A Personal Photograph Browser for Life Log Analysis based on Location, Time, and Person

Ai GOMI
Ochanomizu University
2-1-1 Otsuka, Bunkyo-ku
Tokyo 112-8610 Japan
gomiai@itolab.is.ocha.ac.jp

Takayuki ITOH
Ochanomizu University
2-1-1 Otsuka, Bunkyo-ku
Tokyo 112-8610 Japan
itot@is.ocha.ac.jp

ABSTRACT

Image browsers are important and useful applications for retrieving images from personal photograph collections. Such browsers can be a life log analysis tool to explore the events of photograph owners. This paper presents a novel photograph browser consisting of two linked views. One of the views displays photographs clustered based on their locations and times, and the other displays people clustered based on their co-occurrences in the events. Specifying a photograph, the corresponding time the picture was shot and people in the photograph are highlighted. Specifying a time, corresponding photographs and people are highlighted. Specifying a person, associated photographs are highlighted, and their corresponding times are shown. The mechanism helps users to discover interested photographs and understand the events of photograph owners. This paper presents a real scenario and user experiment, demonstrating the effectiveness of the presented browser.

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces—Graphical User Interfaces (GUI); I.4.9 [Image Processing and Computer Vision]: Applications

General Terms

Algorithm, Design, Experimentation.

1. INTRODUCTION

The recent revolution of digital camera technology has resulted in much larger photograph collections. Many camera users store tens of thousands of photographs, and such photographs are often expected to be used as life logs. Image browsers are important and useful applications to view and inspect such photograph collections. We also expect that they are useful for the analysis of personal life logs. We expect that by creating a novel photograph browser we can quickly understand the timeline of the events of photograph owners, visited locations, their social networks, their favorite scenes to shoot as photographs, and various interesting information.

Generally, we need attributes associated with the photographs, so that we can smartly organize, browse, and retrieve them. We created a questionnaire asking "Which attributes do you think is important to retrieve specific photographs from your personal collections?" We obtained a result that time, location, keywords, features (e.g. color), and people (e.g. owner, photographer, people taken in the photographs) are important attributes for photograph retrieval [3]. Here, we do not think keywords and features are always reliable for photograph retrieval, though there have been many research works on content-based image retrieval (CBIR) techniques which deal with keywords and features. Reflecting the questionnaire results and the above discussion, our study focuses on using time and location information for browsing and the analysis of personal photograph collections. We developed an image browser MIAOW [3], which displays photographs hierarchically clustered based on their locations and times. MIAOW utilizes a view of 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.

In addition, we think a person's identity is also good information to retrieve personal photographs. We often look for photographs of specific people with location and/or time conditions, such as "I want all photographs taken of my children during travels", or "I want all photographs taken of my specific classmate during 2006 to 2008". It is not always easy to look for such sets of photographs, if the owners of the photographs simply store them by dividing into location- or time-based folders, or using existing commercial photograph management software or Web services. On the other hand, recent novel face recognition techniques have realized almost perfect recognition of people from photograph collections, and some of the techniques have been published as Web APIs. We therefore think it is valuable to develop personal photograph browsers that use personal identifies for interactive photograph retrieval.

This paper presents an extension of a personal photograph browser MIAOW [3], which displays photographs hierarchically clustered based on their locations and times, as well as people clustered based on their co-occurrences. The extended browser displays a collection of photographs in a view of a 3D space, as previous MIAOW displays. This paper calls this view LTView (Location and Time View). In addi-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SAC'11 March 21-25, 2011, TaiChung, Taiwan.
Copyright 2011 ACM 978-1-4503-0113-8/11/03 ...$10.00.

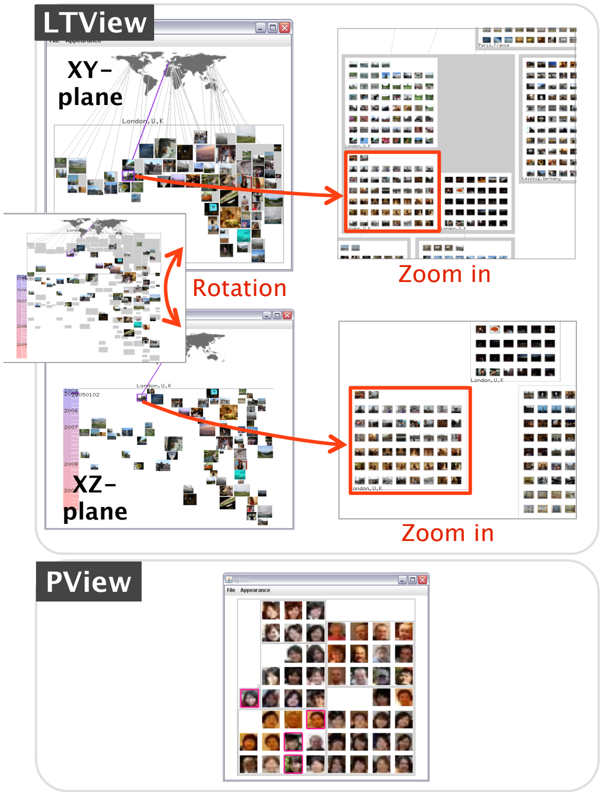

Figure 1: The presented photograph browser. LTView displays all photographs clustered based on their locations and times in a 3D space. LTView assigns location to X- and Y-axes, and time to Z-axis, and places all the clusters of the photographs. LTView first displays representative photographs of the clusters, and then displays all photographs in the clusters while zooming into the clusters. PView displays all faces of the people appeared in the photographs. The people are clustered based on their co-occurrences in the photographs.

tion, the browser provides another view to display people in the photographs. This paper calls this view PView (Person View). Figure 1 shows a snapshot of the presented browser.

As a preprocessing step, the technique first applies a clustering algorithm to divide the photographs based on their longitudes and latitudes. It then divides the photographs contained in each cluster based on their times. Consequently, a collection of photographs constructs a two-level hierarchical structure. At the same time, the technique extracts people from each of the photographs by applying a face recognition method. It then constructs a photograph-person matrix, and applies a clustering algorithm to the people, to construct groups of people according to their co-occurrences.

The presented browser places all clusters onto XY-, XZ-, and YZ-planes of a 3D space in LTView, representing the hierarchy of photographs as nested rectangular regions. Also, the browser places all person clusters as nested rectangular regions in PView, which has a linkage mechanism with LTView. If a user selects a photograph, it highlights the corresponding time in LTView, and people contained in the photograph in PView. If time is selected, it highlights the corresponding photograph in LTView, and people in the photograph in PView. If a person is selected in PView, it highlights photographs which contain that person, and the times of the highlighted photographs in LTView. We expect that the browser makes it easier for users to search for specific photographs, associated with memories of locations, times, or people.

This paper introduces a detailed algorithm used in the presented browser, a use case scenario using a real personal photograph collection, and the results from user experiments.

2. PHOTOGRAPH BROWSER BASED ON LOCATION, TIME, AND PERSON

2.1 Technical Overview

The proposed technique supposes that an unorganized set of photographs is given as an input dataset. We assume that location (longitude and latitude) and time are assigned to the photographs. Also, we assume that people taken in each of the photographs are specified by existing face recognition services. As a result, we describe the i-th photograph, p_i, as follows:

$$p_i = (\varphi_i, \lambda_i, t_i, f_{1i}, f_{2i}, f_{m_i i}) \qquad (1)$$

Here, φ_i is the latitude of p_i, λ_i is the longitude of p_i, t_i is the time of p_i, f_{ji} is the j-th person taken in p_i, and m_i is the number of people taken in p_i.

Also, we suppose functions to calculate the 3D position (x_i, y_i, z_i) of p_i in LTView as follows:

$$x_i = f(\lambda_i), y_i = g(\varphi_i), z = h(t_i) \qquad (2)$$

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

As preprocessing, our technique first divides photographs according to their x and y values, and constructs higher-level clusters of photographs. It then divides the photographs in each of the clusters according to their z values, and constructs lower-level clusters of photographs. This paper calls the lower-level clusters events. Consequently, it constructs a two-level hierarchy of the photographs. After the clustering process, it selects a representative photograph for each cluster. In the current implementation, each representative photograph is selected randomly . Meanwhile, the technique divides people according to their co-occurrences in the events.

The technique then places the photographs into LTView, and people into PView. It places the photograph clusters as nested rectangular regions onto XY-, XZ-, and YZ-planes in LTView. Also, it places the person clusters as nested rectangular regions onto a screen in PView. Here, we use a 3D space and an orthogonal coordinate system in LTView, where X- and Y-axis correspond to location, and the Z-axis corresponds to time. The technique first calculates the positions of clusters on the XY-plane. It represents the two-level hierarchy as nested rectangles, while avoiding overlapping the rectangular regions at the same level and attempting to minimize the display space. The technique finally calculates the positions of clusters on the XZ- and YZ-planes.

The technique is an extension of MIAOW [3], a previous version of our photograph browser. Existing features of MIAOW and new contributions in this paper are as follows:

Existing features of MIAOW: Photograph clustering (see Steps 1-1 and 1-2 in Section 2.2), and placement in LTView (see Section 2.3).

New contributions in this paper: Person clustering (see Steps 1-3 in Section 2.2), placement in PView (see Section 2.4), and interaction between LTView and PView (see Section 2.5).

2.2 Photograph and Person Clustering

For preprocessing, the technique constructs clusters of photographs and people. The clustering process consists of the following three phases, as shown in Figure 2, where Phases 1-1 and 1-2 have been presented in [3]:

Phase 1-1: Constructs high-level clusters of photographs based on their locations.

Phase 1-2: Divides photographs in each of the high-level clusters based on their times, and constructs low-level clusters (called "events" in this paper).

Phase 1-3: Constructs clusters of people based on their co-occurrence in each of the events.

Image clustering has been an active research topic. Naaman et al. presented an image clustering technique [13] that first divides images based on their times, and then divides again based on their locations. On the other hand, it is important to divide photographs by locations first in our technique, because our design aims to place all clusters of photographs based on their locations on the XY-plane.

Phase 1-1 simply calculates distances between any arbitrary two photographs on the XY-plane, then constructs a dendrogram based on the distances, and finally divides the photographs according to a predefined distance threshold. After the clustering, it calculates the average longitude and latitude values for each cluster, and use them for location-based rectangle placement as described in Section 3.3. We empirically determined the threshold of clustering as 0.01, which corresponds to (0.56234 degree) / (111 km) on the equatorial line. This phase may generate extremely large clusters, and in this case it is desirable to divide the photographs again using smaller thresholds. Also, this phase may put photographs near the borders of multiple clusters into undesirable clusters. In this case we may need to take times into account to adequately select the clusters of photographs near the borders.

Phase 1-2 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 photographs 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 Section3.3. Through empirical analysis, we determined the threshold of clustering as 3.374 days. However, we may need to adaptively control the threshold so that we can generate semantically adequate clusters.

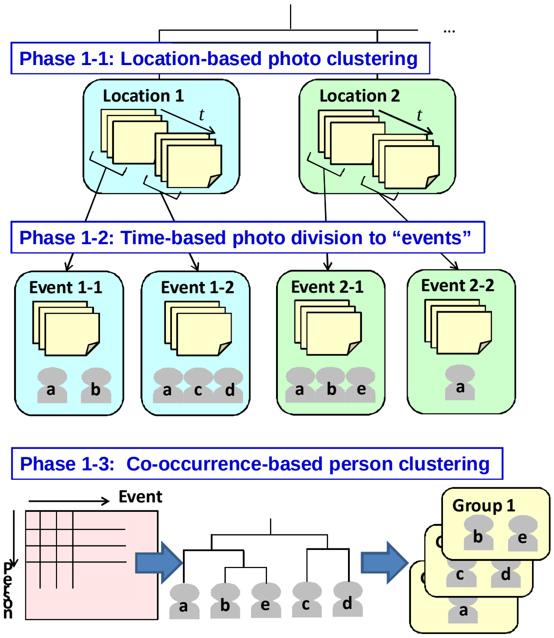
...
Phase 1-1: Location-based photo clustering
Location 1
t
Location 2
t
Phase 1-2: Time-based photo division to "events"
Event 1-1
a b
Event 1-2
a c d
Event 2-1
a b e
Event 2-2
a
Phase 1-3: Co-occurrence-based person clustering
Event
Person
a b e c d
Group 1
b e
c d
a

Figure 2: Photograph and person clustering.

Phase 1-3 first counts the number of photographs q_{ij} clustered in the i-th event, containing the j-th person. Here the j-th person can be represented as a n-dimensional vector $(q_{1j}, q_{2j}, ..q_{nj})$, where n is the number of events. It then constructs a dendrogram of the people according to the Euclidian distances between the n-dimensional vectors of arbitrary pairs of the people. Finally, it divides the people by cutting the dendrogram according to a pre-defined threshold value.

Our implementation uses Google Picasa API for face recognition. The implementation first sends all photographs to the API. The API clips facial images from the photographs, and returns groups of clipped faces. Here, each face group contains faces which are determined as the same person. The implementation assigns IDs to the groups, and picks up a representative face image for each of the sets. Our implementation does not assign labels (e.g. person names) to the face groups, but just shows representative face images of the groups on PView.

2.3 Photograph Placement in LTView

Photograph placement in LTView has been presented in [3]. LTView applies a Treemap-like space-filling hierarchical data visualization technique [9] to place the photograph clusters by the following phases:

Phase 2: Places a set of clusters as nested rectangles onto the XY-plane, reflecting the latitudes and longitudes of the clusters.

Phase 3a: Places a set of clusters as nested rectangles onto the XZ-plane, reflecting the times of the clusters, and X-coordinate values calculated in Phase 2.

Phase 3b: Places a set of clusters as nested rectangles onto the YZ-plane, reflecting the times of the clusters, and Y-coordinate values calculated in Phase 2.

2.4 Person Placement in PView

The technique applies a Treemap-like space-filling hierarchical data visualization technique [9] to place the person groups onto a screen in PView. It places a set of photographs of people onto a display space consisting of the following two phases:

Phase 4-1: Places a set of photograph thumbnails of the people using grid layout, and encloses them by a rectangular border. It repeats this process for all of the groups.

Phase 4-2: Packs all the rectangles corresponding to the person groups into a rectangular region.

The person groups are often meaningful (e.g. family, classmates, colleagues), and even more so if we can manually assign names to the groups. Our implementation can display such group names along the rectangular borders in PView.

2.5 Linkage between LTView and PView

We implemented an association mechanism between LTView and PView, as shown in figure 3.

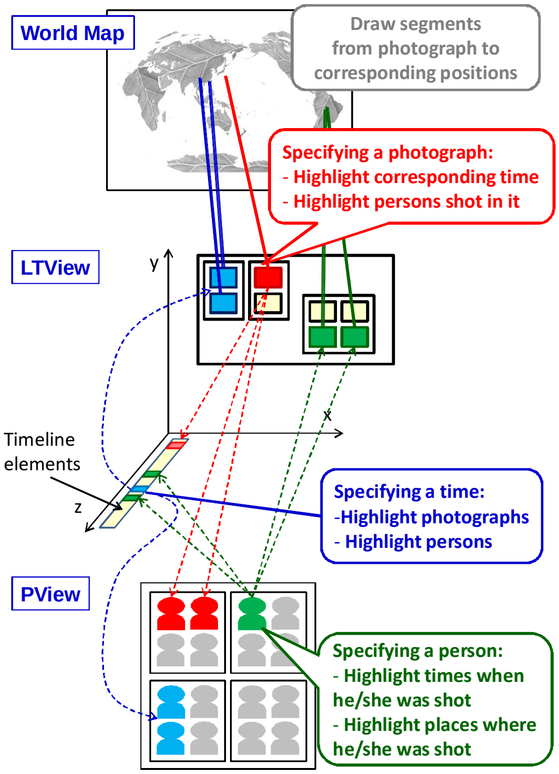
World Map
Draw segments from photograph to corresponding positions
Specifying a photograph:
- Highlight corresponding time
- Highlight persons shot in it
LTView
y
x
Timeline elements
z
Specifying a time:
-Highlight photographs
- Highlight persons
PView
Specifying a person:
- Highlight times when he/she was shot
- Highlight places where he/she was shot

Figure 3: Association between LTView and PView.

Our implementation of LTView places a world map behind the XY-plane. Also, it places a set of colored small rectangular elements, called "timeline elements" in this paper, which correspond to events and are sorted in the order of their times along the Z-axis. Here, all photographs and timeline elements in LTView are cursor-sensitive. Also, all people in PView are cursor-sensitive.

If a user picks a photograph by the cursor, our implementation takes the following actions: 1) it highlights the timeline element corresponding to the event to which the photograph belongs to, 2) it highlights the people contained in the photograph in PView, and 3) it draws a connection line from the photograph to the corresponding position on the world map.

If a user selects a timeline element by the cursor, our implementation takes the following actions: 1) it highlights a set of photographs belonging to the event corresponding to the specified timeline element, 2) it highlights the people contained in the photographs in the event in PView, and 3) it draws connections from the set of photographs to the corresponding positions on the world map.

If a user selects a person in PView by the cursor, our implementation takes the following actions: 1) it highlights a set of photographs including the person, 2) it highlights the timeline elements if their corresponding events contain photographs with the person, and 3) it draws connections from the set of photographs to the corresponding positions on the world map.

Our implementation draws borders of photographs or people in vivid colors as they are highlighted. Also, it draws timeline elements in vivid colors as they are highlighted.

3. EVALUATION

We implemented the technique 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 collection consisting of 4,804 photographs of size 100x75 pixels stored in JPEG format as thumbnails or representative photographs. The technique constructed 38 high-level clusters and 73 low-level clusters (=events) from the photographs, and recognized 52 people.

3.1 Examples

This section shows several examples of visualization results using three datasets. Figure 4 (Upper) shows an example of displaying the XY-plane. Our implementation displays a world map behind the XY-plane, and draws connecting lines between higher-level clusters and corresponding positions on the map. These results demonstrate that the technique places representative photographs of clusters without overlapping the photographs, attempts to minimize the display area, and preserves the geospatial adjacency among the clusters. Figure 4 (Lower) shows an example of displaying the XZ-plane.

An user evaluation of LTView is introduced in [3]. We asked them to search for specific images specified by various conditions, and measured the time it took them to locate the photographs. We compared LTView and an existing 2D image browser [4], and demonstrated that subjects took 3.5 times longer in average when using the existing technique, and 4 times longer in maximum.

3.2 Real-World Scenario

We visualized a set of photographs of a female student taken from 2004 to 2009, and reviewed her story.

Figures 5 and 6 show visualization results by clicking each year of the timeline elements during 2004 to 2009. Figure 5 (Upper) shows that she mainly took photographs with her classmates in Europe and a Pacific island in 2004. Figure 5 (Center) shows that she took many pictures with laboratory classmates and her family in 2005 in various locations throughout Japan. Actually, that year was her first year

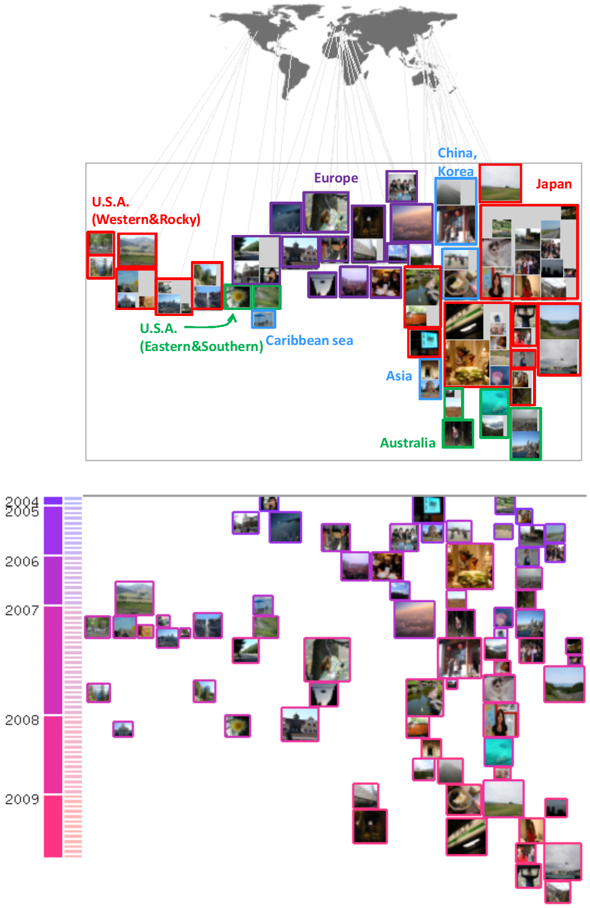

Figure 4: LTView. (Upper) XY-plane. (Lower) XZ-plane.

in her laboratory, and therefore she took many photographs since every event was novel and full of new experiences. Figure 5 (Lower) shows that she shared events with classmates again, and also with some of her family members in 2006. She went to various cities in Europe, and a Caribbean island for the first time. Figure 6 (Upper) shows that she went to many places in the USA in 2007, although she did not go to the USA in other years. On the other hand, she did not go to Europe during that year, although she went there in all other years. That year she attended fewer events with her classmates. Figure 6 (Center) shows that she attended many events with her new colleagues. Figure 6 (Lower) shows that she had a fewer events abroad. The result demonstrates that people and places in her events dynamically vary every year. Users can easily understand which year the photograph owner had many events at specific places or with specific people.

Figure 7 shows a visualization result by selecting people in a specific group. Figure 7 (Upper) shows that she did not have events with friend E after 2006. Figure 7 (Lower) shows that she had events with friend F three times, in 2008 and 2009. Comparing the two results, we understand that the three events in 2008 and 2009 were in her home country, and it seems that she could have invited friend E to the events as well. Discovering this fact, the photograph owner thought that she should plan the next event inviting friend E. We expect that this browser may bring opportunities to revive old friendships as shown in this example. Figure 8 shows a visualization result of exploring photographs of her sister. The result shows that the latest event with the sister was in 2007. At first the photograph owner did not understand why she did not have events with her sister since 2007, and what the latest recorded event was. Zooming in to the highlighted representative photograph and looking at the photographs in the highlighted event, she found that the event was her sister's wedding party. She realized that it was not easy to plan events with the married sister living far away, and thought of planning future events and inviting her sister. This zooming operation allows users to navigate event details by displaying all photographs of the events, and helps them to remember the details.

3.3 User Experience

We asked 12 subjects to experiment with the presented browser and the provided dataset. We briefly explained them how to operate the browser and what kind of photographs the dataset contains in several minutes, and gave 10 minutes to the subjects to freely experiment with them. Then, we asked them to describe the characteristics of the events they found while experimenting with the browser. Below are the common characteristics the subjects determined:

- She went to the USA only in 2007. She went to Europe in other years.
- She had many events with her colleagues since 2008.
- Her father joined every event in her hometown.
- She went to foreign countries with specific people. Other people joined the events only in her home country.
- She had events with classmates frequently.
- She had events with her family every year.
- A baby became part of the events since 2008.

We found that each examinee quickly determined correct characteristics. This result demonstrates that the presented browser is useful for life log analysis of specific people with their photographs. They felt that the browser is user friendly, and intuitively understood the characteristics by looking at the photographs with location, time, and person information. Especially they felt informed by the fact that people were clustered, because they could explore photographs group-by-group.

At the same time, we asked them to give us comments and suggestions on how to further improve the browser. Below are the feedback items from the subjects:

- It is informative if the number of participated events of a person is represented using colors.
- It is useful if the world map is clickable and events can be selected on the map.
- It is informative if countries or regions of clusters are represented by colors.
- It is useful if timeline units can be selected by season or month, not only by year or event.
- The view is clearer if unselected events are hidden.

Especially, first two items are suggested by multiple subjects. These suggestions can all be implemented by small extensions to our current implementation. We plan to implement them and re-evaluate the browser's usability.

4. RELATED WORK

4.1 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. These kinds of interfaces are not always effective for quickly finding all the desired images. More sophisticated user interfaces for image browsing can be valuable to more easily explore personal 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) [15]. These techniques are good at representing distances among images; however, it often overlaps many displayed images. 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. This problem can be solved by applying techniques for grid-based layout of multi-dimensional datasets [2].

Structures such as trees or graphs are also effective for photograph browsing. Bederson presented PhotoMesa [1], which places groups of images into rectangular subregions of display space. Kustanowith et al. presented a technique [12] that radially places clusters of images, and provides focus+context representation. We presented an image browser CAT [4], that represents clusters of images as nested rectangular regions. We also presented MIAOW, which displays location- and time-based clusters of photographs as nested rectangular regions [3].

4.2 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. 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 them.

LTView applies a Treemap-like space-filling hierarchical data visualization method based on nested rectangular packing [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, 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. Moreover, the technique has the capability to control positions of rectangles by referring to templates, which describe the ideal node positions, as described in Section 5 of [9].

4.3 Geography-based Visualization

LTView 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 [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. Wood et al. presented Spatially Ordered Treemaps [16] 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.

LTView places collections of photographs in a 3D space, where three axes of the 3D space correspond to longitude, latitude, and time. Such space has been well studied as a space-time cube, since time and location are quite relevant together while observing human behavior [5]. Several visualization works in space-time cube model [11] demonstrated the usefulness for observation of human life. Such space-time information has been recently applied to image browsing techniques [14].

4.4 Friendship Visualization

Friendship is interesting information to be visualized. We can obtain large-scale friendship network data, due to recent drastic evolution of social networking mechanisms. Some of the works implemented the visualization of people clusters analytically or interactively [6] [8]. Our technique also visualizes people clusters; however it constructs the clusters from their co-occurrences in personal photograph collections.

5. CONCLUSION AND FUTURE WORK

This paper has presented a personal photograph browser based on locations, times, and people. The technique first applies a clustering algorithm to a set of photographs based on their locations and times, and treats the clusters as events. It then clusters people based on their co-occurrences within the events. The browser provides two views: An LTView that places the photographs into a 3D space based on their locations and times, and a PView that visualizes clusters of people. We implemented a linkage mechanism between the two views, so that users can conveniently explore the photographs. The paper presents a real-world usage scenario of exploring a personal photograph collection, and a user experimental result.

Our current implementation of pre-processing is almost automated, except the selection of representative photographs. Some subjects also suggested preferable representative image selection criteria; such as shooting time, frequency of accesses to full-size images, and sizes or numbers of faces taken in the images. Also, image ranking [10] may be a good factor to select representative photographs. We will test such criteria.

In addition, we will perform experiments with larger photograph collections, and additional user evaluations.

Acknowledgements

We appreciate Dr. Kang Zhang of the University of Texas at Dallas, for helping us improve this paper.

6. REFERENCES

[1] B. B. Bederson. Photomesa: A zoomable image browser using quantum treemaps and bubblemaps. In

Symposium on User Interface Software and Technology (UIST2001), pages 71–80, 2001.

[2] R. D. de Pinho, M. C. F. de Oliveira, and A. de Andrade Lopes. An incremental space to visualize dynamic data sets. Multimedia Tools and Applications, 50(3):533–562, 2010.

[3] A. Gomi and T. Itoh. Miaow: A 3d image browser applying a location- and time-based hierarchical data visualization technique. In Advanced Visual Interface (AVI2010), pages 225–232, 2010.

[4] 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.

[5] T. Hagerstrand. What about people in regional science? Papers in Regional Science, 1970.

[6] J. Heer and D. Boyd. Vizster: Visualizing online social networks. In IEEE Symposium on Information Visualization 2005, pages 33–40, 2005.

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

[8] N. Henry and J.-D. Fekete. Matrixexplorer: a dual-representation system to explore social networks. IEEE Transactions on Visualization and Computer Graphics, 12(5):677–684, 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] Y. Jing and S. Baluja. Visualrank: Applying pagerank to large-scale image search, hierarchical data visualization using a fast rectangle-packing algorithm. IEEE Transactions on Visualization and Computer Graphics, 10(3):302–313, 2004.

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

[12] 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 199–196, 2005.

[13] M. Naaman, Y. J. Song, A. Paepcke, and H. Garcia-Molina. Automatic organization for digital photographs with geographic coordinates. In ACM/IEEE Joint Conference on Digital Libraries, pages 52–62, 2004.

[14] D.-S. Ryu, W.-K. Chung, and H.-G. Cho. Photoland: a new image layout system using spatio-temporal information in digital photos. In ACM Symposium on Applied Computing, pages 1884–1891, 2010.

[15] 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.

[16] J. Wood and J. Dykes. Spatially ordered treemaps. IEEE Transactions on Visualization and Computer Graphics, 14(6):1348–1355, 2011.

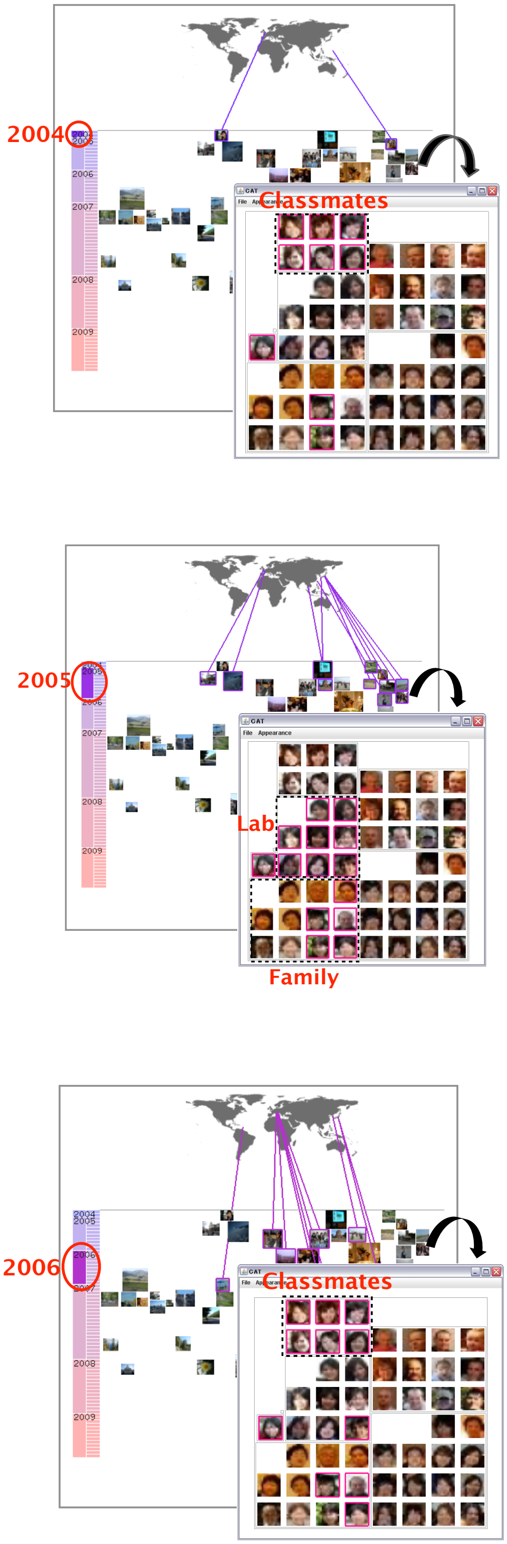

Figure 5: Scenario(1). Selecting each year from the timeline elements, during 2004 to 2006.

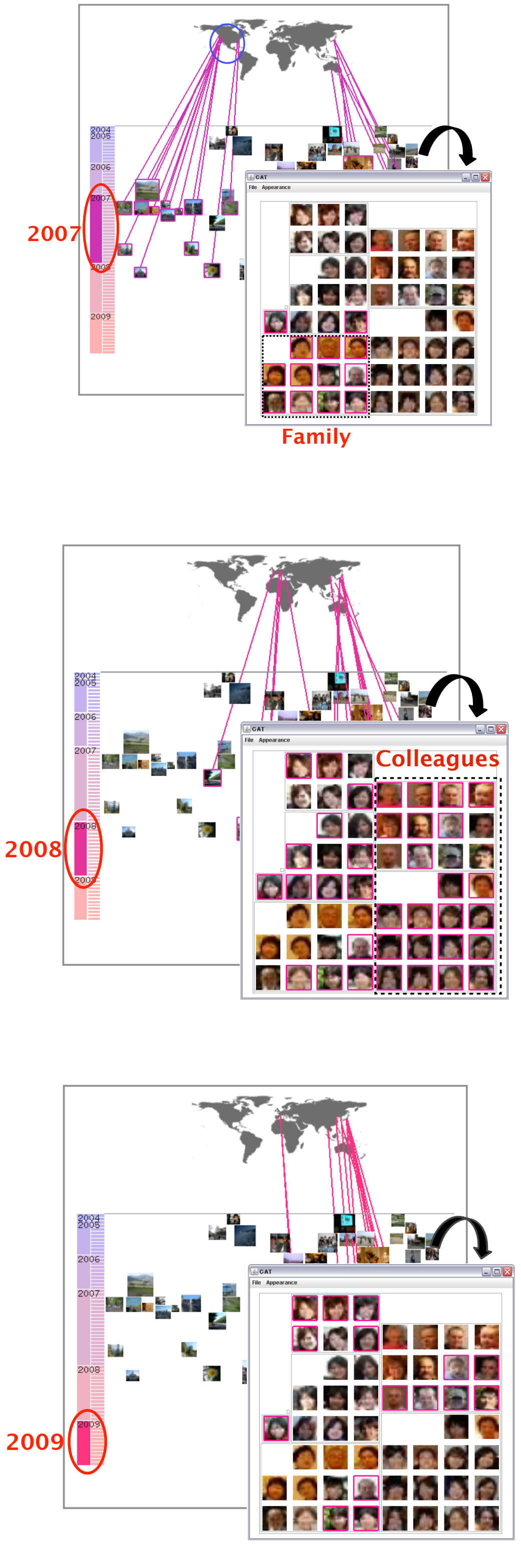

Figure 6: Scenario(1). Selecting each year from the timeline elements, during 2007 to 2009.

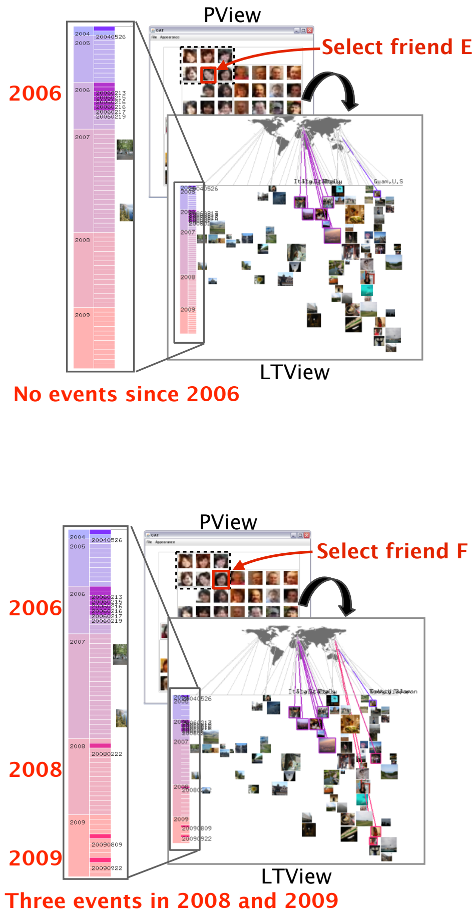

Figure 7: Scenario(2). Events with classmates.

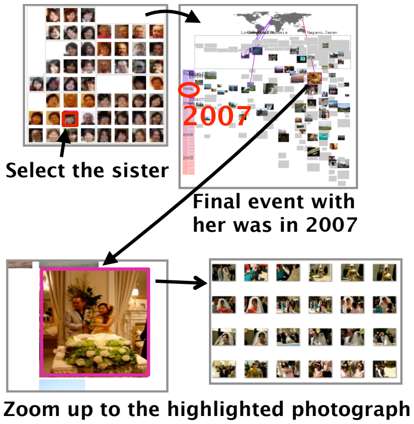

Figure 8: Scenario(3). Final event of a specific person.