MosaiZ: Interactive Image Mosaics

Research Report

Romain Vuillemot Béatrice Rumpler

Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205
F-69621, Lyon, France
romain.vuillemot@insa-lyon.fr

Abstract

In this report we introduce Interactive Image Mosaics (called MosaiZ), a pure content-based interaction model to make current image mosaics interactive and infinitely zoomable. A MosaiZ is a non-linear interaction space, where users can select a source image that will become the *target* image, and so on. Interactions aim at helping users to visually formulate their queries (i.e. images they are looking for) using *target* image features. This way, users visually express their needs in an understandable way for the machine, with no sign or code. We introduce a Graph Scene data structure to connect image mosaics together. We detail an interaction model to browse it, and finally give implementations details of our first prototype.

1 Introduction

It has never been so easy to capture images and make them available to a global community of people. Meanwhile, studies [16, 18] show that people don't invest much time classifying their collections, they prefer to rely on automatic metadata such as photo transfer time or cameras timestamps. But even if metadata are present, they may not be reliable or consistent enough over very large image datasets. Also, user needs may be too fuzzy to be expressed using codes or symbols of the index system. Thus, a content based exploration approach is the only solution. Consequently, leveraging human visual and interaction ability are challenges to tackle. Since screens size remains limited and new interaction device are not widely spread, a better conceptual use of human visual bandwidth has to be found. We think we need to go one step beyond image tiles [16], with an efficient image overload harnessed by a strong control over the interface. This conviction seems counter-intuitive because introducing a visual overload, but exploring new disruptive ideas, such as reification of real world metaphor [4] is a path worth to follow.

1.1 Motivation

Informal observations of people behavior looking at Image Mosaics showed that they unconsciously engage attention to them. This attention span is crucial and priceless in our information society, where relentless streams of information are unwillingly dumped into our brain, mainly through our eyes. Image mosaics are part pieces of art, part technique; they have traveled through the ages, and recently regained interest during last decade with the rise of the digital era. Even if this images layout is sophisticated, everybody knows how to get information out of it, regardless age, culture or education. Images can be fetched in a 2D way as tiles or *source* images, clustered by color or shape, according to the big or *target* image that is visible by stepping backward.

1.2 Image Mosaics Layout

During the last decade, mosaics layouts have been studied and improved using various angles (shape optimization, algorithm optimization, etc.). Plenty of contributions are available, from SIGGRAPH [19] papers to student projects. Recently introduced Jigsaw Image Mosaics [12] are based on a dataset that has shape similarities and shape homogeneity. Collages are another form of mosaics with pieces having different shapes that can be rotated and overlap each other, such as we took a scissor and cut magazines to make a new pictures. Image mosaic softwares are freely available such as KMosaic [13] and MetaPixel [9]. Finally QuadTree based on image segmentation are well known and implemented technique that can be integrated as a grid. Existing attempts to make image mosaics interactive are limited, like Gettyimages [7] and Infinite Photograph [10] which provide step by step image navigation. A screen saver feature in MacOS X [8] is available and uses basic recursive navigation in mosaics, with no interaction but passive watching. As far as we know image mosaics have never been studied as a browsing metaphor.

1.3 Goal of the report

In this report we focus on making image mosaics layouts [6] interactive. Image mosaics can pack vast amounts of images with minimal degradation and low cognitive overload due to data continuity. *Source* images (making the mosaic) are indexed based on a matching between their very own content (shape, feature, color gradient, etc.) and a *target* image (global image). Image mosaics provide a space filling organization with an optimal use of screen display: every single pixel holds information on both *source* and *target* image. They induce no clutter or image overlapping; and even with many images, they remain strictly adjacent. Finally, overview and details are given both at a glance. Our hypothesis is that image mosaics fill major design recommendations, such as:

1. **Consistent layout due the recursive nature of zoomable mosaics.** The transitional step (zoom), from *source* to *target* (or the opposite) is a linear purely geometric scaling -and no hyperbolic deformation such as Fisheye -magnification is the same for every image.

2. **Scalability.** A *target* image is summarizing the *source* dataset. In the scenario where *source* images are one pixel by one pixel, nearly 2 million images can be packed in a 1600 x 1200 pixel image, if physical pixels (smallest point that can be addressed on the screen) equal image pixel. Also mosaics can easily be sliced into manageable chunks of images, allowing progressive rendering.

3. **Users know how the system works.** Because image mosaics are based on visual perceptions properties, there are no cultural or context based knowledge (such as classification or clustering). Also the recursively predictable layout allows users to follow a navigation strategy they can easily model in their heads.

2 MosaiZ

We call a zoomable image mosaic layout a MosaiZ. From users' perspective a MosaiZ consists in a sequence of image mosaics in which users can zoom in, pan and rotate. In other words a MosaiZ is a non-linear interaction space, where users can select a *source* image that will become the *target* image, and so on (Fig. 1). Interactions aim at helping users to visually formulate their queries (images they are looking for) using *target* image features. This way, users express their needs in an understandable way for the machine, with no sign or code.

2.1 Generating a MosaiZ Scene Graph

The backbone of a MosaiZ is a scene graph G_{scene} (Fig. 4) (similar as a space scale diagram) connecting images altogether. Every node m of G_{scene} is an image mosaic, and is connected to all sources that are composing that mosaic. Generating G_{scene} is

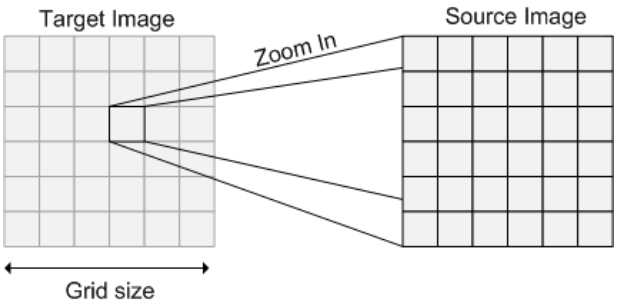

Figure 1: *Target* images are cropped regarding a grid size. Zooming into the grid square magnifies the *target* sub-image (*source* image), which will become the next *target* image

straightforward: 1) Select a dataset and a *target* image m_{root} as scene graph root 2) select a grid size and 3) generate mosaics for each *source* image and go to step 1) (i.e. *source* image becoming *target* image). Post processing such as checking G_{scene} connectivity is required but introduces constraints into the mosaics generation process that results in huge additional processing time. Some tricks (forcing image choice regardless matching function) may be employed to gain connectivity while not disrupting the visual quality of the result.

2.2 Basic Interactions Design

Using a Zoomable User Interface (ZUI) is straightforward and successful way to deal with image management [2]. It has also been used in hierarchical navigation such as in Treemaps [5]. Zooming into mosaics is based on the property that images are made of images themselves, becoming an infinite two dimension plane with no space and distance limit. Semantic zoom [14] is automatic: when the *source* image miniature becomes too small, they are replaced by *target* image pixels (or the opposite).

We combine interaction techniques in a meaningful and consistent way to help users to make the best decision at every node of G_{scene}. Interactions can now be expressed as functions applied to G_{scene} and nodes. We now introduce design details regarding continuous and discrete interactions; then we add extra features to efficiently navigate in MosaiZ.

2.2.1 Continuous interactions using the mouse

A classic panning (translation) and zooming (with uniform scale changes) navigation allow drilling down and rolling up. A preview function is introduced: when the mouse exits the application window, original image is displayed to ease the feature seeking process. When the mouse enters back, the mosaic is on but with a 50% alpha level coupled with the original image (this is a common trick used in image mosaics) that helps to better perform visual query. This cheat feature (so called in the Metapixel library used in the implementation) helps to better make decision on which *target* image to acquire with no zooming. Also when the mouse hovers the *target* image, original *source* images are displayed around pointer vicinity in a transitional way.

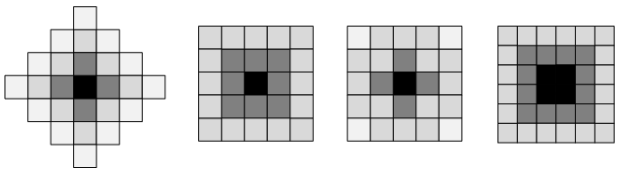

Figure 2: Mouse pointer is a hover layer that helps users to get better *source* images resolution. Multiple shapes are available with variations in size and transparency level (the darker the square is, the more *source* image is visible).

2.2.2 Discrete interactions using the keyboard

Arrow keys help to pan around. But *target* image acquisition is hard since the system has no information about users' visual focus. Our solution is to use the numeric pad; and for every key press users progressively select areas of the screen (Fig. 3).

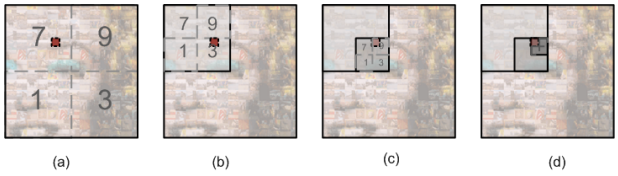

Figure 3: Image *target* acquisition through the keyboard is performed by pressing numeric pad keys (7, 9, 1 and 3). Keys are tied to one quarter of the screen. Every time a key is pressed, the region available is a quarter of the previous one.

2.2.3 Flipping

This is always convenient for users to remain at a same level of zoom and navigate in breadth as opposed as in depth. Authors in [5] provide a flipping interaction that we use and combined to Automatic

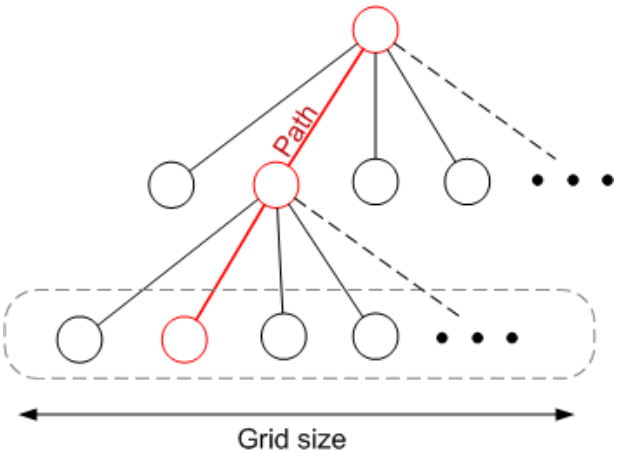

Figure 4: Graph Scene (G_{scene}) holds *source*/*target* links between mosaics. A node (circle) is a mosaic is connected to other mosaics by means of edges (plain line). The grid size can be seen as the number of children for every node. User activity is a path in the graph.

Touring (described below) preventing repetitive actions.

2.2.4 Resizing the Mosaic Grid

Dynamically changing the mosaic grid size is a powerful feature since it will allow changing the query bounds. Grid resizing is done by using mouse scroll and pressing control key at the same time. Available grid sizes have to be set during the image mosaics generation step since it is a time consuming process for fine grids.

2.3 Dynamic Interactions

The user's activity is tightly coupled with time. This activity ranges from data (track, history) to knowledge production (bookmark, export). We want to enhance our design with features supporting long term efforts [17] and to maximize activity by reducing mechanical efforts. All the interactions involving previous/next steps will be said to be *dynamic*.

2.3.1 History

It is strongly advised that systems must give users the opportunity to make mistakes and go backward [22]. We added right click to go to the previous state (previous scene graph node). But also visual clues such as red squared images or making *target* image reddish shows previously visited ones. History or statistics can also be exported such as a list or a graph to be browsed in another interactive environment, and reset to get a fresh start.

2.3.2 Rendez-Vous Point

Every exploratory task involve non predictable, confuse and irrational behaviors. Sometimes users just want to find by serendipity or to have fun with the tool, rather than following beaten tracks. For that purpose, we allow users to set a Rendez-Vous Point: users press a key, explore, and by releasing the key will go back to the Rendez-Vous. No history is recorded in this incognito mode. The way back to the Rendez-Vous is a fast automatic animation to quickly remember what has been done during that recreation time. Letting users pressing the key is important to remind them that this is just a short term image exploration process.

2.3.3 I am Feeling Lucky

Systems -and especially new ones- should not let users stuck in a situation where they can't do anything or don't know what the next step is. In a MosaiZ, spatial freedom and zoomable process require users' efforts. Situations such as desert fog [11] -being lost in the multi-scale space with not enough information to take a decision- have to be tackled. A system-based approach is implementing this function and accessible by double clicking (instead of single click). The system will look for the next image, regarding a user-selected heuristic that may -for instance- be browsing new images only. It may also be seen as a shortcut to repetitive actions users perform. Other heuristics such as browsing by chronological order, image size, and social popularity would take advantage of metadata attributes and next image may hold the Guess function. A pre-processing of those heuristics can be performed and resulting in weighting G_{scene} edges that are read by the client interactive application (see Implementation notes section).

2.3.4 Dashing

A second way to automatically explore the data space is a brute straightforward zooming, where users are setting only the direction by means of arrow keys or mouse moves. The result is a time dependent interface such as Dasher [21] -which is using letters flow to compose phrases. This way we reduced users' mouse amplitude movement. In that case user's attention is required and is parameterizable. We introduce a time factor t that is used to define zooming speed:

$t_{touring} = s \times (t - t_{start})$ with time-lapse visualization ($s < 1$), real-time visualization ($s = 1$) or slow motion visualization ($s > 1$).

2.3.5 Other grids and pointer shapes

Our modular and service oriented implementation architecture is designed in a way that including those advances will not require upgrading client application.

3 Implementation notes

We implemented MosaiZ in a twofold approach (Fig. 5).

3.1 Local Java application

Using Picollo2D (open source project based on [3]), a direct-manipulation graphics library that supports constructing ZUI, the application is a basic frame containing a layer on which three layers (mosaics) are tied at a time (previous, current and next mosaic if known). Pointer layer is added and tied to mouse cursor, to display child images according to the pointer location, size and shape. A set of event listeners trigger node rotation when users get close enough to an image (scale > 1) or click on it. We used mipmaps (pre-calculated, optimized collections of images) as a cache to quickly provide images into the application, and to prevent aliasing effects. MosaiZ implementation code contains a dozen of classes and a total of 1200 lines of code and 2 shell scripts. A modular implementation makes it easy to add new touring heuristics or other mosaic grids, and users select them in the control panel window (Fig. 6). Memory management was the major concern since we want to infinitely explore the image space.

3.2 Remote server generating the scene graph

A remote server [20] generates G_{scene} offline (using Metapixel 1.0.2 library) and it took 2h32mins to generate all the 6 x 250 mosaics (on a commonly available computer system 3Ghz Intel PCs with 2Go RAM and running Windows XP). G_{scene} is available as a resource over HTTP with no firewall restrictions and keeps track of users' activity (session, frame rate, grid picking, etc.). The MosaiZ client interface is implemented using Picollo2D (open source project based on [3]), a direct-manipulation graphics library that supports constructing ZUI as a Java application.

Based on image descriptors uploaded the by the client application, the server is keeping track of users' activity [20]. Since image mosaic generation is not fully predictable, we wanted to avoid bottlenecks in user experience, and generated the graph scene once for all in an offline mode. We used Metapixel to generate the mosaics with different grid sizes. Activity logs (indicators such as path in the MosaiZ, frame rate, session times, features usage, etc.) are uploaded on the server for analysis after every session, in order not to bias the performances.

3.3 Dataset Preparation

We started using a relatively small dataset to circumnavigate technical limits (memory and mosaics generation). We used a dataset of 250 photos from New York City with heterogeneous subjects (buildings, animal, people, textures, indoor, outdoor, flash, no flash, landscape, etc.). Pictures are homogeneous in resolution and compression, taken by the very same person using a Nikon D80 camera. Pictures were rotated if needed, resized and cropped. We selected 6 different grid sizes (10, 16, 20, 25, 40, 80 pixels). It took 2h32mins to generate all the 6 x 250 mosaics (on a commonly available computer system 3Ghz Intel PCs with 2Go RAM and running Windows XP). All those dataset preparation steps are performed using batch scripts.

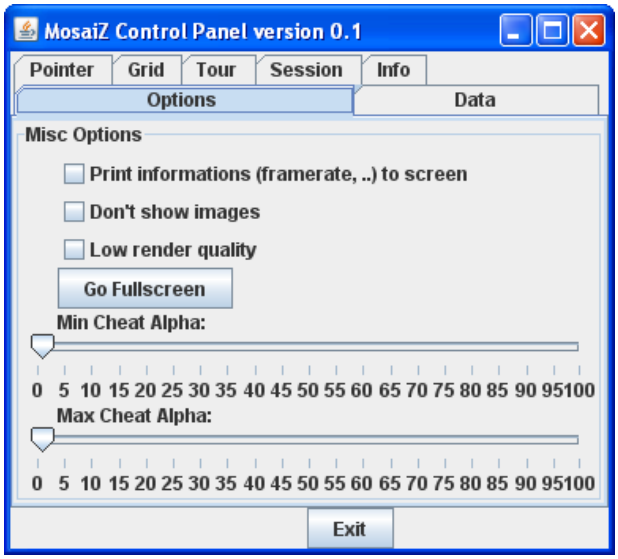

Figure 6: Control Panel allows users to personalize the MosaiZ and experiments modules that can be added to evaluate new heuristics

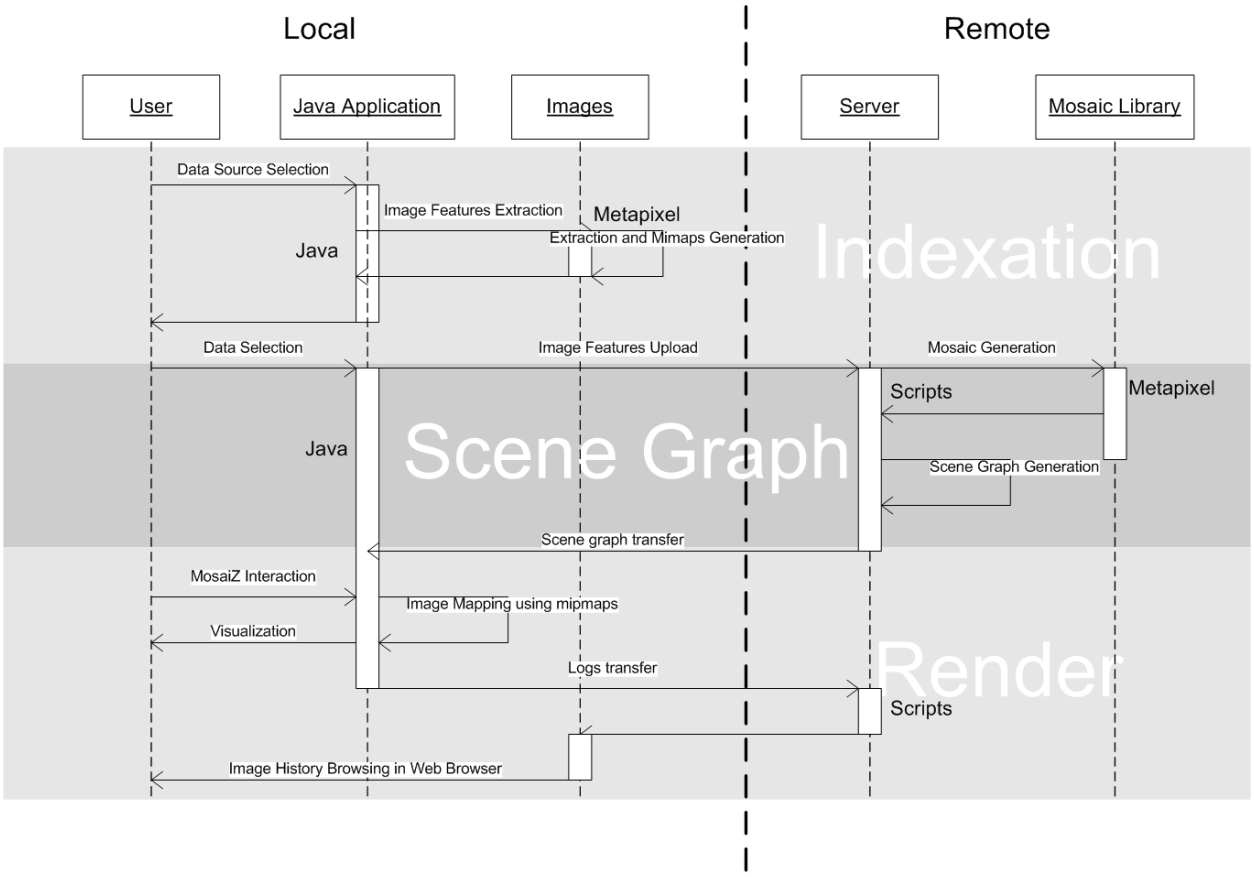

Figure 5: Twofold MosaiZ implementation architecture. Local interactive client remains on users' side (left). A connection to a remote server (right) allows (layers from top to bottom): session opening, features upload, scene graph download and finally users interactions. Logs are uploaded when session ends, and users access to their browsing history using an ordinary web browser.

4 Preliminary experiments and discussions

4.1 Usability Testing

From a usability perspective, we closely worked with a digital artist which is a knowledgeable and experienced user of digital image management. Our expert's major satisfaction was not starting his session from scratch, but having a basic ground of exploration (information overload) and a feeling of freedom by always being surrounded with images (thus preventing Desert Fog [11] -being lost in the multi-scale space with not enough information to take a decision). Regarding content findings, MosaiZ are efficient to visually explore dual tones images (e.g. landscape pictures such as on the cover figure). Our expert considered MosaiZ to be a good storytelling environment since there are no data gaps or discontinuities, and can provide nice slideshow transitions. MosaiZ would be a good system to explore the rough output from another system (such as the Flickr query result in the Introduction). The major limit is the very long mandatory pre-processing offline mode that can not make MosaiZ a real time exploratory tool. Also, it is efficient with large results dataset (that add diversity to the mosaic) but not below a low number of results. Our expert requested a presentation mode, with content-based transitions, using navigation logs or annotated images that could be embedded in web applications or available on mobile. The uniform scale changes were appreciated with no distortion. A controlled experiment with other image browsing softwares is required to validate this claim and compare performances. Controlled experiment with non-experienced users and content-based browsing softwares are also required to validate claims and compare performances.

4.2 Technical Validation

From a technical perspective, our first result is the architecture validation (a light local client loosely coupled to a remote server) that gives a reactive interface even if remote pictures are not available (if an image is temporary missing, there is a default replacement). Performances were a major concern in our approach. While interacting with a visual display, the frame rate is a good indicator about the smoothness of an application.

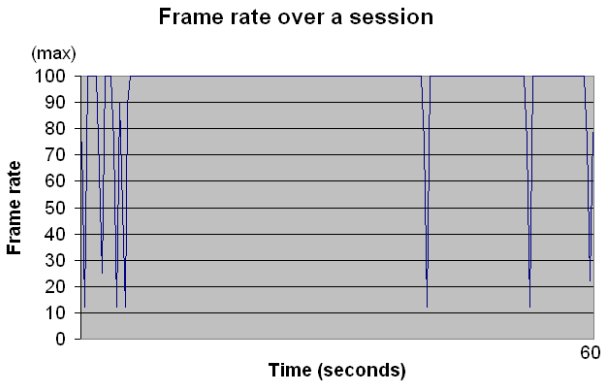

Figure 7: Frame rate during a randomly chosen 60-second session sample. Every drop to 11s, 12s, indicate zooming interaction. 100 is the maximal frame rate and is reach for all the other activities.

We noticed below 15 frames per second animation is not smooth anymore. Results (Fig. 7) show some drops in the rate that are tied to a user zooming action to a new mosaic. Note that the application remains working even if remote pictures are not available (i.e. the display is not refresh) letting users the nice feeling to still keep control over the interface.

5 Perspectives

Our next iteration is technical, to make the architecture scalable up to million of images visible at a time. We will rely on our systematic user evaluation that let us to identify bottlenecks, better understand and narrow down the design space (i.e. possible path in G_{scene}). Thanks to our modular and service oriented architecture, scalability will concern server side only without upgrading client application. Similarly, new additions (new segmentations, new mosaic layouts, etc.) won't require client updates. As long term perspective, we also want to explore how the interaction model may be extended to *Droste effect* or *mise en abyme* which are other kind of recursive pictures. In general we want to apply our systematic user evaluation and design exploration to make Informative Art [15] interfaces interactive as well.

Collaboration and social experience . A MosaiZ instance (i.e. a path in G_{scene}) can be annotated and shared, such as a photo presentation. Path adjustment is made possible, by unchecking unwanted photos. Using other modalities is an interesting track of study, such as vocal annotation or music. Experiences gathered through massive MosaiZ usage may reinforce the feeling lucky heuristic (G_{scene} edges weight). This social feedback may also be integrated into static social image mosaics (e.g. a color code is showing the most frequently clicked targets and result in heat map). Storytelling [1] is a feature expected by image authors and so it was by our expert.

MosaiZ communication and presentations (based on user's personalization) will be the next challenges we will tackle. We will also enhance the dataset preparation steps which are crucial in making G_{scene} as optimal as possible.

References

[1] Marko Balabanović, Lonny L. Chu, and Gregory J. Wolff. Storytelling with digital photographs. In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 564–571, New York, NY, USA, 2000. ACM.

[2] Benjamin B. Bederson. Photomesa: a zoomable image browser using quantum treemaps and bubblemaps. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 71–80, New York, NY, USA, 2001. ACM.

[3] Benjamin B. Bederson, Jesse Grosjean, and Jon Meyer. Toolkit design for interactive structured graphics. *IEEE Trans. Softw. Eng.*, 30(8):535–546, 2004.

[4] Alan F. Blackwell. The reification of metaphor as a design tool. *ACM Trans. Comput.-Hum. Interact.*, 13(4):490–530, 2006.

[5] Renaud Blanch and Eric Lecolinet. Browsing zoomable treemaps: Structure-aware multi-scale navigation techniques. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1248–1253, 2007.

[6] Adam Finkelstein and Marisa Range. Image mosaics. *Lecture Notes in Computer Science*, 1375:11–??, 1998.

[7] http://10ways.gettyimages.com/usa/information_interactive.htm. gettyimages. 2009.

[8] http://www.apple.com/macosx/features/300.html. Mac OS X Leopard. 2009.

[9] http://www.complang.tuwien.ac.at/schani/metapixel/. Metapixel - A Photomosaic Generator. 2009.

[10] http://www.thegreenguide.com/infinite photograph. Infinite Photograph. 2009.

[11] Susanne Jul and George W. Furnas. Critical zones in desert fog: aids to multiscale navigation. In *UIST '98: Proceedings of the 11th annual ACM symposium on User interface software and technology*, pages 97–106, New York, NY, USA, 1998. ACM.

[12] Junhwan Kim and Fabio Pellacini. Jigsaw image mosaics. *ACM Trans. Graph.*, 21(3):657–664, 2002.

[13] Kihwan Kim, Irfan Essa, and Gregory D. Abowd. Interactive mosaic generation for video navigation. In *MULTIMEDIA '06: Proceedings of the 14th annual ACM international conference on Multimedia*, pages 655–658, New York, NY, USA, 2006. ACM.

[14] Ken Perlin and David Fox. Pad: an alternative approach to the computer interface. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 57–64, New York, NY, USA, 1993. ACM.

[15] Johan Redström, Tobias Skog, and Lars Hallnäs. Informative art: using amplified artworks as information displays. In *DARE '00: Proceedings of DARE 2000 on Designing augmented reality environments*, pages 103–114, New York, NY, USA, 2000. ACM.

[16] Kerry Rodden and Kenneth R. Wood. How do people manage their digital photographs? In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 409–416, New York, NY, USA, 2003. ACM.

[17] Ben Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1997.

[18] Ben Shneiderman, Benjamin B. Bederson, and Steven M. Drucker. Find that photo!: interface strategies to annotate, browse, and share. *Commun. ACM*, 49(4):69–71, 2006.

[19] Nicholas Tran. Generating photomosaics: an empirical study. In *SAC '99: Proceedings of the 1999 ACM symposium on Applied computing*, pages 105–109, New York, NY, USA, 1999. ACM.

[20] Romain Vuillemot, Béatrice Rumpler, and Jean-Marie Pinon. *Enterprise Information Systems 10th International Conference, ICEIS 2008, Barcelona, Spain, June 12-16, 2008, Revised Selected Papers*, chapter Dissection of a Visualization On-Demand Server, pages 348–360. Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, April 2009.

[21] David J. Ward, Alan F. Blackwell, and David J. C. MacKay. Dasher - a data entry interface using continuous gestures and language models. In *UIST*, pages 129–137, 2000.

[22] Alan Wexelblat. History-based tools for navigation. In *HICSS*, 1999.