Motion-corrected image denoising for digital photography

Brendan Duncan
Stanford University
brendand@stanford.edu

Abstract

An effective way to reduce noise in images involves taking a burst of snapshots and averaging them together. This requires that they be aligned, and that there is no motion or parallax across the images, otherwise this averaging causes motion blur and ghosting effects.

In this paper, I introduce a new technique for both the alignment and averaging of a burst of photos of a single scene. The alignment is performed using multiple applications of SIFT and RANSAC to match both background and moving foreground objects in a scene. A weighted average is then calculated at each pixel using the bilateral filter to provide an estimate of the denoised result. This weighted average reduces the contribution from noisy pixels as well as from unmatched pixels resulting from motion across images.

1. Introduction

Noise in digital cameras comes from many sources. Some noise, such as fixed-pattern noise and hot pixel noise, is roughly constant across each image. Other noise, such as photon shot noise, dark current noise, and read noise, is not constant across successive images. In particular, photon shot noise is caused by variation in the number of photons arriving at each pixel on the sensor, and depends on the Poisson distribution.

An equation for signal to noise ratio in digital cameras is given below:

$$SNR = \frac{PQ_e t}{\sqrt{PQ_e t + Dt + N_r^2}},$$

where P is the number of photons per second, Q_e is the quantum efficiency, t is the exposure time, D is the dark current noise, and N_r is the read noise. Noise that is constant across images is not shown in the equation because it can be calculated and removed easily.

The above equation shows that increasing the exposure time will increase the signal to noise ratio. However, it is not always possible to increase the exposure time. For example, if the camera is handheld, a long exposure image will be blurry because of camera shake.

Instead, the photographer can take a burst of short exposure images, align them, and average them together. Since averaging together several exposures is effectively increasing the exposure time, the equation shows that this will increase the signal to noise ratio.

Unfortunately, aligning and averaging successive images can introduce motion blur and ghosting effects if there is any motion or parallax across the images.

In this paper, I introduce a new technique for both the alignment and averaging of a burst of photos of a single scene. The alignment uses multiple applications of SIFT and RANSAC to match both background and moving foreground objects in a scene. This ideally matches all regions of a base image to regions in the warped images, even if there was motion in the scene, so that every pixel can be averaged to reduce noise. A weighted average is then calculated at each pixel using the bilateral filter to provide an estimate of the denoised result. This reduces the contribution from noisy pixels as well as from mismatched pixels resulting from motion across images.

This is designed as a post-processing technique with photography in mind. It does not require a high frame rate sequence of images, as do many video denoising algorithms. The input is a burst of images, and the output is a single low-noise version of a user-specified base image.

2. Previous work

A technique explained by Adams *et al.*[1] describes a way to quickly align viewfinder frames. An application of their alignment involves averaging successive frames to remove noise, but their results showed amplified motion blur when an object in the scene moves.

There are several approaches for combining aligned images that do attempt to deal with motion across images. These techniques are common in HDR imaging, wherein several aligned images of different exposures are combined to create a high dynamic range image. Many of these methods, such as that proposed by Gallo *et al.*[9], attempt to

remove the moving object completely from the resulting combined image. I take a different approach, assuming that the user would like the resulting image to include any foreground objects that were present in the base image. This assumption is consistent with a use case where the user takes a burst of images. In this case, the user wants the resulting picture to capture all objects in the scene at the moment when the shutter release was pressed.

Methods for computing dense optical flow, such as that proposed by Lucas and Kanade[12], attempt to detect the motion of pixels across images. These techniques could in theory be used to detect areas where motion is occurring, and then match and average these pixels with the corresponding displaced pixels in another image. In practice however, I found that dense optical flow techniques were not able to accurately describe object motion unless the object had moved very little between the two frames. Although this would be a feasible solution for denoising a high frame rate video, I did not want to restrict the method to dealing with this case. Examples of optical flow-based motion segmentation techniques are described by Ogale *et al.*[13] and Wong *et al.*[18].

There are also several video denoising methods that essentially extend existing image denoising methods to videos. An example of a video denoising technique that modifies the non-local means[4] image denoising algorithm to work with videos is proposed by Seo *et al.*[16]. Another example of an image denoising technique being extended to video denoising is given by Potter *et al.*[15], which uses learned dictionaries and sparse representations[6]. Both of these techniques work by treating a video as a single three dimensional volume, and applying a slightly modified version of the original image denoising algorithm. They often rely on groups of pixels that are similar both in the space and time dimensions. These methods are less useful for photographers taking a burst of photographs because the low frame rate allows large motion between successive images, and thus there may be large changes across the time dimension. Because of this, in regions with motion, a group of similar pixels cannot be extended in time, which roughly reduces the video denoising technique to the original image denoising algorithm.

At the core of the method described by Bennett and McMillan[3] is the Adaptive Spatio-Temporal Accumulation (ASTA) filter. This is similar to the bilateral filter[17] but modified to work in time and space rather than just space (although it does not simply treat the video as a 3D volume as in the methods described above). The ASTA essentially performs a bilateral filter across time when possible, but in regions with mismatched areas due to motion, performs averaging across space instead. Filtering across time is preferred because spatial filtering causes loss in textures and details. The method I propose attempts to determine the motion and align unmatched regions using a perspective warp so that, ideally, only temporal filtering need be performed.

3. Feature detection

The first step of the algorithm is to use the scale-invariant feature transform (SIFT)[11] to detect and describe features in each of the images. The SIFT descriptors corresponding to the features of the base image are stored in a k-d tree[8], which will allow us to perform nearest neighbor search to find shared features across successive images.

4. Fit calculation and perspective warp

After calculating features for each of the images, the next step is to align the images based on these features. To do this, I calculate a perspective warp that aligns a large number of features in a given image to those in the base image. First, I find shared features between each of the unaligned images and the base image using best bin first search[11]. Since it is assumed that each image is of roughly the same scene, there should be few outliers for a perspective warp fit. Therefore, I use RANSAC[7] to calculate a perspective warp from each image to the base image. RANSAC will randomly choose a small subset of shared SIFT features, and determine the best perspective fit to match these features by calculating the least-squares solution to the following matrix equation:

$$\begin{bmatrix} x_1 & y_1 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_1 & y_1 & 0 & 1 \\ x_2 & y_2 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_2 & y_2 & 0 & 1 \\ & & \cdots & & & \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \cdots \\ \cdots \end{bmatrix}$$

This equation is derived from the affine warp and translation for a given point in an image:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

where x and y are the feature coordinates in the image to warp, u and v are the displaced pixel coordinates (the coordinates of the matched feature in the base image), the matrix **M** represents the affine warp (rotation, scale, and stretch), and **t** represents a translation.

After calculating a fit for a random sample, RANSAC calculates an error for the fit across all features shared between the two images. It selects other random samples, calculates a fit for those, then chooses the fit with least error after a predetermined number of iterations.

The first RANSAC fit will align the regions of the scene with the most features. Generally this means that

the background objects are aligned first (although it may be that foreground objects are aligned first, if there are more matched features in the foreground). This background alignment accounts for camera motion between the two pictures.

Background Alignment

After the first fit is calculated, I run RANSAC again, but this time using only the outlying SIFT features from the first RANSAC fit. This will align regions that were not aligned previously; if the background was aligned in the first application of RANSAC, foreground moving objects will be aligned in this step.

Foreground Alignment

I continue running RANSAC on outliers until no further fit can be found. This is done to allow matching of multiple differently moving objects. Ideally, these multiple applications mean that all features will be matched in the base image, and that noise can be reduced in all regions of the base image.

5. Calculating a per-pixel weighted average

At this stage, we have a base image and a sequence of warped images. This sequence of images may be larger than the original input sequence because multiple perspective warps may have been performed on a single input image. For a given warped image, some regions match those in the base image, but not necessarily all, because of the possibility of independently moving objects in a scene.

I produce a result image by combining each pixel in the base image with corresponding pixels in each of the warped images using a weighted average. The weighted average should assign a negligible weight to mismatched areas; it should also assign less weight to noisy pixels.

With these goals in mind, I apply the bilateral filter[17] to the base image to get an estimate of the denoised image. A bilateral filter is a good model because it is a nonlinear filter that determines weights given to surrounding pixels based not only on their physical distance, but also on the difference in their values, with larger distances given smaller weights. It therefore ideally does not average across edges, which roughly means that each pixel is only averaged with pixels in the same image region. This type of average reduces noise in each region without blurring edges. There are many fast approximations of the bilateral filter, such as the one described by Paris and Durand[14].

The bilateral filter formula for color images is given below:

$$BF[\mathbf{I}]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S_{\mathbf{p}}} G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|)\, G_{\sigma_r}(\|\mathbf{I}_{\mathbf{p}} - \mathbf{I}_{\mathbf{q}}\|)\, \mathbf{I}_{\mathbf{q}},$$

where G is the Gaussian function

$$G_{\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}},$$

$\mathbf{p}$ and $\mathbf{q}$ are x, y pixel coordinates and $\mathbf{I}$ is three channel image. The Gaussian functions G_{σ_s} and G_{σ_r} provide the weights for the weighted average of the set $S_{\mathbf{p}}$ of pixels $\mathbf{I}_{\mathbf{q}}$ surrounding pixel $\mathbf{I}_{\mathbf{p}}$. $W_{\mathbf{p}}$ is the sum of the Gaussian weights calculated for each pixel in $S_{\mathbf{p}}$ ($\frac{1}{W_{\mathbf{p}}}$ is the normalizing term). The above equation is calculated at every pixel $\mathbf{I}_{\mathbf{p}}$ in the image. σ_s and σ_r are user defined parameters that represent the standard deviations for the distance between pixel locations in x, y space and between pixel values in RGB space, respectively.

To calculate the per-pixel weighted average, I follow a similar method to the bilateral filter, except that no spatial filtering is done and the Gaussian weight is determined based on the distance between the bilateral filtered estimate and the warped input image or base image in RBG space:

$$\mathbf{R}_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{i=0}^{n} G_{\sigma_r}(\|BF[\mathbf{I}_0]_{\mathbf{p}} - \mathbf{I}_{i\mathbf{p}}\|)\, \mathbf{I}_{i\mathbf{p}}$$

$\mathbf{I}_0$ is the base image and $\mathbf{I}_i$ for $1 \leq i \leq n$ is one of the warped images. The σ_r here need not be the same as the one used for the bilateral filtered estimate. $\mathbf{R}$ is the result image of the algorithm.

Note that the pixels from the bilateral-filtered estimate do not contribute to the result, but only provide the mean

of the Gaussian distribution. Therefore, no spatial filtering occurs, but only filtering across time, which is better for preserving textures.

Below is an example of the weights calculated for one of the warped images. It is a grayscale image with the highest weight being assigned a value of full white. The perspective warp calculated for this particular image was for background alignment, so the bicyclist is mostly black, while the background is mostly white.

Weights calculated

This weighted average is better at reducing noise than a simple average, as shown in the following two images. The first is a simple average of three aligned images; the second is the result of the proposed weighted average.

Simple average

Proposed weighted average

6. Correcting weights for unmatched objects

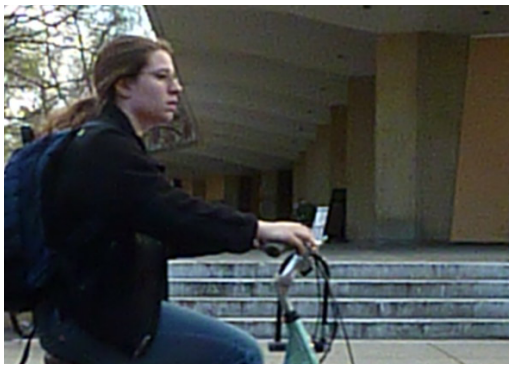
Photon shot noise is roughly expected to follow a normal distribution because a Poisson distribution approaches a normal distribution for a large expected mean.

Although noise roughly follows a normal distribution, pixels that include motion are not expected to. For example, a pixel in the base image may contain a foreground object that has moved in subsequent images. Assume that the object has left the scene in the majority of the images, and therefore is not matched in most of the warped images. Although these unmatched pixels will have a low weight based on the weighted average described above, there is no limit to the number of unmatched images, so the weights of the unmatched pixels could become large enough to contribute noticeably to the pixel in the final image. Therefore, the weighted average should be slightly modified to account for this corner case (in most cases this is not necessary). When pixel weights are determined, first average together pixels with similar weights, using the same weighted average, then average the results together as normal, with weights determined based on the average of the previous weights. This limits the effect of a large number of unmatched pixels.

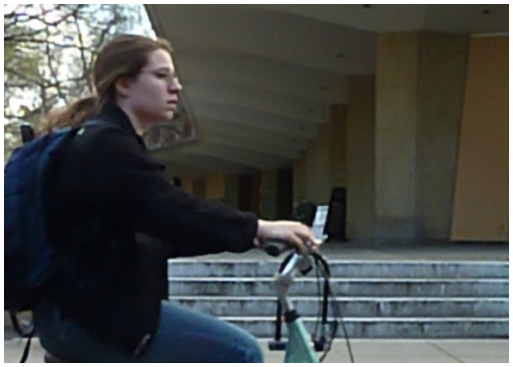
One can classify pixels as having similar weights using the distance from the expected mean. This can be a user-defined parameter. If the value is $\frac{\sigma_r}{5}$, then pixels with, for example, $2\sigma_r$ to $2.2\sigma_r$ distance from the bilateral filtered estimate image would be averaged together before being averaged with other pixels.

7. Results and comparisons

In the example shown below, noticeable noise reduction is visible in both the background and the foreground (bicyclist). This result was achieved with an average of six images.



Original image



Result of algorithm

The algorithm is also useful for denoising a burst of images in which parallax makes simple alignment and averaging impossible, as shown in the following images.

Simple average

Base image

Result of algorithm

The result shown here is the combination of six images. There is a significant reduction in noise in both the foreground object and in the background. The weighted average has also prevented any ghost images from the unmatched regions due to parallax.

Because this algorithm averages only in time and not in space, textures will be well-preserved. The following images compare a bilateral filtered image with the result of the algorithm, using the same σ_r for both.

Result of bilateral filter

Result of algorithm

8. Conclusion and Future Work

Results may improve slightly if a perceptual color space is used to determine color differences. Examples of perceptual color spaces I would like to test results in are CIE L*a*b* and the one described by Chong, *et al.*[5].

Another possible way to improve this method would be to use an adaptive σ_r when computing weighted averages, such as the one proposed for the adaptive bilateral filter[10]. This has the advantage of determining a good σ_r for different regions without requiring that the user adjust this parameter.

Both of these potential improvements have been shown to improve the results of the bilateral filter, so I predict they would work well with my technique.

An improvement in performance could potentially be made by using a different feature detector besides SIFT. A good candidate might be SURF (Speeded Up Robust Features)[2]. However, depending on the scene and type of motion, it may not be necessary to use scale-invariant features, which could mean an even further increase in performance.

References

[1] A. Adams, N. Gelfand, and K. Pulli. Viewfinder alignment. In *Eurographics*, 2008.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding*, 2008.

[3] E. P. Bennett and L. McMillan. Video enhancement using per-pixel virtual exposures. *ACM SIGGRAPH 2005 Papers*, 2005.

[4] A. Buades, B. Coll, and J. Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 4:490–530, 2005.

[5] H. Y. Chong, S. J. Gortler, and T. Zickler. A perception-based color space for illumination-invariant image processing. In *ACM SIGGRAPH 2008 papers*, New York, NY, USA, 2008.

[6] M. Elad and M. Aharon. Image denoising via learned dictionaries and sparse representation. In *International Conference of Computer Vision and Pattern Recognition*, 2006.

[7] M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[8] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–226, 1977.

[9] O. Gallo, N. Gelfand, W. Chen, M. Tico, and K. Pulli. Artifact-free high dynamic range imaging. In *International Conference on Computational Photography*, 2009.

[10] C. Liu, W. T. Freeman, R. Szeliski, and S. B. Kang. Noise estimation from a single image. In *Conference on Computer Vision and Pattern Recognition*, 2006.

[11] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[12] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of Imaging Understanding Workshop*, pages 121–130, 1981.

[13] A. S. Ogale, C. Fermller, and Y. Aloimonos. Motion segmentation using occlusions. *Transactions on Pattern Analysis and Machine Intelligence*, 27(6):988–992, 2005.

[14] S. Paris and F. Durand. A fast approximation of the bilateral filter usinga a signal processing approach. In *Proceedings of European Conference on Computer Vision*, 2006.

[15] M. Potter and M. Elad. Image denoising via learned dictionaries and sparse representation. *Transactions on Image Processing*, 18(1), 2009.

[16] H. J. Seo and P. Milanfar. Video denoising using higher order optimal space-time adaptation. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2008.

[17] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the Sixth International Conference on Computer Vision*, 1998.

[18] K. Y. Wong and M. E. Spetsakis. Motion segmentation and tracking. In *International Conference on Vision Interface*, volume 15, pages 80–87, 2002.