

NinjaEdit: Simultaneous and Consistent Editing of an Unorganized Set of Photographs

Koichiro Honda*
The University of Tokyo

Takeo Igarashi†
The University of Tokyo



Figure 1: Example output of this system. Once an edit operation is done to one image, it is immediately propagated to other images.

Abstract

We propose a novel technique to edit multiple photographs of a scene easily and simultaneously, maintaining their visual coherency. When an editing operation is applied to a photograph, the operation is immediately propagated to the other photographs. The system first reconstructs 3D geometric scene for the set of photographs to find corresponding pixels among the images. The system then applies an equivalent editing operation on each other photograph in the set, keeping visual consistency. We demonstrate with several example sets of photographs.

CR Categories: I.4.0 [General]: Image Processing, Batch Processing—;

Keywords: Image Editing, User Interface, Batch Processing, Computer Vision

1 Introduction

Image processing is one of the most fundamental tools for digital photography. Photographers commonly apply various visual effects on photographs to enhance visual quality. Over decades, a large amount of efforts has been put into developing various image processing techniques. Thanks to these contributions, we can improve photographs by adding effects on them intuitively and without extensive technical background.

In this paper, we propose a novel editing technique for multiple photographs of a scene. With our method, you can apply effects on photographs easily and simultaneously, without losing visual coherency (See Figure 2). Generally speaking, when we apply visual effects to a set of images that capture the same scene (e.g. draw characters onto a wall in all images that capture it), we must consider spatial relationship among images. However, it is very hard and complicated to do it manually. One main reason for this is the difficulty to keep visual coherence among images. You might be able to cope with two images, but it is almost impossible to handle dozens of images.

Most of image editing tools have batch processing features to manage multiple images. However, they are limited to simpler effects, such as resizing, cropping, and tone mapping to the whole images, etc. That is, they are not aware of the contents of images. What we aim at here is more intelligent batch processing, which is adaptive

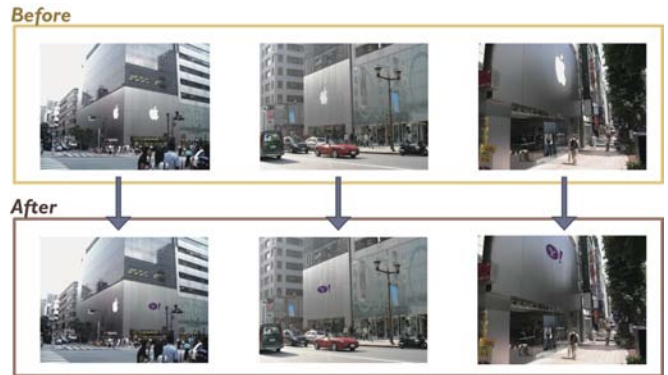


Figure 2: The goal of our work. User applies an editing operation to a photo (left) and it propagates to the other photos (middle, right) the consistent operation at the same time.

to the objects within images. Our system does not apply the same effects, but equivalent effects to each image (See Figure 1).

There exists other works utilizing multiple images for retrieving information. Sinha et al. adopted an interactive approach to construct architectural 3D model from an unordered set of multiple images by allowing users to designate plain surfaces [Sinha et al. 2008]. Snavely et al. created a 3D viewer for a collection of photographs from online, in which we can take a virtual tour and walk around famous sites in 3D visual interface [Snavely et al. 2006].

Our system also considers 3D information as a guide for transporting editing operations among images. In order to keep visual consistency between images, we reconstruct a 3D scene in advance to know spatial relationship between the images. In this process, we use a standard reconstruction technique based on sparse feature points. After reconstruction, effects are interactively transferred to other images while editing.

2 Algorithm

2.1 Scene Reconstruction

We use Snavely’s method [Snavely et al. 2006] to reconstruct a 3D scene from a set of photographs. First, we find SIFT keypoints in each image and match them between each pair of photographs. Then, we estimate fundamental matrices for each pair followed by a non-linear optimization process which minimizes reprojection er-

*e-mail:koichiro@ui.is.s.u-tokyo.ac.jp

†e-mail:takeo@acm.org

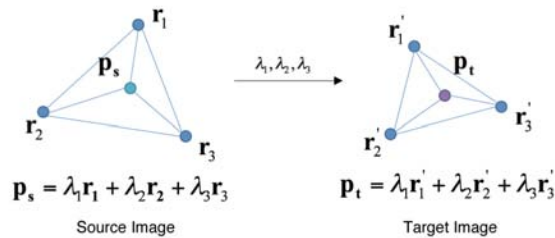


Figure 3: Algorithm overview. Pixels are parameterized by three sparse points via barycentric coordinates, then the parameters are propagated to other images to estimate the corresponding pixels.

rors among images. See [Snavely et al. 2006] for more detail.

2.2 Consistent Edit Propagation

To edit all images consistently, we need to propagate an edit operation from one image to others. We achieve this by propagating the locations of editing, not by propagating the edit itself. The next paragraph explains how we estimate corresponding regions. This calculation is executed immediately after the initial edit operation.

To estimate the corresponding region on another image, we use barycentric interpolation. Suppose when you select a point P_S on the source image S , and the system will find the corresponding point P_T on the target image T . Let the coordinate of P_S be \mathbf{x}_S , and that of P_T be \mathbf{x}_T . First, we find three nearest keypoints to the point of editing, which both S and T have in common. Then we parameterize \mathbf{x}_S with the nearest points as follows.

$$\mathbf{x}_S = \lambda_1 \mathbf{r}_1 + \lambda_2 \mathbf{r}_2 + \lambda_3 \mathbf{r}_3 \quad (1)$$

$$\lambda_1 + \lambda_2 + \lambda_3 = 1 \quad (2)$$

where $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$ are the vectors of three nearest feature points, and $\lambda_1, \lambda_2, \lambda_3$ are the corresponding parameters. Once we get the parameters, we estimate \mathbf{x}_T as follows

$$\mathbf{x}_T = \lambda_1 \mathbf{r}'_1 + \lambda_2 \mathbf{r}'_2 + \lambda_3 \mathbf{r}'_3 \quad (3)$$

In the same way, we estimate the corresponding points on the remaining images. Figure 3 shows a brief overview of this algorithm. Feature points selected for estimation depend on the source and target image, even when transporting the same point. This is because there are few feature points that all images have in common; so it is not practical to adopt these points as a guide.

3 Results

We demonstrated our algorithm with two sets of photographs. One captures a poster on a door, and the other captures a paper bag on a desk. We implemented an image composition effect for an experiment. Image composition effect is propagated like the following: firstly, you specify four corners of a rectangle in one image, which denotes a bounding box where you want to attach an image. Then, corresponding bounding boxes in other images are calculated immediately and the same image is attached onto other images. Figure 4 shows the original images and final outputs. On Figure 4 (top), we scribbled on a paper bag by filling o in black. Fill operation was added to the upper-left close-up image, then the system propagates it to other images. On Figure 4 (bottom), we replaced the poster displayed on the door with another. We did on the bottom right image, and then the operation was transferred to other images. It keeps visual consistency, and completes in realtime.



Figure 4: Results. Left: original, Right: outputs. (Top) Filling o in black on a paper bag on a desk. (Bottom) Replacing a poster attached on a door.

4 Conclusion

We proposed a new approach designed for editing multiple images simultaneously and consistently by acquiring spatial relationship of images using 3D scene reconstruction. While editing images, we interactively detect the corresponding editing locations on other images by calculating barycentric coordinate-based parameters. Finally, we demonstrated it with several sets of photographs.

Acknowledgements

We would like to thank the Information-Technology Promotion Agency Japan for funding this research.

References

- SINHA, S. N., STEEDLY, D., SZELISKI, R., AND AGRAWALA, M. 2008. Interactive 3d architectural modeling from unordered photo collections. *ACM Transactions on Graphics* 27, 5, 159.
- SNARELY, N., SEITZ, S. M., AND SZELISKI, R. 2006. Photo tourism: exploring photo collections in 3d. *ACM Transactions on Graphics* 25, 3, 835–846.