MIAOW: A 3D Image Browser Applying a Location- and Time-Based Hierarchical Data Visualization Technique

Author 1
author1@acm.org

Author 2
author2@acm.org

ABSTRACT

Browsing techniques for large image collections become increasingly important for overviewing and retrieving images. We had a questionnaire result that many of photograph collectors think that time and location are useful information to organize, browse, and retrieve them. This paper proposes MIAOW (Memorized Image Album Organized by When/Where); a 3D image browser which represents hierarchically categorized photographs based on their shooting locations and times. MIAOW utilizes a 3D space with an orthogonal coordinate system to place a set of photographs; it assigns two axes to the shooting locations of the photographs, and the other axis to their shooting time. Supposing that all images have shooting locations (longitudes and latitudes) and times, MIAOW hierarchically categorizes the set of images as a preprocessing. MIAOW then places all clusters of the images onto the XY-plane of the 3D space as nested rectangular regions, while it attempts to avoid overlapping, minimize the occupied area, and reflect the locations. It also places the same set of the clusters onto the XZ- or YZ-place, avoids overlaps, minimizes the occupied area, and reflects the time and result of the placement on XY-plane. MIAOW provides an orientation and zooming user interface, so that users can easily navigate between location and time spaces, and zoom into interested clusters of photographs. This paper demonstrates our user experiments showing that the users required less time to search for specific photographs by using MIAOW rather than using an existing browser.

Keywords

Information visualization, hierarchical data, image browser.

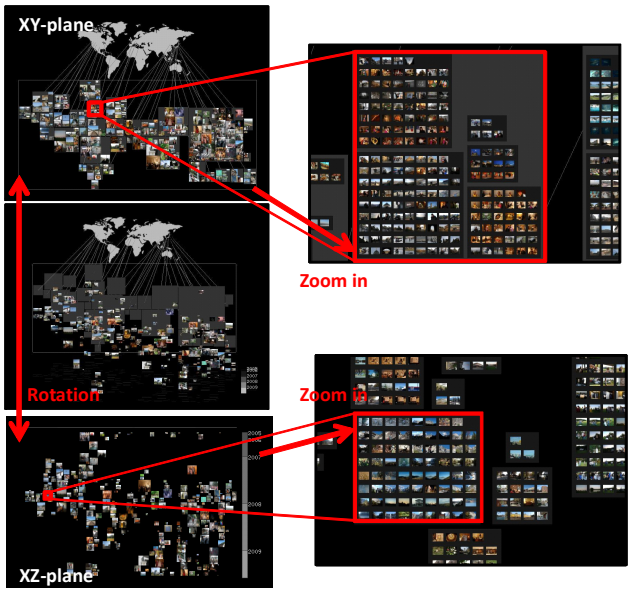

Figure 1: 3D image browser MIAOW. It represents hierarchical clusters of 9,500 photographs as nested rectangular regions. (Upper-Left) XY-plane which the two axes correspond to shooting locations of the photographs, displaying representative photographs of the clusters. (Lower-Left) XZ-plane which the two axes correspond to shooting locations and date of the photographs, displaying representative photographs of the clusters. (Right) Displaying all photographs in the clusters while zooming in.

1. INTRODUCTION

The recent revolution of digital camera technology has resulted in much larger image collections. Many personal camera users store tens of thousands of photograph digitally, and they are expected to be used as life logs. Image browsers are important and useful applications for the overview and retrieval of such photograph collections. Many existing image browsing techniques focus on intelligent layout and navigation of images. Several image browsing techniques apply dimension reduction schemes such as MDS for similarity-based image layout [16, 18]. Several other techniques represent structures of images, such as graphs or clusters, for intelligent layout of the images [2, 10]. Focus+context or zooming interfaces are also useful for image browsing [10, 16].

Generally, we need metadata associated with the photographs, so that we can smartly organize, browse, and retrieve them. We had a questionnaire with the question: "Which metadata do you think important to retrieve specific photographs

from your personal collections?" We asked 24 university students who are collecting photographs, and result was as follows: 18 students selected time, 18 selected location, 10 selected keywords, 5 selected features (e.g. color), and 3 selected owner or photographer.

Here, we cannot suppose that keywords are always assigned to each and every photograph. Several image browsing techniques apply manually assigned annotations or created structures. There have been many user interface techniques to reasonably or enjoyably assign the metadata to sets of images. However, it may often be inconvenient for users to manually assign metadata to every image of their collections. Automatic image annotation is also a potential approach, where various statistical and machine learning techniques have been applied. Though there have been novel techniques in this field, still it is not always reliable to adequately assign annotation to all the images.

On the other hand, content based image retrieval (CBIR) is an extensively explored topic and has been applied to various problems in image search and categorization. Many of studies calculate feature vectors from images using their colors and textures, and some recent studies use regional features or key points. Again, though there have been novel techniques, still it is not always reliable to well-organize every image based on the features.

Reflecting the questionnaire result and the above discussion, our study focuses on using time and location information for personal photograph browsing. We think timestamps are one of the best information to automatically and effectively organize collections of personal photographs. We can safely assume that timestamps are automatically recorded with every image taken by digital cameras. We often copy photographs from digital cameras to computers event-by-event or season-by-season, and therefore it is natural for us to organize images based on temporal information. We think location data will be also good information for image browsers. Recent cell-phones embed digital cameras and GPS (global positioning system functionality), and therefore it is easy to automatically record location information to images taken by them. Based on the above consideration, we discuss how to effectively utilize time and location information to develop a preferable image browser.

This paper presents the 3D image browser MIAOW (Memorized Images Album Organized by When/Where), which represents hierarchically clustered photographs based on their shooting locations and times. MIAOW utilizes a 3D space with an orthogonal coordinate system to place a set of photographs; it assigns two axes (X and Y in this paper) to the shooting locations of the photographs, and the other axis (Z axis in this paper) to their shooting time. We expect that MIAOW makes it easier for users to search for personal photographs associated with their memories with their locations and times. Figure 1 shows a snapshot of photograph browsing by MIAOW.

As a preprocessing, MIAOW firstly applies a clustering algorithm to divide the photographs based on their longitudes and latitudes. It then divides the photographs in each cluster based on their times. Consequently, a collection of photographs constructs a two-level hierarchical structure. MIAOW then places all image clusters onto XY-, XZ-, and YZ-planes of the 3D space, representing them as nested rectangular regions. MIAOW firstly places all clusters of the images onto the XY-plane of the 3D space as nested rectangular regions reflecting the locations. This process is similar to a previously presented image browser (Clustered Album Thumbnail) [5], which applies hierarchical rectangle packing algorithm to effectively place a set of images. MIAOW then places a same set of clusters onto the XZ- or YZ-place while reflecting the time and the placement on XY-plane. This paper presents a new rectangle placement algorithm that reflects a previously solved placement result along an axis (X- or Y-axis in this case), and reflects a variable (time in this case) along another axis (Z-axis in this case).

MIAOW provides orientation and zooming user interface. The orientation interface allows users to navigate between location and time spaces, so that they can easily find desired clusters of photographs. When a user would like to look at the photographs taken at a certain location, and the user vaguely remembers when the photographs were taken, it is convenient to look all the photographs on the XY-plane first, and then switch to the XZ- or YZ-plane so that the user can view the timeline of the photographs taken at the location. Or, when a user would like to see the photographs taken at a certain time, it is convenient to look at all the photographs on the XZ- or YZ-plane first, and then shuttle to XY-plane so that the user can look the locations of the photographs. Meanwhile, the zooming interface is useful to focus on interested clusters of photographs. Our current implementation firstly displays representative images for each cluster, and then switches them into the individual images in the clusters when a user zooms into them. This mechanism is preferable both for visual recognition and system performance, as discussed in [5].

2. RELATED WORK

2.1 Image Browsing

Many image browsing interfaces, such as image search engine Web sites, simply provide a set of images in grid layout in the ranking order of a certain similarity criteria with respect to the query. These kind of interfaces is not always effective for quickly finding all the desired images. More sophisticated user interfaces for image browsing can be valuable to easily explore personal image collections. Kang et al. presented a technique for exploring personal image collections [12] which consists of various query interfaces, thumbnail and detail viewers; however, it does not focuses on hierarchical representation of image collections.

Some browsing techniques focus on the layout of image thumbnails so that they can finely represent content similarity among the images. We can embed images in a low-dimensional Euclidean space preserving the distances between pairs of them, using, for instance, multidimensional scaling (MDS). Semantic Image Browser (SIB) [18] also applied MDS for the similarity-based layout of image thumbnails. Walter et al. presented Hyperbolic Image Browser [16] which scatters images onto a hyperbolic space applying MDS, and provides a novel focus+context user interface. These techniques are good at representing distances among images; however, it often overlaps many images each other on the display. We

would like to argue that users may prefer a grid-like layout for the display of image collections, since with this, layout images never overlap.

Information visualization techniques are suitable for image browsing because the main goals of information visualization include the overview and interactive exploration of large datasets, especially structured datasets such as trees or graphs. Chen et al. presented a technique for visualization of a network of images [4]. Jankun-Kelly et al. presented a technique for interactive focus+context graph layout technique [10], which represents a set of linked images by mapping thumbnails onto nodes of the graph.

Bederson presented PhotoMesa [2], which places groups of images into subregions of display space, applying a space-filling hierarchical data visualization technique Quantum Treemap [1]. Kustanowith et al. presented a technique [14] that radially places clusters of images, and provides capabilities to interactively resize the layout for focus+context representation of the collections of images. Gomi et al. presented an image browser CAT [5], that represents clusters of images as nested rectangular regions applying a fast rectangle packing algorithm. These techniques represent collections of images as clusters, and display the images by grid-like layout. MIAOW also represents the clusters of images as nested rectangles.

2.2 Hierarchical Data Visualization

There are many well-known works on hierarchical data visualization, where a large portion of them are categorized as tree-based approaches, and the others are space-filling approaches. Tree-based approaches, such as Hyperbolic Tree [15] and Cone Tree [3], provide good navigation and exploration capabilities for large-scale hierarchical data. On the other hand, Space-filling approaches, such as Treemaps [11], divide the display area into subspaces according to the hierarchy of input data, and spread the whole data onto the display area. One of the features of the space-filling approaches is the all-in-one visualization of lower-level data items of hierarchical data, rather than the representation and navigation of hierarchy relationship between parent and child nodes. Since the goal of this paper is the all-in-one visualization of clustered images, space-filling approaches are appealing for browsing of large collections of clustered images.

MIAOW applies a Treemap-like space-filling hierarchical data visualization method based on nested rectangular regions [8, 9]. It places thousands of leaf-nodes into one display space while satisfying the following conditions: 1) no overlapping between the leaf-nodes and branch-nodes in a single hierarchy of other nodes, 2) drawing of all leaf-nodes by equally shaped and sized icons, 3) minimization of the display area requirement, and 4) minimization of aspect ratio and the area used for rectangular subspaces. A desirable trait of the technique is the representation of lower-level data items as clickable and equally-sized thumbnails, which has also been addressed by Quantum Treemap. Experiments described in [9] discuss trade-offs between Quantum Treemap and the newer technique, where the latter technique yielded better results regarding the aspect ratio of subregions and the stability of the layout among similar hierarchical data. One more advantage of the latter technique is the flexible control of the placement of rectangular subspaces. As described in Section 5 of [9], the technique has a capability to control positions of rectangles by referring templates, which describe the ideal node positions. This feature is applied by MIAOW, to effectively represent locations and times of image clusters.

2.3 Rectangle Placement with Geo-Spatial Constraints

MIAOW places rectangular regions onto a 2D space while attempting to reflect their pre-assigned positions. There have been many visualization techniques that represent data items as rectangles and place them reflecting pre-assigned positions. The hierarchical data visualization technique applied to the image browser [8, 9] is a typical technique that attempts to reflect the pre-defined positions.

Heilmann et al. presented RecMap [7] that represents geographic items (e.g. states in the USA) as rectangles while preserving important geo-spatial constraints. It attempts to minimize the errors of areas, shapes, topology, and positions of the rectangles, and empty area, while it preserves other constraints of the rectangles, by applying optimization schemes. The paper demonstrates good-looking visualization results; however, the following small bottlenecks are still remaining: 1) the technique requires a constraint that every rectangle should be neighbor of at least one rectangle, 2) the technique required 55 seconds in their experiments, and 3) the result in their paper contained inaccurate portions, such as where San Diego was located north of LA in Figure 8 (c) in [7].

Wood et al. presented Spatially Ordered Treemaps [17] that calculates orders of nodes based on spatial consistency and divides a rectangular display space by the calculated order. It can generate good cartograms while it inherits good properties of Treemaps; however, it does not always preferably represent geo-spatial properties, if the target regions are not originally close to rectangular regions, or they contain disjoint regions.

2.4 Space-Time Cube

Time and location are quite relevant while observing human behavior. Hagerstrand [6] introduced the space-time cube model, a 3D space in which x- and y-axes are assigned to geography, and the z-axis is assigned to time. This model is famous for the representation of space-time behavior of humans. Several visualization works in space-time cube model [13] demonstrated the usefulness for observation of human life. MIAOW places collections of photographs into a space-time cube, aiming to retrieve photographs along human life, or observe human behaviors themselves.

3. PRESENTED TECHNIQUE

3.1 Technical Overview

Figure 2 shows the processing flow of image clustering and browsing processes. Here, we assume that a collection of images each assigned with location (longitude and latitude) and time are given. Also, we assume functions to calculate x-, y-, and z-coordinate values from the given values as follows:

$$x = f(lo), y = g(la), z = h(t) \quad (1)$$

Here, lo is the longitude, la is the latitude, and t is the time. Our current implementation simply applies linear functions to f, g, and h. It assumes that longitude lines are perpendicular, and latitude lines are horizontal to the display space. Consequently, our implementation places clusters of images on a Mercator projection map. We decided to apply a Mercator projection map because it is one of the most popular projection maps; however, we can apply other various projections as well by replacing the functions f and g.

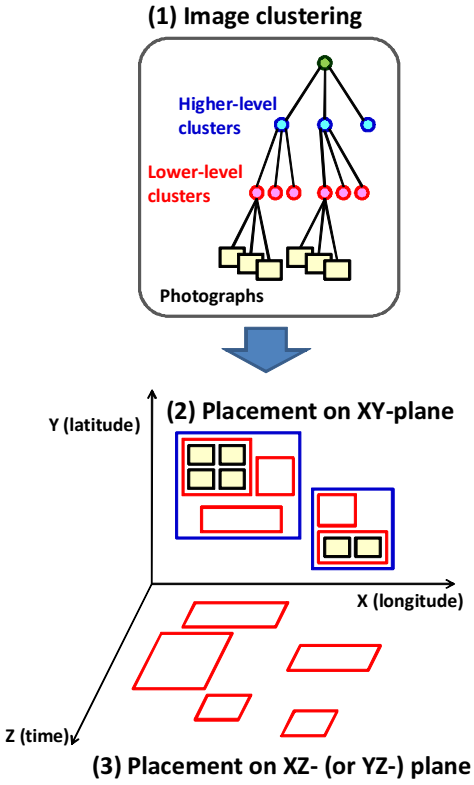

Figure 2: Processing flow of MIAOW.

As a preprocessing, MIAOW first divides images according to their x and y values, and constructs higher-level clusters of images. It then divides the images in each of the clusters according to their z values, and constructs lower-level clusters of images. After the clustering process, it selects a representative image for each cluster. Consequently, it constructs a two-level hierarchy of the images, as shown in Figure 2(1).

MIAOW then places the set of image clusters as nested rectangular regions onto XY-, XZ-, and YZ-places. Here, we suppose a 3D space and an orthogonal coordinate system, where X- and Y-axis correspond to location, and the Z-axis corresponds to time. MIAOW first calculates the positions of clusters on the XY-plane, as shown in Figure 2(2). It represents the two-level hierarchy as nested rectangles, while it avoids the overlapping same-level rectangular regions, attempts to minimize the display space, and reflects the location. MIAOW finally calculates the positions of clusters on the XZ- and YZ-planes, as shown in Figure 2(3), by a new algorithm presented in this section. It represents only lower-level clusters as rectangular regions while avoiding overlapping, attempts to minimize the used display space, reflects the X- or Y- coordinate values of the lower-level rectangles calculated during placement on the XY-plane, and reflects the time.

3.2 Image Clustering

MIAOW constructs a two-level hierarchy of images by applying two-level clustering. It first divides images by shooting locations, and then divides them by shooting time. It is important to divide by locations first, because our design aims to place all clusters of images according to their locations on the XY-plane.

Our implementation of the first clustering simply calculates distances between any arbitrary two images on the XY-plane, then constructs a dendrogram based on the distances, and finally divides the images according to a predefined distance threshold. After the clustering, it calculates the average longitude and latitude values for each cluster, which are used for location-based rectangle placement as described in Section 3.3.

Our implementation of the second clustering simply sorts the photographs in each of the higher-level clusters based on their shooting times, then constructs a dendrogram based on the time differences, and finally divides the images according to a predefined difference threshold. After the clustering, it finally calculates the average time values for each cluster, which are used for time-based rectangle placement as described in Section 3.4.

3.3 Location-Based Hierarchical Data Placement

MIAOW applies a Treemap-like space-filling hierarchical data visualization technique [8, 9]. It places a set of images onto a display space based on a bottom-up packing algorithm consisting of the following three phases:

Phase 1-1: The technique first places a set of image thumbnails in a lower-level cluster using grid layout, and encloses them by a rectangular border. It repeats this process for all of the lower-level clusters.

Phase 1-2: The technique then packs and encloses all the rectangles corresponding to the lower-level clusters that belong to the same higher-level cluster by a rectangular border. It repeats this process for each of the higher-level clusters.

Phase 1-3: The technique finally packs the rectangles of all the higher-level clusters, and encloses them by a rectangular border. While packing the rectangles, it reflects the locations of the clusters.

Since the technique places representative images of clusters into the rectangular borders, aspect ratios of the rectangular areas should be as close as possible to the aspect ratios of the representative images. For this requirement, we slightly modify the condition for rectangle placement described in [9] as follows: In Phase 1-1, the technique calculates the horizontal and vertical number of images in the grid layout so that the ratio of the numbers is as close as possible to

the aspect ratio of the representative image of the cluster. In Phases 1-2 and 1-3, we slightly modify the calculation of aspect ratio of rectangles. Here, the rectangle packing technique attempts to minimize the following value:

$$aA + rR + dD \tag{2}$$

Note that a, r, and d are user-defined positive constant values, A is the ratio of aspect ratio of the rectangular border between before and after rectangle placement process, R is the aspect ratio of the rectangular border between before and after rectangle placement process, and D is the distance between the actual position of the rectangle and its ideal position described in a template.

We modify the variable A as $A = A_2/A_1$, for the above requirement. Here, the technique calculates A_1, which is the error of aspect ratio of rectangular border against the ideal aspect ratio, before the rectangle placement. Similarly it calculates A_2, which is the error after the rectangle placement. Here the ideal aspect ratio is the aspect ratio of the representative image in Phase 1-2, or the aspect ratio of the window space in Phase 1-3.

Another requirement for the data layout is reflection of geospatial information. The rectangle packing algorithm applied in this paper has the capability to refer to the templates that describe cluster locations. Details of template-based data layout algorithm are described in Section 5 of [9]. The technique prepares the templates which describe X- and Y-coordinate values calculated from longitude and latitude values of the higher-level clusters. Our implementation treats $d = 0$ in the Phase 1-2, and d as an adequate positive value in the Phase 1-3, because it applies the template only to higher-level clusters in the Phase 1-3.

3.4 Time-Based Hierarchical Data Placement

After completing the rectangle placement on the XY-plane, MIAOW places rectangles on the XZ- and YZ-planes, where Z-axis corresponds to the shooting time of the images. Here, we would like to design the image browser so that we can smoothly navigate between location-based and time-based planes. To realize seamless switching between XY- and XZ- (or YZ-) planes, we invented a new hierarchical data placement algorithm, which preserves the X- or Y-coordinates values calculated on the XY-plane.

Figure 3 illustrates the algorithm of the time-based hierarchical data placement. This paper explains the data placement on the XZ-plane, but YZ-plane can be also filled by the same algorithm. The algorithm consists of the following three phases to place a set of images onto the XZ-plane:

Phase 2-1: The algorithm first packs all rectangles corresponding to lower-level clusters onto the XZ-plane while preserving their X-coordinate values calculated during Phase 1-2.

Phase 2-2: The algorithm then adjusts Z-coordinate values of the rectangles so that their positions well represent the times of the clusters.

Phase 2-3: The algorithm finally calculates the positions of the images inside the rectangles.

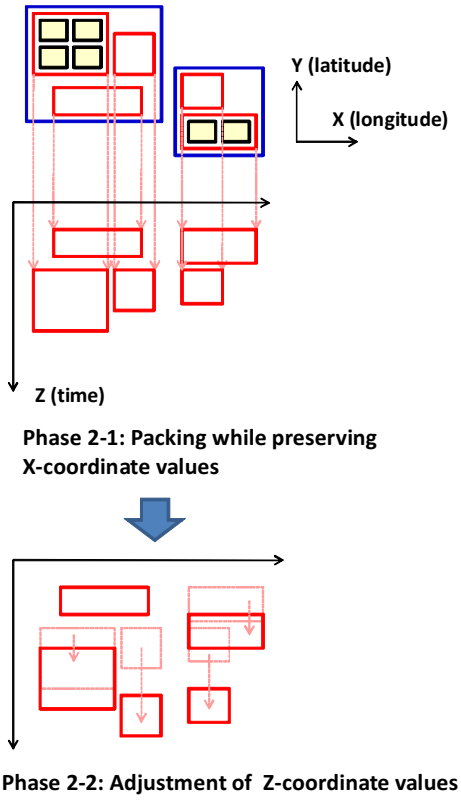

Figure 3: Rectangle placement on the XZ-plane.

Note that this process only places lower-level clusters on XZ- and YZ-planes. In other words, it does not pack the rectangles corresponding to the lower-level clusters independently for each parent higher-level cluster; instead it packs the lower-level clusters all-in-one.

In Phase 2-1, the algorithm first sorts all the lower-level clusters in ascending order of the time. It then places the rectangles corresponding to the lower-level clusters one-by-one, while preserving their X-coordinate values. Defining the minimum Z-coordinate value Z_{min}, the process looks for the position of the current rectangle R satisfying the following conditions:

- R does not overlap with any already placed rectangles.
- R touches any of the already placed rectangles, or the line $z = Z_{min}$.

In Phase 2-2, the algorithm then adjusts the Z-coordinate values of the rectangles so that their locations adequately represent their shooting times. It defines the maximum Z-coordinate value Z_{max} by scanning the positions of the all rectangles. It then calculates the ideal Z-coordinates of the rectangle R_i as follows:

$$Z_i = \frac{t_i - t_{min}}{t_{max} - t_{min}}(Z_{max} - Z_{min}) + Z_{min} \tag{3}$$

Here, t_i is the time of R_i, t_{max} and t_{min} are the maximum and minimum values of the times of the rectangles.

The process then looks for the new positions of the rectangles one-by-one, where they do not overlap with any other rectangles, and are located as close as possible to Z_i, while preserving their X-coordinate values.

3.5 Orientation and Zooming User Interface

Our implementation supports various mouse operations for image browsing. It assigns the left button to translation, the right button to orientation, and the wheel to zooming operation.

MIAOW supports rotation around X- or Y-axis according to the mouse operation. When a user moves the cursor vertically, it converts the movement to a rotation around X-axis. When moving horizontally, it applies a rotation around Y-axis. Figure 4 illustrates the operation while rotating the images around the X-axis. During this operation, MIAOW assigns positions of images into the 3D space. For example, if an image us placed at (x_0, y_0) on the XY-plane and (x_0, z_0) on the XZ-plane, the image floats at (x_0, y_0, z_0) in the 3D space, while rotating around the X-axis. Also, MIAOW inversely rotates the images against the operation, so that the images always face towards the viewer.

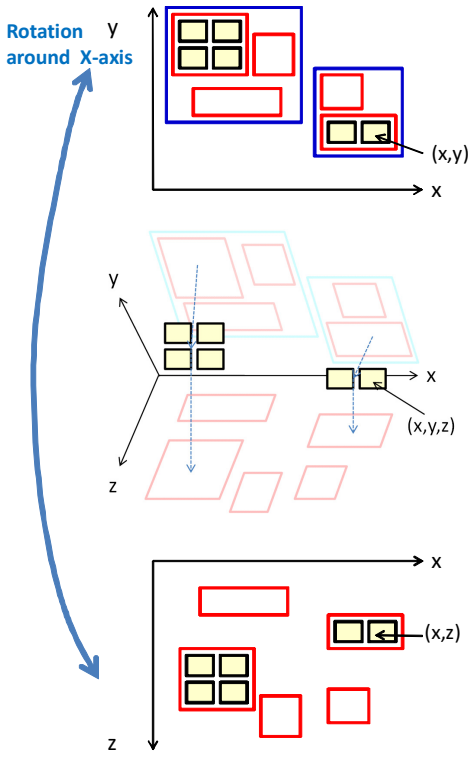

Figure 4: Rotation around X-axis.

MIAOW switches the displayed image according to zooming operation. When zooming out, it displays representative images of higher-level clusters. Zooming in, it switches to representative images of lower-level clusters, and finally to each image thumbnail. The representative images are displayed inside the rectangular borders representing the clusters. MIAOW stretches the representative images if the aspect ratios of the rectangular borders are not equal to those of representative images.

In our implementation, MIAOW initially displays the XY-plane, loading only representative images of higher-level clusters from the hard disk drive into the main memory. It then loads each image thumbnails in the focused clusters on the fly, or frees memory space for image's thumbnails in the defocused clusters, during zoom operations. This mechanism helps optimizing both frame rate and memory usage. Also, it loads full-size images when the image thumbnails are pointed to by a cursor, and displays the full-size images as adequately sized.

4. EXAMPLE AND EVALUATION

We implemented MIAOW with Java JDK 1.6.0, and tested on an IBM ThinkPad T500 (CPU 2.8GHz, RAM 2GB) running Windows XP SP3. We used our personal photograph collection with images of size 100x75 pixels stored in JPEG format as thumbnails or representative images, constructing three hierarchical datasets. Table 1 shows the sizes of three datasets. The table also shows the computation times for calculating positions of all thumbnails and rectangular regions. The result demonstrates that MIAOW calculates the positions within a reasonable time. Figure 1 shows Dataset 3.

Table 1: Datasets used in our experiments.

Dataset	1	2	3
Images	1015	5000	9500
Higher-level clusters	51	243	295
Computation time (msec.)	32	1297	1501

4.1 Example

This section shows several examples of visualization results using the three datasets. Figure 5 shows examples of displaying the XY-plane applying Datasets 3. Our implementation displays a world map behind the XY-plane that images are placed, and draws connecting lines between higher-level clusters and corresponding positions on the map. These results demonstrate that MIAOW places representative images of clusters while it avoids overlap among the images, attempts to minimize the display area, and preserve the geospatial adjacency among the clusters. Figure 6 shows an example of displaying the XZ-plane applying Dataset 1. Our implementation displays a vertical bar which denotes the years, so that users can easily find images taken in the specific years. Figure 7 shows an example of rotation operation from the XY-plane to the XZ-plane when applying Dataset 2. This example demonstrates that MIAOW can seamlessly switch between XY- and XZ-plane by the rotation operation.

4.2 Experimental Test

We had an experimental test with 12 students majoring computer science in our laboratory. We asked them to play with MIAOW and an existing 2D space-filling image browser for just a few minutes to master the operations. Then, we asked them to search for images specified by various conditions, such as "an image of Kawaguchiko-lake taken in May 2009".

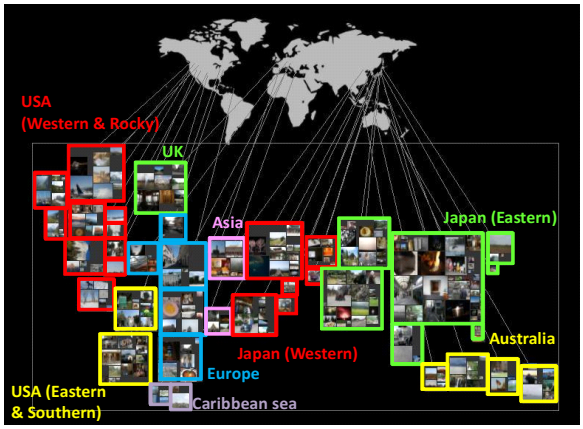

Figure 5: XY-plane (using Dataset 3).

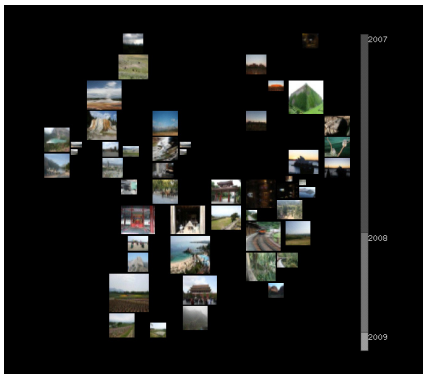

Figure 6: XZ-plane (using Dataset 1).

Here, the image collection used in the search test (Dataset 3) included photographs taken at the events of our laboratory, which should be memorized with locations and times. We used another image collection while playing with the browsers to master the operations. We then asked the examinees to search for three photographs, and measured the times taken to search for them. In this experiment we used CAT [5] as an existing image browser, but we expect we will get similar result by using other existing 2D space-filling image browsers such as PhotoMesa [2].

Table 2 shows maximum, minimum, and average times taken to search for specific images by examinees. This result demonstrates that MIAOW is much more effective to search for specific images if they are associated with locations and times in the user's memories.

Table 2: Time taken to search for specific images. (sec.)

	Minimun	Average	Maximum
CAT	46.6	211.25	637.4
MIAOW	22.6	62.4	162.8

We heard from the examinees how MIAOW was better than the existing image browser. All of the examinees mentioned that they could quickly zoom in the specific cluster on the

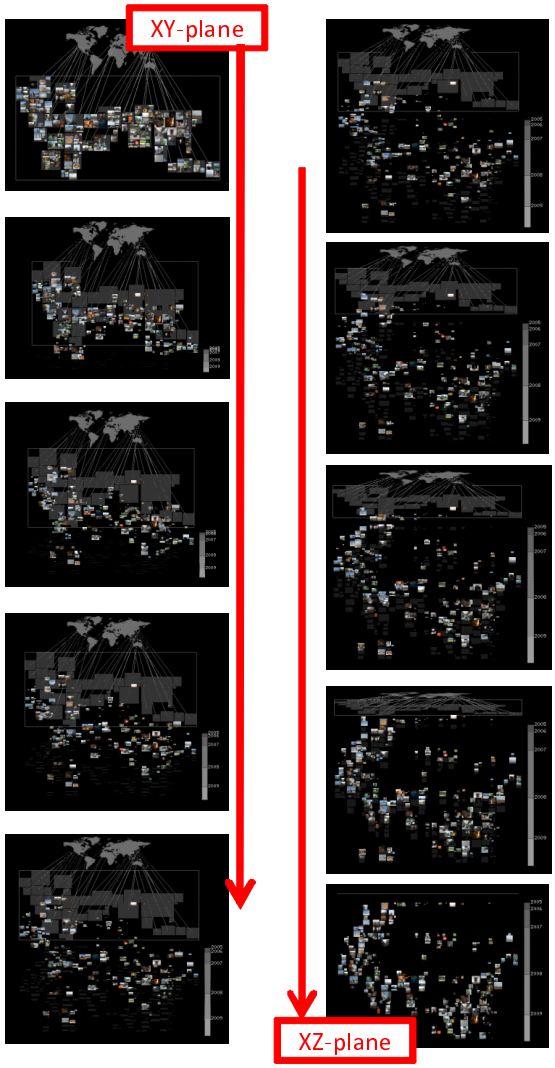

Figure 7: Rotation from XY-plane to XZ-plane (using Dataset 2).

XY-plane when they heard the conditions, because the locations in the three questions are famous regions of Japan. At that time MIAOW displayed the lower-level clusters inside the higher-level cluster which they zoomed in. Some of the examinees luckily zoomed in the correct lower-level cluster, and easily found the desired image. Others rotated to the XZ-plane and zoomed in the lower-level clusters corresponding to 2009, and quickly found the desired images. We found that most of the examinees operated MIAOW in a similar way, and quickly found the target images. Also, many examinees highly valued that the rotation mechanism was intuitive and easily comprehensive.

5. CONCLUSION AND FUTURE WORK

This paper presented MIAOW, a 3D image browser which represents hierarchically clustered photographs based on their shooting locations and times. MIAOW places the images onto the XY-plane assigning the two axes to their longitudes and latitudes, and onto the XZ- and YZ-planes assigning

the Z-axis to their shooting time. The paper introduced examples and experiments, and discussed how MIAOW is effective to search for photographs.

We received various comments and suggestions from the examinees. Below is the discussion how we will solve the issues as future work.

[Selection of representative images:] Some examinees mentioned that representative images were not always adequately selected. Our current implementation simply selects images which are the closest to the centers of the clusters, but obviously this is not always adequate. Some examinees also suggested preferable representative image selection criteria; such as shooting time, frequency of accesses to full-size images, and sizes or numbers of faces shot in the images. We would like to test such criteria.

[Size variation of representative images:] Some examinees mentioned that it might be sometimes difficult to discover small clusters, because our algorithm encloses small clusters by small rectangles, and therefore displays representative images of the small clusters relatively small. We had a user study of sizes of representative images in our image browsers, and actually found that the discovery of small clusters was easier when MIAOW displays all the representative images as equally-sized. However, equally-sized display of representative images causes another issue. The sizes of the rectangles enclosing the clusters are not always equal, because our browser displays thumbnails as equally-sized, and the numbers of thumbnails in the clusters are not always equal. If MIAOW would display representative images equally-sized, it would cause visual gaps when switching the display from representative images to rectangular regions filled with thumbnails. We need to carefully discuss and improve this issue as a future work.

[Smoothness of zooming operation:] Even MIAOW displays representative images equally-sized and shaped as rectangular regions of clusters, some of the examinees thought that the switch of the display might be too sudden. We will solve this issue by implementing smooth appearance/disappearance of representative images and thumbnails. Also, some of the examinees demanded ZUI (Zoomable User Interface) like operations that focus on the corresponding cluster by double-clicking a representative image. We would like to implement and test it as future work.

In addition to the above discussion, we would like to have experiments with larger image collections, and numerical evaluation of the layout results.

6. REFERENCES

[1] B. Bederson and B. Schneiderman. Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies. ACM Transactions on Graphics, 21(54):833–854, 2002.

[2] B. B. Bederson. Photomesa: A zoomable image browser using quantum treemaps and bubblemaps. In Symposium on User Interface Software and Technology, pages 71–80, 2001.

[3] J. Carriere and R. Kazman. Research paper: Interacting with huge hierarchies beyond cone trees. In IEEE Information Visualization 95, pages 74–81, 1995.

[4] C. Chen, G. Gagaudakis, and P. Rosin. Content-based image visualization. In International Conference on Information Visualization (IV00), pages 13–18, 2000.

[5] A. Gomi, R. Miyazaki, T. Itoh, and J. Li. Cat: A hierarchical image browser using a rectangle packing technique. In 12th International Conference on Information Visualization (IV08), pages 82–87, 2008.

[6] T. Hagerstrand. What about people in regional science? Papers in Regional Science, 24(1):6–21, 1970.

[7] R. Heilmann, D. A. Keim, C. Panse, and M. Sips. Recmap: Rectangular map approximations. In IEEE Information Visualization 2004, pages 33–40, 2004.

[8] T. Itoh, H. Takakura, A. Sawada, and K. Koyamada. Hierarchical visualization of network intrusion detection data in the ip address space. IEEE Computer Graphics and Applications, 26(2):40–47, 2006.

[9] T. Itoh, Y. Yamaguchi, Y. Ikehata, and Y. Kajinaga. Hierarchical data visualization using a fast rectangle-packing algorithm. IEEE Transactions on Visualization and Computer Graphics, 10(3):302–313, 2004.

[10] T. J. Jankun-Kelly and K.-L. Ma. Moiregraphs: Radial focus+context visualization and interaction for graphs with visual nodes. In IEEE Information Visualization 2003, pages 59–66, 2003.

[11] B. Johnson and B. Shneiderman. Tree-maps: A space filling approach to the visualization of hierarchical information space. In IEEE Visualization '91, pages 275–282, 1991.

[12] H. Kang and B. Shneiderman. Visualization methods for personal photo collections: Browsing and searching in the photofinder. In IEEE International Conference on Multimedia and Expo 2000, pages 1539–1542, 2000.

[13] M. Kraak. The space-time cube revisited from a geovisualization perspective. In 21th International Cartographic Conference, pages 1988–1996, 2003.

[14] J. Kustanowitz and B. Schneiderman. Meaningful presentations of photo libraries: Rationale and applications of bi-level radial quantum layouts. In ACM/IEEE Joint Conference on Digital Libraries, pages 188–196, 2005.

[15] J. Lamping and R. Rao. The hyperbolic browser: A focus+context technique for visualizing large hierarchies. Journal of Visual Languages and Computing, 7(1):33–55, 1996.

[16] J. A. Walter, D. Webling, K. Essig, and H. Ritter. Interactive hyperbolic image browsing - towards an integrated multimedia navigator. In ACM SIGKDD, pages 111–118, 2006.

[17] J. Wood and J. Dykes. Spatially ordered treemaps. IEEE Transactions on Visualization and Computer Graphics, 14(6):1348–1355, 2008.

[18] J. Yang, J. Fan, D. Hubball, Y. Gao, H. Luo, W. Ribarsky, and M. Ward. Semantic image browser: Bridging information visualization with automated intelligent image analysis. In IEEE Visual Analytics in Science and Technology, pages 191–198, 2006.