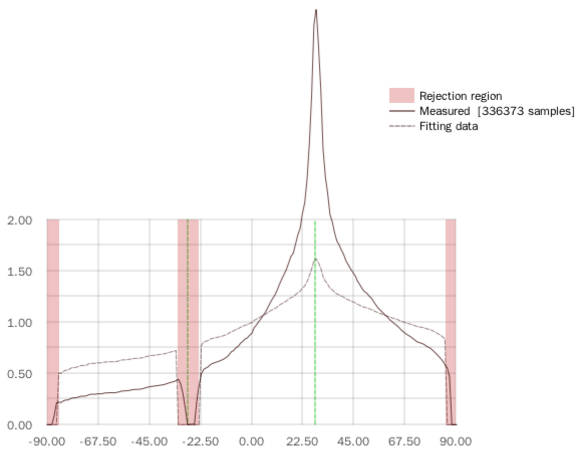


Figure 5: *Visualization of logarithmic fitting conditions, plotted together with the original data. Note that the data is completely ignored in the rejection regions, represented by red.*

stable logarithmic BRDF, avoiding the problems of near zero values, it is formulated according to Equation 1, where σ is a user parameter for controlling the contrast reduction. The impact of the contrast reduced BRDF for fitting can be seen in the screenshot in Figure 5, where $\sigma = 0.1$.

$$\rho_{log} = \log_{10}(1 + \frac{\rho}{\sigma}), \qquad \sigma > 0 \tag{1}$$

4.4. Calibration

Of great importance for the quality of a measurement process outcome is having a good calibration. Most calibrations can be performed when setting up the capturing system, but some need consideration during usage. For example can a color calibration be needed, *e.g.* when changing the light source lamp. The software offers standard color profiling – provided by the Argyll Color Management System (ACMS) [Gil08] – in an easy to use wizard interface (Figure 6), targeted for non-research production. Since the measurement rig enables light incident from all directions in the hemisphere, it would be an ideal stage for capturing a color chart and evaluating its diffuse color. However, the capturing distance puts restrictions to the size of the color chart, and a more general approach is therefore taken. The camera arm is rotated by 90°, so that the chart can be captured at any distance. This produces a color transformation that is calibrated in color, but not in scale cause of the capturing lighting conditions. The correct scaling is found by evaluating the diffuse color of an arbitrary reference patch in the measurement setup, with light incident from a set of directions representing the hemisphere. Applying the color profile, it is used to find the correct scaling.

The created color profile can be applied directly in the measurements, but a more general approach is to use it externally on to the data when performing a fitting, enabling changing of profile.

5. Discussion

Four material measurements are visualized in Figure 7, both plotted for red, green and blue color channels in cartesian coordinates, drawn as colors for the excitant hemisphere from above, and rendered in 3D. A Ward BRDF model has been fitted to the materials; it is plotted together with the measured data in the cartesian plot, and drawn next to the higher dimensional visualizations. The fittings have been performed with the default settings, and the calibration used was created in the calibration wizard. The measurements shows very dense data, for visualization; in practical situations measurements can be significantly sparser without sacrificing fitting quality.

The figure also shows the importance of having good reflectance visualizations in the fitting process. With information only from the cartesian plot, the optimization of the

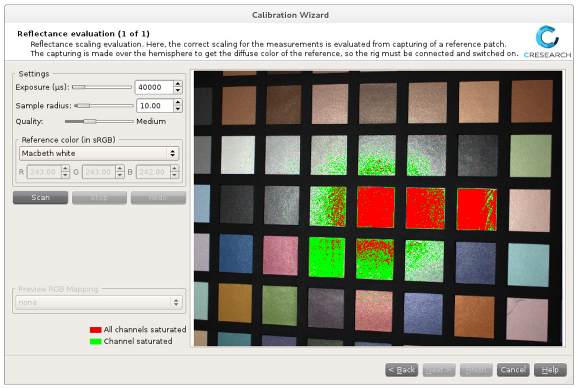


Figure 6: *Screenshot from the calibration wizard. The example shows the interface for evaluation of the reflectance scaling.*

Ward model to the silver lacquer material seem to be ill-fitting. However, the geometric rendering more clearly illustrates a distribution with a narrow specular peak and a more spread specular base, where the fitted BRDF has conformed to the base, resulting in a globally more optimized result. In the hemispherical color rendering this global optimization can also be seen, where the high specular peak is cropped to white. In conclusion – the visualizations of the distribution where the entire hemisphere of excitant reflectance is shown, can give valuable information of the material appearance behavior and the parameter optimization result that is difficult to see in a classic 2D plot.

Having a construction with a 4-axis capturing stage and a high performing camera, we can make measurements of high quality, with high precision information on reflectance directions and a complete image of information for each measurement position. Using this information to calculate the reflectance averaged over a small user specified pixel region we yield data with large noise reductions as compared to image-based approaches where large areas of the reflectance hemisphere are captured simultaneously. The improved data comes at the cost of longer measurement times, but with the system construction and software, enabling continuous movement while capturing, and the possibility to extend measurements to use more of the captured image data, the time can be significantly reduced as compared to 4-axis gonioreflectometers that need to halt at every measurement position.

The measurement time depends to a large extent on the material; a highly specular material is slower to scan compared to a diffuse since the specular peak have to be traversed several times with shortened exposure time. The time also depends on the stepper motor controller settings, *i.e.* max speed and acceleration. How fast the rig can be set to move is governed by the frame rate of the camera and the exposure time used. The utilized camera can capture at 16 frames per

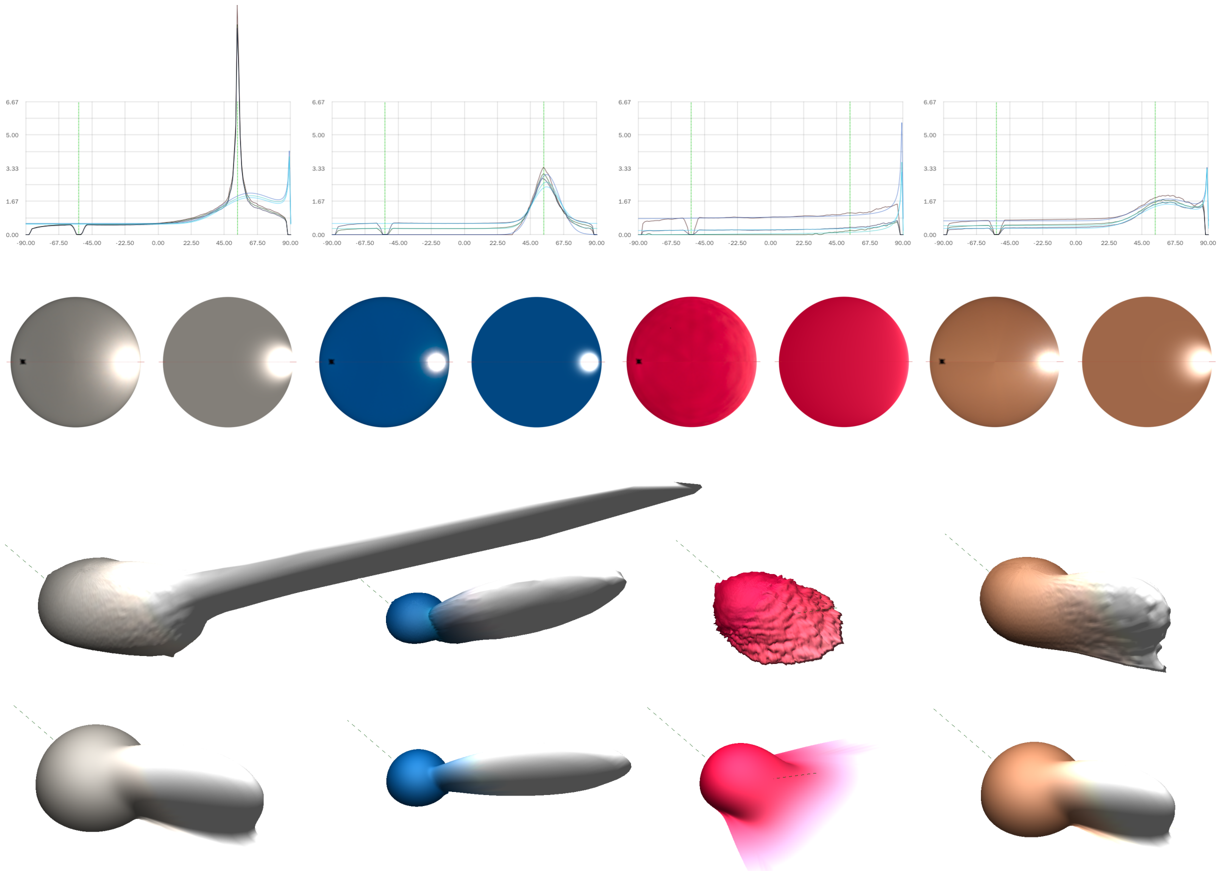


Figure 7: *Different material measurements, together with a fitted Ward BRDF. From left to right: silver lacquer, blue lacquer, red textile, wood.*

second at highest resolution. However, if only some part of the sensor need to be used, the frame rate can be increased. In the examples in Figure 7, the measurements were sampled uniformly spaced at every 1° in both φ_o and θ_o, and the stepper motors ran at a max speed of $20^\circ/s$. For one incident light direction, scanning took approximately between 40 and 50 minutes. For a good optimization, a set of incident direction need to be used, but the excitant directions could be sampled significantly sparser, and therefore the total scanning – for use in fitting – does not need to take more than 10–15 minutes for isotropic materials.

The parameter optimization time depends on parameters such as number of measured samples, shape of BRDF, number of parameters in the BRDF model, model function, CPU used etc. In most cases it is finished in a matter of seconds, or possibly minutes with a dense measurement sampling scheme.

We have only used our measurement system assuming isotropic materials, thus needing only one dimension – the polar angle – in the incident light, and with fitting to isotropic BRDF models. However, the system is fully capable of measuring anisotropic materials, adding the dimension of the incident azimuthal light angle.

6. Further work

The developed system is centered around the different interaction and visualization possibilities, for achieving a system for easy use in many application areas. To further confirm that the requirements are fulfilled, an evaluation should optimally be performed. The capturing pipeline should be tested by a number of people with different backgrounds in the area, to evaluate the measurement, visualization and fitting interfaces with respect to parameters such as usability, efficiency and quality of outcome.

The capturing device is at the moment used in its simplest form for BRDF measurements, with a single reflectance measurement for each position of the scanning rig. Further work is intended with the setup though, and since emphasis has been on extendibility a strong research platform is provided for such work.

In the current system only a fraction of the pixels in an image are used, and many extensions of the setup could be done using more of the information. With this data multiple samples can be retrieved simultaneously, either for capturing different light directions on a homogenous material, or for calculating values for a spatially varying BRDF. Another useful extension would be to estimate a non-planar geometry and use for scanning.

The current measurements are done at the spectral locations achieved with the RGB-filters of the camera. Generalizing this for larger coverage of the spectral domain could be done with for example a filter setup.

The measurements we have done are all sampled uniformly by specifying a range and a sampling density for each dimension of the BRDF $(\varphi_i, \theta_i, \varphi_o, \theta_o)$. Having a setup like ours, however, it would be interesting to investigate the possibilities of having an adaptive sampling scheme. Since the fastest changes in a materials reflectance happens close to the specular peak, around the perfect reflection direction, an adaptive scheme would measure this area more thoroughly, and differently depending on the material. Alternatively, a fixed sampling scheme where the specularity is sampled more densely than the diffuse parts, would yield measurements more effective than uniform sampling.

7. Conclusion

In this paper we have demonstrated our flexible material reflectance measurement system, centered around the interaction and visualization possibilities. We have shown how the developed software enables an abstraction of the calculations and understanding needed in measurement of reflectance and optimization of BRDF models to measured data. The user interface of the software provides a scanning and fitting environment which could be used at a highly automated level in a simple physical material modeling work-flow. We have also pointed to the importance of having different and comprehensive visualization tools, both for insight in to the reflectance properties of the material and for confirming the accuracy of an optimization process.

References

[Bli77] BLINN J. F.: Models of light reflection for computer synthesized pictures. In *Proceedings of the 4th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1977), SIGGRAPH '77, ACM, pp. 192–198. 2

[CT81] COOK R. L., TORRANCE K. E.: A reflectance model for computer graphics. In *Proceedings of the 8th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1981), SIGGRAPH '81, ACM, pp. 307–316. 2

[DM97] DEBEVEC P. E., MALIK J.: Recovering high dynamic range radiance maps from photographs. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., pp. 369–378. 3

[DR97] DAVIS K. J., RAWLINGS D. C.: Directional reflectometer for measuring optical bidirectional reflectance. Patent, 06 1997. US 5637873. 2

[DW04] DANA K. J., WANG J.: Device for convenient measurement of spatially varying bidirectional reflectance. *J. Opt. Soc. Am. A 21*, 1 (Jan 2004), 1–12. 2

[Erb80] ERB W.: Computer-controlled gonioreflectometer for the measurement of spectral reflection characteristics. *Appl. Opt. 19*, 22 (Nov 1980), 3789–3794. 2

[Gil08] GILL G.: Argyll color management system. [web page] http://www.argyllcms.com, 2008. [Accessed on 12 Oct. 2011.]. 6

[HGH06] HÜNERHOFF D., GRUSEMANN U., HÖPE A.: New robot-based gonioreflectometer for measuring spectral diffuse reflection. *Metrologia 43*, 2 (2006), S11. 2

[Kaj86] KAJIYA J. T.: The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1986), SIGGRAPH '86, ACM, pp. 143–150. 2

[Lou04] LOURAKIS M.: levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++. [web page] http://www.ics.forth.gr/~lourakis/levmar/, Jul. 2004. [Accessed on 12 Oct. 2011.]. 5

[MDL*98] MATTISON P. R., DOMBROWSKI M. S., LORENZ J. M., DAVIS K. J., MANN H. C., JOHNSON P., FOOS B.: Handheld directional reflectometer: an angular imaging device to measure brdf and hdr in real time. *Proc. SPIE 3426*, 1 (1998), 240–251. 2

[MPBM03] MATUSIK W., PFISTER H., BRAND M., MCMILLAN L.: A data-driven reflectance model. In *ACM SIGGRAPH 2003 Papers* (New York, NY, USA, 2003), SIGGRAPH '03, ACM, pp. 759–769. 2

[MWL*99] MARSCHNER S. R., WESTIN S. H., LAFORTUNE E. P., TORRANCE K. E., GREENBERG D. P.: Image-based brdf measurement including human skin. In *Rendering Techniques '99, Proceedings of the Eurographics Workshop in Granada, Spain* (1999), pp. 131–144. 2

[MWLT00] MARSCHNER S. R., WESTIN S. H., LAFORTUNE E. P. F., TORRANCE K. E.: Image-based bidirectional reflectance distribution function measurement. *Appl. Opt. 39*, 16 (Jun 2000), 2592–2600. 2

[NDM05] NGAN A., DURAND F., MATUSIK W.: Experimental analysis of brdf models. In *Proceedings of the Eurographics Symposium on Rendering* (2005), Eurographics Association, pp. 117–226. 2

[PB96] PROCTOR J. E., BARNES P. Y.: Nist high accuracy reference reflectometer-spectrophotometer. *Journal of Research of the National Institute of Standards and Technology 101* (1996), 619–627. 2

[War92] WARD G. J.: Measuring and modeling anisotropic reflection. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1992), SIGGRAPH '92, ACM, pp. 265–272. 2

[WLL*08] WEYRICH T., LAWRENCE J., LENSCH H., RUSINKIEWICZ S., ZICKLER T.: Principles of appearance acquisition and representation. In *ACM SIGGRAPH 2008 classes* (New York, NY, USA, 2008), SIGGRAPH '08, ACM, pp. 80:1–80:119. 1

[WSB*98] WHITE D. R., SAUNDERS P., BONSEY S. J., VAN DE VEN J., EDGAR H.: Reflectometer for measuring the bidirectional reflectance of rough surfaces. *Appl. Opt. 37*, 16 (Jun 1998), 3450–3454. 2

Quantification of gaseous structures with volumetric reconstruction from visual hulls

S. Seipel[1,2] and P. Jenke[1]

[1]Faculty of Engineering and Sustainable Technology, University of Gävle, Sweden
[2]Centre of Image Analysis, Uppsala University, Sweden

Abstract

3D reconstruction from visual hulls is a well established technique for camera based reconstruction of 3D objects in computer graphics. We propose in this paper to employ visual hull techniques to quantify the volume of diffusely defined gaseous structures. In our evaluation, visual quality of the 3D reconstructions is secondary. Instead, using synthetic ground truth data, we determine the number of independent silhouette images needed to achieve a stable volume estimate. We also estimate the influence of different segmentation results of the silhouette images on final volume estimates. Our results show that comparably few camera views yield to convergent volume estimates. For the type of 3D data studied, visual hull reconstructions overestimate actual volumes with about 50%. This proportion seems to be consistent for different data sets tested and may serve for re-calibration of volume estimation of gaseous structures.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Digitizing and scanning I.4.1 [Computer Graphics]: Digitization and Image Capture—Imaging geometry I.4.9 [Computer Graphics]: Applications—I.4.10 [Computer Graphics]: Image Representation—

1. Introduction

In context of environmental analysis, methane (CH_4) is one of the most critical greenhouse gases, as it remains in the atmosphere for approximately twelve years. According to Wuebbles [Wue02] methane is more than 20 times more effective in trapping heat in the atmosphere than carbon dioxide (CO_2). Methane gas is released from a variety of natural and man-made sources. Emissions of methane from open dumping and improper land filling of municipal solid waste are estimated to contribute to 3 - 19% of the anthropogenic emissions word-wide [Tal07]. Identification, quantification, and reduction of CH_4 emissions from landfills can make a significant contribution to decrease the greenhouse gas stock. Among the largely available and low cost methods to detect methane gas are pellistor based sensors. They are used for fixed point measurements of gas concentrations at specific positions. Recently, infrared (IR) based measuring principles have been used to develop new sensors for fixed point measurements. They exploit the fact that many gases absorb energy within a narrowly defined band in the IR spectrum. While traditional fixed point measurements can reveal concentrations of gases at specific locations, they are of limited value for quantifying the actual volume and its spatial distribution. Very recent research aims at using novel IR imaging devices to capture gas distributions [SM10]. In previous work segmentation methods based on optical flow analysis have been proposed to identify the presence of gas in the camera image [hS11].
The objective of the work presented here is to quantify and describe the 3D spatial distribution of gaseous structures using IR camera images from multiple views. In particular we aim to determine the number of camera views needed in order to obtain volume estimates of sufficient precision. Another objective is to evaluate the potential effect of different segmentation results of the 2D silhouette images on the final 3D volume estimate.

2. Related work

Reconstruction of 3D geometric objects is the purpose of several different multi-view geometry algorithms in computer vision. One class of these algorithms identifies feature points in different views [HZ03]. Using a RANSAC procedure, a robust estimate of the fundamental matrix is obtained by applying the eight-point algorithm based on best estimates for corresponding points [Har97]. The fundamental matrix describes the geometric relationships between two camera views and it is subsequently used to triangulate the positions in 3D space for sets of homologous feature points [LF95]. One advantage of these image based algorithms is their capability to describe surfaces of objects through dense point clouds, which form the basis of highly detailed polygonal surfaces.

Images of methane gas (and other gases) generated by IR imaging devices are, however, not well suited for this type of multi-view 3D reconstruction approach. As is shown in Figure 1, gas becomes even in the infrared spectral range apparent only as a transparent shadow, since gas only absorbs (attenuates) the IR radiation from the background. Gas clouds do therefore not expose any clearly structured surface towards the observer. Instead, they can be identified only as diffuse areas with rather fuzzy boundaries.

For this type of boundary based 2D object representations, visual hull algorithms seem to be a more feasible approach for 3D reconstruction. Reconstruction of the visual hull of an object assumes that an object is captured with many different camera views of known positions and directions [Lau94, LW10]. After determining the silhouette of the object in the camera images, silhouette edges are projected back into 3D space forming a set of general cones. The intersection of all cones represents the visual hull of the object, and it is an estimate for the geometry of the reconstructed object. Laurentini has shown that for non-concave geometries an object can be reconstructed faithfully with $O(n^5)$ silhouettes where the number of faces of this object is n [Lau97]. It is obvious that the accuracy of the reconstructed geometry increases as the number of camera views is increased. Nevertheless, concavities in an object cannot generally be reconstructed accurately [Lau95]. Most algorithms published for reconstruction of the visual hull [Sze93, LMS03, LBN08] use discrete silhouette representations from real or synthetic camera images. Efficient memory techniques have to be employed to maintain adequate resolutions for the reconstructed volume. Szeliski proposed an octree for backprojection [Sze93]. Recent implementations use 3D texture mapping [LMS03, LMS04, LBN08] and GPU accelerated algorithms [LMS04, KSK*08, LBN08] for the back-projection into 3D space. These algorithms have generally limited precision depending on texture sizes and render buffer limits.

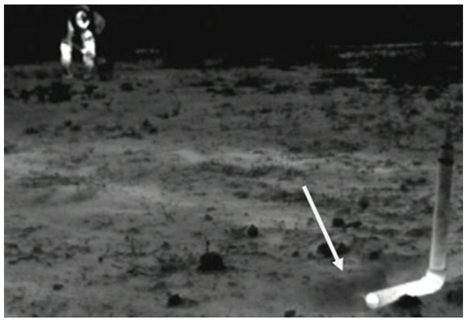


Figure 1: *Infra-red (IR) image showing methane gas leaking from a deduction pipe.*

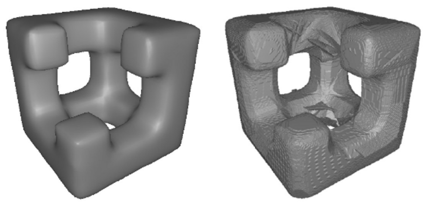


Figure 2: *An example of a polygonal model (left) and its volumetric reconstruction using visual hull methods (right).*

3. Method

3.1. Evaluation approach

Figure 2 shows an example of an original polygonal object rendered with simple Phong shading (left) and its visual hull retrieved from a set of silhouette images (right). The reconstructed object is initially a discrete voxel model, which can be transformed into an explicit polygon representation using iso-surface reconstruction. Alternatively, as is the case here, it can be rendered directly using a volume ray-caster. The picture of the visual hull in Figure 2 clearly shows that visual artifacts are inevitable as fine detail is not preserved in the reconstruction process. For most visual applications, the reconstructed objects are again texture mapped with the original camera images which makes artifacts less salient. In our specific application we aim at using visual hulls to estimate the volume of gas and smoke clouds. For this purpose, we have some different requirements and limitations:

- As the visual hull is primarily intended for volume quantification, visual artifacts are not critical.
- Monitoring of gas leakage will require expensive IR imaging, which calls for a low number of different camera positions.
- Detection of gas clouds in IR imagery is a very delicate image processing task, which at present state implies segmentation results with some uncertainty.

In our experiments we want to find out how many camera images are sufficient to reconstruct gas clouds with an acceptable quantification error. Another issue is to identify, to what extend different segmentation results in the camera images affect the quantification of the reconstructed volume. Figure 3 illustrates the different steps in our evaluation approach; they are described in the following subsections.

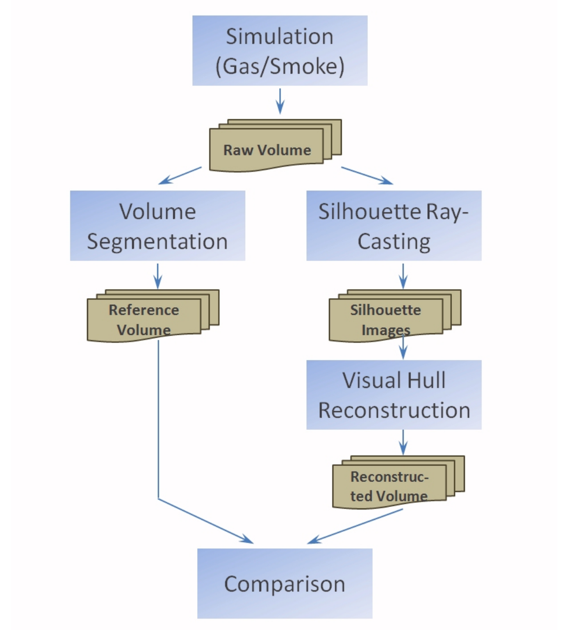


Figure 3: *Flowchart illustrating the evaluation approach.*

3.2. Data acquisition by simulation

For evaluation of the volumetric accuracy of visual hull reconstructions, a known ground truth data set is needed. Smoke and non-visible gases are, as stated before, very diffusely defined and no practical means exist to capture a reliable reference volume of gas. We therefore decided to use a synthetic dataset as reference standard. Various methods exist in computer graphics to simulate fluids and smoke, among others wavelet-based approaches for volumetric simulation [KTJG08]. For our experiment we used the reference implementation by Kim [KT] and simulated volumetric smoke on a 512^3 voxel grid. From each simulation sequence we captured 300 snapshots (volumes) and we sporadically chose a few volumes out of these sequences as reference volumes. We parametrized the smoke/gas simulation so as to accomplish plausible appearances of smoke plums.

3.3. Volume segmentation

The simulation of gas and smoke results in raw data that represents smoke through density values in the range between 0.0 and 1.0. As in a real scenario, the appearance of a gas cloud is not well defined by clear borders or contours in the density field. The identification of gas will rather depend on the employed segmentation methods and parameters; this will imply some uncertainty in regard to the actual shape of smoke. We mimic this uncertainty in the segmentation by applying different thresholds to the raw data, in order to retrieve *reference volumes* of varying conformance with the actual gas shape. The segmented reference volumes serve as ground truth for the volumes to be estimated using the visual hull reconstruction.

3.4. Silhouette image rendering

The reconstruction of the 3D geometry of the reference volume requires a set of silhouette images of the same raw dataset. We implemented a ray-caster with a simple ray evaluation function that evaluates to one as soon as a voxel above the threshold along the ray has been encountered. The resulting images are binary silhouettes of the raw volume from an arbitrary number of randomly chosen camera positions on a sphere around the raw data volume.

3.5. Visual hull reconstruction

The reconstruction of the volume using the visual hull approach is accomplished following an algorithm similar to the one published in [KSK*08]. The algorithm operates in a slice-order; that is, the volume is reconstructed by rendering subsequent slices of the volume using screen-filling quads. For each slice, every binary silhouette image is rasterized into the frame-buffer. Hereby the corresponding camera parameters are used to configure the texture projection matrix. The stencil-buffer is used to increment buffer elements where the silhouette image is not empty. The overlapping region in each slice is hence determined by all buffer elements that are equal to the number of camera views. The resulting frame-buffer for each slice is copied back into the volume memory. To reconstruct a volume of size 512^3, 512 render passes are required, each rasterizing n silhouette images into the stencil buffer. The volume resulting from this stage is in the following referred to as the *reconstructed volume*.

3.6. Comparison

The final step of our evaluation is a simple volumetric comparison, whereby classified voxels in the *reference volume* are enumerated and compared with the number of voxels in the *reconstructed volume*. Here, we take the number of voxel as a discrete measure for the volume of smoke in each volume, which allows for a fairly basic overall quantitative comparison. In order to get a more qualitative picture of the differences between *reference volume* and *reconstructed volume*, a voxel wise comparison can be achieved by calculating the subtraction volume.
Variables to be controlled in this evaluation are the number of silhouette images for visual hull reconstruction, and the chosen segmentation threshold.

4. Results

This section presents some results of our initial simulation based evaluation. Figure 4 shows in the upper row an example of one gas volume rendered with surface shading. The

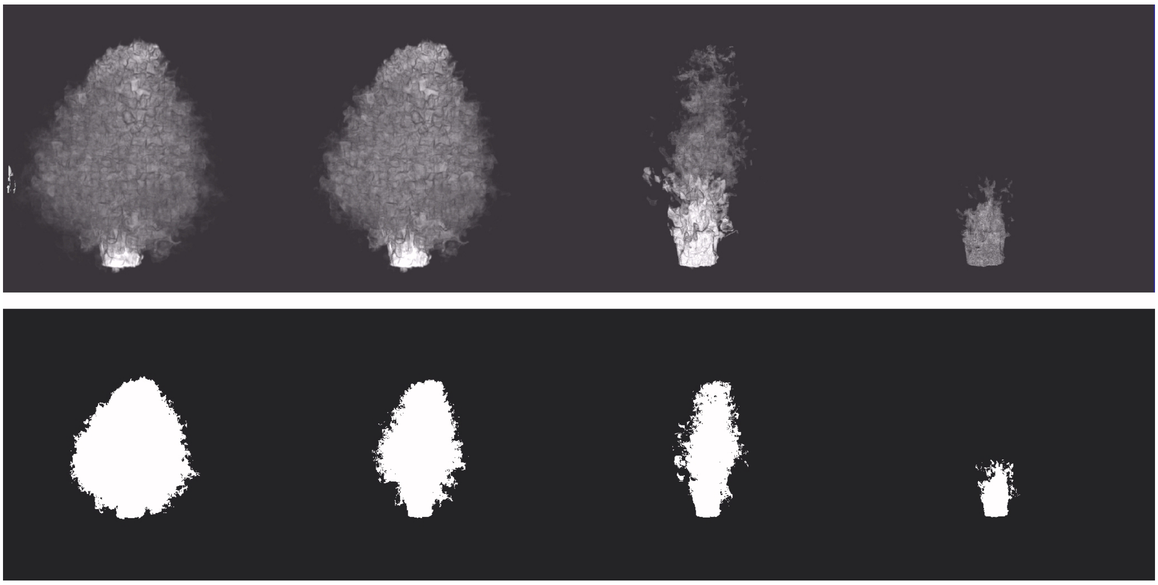


Figure 4: *Results of the simulated smoke dataset. The upper row shows surface renderings of the smoke volume for different volume segmentation thresholds. The bottom row shows silhuette images rendered with a ray-evaluation function that binarizes voxels with corresponding thresholds. Thresholds from left to right are: 0,01; 0,1; 0,2; and 0,4.*

four different pictures show the same raw data volume segmented with four different threshold values. Apparently the size and structure of the gas cloud change dramatically as the threshold is increased. In the bottom row, the same raw data volume has been rendered as a silhouette image using the corresponding threshold values and from one camera position. Note that the camera position for silhouette rendering is not exactly the same as the one used to render the pictures in the upper row. The pictures of the silhouettes illustrate that a sharp definition of the borders of a smoke/gas cloud is problematic, as smoke is very diffusely defined without sharp borders. Over- or undersegmented silhouette images are therefore likely to occur and they hamper reliable volume quantification.

Figure 5, upper row, shows surface rendered images of the reconstructed volumes. In this figure, reconstructions were accomplished with increasing numbers (3, 7, 14, and 35 images) of silhouette images and for the reference volume with a fixed threshold of 0,01 (same as the leftmost volume in Figure 4). The image series demonstrates that the overall volume for a simple convex object is reconstructed fairly well with only a few reconstruction views. Results are nevertheless not acceptable from a visual quality point of view, even in the reconstruction with 35 silhouettes. That is, visual appearance of the reconstruction still lacks visual detail that is present in the ground truth data volume. The bottom row of Figure 5 shows a semitransparent rendering of the subtraction volume. It illustrates where and how the *reference volume* and the *reconstructed volume* differ. Yellow areas indicate voxels that have been reconstructed without actually being part of the original volume. Blue voxels (on top) indicate small structures that have been lost in the reconstruction process. At large, the reconstruction is conservative and yields to volumes, which are bigger than the original volume.

A quantitative comparison of the reconstructed volume with the original volume is illustrated in Figure 6. It shows the ratio R/O of the volume sizes of the reconstructed volume with the original (reference) volume. It is clearly visible that this ratio decreases for an increasing number of silhouettes used for reconstruction, which suggests that the reconstruction of the volume becomes increasingly refined for more reconstruction views and hence better approximates the original object. Figure 6 contains observations for the same dataset and ten different segmentation thresholds.

5. Conclusion and Discussion

In this work we have investigated, if visual hull reconstructions can be used to quantify gas volumes recorded with IR imaging devices. Based on the result of our experiments we can so far conclude the following: For the purpose of volume quantification, visual hulls appear as a promising estimation technique, even if only comparably few camera views are used for reconstruction. Our initial results show that reconstructed volume sizes converge comparably soon to some steady level. With approximately 16-20 camera views the estimated volumes reached a fairly constant size, which was not significantly improved even with more camera views. It seems as if the inclusion of even more camera views is im-

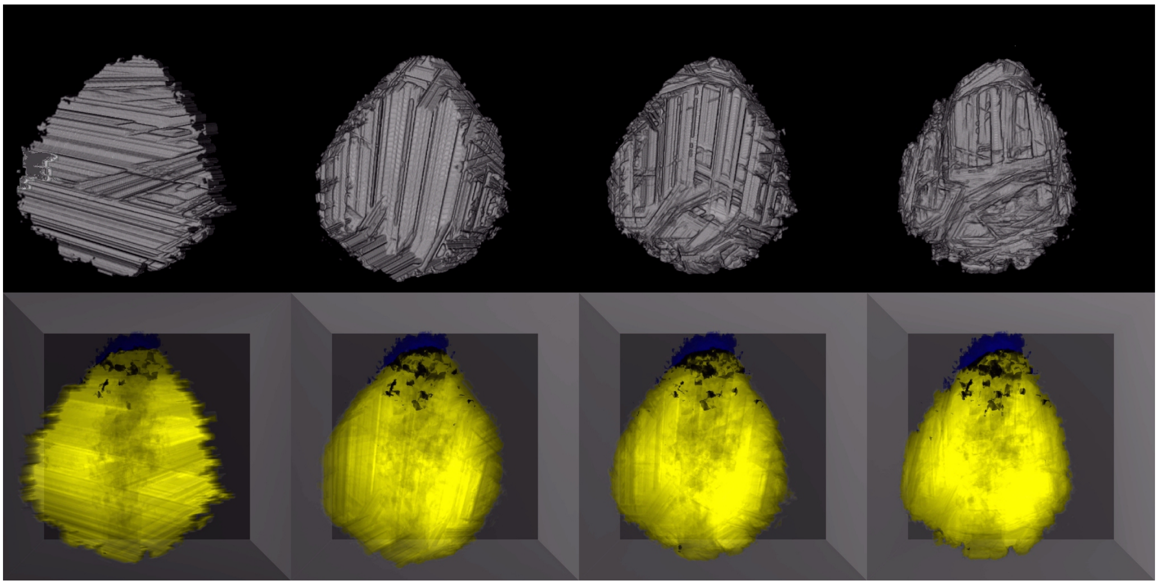


Figure 5: *The upper row shows surface renderings of the the visual hull reconstructions for increasing numbers of silhuette images (from left to right: 3 images, 7 images, 14 images, and 35 images). The bottom row is a visualization of the subtraction volumes for increasing numbers of silhuette images. In all cases, reconstructions were applied for a volume segmented with a threshold of 0,01.*

portant to reveal visual detail; for quantification of the volume, however, these details are not significant. For the type of data volumes tested in our experiment, estimated volume sizes reached a steady value of about 150% of the original volume size for many of the datasets we tested. This value might therefore be instrumental for calibration of the volume estimation procedure in this type of application.

Another interesting finding is that estimation of volume sizes follows the same trend regardless of the chosen segmentation threshold. However, absolute levels do vary significantly depending on the segmentation results. While for most segmentation thresholds (between 0,01 and 0,1) volume estimates are confined within narrow ranges, for higher and lower thresholds the volume estimates can differ by approximately 100% throughout. A comparison with Figure 4 may give one explanation for this. Depending on the segmentation criterion (here threshold) over- or undersegmentation can yield structurally very different results, which are partly reflected by hugely varying volume estimates in the reconstruction. For a final practical application this implies that reliable segmentation of the gaseous structures in the original IR image material is as least as important as maintaining a sufficient number of different camera views.

6. Acknowledgments

The authors wish to acknowledge Linus Fredriksson for provision of the images in Figure 2, and Julia Åhlén for the picture in Figure 1. We also wish to acknowledge the DaGIS project at Future PositionX, Gävle, through which this research has partly been funded.

References

[Har97] HARTLEY R.: In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*, 6 (1997), 580–593. 2

[hS11] ÅHLÉN J., SEIPEL S.: Indication of methane gas in ir-imagery. In *Proceedings of IADIS International Conference Computer Graphics, Visualization, Computer Vision and Image Processing 2011 (CGVCVIP 2011)* (2011), pp. 187–192. 1

[HZ03] HARTLEY R., ZISSERMAN A.: *Multiple View Geometry in Computer Vision, 2nd ed.*, 2nd ed. ed. Cambridge University Press, 2003. 2

[KSK*08] KIM H., SAKAMOTO R., KITHARA I., TORIYAMA T., KOGURE K.: Compensated visual hull with gpu-based optimization. *Advances in Multimedia Information Processing Ű PCM 2008 5353* (2008), 573–582. 2, 3

[KT] KIM T., THÜREY N.: Wavelet turbulence source code. http://www.cs.cornell.edu/ tedkim/WTURB/source.html, visited 10/16/2011. 3

[KTJG08] KIM T., THÜREY N., JAMES D., GROSS M.: Wavelet turbulence for fluid simulation. *ACM Trans. Graph. 27* (August 2008), 50:1–50:6. 3

[Lau94] LAURENTINI A.: The visual hull concept for silhouette-based image understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence 16*, 2 (1994), 150–162. 2

[Lau95] LAURENTINI A.: How far 3d shapes can be understood from 2d silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence 17*, 2 (1995), 188–195. 2

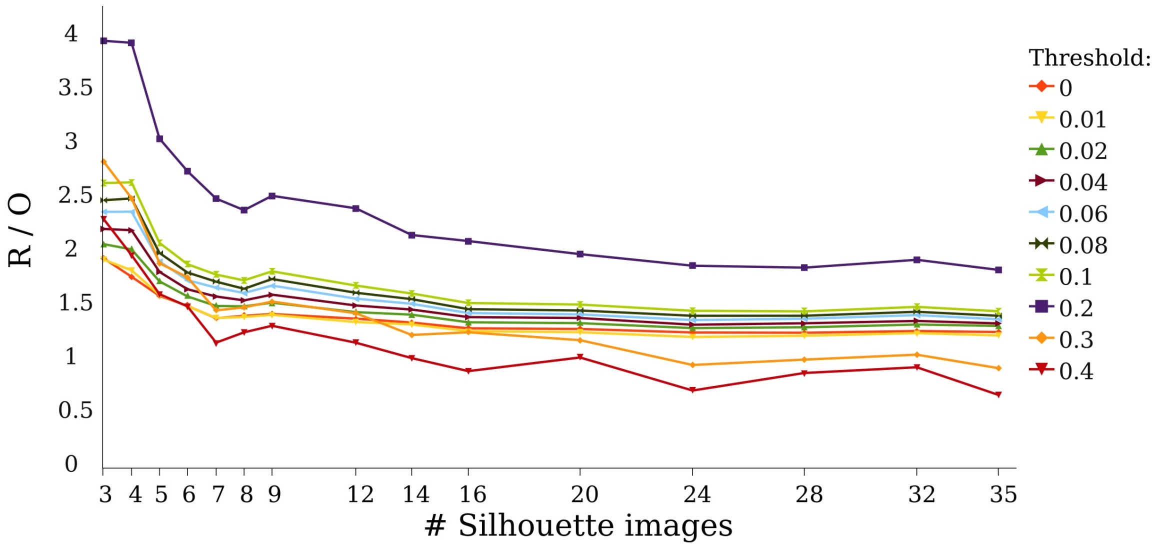


Figure 6: *Diagram showing the ratio of the volume sizes R/O of the reconstructed volume and the original (reference) volume for increasing numbers of silhouette images. Data series are plotted for the same raw data volume but with varying segmentation thresholds.*

[Lau97] LAURENTINI A.: How many 2d silhouettes does it take to reconstruct a 3d object? *Computer Vision and Image Understanding 67*, 2 (1997), 81–87. 2

[LBN08] LADIKOS A., BENHIMANE S., NAVAB N.: Efficient visual hull computation for real-time 3d reconstruction using cuda. In *Proceedings of Computer Vision and Pattern Recognition Workshops - CVPRW '08* (2008), pp. 1–8. 2

[LF95] LUONG Q.-T., FAUGERAS O.: The fundamental matrix: theory, algorithms, and stability analysis. *International Journal of Computer Vision 17* (1995), 43–75. 2

[LMS03] LI M., MAGNOR M., SEIDEL H. P.: Improved hardware-accelerated visual hull rendering. *Vision, Modeling, and Visualization*, Nov. 2003 (2003), 151–158. 2

[LMS04] LI M., MAGNOR M., SEIDEL H.: A hybrid hardware-accelerated algorithm for high quality rendering of visual hulls. In *Proceedings of Graphics Interface 2004* (2004), pp. 41–48. 2

[LW10] LIANG C., WONG K. K.: 3d reconstruction using silhouettes from unordered viewpoints. *Image and Vision Computing 28* (2010), 579–589. 2

[SM10] SAFITRI A., MANNAN M. S.: Methane gas visualization using infrared imaging system and evaluation of temperature dependence of methane gas emissivity. *Ind. Eng. Chem. Res. 49*, 8 (2010), 3926–3935. 1

[Sze93] SZELISKI R.: Rapid octree construction from image sequences. *CVGIP-Image Understanding 58*, 1 (1993), 23–32. 2

[Tal07] TALYAN V.: Quantification of methane emission from municipal solid waste disposal in delhi. *Resources, Conservation and Recycling 50*, 3 (2007), 240–259. 1

[Wue02] WUEBBLES D.: Atmospheric methane and global change. *Earth-Science Reviews 57* (2002), 177–210. 1

Considerations toward a Dynamic Mesh Data Structure

S. Pena Serna[1], A. Stork[1,2] and D. W. Fellner[1,2]

[1]Fraunhofer IGD, Germany
[2]TU Darmstadt, Germany

Abstract

The use of 3D shapes in different domains such as in engineering, entertainment, cultural heritage or medicine, is essential for representing 3D physical reality. Regardless of whether the 3D shapes are representing physically or digitally born objects, meshes are a versatile and common representation for the 3D reality. Nonetheless, the mesh generation process does not always produce qualitative results, thus incomplete, non-orientable or non-manifold meshes frequently are the input for the domain application. The domain application itself also demands special requirements, e.g. an engineering simulation requires a volumetric mesh either tetrahedral or hexahedral, while a cultural heritage color enhancement uses a triangular or quadrangular mesh, or in both cases even hybrid meshes. Moreover, the processes applied on the meshes (e.g. modeling, simulation, visualization) need to support some operations, such as querying neighboring information or enabling dynamic changes of geometry and topology. These operations need to be robust, hence the neighboring information can be consistently updated, during the dynamic changes. Dealing with this mesh diversity usually requires dedicated data structures for performing in the given domain application. This paper compiles the considerations toward designing a data structure for dynamic meshes in a generic and robust manner, despite the type and the quality of the input mesh. These aspects enable a flexible representation of 3D shapes toward general purpose geometry processing for dynamic meshes in 2D and 3D.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations

1. Introduction

The use of 3D shapes in different domains such as in engineering, entertainment, cultural heritage or medicine, is essential for representing 3D physical reality. This is true for several 3D shape representation schemes such as point clouds, isosurfaces, subdivision surfaces, meshes, parametric surfaces and B-reps, among others. These representation schemes have a tight inherited relationship with the production technique. In the case of physically born objects, for example a laser scanner can typically generate a point cloud suitable for triangulation, while a CT scanner can usually generate volume data suitable for isosurfacing. On the other hand and referring to digitally born objects, CAD systems ([Lee99]) classically model with B-reps or CSG schemes and free-form 3D modeling systems normally work with parametric, polygonal or subdivision surfaces. Although the representation schemes for digitally born objects can easily be transformed into the representation schemes of the physically born objects, the opposite case is not a straightforward process, e.g. producing a B-rep with its corresponding geometric and topological description from a polygonal mesh is a complex procedure. Furthermore, the application context also influences the utilization of specific shape representation schemes, for instance in engineering B-reps, parametric surfaces and meshes are commonly used; polygonal, subdivision or parametric surfaces are frequently employed in the entertainment industry; in the cultural heritage field subdivision or polygonal surfaces and meshes are widely established; while B-reps, parametric surfaces and meshes are extensively exploited in the medicine sector.

Meshes are commonly used in the above mentioned domains and within these domains; meshes commonly support visualization, analysis, animation, modeling and simulation processes or other dedicated domain applications. The performance of these domain applications greatly profits from the neighboring information of the mesh. For instance, a curvature analysis computes the vertex curvature for the 1-ring neighborhood ([BKP*10]) and a non-diagonal entry in a linear system of a simulation problem is computed by means of the contribution of the elements around an edge ([PSSSM09]). Nonetheless, the mesh generation process ([FG00]) does not always produce qualitative results, thus incomplete, non-orientable or non-manifold meshes frequently are the input for the domain application. Representing such meshes require generic and robust methods, in order to reliably compute the neighboring information. Additionally, the representation of dynamic meshes demands efficient methods for consistently updating the corresponding changes in the topology of the mesh.

There is a trend in the computer graphics community and especially in the geometry processing research field, focusing on the analysis and processing of dynamic meshes. This trend is enabling the discovering and the development of sophisticated methods and algorithms, leading to a more realistic reproduction of the reality, by means of either faithful animations or novel procedures for analyzing the behavior of the reality. Several methods and algorithms have been developed, in order to deal with dynamic meshes. Nonetheless, the majority of these approaches only consider the geometry of the mesh as dynamic, while the topology of the mesh remains unchanged. This consideration is justified by the complexity and the performance overhead that operations with dynamic topology, such as meshing, remeshing and multiresolution generate. Nonetheless, the implementation of algorithms with only dynamic geometry causes several artifacts and limitations, which need to be compensated, in order to make them reliable. On the other hand, an integral processing of meshes, where geometry and topology can dynamically be handled, can unlock the potential of current algorithms and it will also enable the combination of many operations, which today are considered as divergent.

We present in this paper the considerations toward designing a data structure for dynamic meshes in a generic and robust manner, despite the type and the quality of the input mesh. The methodology is generic enough to deal with triangular and quadrilateral meshes or tetrahedral and hexahedral meshes, or even hybrid surface or volumetric meshes. The robustness of the methods enables the representation of incomplete, non-orientable, non-manifold meshes or meshes with a high genus, thus the neighboring information is always consistently computed regardless of the quality of the input mesh. We also propose recommendations on how to identify geometric degenerecies or topological inconsistencies. Additionally, we present a possible implementation of the methods for computing and querying the neighboring information. We believe that these methods allow for a flexible representation of 3D shapes toward general purpose geometry processing for dynamic meshes in 2D and 3D.

2. Related Work

There is a substantial work in the scientific community in developing mesh data structures, most of them dedicated to specific domain application. We classify the proposed data structures in data structures for static meshes and data structures for dynamic meshes, regardless the dimensionality of the mesh.

2.1. Data Structures for Static Meshes

One of the pioneers in this area is Baumgart [Bau72], [Bau75], who introduced the winged edge data structure. In a later work, Muuss and Butler [MB91] described the advantages of the radial-edge data structure for representing non-manifold geometry and the corresponding Boolean operations for modeling. Campagna et al. [CKS98] proposed a data structure for triangular meshes called directed edges. Kettner [Ket98], [Ket99] presented a design framework for combinatorial edge-based data structures of polyhedral surfaces and planar maps, where the half-edge data structured was chosen and implemented (in CGAL). Levy et al. [LCCC01] proposed the circular incident edge list (CIEL) data structure for generating iso-surfaces in an unstructured grid, based on the notion of oriented half-edge. Botsch et al. [BSBK02] developed OpenMesh, an implementation of the half-edge data structure for static polygonal meshes. Kallmann and Thalmann [KT02] implemented a data structure for representing planar meshes in a compact manner, by means of vertex adjacency. Allegre et al. [ABGA04] proposed a hybrid shape representation, by means of combining a skeletal implicit surfaces and a polygonal mesh, which are assembled with an extended CSG tree. Alumbaugh and Jiao [AJ05] developed a compact array-based representation for surface and volume meshes, as a generalization of the half-edge and half-face data structures respectively. Blandford et al. [BBCK03], [BBCK05] proposed a compact representation for simplicial meshes, based on storing the link for a set of $(d-2)$ simplices.

Weiler et al. [WMKE04] presented a tetrahedral strips encoded in texture maps for rendering purposes, by means of ray casting algorithms. Cignoni et al. [CGG*04] developed a technique for out-of-core construction and view-dependent visualization of large surface models, based on a tetrahedral spatial partition of the model. De Floriani and Hui [DFH03] presented a data structure for non-manifold three-dimensional simplicial complexes also able to represent one- and two-dimensional top simplexes (wire-edge and dangling faces respectively). De Floriani et al. [DFGH04] proposed a data structure for d-dimensional non-manifold simplicial complexes, based on encoding boundary and co-boundary relations within an incident graph. De Floriani and Hui [DFH05] compared different data structures for simplicial complexes, concluding that the most compact data structures are adjacency-based, while almost all support optimal retrieval of topological information. De Floriani [DFKP05] created a survey on data structures for Level-of-Detail models of freeform geometry, including point-based, triangle-based and tetrahedron-based data structures for regular and irregular meshes. Lage et al. [LLLV05] developed a compact half-face data structure for manifold tetrahedral meshes, which is able to cope with scalability issues according to the application. Hui et al. [HVDF06] proposed a decomposition approach for representing non-manifold simplicial complex 3D shapes as a collection of tetrahedra, dangling triangles and wire edges.

Sander et al. [SNCH08] presented a scheme for efficient traversal of mesh edges for rendering purposes, using adjacency primitives and ordering these primitives for vertex cache locality. Gurung and Rossignac [GR09], [GR10] developed a compact representation for tetrahedral meshes, based on the sorted opposite table concept, which requires 4 references and 9 bits per tetrahedron. They provide a set of wedge-based operators for querying and traversing the mesh. Sieger and Botsch [SB11] presented the design decisions for a polygonal mesh data structure, aiming to improve usability, performance and memory consumption. Their work is based on an array-based halfedge data structure. Gurung et al. [GLLR11a] introduced a compact triangle mesh representation, which only needs about 2 integer reference per triangle, by means of combining the use of a quad mesh to represent the connectivity of the triangle mesh and a special sorting algorithm (SOT), nonetheless they need to rebuild the S table, when the connectivity changes. Gurung et al. [GLLR11b] developed the LR (Laced Ring) data structure for representing the connectivity of manifold triangle meshes. Its supports constant-time adjacency queries and it is suited for meshes with fixed connectivity, as any changes to the connectivity require a rebuilt of the data structure. This is a characteristic, which is shared by the data structures for static meshes. These data structure requires a precomputation step, to encode the neighboring information and any change in the topology of the mesh requires a complete re-computation of the information.

Furthermore, the capabilities of GPU computing are enabling new opportunities for visualization, but most important for computing time consuming applications. In this context, Lefohn et al. [LSK*06] developed a template library (Glift), abstracting the random access to GPU data structures such as stack, quadtree, or octree. However, they did not investigate the implementation and abstraction of mesh connectivity structures. Lefebvre and Hoppe [LH06] proposed GPU-based hashing structure for efficient random access of sparse data. In order to handle 3D domains, they proposed a hashing function for mapping the 3D domain to a 2D texture, e.g. by storing 2D pointers of the connectivity of 3D-adjacent voxel elements, generated by the intersection of the voxels with the surface mesh. DeCoro and Tatarchuk [DT07] introduced the probabilistic octree data structure, similar to adaptive octree structures, for vertex clustering in the context of a mesh simplification algorithm. Hu et al. [HSH09] created a data structure on the GPU for progressive surface meshes, by building a static buffer for the vertices and a dynamic buffer for the topology, although to maintain the dynamic buffer, they also required additional static and dynamic data structures. GPU-based data structure are very promising, however many of the current implementations deal with pre-computed dynamic data for efficient visualization purposes. Dynamic changes (e.g. on the topology of the mesh) are still very expensive.

2.2. Data Structures for Dynamic Meshes

The small number of data structures dealing with dynamic meshes, allow for a limit set of topological operation on the mesh. Weiler [Wei85] evaluated 4 different edge-based data structures for solid modeling operations (e.g. Euler operations). Chen and Akleman [CA03] developed a set of topological validity algorithms for different 2-manifold data structures in the context of mesh modeling. Danovaro and De Floriani [DDF02] proposed a compact data structure for a half-edge multi-resolution of tetrahedral meshes, built through full-edge collapses. De Floriani and Hui [DFH04] presented edge collapse and vertex split update operations for non-manifold simplicial objects and their corresponding encoding for a non-manifold indexed data structure. De Floriani et al. [DFMPS02], [DFMPS04] developed a data structure and the algorithms for dealing with non-manifold multi-resolution simplicial meshes. Danovaro et al. [DDFM*05] proposed the half-edge tree data structure for compactly encoding Level-of-Detail tetrahedral meshes, built through the application of half-edge collapses. Tobler and Maierhofer [TM06] developed a surface mesh data structure for rendering and subdivision operations, by means of representing the rendering information as indexed face set and separately representing the topological information for rendering and subdivision. Attene et al. [AGFF09] proposed a set of algorithms for converting a non-manifold tetrahedral mesh to a combinatorial 3-manifold or a PL 3-manifold by means of local modifications on the mesh. The proposed data structures for dynamic meshes are mainly dealing with multi-resolution problems (e.g. Level-of-Detail), however there are limitations regarding either the type of the mesh or the quality of the input mesh. Thus, we present the considerations toward designing a data structure for dynamic meshes in a generic and robust manner.

3. Memory and Performance

Data structures for static meshes usually aim to compactly encode the topology (and neighboring information) of the mesh, exploiting sequential dependencies and enabling a minimal set of queries for maximizing the performance of the dedicated domain application. Any other query, which is not foreseen in the original design, is normally very expensive. Thus, these data structures achieve minimal memory consumption and maximal performance for the given application. On the other hand, data structures for dynamic meshes cannot encode the topology (and neighboring information) with sequential dependency, since it will require a re-build of the encoding if modifications on the topology are performed. Hence, the memory consumption cannot always be optimized. This implies that the data structure needs a mechanism to rapidly update the neighboring information and additionally to increase or decrease the number of components without expensive memory handling (0D - vertices: V, 1D - edges: E, 2D - faces: F, 3D - cells: C). Memory buffers are a solution for avoiding the costly memory handling for removal or addition of components, nevertheless in order to be efficient, the accurate estimation of the number of components is crucial. In this context, the Euler Formula gives a very good estimation for surface meshes with genus 0:

$$F + V - E = 2.$$

Thus, for a given number of faces (F) and vertices (V), the number of edges is:

$$E = F + V - 2.$$

A similar formula holds for volume meshes with genus 0:

$$V + F - E = C + 1.$$

Table 1 presents the behavior of the formula for the tetrahedral (Figure 1) and hexahedral meshes (Figure 2). Hence, for a given number of cells (C) and vertices (V), the number of edges can by estimated by

$$E \approx (2 \times V) + C,$$

and in a similar way, the number of faces can be estimated by

$$F \approx V + (2 \times C).$$

This estimation is much more accurate than the typical estimation for tetrahedral meshes ($E \approx 6 \times C$ and $F \approx 4 \times C$) and for hexahedral meshes ($E \approx 12 \times C$ and $F \approx 6 \times C$). If the meshes (surface or volume) contain genera, shells and loops, an approximation can be made, based on the previous formulas.

Table 1: *Topological components of two tetrahedral (rows 1. Sphere and 2. Gargoyle) and two hexahedral (rows 3. Hex2 and 4. Rubber) volume meshes with genus 0.*

V	E	F	C	$V+F-E$
13	42	50	20	21
221039	1152799	1723886	792125	792126
12	20	11	2	3
9367	24354	20812	5824	5825

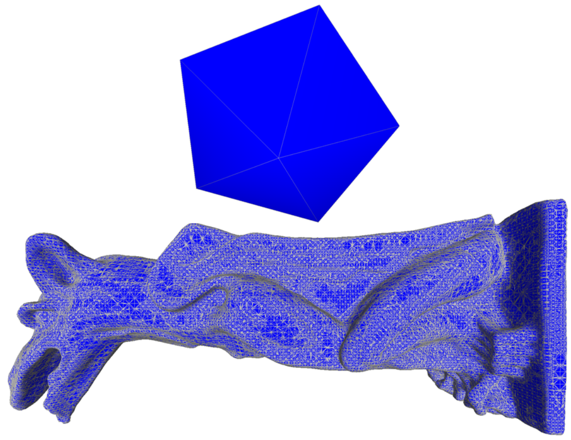


Figure 1: *Tetrahedral meshes with with genus 0: sphere and gargoyle.*

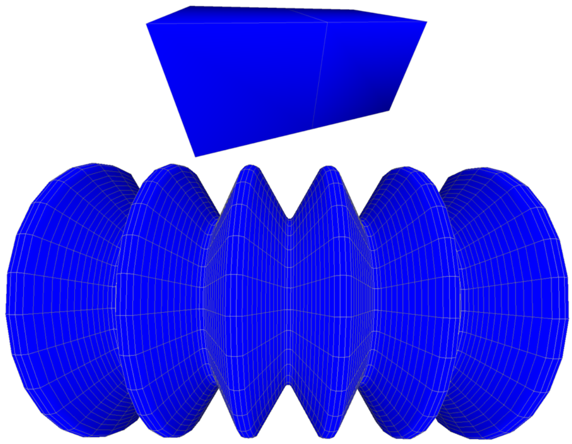


Figure 2: *Hexahedral meshes with genus 0: hex2 and rubber.*

4. Neighboring Information

After having the estimation regarding the number of components, it is important to define the needed neighboring information, in other words, the needed relationships for streamlining the application. For instance, a simulation process requires the computation of the contribution of the cells around an edge and the contributions of the cells around a vertex, therefore these relationships should efficiently be queried. These relationships can be pre-computed during the initialization phase of the mesh or these can be computed on demand during the querying process. The design decision is influenced by three factors: i) memory consumption, ii) querying performance, and iii) updating performance. If the relationships are pre-computed, more memory is required, the updating process is more time consuming, but the querying process is very fast. On the other hand, if the relationships are computed on demand, less memory is required, the update process is less expensive, but the querying process is more time consuming.

An efficient querying process (considering that sequential dependencies cannot fully be exploited) can be achieved by means of creating *topological templates*. This is motivated by the hierarchical creation of the topological components of the mesh (*Cell*, *Face* and *Edge*). For instance, when creating a tetrahedron, the tetrahedron is reponsible for creating its triangles and at the same time, each triangle is reposible for creating its edges. Hence, given this hierachical process, the edges of the tetrahedron can be inferred from the edges of the triangle by means of the *topological template*. Table 2 presents an example of a topological template for a tetrahedron, based on the schematic indications from Figure 3.

The table presents the 4 triangles (F_i) of the tetrahedron and the generated edges (Fe_i) for each triangle. The arrows indicate the correspondence between the edges of the triangles with the edges of the tetrahedron (Ce_i). For this particular case of the topological template of the tetrahedron, a simple exercise was performed with the gargoyle of Figure 1 with about 800K tetrahedra, in order to compare the performance difference between the pre-computing and computing on demand strategies. The task aimed to traverse the 6 edges of the tetrahedra in a for loop: the pre-computation case required about 40ms, while the computation on demand required about 240ms, 6 times more, nonetheless the initialization process and the updating process were faster. In both cases, the topological template was used for pre-computing and for computing on demand.

The topological templates do not only improve the querying process, they also improve the initialization and updating process, because of the built hierarchal information. Referring back to the schematic example of Figure 3, a concept of *smart edges* is represented by two opposite edges, in this case e_i and e_{i+1}. Therefore when adding a new tetrahedron to the mesh, instead of looking for existing faces (triangles) in the mesh, by means of querying faces around the vertices of the tetrahedron, the existence of the smart edge is beforehand evaluated. If the edge exists, the query is reduced to the faces around the edge, which are much less than the faces around the three vertices of the needed face.

In the case that the edge does not exist, the face cannot exist either, therefore the face can directly be created. A similar process is applied, when creating a face, but in this case, *smart vertices* are the reference for querying for the edges of the face. It is worth mentioning that depending on the domain application, the querying process can also be streamlined by means of using spatial data structures (e.g. k-dimensional tree - k-d tree, binary space partitioning - BSP, octree, bounding volume hierarchy - BVH, etc.), which assist in finding subparts of the mesh in 3D space, avoiding querying and evaluating every single component of the mesh, for instance for efficient collision detection or mesh Boolean operation applications.

Table 2: *Topological templates for a tetrahedron, showing the correspondence between the edges of the triangles and the edges of the tetrahedron, given the hierarchical creation process.*

F_i	Fe_i	Fe_{i+1}	Fe_{i+2}
V_i, V_{i+1}, V_{i+3}	V_i–V_{i+1} $\to Ce_i$	V_{i+1}–V_{i+3} $\to Ce_{i+3}$	V_{i+3}–V_i $\to Ce_{i+4}$
V_{i+1}, V_i, V_{i+2}	V_{i+1}–V_i	V_i–V_{i+2}	V_{i+2}–V_{i+1}
$V_{i+2}, V_{i+3}, V_{i+1}$	V_{i+2}–V_{i+3} $\to Ce_{i+1}$	V_{i+3}–V_{i+1}	V_{i+1}–V_{i+2} $\to Ce_{i+2}$
V_{i+3}, V_{i+2}, V_i	V_{i+3}–V_{i+2}	V_{i+2}–V_i $\to Ce_{i+5}$	V_i–V_{i+3}

5. Mesh Modifications

The access to the neighboring information of the mesh, either by pre-computing or by computing on demand, enables efficient processes (e.g. modeling, simulation, visualization) of the domain application. In the case of dynamic meshes, these processes generally change the geometry and the topology of the mesh. There are two different approaches, either the geometry remains constant and the topology is changed (e.g. progressive meshes) or the geometry is changed and those changes invoke modification in the topology (e.g. mesh modeling). In either case, the typical actions on the mesh are called *topological operations*. The basic operations are: i) edge-split, ii) edge-collapse, and iii) edge-swap, and they aim to correct degeneracies on the mesh (e.g. inverted elements – faces or cells) or to improve the quality of the mesh (e.g. condition number for numerical simulations).

Although the operations require several steps, the mesh data structure only needs to support vertex or element removal and vertex or element addition. For instance, edge-split involves removing all the elements around an edge. Edge-collapse requires removing all the elements around one of the vertices of the edge to be collapsed (sometimes also merging the two vertices), and therefore removing the vertex (with its correspinding references). Edge-swap implies removing the elements around the edge to be swapped. The addition of new vertices or elements is already provided; since these operations are used during the initialization of data structure. Any other operation should be created on a top layer, in order to avoid overloading the mesh data structure with multiple application-motivated operations.

If the modifications of the mesh are not carefully applied,

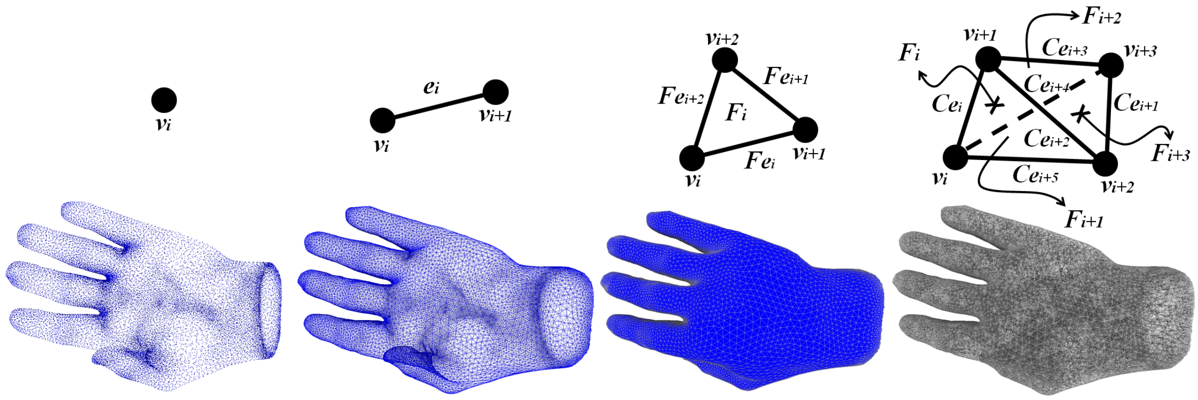


Figure 3: *Topological components of a mesh: vertices V, edges E, faces F, and cells C (model from the AIM@SHAPE repository).*

the mesh can easily become inconsistent. Thus, the data structure should also support basic geometric and topological tests, to warn the user if needed. For instance, to check the internal angles of faces and dihedral angles of cells, or to check the manifoldness of the mesh (for 2D: two faces per edge, for 3D: two cells per face or in both cases orientation of components). Nevertheless, the user should be able to enable or disable those checks, according to the needs of the application. An additional important aspect given the continuous change in the mesh is the ability to know the boundary of the mesh at any time without the need to traverse the mesh, especially for vertices and edges.

In the case of volume meshes, the faces on the boundary are easily recognized by having only one associated cell and this fact can be used for transferring information to its associated edges and vertices during the initilization and update processes. If a reference counter is associated to each vertex and edge, this needs to increase when an incident face is identified as being on the boundary and this needs to decrease, when an incident face is identified as not being on the boundary (e.g. when the neighboring cell is created). By the end of the initialization process or after the update process, a counter of 0 means that the edge or vertex is not on the boundary, any number above 0 indicated that the edge or vertex is on the boundary. This information is relevant for rapidly visualizing the changes in the mesh without performance drawbacks. Figure 4 illustrates on the top the modification of different surface and volume meshes and on the bottom the deformation (drag of a hole) of a tetrahedral mesh within a simulation environment.

6. Conclusions

There are several mesh data structures for triangular meshes and some for tetrahedral meshes. The exploration of quadrangular and hexahedral data structures is very limited in the community. Many of the data structures are designed for minimizing the memory consumption for specific domain applications. Nevertheless, there are not enough data structures, which robustly represent 3D shapes regardless of the quality of the given input mesh and there are very few data structures supporting dynamic meshes. These two aspects are an essential requirement toward general purpose geometry processing in 2D and 3D. We presented in this paper considerations in terms of memory and performance, neighboring information, and mesh modifications, toward a data structure for representing dynamic meshes in a generic and robust manner. We will further develop our ideas, in order to find the most appropriate tradeoff between the different exposed aspects, as well as new strategies for improving the performance without affecting the memory consumption and keeping the flexibility.

7. Acknowledgements

This work is partially supported by the European projects VISTRA (FP7-FoF-ICT-2011.7.4-285176) and 3D-COFORM (FP7-ICT-2007.4.3-231809).

References

[ABGA04] ALLEGRE R., BARBIER A., GALIN E., AKKOUCHE S.: A hybrid shape representation for free-form modelling. In *Proceedings of the Shape Modeling International 2004* (Washington, DC, USA, 2004), IEEE Computer Society, pp. 7–18. 2

[AGFF09] ATTENE M., GIORGI D., FERRI M., FALCIDIENO B.: On converting sets of tetrahedra to combinatorial and pl manifolds. *Computer Aided Geometric Design 26*, 8 (November 2009), 850–864. 3

[AJ05] ALUMBAUGH T. J., JIAO X.: Compact array-based mesh data structures. In *Proceedings of the 14th International Meshing Roundtable* (September 2005), Springer-Verlag, pp. 485–504. 2

[Bau72] BAUMGART B. G.: *Winged edge polyhedron representation.* Tech. rep., Stanford University, Stanford, CA, USA, October 1972. 2

[Bau75] BAUMGART B. G.: A polyhedron representation for

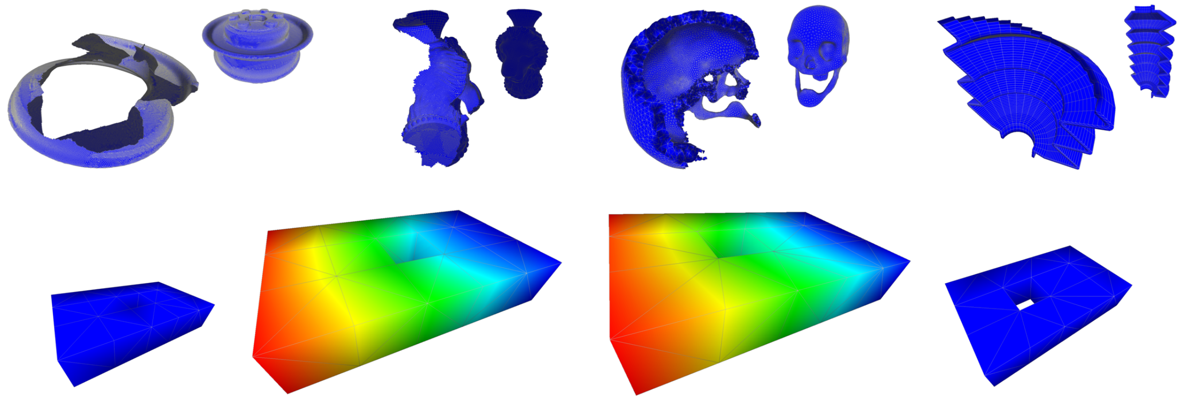


Figure 4: *Modification of an original triangular, quadrangular, tetrahedral and hexahedral mesh (some models from the AIM@SHAPE repository) on the top and a deformation of a tetrahedral mesh with a coupled simulation feedback on the bottom.*

computer vision. In *Proceedings of the May 19-22, 1975, National Computer Conference and Exposition* (New York, NY, USA, 1975), AFIPS '75, ACM, pp. 589–596. 2

[BBCK03] BLANDFORD D. K., BLELLOCH G. E., CARDOZE D. E., KADOW C.: Compact representations of simplicial meshes in two and three dimensions. In *Proceedings of the 12th International Meshing Roundtable* (September 2003), pp. 135–146. 2

[BBCK05] BLANDFORD D. K., BLELLOCH G. E., CARDOZE D. E., KADOW C.: Compact representations of simplicial meshes in two and three dimensions. *International Journal of Computational Geometry and Applications 15*, 1 (2005), 3–24. 2

[BKP*10] BOTSCH M., KOBBELT L., PAULY M., ALLIEZ P., LEVY B.: *Polygon Mesh Processing*. AK Peters, 2010. 2

[BSBK02] BOTSCH M., STEINBERG S., BISCHOFF S., KOBBELT L.: Openmesh - a generic and efficient polygon mesh data structure. OpenSG Symposium, 2002. 2

[CA03] CHEN J., AKLEMAN E.: *Topologically Robust Mesh Modeling: Concepts, Data Structures and Operations*. Tech. rep., Texas A&M University, 2003. 3

[CGG*04] CIGNONI P., GANOVELLI F., GOBBETTI E., MARTON F., PONCHIO F., SCOPIGNO R.: Adaptive tetrapuzzles: efficient out-of-core construction and visualization of gigantic multiresolution polygonal models. *ACM Transactions on Graphics 23*, 3 (August 2004), 796–803. 2

[CKS98] CAMPAGNA S., KOBBELT L., SEIDEL H.-P.: Directed edges - a scalable representation for triangle meshes. *Journal of Graphics Tools 3*, 4 (December 1998), 1–11. 2

[DDF02] DANOVARO E., DE FLORIANI L.: Half-edge multi-tessellation: A compact representation for multi-resolution tetrahedral meshes. In *Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission* (Washington, DC, USA, 2002), 3DPVT '02, IEEE Computer Society, pp. 494–499. 3

[DDFM*05] DANOVARO E., DE FLORIANI L., MAGILLO P., PUPPO E., SOBRERO D., SOKOLOVSKY N.: The half-edge tree: A compact data structure for level-of-detail tetrahedral meshes. In *Proceedings of the International Conference on Shape Modeling and Applications* (Washington, DC, USA, 2005), SMI Š05, IEEE Computer Society, pp. 332–337. 3

[DFGH04] DE FLORIANI L., GREENFIELDBOYCE D., HUI A.: A data structure for non-manifold simplicial d-complexes. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing* (New York, NY, USA, 2004), SGP '04, ACM, pp. 83–92. 2

[DFH03] DE FLORIANI L., HUI A.: A scalable data structure for three-dimensional non-manifold objects. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing* (Aire-la-Ville, Switzerland, Switzerland, 2003), SGP '03, Eurographics Association, pp. 72–82. 2

[DFH04] DE FLORIANI L., HUI A.: Update operations on 3d simplicial decompositions of non-manifold objects. In *Proceedings of the ninth ACM Symposium on Solid Modeling and Applications* (Aire-la-Ville, Switzerland, 2004), SM '04, Eurographics Association, pp. 169–180. 3

[DFH05] DE FLORIANI L., HUI A.: Data structures for simplicial complexes: an analysis and a comparison. In *Proceedings of the third Eurographics Symposium on Geometry Processing* (Aire-la-Ville, Switzerland, Switzerland, 2005), SGP '05, Eurographics Association, pp. 119–128. 2

[DFKP05] DE FLORIANI L., KOBBELT L., PUPPO E.: A survey on data structures for level-of-detail models. *Advances in Multiresolution for Geometric Modelling 1* (2005), 57–82. 2

[DFMPS02] DE FLORIANI L., MAGILLO P., PUPPO E., SOBRERO D.: A multi-resolution topological representation for non-manifold meshes. In *Proceedings of the seventh ACM Symposium on Solid Modeling and Applications* (New York, NY, USA, 2002), SMA '02, ACM, pp. 159–170. 3

[DFMPS04] DE FLORIANI L., MAGILLO P., PUPPO E., SOBRERO D.: A multi-resolution topological representation for non-manifold meshes. *Computer-Aided Design 36*, 2 (2004), 141–159. 3

[DT07] DECORO C., TATARCHUK N.: Real-time mesh simplification using the gpu. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games* (New York, NY, USA, 2007), I3D '07, ACM, pp. 161–166. 3

[FG00] FREY P. J., GEORGE P.-L.: *Mesh Generation: Application to Finite Elements*. HERMES Science, 2000. 2

[GLLR11a] GURUNG T., LANEY D., LINDSTROM P., ROSSIGNAC J.: Squad: Compact representation for triangle meshes. *Computer Graphics Forum 30*, 2 (2011), 355–364. 3

[GLLR11b] GURUNG T., LUFFEL M., LINDSTROM P., ROSSIGNAC J.: Lr: compact connectivity representation for triangle meshes. *ACM Trans. Graph. 30*, 4 (July 2011), 67. 3

[GR09] GURUNG T., ROSSIGNAC J.: Sot: compact representation for tetrahedral meshes. In *Proceedings of the 2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling* (New York, NY, USA, 2009), SPM '09, ACM, pp. 79–88. 3

[GR10] GURUNG T., ROSSIGNAC J.: *SOT: compact representation for tetrahedral meshes*. Technical Report GT-IC-10-01, Georgia Institute of Technology, 2010. 3

[HSH09] HU L., SANDER P. V., HOPPE H.: Parallel view-dependent refinement of progressive meshes. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games* (New York, NY, USA, 2009), I3D '09, ACM, pp. 169–176. 3

[HVDF06] HUI A., VACZLAVIK L., DE FLORIANI L.: A decomposition-based representation for 3d simplicial complexes. In *Proceedings of the fourth Eurographics Symposium on Geometry Processing* (Aire-la-Ville, Switzerland, Switzerland, 2006), SGP '06, Eurographics Association, pp. 101–110. 3

[Ket98] KETTNER L.: Designing a data structure for polyhedral surfaces. In *Proceedings of the Fourteenth Annual Symposium on Computational Geometry* (New York, NY, USA, 1998), SCG '98, ACM, pp. 146–154. 2

[Ket99] KETTNER L.: Using generic programming for designing a data structure for polyhedral surfaces. *Computational Geometry: Theory and Applications 13*, 1 (May 1999), 65–90. 2

[KT02] KALLMANN M., THALMANN D.: Star-vertices: a compact representation for planar meshes with adjacency information. *Journal of Graphics Tools 6*, 1 (January 2002), 7–18. 2

[LCCC01] LÉVY B., CAUMON G., CONREAUX S., CAVIN X.: Circular incident edge lists: a data structure for rendering complex unstructured grids. In *Proceedings of the Conference on Visualization '01* (Washington, DC, USA, 2001), VIS '01, IEEE Computer Society, pp. 191–198. 2

[Lee99] LEE K.: *Principles of CAD / CAM / CAE Systems*. Addison-Wesley, 1999. 1

[LH06] LEFEBVRE S., HOPPE H.: Perfect spatial hashing. *ACM Trans. Graph. 25*, 3 (July 2006), 579–588. 3

[LLLV05] LAGE M., LEWINER T., LOPES H., VELHO L.: Chf: A scalable topological data structure for tetrahedral meshes. In *Proceedings of the XVIII Brazilian Symposium on Computer Graphics and Image Processing* (Washington, DC, USA, 2005), IEEE Computer Society, pp. 349–356. 3

[LSK*06] LEFOHN A. E., SENGUPTA S., KNISS J., STRZODKA R., OWENS J. D.: Glift: Generic, efficient, random-access gpu data structures. *ACM Trans. Graph. 25*, 1 (January 2006), 60–99. 3

[MB91] MUUSS M. J., BUTLER L. A.: Combinatorial solid geometry, boundary representations, and non-manifold geometry. *State of the Art in Computer Graphics: Visualization and Modeling 1* (June 1991), 185–223. 2

[PSSSM09] PENA SERNA S., SILVA J., STORK A., MARCOS A.: Neighboring-based linear system for dynamic meshes. In *Proceedings of the 6th Workshop in Virtual Reality Interactions and Physical Simulations* (Aire-la-Ville, Switzerland, Switzerland, 2009), VRIPHYS '09, Eurographics Association, pp. 95–103. 2

[SB11] SIEGER D., BOTSCH M.: Design, implementation, and evaluation of the surface_mesh data structure. In *Proceedings of the 20th International Meshing Roundtable* (2011), IMR, pp. 533–550. 3

[SNCH08] SANDER P. V., NEHAB D., CHLAMTAC E., HOPPE H.: Efficient traversal of mesh edges using adjacency primitives. *ACM Trans. Graph. 27*, 5 (December 2008), 144:1–144:9. 3

[TM06] TOBLER R. F., MAIERHOFER S.: A mesh data structure for rendering and subdivision. In *Proceedings of WSCG: International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision* (2006), pp. 157–162. 3

[Wei85] WEILER K.: Edge-based data structures for solid modeling in curved-surface environments. *IEEE Computer Graphics and Applications 5*, 1 (January 1985), 21–40. 3

[WMKE04] WEILER M., MALLON P. N., KRAUS M., ERTL T.: Texture-encoded tetrahedral strips. In *Proceedings of the 2004 IEEE Symposium on Volume Visualization and Graphics* (Washington, DC, USA, 2004), VV '04, IEEE Computer Society, pp. 71–78. 2

Short Papers

Gestural 3D Interaction with a Beating Heart: Simulation, Visualization and Interaction

F. Ioakemidou[2], F. Ericson[2], J. Spühler[1], A. Olwal[3], J. Forsslund[2], J. Jansson[1], E.-L. Sallnäs Pysander[2] and J. Hoffman[1]

[1]Numerical Analysis, School of Computer Science and Communication, KTH, Sweden
[2] Human-Computer Interaction, School of Computer Science and Communication, KTH, Sweden
[3] Media Laboratory, School of Architecture and Planning, MIT, USA

Abstract

The KTH School of Computer Science and Communication (CSC) established a strategic platform in Simulation-Visualization-Interaction (SimVisInt) in 2009, focused on the high potential in bringing together CSC core competences in simulation technology, visualization and interaction. The main part of the platform takes the form a set of new trans-disciplinary projects across established CSC research groups, within the theme of Computational Human Modeling and Visualization: (i) interactive virtual biomedicine (HEART), (ii) simulation of human motion (MOTION), and (iii) virtual prototyping of human hand prostheses (HAND). In this paper, we present recent results from the HEART project that focused on gestural and haptic interaction with a heart simulation.

Keywords: *Virtual and Interactive Environments*

1. Heart

Cardiac disease is the major cause of death in western society. The vision of the HEART project is to build a virtual human heart from medical imaging data, together with an interactive interface (see Figure 1) with visual, haptic and sonic feedback. External collaborators include Umeå University, Linköping University, KTH School of Technology and Health, and the Barcelona Supercomputing Center. The basic idea is to use medical imaging technology to create a virtual model of the heart wall motion, blood flow dynamics, and mechanical stresses. Visual, haptic and sonic interaction with the heart model can be used to get an enhanced understanding of the heart function, and for help in diagnosis as a complement to medical imaging information. In addition, the interactive interface opens for modifications of the virtual heart, simulating cardiac disease or available treatment options, such as choosing between different mechanical heart valves, and thus offering a tool for decision support in planning surgery or other treatment.

Today the model takes the form of a left ventricle blood flow simulation, based on a geometrical model from patient-specific ultrasound data (Figure 2), including a visual and haptic interface [Aec09], [HJ06].

Figure 1: *3D interaction with a simulation of the blood flow in the left ventricle of the heart.*

2. The numerical simulation

The simulation is produced by first defining a moving geometry from measurements for which a mesh can be generated (described in subsection 2.1), the incompressible Navier-Stokes equations are then posed on the geometry (described in subsection 2.2), and numerically solved by a finite ele-

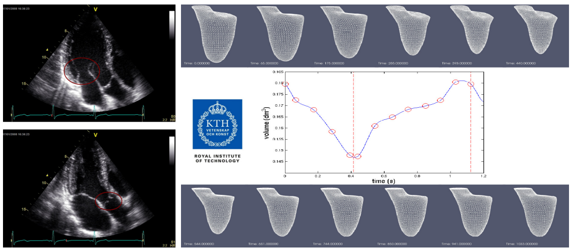


Figure 2: *Geometric model of the left ventricle constructed from ultrasound in collaboration with Umeå University (Mats G. Larson, Anders Waldenström, Ulf Gustafsson and Per Vesterlund).*

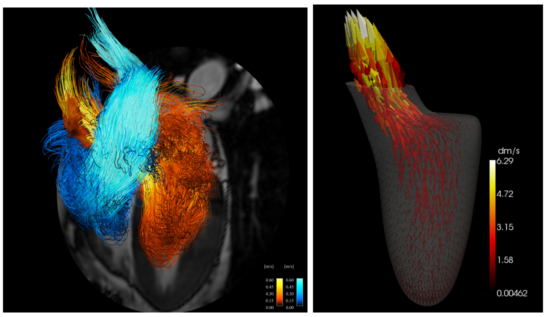


Figure 3: *MRI measurement (left) from Linköping University (Tino Ebbers), and computer simulation (right) of the blood flow in the heart in collaboration with KTH-STH (Lars-Åke Brodin and Michael Broomé).*

ment method (FEM) (described in subsection 2.3). The software implementation of the method is briefly described in subsection 2.4 together with the results.

2.1. From patient-specific ultrasound to the geometrical heart model

In our approach [Aec09] the blood in the left ventricle (LV) is driven by the prescribed movement of the inner heart wall. The model geometry of the LV is based on ultrasound measurements of the position of the inner wall at three different levels at twelve specific time points during the cardiac cycle. This is of relevance, as the movement of the heart wall is not uniform during the heart cycle that lasts about one second. Surface meshes of the chamber are constructed from these measured points, for each sample time, as shown in Figure 2. We then build a three-dimensional mesh of tetrahedrons at the initial time and apply Hermite interpolation to allocate the position of all boundary nodes at every time step. The Laplace mesh smoothing algorithm is used to deform the mesh to fit the given surface meshes during the cardiac cycle.

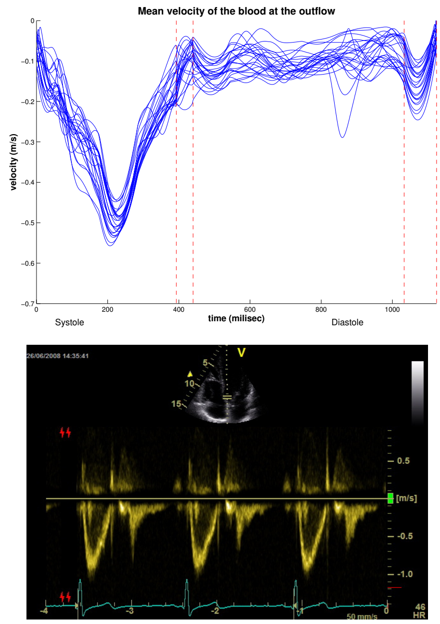


Figure 4: *Mean velocity measured at arbitrary points at the outflow in the simulation and velocity at the outflow derived with help of ultrasound measurements*

2.2. Mathematical model

We simulate the blood flow with the incompressible Navier-Stokes equations. The Arbitrary Lagrangian-Eulerian description [DHPRF04] is used to account for the movement of the mesh. We seek the velocity u and pressure p:

$$\begin{aligned} \dot{u} + ((u - \hat{w}) \cdot \nabla)u - \nu\Delta u + \nabla p &= 0 \\ \nabla \cdot u &= 0 \end{aligned} \tag{1}$$

with ν the kinematic viscosity and $\hat{w}$ the mesh velocity.

By defining different boundary conditions, we divide the cardiac cycle into the basic stages of diastole and systole. A no-slip boundary condition on the wall and closed valves is applied and the pressure is prescribed to model the inflow through the mitral valve and the outflow through the aortic valve.

2.3. FEM with Streamline Diffusion Stabilization

We apply the General Galerkin (G2) cG(1)cG(1) Finite Element Method (FEM) [HJ06] with piecewise continuous linear solution in time and space for solving the governing equation 1. For the standard FEM formulation of the model we only have stability of the discrete solution U but not of its spatial derivatives. This means that the solution can be oscillatory, causing inefficiency by introducing unnecessary error. We therefore choose a weighted standard

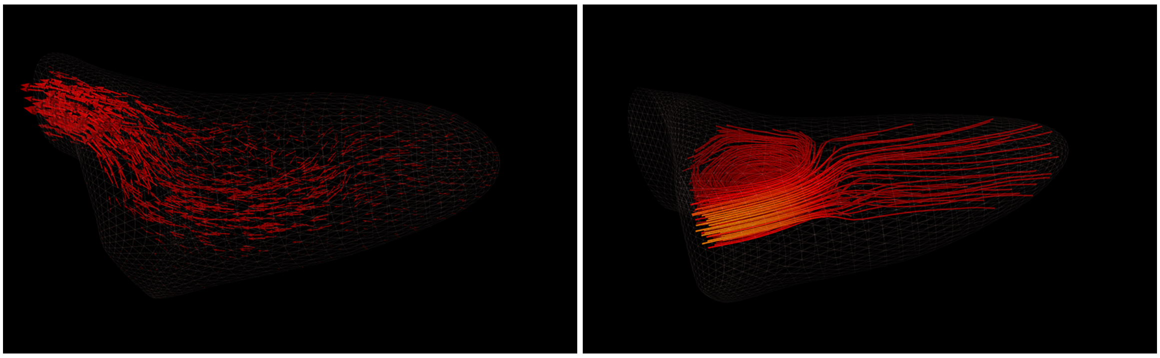


Figure 5: *Simulated blood flow velocity represented as directional 3D glyphs (left) and streamlines (right).*

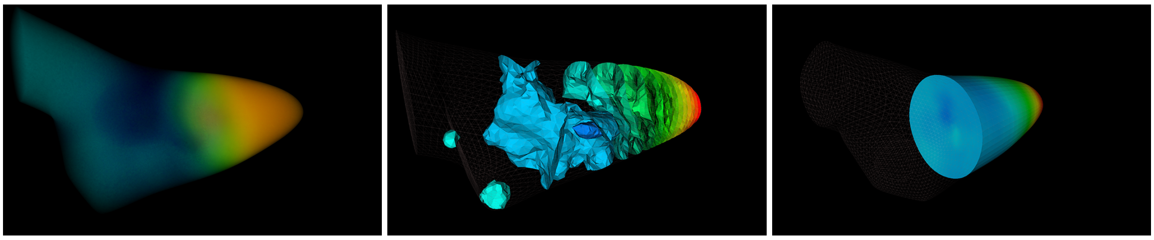


Figure 6: *The pressure data visualized with volume rendering (left), ISO surfaces (center), and using a cut plane for slicing (right).*

FEM/streamline diffusion method as follows: find the discrete velocity U and pressure P in the corresponding discrete function spaces V_0 and Q with test functions v and q such that

$$\begin{aligned}&((U^n - U^{n-1})k_n^{-1} + (\bar{U}^n - \hat{W}^{n-1})\cdot\nabla\bar{U}^n, v)\\&+(2\nu\varepsilon(\bar{U}^n),\varepsilon(v)) - (P^n,\nabla\cdot v) + (\nabla\cdot\bar{U}^n, q)\\&+\mathrm{SD}_\delta(\bar{U}^n, P^n; v, q) = 0 \quad \forall (v,q)\in V_0^n\times Q^n \qquad (2)\end{aligned}$$

where $\bar{U}^n = 1/2\left(U^n + U^{n-1}\right)$, $\varepsilon(u) = \frac{1}{2}(\nabla u + \nabla u^T)$, k_n is the time step on time interval n, and with the stabilization term defined as

$$\begin{aligned}&\mathrm{SD}_\delta(\bar{U}^n, P^n; v, q) =\\&(\delta_1((\bar{U}^n - \hat{W}^{n-1})\cdot\nabla\bar{U}^n + \nabla P^n - f), (\bar{U}^n - \hat{W}^{n-1})\cdot\nabla v + \nabla q)\\&+(\delta_2\nabla\cdot\bar{U}^n, \nabla\cdot v) \qquad (3)\end{aligned}$$

δ_1 and δ_2 are the stabilization parameters.

2.4. Simulation results

The simulations are done in DOLFIN [LW10], a differential equation problem solving environment, and UNICORN [HJNJ11], a unified continuum mechanics solver, which are developed as a part of the software project FEniCS and where we are active contributors.

The mean velocity at the outflow and inflow, as well as the pressure, are quantities that are interesting for medical examination and that can also be used for evaluation of the model. An example is shown in Figure 4.

3. Gestural interaction with a 3D Visualization

The interactive HEART demonstration is a public showcase designed to aid the perception and understanding of the simulation. Blood flow and pressure are visualized in 3D and can be manipulated using physical movement and gestures.

3.1. Visualization of blood flow

The system was developed using the Visualization Toolkit (VTK, http://www.vtk.org) which provides efficient implementations of many advanced graphics techniques. Our visualization animates deformations of the heart geometry throughout a heartbeat while illustrating other properties of the simulation.

The blood flow velocity field produced by solving eq. 2 in UNICORN is illustrated by arrows proportional in size to the flow velocity. Optionally flow can be visualized by animated streamlines, both alternatives shown in Figure 5.

The pressure data (also from eq. 2) is visualized using hardware accelerated volume rendering techniques. Optionally ISO surfaces can be used or an interactive cut plane. Alternatives shown in Figure 6.

The system also supports stereoscopic rendering with head tracking, as shown in Figure 7.

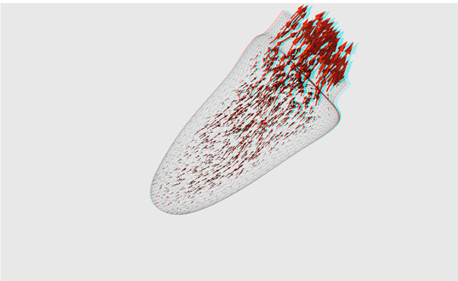


Figure 7: *The system also supports stereoscopic rendering, here enabled in anaglyph (red/cyan) mode.*

3.2. Gestural interaction using a depth-sensing camera

A depth-sensing camera (Microsoft Kinect) was used to acquire input for natural interaction. The OpenNI tracking library (http://www.openni.org) was combined with the OpenCV computer vision library (http://opencv.willowgarage.com) to recognize gestures and track the joints of users in 3D.

The OpenNI library provides real time 3D tracking of human body joints, of particular interest to us are hands and head. In order to provide the algorithm with necessary body metrics the user must first take the calibration pose shown in Figure 8.

Figure 8: *For skeleton tracking of the user, a one-time calibration pose is required initially.*

3.3. Detection of open/closed hands

The 3D tracking provides accurate positional data, which we use to manipulate parameters of the visualization. Since we track the user's hands constantly we need to provide a way for users to easily initiate / abort interaction with the simulation.

By swiping one closed hand in the air the user can turn the simulation around to inspect it from different viewpoints. The camera can also be zoomed out and in by closing both

Figure 9: *The user "grasps the heart" to rotate it with the right hand.*

hands and moving them closer together or further apart respectively, as shown in Figure 9 and Figure 10.

The detection of open/closed hands is achieved by the combination of different techniques which aim to cover the majority of cases. First the a polygon approximation of the hand contour is acquired from the depth camera image, in which we can locate and isolate the hands using the information of the tracked joints. Similarity of the the hand contour area to the area of its convex hull is often a strong indication of a closed hand. To improve this estimate we analyze in detail the differences between the contour and hull. Finally we apply a custom algorithm to look for sharp consecutive changes of direction along the contour, usually signifying outstretched fingers. By experimenting with the parameters of these algorithms we arrived at a solution that makes accurate predictions in real time for most cases and does not produce any significant lag. See Figure 11.

3.4. Haptic feedback

Additional experiments were conducted using a desktop application built in C++ and CHAI 3D (http://www.chai3d.org/), which allowed the use of haptic devices (e.g., SensAble Phantom Omni, shown in Figure 12, left) for experiencing the dynamics in a heart beat. It allowed the user to move a 3D cursor inside the blood flow simulation and "feel" the movement of particles through the haptic feedback propagated through the device's stylus. The particular haptic device used could, however, only generate forces to control a single point's 3D position, which would be mapped to the stylus's tip. Haptic devices with 6 degrees of freedom (i.e., 3D position and 3D torque) are significantly more costly and complex, but most still only allow interaction at a single point.

3.5. Lo-fi haptics with motorized faders

As an alternative solution, a series of experiments were also conducted using a low-cost control fader device (Behringer BCF2000, shown in Figure 12, right) that has eight motorized faders. This device allowed multi-dimensional input (from the user) and output (from the simulation). The fader device was interfaced using the MIDI protocol and made it

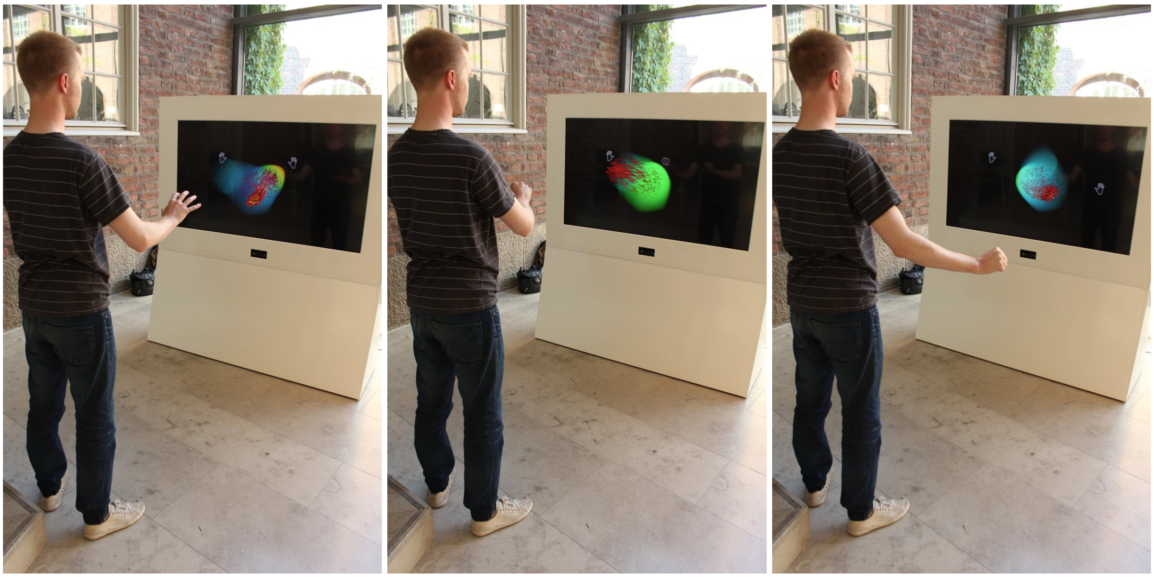


Figure 10: *From left to right: a) The user has both hands open, as also indicated by the on-screen hand symbols. b) The right hand is closed, which allows the user to rotate the model, by moving the closed hand. c) After moving the hand the user observes the simulation from another angle, the hand must now be opened to stop rotating.*

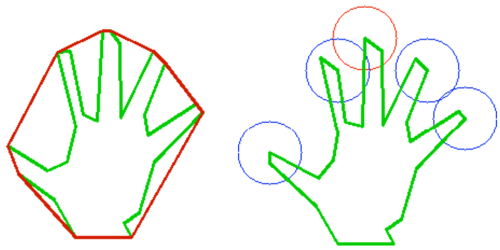


Figure 11: *Hands are analyzed by comparing the contour to its convex hull, and by using a finger detection algorithm. This makes it possible to establish whether hands are open or closed.*

possible to map various parameters to fader movement on the device. More motorized faders are straightforward to add by daisy-chaining multiple devices, which is natively supported by this type of the device as well as the MIDI protocol.

References

[Aec09] AECHTNER M.: *Arbitrary Lagrangian-Eulerian Finite Element Modelling of the Human Heart*. Master's thesis, 2009. 1, 2

[DHPRF04] DONEA J., HUERTA A., PONTHOT J.-P., RODRIGUEZ-FERRAN A.: *Arbitrary Lagrangian-Eulerian Methods*. Wiley, 2004, pp. 413–437. 2

[HJ06] HOFFMAN J., JOHNSON C.: *Computational Turbulent Incompressible Flow: Applied Mathematics Body and Soul Vol 4*. Springer-Verlag Publishing, 2006. 1, 2

[HJNJ11] HOFFMAN J., JANSSON J., NAZAROV M., JANSSON N.: Unicorn: A Unified Continuum Mechanics Solver. In *Automated Scientific Computing*. Springer, 2011. 3

[LW10] LOGG A., WELLS G. N.: DOLFIN: Automated finite element computing. *ACM Trans. Math. Softw. 37*, 2 (2010), 1–28. 3

Figure 12: *Left) The Omni haptic device from SensAble uses a mechanically tracked stylus that provides 6 degrees of freedom input (3D position + 3D orientation) and force feedback in 3 dimensions. Right) The Behringer BCF2000 is a fader controller device, with eight motorized faders that can be mechanically controlled from a computer application using the MIDI protocol.*

Multi-State Device Tracking for Tangible Tabletops

Ali Alavi[1], Brice Clocher[1], Allen Smith[1], Andreas Kunz[2], and Morten Fjeld[1]

[1]t2i Lab, Chalmers TH, SE-41296 Gothenburg, Sweden
[2]ICVR, ETH Zurich, CH-8092 Zurich, Switzerland

Abstract

On tangible tabletops, Tangible User Interfaces (TUIs) can signalize their identity, position, orientation, and state by active infrared light. This provides rich interaction capabilities in complex, dynamic scenarios. If TUIs have to transfer additional high-resolution information, many bits are required for each update. This has a negative impact on the overall update rate of the system. In the first part of this paper, we present an in-house map application where interaction with time-dependent contour lines may benefit from high-resolution TUI states. Prototypical TUI concepts such as slider, ruler, and dials further motivate the benefit of high-resolution tracking. In the second part of the paper, we depart from a device tracking overview and then show how tangible devices for tabletops typically use infrared (IR) emitters and a camera to send information about their position, orientation, and state. Since transferring many additional information bits via a normal camera-based tabletop system is not feasible anymore, we introduce next a new system setup that still offers a sufficiently high update rate for a smooth interaction. The new method can be realized as a tabletop system using a low-cost camera detecting position, combined with a low-cost infrared receiver detecting the state of each device. Since both kinds of sensors are used simultaneously we call the method "dual mode." This method combines a camera-based tracking with the possibility to transfer an almost unlimited amount of states for each device.

Categories and Subject Descriptors (according to ACM CCS):
General terms: Algorithms, Performance
Keywords: Active tangible devices, tabletop, dual mode, IR tracking, multiple TUI

1. Introduction

The advent of widely available interactive tabletops has created high expectations among users for such systems. Tangible tabletops where active devices (TUIs) are tracked to inform about their identity, position, orientation, and state can provide rich interaction within complex, dynamic scenarios. InfrActables [GSK06] delivers such an experience by providing users with active TUIs. In that system, TUIs have form factors such as pen, handle, ruler, and color tool, which may enable a more natural interaction style. Each of these TUIs has a few state-triggering widgets, such as buttons on top of the bricks or a micro switch under the pen's tip. Thereby, users can control states while positioning the TUI on the surface. Informing about TUI states over a large range at a high-resolution requires sending many bits for each update and comes at the cost of system update rate. However, tracking of TUIs has so far been limited to the spatial and temporal resolution of the camera CCD chip. Future scenarios where TUIs control dynamic high-resolution parameters of dynamic map-based scenarios, economic simulations, or science education are promising [KF10]. In such application, TUIs with dynamic high-resolution input streams will enable system designers to use richer forms of interaction. For instance, a slider, ruler, or dial equipped with a potentiometer can send its input value (sampled though an A/D converter) offering up to 2048 adjustment levels. To implement such functions for systems like InfrActables, a TUI needs to employ a much higher number of state bits than today. We could not utilize this number of states on the initial system because the update rate would then drop dramatically. In this paper, we address this problem by introducing a new additional state detection method for use in active tangible tabletops. This method allows us to build tangible tabletops with a high number of states using low-cost components.

The first part of this paper presents an existing map-based application where interaction with time-dependent contour

Figure 1: *A tangible tabletop map-based application for TUI and pen-based interaction with Google Maps.*

lines may benefit from high-resolution TUI states. Previously researched TUIs such as slider, ruler, and dials can be expected to work with our system. Besides being a research contribution in and of itself, our application serves also as motivation to research improved multi-state device tracking. In the second part of this paper, we first give an overview of alternative forms of tabletop tracking before we focus on the tracking of actively emitting devices. The second contribution of this paper, an improved multi-state device tracking method, called "dual mode", is then presented. The paper summarizes the current status and indicates potential future work.

2. Tangible Tabletop Map-based Application

Users involved in time-critical planning with interactive maps sometimes use physical tools like ruler, dials, and pens in order to share knowledge with each other and collaborate in creating a common operational picture. Previous research has demonstrated that tangible tabletops can help in these tasks [PIA09]. For the purpose of crisis resource management, we have built an interactive table with tangible devices and an information visualization framework prototype. The framework allows creating crisis management scenarios using Google Maps-based Flash applications. To interact with this application, we propose actively emitting TUIs such as slider, ruler, and dials. While the design and use of these devices has been proposed in related projects [GSK06] [SJG*06] [WWJ*06], we consider the tracking method presented next as critical in making the use of such devices successful.

One of the most important features of a tangible crisis management application is believed to be time-dependent shape, or contour line, editing. Indeed, crisis management experts often draw shapes on paper maps, for instance, in order to represent the spreading of a fire. Those shapes are

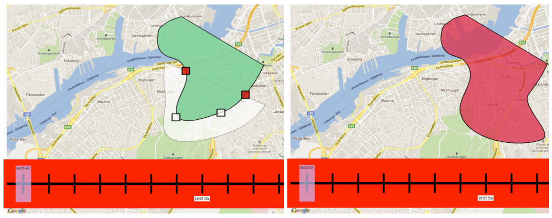


Figure 2: *Map interaction screenshot: Shape control is done by moving the nodes of a parametric curve; time control employs time-line at the bottom.*

associated with thematic, spatial, and temporal content. In the case of a fire, a shape may represent an area that is burning at a specific time. In one related example, Igarashi et al. [IMH05] presented algorithms and applications where users can move and deform a two-dimensional shape without manually establishing freeform deformation (FFD) domain beforehand. Inspired by this work, one of our current implementations employs a so-called parametric curve description. Using Flash parametric curve libraries, we have been able to implement from a rather high level of abstraction (Figure 2). When expert users receive data about the development of a phenomenon, they often model this development using time-dependent shapes. A first problem encountered when drawing shapes on a map is how to associate temporal and thematic content with a shape. In our system this raises two issues: firstly, how tangible interaction can make shape and time control precise but still easy and intuitive, and, secondly, how a user can modify shapes while still being able to keep track of their associated temporal, spatial, and thematic contents. As for the first issue, we conjecture that TUIs such as slider, ruler, and dials may benefit shape and time control. Figure 3 shows some prototypical uses of multi-state devices such as dials and a frame. Tracking TUI states with values assuming values over a large range at a high-resolution requires sending many bits for each update without compromising the system update rate. Achieving this without making compromising on system update rate is the focus for the remainder of this paper.

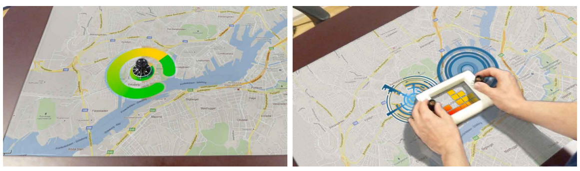


Figure 3: *Multi-state device on existing tabletop system showing tangible continuous parameter control for values such as time, radiation level, or population (left); selection frame with two-handed continuous control of radar visualizations or mode selectors (right). (Simulated images)*

2.1. Device tracking

Since a tabletop system is both a display and a multitouch input device at the same time, the complementary use of physical devices providing input may benefit certain types of applications [SR09]. Such devices may improve the fluidity and reduce the cognitive load of the user-system interaction [FB09]. Thus, the tabletop system must be able to distinguish between intended input from devices, and unintended input from other objects on the table [KF10]. Furthermore, tabletop systems must be able to detect multiple devices simultaneously when one or multiple users are interacting with the system.

2.2. Tracking in Tabletop Systems

In order to enable intuitive interaction with the content visible on the tabletop, devices other than the mouse and keyboard must be used. There is a class of devices that are easily identifiable by their inherent function known as 'physical icons' or phicons [IU97]. In this case, each device usually has a static association so that the tabletop system is able to detect its identifier (ID) in addition to its position. Once the device's ID is known to the system, the underlying functionality is also defined since the association cannot be changed. However, other tabletop devices might have a dynamic association that allows for simpler detection algorithms. In the latter case, the devices have a more general character, and so the intuitiveness is only guaranteed by the displayed content, i.e., the graphical user interface (GUI). The dynamic association is user-triggered and follows predefined steps. These steps may require some learning on the part of the user.

Tabletop systems must be able to detect the position of an interaction device and, in the case of phicons or other specialized input devices, their ID. While it is important in a global context for the position of the device to be displayed on the tabletop's surface, the ID is relevant for integrating a device's specialized functionality into a specific application. More degrees of freedom (DOF) than given by planar interaction become relevant. For instance, the z-coordinate may be used to distinguish between writing and pointing in pen-based interaction. Additionally, the tracking and detection system's latency should be below the user's perceptual threshold, otherwise user irritation may occur. During normal operation on a tabletop system, various objects may be placed on the surface which are not meant for interaction, but which could interfere with the system, e.g. by shadowing effects. Unlike a mouse, which is a relative pointing device that detects the travelling distance and orientation, all tracking systems for tabletop systems allow absolute pointing: the object is detected at precisely the place where the user puts the device.

2.3. Tracking Active Devices

Our work has been inspired by two contributions in the area of tangible tabletops using active TUIs: SmartFiducial [HK11] and MightyTrace [HKK08] [Hof11]. In SmartFiducial, active tangibles communicate with a host computer using wireless radio frequency (RF) transmission. Object positions are detected using a visual tracking system, thus adding a further level of complexity to the TUI design, as well as the host computer. Moreover, there is always a risk of unintended interference with RF communication. Hence, we deemed this approach to be unsuitable for our application. Inspired by the fast IR-sensors in MightyTrace, we decided to apply them as additional sensors to a camera-based system. In MightyTrace, a matrix of IR sensors is used to detect the position and state of TUIs. Each device is assigned a specific time frame during which its LED is turned on. Thus, each device can be detected unambiguously. For sending states, there are even more time slots for each device. For example, if a device has eight different states it then needs four time slots, one for detecting its position and three for sending its state bits (thus eight different states). However, the main restriction with such a time-multiplexed approach is that the system update rate is limited not only by the number of devices on the surface, but also by the number of states each device may have.

2.4. Dual Mode Tracking Method

Here, we introduce a new approach to improve multi-state IR tracking of tangibles on tabletops. Mainly, we aim to design low-latency active devices, that is, devices with more states without reducing the high refresh rate of the system. Moreover, we are interested in a cost-effective solution. We suggest a distinct way to combine a low-cost camera for position detection and a low-cost IR receiver for state detection of each device, in what we call a dual mode approach.

Considering the physical size of a table and the states required for tools such as sliders, rulers, and dials, we may simultaneously have five devices on the surface, each with 2048 possible states or adjustment levels. To meet this requirement, we use two different receivers: a camera capable of detecting positions, and an IR receiver capable of detecting the states of the devices. Concerning latency requirements, two terms are frequently employed: "update rate," which is the number of positions and states being updated per second, and "lag," which is the "response time" of the system to user input.

Much like in QualiTrack [HNK09], the devices and the camera are synchronized. Moreover, each device is assigned to a specific time slot. All devices emit IR light on each frame, except on their assigned time slot (Figure 4). Since the camera sees all the devices in every frame except one, the average update rate of the system equals f (M-1)/M, where f is camera frame rate (Hz) and M is the maximum number of devices we want to have on the table (which is preconfigured and constant in the system). Camera frames are indexed in cycles of M frames. Since newly placed devices wait for up to one cycle to start IR transmission, the maximum setup delay equals M/f. Thus, using the dual mode method, the number of devices does not reduce the system's update rate, nor does it increase its lag. Only the setup delay will be negatively affected. We can now unambiguously identify each device and its position. Device tracking uses a blob-tracking algorithm [SHB*10]. While we assume that a device has one LED source only, instrumenting a device with two or more sources and combining their positions can give device orientation [Hof11].

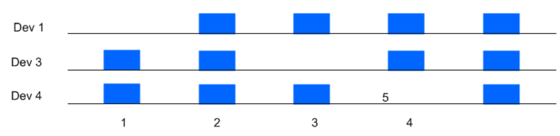


Figure 4: *Position information transfer example with M set to 5 and with devices 1, 3, and 4 (rows) present. A cycle of 5 frames (columns) is shown. Blue cells are IR flashes sent by the devices to be detected by the camera.*

Each device transmits its state information using its IR LED between two synchronization signals, i.e., the speed of transmitting the state information is significantly higher than the speed of the camera (Figure 5). This is feasible since the state information is read by a simple IR receiver and not by the camera. The interval between two consecutive camera frames is further divided into M sub-frames. Within each sub-frame, only the corresponding device sends its state information. The bit rate, R, of the sensor we employ can be up to 22 kbps. Hence, with the camera exposure time, e, and f and M as defined above, the maximum number of state bits per device equals: R ((1/f - e)/M). For example, with M set to 5, f at 60 Hz, and e at 10 ms, each device can transmit up to 29 bits of state information, allowing more than half a billion states per device.

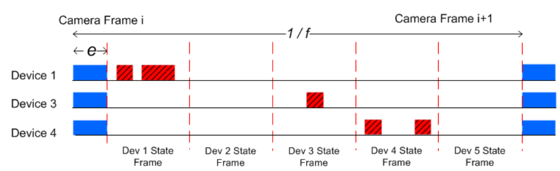


Figure 5: *State information transfer example with M set to 5 and with devices 1, 3, and 4 (rows) present. One interval between camera frames i and i+1 is shown. Blue cells are IR flashes to be detected by the camera; red cells are IR flashes to be detected by the IR receiver.*

2.5. Current status and future works

At an application level, we plan to further examine tasks requiring the use of multi-state tangible devices on tabletops. To this end, we plan to realize the slider, ruler, and dials accompanying the pen for use with our map-based tabletop application. Drawing on design principles and solutions from previous work [HKK08] [HNK09] [GSK06], this will call for engineering new TUIs tailored to this use. In a later phase, we foresee designing, running, and analyzing user studies to validate the usability and acceptance of the solutions.

At a tabletop device tracking level, we have evaluated the feasibility of the proposed method by implementing its essential subsystems. In particular, we implemented the IR receiver and changed the QualiTrack TUIs to send state information using our dual mode method. We also investigated whether the battery operated TUIs allow us to send signals powerful enough to be detected by the IR receiver, considering the distance between them. Our findings show that it is feasible to implement a tabletop using our new method. A next step in this project will be to implement a complete tangible tabletop using the method with the suggested slider, ruler, and dials.

2.6. Acknowledgements

We greatly thank Erik Tobin for improving the readability of this text.

References

[FB09] FJELD M., BARENDREGT W.: Epistemic action: A measure for cognitive support in tangible user interfaces? *Behavior Research Methods 41*, 3 (2009), 876–881. 3

[GSK06] GANSER C., STEINEMANN A., KUNZ A.: Infractables: Multi-user tracking system for interactive surfaces. In *Proc. of the IEEE conference on Virtual Reality (VR '06)* (2006), pp. 253–256. 1, 2, 4

[HK11] HOCHENBAUM J., KAPUR A.: Adding z-depth and pressure expressivity to tangible tabletop surfaces. In *Proceedings of the International Conference on New Interfaces for Musical Expression* (Oslo, Norway, 2011), Jensenius A. R., Tveit A., Godøy R. I., Overholt D., (Eds.), pp. 240–243. 3

[HKK08] HOFER R., KAPLAN P., KUNZ A.: Mighty trace: Multiuser technology on lcds. In *Proc. CHI '08* (2008), pp. 215–218. 3, 4

[HNK09] HOFER R., NESCHER T., KUNZ A.: Qualitrack: Highspeed tui tracking for tabletop applications. In *Human-Computer Interactionâ INTERACT 2009*, Gross T., Gulliksen J., Kotz P., Oestreicher L., Palanque P., Prates R., Winckler M., (Eds.), vol. 5727 of *Lecture Notes in Computer Science*. Springer Berlin, Heidelberg, 2009, pp. 332–335. 3, 4

[Hof11] HOFER R.: *Tracking technologies for interactive tabletop surfaces*. PHD Thesis, ETH Zurich, 2011. 3, 4

[IMH05] IGARASHI T., MOSCOVICH T., HUGHES J.: As-rigid-as-possible shape manipulation. *ACM Trans. Graph. 24*, 3 (July 2005), 1134–1141. 2

[IU97] ISHII H., ULLMER B.: Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proc. CHI '97* (1997), pp. 234–241. 3

[KF10] KUNZ A., FJELD M.: From table-system to tabletop: Integrating technology into interactive surfaces. Human-Computer Interaction Series. 2010, pp. 53–72. 1, 3

[SHB*10] SCHOENING J., HOOK J., BARTINDALE T., SCHMIDT D., OLIVER P., ECHTLER F., MOTAMEDI N., BRANDL P., ZADOW U.: *Building Interactive Multi-touch Surfaces, Tabletops - Horizontal Interactive Displays*. Springer, 2010. 4

[SJG*06] SHAHROKNI A., JENARO J., GUSTAFSSON T., VINNBERG A., SANDSJO A., FJELD M.: One-dimensional force feedback slider: going from an analogue to a digital platform. In *Proc. NordiCHI '06* (2006), pp. 435–456. 2

[SR09] SHEN C., RYALL K. E. A.: Collaborative tabletop research and evaluation: Interfaces and interactions for direct-touch horizontal surfaces. In *Interactive artifacts and furniture supporting collaborative work and learning*, Dillenbourg P., Huang J., Cherubini M., (Eds.). Springer Science and Business Media, New York, Oslo, Norway, 2009, pp. 111–128. 3

[WWJ*06] WEISS M., WAGNER J., JANSEN Y., JENNINGS R., KHOSHABEH R., HOLLAN J., BORCHERS J.: Slap widgets: bridging the gap between virtual and physical controls on tabletops. In *Proc. CHI '09* (2006), pp. 481–490. 2

Visualization in ViSuCity, a tool for sustainable city planning

Yifang Ban[1], Pontus Jakobsson[2], Lars Kjelldahl[3] and Ulf Ranhagen[4]

[1]Geoinformatics, KTH (yifang@kth.se)
[2]Sightline (pontus.jakobsson@sightline.se)
[3]CSC, KTH (lassekj@csc.kth.se)
[4]Urban planning, KTH (ulf.ranhagen@sweco.se)

Abstract

This paper gives an overview of several aspects of visualization for city planning as they were used in the project ViSuCity. The overall objective of ViSuCity is to develop an effective web-based, interactive visualization demonstrator, ViSuCity, to support sustainable city planning in terms of information sharing, analysis, development, presentation and communication of ideas and proposals throughout the city planning processes. In this paper, we discuss and show some results regarding LOD, scalability, streaming, and examples of visualization of roads, etc that are important for city planning.

Categories and Subject Descriptors (according to ACM CCS): I.3.8 [Computer Graphics]: Applications—

Keywords: *visualization, computer graphics, LOD, urban planning, city planning, streaming, scalability*

1. Introduction

ViSuCity, a Visual Sustainable City Planning Tool, is a project that was jointly funded by the Knowledge Foundation, Swedish Foundation for Strategic Research, Vinnova, etc. in 2008 through its first call for Visualization Demonstrators (Heavy). The project consortium has many partners from academia (KTH Geoinformatics, Urban Planning and Computer science and communication), industry (Sightline, Sweco, Digpro, Blom) and community (Stockholm City Planning Administration, Stockholm County's Regional Planning & Transport Office and the Swedish Road Administration).

The overall objective of this research was to develop an effective web-based, interactive visualization demonstrator, ViSuCity, to support sustainable city planning in terms of information sharing, analysis, development, presentation and communication of ideas and proposals throughout the city planning processes. Parameters relevant for an integrated sustainable city planning, such as transportation system, infrastructure, energy, water and waste management, green structure, etc will be integrated to enhance the quality of both the planning process and the planning results. A short overview of the project can also be found in [ViS].

2. Research questions in the project ViSuCity

The research and development of the ViSuCity project was focused on the following eight research issues:

- How can a visualization framework be used to enhance the quality of both the planning process and the planning result including parameters relevant for an integrated sustainable city planning, such as transportation system, technical infrastructure, energy supply and efficiency, water and waste management, green structure, etc.?
- How can we efficiently generate 3D models of the present & past (if possible) built environment using remotely sensed data and existing 2D data?
- How can GIS and BIM be integrated in the most efficient manner for planning?
- How can we refine spatial, object and web database technologies for effectively retrieving & managing 3D data & other spatial data related to sustainable planning?
- What is the most efficient way to present large volume of planning data to planners, decision-makers and public via internet?
- How can we implement easy-to-use tools such as multi-criteria evaluation (MCE) for analysis of sustainable planning alternatives? How to effectively visualize results of

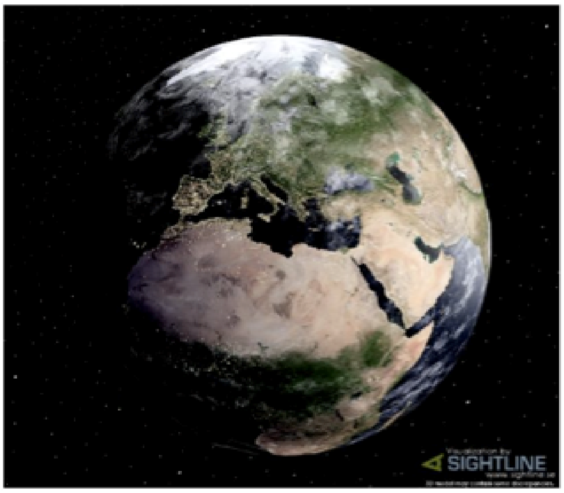


Figure 1: *The image is streamed and when coming closer more details are being presented.*

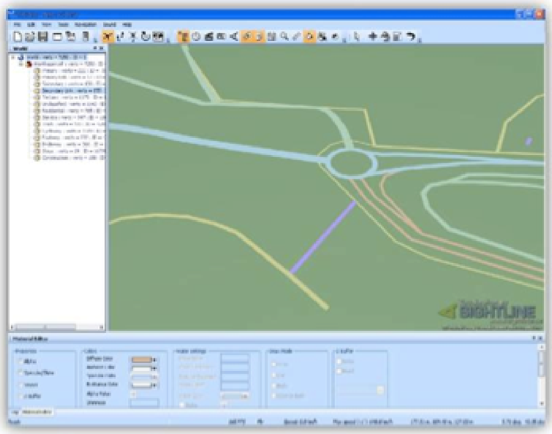


Figure 2: *Roads are being connected using different algorithms and techniques, here set up with the editor in the Neo4 system in ViSuCity.*

MCE and various sustainability indicators at different scales to support sustainable planning?

- How can we promote participations from different stakeholders and the public by giving users the ability, e.g., to a) easily discern, enlarge, turn and enlarge/diminish certain aspects within integrated structures, b) redline or edit certain features in a planning scenario?
- How can interdisciplinary collaboration among experts and other stakeholders within and between the private and public sector be strengthened by new visualization tools in order to enhance the quality of the planning process and the quality of planning proposals with regard to sustainable urban development? Is the demonstrator effective and easy to use? How can the usability successively be followed up and analyzed in the chosen case studies?

3. Methodology and implementation

The research questions in this project have been treated with different methodologies as it is a very interdisciplinary project. It would not be possible to go into detail on methodology for all the research questions and implementation issues in the project. We will in this short paper focus on a few questions, mainly with relations to visualization. Available tools were investigated in order to see how they could be used for the purpose of the project. Examples of tools and techniques investigated were OpenStreetMap, CityEngine, Generative Modeling Language (GML), Level of detail and streaming. The techniques were studied together with the software system Neo4. More details on the project can be found in [ViS10].

4. Visualization details

Here we will highlight a few of the visualization techniques that have been used and investigated in the project.

Streaming: On a computer, the user can interactively display a city plan, in ViSuCity, in less graphical complexity and have a shorter visibility distance of the city plan in full detail. When the user moves about in the 3D city plan details starts to show up as the user comes closer. These details loaded in a data stream either from the local hard drive or streamed over the internet directly in relation to how the user is acting. As the performance of computers grows, the visibility distance can be greater and greater meaning that more data in full quality can be presented at the same time, thus the visibility distance is one parameter that ViSuCity can control to adopt to the capability of the computer. This presents the two fundamental benefits with interactive realtime streaming of 3D content, one is downloading content and presenting on demand of the users actions in navigation the city plan, and in the other end the data to be streamed, the amount is determined by how much the computer is capable to present. In a master thesis project streaming is used to show various levels of details [Tho10], as illustrated in a global view in Figure 1.

Scalability: In the project data about rendering quality has been collected and can be used to adjust the quality according to the computer and the need of the user, who can set the quality in a dialogue (set to High, Medium or Low).

Road visualization: to connect roads in crossings turned out to be a non trivial problem and different options and algorithms were investigated in order to achieve a good visual result also when angles between roads meet in angles close to 180 degrees [Bar10]. An example of road connections is shown in Figure 2.

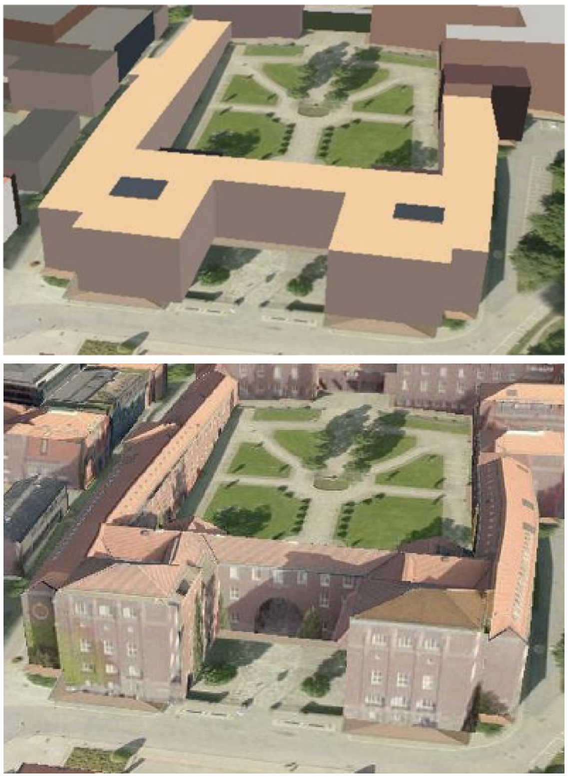


Figure 3: *Two 3D models in different level of details from Blom's production environment used in the project.*

LOD: level of detail is a common technique for adjusting the presentation in a balance between quality and computational speed. This technique has been used in ViSuCity. An example can be found in Figure 3.

Compression: The need for compression is a key feature for many graphics applications both for hardware oriented systems and for software systems that need extensive storage for the data in urban planning. Wavelet based point compression has been investigated in the ViSuCity project [WQ09], see Figure 4.

5. Visualization in context of ViSuCity

The ViSuCity project includes many research questions as described above. In this paper we focus on a few of the questions that are related to visualization and graphics although it is hard to extract visualization issues without taking other aspects into consideration. The purpose of visualization in ViSuCity is to help the user to take the decisions needed for the city planning process. It is also important to realize the very complex process for city planning including many different users such as politicians, experts, citizens, representing different spatial and/or planning levels from the national and regional scale to the detailed scale including city district level, block level, building level, apartment level and equipment level. Included are also different aspects such as planning of landscape, public space, transportation, waste, water, energy, social and economic aspects. An overview of the process used can be found in Figure 5. For many of the stages in the figure appropriate visualizations are needed.

Figure 4: *Automatic grouping of a LiDAR point cloud into objects with distinct height differences. These objects are then classified into terrain, buildings and trees.*

It can also be important to use tools for visualization together with tools for evaluation. The Figure 6 is intended to illustrate this fact.

6. Some examples of results

A final report from the project is given in [ViS10]. Examples of reports and papers from the project are given in [Alm10, Bar10, MB11, Mao10, Ols09, Par10, Poo09, Ran, She, Tho10, WQ09].

One result of the project was a demonstrator. The demonstrator's key features and capabilities are summarized as follows:

- Flexible tools to efficiently import existing 3D models and deploy available GIS and other data to generate a 3D model of the actual and current situation that relies on open standards for data exchange (IFC, GML, CityGML).
- Able to easily import and manipulate alternative volume studies and detailed studies created by architects and engineers in this interactive 3D environment. - Able to import and visualize traffic and transportation, technical infrastructure, environmental parameters other than buildings and adopt it to the visualization.
- Multi-criteria evaluation tools (see Figure 7) to analyze certain aspects of urban planning alternatives with regard to sustainability for example accessibility to public transportation, energy, water and waste flows, microclimatic

	Urban Development Process			
Development achievements	**Programming**	**Projection**	**Implementation**	**Administration**
Examples where visualisation supports development of suggestions and solutions	Comparison between system alternatives for integrated water, sewage and waste management	Further development of chosen system alternative	Support for implementation – follow the building and construction process	Visualisation supporting the follow-up of the district's performance
Examples where visualisation supports evaluation and follow-up	Visualised multi criteria analysis linked to system alternatives	Follow-up model with indicators and target levels – hierarchical construction from object to region	Follow-up model with gap analysis target – performance during implementation	Follow-up model with gap analysis target – performance during administration stage
Programme development	Interconnecting base model for all input elements and multi criteria analysis	Base model with illustration possibilities, showing target levels and simulated measurements	Base model illustrating the implementation and showing the construction process and preliminary performance in different construction stages	Base model showing performance in for example three levels (not fulfilled, fulfilled, highly fulfilled

Figure 5: *Different stages in the urban development process in ViSuCity.*

conditions, air pollution and noise etc by innovatively visualize information in 3D,4D and 5D. This will facilitate the weighing of criteria as well as the evaluation of alternatives of these criteria as important part of the MCE procedure.
- Tools to aid decision support, alternative studies, redlining, commenting, analysis of consequences, etc.
- Possibilities to analyze effects and consequences of physical plans with regard to all these aspects combined.
- Able to adapt to the different stages of the planning process, for local computer work and web-based presentation.
- Able to deploy the city plan project as an interactive 3D presentation to be experienced on the web in an easy to use platform.

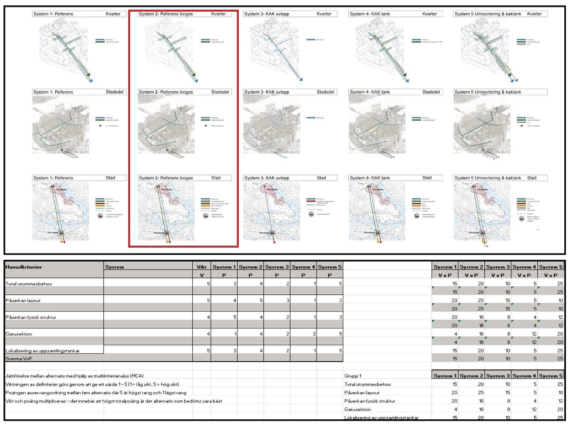


Figure 6: *Different alternatives are visualized and the table gives evaluation criteria.*

An example from the prototype of ViSuCity can be found in Figure 7.

7. Discussion and conclusions

The need for visualization is obvious in a project for city planning. There are several different aspects that have been met and investigated in ViSuCity. They include:

- pure visualization problems and techniques
- integration with evaluation criteria
- modeling and data collection to be used for visualization
- user oriented adjustments when designing the interface of the system
- sustainable aspects of the system

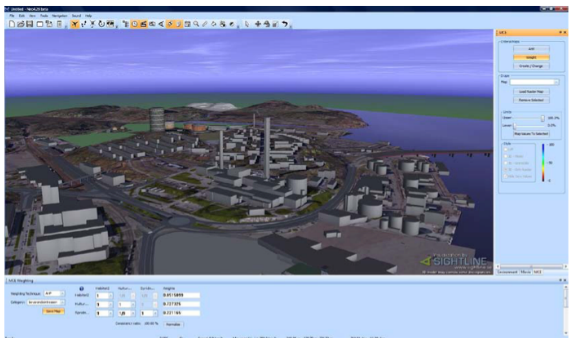


Figure 7: *Multi-Criteria Evaluation in ViSuCity.*

Visualization is a key issue in a city planning tool like ViSuCity and it is important to keep up with the fast evolution of the techniques both from a hardware point of view and from a software and algorithmic point of view. In spite of this it is clear that the integration with other aspects in the system is also a complicated problem. Many challenges remain to be solved and treated when designing and building tools for sustainable city planning. They include such things as to make the visualizations more accessible for non experts.

Acknowledgements

We would like to express our gratitude for the support of this project by the Knowledge Foundation, Swedish Foundation for Strategic Research, Vinnova, The Swedish Foundation for Health Care Sciences and Allergy Research, and Invest in Sweden. We would also like to thank all project partners for their contributions to the project.

References

[Alm10] ALMLÖF E.: Vad är viktigast i staden? M.Sc. thesis, supervisor: Mats Dunkars, 2010. 3

[Bar10] BARKDAL K.: Generating infrastructure in 3D for the planning of cities. supervisors: Lars Kjelldahl, Pontus Jacobsson, 2010. 2, 3

[Mao10] MAO B.: Visualisation and generalization of 3D city models, licentiate thesis in Geoinformatics, KTH, 2010. 3

[MB11] MAO B., BAN Y.: Visualization of 3D city model through the Internet using CityGML and X3DOM. *Cartographica 46*, 2 (2011). 3

[Ols09] OLSSON E.: Exporting 3D Geoinformation from Baggis Database to CityGML, supervisors: Peter Axelsson, Digpro & Yifang Ban, KTH Geoinformatics, 2009. 3

[Par10] PARSANEZHAD P.: Towards an Optimal Web-based Visualization Tool. A Comparative Analytical Survey over Visualization Techniques for Enhancing Stakeholders Participation in Planning. supervisor: Ulf Ranhagen, 2010. 3

[Poo09] POON C.-H.: Interaktiv Multikriteri-Analys, M.Sc. thesis, supervisors: Mats Dunkars, Yifang Ban, 2009. 3

[Ran] RANHAGEN U.: the SymbioCity Approach. Conceptual framework for support to sustainable urban development in low and middle income countries. Government Offices of Sweden. Sida, 2007 and 2010. 3

[She] SHENG N.: Modelling and Visualization of environmental and ecological indicators (pollution, noise, etc.), Supervisor: Yifang Ban, KTH, to be completed in April, 2011. 3

[Tho10] THORDARSSON F.: Streaming av komplexa urbana 3D-visualiseringar. Supervisor: Lars Kjelldahl, Pontus Jacobsson, 2010. 2, 3

[ViS] ViSuCity—A Visual Sustainable City Planning Tool, http://www.infra.kth.se/GIS/research/projects/ViSuCity/. 1

[ViS10] ViSuCity: A Visual Sustainable City Planning Tool, final report, DnR 2007/0410, November 2010. 2, 3

[WQ09] WIMAN H., QIN Y.: Fast Compression and Access of LiDAR Point Clouds Using Wavelets, Urban Remote Sensing Joint Event, Shanghai, China, 2009. 3

Posters

Augmented Reality-based Industrial Robot Control

B. Akan , A. Ameri E. and B. Çürüklü

Mälardalens Högskola, Box 883, 721 23 Västeras, Sweden

Abstract

Most of the interfaces which are designed to control or program industrial robots are complex and require special training for the user. This complexity alongside the changing environment of small medium enterprises (SMEs) has lead to absence of robots from SMEs. The costs of (re)programming the robots and (re)training the robot users exceed initial costs of installation. In order to solve this shortcoming, we propose a new interface which uses augmented reality (AR) and multimodal human-robot interaction. We show that such an approach allows easier manipulation of robots at industrial environments.

Categories and Subject Descriptors (according to ACM CCS): H.5.1 [Computer Graphics]: Multimedia Information Systems—Artificial, augmented, and virtual realities

1. Introduction

Almost all of the mass-market production plants use robots for different tasks such as welding, palletizing and tending. Although the benefits of using robots in industrial environments is well-known and the cost of robot hardware has decreased, still there are not many small medium enterprises (SMEs) that use them. One of the reasons lies with the (re)programming costs of such robots. SMEs have more dynamic production line, therefore changes in robot tasks will add extra costs for (re)programming them. If industrial robots could be easily programmed by the engineers that work in the production line, they might be more useful in the SME sector. With this sense, we aim to develop an easier interface for interacting with industrial robots.

Traditional way of programming industrial robots involves the use of teach pendant. Programmers use it to move the robot's tool center point (TCP) through the desired points. This approach has several shortcomings: (i) Jogging an industrial robot with 6 degrees of freedom with a joystick with three degrees of freedom is very time consuming and cumbersome; (ii) the operator doesn't get any visual feedback of the process result before the program has been generated and executed by the robot; (iii) many iterations are needed for even the simplest task [PPS*03]. Although 6DOF joysticks are available [SF09], they are not industry standards yet.

New robot programming interfaces should address these shortcoming. A visual feedback of what the robot is viewing and what it is (and will be) doing, can help the operator to understand robot actions, specially with complex tasks. This visual feedback can be in the form of virtual and/or augmented reality and addition of an object-based interface to it, will allow the user to define robot movements based on the objects. This approach would speed up the programming phase of the industrial robot. Our proposed system uses an AR interface on a tablet device. Manipulation is done at physical object level. The user can issue high level commands such as pick or put by clicking on virtual objects in the augmented display, therefore removing the necessity to jog the robot at a lower level. With this approach the user will get more and better feedback about the task and have better control of robot states. The path the robot will take can be virtually drawn and the robot can be virtually simulated removing the necessity of doing several iterations by the robot. Furthermore, it can provide usability of previously learned/taught skills at higher levels.

Augmented reality (AR) is a term used for overlaying computer generated graphics, text and three dimensional (3D) models over real video stream. Virtual information is embedded into the real world, thereby augmenting the real scene with additional information. Augmented reality proved to be useful in several industrial cases, for visualizations. Regenbrecht et al. [RBW05] have applied AR

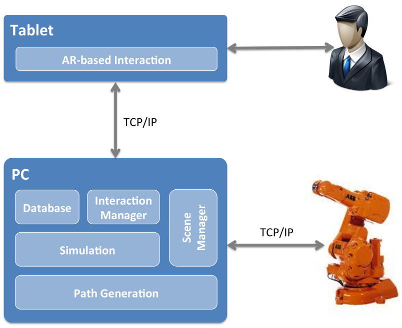


Figure 1: *Show the block diagram of the whole system, including the tablet device, PC server and the robot controller.*

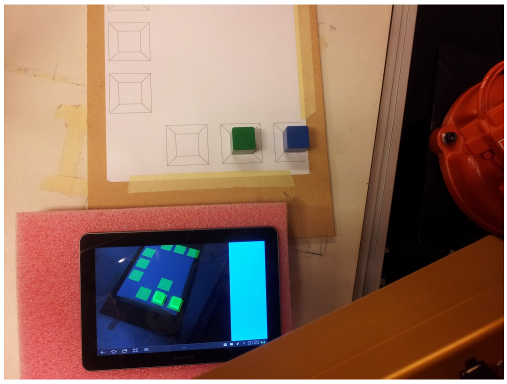


Figure 2: *Shows a screenshot of the system alongside the real workbench.*

in different automotive and aerospace industrial scenarios successfully. Their field of research included applying AR technology to service/maintenance, design/development and training. However AR can be very beneficial for programming industrial robots as well, whether it is remote or local.

2. Architecture

The proposed system consists of three physical parts; a tablet device, a standard PC server and the robot controller and the physical robot itself (See fig. 1). The tablet device works as the user interface to the system. It includes an augmented reality interface that shows the robot's view of the workspace. Users can interact with the virtual objects and the input from the tablet is transferred to the PC for processing and then the processed results are transferred back to the tablet device as output to the user. If the user accepts the current program then the PC side server compiles the robot program into RAPID language and sends it to the robot controller to be executed.

2.1. Tablet device

For this project, a tablet device with 10.1 inch screen that can handle multi-touch is used. The tablet runs on Android operating system. Since the computational capabilities of these devices are limited, most of the work needs to be done else where, therefore the tablet side user interface is designed as a thin-client. The tablet is only responsible for drawing the objects and getting the user input. PC server transfers an image of the robot's view alongside information about the objects to the tablet. The tablet then draws the objects as overlays on the received image, which generates and augmented reality view for the user. These objects can be selected by the user for various commands. The command and target objects are then send to the server for processing. Results of the process will be sent back to the tablet in the form of new object locations and paths that robot's TCP will take to perform them. User can then confirm the actions which in turn will be performed by the robot.

2.2. PC server

The PC server is responsible for all computations. It contains a simulation environment which can calculate forward and inverse kinematics of a general 6 Degree of Freedom (6DoF)industrial robot is used to verify the reachability of the target objects and collision between objects. It also acts as a visual feedback to user displaying the intentions of the robot [AAcA11].

3. Conclusion

In this work in progress paper, an augmented reality interface which runs on a mobile tablet device has been partially developed. However, this is still on going work. This proposed interface can help to overcome some of the problems introduced through using of flex pendant.

References

[AAcA11] AKAN B., AMERI A., ÇÜRÜKLÜ B., ASPLUND L.: Intuitive Industrial Robot Programming Through Incremental Multimodal Language and Augmented Reality. In *Proceedings of the IEEE ICRA 2011* (2011). 2

[PPS*03] PETTERSEN T., PRETLOVE J., SKOURUP C., ENGEDAL T., LKSTAD T.: Augmented reality for programming industrial robots. In *Proceedings of the 2nd IEEE/ACM ISMAR* (Washington, DC, USA, 2003), ISMAR '03, IEEE Computer Society, pp. 319–. 1

[RBW05] REGENBRECHT H., BARATOFF G., WILKE W.: Augmented reality projects in the automotive and aerospace industries. *Computer Graphics and Applications, IEEE 25*, 6 (nov.-dec. 2005), 48 – 56. 1

[SF09] SUNDIN M., FJELD M.: Softly Elastic 6 DOF Input. *International Journal of Human-Computer Interaction 25*, 7 (Sept. 2009), 647–691. 1