Low-cost Computational Astrophotography

Joseph Yen
Electrical Engineering
Stanford University
joyen98@stanford.edu

Peter Bryan
Electrical Engineering
Stanford University
jpbryan6@stanford.edu

Abstract

When taking photographs of the night sky, long exposure times help photographers observe many stars, but the rotation of the earth results in long star streak arcs about the celestial pole. An algorithm for removing these star streaks and substituting accurately placed point-like stars is designed and implemented. Images containing star streaks are transformed into ones with un-streaked starry skies while retaining color and brightness information. The procedure is shown to work consistently on a 35-image dataset.

1. Introduction

Astrophotography is a widespread pastime in which DSLR cameras are used to take images of objects in space. Certain factors make this a very expensive hobby. One of these is the rotation of the earth. Very long exposure times are required to capture photos of faint celestial bodies, and the earths rotation causes these objects to move in the sky during the exposure. This causes the formation of ”star streaks,” bright lines in the photograph tracing out stars' trajectory across the sky. These streaks can be aesthetically pleasing and many photographers deliberately create streaked images. However, in producing an accurate image of the sky, they are detrimental. To compensate for this motion, astrophotographers use moving camera and telescope mounts to keep objects steady in the frame. These mounts can cost thousands of dollars, making astrophotography prohibitively expensive for most people. We hope to use computational techniques to remove the need for these rotating mounts and instead post-process these long-exposure images.

2. Related Work

Currently, most astrophotographers compensate for the rotation of the earth by physically moving the camera along with the earth. However, some attempts have been made to remove this requirement. To accurately localize stars in streaked images, one researcher has re-mapped the star streak to polar coordinates, so that all stars will have the same point-spread function. This technique was designed for locating specific stars for the purpose of identifying specific stars, but will be applicable for our purposes when removing star streaks [4]. Other factors besides streaking can cause quality degradation in star images, and attempts have been made to correct this. These have used such techniques as Richardson-Lucy deblurring [2] and maximally sparse optimization [1].

3. Data Collection

Initially, we intended to produce our own images of starry skies by taking photos of the night sky at Stanford. However, due to unfortunately cloudy and rainy weather, we were unable to take acceptable photos that could be post-processed. Instead, we collected a data set of 35 photos of star streaks gathered from the internet, via a Google Image search. To ensure that our procedure would be able to handle the images, we selected images in which the celestial pole was within the photograph. The images also had varied parameters such as foreground/background elements, amount of star streaks, illumination, and noise. Any watermarks and borders left in the photographs were cropped out to avoid the possibility that they could be segmented as stars.

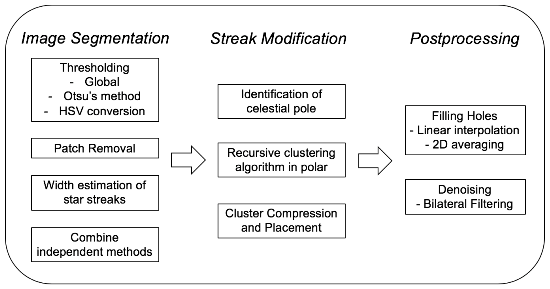

Figure 1. Image processing pipeline

4. Image Segmentation

In order to effectively convert the star streaks into points, image segmentation is needed to separate out the streaks, as detailed in Figure 1. The star streaks could be of varying brightness and color, even along a single streak. In addition, even though the majority of the sky is relatively dark at night, there could still be some color and brightness gradients in different parts of the sky as well. Finally, there will often be objects in the foreground that will need to be separated from the remaining star streaks in the background.

4.1. Thresholding

We perform three types of thresholding to segment the star streaks from the rest of the image. Each method is performed independently before combining to minimize the chance for unwanted segmentations to be compounded forward, as shown in Figure 2.

First, we use global thresholding by looking at the grayscale versions of the original images. If the pixel value on the black and white image is greater than a global threshold, we consider it part of a star streak in the color image. This will allow us to catch the brightest star streaks, while retaining the color information. However, this could cause large patches of images to be considered as stars and removed from the image if their grayscale values are too high. To address this, we perform a second iteration through the image and refill the pixels with their original values if the proportion of information in a given window that was removed exceeded a certain percentage, as shown in Figure 3.

We then use Otsu's method as a clustering-based image thresholding technique to effectively separate the background of the image from the foreground while segmenting a different set of star streaks [3]. Using a binary version of the original image, Otsu's method calculates an optimal threshold such that the intra-class variance is at its maximum while the inter-class variance is at its minimum. The stars are then segmented based on this calculated threshold and removed from the image. Again, to minimize the amount of dark patches, we iterate through the image again and replace the removed pixels with their original values if too much information was removed in a given region.

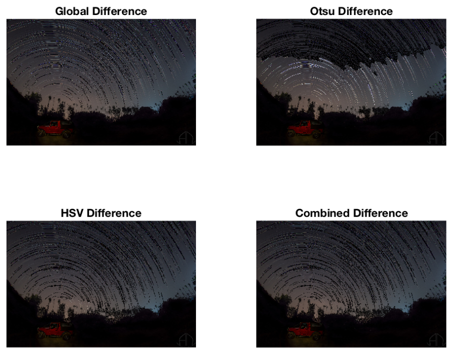

Figure 2. Different thresholding methods

Finally, we perform thresholding by looking at the HSV values converted from the RGB image. Converting the image to HSV representation allows us to look at the hue, saturation, and value separately unlike in the RGB space. This will help us to catch some of the more faded streaks in the image. After converting the the HSV space, we look at the value array and only segment out the pixels that have a higher value in comparison to a modified mean. Similar to the previous thresholding methods, we post-process the resulting image to remove the dark spots.

4.2. Width Estimation

To take into account the individual star streaks could be varying in brightness and color along its path, we also include the neighboring pixels that are within a certain gradient from the pixels that were removed by the thresholding methods. This will allow us to capture a more representative portion of the star streak while leaving less behind, as even though the brightness tends to be highest in the center of the streak, dimmer portions along the width of the star are also part of the streak. This method also helps account for varying streak widths, in comparison to just using a defined region size of neighbors, as shown in Figure 4.

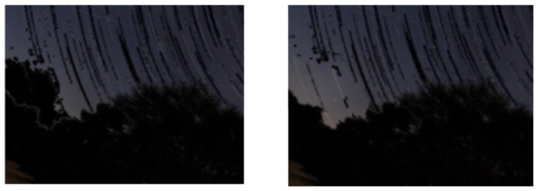

Figure 3. Patch removal: before vs. after

Figure 4. Width estimation

4.3. Combining Methods

After performing the three independent thresholding methods and accurately estimating the star widths, we combine the resulting star streaks in order to maximize the number of streaks we remove from the image segmentation. As an extra check, we look at the amount of information that had been taken out of the original image before combining the results. As certain methods would work better than others on images with varying parameters such as illumination and quality, we only combine the ones that removed a balanced amount of pixel values so that we do not lose too much information in comparison with the original image, as shown in Figure 2.

We also experimented with other segmentation techniques such as edge detection, but the results were even worse than those of global thresholding. This may be because the star streaks are so densely packed in the data images. Other problems included many artifacts along the foreground/background boundaries, especially along the horizon. As such, we balanced multiple thresholding methods instead, which gave significantly better results.

5. Star Streak Modification

5.1. Polar PSF Calculation

After identifying the star streaks, the next step is to transform the streaks into individual stars, as detailed in Figure 1. Because the star streaks are caused by the rotation of the earth, they form a circular pattern

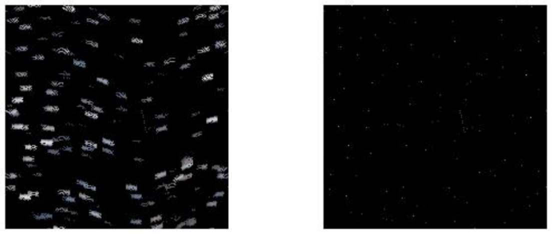

Figure 5. Polar transform vs compressed clusters

around the celestial north pole, as shown in Figure 5. Because of this, there is not a uniform ”point spread function” applied to the stars in Cartesian coordinates that could be used to perform deconvolution. We transform the star streak image into polar coordinates with the origin at the celestial pole to have each individual streak appear as an approximately horizontal line of a uniform length [1].

A few possibilities exist to identify the star's axis of rotation. An algorithm is used to automatically locate the coordinates of the axis, even if the axis lay outside the image [1]. We chose to instead test our procedure exclusively on images in which the celestial pole lay within the image, and to have the user manually indicate the location of the axis by clicking on it when prompted.

5.2. Star Clustering

After transforming the streaks into polar coordinates, we were able to convert them into points. Initially, deconvolution based on a uniform point-spread function was attempted. However, the star streaks in the images tested did not form uniform circles, and so did not form perfectly uniform lines in polar coordinates. Deconvolution, then, was not an effective strategy. Instead, a recursive clustering algorithm was implemented. In this algorithm, a bright pixel was identified, and its neighbors were added to a cluster if they were not surrounded entirely by dark pixels. This process was repeated until the complete cluster had been formed. The cluster (i.e. an individual streak) was then compressed into a single pixel, by averaging the values of all pixels in the cluster, and setting the values of each cluster in the pixel to zero, except for the center pixel, which was set to this average value. This procedure was then repeated until each cluster had been identified, as shown in Figure 5.

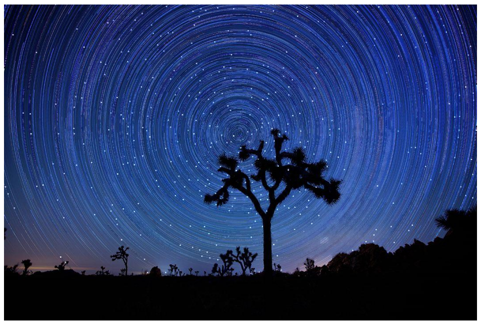

Figure 6. Star replacement

5.3. Star Replacement

After blurring the image to remove the dark patches and transforming the star streaks into individual pixels, these images must be combined. The star pixels are still in polar coordinates, so the clustered and compressed image is converted back to Cartesian coordinates, with the origin again at the celestial pole. After this transformation, the stars are correctly placed in space. However, they consist of single pixels, which will not appear aesthetically pleasing to a user. The image containing only the star pixels is convolved with a 3-pixel wide Gaussian kernel in order to cause the stars to appear broader. The standard deviation of the Gaussian was made to depend on the size of the image, so as to avoid creating huge stars in small image of the sky. At this point, any stars that overlap with each other are removed. This is done because any pixels thought to be stars that are found so close to each other are almost certainly artifacts, originating from background details that were not removed during the segmentation process. After this widening and filtering of the star image, the background image and the star image are simply added to each other.

6. Image Post-processing

6.1. Hole Filling

After segmenting the bright star streaks from the image, many empty holes remain in pixel locations where the star streaks originally occupied. In order to remove these dark spots, we fill these holes with interpolated values from the surrounding pixels using both a linear and a two-dimensional averaging function. First, we iterate through the pixels in the image where the stars have been removed. For each horizontal line of connecting pixels that have empty values, we replace them with the average value of the edge pixels that bound them. Then, we run a two-dimensional averaging function in the remaining darkened regions of the interpolated image in the night sky, as shown in Figure 7.

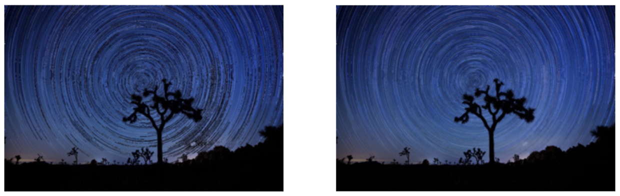

Figure 7. Hole filling: before vs. after

6.2. Bilateral Filtering

After filling the holes, parts of the night sky still seem noisy in the images, with sometimes sharp color changes between neighboring pixels. To diminish some this noise, we use a bilateral filter based on a Gaussian distribution to smooth the image. This is applied to the images with high enough resolution, where the modified window size will not impact the newly added point stars. One of the benefits of bilateral filtering is that it allows us to preserve edges while filtering, in comparison to a simple median filter. This is useful because we want to try to preserve the edges in the foreground while smoothing out the night sky in the background. The point stars are also smoother and better blended into the background night sky.

7. Results and Discussion

Qualitatively, the images produced by the algorithm are visually pleasing, as shown in Figure 8. Stars are localized at the center of what was a star streak in the original image and retain their color. The background sky is filled in and blurred so the spaces where star streaks were removed do not appear harsh to the viewer, and this process works well even in images with color gradients. The edges are also preserved well qualitatively when bilateral filtering is performed, but the degree is difficult to measure quantitatively due to the lack of ground truth images. Some problems do remain, however. For instance, the segmentation is not perfect so some star streaks, especially fainter ones, are not removed. Some extremely bright sections of

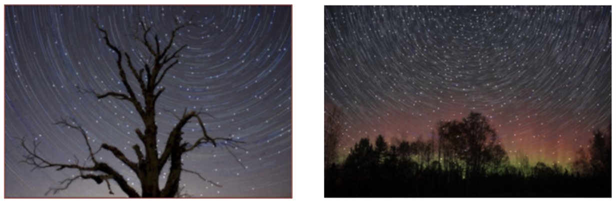

Figure 8. Sample processed images

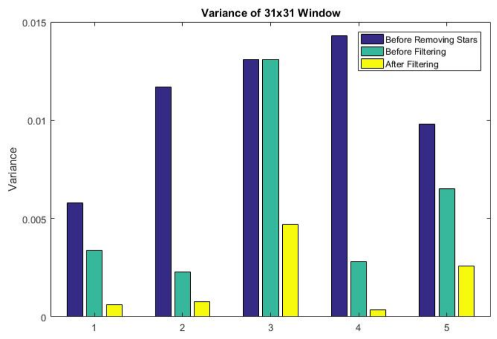

Figure 9. Variance

the foreground may also be interpreted as stars, resulting in false ”stars” in the foreground. The filling-in process is also not ideal, so the background can appear somewhat blotchy. All stars are also the same shape and size, which is not true for the real sky.

Quantitatively, in order to evaluate how well the star streaks were removed, we measured the variance of a window of the original color image and compared it to the variance of the same patch after hole filling. The lower the variance, the more star streaks were removed. As shown in Figure 9, we see that for a 31x31 pixel window surrounding the celestial axis, the average variance decreases from 1.09e-2 to 5.62e-3, which means that a significant amount of star streaks were removed. In order to evaluate the denoising done by the bilateral filter, we again measure the variance, but this time of a window of the image before inserting the stars, both before and after the bilateral filter was applied. This approximates the noise level of the image. As shown in Figure 9, the variance decreases after the bilateral filter is applied across all images. Taking the average, the variance decreases from 5.62e-3 to 1.81e-3. In addition, we find the timing for our algorithm to be quite efficient and mainly dependent on the image resolution. On average, the full image processing algorithm ran in 20 seconds for medium-sized images (around 500k pixels) and 80 seconds for large, high-quality images (around 3 million pixels).

8. Conclusion and Future Work

We were able to create an algorithm that removes most star streaks from an image, and replaces them with appropriately-colored point-stars. Some streaks remain, causing the sky to appear much uniform than in the unedited image, but by denoising using bilateral filtering, we are able to create more aesthetically pleasing images that accurately represent the placement of stars in the sky. To further improve the process, we could investigate other image segmentation procedures, including machine learning algorithms, that could allow us to more effectively distinguish star streaks from the rest of the image. The stars are also currently the same shape, which is not representative of actual stars, so we can consider better visualization methods such as incorporating the width estimation. We could also develop better strategies for filling holes left in the image by the removal of star streaks. While we were unable to obtain our own images during the course of this project, we would also want to test this algorithm on images obtained directly from a camera, without any post-processing.

Contributions

Joseph: data collection, thresholding, width estimation, other segmentation, polar PSF, hole filling, denoising, results and discussion

Peter: data collection, polar PSF, clustering, star replacement, results and discussion

References

[1] B. D. Jeffs and M. Gunsay. Restoration of blurred star field images by maximally sparse optimization. *IEEE Transactions on Image Processing*, 2(2):202–211, April 1993.

[2] L. W. H. W. Y. H. Laili Su, Xiaopeng Shao. Richardson-lucy deblurring for the star scene under a thinning motion path. 9501, 2015.

[3] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, Jan 1979.

[4] B. Sease and B. Flewelling. Polar and spherical image transformations for star localization and rso discrimination. 01 2015.