An Occlusion-Reduced 3D Hierarchical Data Visualization Technique

Reiko Miyazaki, Takayuki Itoh
Ochanomizu University
{reiko, itot}@itolab.is.ocha.ac.jp

Abstract

Occlusion is an important problem to be solved for readability improvement of 3D visualization techniques. This paper presents an occlusion reduction technique for cityscape-style 3D visualization techniques. The paper first presents an algorithm for occlusion reduction. It generates bounding boxes of 3D objects on the 2D display space, moves them to reduce their overlap, and finally reversely projects their movements onto the 3D space. The paper then presents an application of the algorithm to our own hierarchical data visualization technique, and a music browser based on the technique. The paper also shows several numerical evaluations that denote the effectiveness of the presented technique.

1 Introduction

Large-scale visualization techniques became important due to the explosion of information. Many visualization works have focused on efficiency of display usage, because often we would like to look information as much as possible in one display space. Many other works have focused on interactivity so that users can flexibly explore the information. 3D information visualization techniques became also very interactive due to performance evolution of 3D graphics technology. SDM [3] is a typical interactive 3D information visualization system that displays information by a cityscape representation. In 3D visualization techniques we often cause occlusion among objects, and it often prevents the readability of visualization results. This Occlusion is an important problem [4] to be solved, to satisfy the above requirement by 3D visualization techniques.

The paper presents a technique that reduces occlusion for cityscape-style 3D information visualization techniques. The paper first presents an algorithm to avoid occlusion among 3D objects, by moving bounding boxes of the 3D objects in a 2D display space. The algorithm first connects the center points of adjacent bounding boxes by generating a Delaunay triangular mesh on the 2D space. It then moves the bounding boxes to avoid the overlap, by expanding edges of the triangles. The algorithm attempts to minimize the sum of distances between the current positions and the positions calculated as the results of the edge expansion process, and finally obtain good positions of the bounding boxes. It effectively reduce the occlusion of 3D objects by reversely projecting the bounding boxes onto the 3D space and moving the 3D objects according to the movement of the bounding boxes.

The paper then presents an application of the algorithm to our own cityscape-style hierarchical data visualization technique [5]. The hierarchical data visualization technique represents clusters by nested rectangular regions, and leaf-nodes of the hierarchical data by 3D bars. Our implementation generates bounding boxes on the display space, for each 3D bar, or sets of 3D bars in the clusters. It then calculates the optimal positions of the bounding boxes, reversely projects them onto the 3D space, and finally moves the clusters and bars according to the results of the reverse projection. Here, our implementation causes a trade-off between the reduction of occlusion and amount of visible information. 3D bars repulse each other to reduce the occlusion, and therefore many of them go outside the display space. Consequently, less information is displayed when occlusion is reduced successfully. We solved this problem by weighting the movement of 3D bars, so that the technique can strategically avoid the occlusion around the region where a user focuses on. This implementation realizes the focus and context representation, which clearly displays the focused parts without occlusion, and displays information as much as possible in one display space.

We developed an application of the presented technique, called *PileView*, which represents a set of music sound files. We suppose that the music sound files construct a hierarchy based on their metadata (genre, artist name, and truck number). The application represents the files as thumbnail images, and they are piled in a 3D space for each artist. It then places the set of piles by applying our hierarchical data visualization technique, and moves them to avoid the occlusion. The paper shows numerical evaluation (e.g. reduction ratio of occlusion, and ratio of empty areas) of the presented technique by using PileView.

2 Related Work

It is often difficult to display large scale statistics data onto a 2D space. We therefore often represent such data as a set of 3D bars that are spread onto the 2D space. Such 3D-based representation is enough common to be supported by recent consumer spreadsheet software. We often call such representation "cityscape", because the 3D bars look a group of buildings in a city. Occlusion of the 3D bars is a serious problem, and several cityscape-style techniques support interactive mechanism to visualize the invisible parts of the data on demand [2] [3]. Occlusion reduction is an important issue to improve the readability of initial view of the cityscape-style visualization techniques.

Tree and graph are typical data structures for information visualization. Actually, we have large and complicated tree or graph data in our daily life. While several 2D-based tree or graph visualization techniques have aimed all-in-one display of the whole data [1] [5], interactive 3D information visualization techniques are also useful for such data structures. ConeTree [8], InformationCube [7], and H3 [6] are typical famous interactive 3D techniques for visualizing trees or graphs. Again, such techniques may cause occlusion of 3D objects and degrade the readability, and interactive operations are necessary to visualize the invisible parts.

The occlusion reduction technique presented in this paper is inspired by a technique for interactive layout of small objects in a 2D display space [9]. As this technique attempts to keep adequate distances among the objects, our occlusion reduction algorithm also attempts to keep adequate distances among 3D objects in the 2D display space.

3 Occlusion Reduction

This section presents the detailed algorithm of our occlusion reduction technique. This section supposes a cityscape-style 3D visualization system shown in Figure 1(a). The technique effectively moves 3D bars to reduce occlusion, as shown in Figure 1(c).

3.1 Bounding Box

Red boxes in Figure 1(a) are bounding boxes that surround 3D bars in a display space. The technique moves bounding boxes in the display space to avoid the overlap. This section describes the technical detail to move the bounding boxes, while Figure 1(b) also shows the technical detail. The technique then reversely projects the moved bounding boxes onto the 3D space, shown as green boxes in Figure 1(c). It finally moves the 3D bars according to the reserve projection results, and consequently it reduces the occlusion of the 3D bars. Figure 1(c) shows that the algorithm successfully reduces the overlap of 3D bars.

3.2 Adjustment of positions of bounding boxes

The technique adjusts the positions of bounding boxes in a 2D display space, so that it can reduce the overlap of the bounding boxes. Following is the processing flow of the adjustment of the positions:

1. Apply Delaunay triangulation to connect the center points of the bounding boxes.

2. Calculate the areas of overlapped regions for pairs of bounding boxes connected by the edges of the triangular mesh.

3. Calculate the best positions of a pair of bounding boxes, if its overlapped area is larger than 0. Here, the technique calculates the best positions that are on the line expanding the current triangle edge, the pair of bounding boxes touch each other without overlapping, and their movement is the smallest. Figure 1(b) shows the calculation of the optimal positions.

4. Minimize the sum of distances between current and best positions of the bounding boxes. Here, the process fixes the position of bounding boxes which has no overlaps with any other bounding boxes.

Let us formulate the step 4 as follows:

$$argmin\{\Sigma_{i,j} \mid v_{ij} - dv_{ij} \mid^2 + \mid \left(\frac{1}{n}\Sigma_i v_i\right) - c \mid^2\} \quad (1)$$

$$v_{ij} = v_i - v_j, dv_{ij} = dv_i - dv_j$$

Here, v_i and v_j are current positions of two vertices of a triangle edge. dv_i and dv_j are their best positions. n is the number of vertices of Delaunay triangles, which corresponds to the number of bounding boxes. c is the position of the center of the display space. The algorithm calculates the positions of the vertices to minimize the value calculated by the formula (1). The former term of the formula works to make vertices closer to their best positions, and the latter term works to prevent to make the occupied space unnecessarily larger.

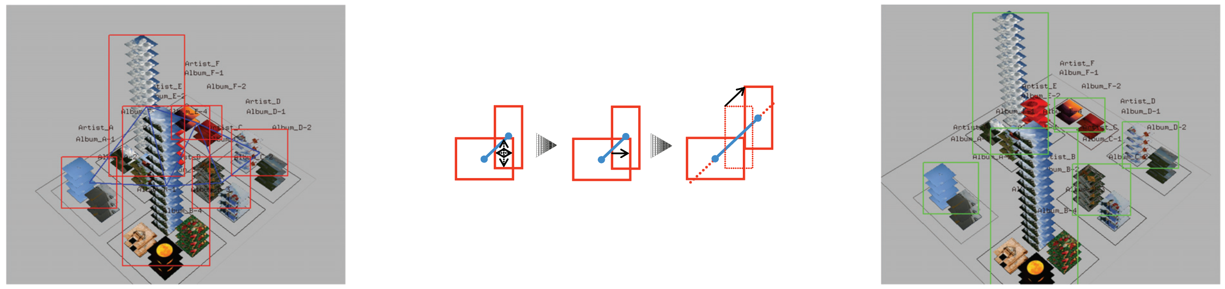
The technique iterates the above process to obtain the better results. Our implementation repeats the process up to 10 times.

3.3 Reverse projection to 3D space

Finally, the technique reversely projects the bounding boxes moved in the 2D display space described in Section 3.2. Following is the process to reversely project the bounding boxes, and calculate the new positions of 3D bars.

1. Let the center point of a moved bounding box in the 2D display space as (x', y', z'). Here, we suppose that x' and y' has been calculated by the process described in Section 3.2.

2. Acquire the z value at (x', y') from the depth buffer. Our OpenGL-based implementation acquires by glReadPixels function.



(a) Definition of bounding boxes (drawn as red rectangles)

(b) Positioning of overlap-avoided bounding boxes

(c) Feedback onto the 3D space

Figure 1: Occlusion reduction.

3. Reversely project the position (x', y', z') in the 2D display space as (x, y, z) in the 3D space. Our OpenGL-based implementation calculates the position by gluUnProject function.

4 Occlusion-Reduced Hierarchical Data Visualization

This section presents the implementation of our visualization technique applying the occlusion reduction algorithm, applying our own hierarchical data visualization technique [5].

4.1 3D Visualization Supposed in This Paper

Figure 2 shows an example of visualization by our technique. Supposing hierarchical data shown in Figure 2(a), our original hierarchical data visualization technique represents the data as shown in Figure 2(b). The technique represents leaf-nodes of the hierarchical data as icons, and branch-nodes as nested rectangular regions. The technique aims all-in-one display of the leaf-nodes in the whole hierarchical data, rather than interactive navigation of the hierarchy. Though the visualization algorithm is 2D technique, our implementation displays the information as cityscape-style, by assigning heights to icons and drawing them as 3D bars.

Figure 2(c) shows PileView, a music browser introduced in the next section. PileView represents hierarchical structure of music contents like cityscape-style visualization techniques, by piling up the leaf-nodes.

4.2 Processing Flow

4.2.1 Step 1: Initial Layout

Our visualization technique represents leaf-nodes of the hierarchical data as 3D bars, and branch-nodes as nested rectangular regions. The technique places the data items based on a bottom-up layout algorithm. The algorithm first places the sets of 3D bars in the lower hierarchy of the data, and encloses the sets by rectangular borders. It then tightly packs the sets of rectangular regions, and again encloses by larger rectangular borders. Repeating the process from the lowest to the top of the hierarchy, it places all the data items onto the display space.

While placing the 3D bars, this step sorts the 3D bars in each cluster based on their heights in order. It then places short bars closer, and tall bars far, so that it reduces the occlusion of short bars by tall bars.

4.2.2 Step 2: Occlusion Reduction in Focused Regions

After completing the initial layout in Step 1, the technique attempts to reduce the occlusion by applying our algorithm described in Section 3.

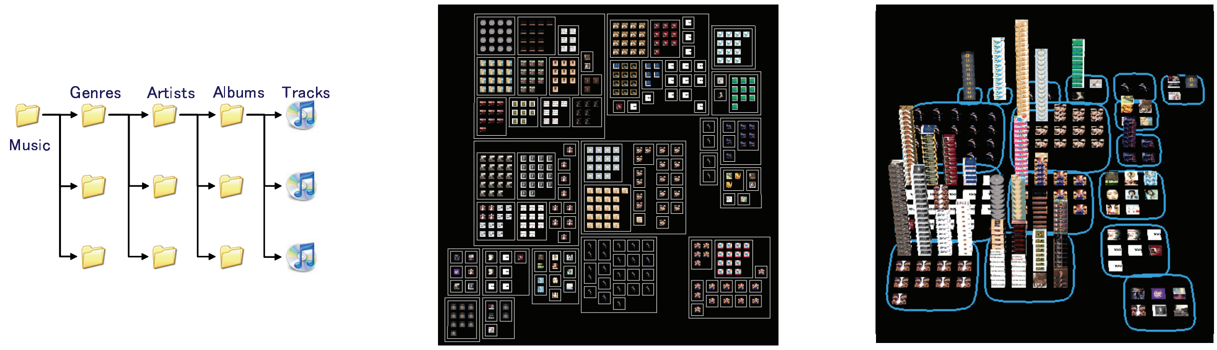
We suppose that a user is focusing on a part of the visualization result, and usually he/she would like to clearly visualize the focused part. The technique supposes that a user is focusing on the rectangular regions that the cursor is pointing. The technique first applies the occlusion reduction for bars of focused regions, when initial layout is calculated, or a user moves the cursor. It encloses all bars in the focused regions by bounding boxes, and applied the occlusion reduction algorithm to the bounding boxes, and moves the bars to reduce the overlap.

4.2.3 Step 3: Occlusion Reduction for All Rectangular Regions

The technique then applies the occlusion reduction for all rectangular regions. It generates bounding boxes for sets of bars of all the rectangular regions, and moves them to reduce the overlap. We had experiments of this two-step occlusion reduction comparing with simple one-step occlusion reduction, and found that the two-step occlusion reduction was much better at the focused regions.

5 PileView: A Music Browser Applying Our Visualization Technique

This section describes the implementation of PileView, which displays a collection of tunes as piled icons. We sup-



(a) Example of hierarchy (music metadata)

(b) Our own hierarchical data visualization technique

(c) 3D visualization by PileView

Figure 2: Hierarchical data visualization presented in this paper.

pose the tunes construct hierarchy based on their metadata. This section calls a leaf-node as *node*, a group of nodes piled up as a tower as *pile*, and a group of piles enclosed by a rectangular region as *frame*.

The pile-based representation is useful to reduce the display area occupied by data items comparing with 2D visualization techniques. Figures 2(b) and 2(c) show the same data; however, piled 3D representation in Figure 2(c) draws icons larger. Pile-based representation reduces the display area while it is possible to click each of the piled icons unless they are occluded. We suppose that the pile-based representation is useful if the occlusion is effectively reduced.

Figure 3 (Left) shows an overview of PileView. It displays all the tunes in the input hierarchical data in one display space, and provides a user interface to select arbitrary tunes. It overdraws text information (music metadata) in the upper-left part of the display space. We developed PileView with Visual C++ and OpenGL, and executed on Thinkpad T60 (Intel Core Duo Processor T2500 2GHz, RAM 2GB) with Windows XP Service Pack 2.

5.1 Input Data

We developed a program to extract metadata of music sound files registered in a music browsing software iTunes[1], using iTunesAPI. Our implementation extracts various metadata, including genre name, artist name, album name, track number, link to the music sound file, and link to the jacket image.

We gathered input data from a set of music sound files registered in iTunes of an author, including 5 genres as frames, 39 artists as piles, and 188 tracks as nodes. We then constructed the hierarchy of the music sound files according to the metadata.

5.2 GUI operation

This section calls a frame pointed by a cursor ”focus frame”. PileView calculates the new positions of piles and frames when the cursor moves and changes focus frames. Consequently, it updates the layout so that it reduces the occlusion among focus frames and their piles. Pileview smoothly moves them applying morphing animation, from the original positions to the new positions, to avoid the sudden change of layout.

PileView can assist users to find arbitrary tunes from large number of collections. It starts playing the music when a user clicks a node.

6 Evaluation

This section introduces the results of numerical evaluations of the presented technique. We can evaluate that the technique is effective, if it satisfies the following two conditions:

- Less occlusion comparing with 3D visualization without applying the occlusion reduction technique.
- Less layout area comparing with 2D visualization by our original hierarchical data visualization technique [5].

The evaluation results in this section show that the technique satisfies the above conditions.

6.1 Evaluation Criteria

We numerically evaluated the following three values (e1, e2, e3) described below, and compared the evaluation of three visualization results (v1, v2, v3) shown in Figure 4.

e1: Ratio of areas of overlapped regions of bounding boxes in focused frames. It is a ratio of total area of overlapped regions, shown as a black region in Figure 3 (Right),

[1] iTunes is a trade mark of Apple Inc.

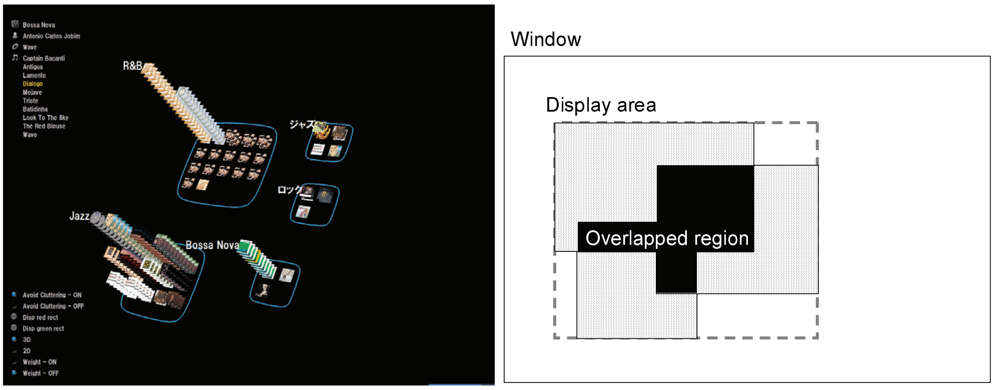
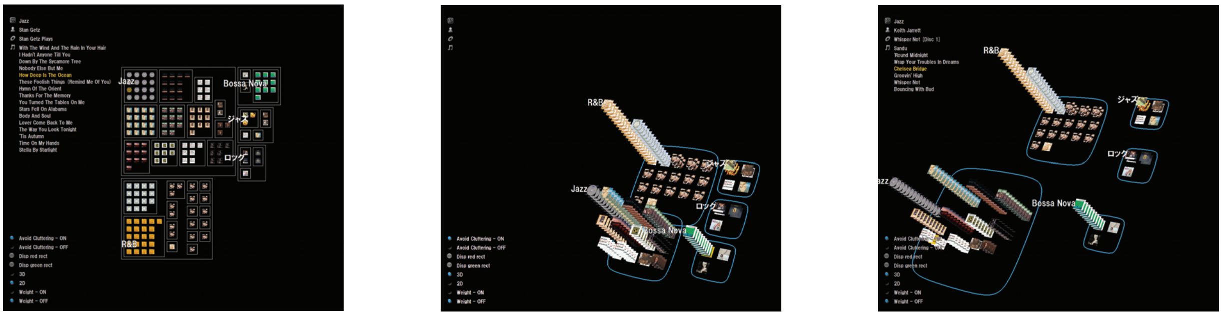


Figure 3: (Left) A music browser applying the presented technique. (Right) Bounding boxes.



v1: 2D visualization (by our original technique)

v2: 3D visualization (without occlusion reduction)

v3: 3D visualization (with occlusion reduction)

Figure 4: Visualization results.

against the total area of bounding boxes in focused frames. We can evaluate that occlusion is successfully reduced in the focused regions when e1 gets lower.

e2: Ratio of areas of overlapped regions of all bounding boxes. It is a ratio of total area of overlapped regions against the total area of bounding boxes. We can evaluate that occlusion is totally reduced when e2 gets lower.

e3: Ratio of expansion of display area. It is a ratio of expansion of the display area, shown as a dotted rectangle in Figure 3 (Right). We can evaluate that all information is compactly displayed when e3 gets lower.

6.2 Evaluation Results

Tables 1 and 2 show the evaluation results. Here, we focused five frames f1 to f5 independently while playing with the 3D visualization with occlusion reduction shown as v3 in Figure 4. Tables 1 shows e1 to e3 values of before and after the occlusion reduction, measuring with the five focus frames independently. Table 2 shows the comparison of the three implementations v1 to v3 shown in Figure 4. Here, we zoomed up the 2D visualization result, shown as v1 in Figure 4, until the sizes of jacket images in the display space got almost similar to the other visualization results.

6.3 Discussion

Table 1 denotes that the presented technique applied the occlusion reduction for various focus frames. In our experiments, piles are partially overlapped in f1, totally overlapped in f2, not overlapped in f3 to f5. In any cases, e1 and e2 values are decreased after the occlusion reduction process. This result denotes that the presented technique effectively reduces the occlusion for various focus frames.

Table 2 denotes that three implementation are numerically compared. It denotes that the presented technique drastically reduced the occlusion around the focus frames, because e1 value of v3 is much smaller than e1 value of v2. Also, it denotes that the presented technique totally reduced the occlusion, because e2 value of v3 is much smaller than e2 value of v2. We can evaluate that the presented technique effectively reduces the occlusion from the above results.

Table 2 also denotes that e3 value of v3 is much smaller than e3 value of v1. We can evaluate that the presented

Table 1: Comparison between before and after avoidance of occlusion.

	e1		e2		e3
Frame	Before	After	Before	After	
f1	42.69	3.70	47.11	8.70	142.67
f2	78.03	20.38	57.78	8.02	173.08
f3 to f5	0.00	0.00	52.60	6.01	156.64
Average	60.36	12.04	52.54	6.95	157.13

Table 2: Comparison of visualization techniques.

	v1 2D visualization	v2 3D visualization before occlusion reduction	v3 3D visualization after occlusion reduction
e1	0.00	60.36	12.04
e2	0.00	52.54	6.95
e3	2.75	1.00	1.57

technique keeps smaller layout area comparing with 2D visualization, even though occlusion reduction process expands the layout area of 3D cityscape-style visualization.

7 Conclusion and Future Works

The paper presented a technique to reduce the occlusion for cityscape-style 3D visualization techniques. We first presented an algorithm to reduce the occlusion, which generates bounding boxes of 3D objects in the 2D display space, and optimizes their movements to effectively reduce the overlap. We then presented the application of the algorithm to our own hierarchical data visualization technique, and the implementation of a hierarchical music browser PileView. We also provided the numerical evaluation that demonstrated the effectiveness of the presented technique, which totally reduced the occlusion, and kept smaller layout area than our 2D visualization technique.

Our potential future work is as follows:

- Reconsideration of shapes of bounding boxes. We do not think that box is always the best shape for surrounding 3D objects. We would like to implement other efficient shapes (e.g. ellipses) as bounding objects.
- More precise numerical evaluations, calculating the overlap of icons themselves, not calculating the overlap of bounding boxes.
- Subjective evaluations with experimental users.
- Implementation and experiments with other cityscape-style 3D visualization techniques.
- Other applications in addition to the music browser.

References

[1] B. Bederson, B. Shneiderman, Ordered and Quantum Treemaps: Making Effective Use of 2D Space to Display Hierarchies, *ACM Transactions on Graphics*, 21(4), 833-854, 2002.

[2] A. Chaudhuri, H.-W. Shen, A Self-adaptive Treemap-based Technique for Visualizing Hierarchical Data in 3D, *IEEE Pacific Visualization Symposium*, 105-112, 2009.

[3] M. C. Chuah, S. F. Roth, J. Mattis, J. Kolojejchick, SDM: Selective Dynamic Manipulation of Visualizations, *User Interface Software and Technology (UIST '95)*,. 61-70, 1995.

[4] N. Elmqvist, M. E. Tudoreanu, Evaluating the Effectiveness of Occlusion Reduction Techniques for 3D Virtual Environments, *ACM Symposium on Virtual Reality Software and Technology 2006*, 9-18, 2006.

[5] T. Itoh, H. Takakura, A. Sawada, K. Koyamada, Hierarchical Visualization of Network Intrusion Detection Data, *IEEE Computer Graphics and Applications*, 26(2), 40-47, 2006.

[6] T. Munzner, H3: Laying Out Large Directed Graphs in 3D Hyperbolic Space, *Proceedings of IEEE Symposium on Information Visualization*, 2-10, 1997.

[7] J. Rekimoto, M. Green, The Information Cube: Using Transparency in 3D Information Visualization, *Proceedings of the Third Annual Workshop on Information Technologies & Systems (WITS 93)*, 125-132, 1993.

[8] G. G. Robertson, J. D. Mackinlay, S. K. Card, Cone Trees: Animated 3d Visualizations of Hierarchical Information, *Proceedings of the SIGCHI Conference on Human factors in Computing Systems*, 189-194, 1991.

[9] N. Watanabe, M. Washida, T. Igarashi, Bubble Clusters: An Interface for Manipulating Spatial Aggregation of Graphical Objects, *ACM User Interface Software and Technology*, 173-182, 2007.