Reflection Removal Algorithms

Matthew Hu
Stanford University
matthu@stanford.edu

Abstract

Reflection in images is a big problem in photography. This report presents and evaluates three separate algorithms described in literature that attempt to computationally remove reflections. Multiple images are tested on each algorithm and various test cases are shown. The strengths and weaknesses of each algorithm are outlined and evaluated.

1. Introduction

Photographers are sometimes faced with the difficulty of taking an image of a scene with a reflective surface. This can cause many issues with reflections obstructing the desired scene and unwanted interference. There are many different ways photographers work around this by either adjusting the lighting, their positioning, or adding polarizers to their camera. However, the vast majority of people do not have access to polarizers and it is often inconvenient to adjust one's positioning or lighting for a better picture. This report explores some of the algorithms that attempt to computationally remove reflections in images.

Figure 1. Examples of reflections in photography

2. Overview

There are two paradigms on how to solve this issue of image reflections. There are several papers which attempt to remove, or at least mitigate reflections from a single image input. These algorithms attempt to separate the reflection and transmission layers with different objective functions that favor images with sparse gradients. [3] [4] However, this is a fairly difficult problem to solve with just a single input image. At a very high level, we can model the image using the equation $I = T + R$ where I is the resulting image and T and R are the transmission and reflection layers respectively [2]. The resulting image is simply a linear combination of the scene through the window and the reflected image. Trying to obtain two variables from just a single image is somewhat of an ill-posed problem, so different image priors are added to the objective function. [3]

Recently, a more popular approach to this problem involves taking a series of shifted images to separate the reflection and transmission. Often, the two layers are located at different depths, so shifting the camera will cause the two layers to move at different rates. This difference in motion can then be used as a robust way of separating the reflection and transmission layers [6][5].

3. Algorithm Descriptions

In this section, I will be exploring three separate approaches to reflection removal: Sparse blind separation with motions, superimposed image decomposition, and ghosting cues detection. Each algorithm models reflection differently and utilizes different objective functions to separate the reflection and transmission layers. The goal of this paper is to evaluate each algorithm separately to determine the strengths and weaknesses of each approach. A mix of real images taken from the internet and from my own camera will be used to test each method.

3.1. Sparse Blind Separation with Motions (SPBS-M)

The first algorithm I looked at was SPBS-M, which follows the paradigm of taking a series of shifted images to help separate the reflection and transmission layers. The mixing model used by the authors is as follows: for a sequence of images, there are m images that each contain n layers shown by the equation below [1].

$$I_i(x) = \sum_{j=1}^{n} a_{ij} L_j(f_{ij}(x)), \quad i = 1, ..., m, \qquad (1)$$

In this equation, x is the vector representing the pixel location, f_{ij} is the motion transformation of each image, L_j is the jth layer, and a_{ij} is the mixing coefficient for each layer. Essentially, each image is simply composed of a weighted sum of multiple layers that are shifted from image to image. This paper attempts to estimate each of these coefficients in order to separate each layer $L_1 ... L_n$. This is done by taking advantage of general properties and statistics of natural images. The authors examined over 130,000 images and created hypotheses on the sparsity of image gradients, the noncorrelation of the gradients of different locations in the same image, the joint behaviors of different gradients in the same image, and the independence of the gradients and pixel values of different images. These hypotheses applied to an objective function which was used to find the motion and mixing parameters of a single layer. A similar approach is done to find the parameters and coefficients of the second layer. Finally, each layer is reconstructed using the coefficients found in the previous steps. The final objective function attempts to reconstruct layers that fit the mixing model detailed above and the extracted gradients of each layer [1].

3.2. Superimposed Image Decomposition (SID)

The superimposed image decomposition (SID) method proposed by Guo, Cao, and Ma also takes in a series of shifted photos as input but has a very different approach in solving the problem [2]. First, one small advantage of this algorithm is that it is more flexible in terms of image translation and transformation. As shown in the images in figure 2, the algorithm will first preprocess the images so that the shifted and rotated images are all centered on a flat plane. However, this transformation needs to be encoded into the program for each set of images. This is not done automatically, so some preprocessing of each dataset needs to occur before reflection removal can occur.

The model this algorithm uses is shown in the equation belows:

$$\mathbf{F} \circ \Gamma = \mathbf{T} + \mathbf{R} + \mathbf{N} \qquad (2)$$

Here, $\mathbf{F} \circ \Gamma$ is the set of input images mapped to a matrix with the homographic transformations applied. $\mathbf{T}$ is the transmission matrix which contains all the transmission images of the entire set, and $\mathbf{R}$ and $\mathbf{N}$ correspond to the reflectance layer and noise respectively [2]. To solve this problem, a single objective function is formed and three structural priors are used: the correlation of the transmitted layer in a single set, the sparsity of the gradients, and the independence between the reflected and transmitted layers [2].

(a) Input Image (b) Transformed Input Image

(c) Transmission Layer (d) Reflection Layer

Figure 2. Example of image transformation in SID

3.3. Ghosting Cues

Unlike the other two algorithms, Shih's method of searching for ghosting cues only takes in a single image as input [4]. This method searches the image for ghosting artifacts, which are shifted, secondary reflections in an image. The image model they use is shown in the equation below.

$$I = T + R \otimes k + n \qquad (3)$$

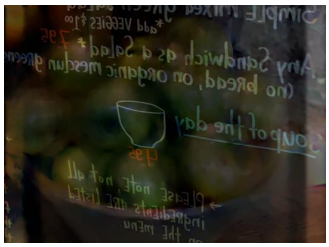
Here I is the input image, T and R are the transmission and reflection layers respectively, n is the noise term, and $R \otimes k$ is the convolution of the reflection layer with the ghosting kernel k. The general cost function to be minimized is shown below.

$$min_{T,R} \ \frac{1}{\sigma^2}||I - T - R \otimes k||_2^2 - \sum_i log(GMM(P_iT)) - \sum_i log(GMM(P_iR)), \ \ s.t. \ 0 \leq T, R \leq 1 \tag{4}$$

The first term is simply to minimize the residuals, but the last two terms come from the Gaussian Mixture Models (GMM) prior added to improve image output. The variables P_iT and P_iR are simply the i^{th} patch in T and R respectively [4]. Extra terms are later added for optimization.

The first example I ran was the image provided by the paper itself, and the results are shown in Figure 3. We can see that we get a very clean separation of the transmission and reflection layers. However, this is a synthetic image, and when we tried the same algorithm on other images, this method had difficulty producing similar results. Also, this image was downsampled to 432x320 pixels and it still took over 2 hours to run.

(a) Input Image (b) Transmission Layer



(c) Reflection Layer

Figure 3. Example image shown in the paper

4. Performance Evaluation

The images I used to evaluate the algorithms included a mix of images provided by the papers themselves, other images found online, and pictures I took on my own. For this report, I simply show 4 examples that illustrate some of the advantages and disadvantages of each algorithm. Each example contains a set of shifted images that allow us to draw comparisons between the different methods.

Example 1 shown in Figure 4 is a dataset that contains 20 shifted images. We can see from the results that both the SPBS-M and SID algorithms performed fairly well in this scenario. There is a fairly clean separation between the transmission and reflection layers that allow us to get a much cleaner view of the drawing. SID seems to perform slightly better in this case as there are still small reflections in the transmission of the SPBS-M layer, but the differences overall seem fairly minor. The algorithm with ghosting cues did not perform as well. It was able to attenuate some of the background reflections, but there are still strong reflections in the transmission layer. With large, well-aligned datasets, SPBS-M and SID perform much better at removing reflections.

Example 2 shown in Figure 5 is a smaller dataset of 4 images. However, this time the reflections on each image are very different from image to image. The photographer and man standing on the staircase are in different positions each time. As a result, the reflectance layer from SPBS-M is time-aliased. We see a blurry image of both men as they move across the 4 images. You also see some of the time-aliasing from SPBS-M in figure 4. SPBS-M produces only a single pair of transmission and reflection images from the set of images taken. As a result, there is some aliasing and blurring if there is any significant movement or change in the scene. We can also see some of the effects of this blurring in the transmission layer as well. However, the SID algorithm generates a separate reflection and transmission layer for each image. As a result, we don't get the same time-aliasing effects and we get a cleaner separation of the transmission and reflection layer.

Example 3 shown in Figure 6 is a dataset that only consists of 2 shifted images. This input data comes from taking a screenshot of the SPBS-M paper, and as expected, the SPBS-M algorithm performs extremely well. This example shows the potential of the SPBS-M algorithm when operating within its proper bounds of relatively static image with only translational shifts in the image. We get an almost perfect separation between the transmission and reflection layers with very few artifacts in either layer. The SID algorithm struggles here a bit because there are only 2 shifted images. It is able to mitigate some of the reflection in the transmission layer, but there are still a lot of artifacts in both layers. Both SID and ghosting cues perform fairly poorly in this example. However, SID still forms a little better as we can see a clear outline of the tree and photographer in the SID reflection layer.

Example 4 in Figure 7 is a stress test for the different algorithms to evaluate their performance under suboptimal conditions. There were eight shifted images in this dataset and the scene taken has both texture and depth. All of the algorithms described in this report used sparse gradients as one of the assumptions. Figure 7 is an image taken with

Image Size	SPBS-M	SID	Ghosting Cues
Ex1: 220 x 200	15 min	30s	30 min
Ex2: 200 x 180	1 hr	20s	30 min
Ex3: 450 x 400	1.5 hrs	60s	4 hrs

Table 1. Speed Analysis of the three algorithms

a large tree in the background which causes a lot of issues in the resulting layers. All three algorithms failed to separate the reflection and transmission layers of this image. You can see the reflection of the computer and monitor in both layers for all 3 algorithms. The SPBS-M algorithm managed to preserve most of the detail. You can still see some of the texture on the tree in the transmission layer. The SID algorithm, on the other hand had very little detail. Most of the image is blurred out and it's difficult to see any sort of texture in the transmission layer. However, the SID did perform the best at separating reflection from transmission. The laptop, monitor, and light are all very clear in the reflection layer, and these reflections are very much attenuated in the transmission layer. Ghosting cues also fails to separate the two layers. Also, the tree appears very patchy, which is likely due to the GMM priors added to the objective function. Another difficulty with this image is depth. The other examples had two very clear depths, which makes the layer detection due to motion differences easier to determine. A range of depths interferes with that property, making it harder to separate the layers.

I also ran a speed test of all three algorithms and the results are shown in the table. SID is the fastest algorithm by a fairly large margin. It only took on the order of 1 min to process small image for SID. SPBS-M and ghosting cues take significantly longer to process. These algorithms take over an hour to run on tiny images. I tried running some of these algorithms on larger images like 600x750 pixels, but my system crashed and was unable to complete the process after 8 hours of processing.

5. Conclusion

In this report, I evaluated 3 separate reflection removal algorithms and compared the strengths and weaknesses of each one. SID is by far the fastest algorithm, but it performs poorly when there are only a few images in the dataset and when the scene contains a high amount of texture. SID also allows for shifts and transformations in the scene, but all of this data needs to be encoded by the user into the algorithm. This does not happen automatically. SPBS-M is slower and performs well under a relatively static scene. However, when there is a lot of movement in the reflection layer, this can cause some time-aliasing artifacts to appear in both layers. The algorithm using ghosting cues was not able to cleanly separate reflections on any of the images tested. It also causes patchy artifacts to appear when we use a scene with high textures. In general, algorithms which take in a series of images seem to perform much better than algorithms that simply operate on a single image. By introducing movement to the scene, we are better equipped to separate the two layers by calculating how each layer moves in time. However, this also means that if we have a scene with a range of depths, then we will have a harder time of separating layers. These algorithms rely on motion parallax to separate layers, so its more difficult when a scene has many different depths.

There are many more algorithms that can be evaluated for reflection removal. Right now, the most recent and popular algorithm seems to be Xue's "A computational approach for obstruction-free photography" [6]. In this algorithm, panorama sweep with a smartphone is sufficient to remove reflections on high-quality, complex images. Many of the results shown seem very impressive, so that would be the next algorithm to evaluate in the future.

References

[1] K. Gai, Z. Shi, and C. Zhang. Blind separation of superimposed moving images using image statistics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):19–32, Jan 2012.

[2] X. Guo, X. Cao, and Y. Ma. Robust separation of reflection from multiple images. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 2195–2202, June 2014.

[3] A. Levin, A. Zomet, and Y. Weiss. Separating reflections from a single image using local features. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–306–I–313 Vol.1, June 2004.

[4] Y. Shih, D. Krishnan, F. Durand, and W. Freeman. Reflection removal using ghosting cues. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 3193–3201, June 2015.

[5] S. N. Sinha, J. Kopf, M. Goesele, D. Scharstein, and R. Szeliski. Image-based rendering for scenes with reflections. In *ACM Trans. Graph. (August 2012)*. ACM SIGGRAPH, August 2012.

[6] T. Xue, M. Rubinstein, C. Liu, and W. T. Freeman. A computational approach for obstruction-free photography. *ACM Trans. Graph.*, 34(4):79:1–79:11, July 2015.

(a) Input Image (b) Input Image (c) SPBS-M Transmission Layer (d) SPBS-M Reflection Layer

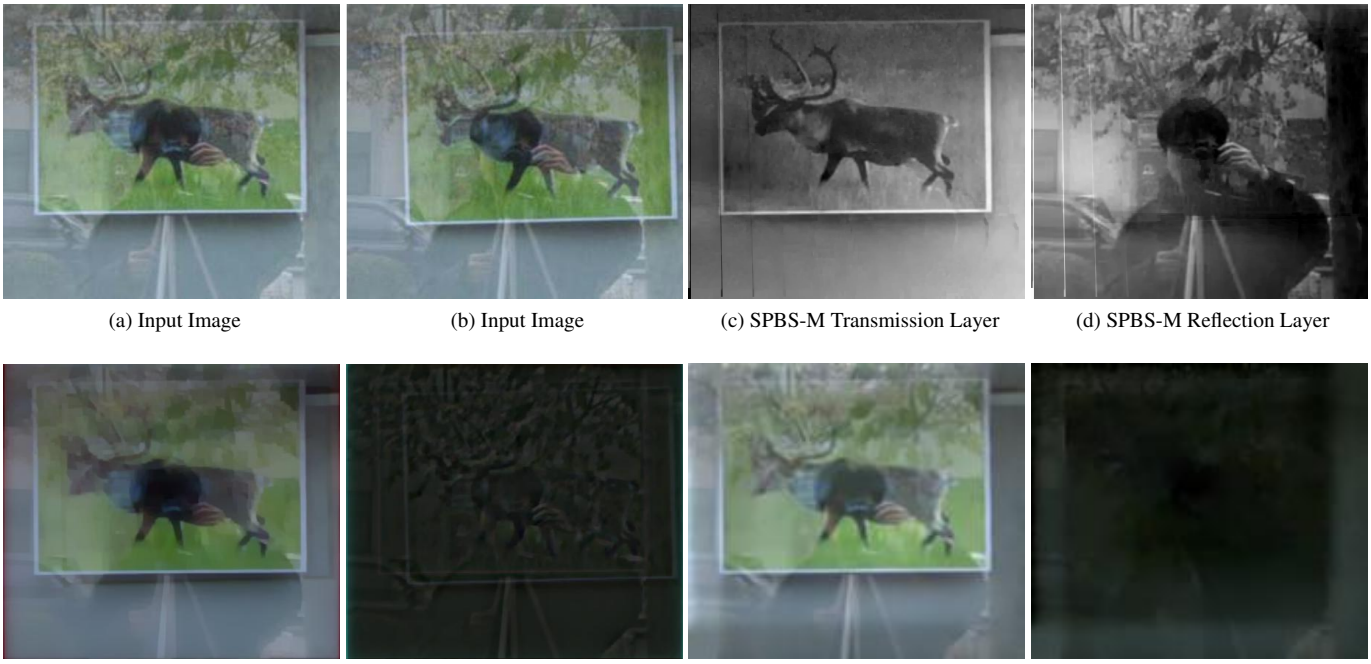
(e) SID Transmission Layer (f) SID Reflection Layer (g) Ghosting Transmission Layer (h) Ghosting Reflection Layer

Figure 4. Example 1 dataset with 20 images

(a) Input Image (b) Input Image (c) SPBS-M Transmission Layer (d) SPBS-M Reflection Layer

(e) SID Transmission Layer (f) SID Reflection Layer (g) Ghosting Transmission Layer (h) Ghosting Reflection Layer

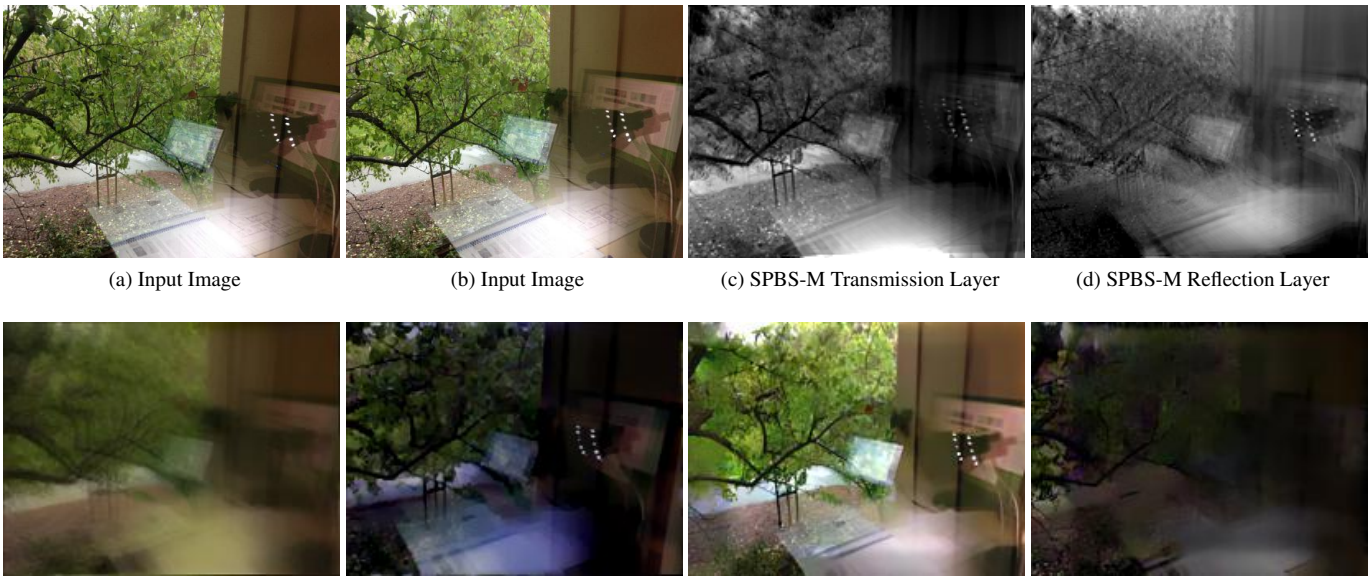
Figure 5. Example 2 dataset with 4 images



(a) Input Image (b) Input Image (c) SPBS-M Transmission Layer (d) SPBS-M Reflection Layer

(e) SID Transmission Layer (f) SID Reflection Layer (g) Ghosting Transmission Layer (h) Ghosting Reflection Layer

Figure 6. Example 3 dataset with 2 images



(a) Input Image (b) Input Image (c) SPBS-M Transmission Layer (d) SPBS-M Reflection Layer

(e) SID Transmission Layer (f) SID Reflection Layer (g) Ghosting Transmission Layer (h) Ghosting Reflection Layer

Figure 7. Example 4 dataset with 8 images