A METHOD FOR REVERSING THE EFFECT OF DIGITAL IMAGE BLURRING: A SPACE DOMAIN INVESTIGATION

Gabriel Scarmana
School of Built Environment and Surveying, University of Southern Queensland, Australia

KEY WORDS: Image restoration, image blurring, image compression

ABSTRACT

Mobile mapping technology has transformed the way in which we capture and map our surroundings. The widespread use of mobile devices such as smart phones and drones has made data collection more efficient and accessible than ever before. However, the image quality of this data is often compromised due to the static or motion blurs resulting from the device being stationary or moving during the data collection process. This can lead to a loss of information, making the data less useful.
To address this issue, an inverse filtering or deblurring method based on the theory of least squares is examined. This method implements a deconvolution process in the space domain and can restore the original image with a high degree of accuracy if the model of the function or filter that blurred the image is known or can be established.
The accuracy and validity of the deblurring results are presented in terms of the root mean square error (RMSE) of the differences of pixel intensity values between the original and de-blurred images. In tests using grey-scale aerial images of varying entropies and different types of blurring, the RMSE value never exceeded +/- 5 pixel intensity values. This discrepancy is due to the rounding of pixel values resulting from image operations.
The deblurring method presented in this work is an adaptation and extension of a previously described process, tailored specifically for filtering and restoring images - particularly aerial imagery - affected by static and motion blur. This process could also be applied in image compression processes and techniques of transmission over digital links, where blurring filters can suppress noise and increase the dependence between neighbouring pixel values, thereby improving the compression ratio (CR).

1. INTRODUCTION

In general, deblurring an image is carried out by deconvolution which is a mathematical technique that can be used to undo the effect of a known blurring function (i.e., a kernel). The main idea behind deconvolution is to estimate the original, unobserved image, from the observed, blurred image by removing the effect of the applied blurring filter (Prost, 2019).

Hence, if the model of the blurring function or kernel that corrupted an image can be modelled, the original image can be restored with a high level of accuracy. Deblurring a digital image with knowledge of the kernel that blurred the original image is a well-posed problem that can be approached using several methods. For example:

Inverse Filtering: Inverse filtering is a simple method for deblurring a digital image with knowledge of the blurring kernel. It involves applying an inverse filter to the blurred image to restore its original sharpness. The inverse filter is the inverse of the point spread function (PSF) that caused the blur.

The idea behind this method is that if the image was blurred by a known PSF, then dividing the blurred image by the PSF should remove the blur. However, this method is sensitive to the estimation of the blurring function and can result in the amplification of noise in the image. Additionally, the inverse filter may not exist, or it may not be stable, which can result in over-sharpening or the introduction of unrealistic details into the image (Gonzalez and Woods, 2017).

Wiener Filtering: Wiener filtering is an extension of inverse filtering that takes into account the presence of noise. It uses a statistical model of the noise and the PSF to restore the image. The Wiener filter minimizes the mean square error between the original image and the de-blurred image, subject to a constraint

on the noise level. Wiener filtering is a powerful method for deblurring images and is often used in image processing applications. This method is more robust than inverse filtering and can produce a de-blurred image that is free of unrealistic details and noise amplification (Russ and Neal, 2016).

Constrained Least Squares (CLS) Filtering: CLS filtering is a method for deblurring digital images that uses a regularization term to avoid over-sharpening the image. The regularization term is used to constrain the solution so that the de-blurred image is smooth and does not contain any unrealistic details. CLS filtering is a more robust method for deblurring images compared to inverse filtering and is less sensitive to the estimation of the blurring function. This method can produce a de-blurred image that is as close as possible to the original sharp image while avoiding over-sharpening and unrealistic details (Reddy et al. 2015).

Richardson-Lucy Deconvolution: Richardson-Lucy (RL) deconvolution is a widely used method for deblurring images with known PSFs. This method is based on the principle of maximum likelihood estimation and iteratively updates the deblurred image until a solution that satisfies the constraints is found. RL deconvolution is a fast and effective method for deblurring images, but it may amplify the noise in the image and produce unrealistic details if the iteration process is not stopped at the appropriate time (Solomon and Breckon, 2010).

In recent years, deep learning approaches have also been applied to the problem of deblurring digital images. These methods use convolutional neural networks (CNNs) trained on

large datasets of blurred and de-blurred images to learn the mapping from blurred images to de-blurred images.
Deep learning approaches have shown promising results for deblurring digital images in a variety of different scenarios and are able to handle complex blurring functions. However, deep learning approaches require large amounts of training data and can be computationally expensive (Koh et al, 2021; Albluwi et al. 2018).

2. KNOWLEDGE OF THE BLURRING KERNEL

It is known that blurring can be manually achieved by applying operations to groups of pixels. At the core of this operation is a convolution mask or kernel. To calculate one pixel using a convolution mask of size m x n, m * n multiplications, m * n - 1 additions, and one division are needed. For example, to perform a 3 x 3 convolution on a 1024 x 1024 colour image, 27 million multiplications, 24 million additions, and 3 million divisions are needed.

Larger convolutions, such as using kernels of 5x5 or 8x8 on bigger images, require even more computation. For the case of blurring or smoothing images the following filters are frequently applied: (1) Mean or average filter (2) Weighted average filter and (3) Gaussian filter. The reader is referred to Russ et al. (2016) for comprehensive details of these conventional image filters.

In the context of this work, during blurring, the centre of the convolution kernel passes over each pixel in the image. The process multiplies each number in the kernel by the pixel intensity value directly underneath it (Madhuri et al, 2014). This should result in as many products as there are numbers in the kernel (per pixel). This process is condensed in Equations (1) and (2) below for the simplistic case of a 3x3 image represented by 9 pixels (i.e., p1...p9). Equation 2 only defines the filtered pixel C5.

$$\begin{vmatrix} p1 & p2 & p3 \\ p4 & p5 & p6 \\ p7 & p8 & p9 \end{vmatrix} * \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = \begin{vmatrix} C1 & C2 & C3 \\ C4 & C5 & C6 \\ C7 & C8 & C9 \end{vmatrix} \quad (1)$$

$$[C5] = (p1*a)+(p2*b)+(p3*c)+(p4*d)+ + (p5*e)+(p6*f)+(p7*g)+(p8*h)+(p9*i) \quad (2)$$

The above equations show that the element at C5, which is the central element of the resulting filtered image, is a weighted combination of all the entries in the image matrix. The weights are determined by the values in the kernel. Similarly, the other elements in the filtered image are calculated by positioning the centre of the kernel on each boundary point of the image and computing a weighted sum. The output values of a given pixel in the filtered image are calculated by multiplying each kernel value by the corresponding input image pixel values.

It's worth noting that even though the kernel may overlap with several dissimilar pixels or in some cases, no pixels at all (i.e., edges), the only pixel that it ultimately affects is the source pixel underneath the centre element of the kernel. The numbers inside the kernel are what influence the overall result of the filtering. In other words, the kernel (or more specifically, the values within the kernel) determines how to transform the pixels from the original image into the pixels of the processed image. As demonstrated in the next section the kernel used to blur an image is conveniently used to reverse the blurring effect.

3. REVERSING THE BLURRING EFFECT

The method considered here employs a linear algebraic technique and is illustrated by using a 3x3 kernel which is used for blurring a given image P (with pixel intensities of 256, 100, 80, etc.). The outcome of this blurring process is displayed in Figure 1(c). That is, image C.

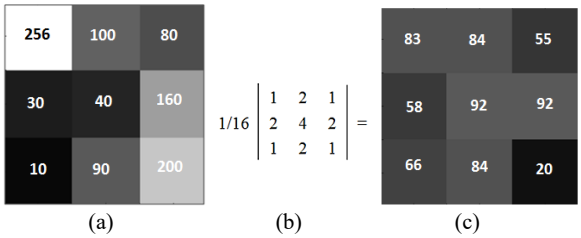

Figure 1. In line with equations 1 and 2, the original image P in (a) is multiplied (or convoluted) using the normalized kernel in (b). The result of the convolution is shown in (c) that is, image C. The outside edges were processes as having 0 value.

The pixels of image C (83, 84, 55...etc.) are calculated based on Equations 1 and 2. The complete set of reduced equations (observation equations) for this example is given below.

$83 = (4p_1+2p_2+2p_4+p_5)/16$
$84 = (2p_1+4p_2+2p_3+p_4+2p_5+p_6)/16$
$55 = (2p_2+4p_3+p_5+2p_6)/16$
$58 = (2p_1+p_2+4p_4+2p_5+2p_7+p_8)/16$
$92 = (p_1+2p_2+p_3+2p_4+4p_5+2p_6+p_7+2p_8+p_9)/16$
$92 = (p_2+2p_3+2p_5+4p_6+p_8+2p_9)/16$
$66 = (p_4+2p_5+p_6+2p_7+4p_8+2p_9)/16$
$84 = (p_5+2p_6+2p_8+4p_9)/16$
$20 = (2p_4+p_5+4p_7+2p_8)/16$

The observation equations can also be utilized in reverse to restore the pixels of the original image in Figure 1(a) using a least squares method. Therefore, in matrix form

$$[A] = 16 * \begin{bmatrix} 4 & 2 & 0 & 2 & 1 & 0 & 0 & 0 & 0 \\ 2 & 4 & 2 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 2 & 4 & 0 & 1 & 2 & 0 & 0 & 0 \\ 2 & 1 & 0 & 4 & 2 & 0 & 2 & 1 & 0 \\ 1 & 2 & 1 & 2 & 4 & 2 & 1 & 2 & 1 \\ 0 & 1 & 2 & 0 & 2 & 4 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 & 2 & 1 & 2 & 4 & 2 \\ 0 & 0 & 0 & 0 & 1 & 2 & 0 & 2 & 4 \\ 0 & 0 & 0 & 2 & 1 & 0 & 4 & 2 & 0 \end{bmatrix} \quad \text{and} \quad [C] = \begin{bmatrix} 83 \\ 84 \\ 55 \\ 58 \\ 92 \\ 92 \\ 66 \\ 84 \\ 20 \end{bmatrix}$$

The solution of the above system has [C] representing the grey-scale pixels of the filtered image, [p] pertaining to the required grey levels of the original image, and [A] being the matrix of coefficients. As demonstrated below, the original image is fully and accurately recovered. That is,

255	100	79
29	40	161
11	90	200

$$[p] = [A^T A]^{-1} * [A^T C] * 16$$

The deblurring process outlined in this simplistic example was carried out using MATLAB (Gonzalez et al., 2020) and then applied to grey-scale images of any dimension. The following section includes an example of the various tests undertaken with images of different levels of high/low frequency details and inherent entropies. In summary the following steps illustrate the basic flow of this process:

1) Input a grey scale image.
2) Input a kernel (filter function).
3) Convolute the input image using the kernel.
4) Generate a system of equations for each convolved pixel expression based on equation 2.
5) Solve the system of equations in reverse to recreate a reconstructed version of the original image.
6) Output/display the reconstructed grey scale image.
7) Compare the original image with its reconstructed counterpart by subtracting said original image from the reconstructed.
8) Calculate the RMSE (Root Mean Square Error) of the differences of pixel intensity values between the two images.
9) Output the RMSE value and evaluate the level of accuracy achieved.

4. TESTS

Following the concepts described in the previous sections, the image "urban.BMP" shown in Figure 2a (1000^2, 1 MB) was compressed and saved by way of the lossless .PNG (Portable Networks Graphics) protocol (McAndrew, 2016; Scarmana, 2014), resulting in a storage capacity of 0.5 MB. This compressed image "urban.PNG" was then divided into 40 blocks of 50^2 pixels each. This method of block division was chosen to make the computations faster (i.e., solving the system of observation equations).

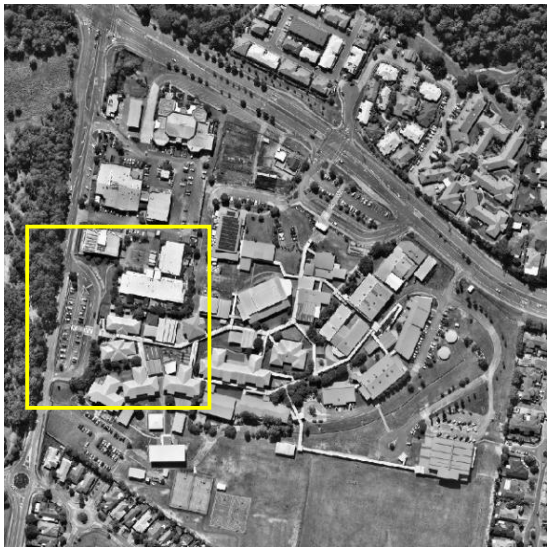

Figure 2a. The original image referred to as urban.bmp. Below is the kernel used to obtain the blur image in Figure 2b.

$$\frac{1}{273}\begin{vmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{vmatrix}$$

Kernel 1

Each block of 50^2 pixels was filtered using Kernel 2 given in Figure 2a, resulting in the image shown in Figure 2b (only a section is displayed to demonstrate the effect of blurring). It's worth mentioning that the entire blurred image only required 0.22 MB for storage, which represents a CR of 4.5 units when compared to the original "urban.BMP".

Figure 2b. Image 2a was filtered via an approximate and normalised Gaussian filter (Kernel 1). Only a section is shown for visual purposes

The RMSE of the difference in pixel intensity values between "urban.PNG" and the reconstructed image in Figure 2c was +/- 4, with a minimum and maximum difference of -6 and +8 respectively.

Figure 2c. A section of the reconstructed image by reversing the effect of Kernel 1.

The system of equations used to solve for these 2500 pixels can be replicated exactly in the same way for each of the remaining 40 blocks, with the only variation being the Cn coefficients which of course would have different values for each block.

In another test, image 2a was filtered using a different kernel. In this case the 5x5 kernel (i.e., Kernel 2) simulated a horizontal motion blurring effect as shown in Figure 2d. The restoration is illustrated in Figure 2e. In this instance, The RMSE of the difference in pixel intensity values between "urban.PNG" and the reconstructed image was +/- 5, with a minimum and maximum difference of -5 and +9 respectively.

Figure 2d. The original image 2a blurred using Kernel 2, which simulates a horizontal motion blurring effect.

$$1/5 \begin{vmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{vmatrix}$$

Kernel 2

Figure 2e. The reconstructed image by reversing the motion blurring effect of Kernel 2.

5. HOW MUCH BLUR?

The amount of blur that can be applied to an image before it becomes irrecoverable depends on several factors, including the resolution of the original image, the level of detail present in the image, and the amount of blur applied. In general, the more significant the level of detail in an image, the less blur it can tolerate before the loss of information becomes apparent. Images with high levels of detail, such as landscapes or portraits, are more sensitive to blur than images with fewer details, such as solid colour backgrounds or abstract art.

The resolution of the image is also an essential factor in determining the amount of blur that can be applied. Images with high resolutions contain more pixels, allowing for more detail to be captured. Such images can withstand higher levels of blur before becoming irrecoverable. On the other hand, images with lower resolutions are already prone to losing detail and can be irrecoverable even with minimal blurring (Toshiyuki et al., 2022).

The amount of blur applied is also crucial in determining the recoverability of an image. Small amounts of blur may not significantly affect the image's recoverability, while excessive blurring can lead to complete loss of detail. Therefore, it's essential to establish a balance between the amount of blur applied and the amount of detail lost.

For the cases studied in this contribution and using the same kernel given in Figure 2a, it was found that applying more than twice the same kernel in succession would result in unsatisfactory restoration results (i.e., added unwanted noise artefacts).

6. APPLICATIONS

This section gives some examples of how and where image deblurring can and has been used in mobile mapping technology tasks:

Mobile mapping services utilise street view images captured by vehicles to provide users with a complete view of a location. However, these images can often be blurry due to motion blur caused by the vehicle's movement. By applying image deblurring algorithms, the clarity of these images can be improved.

Mobile-based augmented reality (AR) apps overlay digital information onto the real world using the camera of a smart phone or tablet. However, motion blur or shaking can compromise the quality of the camera's image, reducing the accuracy of the AR overlay. Image deblurring can remove motion blur, enhancing the accuracy of the AR overlay

Mobile-based 3D mapping apps use the camera of a mobile device to capture images of a location from different angles, creating a 3D model of the environment. But if the camera is moving or shaking, the resulting images can be blurry and distorted, leading to inaccurate 3D models. Image deblurring can improve the quality of these images, ensuring the resulting 3D model is accurate.

To investigate the impact of motion blur on images captured during a mobile mapping task from an entry-level drone, a controlled experiment was conducted on a coastal area. The drone (DJI Min 3, 700p and 30 fps) was flown at an average altitude of 80 metres above the land, and the video camera

mounted on it captured images accordingly at a constant speed (i.e., 22 Kph).

Despite the drone being set on a horizontal path, sudden changes of perspective views along the selected path caused motion blur, resulting in a mostly horizontal motion blur in many of the captured images. To mitigate the impact of motion blur on the images, the least square deblurring algorithm was used in this experiment, and it was iteratively applied using Kernel 2 as shown in Figure 2a. The results of this controlled experiment are summarized as follows:

Number of blurry images extracted from the video: 500.
Percentage of images requiring deblurring: 80%
Deblurring algorithm used: Least squares deblurring algorithm.
Average processing time per image: 10 seconds.
Success rate of deblurring algorithm: 80%.
No. of images requiring manual touch-up after deblurring: 53.

The percentage of images requiring deblurring highlights the impact of motion blur on images captured during a mobile mapping task from a drone. The least squares deblurring process was able to restore the sharp image from the observed blurry image in most cases, resulting in a success rate of 80%. Visual results from this test were like those indicated in Figure 2e.

The average processing time of 10 seconds per image is reasonable. However, the relatively high number of images requiring manual touch-up after deblurring suggests that the algorithm may need to be refined or combined with other approaches to improve its accuracy and/or the selection of a more compatible kernel. A set of the de-blurred images obtained in this controlled test are available from the author as required

7. CONCLUSIONS AND DISCUSSION

In this study, the data representing original grey-scale images were first acquired. The pixel data was then processed using a blurring function that emulated a Gaussian filter with the objective of blurring said images. The original images were then restored by way of a least squares mathematical model.

This reconstruction was possible because the lost details were still implicitly present in the filtered image. The "hidden" information could be restored by knowing the details of the original filtering function or kernel. The original image could not be recovered exactly because of the various logical errors associated rounding pixel values to the nearest integer.

Tests are still required to determine the effect of other types of scaled filters (i.e., symmetric and/or non-symmetric filters) using the principles described in this work. By the same token, more research may determine whether small kernels could be applied perhaps more than once to produce a similar but not identical reversal effect as compared to a single pass with a large filtering kernel. Also, tests may be required to ascertain the use of a separable convolution process which would in principle speed the whole blurring/deblurring process.

The proposed method for deblurring an image involves using prior information to achieve more accurate results. Reversing a low-pass filter by applying sharpening or high-pass kernels to a blurred image does not yield satisfactory outcomes. Sharpening operations only highlight features of an image, but do not add any new or original information. Additionally, blurring and sharpening are not true inverses, meaning that applying a blur filter and then trying to restore it with a high-pass filter will not reconstruct the original image.

As a final note, the proposed technique offers a prospect for encoding image data into a filtered form, with the ability to modify the filtering effect according to different blurring kernels. The image can be restored and accurately recovered with knowledge of the initial filtering function or kernel, which can be readily changed and re-scaled to suit different requirements. A suggestion regarding the possibility of using this contribution as an application related to image compression was also presented.

REFERENCES

Albluwi F., Krylov V.A. and Dahyot R. 2018: Image Deblurring and Super-Resolution Using Deep Convolutional Neural Networks, 2018 IEEE 28th International Workshop on MLSP, Aalborg, Denmark, pp. 1-6.

Gonzalez R. C. and Woods R. E. 2017: Digital Image Processing, 4th Edition, Prentice Hall. 1192 pages.

Gonzalez R. C., Woods R. E. and Eddins S.L. 2020: Digital Image Processing Using MATLAB 3rd edition, Gatesmark, ISBN-10 :0982085419. 609 pages.

Koh J., Lee J. and Yoon, S. 2021: Single image deblurring with neural networks: A comparative survey. Computer Vision and Image Understanding. 203. 103134. 10.1016/2020.103134.

McAndrew A. 2016: A Computational Introduction to Digital Image Processing. ISBN 9780367783334. Chapman & Hall. 560 pages.

Prost G. L. 1997: Remote Sensing for Geologists: A Guide to Image Interpretation. CRC Press. ISBN:9782884491013. 326 p.

Russ J. C. and Neal B. F. 2016: The Image Processing Handbook. ISBN 9781138747494, CRC Press. 1056 pages.

Reddy M. and Shankar H. 2015: Textbook of Digital Image Processing. BS Publisher. ISBN: 9789385433368. 312 pages.

Scarmana G. 2014: Lossless data compression of grid-based digital elevation models: A PNG image format evaluation. ISPRS. Annals of Photogrammetry and Remote Sensing, isprsannals-II-5-313.

Showengerdt R. A. 2007: Remote Sensing: Models and Methods for Image Processing. Elsevier. 509 pages.

Solomon C. and Breckon T. 2010: Fundamentals of Digital Image Processing: A Practical Approach with MATLAB. Wiley and Blackwell. ISBN: 9780470844731. 344 pages.

Toshiyuki H. and Takashi T. 2022: Estimation and Sharpening of Blur in Degraded Images Captured by a Camera on a Moving Object. Sensors 2022, 22(4), 1635; https://doi.org/ 10.3390/ s22041635.

Xin Li. 2012: Image Restoration: Fundamentals and Advances. Digital Imaging and Computer Vision. 1st Edition. CRC Press. Taylor and Francis. 378 pages.