Bars Removal by Multiview Images

YiChang Shih, Hsin-Jung Yang

May 12, 2011

1 Introduction

Imagine that you are walking around a zoo. You want to take a good picture of a lion, but the bars in front of the lion are really annoying. What can you do? You probably don't want to reach into the cage and take the risk of losing your hand. In this project we propose an image post-processing technique by multiview images to remove bars. A user take multiple pictures from different viewpoints, and then get a synthesized picture without bars as output.

Our algorithm is made of 3 main stages: in the first stage, we segment the bars, and in the second stage, we align the animal images. In the third stage, we perform patch matching, and find the patch to complete the pixels hidden by the bars. We also address the issue that the animal might move or change pose between two viewpoints.

The results show that our algorithm can remove most bars, and generate visually good composite images.

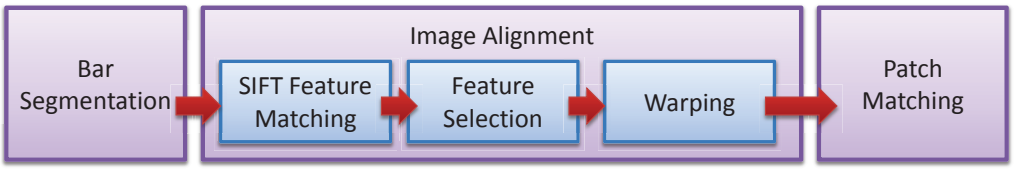

Figure 1: Algorithm Flowchart

2 Related Work

Texture Synthesis Our work is to fill the pixels on the bars, so it is similar to texture synthesis and image completion. This field has been studied for a long time[1, 2]. In particular, our method is similar to Efros's work [1]. However, instead of a fixed non-parametric model in their work, our model for image completion depends on the position of unknown pixels. Besides, in our work, we need to carefully handle the bars in the images.

Feature-Based Alignment In the first stage of our algorithm, we apply the feature-based image alignment to match the animal images. Our work is more sophisticated than the traditional alignment, because we need to remove the features of bars.

3 Algorithm Overview

In order to remove bars from the reference image, the first step is to segment the bars. Then, we want to choose the best candidate from other viewpoints to patch up the bars. As shown in Figure 1, after segmentation, we perform the feature-based alignment to label corresponding pixels in other images. Because the center of projection is not fixed when taking pictures, it is impossible to perfectly align those images. Therefore, we perform patch matching to search for the best candidate instead of directly pasting the corresponding pixels.

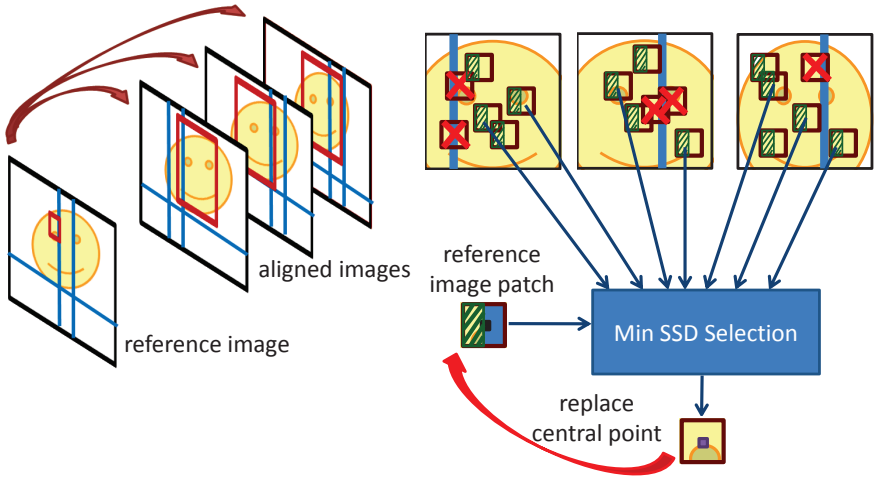

Figure 2: Patch Matching Algorithm

3.1 Bar Segmentation

Observing that bars are usually monochromatic, we manually select some bar pixels and train a Gaussian model. Then we use it to segment bars.

3.2 Feature-Based Alignment

To align an image to the reference viewpoint, first, we use the scale-invariant feature transform (SIFT) [3] to extract features and match them with the features on the reference image. Then, we perform feature selection to remove bar features. At the final step, we warp the image to find the corresponding position of each pixel.

3.3 Patch Matching

Figure 2 shows how the patch matching algorithm works. First, we crop a template (typically 5x5) from the reference image. Then, we crop larger patches from other viewpoint images at the corresponding positions. Then we do template matching on those patches, and find the best candidate.

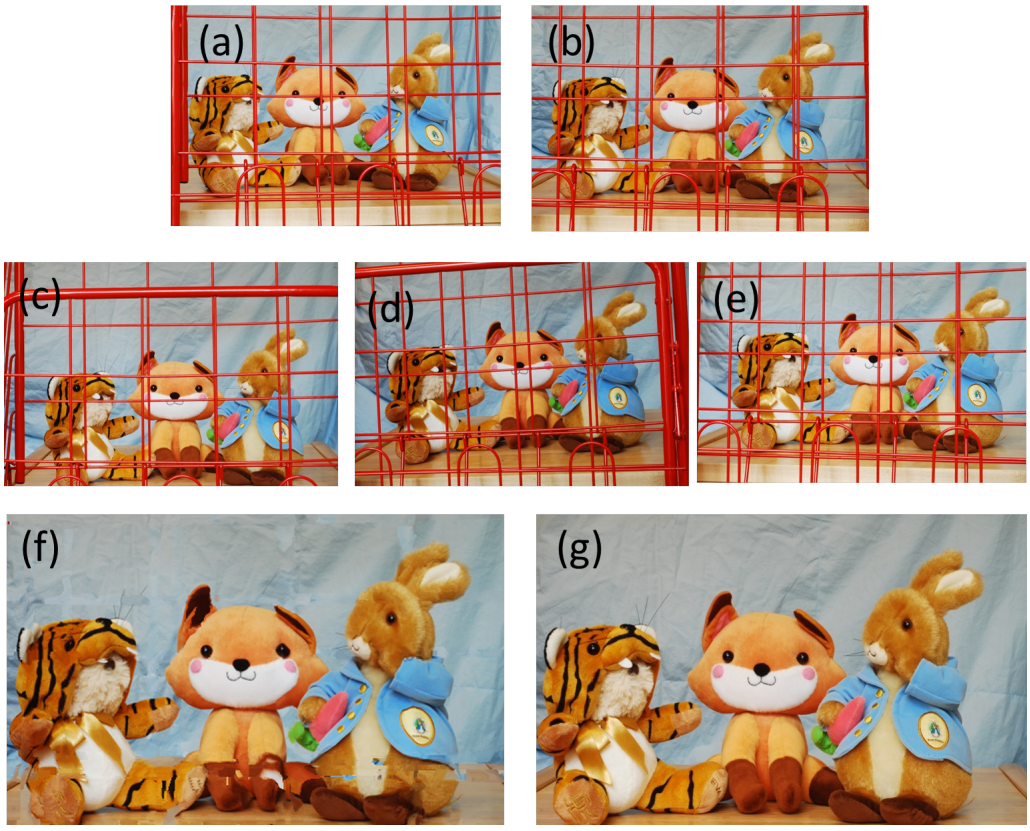

Figure 3: Bar removal from multiviews. (a)-(e) Images from 5 viewpoints. (f) Our result (g) Ground truth

4 Result

Figure 3 shows our result. All the images are taken by Nikon D80, 50mm lens. If we take a closer look at Fig. 3, there are some differences between our result and the ground truth, for instance, the right eye-ball and the right ear of the fox. But the result in general looks visually good.

Figure 4 shows the non-static scene (the animals move between two viewpoints). Note that the occluded parts (the right side of the fox) cannot be recovered.

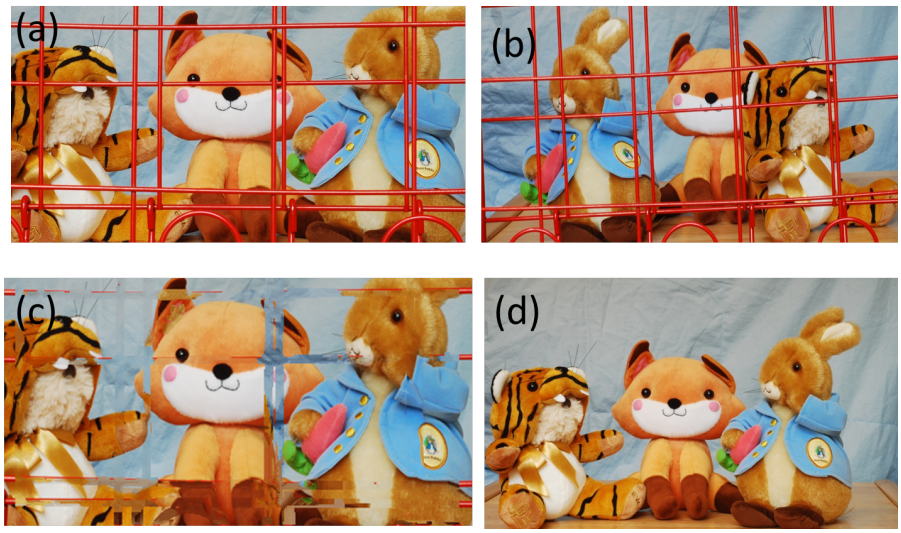

Figure 4: Non-static scene. (a)-(b) Images from 2 viewpoints. The rabbit and the tiger switch the positions between (a) and (b). (c) Our result. (d) Ground truth.

References

[1] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling," in *iccv*. Published by the IEEE Computer Society, 1999, p. 1033.

[2] A. Efros, W. Freeman *et al.*, "Image quilting for texture synthesis and transfer," in *Proceedings of SIGGRAPH 2001*. Citeseer, 2001, pp. 341–346.

[3] D. Lowe, "Object recognition from local scale-invariant features," in *iccv*. Published by the IEEE Computer Society, 1999, p. 1150.