International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:5, No:9, 2011

Photo Mosaic Smartphone Application in Client-Server Based Large-Scale Image Databases

Sang-Hun Lee, Bum-Soo Kim, Yang-Sae Moon, and Jinho Kim

***Abstract*—** In this paper we present a photo mosaic smartphone application in client-server based large-scale image databases. Photo mosaic is not a new concept, but there are very few smartphone applications especially for a huge number of images in the client-server environment. To support large-scale image databases, we first propose an overall framework working as a client-server model. We then present a concept of *image-PAA* features to efficiently handle a huge number of images and discuss its lower bounding property. We also present a best-match algorithm that exploits the lower bounding property of image-PAA. We finally implement an efficient Android-based application and demonstrate its feasibility.

***Keywords*—** smartphone applications; photo mosaic; similarity search; data mining; large-scale image databases.

I. Introduction

OWING to advances in mobile processors and wireless internet technologies, we can execute most of PC-based applications also in portable smartphones at anytime and anywhere [1]. Due to the portability of smartphones, we can easily use a large number of useful applications deployed through App stores or markets [2, 3]. However, smartphones still have a limitation of less memory space and low computing power, and this makes it difficult to support a large volume of multimedia data in the complicated applications.

In this paper we propose a working framework for the photo mosaic [4] smartphone application, which is efficiently working in large-scale image databases, and we also implement its prototype application. To overcome the memory limitation, our working framework is designed as a client-server model. Client and server functions are follows:

- **Client**: The first function is to get a *target image* given by a user as an input, to extract appropriate RGB histograms from grid-cells of the target image, and to send the cell-based RGB histograms to the server. The second function is to get a mosaic image from the server and to display it to the user.
- **Server**: It first constructs a mosaic image by comparing the target histograms with its large number of (preprocessed thumbnail) images stored in the database, and it then sends the mosaic image to the client.

Sang-Hun Lee, Bum-Soo Kim, Yang-Sae Moon, and Jinho Kim are with Dept. of Computer Science, Kangwon National University, Kangwon 200-701, South Korea (e-mail: {sanghun, bskim, ysmoon, jhkim}@kangwon.ac.kr, fax: +82-33-250-8440.).

We here note that similarity search in a huge number of images is a major performance bottleneck. Thus, to achieve the higher search performance, we present a novel concept of *image-PAA* as a dimensionality reduction function [5]. Using its lower bounding property we then implement an efficient photo mosaic application that works even in millions or tens of millions of images.

II. Related Work

Photo mosaic, proposed by Silvers et al. [4], is an automatic (or semi-automatic) technique that converts an original photo image to a corresponding mosaic image by combining a number of thumbnail images (i.e., reduced small size images). As shown in Fig. 1, a photo mosaic technique generally consists of three representative steps: target division, similarity search, and mosaic generation.

1. **Target division** divides an input (i.e., target) image to a small-sized fixed number of grid-cells.
2. For each grid-cell, **similarity search** finds its best-match (thumbnail) image from the database, which will replace the grid-cell.
3. **Mosaic generation** constructs a photo mosaic by replacing original grid-cells with their corresponding best-match images.

Target division Similarity search Mosaic generation

Fig. 1 Three steps of generating a mosaic image.

There have been several efforts to improve the photo mosaic approach by Silvers et al. [4], and the followings are the three representative ones. First, Finkelstein and Range [6] adopt various shapes rather than rectangles for organizing grid-cells and use color adjustment techniques. Second, Hoff et al. [7] propose a novel photo mosaic solution that generates fine-grained grid-cells by using weighted Voronoi diagrams. Third, Tran [8] focuses on measuring the quality (i.e., the effectiveness) of photo mosaics and evaluating the performance of existing approaches. In [8], Tran points out that efficiency of photo mosaic algorithms is an important issue in large-scale image databases and presents various criteria to evaluate those

algorithms. All these previous solutions, however, are based on the PC-based stand-alone environment with small-scale databases, and to our best knowledge, there is no attempt to support smartphones in the client-server environment with large-scale databases. Thus, in this paper we design and implement a smartphone application that works in large-scale image databases on the client-server model.

III. Working Framework

Fig. 2 shows the overall working framework of the proposed photo mosaic application that works in the client-server model. In this framework, the client sends a photo, which is taken by the camera or retrieved from the photo gallery, to the mosaic server. The server sends the mosaic image, which is constructed from a huge number of data images, to the client.

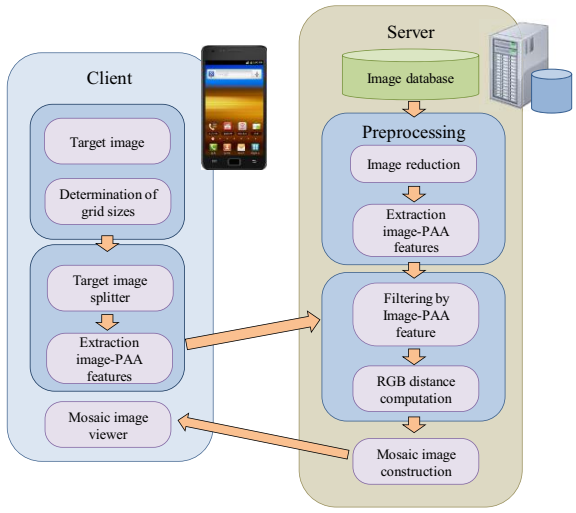

Fig. 2 Working framework of the proposed mosaic application.

A. Client Functions

As shown in Fig. 2, client functions include the input interface, the cell-based feature extraction, and the mosaic image viewer. First, the input interface gets target images with appropriate grid setting parameters. Second, the cell-based feature extraction obtains RGB histograms and image-PAA features from the divided grid-cells. More precisely, in this step, the target image splitter divides the target image into the specified number of grid-cells, and for each cell, it obtains RGB histograms and extracts the fixed number of image-PAA features. The original PAA [5] extracts a few averages from a 1-dimensional (*1-D* in short) time-series as its feature vector. As a 2-D extension of PAA, image-PAA extracts a few 2-D averages from a 2-D image as its 2-D feature vector. It is easy to prove that image-PAA features have the lower bounding property against the original 2-D RGB histograms [9]. Thus, the client sends a set of cell-based RGB histograms with their corresponding image-PAA features to the server. Third, the mosaic image viewer simply displays the resulting image, which is received from the server, to the user.

B. Server Functions

As shown in Fig. 2, server functions include the preprocessing, the image selection, and the mosaic construction. First, we execute the preprocessing only once for each image stored in the database. Through the preprocessing, we maintain each data image with its corresponding RGB histograms and image-PAA features. Second, in the image selection, for each grid-cell of the target image, we choose a data image whose RGB histograms are closest to those of the grid-cell. (We call this closest data image the *best-match* image.) Distance computation of RGB histograms, however, is a very time-consuming process. Thus, we here exploit the lower bounding property of image-PAA to prune the unnecessary data images at an early stage whose RGB histograms are far from those of the grid-cell. Distance computation of image-PAA features is very fast since their dimensionality is much smaller than that of RGB histograms. And accordingly, for all the grid-cells, we can choose their best-match images very fast even in a huge number of data images. Third, in the mosaic construction, we form a mosaic image by arranging the best images to the grid-cells.

Fig. 3 shows the pseudo code algorithm of identifying best-match images in the image selection. The inputs to the algorithm are a set of data images stored in a large image database and a set of grid-cells received from the client. The output is a list of best-match images to be used in constructing a photo mosaic. In the algorithm, for each grid-cell, we find a best-match image from the database (Lines 1 to 14). To this end, for each data image, we first compute the PAA-based lower bound to the current grid-cell (Line 4). We then use it to prune non-similar images (Line 5). That is, if the lower bound is greater than the smallest distance seen so far (= *sofardist*), we discard the corresponding image without computing the actual distance of RGB histograms. We compute the actual distance only if the image is not pruned by the PAA-based distance (Line 6). We finally obtain the best-match image whose distance to the grid-cell is smallest (Line 9) and include it into the output list (Line 13).

IV. Prototype Application

We have designed and implemented a prototype photo mosaic application on the following client-server environment. The client is Samsung GALAXY S2 [10] with Exynos Dual Core 1.2GHz, 8GB RAM, and a 16GB SD card. The client application is developed using Eclipse 4.5 (Galileo) with Java language, and it is working on Android version 2.2 (Froyo) or later. The server is a PC with Intel Core2 Quad CPU 2.5GHz, 2GB RAM, and a 300GB hard disk, and its software platform is Microsoft Window 7 operating system.

We maintain total 16,384 data images with their thumbnail images in the server. We fix the size of target images to 480 ×

680 and the size of resulting photo mosaics to 1,920 × 2,560. That is, for a given input image of size 480 ×680, we generate a mosaic image of size 1,920 × 2,560, which is 16 times larger than the input image. The reason why we set the resulting mosaic image to be larger than the target image is that we want to identify the best-match image of each grid cell. That is, when a user magnifies the mosaic image, s/he naturally wants to know which images are used for the grid-cells, and thus, we generate a large output image for a given target image.

Algorithm Best-Match
Input:
D: a set of data images in an image database;
T: a set of grid cells received from the client;
Output:
R: a list of images to construct the photo mosaic of **T**;
begin
1. **foreach** grid cell *t* of **T** **do**
2. *sofardist* := ∞;
3. **foreach** data image *d* in **D** **do**
4. *paadist* := the lower bound distance between *image-PAA features* of *t* and *d*;
5. **if** (*paadist* < *sofardist*) **then**
6. *curdist* := the actual distance between *RGB histograms* of *t* and *d*;
7. **if** (*curdist* < *sofardist*) **then**
8. *sofardist*:= *curdist*;
9. *bestone* := *d*:
10. **end-if**
11. **end-if**
12. **end-for**
13. Put *bestone* into **R**;
14. **end-for**
end

Fig. 3 Pseudo code algorithm of identifying best-match images.

Fig. 4 shows the input interface of the smartphone application implemented. As shown in the figure, a user chooses a photo from the photo gallery or takes a new photo, and s/he shakes the smartphone or clicks the "Mosaic" button to deliver the target image to the server. For the input of Fig. 4, the client shows the resulting images of Figs. 5 and 6 as the output.

Fig. 4 The input interface and its working mechanism.

Fig. 5 shows a resulting photo mosaic consisting of 4,800 grid-cells of size 8 × 8. In this example, total 63 different images are used to compose the mosaic image of Fig. 5. (Note that, because a large number of grid-cells of the target image are very similar to each other, only a small number of data images are repeatedly used in the resulting image.) By simply touching the grid-cells, we can see their magnified original photos. Fig. 6 shows another photo mosaic of 19,200 grid-cells of size 4 × 4. Total 136 data images are used in the photo mosaic of Fig. 5. The reason why the number of images is larger in Fig. 6 than in Fig. 5 is that the number of different grid-cells of Fig. 6 is larger than that of Fig. 5. We note that, comparing with Figs. 5 and 6, there is no meaningful difference in their qualities of photo mosaics. (Actually, the quality of Fig. 6 seems to be worse than that of Fig. 5.) This means that the higher number of grid-cells does not always incur the higher quality of photo mosaics. Through a series of experiments, we conclude that finding an optimal size of grid-cells for the best quality is an important and challenging research issue, and we leave it as a future study.

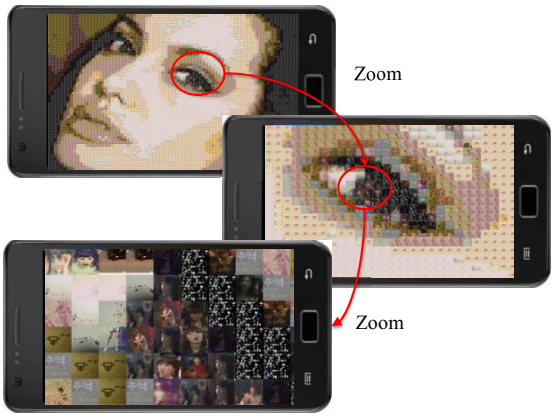

Fig. 5 A resulting photo mosaic of 4,800 grid-cells of size 8 × 8.

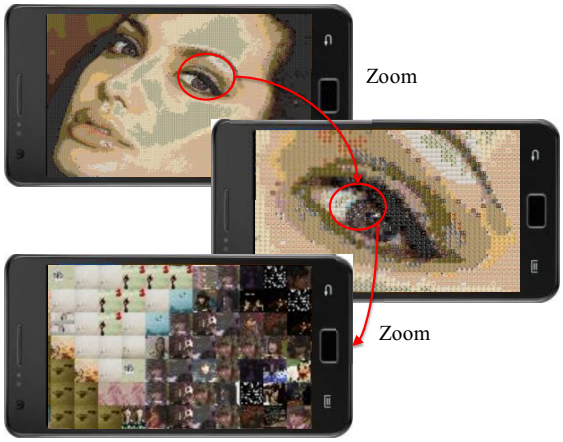

Fig. 6 A resulting photo mosaic of 19,200 grid-cells of size 4 × 4.

To evaluate the efficiency of image-PAA, we next measure the actual execution time of two cases: one for using image-PAA and another for not using image-PAA. We perform the experiment in the server which it performs image-PAA. Fig. 7 shows the execution time of two cases by varying the number of data images (i.e., images in the database). In the figure, *Image-PAA* means the case of using image-PAA; *Naïve-method* the case of not using image-PAA. As shown in the figure, *Image-PAA* outperforms *Naïve-method* by approximately two times. (Note that the y-axis is a log scale.) In particular, as the number of data images increases, the performance difference also increases. This means that the proposed image-PAA is very suitable for the large-scale image databases.

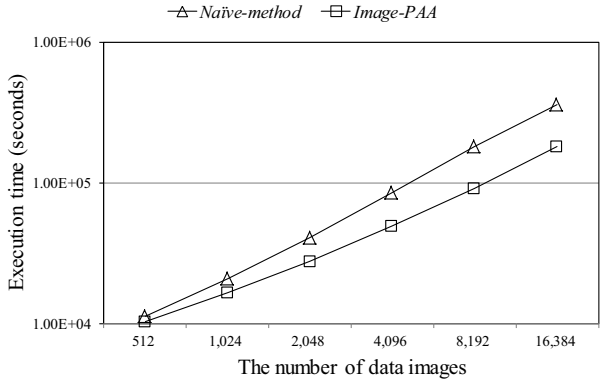

Fig. 7 Exection times by varying the number of data images.

V. Conclusions

In this paper we present an efficient smartphone application that provides photo mosaic functions in large-scale image databases. Contributions are summarized as follows. First, we present a client-server framework that makes it feasible to maintain a huge number of various images in the server and at the same time to provide the user-friendly interface to the smartphone users. Second, by presenting a novel lower bound, named image-PAA features, we provide the quick response for a large-scale image database. Third, we have implemented an Android-based prototype application and confirmed its feasibility. Fourth, we have discussed the efficiency of the proposed image-PAA through the real experiments. As the future work, we will consider the color adjustment technique to make the photo mosaic become more similar to the original one, and we will try to eliminate the duplicated images to maximize the number of data images used in the resulting photo mosaic. We will also handle an ultra-large number of movie frames as data images using the multi-dimensional index such as R*-tree [11] and Antipole-tree [12] to boost the search performance much more.

Acknowledgment

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MEST) (No. 2011-0013235).

References

[1] Wikipedia, *http://en.wikipedia.org/wiki/Smartphone.*
[2] Apple app store, *http://www.apple.com/iphone/apps-for-iphone/*
[3] Google Andorid market, *https://market.android.com/*
[4] R. Silvers and M. Hawley, Photomosaics, New York, NY, Henry Holt & Co., Inc., 1997.
[5] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases," *Knowledge and Information Systems*, Vol. 3, No. 3, pp. 263-286, Aug. 2001.
[6] A. Finkelstein and M. Range, "Image Mosaics," In *Proc. of the 7th Int'l Conf. on Electoronic Publishing*, London, UK, pp. 11-22, Mar. 1998.
[7] K. E. Hoff, T. Culver, J. Keyer, M. Lin, and D. Manocha, "Fast Computation of Generalized Voronoi Diagrams using Graphics Hardware," In *Proc. of the 26th Annual Conf. on Computer Graphics*, ACM SIGGRAPH, Los Angeles, CA, pp. 277-286, Aug. 1999.
[8] N. Tran, "Generating Photomosaics: an Empirical Study," In *Proc. of the ACM Symp. on Applied Computing*, ACM SAC, New York, NY, pp. 105-109, Feb. 1999.
[9] G. M. Morton, "A Computer Oriented Geodetic Data Base and a New Technique in File Sequencing," Technical Report, IBM, Ottawa, Canada, 1966.
[10] Samsung GALAXY S2 official Web site: *http://www.samsung.com/global/microsite/galaxys2/html/.*
[11] N. Beckmann, H.-P. Kriegel, R.Schneider, and B. Seeger, "The R*-tree: An Efficient and Robust Access Method for Points and Rectangles," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, Atlantic City, NJ, pp. 322-331, May 1990.
[12] G. D. Blasi, and P. Maria, "Fast Photomosaic," In *Proc. of the 13th Int'l Conf. in Central Europe on Computer Graphics*, *Visualization and Computer Vision 2005*, ACM/WSCG, pp.15-16, Jan. 2005.

Sang-Hun Lee received B.S. (2011) degree in Dept. of Computer Science from Kangwon National University, Korea. He is currently an M.S. student at Kangwon National University. His research interest includes data mining, knowledge discovery, and multimedia data retrieval.

Bum-Soo Kim received his B.S.(2006) degree in Computer Engineering from Halla University and M.S.(2008) degree in Computer Science from Kangwon National University. He is currently a Ph.D. candidate in the Department of Computer Science at Kangwon National University. His research interests include data mining, knowledge discovery, data warehousing & OLAP, and data mining applications.

Yang-Sae Moon received B.S.(1991), M.S.(1993), and Ph.D.(2001) degrees in Computer Science from Korea Advanced Institute of Science and Technology (KAIST). From 1993 to 1997, he was a research engineer in Hyundai Syscomm, Inc., where he participated in developing 2G and 3G mobile communication systems. From 2002 to 2005, he was a technical director in Infravalley, Inc., where he participated in planning, designing, and developing CDMA and W-CDMA mobile network services and systems. He is currently an associate professor at Kangwon National University. He was a visiting scholar at Purdue University in 2008 to 2009. His research interests include data mining, knowledge discovery, storage systems, access methods, multimedia information retrieval, mobile/wireless communication systems, and network communication systems. He is a member of the IEEE, a member of the ACM, and a member of the IEICE.

Jinho Kim received B.S.(1982) in Electrical Engineering from Kyungpook National University and M.S.(1985) and Ph.D.(1990) degrees in Computer Science from Korea Advanced Institute of Science and Technology (KAIST) respectively. From 1990, he joined at the Department of Computer Science, Kangwon National University. Now he is a professor at Kangwon National University. He was a visiting scholar at the University of Michgian in 1995 to 1996 and at the Drexel University in 2003 to 2004 respectively. His research interests include data warehousing and OLAP, data mining, main-memory database systems, XML databases, and information retrieval. He is a member of the IEEE and a member of the ACM.