Process for creating photo mosaics

Mati Kärner
University of Tartu, Institute of Computer Science

Objectives

- Design a process for creating photo mosaics.
- Try out different data structures to speed up the process.
- Propose ways to make mosaics visually more pleasing. Implement some of them.
- Build a working prototype.

Introduction

A photo mosaic is setting of small images, each chosen to fit into particular place of an original image. One of the challenges of designing photo mosaics is striking a balance between discernibility, i.e., validating the legitimacy of each tile, and reproducing the source image as closely as possible [1]. In addition to controlling the tile size, one could increase the number of images in database to improve discernibility, but this may slow down the tile matching procedure.

We use 3000 64x64 sized music albums covers found from the internet and a prototype built for this project to produce photo mosaics. Apart from design and implementation, we present some ideas for improving the overall mosaic creation process.

State of the Art

There are a many papers concerning photo mosaics. Some of them concentrate on tile matching process while other seek ways to improve visual appearance. To provide background, some of them are presented.

Kim and Pellacini propose Jig Saw Mosaic [2], i.e., tiny arbitrarily sized images are used to compose a target image. This method poorly reproduces original photos and mainly works for images with few clearly distinguishable objects. Choi *et. al* use masks on arbitrarily sized tiles [3]; write mask to determine the tile size, energy and edge masks for preserving the edges and margins of the source image. Some autors, as in [4], use different tile blending techniques to achieve artistic effects (e.g. impressionistic paintings).

Process

Gather a large amount of pictures, crop them, calculate *average* color over all pixels P, $|P| = N$:

$$(1/N \sum_{p \in P} p.r, 1/N \sum_{p \in P} p.g, 1/N \sum_{p \in P} p.b).$$

1. Crop image
2. Draw grid over the source image
   Trivial for monosized tiles
   Otherwise segment the image and fit the tiles
3. Build a kd-tree out of canditate tiles
   $O(N \log N)$
4. For each cell in grid find its NN from the tree
   $O(N \log N)$; cf. linear search $O(N^2)$
5. Paint

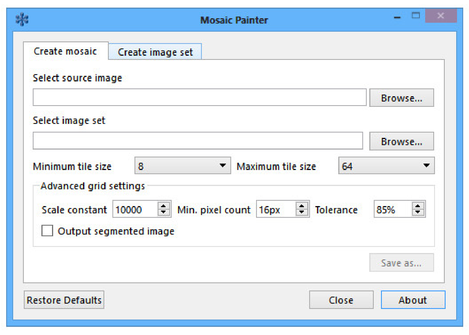

Figure 1: Screenshot of the prototype.

Image Segmentation

Segment image using graph based algorithm [5] proposed by Felzenszwalba and Huttenlocher. Implemented using disjoint-set data structure with union by rank and path compression. Running time is $O(N \log N)$ w.r.t. number of pixels N.

Algorithm assigns weights (Euclidean dist.) to edges (4-connected pixels) and partitions pixels into components using predicate D, which tests whether there is a boundary between components or not.

Draw a grid over segmented image, keep breaking the cell into four until minimum tile size reached or cell is small enough to fit into component. Pick random pixels from cell and identify their components. Fix the tile if some component accounts up to t percent of pixels.

Ideas to try out

1. Use color histograms
2. Try Locality Sensitive Hashing for NN approx
3. Repeat as few tiles as possible
4. Try out puzzle-shaped tiles
5. Use $O(n)$ sort to speed up segmentation process
   Requires integer weights

Results

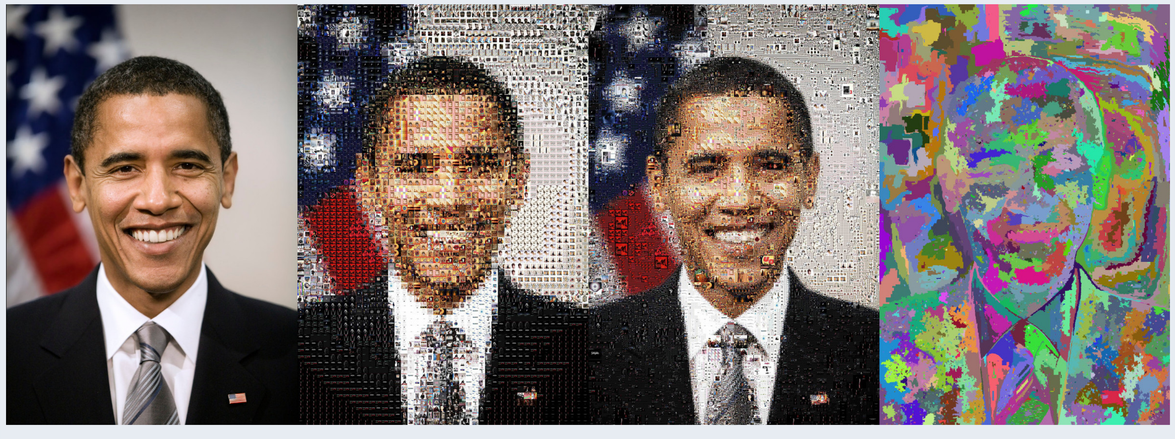

Figure 2: Source image [6] and preview of mosaics (1443x2100px) built with 32x32px tiles and tiles of mixed size.

Conclusion and Future Work

In comparison to monosized tiles, the segmentation process significantly improves the sharpness of the mosaic and helps to reproduce the original image more closely. The randomized tile fitting process helps to reduce the number of tiles making the tile matching process faster, but in the same time requires more sophisticated techniques for improving the discernibility of larger tiles. As for future work we believe the ideas presented here are worth trying and could change the process faster or to better produce mosaics closer to the source image.

References

[1] Jeff Orchard and Craig S. Kaplan. Cut-out image mosaics. pages 79–87, 2008.

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

[3] Yoon-Seok Choi, Bon-Ki Koo, and Ji-Hyung Lee. Template based image mosaics. In *Proceedings of the 6th International Conference on Entertainment Computing*, ICEC'07, pages 475–478, Berlin, Heidelberg, 2007. Springer-Verlag.

[4] Linlin Jing, Kohei Inoue, and Kiichi Urahama. An npr technique for pointillistic and mosaic images with impressionist color arrangement. In *Proceedings of the First International Conference on Advances in Visual Computing*, ISVC'05, pages 1–8, Berlin, Heidelberg, 2005. Springer-Verlag.

[5] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59(2):167–181, September 2004.

[6] Portrait of barack obama. `http://upload.wikimedia.org/wikipedia/commons/0/01/Poster-sized_portrait_of_Barack_Obama_OrigRes.jpg`.

Source & Prototype

- Source: bitbucket.org/unematiii/mosaic
- Release: ut.ee/~b04866/Mosaic Painter/

Contacts

- Email: b04866@ut.ee