Shadows with a Message

Wallace Chigona, Henry Sonnet, Felix Ritter, and Thomas Strothotte

Department of Simulation and Graphics,
University of Magdeburg,
Universitaetsplatz 2, D-39106 Magdeburg, Germany
{chigona|sonnet|fritter|tstr}@isg.cs.uni-magdeburg.de

Abstract. Providing textual annotations for 3D visualizations poses a number of interesting challenges: It is difficult to find a suitable position for the text such that the image-text co-referential connection is obvious for the user and, at the same time, the text does not occlude the 3D model. In this paper we address this problem by using cast shadows as text containers. This approach leverages the existing benefits of the shadow metaphor in visualizations. Problems which arise in this context are also discussed.

1 Introduction

We introduce a novel technique for providing textual annotations for 3D geometrical models without occluding the models and without coming in the way of users' interaction with the model. In addition, our technique provides obvious co-referential links between corresponding image and text, thereby reducing cognitive burden which is involved in linking multiple presentations [8, 9].

The benefits of providing textual annotations for images are long known. For instance, MAYER [8] states that textual annotations enhance users' comprehension of images. It has also been shown that *interactive* 3D presentations of visualizations help users perceive spatial as well as functional relationships among objects [11]. Providing text explanations for interactive 3D illustrations, however, raises a number of interesting challenges: There is a need to present the text such that referential connections between corresponding textual and pictorial representations are obvious and, at the same time, the text should neither occlude nor come in the way of users' interaction with the 3D model. Unfortunately, this is not possible using existing techniques.

We address the above stated problem by positioning text *within* cast shadows of corresponding objects. Cast shadows, as used in [5, 12], are 2D abstractions of objects in a 3D scene. Shadows provide important information about spatial relationships among objects [6, 14]. Therefore, by placing text in a shadow we leverage an integral feature of a 3D visualization. Alternatively, if provided primarily to cater for textual annotations, shadows also offer the above mentioned benefits to the visualization.

The benefits of providing textual annotations in a shadow of a corresponding object in the model include the following:

1. The co-referential link between the shadow and the corresponding object is *natural*. Since the shadow and the text are intimately linked, this means the text-image co-referential connection is also *natural*.

2. Since text is provided within a secondary object, it does not occlude or come in the way of users' interaction with the 3D model.

In overview: After providing background information in Section 2, we discuss the concept which we are developing in this paper in Section 3. User interaction details are presented in Section 4, after which, we discuss implementation details in Section 5. Section 6 presents application areas for the technique developed in this paper. Related work is analyzed in Section 7 and Section 8 draws conclusions to this paper.

2 Background

The concept which we are developing is based on the concepts of DUAL-USE OF IMAGE SPACE (DUIS) [3] and ILLUSTRATIVE SHADOWS [12]. In this section we present summaries of both systems.

2.1 Dual-Use of Image Space

In DUIS text corresponding to images is presented *within* the image space. From a technical point of view, the pixels in the image space represent both readable text and, at the same time, shading information for the images. Weight and width of character glyphs are varied to achieve desired shading of graphical objects. Users find text with weight and width variations irritating and difficult to read. To address this problem DUIS has an option of presenting text in a mode which enhances readability (the *reading mode*).

Unfortunately no 3D provision: DUIS is designed for 2D. Applying it on 3D models poses readability problems due to occlusion and perspective distortion.

2.2 Illustrative Shadows

ILLUSTRATIVE SHADOWS [12] is a concept which allows users to interactively explore a detail-rendered 3D model; to avoid interference with the interactions as well as the view of the model, contextual information is displayed in the background. The concept uses the shadow metaphor: Object shadows projected onto a plane, serving as a 2D information display, are a secondary abstract representation of the 3D model. The shadow projection has two significant effects on the visualization:

1. It facilitates the visual understanding due to additional depth cues.
2. It leaves out details which are not relevant and, at the same time, emphasizes relevant elements. This helps to guide users' focus.

Textual information (in annotation boxes) for objects in the 3D model are connected to respective shadows using referential lines. This indirect connection between the corresponding image and text leads to problems in establishing co-referential links between the image and the corresponding text.

Text positioning problem: ILLUSTRATIVE SHADOWS faces the problem of identifying suitable text positions. It is difficult to find a position for the text near the corresponding object where textual information does not occlude the model. The text positioning problem is more complex in interactive environments: Here a once-correct text position may become invalid due to changes in the 3D model. Further on, interaction may produce undesirable effects such as referential lines crossing each other or crossing the scene.

We are proposing positioning text in shadows of corresponding objects using DUIS techniques. Our approach inherits ILLUSTRATIVE SHADOWS benefits while, at the same time, overcoming the above mentioned problems.

Note that we are dealing only with *cast shadows*. Object cast shadows on a shadow plane and never on other other objects. The shadow is cast along a specific vector (usually in the direction of the plane normal) independent of light sources by multiplying the 3D model with a projection matrix [2].

3 Text in Shadows

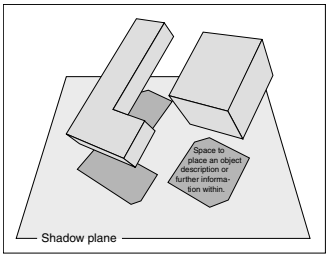

Fig. 1. Shadows with a message: Textual annotations are placed in corresponding shadows.

To explain our concept let us assume a scenario where a user is interacting with a 3D model. Since the main object of interaction is the 3D model, the shadow is positioned behind the 3D model so that it does not occlude the 3D model. When an individual object is selected the following things happen (See Figure 1 and Figure 2-1):

1. The selected object is emphasized, for example, by using color (see Figure 2).
2. The shadow cast by the selected object is emphasized: Its silhouette is rendered using thicker lines.
3. Corresponding textual information appears in the shadow of the selected object.

Positioning text in object shadows in a 3D scene raises a number of new interesting challenges. The following are the main challenges:

1. The shadow may not be accessible to users due to occlusions, i.e. it could be totally or partially occluded by the 3D model or shadows of other objects.
2. The shape of the shadow may not be conducive to reading.
3. The area of the shadow could be too small for the available text.

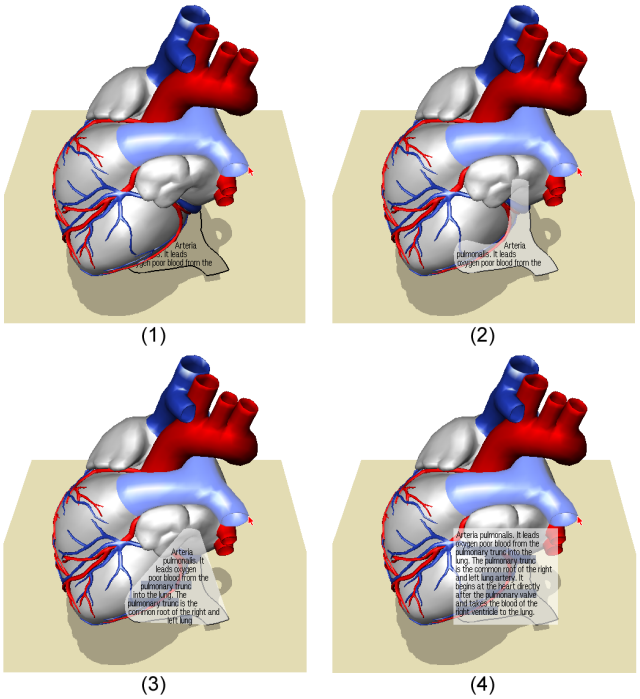

Fig. 2. Text in shadows: The selected object (the one rendered in light-blue) is emphasized. (1) A shadow (and therefore the text) may be partially or totally occluded. (2) To ensure accessibility, the shadow is brought in the foreground. (3 & 4) If the shape of the shadow is not conducive to reading, it can be morphed into its convex-hull or into a rectangle.

3.1 Accessibility of the shadow

Occlusion by 3D model: See Figure 2-1 or the shadow of the L-shaped object in Figure 1. Consequently users cannot access or interact with the text. Solution: When users click a visible portion of the shadow, the shadow moves to the foreground as illustrated in Figure 2-2. The same effect can be achieved by double-clicking the object; this option is particularly necessary when a shadow is totally occluded by the 3D model.

When presented in the foreground, the textual background is semi-transparent so that 3D model in the background remains visible (see Figure 2). This preserves the context.

Occlusion by other shadows: e.g. Figure 4. Since we assume readers would be interested in reading text for the last selected object, we display the shadow layers in chronological order of selection. That is, the shadow of the object which has been selected last is positioned on top. An occluded shadow may be brought in the foreground by either selecting it (if it is partially visible) or selecting the corresponding object.

To ensure readability and legibility of text, only the text for the topmost object is displayed and the shadows of the other selected objects are only highlighted. Displaying text for more than one shadow impedes readability as shown in Figure 4.

3.2 Object shapes and readability

The shapes of shadow silhouettes are usually irregular. This poses readability problems for the following reasons:

1. Lack of familiarity: Humans are used to reading from rectangular surfaces.
2. Text lines presented in concave objects are interrupted by the object shape. These interruptions have negative effects on the reading process [3].

To address these problems, the shape of the shadow *morphs* into the object's convex-hull or into a rectangle (see Figure 2-3 and 2-4). To ensure that users maintain the relationship between the original and the distorted object the distortion is achieved in a smooth continuous animation.

3.3 Discrepancies between object size and amount of text

When there is more text than could fit on the available space, the text is broken into pages which users can page through. However, a special case arises when the space is too small for reasonable amount of text. Here, readers zoom-in to increase the available space. Shape distortion also offers solutions to this problem since in most cases the convex-hull or a rectangle formed from an object has more space than the original.

4 Text-Driven Model Exploration

While reading shadow messages, users frequently wish to look up referenced structures in the 3D model. The visibility of the referenced structure is not guaranteed: objects may be partially or totally occluded by other structures. In such cases it is necessary for users to interact with the model to make the desired structures accessible. However, modifications in the viewpoint or the spatial arrangement of the model result in changes in the shadow projection. This poses a usability challenge when users want to resume reading text for a particular object: Users must relocate their previous reading spot. This is an extraneous cognitive burden to the reading process. The following solutions have been adopted to avoid or attenuate the negative effects:

1. The system provides features to aid users relocating their previous reading spot.
2. Allow users to freeze the shadow so that it does not change when users look up structures in the 3D model.

Our informal tests have shown a positive correlation between the looking up of structures and the encounter of the structure's textual reference (often the name of the structure). In other words, users are likely to look up a structure after encountering its textual reference. This is consistent with the findings of HEGARTY et al. [4] on

the eye movement pattern when users are reading textual annotations for illustrations. HEGARTY et al. observed that after reading a sentence or two, readers look up the part of the illustration which was mentioned in the text. We take advantage of this reading behavior to set *landmarks* in the text body which aid users in relocating their reading positions. Names of the 3D model segmented structures which appear within the text body are highlighted (see Figure 2). A user returning to a page can quickly scan through the landmarks to locate the previous reading position.

Highlighted terms in the text do not only provide landmarks for relocating reading positions, they also serve as means of interaction with the 3D model. When a textual reference of a structure is clicked, the structure is emphasized in the 3D model. Currently visible structures are locally emphasized (we use color and outline to emphasize structures), in contrast, hidden ones require global modifications. We are currently experimenting with exploding the model (adapted from technical illustrations [11]) to reveal otherwise occluded structures. To bring the model to its normal state, the user clicks on the name of the structure again. The shadow and the text, however, is *frozen*, i.e. they do not change along with the 3D model.

5 Implemenation Details

The OPEN INVENTOR library is extended by several nodes for rendering the scene and its shadow projection. Other nodes, which are also part of the scenegraph, are responsible for computing objects' outline, for placing textual information inside shadows and for the smooth distortion of the shadow objects.

5.1 The shadow

The scene is rendered in two steps: First, the illustration plane and the projected model (the shadow). Afterwards the illuminated model is rendered in detail in accordance to its 3D coordinates and material properties. The implication of this rendering order is that the shadow is positioned behind the 3D model so that the model is not occluded. The shadows are managed as layers: The shadow for a selected object is placed at the top and can, therefore, easily be selected. (For details see [12].)

5.2 Positioning text inside a shadow polygon

We require an algorithm which ensures legibility and readability of text as well as permits interaction with the text. The scanning algorithm which we are using meets these requirements. One alternative is to use textures with the textual information which can be mapped onto the shadow polygon. Although this approach might be more efficient (it is hardware-based), it does not permit interacting with elements of the text.

The problem of positioning text in a shadow polygon can be phrased as follows: Given a polygon, find *connected* rectangle blocks where the text can be positioned. The height of the rectangle blocks should be equal to the font height.

OPENGL GLU library tesselation methods are used to compute the outline (the polygons) of the shadows. To optimize readability we use bitmap fonts. For this reason it is necessary to project shadow polygon coordinates into screen coordinates.

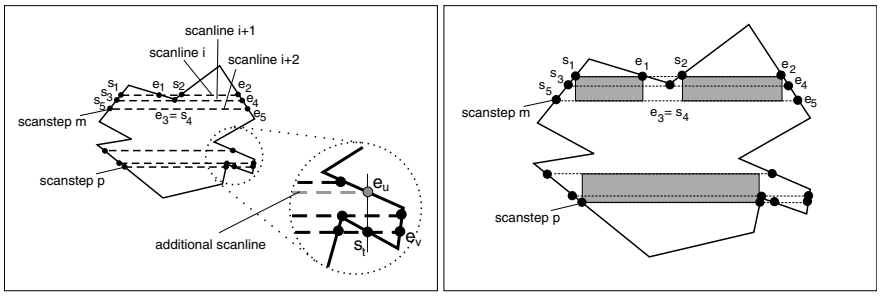

Fig. 3. Left: Polygon scanning; two scansteps (*m* and *p*) and the respective scanlines are illustrated. s_i and e_i are starting or ending intersections of respective scanlines. Right: The resulting blocks where text can be placed.

Our scanning algorithm works as follows:

1. Sort (in ascending order) line segments according to their maximum Y coordinate.
2. *maxY* := polygon's max Y coordinate;
   minY := polygon's min Y coordinate;
   scanIncrement := difference between two sequent scansteps;
   currentScanBase := *maxY* − *scanIncrement*;
   scans := list of Y positions to be scanned;
   $finalIntersections$:= list of final coordinates of current scanstep intersections;
   while *currentScanBase* $\geq$ *minY* **do**
      currentPolyLines := list of active line segments;
      sort *currentPolyLines* in X order;
      clear *scans* and add new scan positions;
      startIntersections := list of intersections of scan line with *currentPolyLines*;
      endIntersections := list of intersections of scan line with *currentPolyLines*;
      for each element of *scans* **do**
         next := next element of *scans*;
         intersections := list of scan line intersections with *currentPolyLines* at *next*;
         add *intersections* to *startIntersections* and *endIntersections* alternately;
      od
      $finalIntersections$:= merge(*startIntersections*, *endIntersections*);
      place text within pairs of $finalIntersections$ coordinates;
      currentScanBase := *currentScanBase* − *scanIncrement*;
   od

Figure 3-left illustrates the scanning algorithm: *m* and *p* are two non-consecutive scan steps. Figure 3-right shows the blocks resulting from the scan algorithm.

Scanstep m: Based on the current scan base, three scan lines are computed as follows:

1. The lower line: The line on the base (this has intersections $\{s_5, e_5\}$).

2. The upper line: The line formed by adding the font height on the base. This forms the following intersections $\{s_1, e_1, s_2, e_2\}$.
3. The line between the lower and upper lines is formed due to an edge which is within the scan step (intersections $\{s_3, e_3, s_4, e_4\}$).

Two lists (*startIntersections* and *endIntersections*) are created from the intersections of the three scan lines: The lists are sorted in an ascending order according to the X coordinate. The two lists are then merged to form a new list ($finalIntersections$) with intersection pairs. Below is a summary of the lists: $startIntersections = \{s_5, s_3, s_1, s_4, s_2\}$, $endIntersections = \{e_1, e_3, e_2, e_4, e_5\}$, $finalIntersections = \{s_1, e_1, s_2, e_2\}$.

The lists merging function searches for connected rectangular blocks within a scan step: Beginning with the first element of *startIntersections*, the X coordinates of *startIntersections* and *endIntersections* elements are compared stepwise. At each step the function searches for the last *startIntersections* element which is smaller or equal to the current element of *endIntersections* (e.g. s_1 and e_1). This pair forms a connected rectangular block.

For the next connected block on the same scan line, the function considers the next *startIntersections* element (s_k); In *endIntersections*: Move to first element which is greater than s_k. The merging function stops when the last *startIntersections* element has been reached and the corresponding *endIntersections* element is found.

Scanstep p: Using the process described above the final segment on the scan step would be $\{s_t, e_v\}$. However, text placed in this segment would overlap the polygonal outline since the height of this segment is less than the font height. To overcome this problem the process has to be *forced* to stop at s_t by adding a new *endIntersection* (e_u). This is achieved by adding to the list of scanline positions a new scanline such that the new scanline intersects the polygon at (e_u). Intersection e_u is a point at which a line starting from s_t and perpendicular to the scan lines intersects the polygon.

6 APPLICATIONS

The techniques developed here have application in interactive exploratory learning environments. It is widely recognized that interactivity enhances learning [10]. In our system, users interact with 3D models without interference from secondary information.

Figure 4 illustrates a potential learning scenario during an apprenticeship. In this case students are learning the spatial composition and functionality of an engine. As they interact with the 3D model textual information is displayed within the shadow. Since the text is displayed in the background and comes in the foreground only upon users' explicit request, it does not interfere with the users' explorative interactions.

7 RELATED WORK

Providing textual information close to corresponding images is a well known problem in the area of label placement in cartography. Area features labeling techniques typically attempt to position labels within features; when the space inside a feature is not

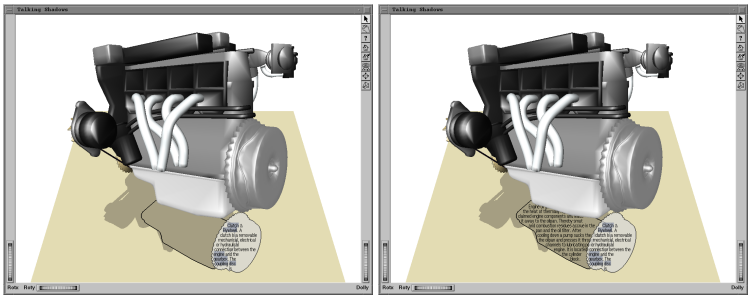

Fig. 4. Two engine elements are selected in the 3D model as well as within the illustration plane. When shadows overlap, the shadow of the object selected last is laid on top. Text is displayed only on the topmost shadow. RIGHT: Displaying text on more than one shadow impedes readability.

sufficient for the label, another position outside the feature is sought. For a detailed list on label placement research refer to the Map Labeling Bibliography [15]. BELL et al. [1] extended the map labeling to cater for augmented reality scenes. The text labels are positioned on projected images of the scene on a head-worn display. Both approaches, however, deal with short text usually just the name of the feature; Long text, as is the case in our approach, lead to the following problems:

1. The text occludes the pictorial information and hinders interactions with the model.
2. Text displayed on a such "textured" background may be difficult to read due to lack of clear contrast between the text and its background [13].

Popup windows may also be used to position textual information close to corresponding images. To avoid the textual information occluding the image, transparent windows may be used [7]. Even though the user may see the image, the text windows hinder interaction with the model.

Another system closely related to our work is the INTERACTIVE SHADOWS [5]. In this approach, in addition to serving as a secondary representation, the shadow is used as an interaction tool for the 3D model. However, the focus is not on integration of textual explanations in visualizations; instead, text is used only to label the shadows.

8 Conclusion

Using the shadow metaphor, ILLUSTRATIVE SHADOWS addressed an interesting interaction problem: Given a 3D model, present corresponding secondary information such that it does not interfere with the rendering as well as the interaction of the 3D model. Here the secondary information is presented as a shadow projected on a plane. The shadow also enhances users' comprehension of the 3D model.

In this paper we use the shadow metaphor to address another critical problem in 3D illustrations: The problem of providing text explanations for the 3D illustrations. We present text corresponding to objects in respective shadows. The techniques for

positioning text in the shadow are adapted from DUIS. The paper has demonstrated problems and corresponding solutions of positioning text in shadows.

Future Work: The problem of occlusion (both 3D model occluding the shadow, or shadow occluding shadow) may be addressed by introducing multiple projection planes and multiple light sources. This option, which was employed in INTERACTIVE SHADOWS [5], would provide more positions where the shadow can be projected. It would also be possible to view text for different objects simultaneously.

For more details about the work (including a video) please refer to:
http://wwwisg.cs.uni-magdeburg.de/research/ts.

References

1. B. Bell, S. Feiner, and T. Höllerer. View Management for Virtual and Augmented Reality. In *Proc of ACM UIST*, pages 101–110, November 2001.
2. J. Blinn. Me and My (Fake) Shadows. *IEEE Computer Graphics & Applications*, 8(1):82–86, Jan. 1988.
3. W. Chigona and T. Strothotte. Improving Readability of Contextualized Text Explanations. In *Proc. of AFRIGRAPH 2003*, pages 141–149. AFRIGRAPH, Feb., 03–05 2003.
4. M. Hegarty, P. Carpenter, and M. Just. Diagrams in the Comprehension of Scientific Texts. In R. Barr, P. M. M.L. Kamil, and P. Pearson, editors, *Handbook of Reading Research. Volume II*, pages 641–668. Mahwah, NJ: Erlbaum, 1996.
5. K. P. Herndon, R. C. Zeleznik, D. C. Robbins, D. B. Conner, S. S. Snibbe, and A. van Dam. Interactive Shadows. In *Proc. of the 5th Annual Symposium on User Interface Software and Technology (UIST'92)*, pages 1–6, Monterey, CA, 1992.
6. G. S. Hubona, P. Wheeler, G. Shirah, and M. Brandt. The Role of Object Shadows in Promoting 3D Visualization. *ACM Transactions on Human Computer Interaction Journal*, 6(3):214–242, Sept. 1999.
7. S. Hudson, R. Rodenstein, and I. Smith. Debugging Lenses: A New Class of Transparent Tools For User Interface Debugging. In *Proc of the 10th ACM UIST*, pages 179–187, 1997.
8. R. Mayer. *Multimedia Learning*. Cambridge Press, Cambridge, UK, 2001.
9. R. Mayer and V. Sims. For Whom is a Picture Worth a Thousand Words? Extensions of Dual-Coding Theory of Multimedia Learning. *Journal of Educational Psychology*, 86:389–401, 1994.
10. D. Mesher. Designing Interactivities for Internent Learning. *Syllabus*, 12(7), 1999.
11. F. Ritter, O. Deussen, B. Preim, and T. Strothotte. Virtual 3D Puzzles: A New Method for Exploring Geometric Models in VR. *IEEE Computer Graphics & Applications*, 21(5):11–13, Sept./Oct. 2001.
12. F. Ritter, H. Sonnet, K. Hartmann, and T. Strothotte. Illustrative Shadows: Integrating 3D and 2D Information Displays. In *Proc. of ACM Conference on Intelligent User Interfaces (Miami, Florida, Jan. 2003)*, pages 166–173. ACM Press, New York, 2003.
13. L. Scharff, A. Hill, and A. Ahumada. Discriminability Measures for Predicting Readability of Text on Textured Backgrounds. *Optics Express: The International Journal of Optics*, 6(4):81–91, 2000.
14. L. R. Wanger. The Effect of Shadow Quality on the Perception of Spatial Relationships in Computer Generated Imagery. In *Proceedings of Conference on Computer Graphics*, pages 39–42. ACM SIGGRAPH, 1992.
15. A. Wolff and T. Strijk. The Map-Labeling Bibliography. |www.math-inf.uni-greifswald.de/map-labeling/bibliography|, 1996.