

ABSTRACT

This paper presents a method for deblurring distorted images using a known convolution kernel. Pre-existing methods and algorithms are presented, each with its advantages and disadvantages. In this implementation, Gaussian kernels are considered for blurring. The algorithmic approach is detailed and the sampled results show the effectiveness of the chosen method.

KEYWORDS: image deblurring, non-blind image deconvolution

1. INTRODUCTION

Image blurring is an undesired artifact commonly found in photography, as a result of camera shake. There are several ways to reduce this unwanted alteration [14][7]. One way is to use faster exposures when taking photographs, but this brings up more disadvantages such as increased noise and decreased depth-of-field. Moreover, not all cameras are capable of this adjustment of settings. Placing the camera on a tripod is another solution, but tripods are space consuming and most photographers (the majority of which are amateur) do not normally carry them around. A software approach, which can be applied to photographs taken by any device, is therefore the most comfortable solution.

Solving the image automatic deblurring problem may help in numerous situations where blur presence leads to numerous and consistent errors in image processing like segment misclassification [13], erroneous (hand)writing detection [8][9][10], faulty acquisition device calibration [11][12], etc.

In computer science, blurring an image is equivalent to applying a filter known as Point Spread Function (PSF) to a sharp image. This is done by convoluting the original image, which will further be referenced as the true image, with the PSF, and optionally adding noise. Therefore, the process of blurring can be expressed mathematically as follows:

$$B = I \otimes K + N$$

In this equation, B is the resulting blurred image, I is the original, K is the convolution kernel (or PSF) and N is the optional additive noise distribution. This expression reveals that deblurring can be reduced to the problem of finding I given B and, optionally, K and N .

If the kernel is known, the deconvolution is called blind. Similarly, if it remains unknown, it is called non-blind. In this case, it needs to be estimated using the blurred image in order to apply deconvolution and obtain a better result.

This paper presents a method for blind image deconvolution based on one of the best known algorithms in this domain called Richardson-Lucy. The original algorithm is improved by adding padding to the edges of the blurred image. This reduces the ringing artifact, which

¹ Engineer, alexandra.ghecenco@cti.pub.ro, "Politehnica" University of Bucharest, 060042 Bucharest, Romania

is an undesired, yet common occurrence. In order to emphasize the effectiveness of the algorithm, the application takes sharp images as input, and then applies a Gaussian kernel of user defined size. The user can then pad the blurred image with custom sized borders before deblurring. The Richardson-Lucy algorithm is iterative; each iteration results in an intermediate output which is presented to the user, who chooses when the deblurring loop should end. Although this algorithm does not have optimal results on Gaussian kernels, the results are quite satisfactory.

The user is free to experiment with kernels and borders of various sizes, as well as different steps of deconvolution, and save the resulting images in order to compare them and decide which combination is closer to the optimal one.

2. RELATED WORK

Jong-Ho Lee and Yo-Sung Ho [1] propose a very effective technique for non-blind deconvolution, a key component of which is adaptive regularization. This consists of extending the blurred image with borders that will help preserve edge information while restoring the latent image. Their paper starts with presenting the ringing artifact introduced by the Richardson-Lucy algorithm. The article explains how this effect is caused by the propagation of errors from the borders inwardly, since pixel information beyond the borders of the image is unavailable. Deconvolution, like its inverse operation, convolution, consists in applying a kernel to the larger image matrix. Naturally, erratic results appear near the edges, where the information for reconstructing the image window is incomplete. These errors are further propagated towards the center and other edges of the image, depending on the corner of start in the deconvolution process. Incorrect pixel values form waves, or rings, in the reconstructed image.

Their solution to the ringing problem is to extend the blurred image with borders that provide a natural continuation of the pixel information near the edges. These borders are processed together with the blurred image and can then be trimmed out of the result. There are more ways of filling this extra space. The present project replicates the last line and the last column respectively to achieve the user specified size. Ho and Lee's paper proposes a mathematical approach for a non-uniform, yet equally useful extension:

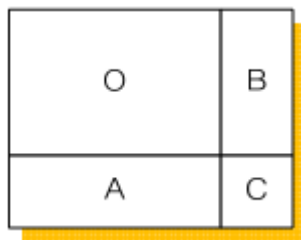
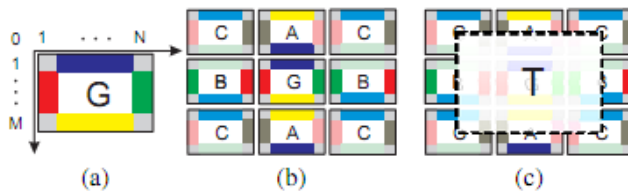


Figure 1. Expansion image. O – original, A,B,C – padding blocks

Their idea is to extend the original image O with three supplementary blocks, first A or B, then C as a combination of the previous two. The formulas they propose can be adjusted and guarantee the preservation of information. The approach presented in this paper is a particular case of their general formulation.

Renting Liu and Jiaya Jia [2] also relate to reducing boundary artifacts in their paper. Their approach is more complex than the simplistic addition of borders. Under the assumption that the convolution kernel is known, they propose generating an extrapolated image from

the blurred one using tiles, which they define as rectangular image blocks that follow certain patterns nearing the edges. Therefore, by tiling a set of blocks in a certain order, it can be guaranteed that no distinct discontinuities be observed between neighboring blocks.



Tile generation.(a) A base block, representing the input image. (b) A set of tiles obtained from the original block arranged in such a way that the boundaries of adjacent blocks have the same intensity values. (c) A rectangular tile (T) composed from the arranged blocks.

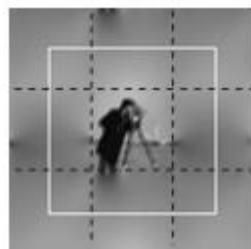


Figure 2. Tile generation. The tile to be used is marked by the rectangle in the middle. The dashed lines mark the blocks.

After extending the original to a tile as shown, the Richardson-Lucy deconvolution algorithm is applied in the frequency domain. All operations are performed on the Fourier transform of the image. In this case, an iteration step in the algorithm comprises in a fast Fourier transform (FFT), a deconvolution step on the result and the inverse transform.

More advanced techniques can be found in the works of Li Xu and Jiaya Jia [3] and Rob Fergus, Barun Singh, Aaron Hertzmann, Sam Rowels and William Freeman [4]. Their papers concentrate the effort on kernel estimation, since their main point of interest is non-blind deconvolution. Xu and Jia construct an edge map of the blurred image and proceed in reconstructing the kernel based upon it. After a sufficient number of iterations, when the outcome is considered adequate enough, a fast deconvolution algorithm is employed. Their results are spectacularly clear and the algorithm, although difficult to implement, is fast.

The other five authors focus their attention on deblurring shaky photographs with no additional information. They use a mathematical model of the distribution of gradients found in most photographs, basing their claims on extensive research in the domain of photography analysis. By superimposing the gradient map of the blurred image given as input with a “standard” one, obtained either mathematically or empirically, from correct photographs, they estimate what the motion that generated the errors must have been. User input is also valuable, since most times the photographer can give an idea on how the camera may have moved, therefore providing a valuable point of departure in the reconstruction. Using the resulting kernel, they employ a standard deconvolution algorithm, obtaining surprisingly accurate and clear images.

3. CONTENT OF THE APPLICATION

The purpose of this application is to provide a simple, fast and effective way of deblurring images, given the blur kernel. The algorithm of choice is a variant of Richardson-Lucy. In order to improve its results, padding was added to the blurred image as presented earlier. A key element of this application is the full control the user has over the whole process: blurring, edge padding and deconvolution. For the purpose of simplicity, images can be blurred in the application with a custom size Gaussian kernel. Then, the user specifies the width and height of the padding borders. After adding them, each iteration of the Richardson-Lucy algorithm is run on the intermediate results, which are visualized by the user. Upon reaching a satisfactory result, the iterative loop can be broken and the final result saved. The following sections detail each operation.

The Algorithm

Blurring the original

The Gaussian distribution is one of the simplest filters that can be applied to an image in order to blur it, or smooth it out. The expression of the Gaussian filter is as follows:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

where x is the distance from the point of origin in the horizontal axis, y is the distance from the point of origin in the vertical axis and σ is the standard deviation of the Gaussian distribution. When applied in two dimensions, on an image, for instance, this formula alone produces a surface whose contours are concentric circles with a Gaussian distribution from the center point. The values of this distribution comprise a convolution matrix, the filter which is applied to the image. This results in each pixel's new value being set to a weighted average of its neighbors' values.

In theory, the Gaussian function at every point on the image will be non-zero, meaning that all the pixels in the image affect the value of a single new pixel. In practice, when working on discrete images, the contribution of pixels further than 3 positions away from the source is negligible.



Figure 3. A sample image and its blurred counterpart using a Gaussian kernel

Padding the edges

As previously detailed, running Richardson-Lucy directly on the blurred image could lead to ringing ripples appearing in the reconstructed image. In order to overcome this, the blurred image can be padded by replicating a customizable number of times the bottom-most line and the right-most column. These form the A and B blocks described in the previous section, with the C block as their intersection.

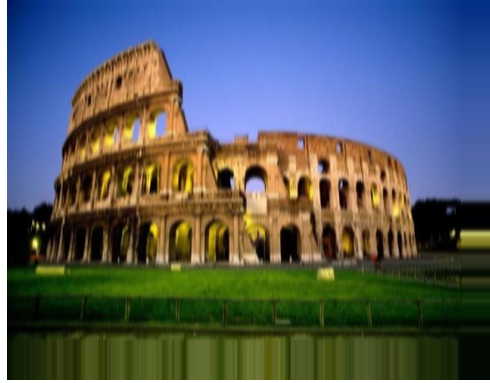


Figure 4. The padded blurred image

Deconvolution

As previously mentioned, the deconvolution algorithm of choice is Richardson-Lucy. This is an iterative procedure which takes as input the blurred image and the PSF (in this case, the Gaussian distribution) and applies a sequence of operations in order to reconstruct the original image. There are more variants of this algorithm. The best known one uses as initial image a random distribution of pixels, adding value to them at each iteration step.

Pixels in the blurred image can be expressed using the PSF and the original (unknown) image:

$$d_i = \sum_j p_{ij} u_j$$

The notation are as follows: d_i is the value of the i th blurred pixel, p_{ij} is the value of the PSF coming from the location j and observed in location i , and u_j is the value of the j th pixel in the original, unknown, image. The purpose of the algorithm is to compute the most likely u_j , one step at a time. This leads to an equation for u_j that can be solved iteratively:

$$u_j^{(t+1)} = u_j^{(t)} \sum_i \frac{d_i}{\sum_j p_{ij} u_j^{(t)}} p_{ij}$$

The first estimation can be a uniform or random distribution of values, which converges towards the original with each step. Although this solution is simple and straightforward and does find the result, it can be very time consuming due to the large number of iterations that have to be performed.

This project proposes an approach that leads to the result significantly faster. The solution is remarkably simple – use the blurred image as initial estimation and, at each iteration step, determine the error an additional blurring with the known kernel would introduce. This error can be used to reverse the blurring effect.

Reconsider the initial blurring equation:

$$B = I \otimes K + N$$

$$B_{ij} = I_{ij} \otimes K_{ij} + N_{ij}$$

Consider there is no noise:

$$B_{ij} = I_{ij} \otimes K_{ij}$$

Leading to:

$$B_{ij} - I_{ij} \otimes K_{ij} = 0$$

$$I_{ij} = I_{ij} + B_{ij} - I_{ij} \otimes K_{ij}$$

$$I_{ij}^{(t+1)} = I_{ij}^{(t)} + B_{ij} - I_{ij}^{(t)} \otimes K_{ij}$$

Or, in a simplified form:

$$I_{ij}^{(t+1)} = I_{ij}^{(t)} + E^{(t)}$$

$$E^{(t)} = B_{ij} - I_{ij}^{(t)} \otimes K_{ij}$$

4. IMPLEMENTATION

The application is implemented in C++. It is compiled for 32bit architectures and runs on all Microsoft Windows distributions from Windows XP to Windows 8.1 (tested on Windows XP, 7 and 8). The GUI is based on Microsoft's MFC framework [5]. The image operations are performed using the OpenCV library [6].

5. RESULTS

Here are a set of sample images and the results obtained using the described algorithm.



Figure 5. Fig. 6. Coliseum – original



Figure 6. Fig. 7. Coliseum – blurred with 15px radius Gaussian filter



Figure 7. Fig. 8. Coliseum – restored using 5 iterations in my application



Figure 8. Fig. 9. Coliseum – restored by the Gaussian filter removal tool in Adobe Photoshop CS3 for a 15px radius filter



Figure 9. Fig. 10. Shark – original



Figure 10. Fig. 11. Shark – blurred with 15px radius Gaussian filter



Figure 11. Fig. 12. Shark - restored using 4 iterations in my application



Figure 12. Fig. 13. Shark - restored by the Gaussian filter removal tool in Adobe Photoshop CS3 for a 15px radius filter



Figure 13. Fig. 14. Zebras – original



Figure 14. Fig. 15. Zebras - blurred with 15px radius Gaussian filter



Figure 15. Fig. 16. Zebras - restored using 4 iterations in my application



Figure 16. Fig. 17. Zebras - restored by the Gaussian filter removal tool in Adobe Photoshop CS3 for a 15px radius filter

6. CONCLUSIONS

The method presented can successfully remove Gaussian blur from distorted images and renders better results than the specialized tool included in Adobe Photoshop CS3. The standard Richardson-Lucy deconvolution algorithm has been improved by using the blurred image as initial estimation; leading to a significantly small number of steps (4-5 deconvolution iterations are sufficient to reconstruct a fairly satisfactory sharp image). Furthermore, padding has been added to the blurred image before deconvolution in order to minimize the ringing effect. As seen in the images, apart from chromatic distortions in the obtained images, the results are surprisingly good given the simplicity of the algorithm.

7. FUTURE WORK

There is much that can be done to improve this method. The sample output images present chromatic artifacts, namely disparate pixels that are too bright. This is an undesired result of repeated addition of the error estimate. One way of overcoming it would be setting a threshold for pixel values at each iteration step, not letting them exceed a brightness limit. This would be the easiest way to eliminate this problem. Another possibility would be to detect exceedingly bright pixels and apply a small windowed blurring filter to the neighboring pixels, smoothing out the large brightness value over a larger area.

Furthermore, the algorithm could be used to eliminate any kind of blurring kernels, not just Gaussian. Keeping in mind that this kernel was solely chosen for its simplicity, once a kernel loading feature is added to the application, no modifications would be needed in the processing sections of the code in order to deal with the new filter. This is therefore a framework upgrade that could be added without much effort.

A valuable addition to the algorithm would be kernel estimation. This would enable the application to be used on any kinds of blurred images. Lastly, since computing the blur kernel may lead to additional noise and artifacts, more complex de-ringing techniques could be added to improve the overall functionality.

8. REFERENCES

- [1] Jong-Ho Lee, Yo-Sung Ho, High-quality non-blind image deconvolution with adaptive regularization, *Journal of Visual Communication and Image Representation*, Volume 22, Issue 7, October 2011, Pages 653-663, ISSN 1047-3203, <http://dx.doi.org/10.1016/j.jvcir.2011.07.010>.
- [2] Renting Liu; Jiaya Jia, "Reducing boundary artifacts in image deconvolution," *Image Processing*, 2008. ICIP 2008. 15th IEEE International Conference on , vol., no., pp.505,508, 12-15 Oct. 2008, doi: 10.1109/ICIP.2008.4711802
- [3] Li Xu and Jiaya Jia. 2010. Two-phase kernel estimation for robust motion deblurring. In *Proceedings of the 11th European conference on Computer vision: Part I (ECCV'10)*, Kostas Daniilidis, Petros Maragos, and Nikos Paragios (Eds.). Springer-Verlag, Berlin, Heidelberg, 157-170.
- [4] Rob Fergus, Barun Singh, Aaron Hertzmann, Sam T. Roweis, and William T. Freeman. 2006. Removing camera shake from a single photograph. In *ACM SIGGRAPH 2006 Papers (SIGGRAPH '06)*. ACM, New York, NY, USA, 787-794. <http://doi.acm.org/10.1145/1179352.1141956>
- [5] Microsoft MFC: <http://msdn.microsoft.com/en-us/library/kkcb3t0w.aspx>
- [6] The OpenCV Library: <http://opencv.org/>

- [7] Mihai Zaharescu, Costin-Anton Boiangiu, "Image deblurring: challenges and solutions," in Proceedings of the 12th International Conference on Circuits, Systems, Electronics, Control & Signal Processing (CSECS '13), Budapest, Hungary, December 10-12, 2013, pp. 187-196
- [8] Costin Anton Boiangiu, Mihai Cristian Tanase, Radu Ioanitescu, "Handwritten Documents Text Line Segmentation based on Information Energy", International Journal Of Computers Communications & Control (IJCCC), ISSN 1841-9836, Vol 9, No 1, pp. 8-15, 2014
- [9] Costin-Anton Boiangiu, Mihai Cristian Tanase, Radu Ioanitescu, "Text Line Segmentation In Handwritten Documents Based On Dynamic Weights", The Proceedings of Journal ISOM Vol. 7 No. 2, pp. 247-254, December 2013
- [10] Costin-Anton Boiangiu, Alexandru Topliceanu, Ion Bucur - "Efficient Solutions for OCR Text Remote Correction in Content Conversion Systems", Journal of Control Engineering and Applied Informatics, ISSN 1454-8658, Volume 15, No. 1, pp 22-32, 2013
- [11] Costin-Anton Boiangiu, Alexandru Victor Ștefănescu, "Target validation and image calibration in scanning systems", in Proceedings of the 1st International Conference on Image Processing and Pattern Recognition (IPPR '13), Budapest, Hungary, December 10-12, 2013, pp. 72-78
- [12] Costin-Anton Boiangiu, Alexandru Victor Stefanescu, Daniel Rosner, Alexandra Olteanu, Anca Morar (2010). „Automatic Slanted Edge Target Validation in Large Scale Digitization Projects” (2010), Annals of DAAAM for 2010, Proceedings of the 21st International DAAAM Symposium, 20-23 October 2010, Zadar, Croatia, pp. 131-132, Vienna, Austria 2010
- [13] Andreea-Mihaela Pintilie, Costin-Anton Boiangiu, "Study Of Neurobiological Images Based On Ontologies Used In Super-Resolution Algorithms", The Proceedings of Journal ISOM Vol. 7 No. 2, pp. 300-308, December 2013
- [14] Mihai Zaharescu, Costin-Anton Boiangiu, "An Investigation of Image Deblurring Techniques", International Journal of Mathematical Models and Methods in Applied Sciences, Volume 8, 2014, pp. 75-83