Joint Virtual Reality Conference of EuroVR - EGVE - VEC (2010)
T. Kuhlen, S. Coquillart, and V. Interrante (Editors)

Automated Positioning of Annotations in Immersive Virtual Environments

S. Pick, B. Hentschel, I. Tedjo-Palczynski, M. Wolter, and T. Kuhlen

Virtual Reality Group, RWTH Aachen University

Abstract

The visualization of scientific data sets can be enhanced by providing additional information that aids the data analysis process. This information is represented by so called annotations, which contain descriptive meta data about the underlying visualization. The meta data results from diverse sources like previous analysis sessions (e.g. ideas, comments, or sketches) or automated meta data extraction (e.g. descriptive statistics). Visually integrating annotations into an existing data visualization while maintaining easy data access and a clear overview over all visible annotations is a non-trivial task. Several automated annotation positioning algorithms have been proposed that specifically target single-screen display systems and hence cannot be applied to immersive multi-screen display systems commonly used in Virtual Reality. In this paper, we propose a new automated annotation positioning algorithm specifically designed for such display systems. Our algorithm is based on an analogy to the well-known shadow volume technique, which is used to determine occlusion relations. A force-based approach is used to update annotation positions. The whole algorithm is independent of the specific annotation contents and considers well-established quality criteria to build an annotation layout. We evaluate our algorithm by means of performance measurements and a structured expert walkthrough.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual Reality I.7 [Document and Text Processing]: Document Preparation - Multi/mixed media—

1. Introduction

One of the major application areas of immersive Virtual Reality (IVR) systems is the interactive exploration of complex simulation data. In order to aid the comprehension of the simulation data, its visualization can be enriched by displaying annotations [BNC$^+$03]. These annotations may contain textual comments or pictorial sketches, typically generated during previous exploration sessions [SBM92].

Problems arise when visually integrating annotations into the existing data visualization. Naive approaches will lead to unfavorable annotation positions. Annotations will occlude data or be occluded by it or other annotations. Even if there existed no occlusions, an unstructured or cluttered annotation layout would make reading annotations or associating them to the data object they annotate difficult. In any case, there is a high propability that access to important information is severly hindered and consequently that information goes unnoted by the user.

One solution to this problem are automated annotation positioning algorithms. To solve the aforementioned problems, these algorithms have to respect certain requirements [HGAS05] with another one arising in the case of IVR systems. Overall five major requirements can be identified:

1. **Adapt to dynamic changes.** Position calculations have to happen in real-time.
2. **Sustain data access.** Prevent or resolve occlusions.
3. **Ease annotation-data association.** Position annotations close to their targets and prevent annotation clutter.
4. **Ensure visual continuity.** The connection between successive annotation layouts needs to be clear.
5. **Consider IVR systems.** The algorithm needs to be applicable to immersive multi-screen display systems.

There exist a series of algorithms that address the first four

DOI: 10.2312/EGVE/JVRC10/001-008

requirements. Unfortunately, they cannot be applied to immersive multi-screen display systems because they tightly bind their calculations to the image plane of a single display. For this reason we propose a new annotation positioning algorithm in this paper that addresses all of the above requirements. Our main contribution is a display-system-independent positioning algorithm applicable to immersive multi-screen display systems but also to 2D displays like desktop monitors. We focus on textual annotations here, but the algorithm can also handle pictorial annotations because it does not depend on any content-specific assumption.

The rest of the paper is structured as follows: section 2 contains an overview of important related work. In section 3 we describe our positioning algorithm. Section 4 contains the results of performance measurements and a structured expert walkthrough. Finally, we conclude our work in section 5 and also give a brief outlook on the next development steps.

2. Related Work

The search for an optimal annotation layout has been shown to be NP-hard in the case of a 2D search domain, like the image plane [MS91]. As a result, several heuristics to reduce the computational costs have been proposed. A similarity of almost all of these approaches is that they compute annotation positions based on the scene's image plane projection.

Preim et al. [PRS97] avoid occlusions by positioning annotations in columns to the left and right of the centered 3D scene. Bell et al. [BFH01] employ an efficient, approximative 2D view space management system to track empty view space regions and place annotations in close proximity to the objects they annotate. An overview over this and other algorithms for the application in Augmented Reality systems is given by Azuma et al. [AF03].

Hartmann et al. [HAS04] use a set of force-based, per-pixel metrics to evaluate the image plane and extract occlusion-free, optimized annotation positions. Connection line intersections are resolved by swapping the positions of the affected annotations. High computational costs keep the algorithm from running in real-time. The algorithm of Stein et al. [SD08] is also based on per-pixel metrics, integrating the resolution of connection line intersections into them.

Hartmann et al. [HGAS05] presented several effective layout styles which are put to use in an algorithm described by Ali et al. [AHS05]. Annotations are positioned outside of an approximating bounding silhouette extracted from the image plane projection. A spring-mass approach is used to evenly distribute annotations. Götzelmann et al. [GHS06] also extract a scene silhouette from the image plane projection but then employ per-pixel metrics to evaluate the empty exterior region. A greedy selection determines initial annotation positions from that. Positional updates are then performed using an agent-based approach. This agent-based approach is also used to achieve visual continuity between successive layouts, while other approaches rely on positional interpolation for that [BFH01, AHS05, SD08].

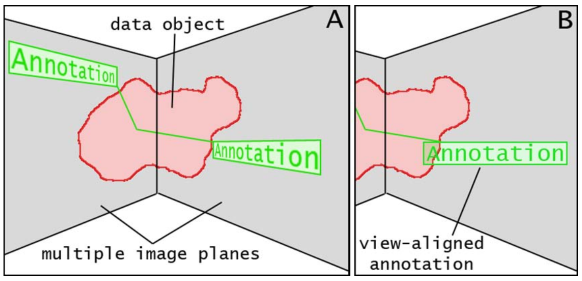

Figure 1: ***A*** *- An inconsistent layout due to per-image-plane application of 2D positioning algorithms (doubling artifacts). Also, the image-plane-aligned annotations make the image plane become apparent. This leads to a reduced degree of immersion.* ***B*** *- Re-aligning annotations cannot effectively counter the negative effects of image plane alignment.*

All these algorithms resolve occlusions and enhance the annotation layout. Unfortunately, they all position annotations in the 2D image space. Therefore, they cannot directly be applied to immersive multi-screen display systems, because this would require to separately calculate an annotation layout for the image plane of each screen. This will eventually produce inconsistent layouts among the different screens, leading to doubling artifacts (cf. fig. 1A). Moreover, placing annotations in the image plane negatively affects user immersion. This cannot easily be countered by re-aligning annotations (cf. fig. 1B).

3. The Annotation Positioning Algorithm

Unlike most positioning algorithms, our algorithm treats annotations as inherent part of the 3D scene. In contrast to existing image-space algorithms, this allows us to move annotations around in 3-space in order to resolve occlusions and to create a holistic annotation layout, which is completely independent from the utilized display system.

This approach requires to specify an initial position for every annotation. We choose the anchor point of the data object an annotation belongs to. An anchor point is determined by voxelizing a data object, extracting a medial line from this representation using a thinning algorithm [PK98], and then choosing the point on the medial line as anchor point, which is closest to the object's center of gravity (CoG).

Obviously, choosing anchor points as initial positions will eventually lead to occlusions. Nevertheless, as our algorithm can resolve these occlusions, we choose these positions because they are easy to compute.

The positioning system itself consists of two components. The first one is a scene representation (sec. 3.1). For every

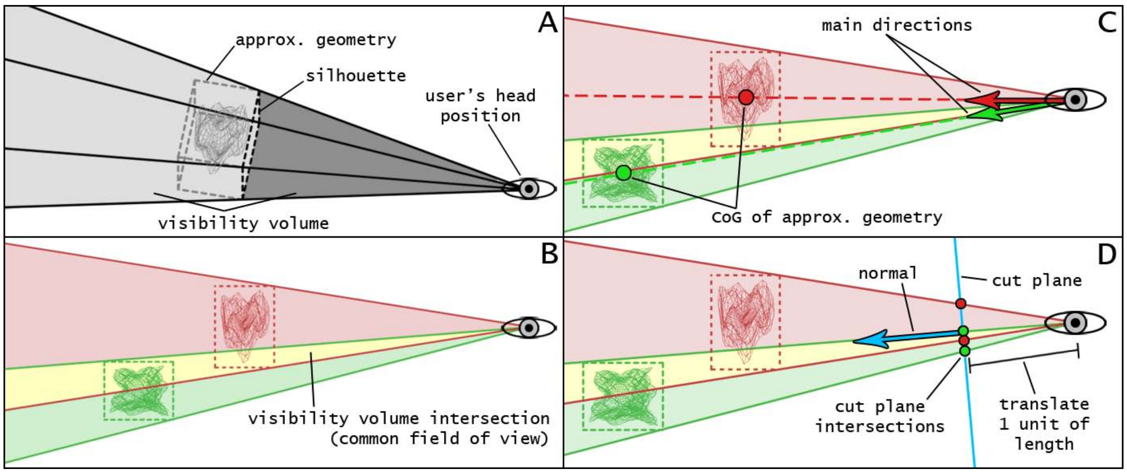

Figure 2: ***A*** *- Visibility volumes are derived from traditional shadow volumes (light grey) and the volume up to the object's silhouette (dark grey).* ***B*** *- The intersection of two visibility volumes indicates that the associated objects occupy the same field of view.* ***C*** *- The main direction of a visibility volume is defined by the vector pointing from the user's head position to the CoG of an object's approximating geometry.* ***D*** *- The average of two main directions defines the normal of a cut plane. Intersecting a visibility volume's edges with this cut plane produces a series of intersection points.*

object, it describes the sub-region of a user's field of view which is occupied by that object. Hence, it can be used to detect occlusions and evaluate the current annotation layout. The second component is an algorithm that automatically updates annotation positions to generate a layout which is geared towards the requirements from section 1 (sec. 3.2).

3.1. Scene Representation & Occlusion Detection

In the scene representation of our positioning algorithm every object, including the annotations, is expressed by a so called *visibility volume*. Visibility volumes are derived from classical shadow volumes [Cro77]. In contrast to them, a visibility volume does not only consist of the volume beginning at an object's silhouette. It also includes the volume between the point light and the object's silhouette (cf. fig. 2A). To compute the visibility volumes a point light is assumed to be placed at the user's head position. A visibility volume describes the set of rays that emanate from the user's head position and eventually hit their associated object. The idea is to interpret these light rays as view directions. Due to this interpretation, a visibility volume can be understood as a way to describe the region of the user's field of view that is occupied by a specific object. The advantage of using visibility volumes is that they enable an efficient occlusion detection independent of the underlying display system. Hence, they can be used with single- as well as multi-screen display systems without any adaption.

In order to efficiently compute visibility volumes we do not use an object's full geometry but rather resort to an approximating, convex bounding geometry. This reduces the computational cost of visibility volume generation as it directly depends on the complexity of an object's mesh. All examples shown in this paper use an object's axis-aligned boundings box. Besides this, using approximating geometries allows to handle data which has been specified using different representations, e.g. voxel data, using a single polygon-based visibility volume algorithm as the required polygon meshes are always available. We choose convex bounding geometries despite the resulting imprecision as we can take advantage of the convexity property in other stages of the algorithm.

The visibility volumes can directly be used to identify occlusions between arbitrary pairs of objects. For this it suffices to test whether two objects' visibility volumes intersect each other. An intersection would indicate that the underlying objects occupy a common region of the user's field of view with respect to their bounding geometries (cf. fig. 2B). Obviously, one of the bounding geometries is occluding the other in such a situation. The test for detecting visibility volume intersections consists of three steps. The main idea is to place a cut plane right in a pair of visibility volumes and to intersect them with this cut plane to extract a cut area for each shadow volume. In case the two cut areas have a common region, it follows that the associated visibility volumes intersect each other. The advantages of this intersection test lie in its simple implementation and the fact that its results can be intuitively integrated into the force-based positioning algorithm, which is described in section 3.2.

Step 1 - Cut plane determination - First, a so called *main direction* is calculated for each visibility volume (fig. 2C). The main direction for a visibility volume is extracted by simply calculating the direction vector from the user's head position to the CoG of the underlying object's approximating

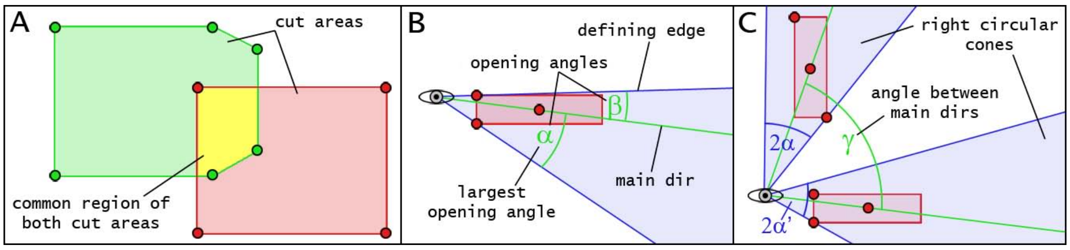

Figure 3: ***A*** *- The convex hull of all intersection points of a visibility volume's edges with the cut plane defines a cut area for that visibility volume (red/green). A pair of cut areas might share a common region (yellow) which indicates an occlusion.* ***B*** *- Using the main direction and the defining edges of a visibility volume, a series of opening angles can be determined.* ***C*** *- The largest opening angle can be used to approximate a visibility volume with a right circular cone. An efficient comparision of the cones can be used for speeding up the occlusion detection.*

geometry. Averaging the main directions yields a direction vector that is used as the normal for defining the required cut plane. The plane is placed at the user's head position and moved one unit of length into its normal's direction in order to be able to calculate intersections between the cut plane and the visibility volumes (fig. 2D).

Step 2 - Cut area calculation - The cut areas are determined by intersecting each visibility volume with the cut plane. For this, simple ray-plane intersections are evaluated while each ray is described by one of the defining edges of a visibility volume (fig. 2D). The convex hull of all intersection points belonging to a specific visibility volume constitutes the visibility volume's cut area (fig. 3A).

Step 3 - Determining the common cut area - The common region of the two cut areas from step 2 is determined by clipping both cut areas against each other. To perform the clipping we adapted the Liang-Barsky line clipping algorithm [Bar84]. It yields the silhouette of both cut areas' intersection. If no such silhouette was generated the two underlying objects do not occlude each other.

For the positioning of annotations every occlusion in which an annotation is participating is relevant. Thus, all pairs of visibility volumes, of which at least one belongs to an annotation, need to be submitted to the above intersection test. Assuming m annotations and n data objects being present in the scene, the complexity for detecting all of the aforementioned occlusions lies in $O(m \cdot (m+n))$. To reduce the impact of this in scenes with a high annotation count, we introduce a broad phase into our algorithm which efficiently checks for a pair of visibility volumes, if the above intersection test needs to be executed or not.

During the creation of a visibility volume a main direction was calculated for it as depicted in figure 2C. From this, a series of opening angles for a visibility volume can be calculated by determining the angles between the main direction and each of the visibility volume's defining edges (fig. 3B). Among these opening angles a largest angle α is identified and used to approximate the visibility volume by a right circular cone with infinite height and aperture 2α (fig. 3C). Performing an intersection test for a pair of view cones breaks down to an angle comparison. Assuming a pair of view cones with aperture 2α and $2\alpha'$, respectively, as well as an angle γ between their main directions, the full intersection test is performed if and only if $\alpha + \alpha' \geq \gamma$ holds.

3.2. Force-based Annotation Positioning

We employ a force-based approach for the automated positioning of annotations. Depending on the spatial configuration of all objects in the current frame, a series of forces affecting each annotation's movement is calculated. These forces aim at resolving occlusions and improving the overall annotation layout. Using the second Newtonian Law of Motion, $\vec{F} = m \cdot \vec{a}$, the acceleration of an annotation is determined. For the sake of simplicity we set the mass m to 1kg. The relation between acceleration and a position is given by

$$\vec{a} = \frac{\mathrm{d}\vec{v}}{\mathrm{d}t} = \frac{\mathrm{d}^2\vec{x}}{\mathrm{d}t^2}.$$

Therefore, in order to calculate positional updates for an annotation, its acceleration needs to be integrated twice over time. Since the acceleration of an annotation is only evaluated in variable but discrete time steps, i.e. each frame, we assume all time-dependent physical quantities to be constant in between frames. Hence, positional updates can be computed based on an explicit Euler scheme:

$$\begin{aligned}\vec{x}(t_i) &= \vec{x}(t_{i-1}) + \vec{v}(t_{i-1}) \cdot \Delta t + \frac{1}{2} \cdot \vec{a}(t_{i-1}) \cdot (\Delta t)^2 \\ \vec{v}(t_i) &= \vec{v}(t_{i-1}) + \vec{a}(t_{i-1}) \cdot \Delta t\end{aligned}$$

with $\Delta t = t_i - t_{i-1}$. Due to the above assumptions and the limitations of the used integration scheme the position calculations are rather imprecise from a numerical standpoint. Nevertheless, the approach is sufficient for designing a real-time positioning algorithm. A direct consequence of using a force-based incremental computation is that visual continuity is guaranteed among consecutive frames as demanded by requirement 4 defined in section 1.

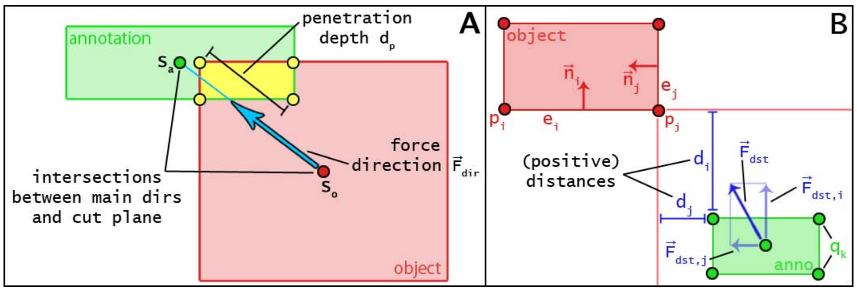

Figure 4: ***A*** *- The occlusion resolution force $\vec{F}_{occ}$ is calculated by multiplying the force direction $\vec{F}_{dir}$ with the force magnitude $f_m(d_p)$. The latter depends on the penetration depth d_p.* ***B*** *- A distance minimizing force $\vec{F}_{dst}$ is defined by a series of forces $\vec{F}_{dst,i}$. Each of these forces aims at reducing a distance d_i between the annotation's and the object's cut areas with respect to an edge e_i of the object's cut area.*

3.2.1. Occlusion Resolution

The occlusion-free placement of annotations is a major criterium for annotation layouts and expressed by requirement 2 from section 1. The force used to resolve occlusions is based on the results from the intersection test described in section 3.1. In case an intersection was found, a cut area configuration like the one in figure 4A has occured. In order to resolve the occlusion, the annotation has to be moved away from the involved object, so that the intersection of both cut areas is eliminated. This is achieved by applying a force $\vec{F}_{\text{occ}}$ to the annotation, which is defined by multiplying a force direction $\vec{F}_{\text{dir}}$ with a force magnitude $f_m(d_p)$:

$$\vec{F}_{\text{occ}} = f_m(d_p) \cdot \vec{F}_{\text{dir}}$$

To calculate the force direction $\vec{F}_{\text{dir}}$, the points resulting from intersecting the involved visibility volumes' main directions with the cut plane from the intersection test, $\mathbf{S}_a$ and $\mathbf{S}_o$, are required. $\vec{F}_{\text{dir}}$ is then defined as the direction vector pointing from $\mathbf{S}_o$ to $\mathbf{S}_a$ (fig. 4A).

To calculate a force magnitude f_m we first project the common region on the inverse force direction. This yields a scalar value which describes the penetration depth d_p of the annotation into the object with respect to the force direction $\vec{F}_{\text{dir}}$ (fig. 4A). The force magnitude f_m is then defined as a strictly increasing function of the penetration depth d_p.

Applying $\vec{F}_{\text{occ}}$ to the annotation obviously moves it away from the object eventually resolving the occlusion. Note that the above force can be applied to resolve the occlusion between two annotations. This is done by assuming one annotation to be fixed and calculating $\vec{F}_{\text{occ}}$ for the unfixed annotation. After that the two annotations swap roles and $\vec{F}_{\text{occ}}$ is calculated for the other annotation. Since the calculations happen in-frame they are order independent.

3.2.2. Placing Annotations Close To Their Target

Requirement 3 defined in section 1 states, that placing an annotation close to its target eases the annotation-data association. This requirement is addressed by a force $\vec{F}_{dst}$ which will be derived in the following.

As a data object's cut area is convex, it can be understood as the intersection of a number of hyperplanes. Every hyperplane corresponds to one edge e_i of the cut area and is defined by a normal $\vec{n}_i$ and a corner point $\mathbf{p}_i$ (fig. 4B). For an annotation's cut area, which consists of a set of corner points $\mathbf{q}_k$, a distance d_i to an edge e_i can be defined:

$$d_i = \min_k \{\text{dot}(\mathbf{p}_i - \mathbf{q}_k, \vec{n}_i)\}$$

Note that these distances can have a negative sign. In order to reduce the distance between an annotation and its data object all distances d_i greater 0 have to be minimized. The convexity property of cut areas makes it theoretically possible to reduce every distance d_i to a value not greater than 0 without introducing annotation-object intersections.

For every positive distance d_i a force $\vec{F}_{dst,i}$ that minimizes the distance d_i to the edge e_i is defined. Its direction $\vec{F}_{dir,i}$ is chosen to be equal to the normal $\vec{n}_i$. Its magnitude $f_{m,i}$ is chosen to be a strictly increasing function of d_i. Combining all forces $\vec{F}_{dst,i}$ yields the force $\vec{F}_{dst}$:

$$\vec{F}_{dst} = \sum_{i \in \{j : d_j > 0\}} \vec{F}_{dst,i} = \sum_{i \in \{j : d_j > 0\}} f_{m,i}(d_i) \cdot \vec{F}_{dir,i}$$

3.2.3. Fixing Annotation Sizes

To ensure the constant legibility of an annotation, its size must neither be too large nor too small. For that reason, we apply a force to an annotation that aims at keeping the annotation roughly in the same distance to the user.

The algorithm shoots a ray from the user's head position through the annotation position $\mathbf{P}$ (fig. 5A). Next, a point $\mathbf{T}$ on the ray in the required distance is calculated. A force $\vec{F}_{\text{size}}$ then pulls the annotation towards that point. The force's magnitude f_m is determined by the distance between the annotation position $\mathbf{P}$ and the point $\mathbf{T}$, using a strictly increas-

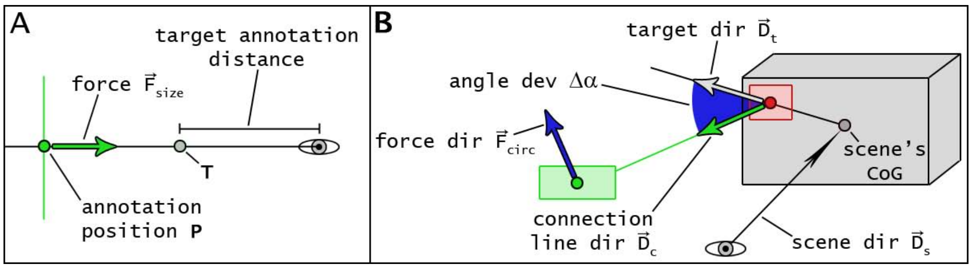

Figure 5: ***A*** *- A force $\vec{F}_{size}$ pulls an annotation towards a point* **T** *which is placed in a specific distance from the user in the direction of the annotation. Hereby, the perceived annotation size is kept roughly constant.* ***B*** *- A force $\vec{F}_{circ}$ is used to create a circular layout. The force minimizes the angle deviation $\Delta\alpha$ between the connection line direction $\vec{D}_c$ and a target direction $\vec{D}_t$. $\vec{D}_t$ encodes the annotation's radial position in the circular layout.*

ing function for f_m. The force $\vec{F}_{\text{size}}$ is defined by:

$$\vec{F}_{\text{size}} = f_m(||\mathbf{T}-\mathbf{P}||) \cdot \frac{\mathbf{T}-\mathbf{P}}{||\mathbf{T}-\mathbf{P}||}$$

3.2.4. Improving The Layout Structure

Arranging annotations in a circular layout around the centered scene has proven to yield a clear annotation layout [HGAS05] further addressing requirement 3 from section 1. Hence, we decided to include a force $\vec{F}_{circ}$ in our system which aims at generating such a layout.

In a first step we determine the direction $\vec{D}_s$ in which the scene is positioned from the user by shooting a ray from the user's head position through the CoG of the scene's bounding box (fig. 5B). The resulting direction vector and the scene's CoG are used to describe a plane. Next, one ray is shot through the center of each data object's anchor point originating in the scene's CoG. The directions of these rays are then projected onto the plane calculated in the previous step and renormalized afterwards resulting in a vector $\vec{D}_t$ (cf. fig. 5B). This direction vector describes the direction in which to place an annotation to yield a circular layout. Since this might lead to an uneven distribution of annotations around the scene due to similar direction vectors $\vec{D}_t$, we employ a system similar to the one from [AHS05] to spread the direction vectors more evenly and reduce annotation clutter.

Finally, in order to position the annotation in the direction $\vec{D}_t$, a radial force $\vec{F}_{circ}$ is applied to it, which minimizes the angle deviation $\Delta\alpha$ between the direction vector $\vec{D}_t$ and the annotation's connection line direction $\vec{D}_c$ (fig. 5B). The magnitude f_m of this force depends on the angle deviation $\Delta\alpha$ in terms of a strictly increasing function:

$$\vec{F}_{circ} = f_m(\Delta\alpha) \cdot \text{sign}[(\vec{D}_c \times \vec{D}_t) \cdot \vec{D}_s] \cdot (\vec{D}_s \times \vec{D}_c)$$

One situation that commonly arises in exploration using IVR systems is, that the user is navigating right into the scene. Hence, the scene is not lying in one specific direction from the user but encloses her. In this case the above calculations can not be performed. Hence, we deactivate the calculation of this force from the moment the user enters the scene's bounding box. It should be noted that a circular layout is not defined in this situation.

3.2.5. Reducing Oscillating Annotation Movements

One situation that may occur using only the above forces is that an annotation starts to oscillate. For example, if an annotation is moved by the force from section 3.2.3, it will eventually overshoot the target distance and has to be moved back by the same force to yet again overshoot the target distance. To prevent these effects, an additional damping force is introduced into the system, which is based on the concept of Stokes friction. Stokes friction models the friction exerted to spherical objects in a viscous fluid:

$$\vec{F}_{fric} = 6 \cdot \pi \cdot r \cdot \eta \cdot \vec{v}$$

For simplicity we set the radius r to 1m. The value for the viscosity η was empirically determined to 0.5 Pa·s. It is large enough to prevent oscillations but small enough to let the annotation layout form rapidly. In addition, we deliberately neglect small velocities, of below 20 cm/s, in order to cancel out minor remaining movements.

Figure 6 shows results achieved by our algorithm on different IVR display system setups.

4. Evaluation

4.1. Performance Measurements

For the performance evaluation of our algorithm, we used a scene that was derived from a numerical flow simulation. Based on the λ_2 criterion [JH95], vortices have been extracted in a pre-process. This resulted in 14 data objects which could subsequently be used as annotation targets. In order to evaluate the runtime performance of our algorithm with respect to the number of annotations, we prepared six synthetic sets of annotations equally distributed over all 14 data objects. The sets consisted of $m \in \{10, 20, 40, 80, 120, 160\}$ annotations respectively. Additionally, a base-line measurement was performed, with no annotations present in the scene. To enforce continuous layout recomputations during each measurement run, the data

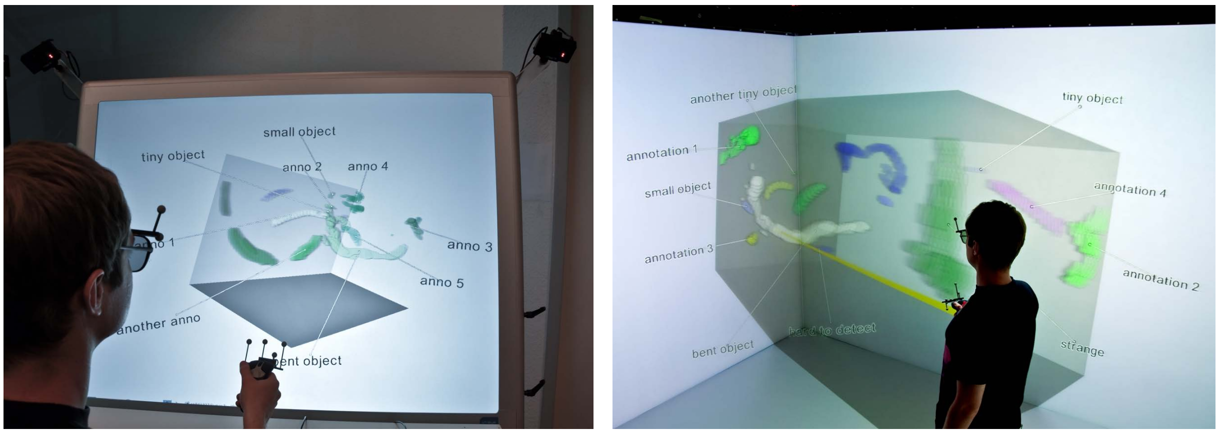

Figure 6: *Results achieved by our positioning algorithm on the semi-immersive, single-screen Imflip 150 [isi] (left) and the fully immersive, multi-screen CAVE (right). (For a colored version see the color plate section.)*

set was rotated 360° about its CoG using the global y-axis as rotation axis. The rotation lasted 5 seconds.

Figure 7 shows the results for two of our IVR systems. We performed measurements for an Imflip 150 Workstation [isi] in stereo mode (Intel Core 2 Quad Q6600, 4GB RAM, Quadro FX 4600, 1440×1050 pixels) and a 4-sided CAVE configuration (2x AMD Opteron 2218, 8GB RAM, Quadro FX 5600, 1600 × 1200 pixels on the front and floor & 1200×1200 pixels on the sides). A general observation is that both systems can sustain an interactive framerate of 30fps for up to between 20 to 40 annotations. Comments from the expert walkthrough, which will be covered in more detail in the next subsection, indicate that already 10 to 15 annotations introduce enough visual clutter to reduce the scene overview. Hence, we conclude that common scenes contain less than 20 annotations which is why our algorithm's performance suffices in general and fulfills requirement 1 from section 1. Note that, even though the complexity class of our algorithm was found to lie within $O(m^2)$ in section 3, the performance decrease is much more gentle in general. This is due to the efficient culling technique described earlier.

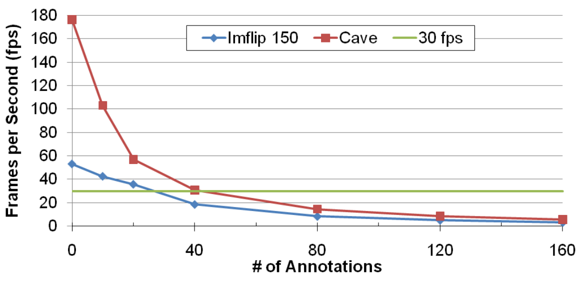

Figure 7: *Performance graph showing framerates for different numbers of annotations in the scene. Depending on the specific system, interactive framerates (>30fps) can be sustained for up to about 20 to 40 annotations.*

4.2. Structured Expert Walkthrough

For an initial evaluation of the quality of our positioning algorithm we performed a structured expert walkthrough, which was conducted using an Imflip 150 Workstation [isi] in stereo mode. We used the same data set for the walkthrough as for the performance measurements. Navigation within the data set was possible by using a tracked 6-Degrees-of-Freedom input device. As we were aiming for a general evaluation of the layout produced during arbitrary navigation, participants neither had to understand the specific composition of the data set nor the fluid mechanical background.

Ten subjects participated in the walkthrough. Participants were between 24 to 30 years old and all but 3 had experience using IVR systems. At the beginning of each test session the purpose of the annotation positioning algorithm was explained and a brief description of its workings was given. Next, the participants could freely navigate through the data set and familiarize themselves with the behaviour of the algorithm. Each participant was also encouraged to navigate into the midst of the simulation data and observe the annotation layout. At the end of the test session the participant was asked to fill out a questionnaire to qualitatively evaluate the produced annotation layouts. All items were phrased as statements and the participant could express her degree of agreement ranging from totally disagree (1) to fully agree (5). Participants were given the chance to further comment on the positioning results. Without giving any specific reason one participant did not complete the whole questionnaire. The partial results are not included in the evaluation.

Figure 8 shows the results from the walkthrough. Most participants agreed that annotations can easily be associated to their data object (S1, median 4). Sometimes annotations were pushed outside the view frustrum. Comments indicated that associating these annotations to their respective data object became more difficult because additional navigation was

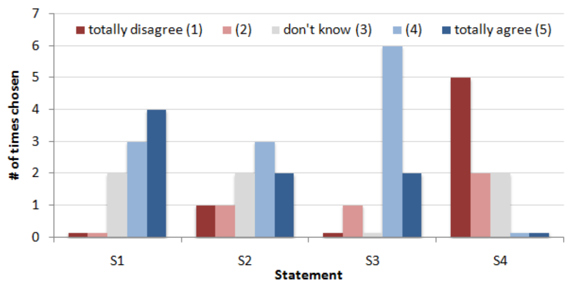

Figure 8: *Participants' degree of agreement. Statements:* ***S1*** *- Associating an annotation to its data object is easy.* ***S2*** *- Automatically calculated positions make annotations easily accessible.* ***S3*** *- The annotation overview is supported by the layout.* ***S4*** *- The movement of annotations is annoying.*

necessary to get them back into view. Some users stated that annotations were positioned too far away from their data objects. Annotation accessibility was evaluated quite diversly (S2, median 4). Again, annotations being outside the display had a negative effect. Nevertheless, most users agreed that the annotation layout supports the annotation overview (S3, median 4). One participant specifically commented on the layout as being too unstructured. The annotation movement was mostly perceived as not being annoying (S4, median 1).

The walkthrough showed that the annotation layout produced by our algorithm is supporting the overview over the annotations and generally well-structured. Nevertheless, issues with annotations leaving the view frustrum or being too far away from their data objects need to be addressed.

5. Discussion & Outlook

We have introduced a new, display-independent annotation positioning algorithm that is applicable to immersive multi-screen display systems used in VR. The algorithm respects established layout quality criteria in order to optimize annotation positions towards a well-structured, holistic annotation layout. Performance measurements have shown that the algorithm can handle an increased number of annotations while sustaining interactive frame rates. The results of a structured expert walkthrough have shown that the layout generated by our algorithm supports the annotation overview. Nevertheless, some issues have also been identified.

Future development will first focus on resolving the issues identified by the walkthrough. We also plan to experiment with tighter bounding geometries used for visibility volume generation as this will lead to more available space to position annotations. Annotations could move closer to their target, and thus, address an issue identified by the expert walkthrough. Additionally, we plan to perform a user study to evaluate the algorithm in a CAVE setup. Furthermore, as our implementation is currently only using a single CPU core, parallelizing it could further improve performance.

Acknowledgements

This work was funded in part by the German Research Foundation (DFG) through project WE 2186/5.

References

[AF03] R. Azuma and C. Furmanski. Evaluating Label Placement for Augmented Reality View Management. In *Proceedings of the IEEE/ACM International Symposium on Mixed and Augmented Reality*, page 66, 2003.

[AHS05] K. Ali, K. Hartmann, and T. Strothotte. Label Layout for Interactive 3D Illustrations. *Journal of the WSCG*, 13:1–8, 2005.

[Bar84] B.A. Barsky. A New Concept and Method for Line Clipping. *ACM Transactions on Graphics*, 3(1):1–22, 1984.

[BFH01] B. Bell, S. Feiner, and T. Höllerer. View Management for Virtual and Augmented Reality. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 101–110, 2001.

[BNC$^+$03] D. A. Bowman, C. North, J. Chen, N. F. Polys, P. S. Pyla, and U. Yilmaz. Information-Rich Virtual Environments: Theory, Tools, and Research Agenda. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, pages 81–90, 2003.

[Cro77] F. C. Crow. Shadow Algorithms for Computer Graphics. *SIGGRAPH Computer Graphics*, 11(2):242–248, 1977.

[GHS06] T. Götzelmann, K. Hartmann, and T. Strothotte. Agent-based Annotation of Interactive 3D Visualizations. In *Smart Graphics*, pages 24–35. Springer, 2006.

[HAS04] K. Hartmann, K. Ali, and T. Strothotte. Floating Labels: Applying Dynamic Potential Fields for Label Layout. In *Smart Graphics*, pages 101–113. Springer, 2004.

[HGAS05] K. Hartmann, T. Götzelmann, K. Ali, and T. Strothotte. Metrics for Functional and Aesthetic Label Layouts. In *Smart Graphics*, pages 115–126. Springer, 2005.

[isi] immersive systems (imsys). Imflip 150 Workstation - http://www.imsys-vr.de/128.0.html?l=1. last visited on: 5th of May, 2010.

[JH95] J. Jeong and F. Hussain. On The Identification of a Vortex. *Journal of Fluid Mechanics*, 285:69–94, 1995.

[MS91] J. Marks and S. Shieber. The Computational Complexity of Cartographic Label Placement. Technical report, Center for Research in Computing Technology, Harvard University, 1991.

[PK98] K. Palágyi and A. Kuba. A 3D 6-Subiteration Thinning Algorithm for Extracting Medial Lines. *Pattern Recognition Letters*, 19(7):613–627, 1998.

[PRS97] B. Preim, A. Raab, and T. Strothotte. Coherent Zooming of Illustrations with 3D-Graphics and Text. In *Proceedings of the Conference on Graphics Interface*, pages 105–113, 1997.

[SBM92] R.R. Springmeyer, M.M. Blattner, and N.L. Max. A Characterization of the Scientific Data Analysis Process. In *Proceedings of the Conference on Visualization*, pages 235–242, 1992.

[SD08] T. Stein and X. Décoret. Dynamic Label Placement for Improved Interactive Exploration. In *Proceedings of the International Symposium on Non-photorealistic Animation and Rendering*, pages 15–21, 2008.