Hedgehog Labeling: View Management Techniques for External Labels in 3D Space

Markus Tatzgern* Denis Kalkofen Raphael Grasset Dieter Schmalstieg

Graz University of Technology

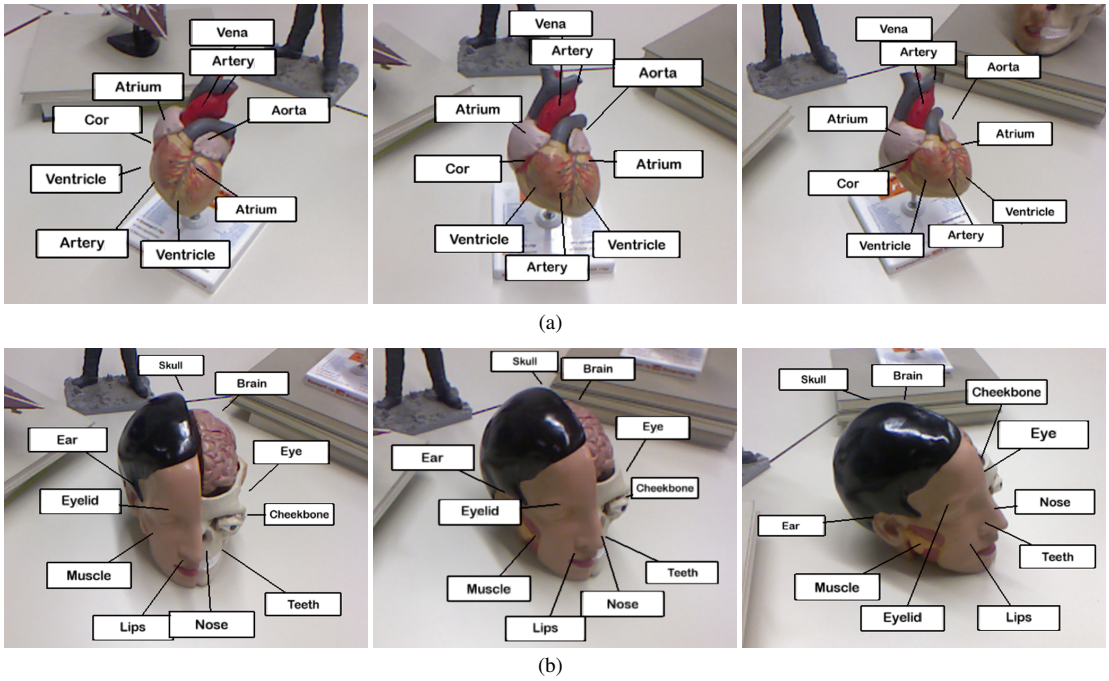


Figure 1: View Management in 3D space. (a) Label placement has been constrained by 3D poles which originate from the center of the object. To resolve occlusions, we move labels along the pole only. (b) Label placement has been constrained by a set of planes in 3D space. Labels are allowed to move within a plane, which is fixed in 3D space. To avoid constant label motion, the label positions are frozen after creating the layout for a viewpoint. The placement is updated only when the viewing angle to the plane grows larger than a threshold.

Abstract

Annotations of objects in 3D environments are commonly controlled using view management techniques. State-of-the-art view management strategies for external labels operate in 2D image space. This creates problems, because the 2D view of a 3D scene changes over time, and temporal behavior of elements in a 3D scene is not obvious in 2D image space. We propose managing the placement of external labels in 3D object space instead. We use 3D geometric constraints to achieve label placement that fulfills the desired objectives (e.g., avoiding overlapping labels), but also behaves consistently over time as the viewpoint changes. We propose two geometric constraints: a 3D pole constraint, where labels move along a 3D pole sticking out from the annotated object, and a plane constraint, where labels move in a dominant plane in the world. This formulation is compatible with standard optimization approaches for labeling, but overcomes the lack of temporal coherence.

*e-mail: tatzgern | kalkofen | grasset | schmalstieg@icg.tugraz.at

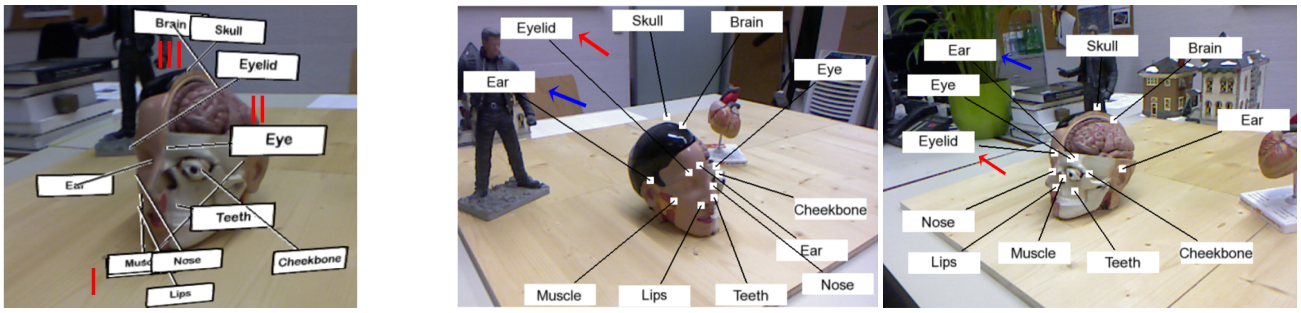
Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities;

1 Introduction

Augmented Reality (AR) systems present information directly within the real world environment. This allows to enrich a user's visual perception with a variety of annotations such as text, image or even video data. However, naively placing annotations generally leads to clutter and occlusion, impairing the effectiveness of AR visualization (Figure 2(a)).

In order to resolve clutter and occlusion issues, so-called *view management techniques* have been proposed [3]. These techniques automatically place annotations either directly on the surface of the object they refer to (using a so called *internal label*), or they place annotations outside the object of interest and draw a 2D line to its center (using a so called *external label*).

In traditional media, such as printed illustrations, an illustrator or graphics designer generally decides on the appropriate label type. The main factors influencing the selection are the available space, personal preferences and intuition. In AR, the choice of label type is also influenced by the registration error. Coelho et al. [5] demon-



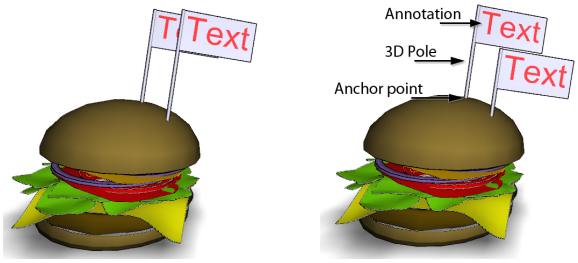
(a) (b)

Figure 2: Problems of presenting external annotations. (a) Without view management different types of occlusions may appear. (I) Annotations occluding each other. (II) Annotations occluding the object of interest. (III) Leader lines crossing each other. (b) When applying view management techniques, occlusions can be resolved but camera movements may cause unpredictable reordering of labels. Rotating the camera causes the labels marked with a red and blue arrow to change their order in y-direction.

strated that external labels are much less prone to disambiguation in imperfect AR environments than internal labels. We agree with this argument and focus in this paper on view management techniques for external labels.

A number of different techniques have been proposed to control the placement of external labels. They have been successfully applied to produce high quality layouts for desktop applications. However, since all of the existing techniques operate in 2D image space, they are prone to unpredictable changes over time. This happens because the distribution of the projected 3D points changes during camera movements, which can force the view management system to frequently re-order external in labels image space. With increasing amount of label movement and re-ordering, the label motion becomes difficult to follow, and the resulting layout becomes unstable over time.This is illustrated in Figure 2(b).

To overcome the problems of such floating labels, traditional desktop applications often display external annotations only when camera movement stops. However, in AR the camera is attached to the user, and thus it is always in motion.



(a) (b)

Figure 3: Illustration of view management in 3D space. (a) External labels occlude each other. (b) Instead of searching for an occlusion free layout in 2D image space, we adjust 3D properties of the label's geometry (in this case the length of the pole) to resolve occlusions between the annotations.

This paper contributes a new view management technique for external labels in 3D space. Since frame incoherent placement of labels is often caused by unpredictable changes of the projection of 3D points into the image space, we annotate the object of interest in 3D object space using external 3D labels. During camera movements, this strategy enables us to apply arbitrary 3D transformations not only to the 3D objects in the scene, but also to their labels. Since labels follow object transformations, our approach allows to better follow label movements over time.

To further support this objective, our view management approach applies changes to the layout based on the 3D geometry of the label. A 3D label consists of a 3D annotation, a 3D pole, and an anchor point. We only allow adjustments to the length of the pole, while the orientation of the pole is fixed in object space. Figure 3 illustrates this strategy, which resembles a "hedgehog". Its application to AR is shown in Figure 1(a).

While this approach produces aesthetic and stable layouts, it may stack labels which anchor points are located close to one another. To resolve stacked label layouts, we introduce layout strategies operating on a 3D object space approach. For example, Figure 1(b) shows the result of a balanced label distribution, which has been computed based on a set of planes in 3D space.

2 Related Work

Annotated objects appeared in the context of illustrations more than one hundred years ago [9]. However, the problem of placing labels was first discussed by cartographers [12] in the 70's. In the early 80's, computer graphics researchers started to develop algorithms that mimic manual label placement [1]. Automatic label placement for interactive graphics is still an active topic of research [6].

Hartmann et al. [11] outlined that artists use either internal or external labels in their illustrations. External labels present the information next to the object of interest, internal labels choose a position directly on the 3D object. Therefore, internal labels can be considered to operate in 3D object space and thus do not suffer from camera movements, if glued to the object of interest.

Algorithms for placing internal labels have been presented across several disciplines of computer graphics research, such as in volume visualization [17, 14], illustrative rendering [7, 15] and AR [22, 3]. While internal labels support frame coherent renderings, they require a certain amount of space to entirely fit on the object of interest [11]. Therefore, such approaches are usually limited to a rather small number of annotations. Coelho et al. [5] demonstrated that internal labels become ambiguous in AR when the registration error increases. This often makes them unsuitable for comprehensible information presentation in AR.

Approaches for automatic external label placement exist as well. In their the seminal work, Bell et al. [3] propose a system for external and internal label placement at run-time using empty space management in screen space. Since this system does not consider leader lines, it may suffer from crossing leader lines or leader lines occluding annotations.

Hartmann et al. [10] propose an image space approach, which evaluates a force field in every frame and updates the label layout accordingly. Their implementation considers leader line crossings and allows to align labels on non-rectangular shapes. This generates

high quality layouts in static scenes without camera motion. However, in dynamic situations the force field changes in every frame which causes labels to constantly move and often jump.

To overcome this problem, Tatzgern et al. [21] proposed an image space approach which freezes the layout as long as the camera is in motion. However, since their system labels the object of interest in image space, leader lines easily cross.

Frame coherent label placement has been discussed for internal and external labels for the special case of cyclic animations [8]. However, since this approach is limited to a static camera position, the technique does not support interactive AR applications.

Shibata et al. [19] demonstrate placement of labels in 3D. However, the view management approach operates in 2D image space. In contrast, our approach is operating entirely in 3D object space. Chigona et al. [4] is the approach closest to view management of external labels in 3D object space. The authors show annotations of the shadow of an object, which is projected to a single plane in 3D. This plane can be considered as an external 3D label. However, the system is limited to the points of interest which cast a shadow into this plane. Moreover, the extension of the shadow area restricts the amount of labels which can be placed, which is similar to the restrictions of internal labels.

In contrast to previous approaches, our system defines labels as elements of the 3D scene. This allows more expressive view management techniques in 3D object space. Labels stay stable during camera movement without imposing placement restrictions.

3 METHOD

Defining all elements of a label in 3D object space allows us to generate more predictable layout changes and thus less distracting label motion during camera movements. Therefore, we build an external label out of a *3D annotation*, a *3D pole* (leader line equivalent in 3D), and an *anchor point*. A 3D annotation can be a 3D or a 2D object (text, image) projected onto a billboard (Figure 3(b)).

Our view management approach consists of an *initialization* phase followed by an *update* phase. In the initialization phase, we place a 3D label for each element which we want to annotate. For each 3D label, we define the position of its anchor point, the orientation and length of its pole as well as the orientation of its annotation. After initializing all labels, we start updating the layout to resolve occlusions and to maintain readability. Based on the readability layout updates can be continuous or discrete.

3.1 Initializing the Layout

During initialization, we position each label in the 3d scene. The behavior of a label and thus the appearance of the layout at runtime depends on the design of a 3D label and the strategy to resolve occlusions. In this section, we discuss design decisions, such as where to place or in which direction to move a 3D label and which constraints this implies.

Orientation of 3D Annotation. The most common 3D annotation is a flat two-dimensional surface in 3D space, which is attached to the pole on one side. This configuration allows to rotate the surface around its pole only. While such 3D labels appear very natural, they easily suffer from perspective distortion, making the information unreadable from certain points of view. If the angle between the view vector and the 3D pole is high, a rotation of the annotation around the pole allows re-orienting the label so that its information can be easily read. However, this is not possible if the angle between the pole and the view vector is small. Such a configuration will cause the annotation to rotate almost around the user's view vector.

Since the orientation of the view vector can change arbitrarily at run-time, we cannot guarantee a sufficiently large angle between the view vector and a 3D pole. Therefore, we allow for unconstrained rotations of the annotation. Instead of attaching one side of the annotation to the upper part of the pole, we attach the tip of the pole to the center of the annotation. This allows to rotate the annotation around all three axis of its local coordinate system which is placed in the center of the annotation. During initialization, we orient annotations parallel to the screen.

Position of 3D Anchor Point. To easily link the annotation to the 3D object, we have to place the anchor point of the label on the 3D object. The most unambiguous position is its center, which we approximate using the center of its bounding sphere.

Length of 3D Pole. The pole has to be long enough so that the projection of the annotation is not covering the 3D object of interest. Yet we want the pole length to be minimized, so that annotated scenes are compact. Since we resolve occlusions after the initialization phase, we can just ignore the length of the pole during initialization by placing the annotation at its anchor point.

Orientation of 3D Pole. When placing an external label in 3D space, the most natural orientation of the pole follows the direction of the surface normal at its anchor point. However, as demonstrated in Figure 4(a), this strategy easily suffers from crossing leader lines after projecting to camera space. Notice the crossing between the pole of the label of the engine and the one of the right door of the car. As the camera rotates around the object, other leader lines cross in image space.

In order to avoid crossing leader lines, we orient 3D poles using the normalized vector which originates from the center of the object's bounding sphere and passes through the anchor point of the 3D label. This strategy is illustrated in Figure 4(b). Note that the leader lines do not intersect anymore.

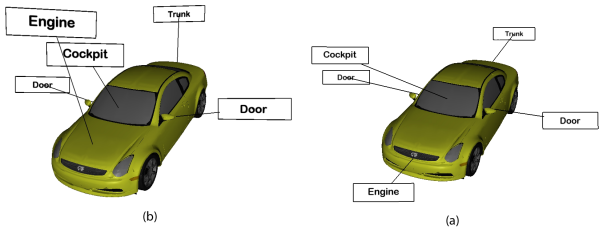


Figure 4: Normal vs. radial 3D pole orientation. (a) Orienting the pole using the face normal at the anchor point results in crossing poles in image space. (b) To avoid crossing poles, we orient 3D labels radially from the center of the bounding sphere of the object, i.e., aligning them with the vector pointing from the center of the bounding sphere to the anchor point of the label.

3.2 Updating the Layout

After initializing labels, or after the camera has moved, occlusions with other labels or scene objects may occur. Since finding the optimal place for all labels has been proven to be NP-hard [16], we formulate the problem of minimizing occlusions as a force-based optimization problem. Our approach is inspired by the image space approach of Hartmann et al. [10]. However, unlike Hartmann et al., we assume constrained motion in 3D space.

One Degree of Freedom. A simple constrained motion would allow every annotation to slide along the pole direction (one degree of freedom). Figure 5 demonstrates the result of this strategy. Notice how occlusions have been resolved for the two annotations, which have been marked with a red exclamation mark.

Three Degrees of Freedom. Resolving occlusions using a single degree of freedom only produces a small amount of very predictable motion. However, this strategy tends to stack annotations if poles have similar orientation in 3D space (see the upper image in Figure 6). To generate more balanced layouts, we additionally allow to move the annotation in the image plane (two additional

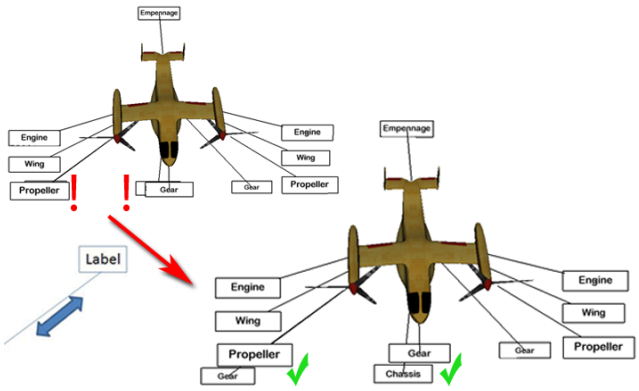


Figure 5: Resolving occlusions using one degree of freedom. To provide predictable movements, we allow moving the 3D label along the pole only. The upper image suffers from two occluded annotations, which we marked using a red exclamation mark. By extending the length of the pole, the system is able to resolves the occlusions.

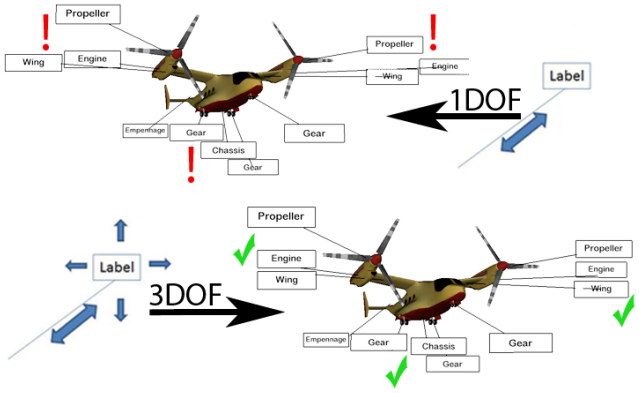


Figure 6: Resolving occlusions using three degrees of freedom. Stacked annotations appear for labels with poles oriented at a similar angle. Stacks have been marked in the upper image using red exclamation marks. To resolve stacks, we allow moving an annotation along the pole and within the X/Y plane of the annotations local coordinate system.

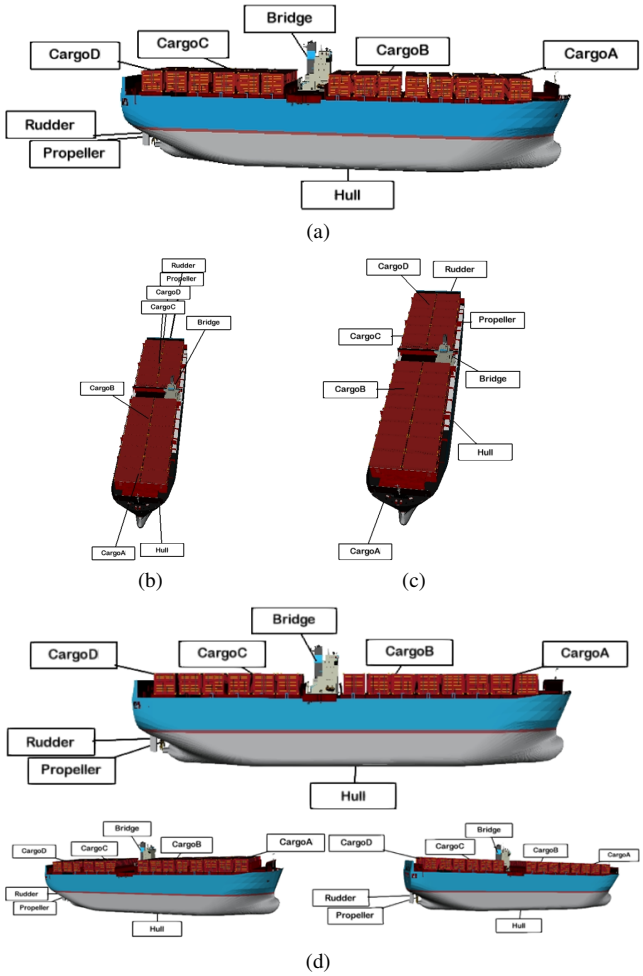


Figure 7: Plane-based occlusion management. The anchor points in this example have all been placed close to the x/y plane of the model. (a) The center-based labeling approach, which uses a common center to orient labels, generates a clear layout if the camera is oriented along the z-axis. (b) If the camera orientation is similar to the x- or the y-axis, the layout suffers from heavily stacked annotations. (c) Since our occlusion management approach is not able to resolve heavy stacking, we group annotations into 3D planes. This allows us to generate more balanced layout, while still providing stable layouts over time. In this example, we use three different planes, one in the front, one in the middle and one for anchor points in the back of the ship. (d) The plane-based label placement with planes generated from viewing along the z-axis.

degrees of freedom). However, to ensure that the pole is always connected to the annotation, we limit the motion in the image plane to the size of the annotation.

This strategy is illustrated in Figure 6. The label layout in the upper image suffers from a number of stacked labels. For example, the three annotations at the bottom (left gear, chassis and right gear) form a stack in y-direction. To resolve this stack, our system moves the annotation of the chassis in the image plane, until it fits next to the gear. Since the orientation of the pole for the right gear does not allow to place the annotation next to the gear, our system cannot resolve this stack. While a perfectly balanced layout may require more freedom of movement, this approach generates reasonably good results with a limited amount of predictable movements for a small set of stacked labels.

Plane-Based Occlusion Management. Our approach to stabilize the layout avoids re-orientations of 3D poles. However, this performs poorly, if many anchor points are in or close to a plane in 3D space. Such configurations lead to generate label poles inside a narrow slab and result in layouts suffering from stacked annotations, if rendered from a viewpoint perpendicular to these planes.

For example, most of the anchor points used to label the ship in Figure 7(a) have been placed close to the x/y plane of the coordinate system of the 3D CAD model. This causes most of the label poles to lie in the close proximity of the x/y plane. When looking at the object along the z-axis, the layout generated with our approach seems adequate (see Figure 7(a)). However, if the viewing direction becomes more similar to the x-axis, the 3D poles line up in image space, causing the algorithm to stack annotations, which cannot be resolved with the constraints we introduced.

To balance the layout, we have to relax our constraints. Therefore we group all labels into a set of planes and we allow searching for suitable positions for annotations within an entire plane. We orient the planes parallel to the view plane and we place them equidistantly in the screen aligned bounding box of the object. Each label is then automatically assigned to the plane that is closest to its

anchor point. This allows us to apply any image-based view management approach from the current point of view. Specifically we achieve balanced layouts using the spring embedding approach proposed by Ali et al. [2]. The result can be seen in Figure 7(c) and Figure 7(d). Both images have been rendered with the same groupings, but from different points of view using differently oriented planes. Note that the stacking of annotations has been resolved.

This approach generates more balanced layouts. However, if occlusions will be resolved in each frame, this approach introduces frame inconsistencies which are similar to those resulting from a force field approach in image space. Therefore, this approach is best applied in combination with a discrete update strategy.

To reduce the amount of motion for plane-based occlusion management we freeze the layout after resolving occlusions, and we update the annotations only if the angle between the view vector and the normal of the annotation's plane grows larger than a user-defined threshold. This behavior is demonstrated in Figure 7(d). The layout has been generated from the point of view in the upper image of Figure 7(d). Both renderings in the lower part of Figure 7(d) use the same layout from a slightly offset point of view.

While, we freeze the position of an annotation within a plane, we still update its orientations in every frame so that it always faces the camera. This approach works well for sufficiently distant labels within the plane, but it may cause occlusions with other annotations if the layout algorithm places them close to each other. For example, both Figure 8(b) and 8(c) use our plane-based occlusion management approach. However, because the annotations in 8(c) have been placed close to each other, occlusions between annotations appear during camera movements. We can enforce a certain spacing between annotations by adding a force to layout algorithm which maintains the spacing. However, with an increasing amount of annotations this approach pushes annotations further outwards.

Therefore, we support an additional discrete update strategy. Since annotations cannot occlude each other if they lie in the same plane, we also support freezing the orientation of annotations (Figure 8(c)). This allows us to avoid occlusions between annotations which lie in the same plane, however, this strategy introduces perspective distortions of annotations. Therefore, it should be used only if necessary and in combination with a small angle between layout updates.

4 Discussion

Figure 8 allows side-by-side comparison of center-based and plane-based label placement. Figure 8(a) uses the center-based approach. Even though we chose a rather spherical object, most anchor points are placed on one side of the object, namely in the face. Therefore, our center-based approach suffers from stacking of annotations and rather long 3D poles. For this configuration, the plane-based approach creates more appealing results. However, plane-based label placement requires to re-orient the label pole and thus should be used only if necessary.

When deciding to use a plane-based layout, one has to take into account that either occlusions between the labels occur, or perspective distortions of annotations. Occlusions occur, when only the position is frozen, but the annotation is always aligned to the screen (Figure 8(c)), perspective distortions appear when both position and orientation are frozen and annotations are always plane-aligned (Figure 8(d)). Since both approaches may impair the comprehensibility of the visualization, center-based 3D label placement should be considered if anchor points are well distributed around the object of interest (Figure 8(e)).

Assuming that labels in different planes can be clearly distinguished using parallax effects, we can further use plane-based label placement to reduce the amount of label fighting by resolving occlusions between labels only if they are grouped into the same plane. To reduce clutter from conflicting labels in the other planes, it may be sensible to allow for transparent labels.

To further enhance plane-based label layouts we can also use other planes rather than those which are parallel to the current image plane. This is particularly useful if there are dominant surfaces

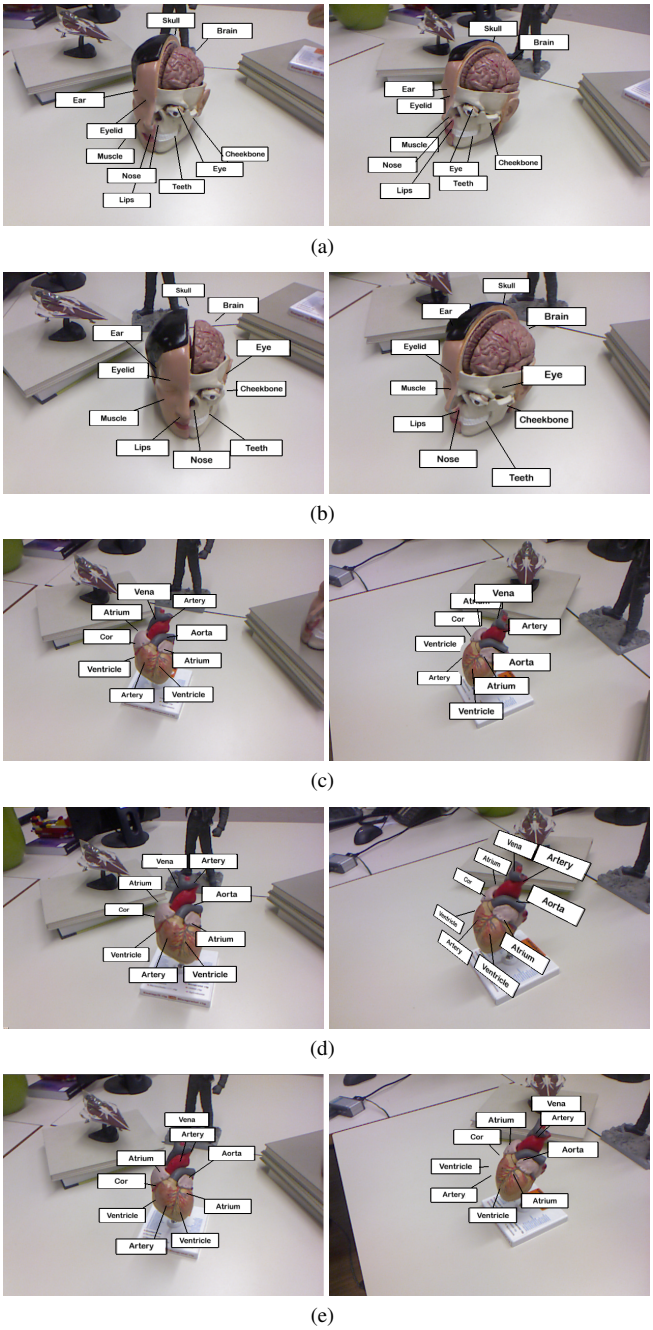


Figure 8: Center-based versus Plane-based Label Placement. (a) Center-based labeling causes unbalanced layouts for this configuration. (b) This can be resolved using plane-based label placement. (c) Plane-based label placement may suffer from occluding annotations if annotation have been placed close to each other. (d) To avoid such occlusions, plane-based label placement in combination with freezing the orientation of annotations can be used. However, this introduces perspective distortions and should be used with care. (e) For this configuration, center-based label placement creates appealing results which neither suffer from occlusion nor from distortion.

of an object, such as a building facade. In such a case, we may want to use labels that are not automatically rotated towards the viewer, but rather remain in the plane of the main surface, and are also displaced in this plane to avoid occlusions.

Because our approach operates in 3D space, annotations become very large in image space depending on their distance to the viewing plane. This reduces the amount of labels we can place. Therefore, we allow to restrict the size of the projected annotation in image space to an upper and lower boundary. If both thresholds are set to the same value, labels have constant size in image space, which allows us to apply any other information filter in 3D or 2D image space ([20, 21]).

For concave objects our center-based hedgehog approach may generate label poles which penetrate unrelated parts of the object between the anchor point and the label's annotation. To handle such situations, we can decompose the object into a set of convex shapes, for which we apply our approach separately. However, this may cause heavy fighting between labels. Such situations can also be handled by rendering labels after scene objects with disabled depth test. In this case, a penetrating pole always appears in front of the object. Since our plane-based approach allows to freely reposition an annotation within the plane, we only have to compensate for penetrating poles in our center-based hedgehog labeling approach.

5 Implementation

Our view management prototype runs in real time (30Hz) for all the illustrations and scenarios presented in this paper with a 640x480 rendering resolution. We deployed our application prototype on a PC running Windows 7, equipped with an Intel i7 CPU quad-core 2.66GHz, 12 GB RAM and an Nvidia 780GTX graphics card. The software was implemented in C++ using OpenSceneGraph[1], an open source scene graph library. The AR framework used in this project uses KinFu, an open source implementation of the Kinect-Fusion approach of Izadi et al. [13], which is available through the Point Cloud Library [18]. KinFu was operated with a Microsoft XBox360 Kinect depth sensor.

6 Conclusion and Future Work

We propose to place external labels in 3D object space and use 3D geometric constraints for achieving label placement. This approach fulfills the desired objectives of layout algorithms (e.g., avoiding overlapping labels), but also behaves consistently over time during viewpoint changes. The two main geometric constraints are a "hedgehog" constraint, where labels originate from a common point and move along a 3D pole stuck into the annotated object, and a plane constraint, where annotations move in a plane that is either parallel to the viewing plane or user-defined in world space.

To the best of our knowledge, our system is the first which resolves occlusions in full 3D space at run-time. We believe that this approach has great potential for future AR and VR applications, which require annotations to be presented continuously over time.

2D view management techniques allow producing high quality still images, but fail for fast and unpredictable camera movements. We trade excessive layout updates for a certain amount of layout distortion. However, if desired, we can continuously orient annotations towards the camera, allowing a result which is visually equivalent to previous state-of-the-art 2D view management technique without their current restrictions.

While our current implementation allows to explore the basic idea of view management in 3D space, a set of open issues remain for future work. For example, we will address proper visualization techniques to handle labels of concave objects as discussed before. Moreover, while the current implementation of our plane-based approach requires the user to set the amount of planes at run-time we plan to develop more advanced plane fitting techniques. Finally, we plan to implement a proper user study to collect user feedback about the configurations of our system depending on the shape of the object, the amount of labels and the application area.

[1] http://www.openscenegraph.org

Acknowledgements

This work was funded by the Christian Doppler Laboratory for Handheld Augmented Reality, the Austrian Science Fund (FWF) under contract P-24021 and the FP7-ICT EU project Nr. 601139 CultAR.

References

[1] J. Ahn and H. Freeman. A program for automatic name placement. In *Auto-Carto 6*, pages 444–455, 1983.

[2] K. Ali, K. Hartmann, and T. Strothotte. Label layout for interactive 3d illustrations. *Journal of the WSCG*, 13:2005, 2005.

[3] B. Bell, S. Feiner, and T. Höllerer. View management for virtual and augmented reality. UIST '01, pages 101–110, 2001.

[4] W. Chigona, H. Sonnet, F. Ritter, and T. Strothotte. Shadows with a message. In *Smart Graphics*, volume 2733 of *Lecture Notes in Computer Science*, pages 91–101. Springer, 2003.

[5] E. M. Coelho and B. Macintyre. Osgar: A scenegraph with uncertain transformations. ISMAR '04, pages 6–15, 2004.

[6] M. Fink, J.-H. Haunert, A. Schulz, J. Spoerhase, and A. Wolff.

[7] T. Götzelmann, K. Ali, K. Hartmann, and T. Strothotte. Adaptive labeling for illustrations. In *Proceedings of Pacific Graphics*, pages 64–66, 196, 2005.

[8] T. G"otzelmann, K. Hartmann, and T. Strothotte. Annotation of animated 3d objects. In *In SimVis, SCS*, pages 209–222, 2007.

[9] H. Gray. *Anatomy of the Human Body*. LEA & FEBIGER, 1918.

[10] K. Hartmann, K. Ali, and T. Strothotte. Floating labels: Applying dynamic potential fields for label layout. In *Smart Graphics*, pages 101–113, 2004.

[11] K. Hartmann, T. Götzelmann, K. Ali, and T. Strothotte. Metrics for functional and aesthetic label layouts. In *Smart Graphics*, pages 115–126, 2005.

[12] E. Imhof. Positioning names on maps. In *The American Cartographer 2(2):128144*, 1975.

[13] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. UIST '11, pages 559–568, 2011.

[14] Z. Jiang, Y. Nimura, Y. Hayashi, T. Kitasaka, K. Misawa, M. Fujiwara, Y. Kajita, T. Wakabayashi, and K. Mori. Anatomical annotation on vascular structure in volume rendered images. *Computerized Medical Imaging and Graphics*, 37(2):131 – 141, 2013.

[15] S. Maass and J. Döllner. Seamless integration of labels into interactive virtual 3d environments using parameterized hulls. Computational Aesthetics'08, pages 33–40, 2008.

[16] J. Marks and S. Shieber. The computational complexity of cartographic label placement. Technical report, Center for Research in Computing Technology, Harvard University, 1991.

[17] T. Ropinski, J.-S. Praßni, J. Roters, and K. H. Hinrichs. Internal labels as shape cues for medical illustration. VMV '07, pages 203–212, 2007.

[18] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). ICRA '11, Shanghai, China, May 9-13 2011.

[19] F. Shibata, H. Nakamoto, R. Sasaki, A. Kimura, and H. Tamura. A view management method for mobile mixed reality systems. In R. van Liere and B. J. Mohler, editors, *IPT/EGVE*, pages 17–24, 2008.

[20] M. Tatzgern, D. Kalkofen, and D. Schmalstieg. Compact explosion diagrams. NPAR '10, pages 17–26, 2010.

[21] M. Tatzgern, D. Kalkofen, and D. Schmalstieg. Dynamic compact visualizations for augmented reality. VR '13, pages 27–30, 2013.

[22] J. Wither, C. Coffin, J. Ventura, and T. Hollerer. Fast annotation and modeling with a single-point laser range finder. ISMAR '08, pages 65–68, 2008.