An Integrated Algorithm For Motion Deblurring and Novel Blurring

Ian Tagge
The University of Portland
Portland, OR, 97203
itagge@up.edu

Abstract

Moving objects or camera motion can cause motion blur in conventional single-exposure photography. Raskar, Agrawal and Tumblin at MERL have developed an algorithm that compensates for blurring effects due to relative motion on an image. This project extends their work by building a user interface in order that their work may be more easily applied. The GUI takes the blurred image as input. Interactively, the user specifies parameters for the deblurring function for each input image. The results are displayed in the GUI as the constraints are adjusted. Users may also specify parameters to reblur the image and create novel effects that are not producible by normal object motion or camera motion.

Figure 1. Blurred Frequency Grating; Restored Frequency Grating; Coded Exposure Taxi; Deblurred Taxi; Taxi modified with Ghost Blur filter.

Introduction to Coded Exposure Motion Deblurring

Coded Exposure Motion Deblurring (CEMD) requires a special method of image acquisition in which a digital SLR is fitted with an LCD shutter that is digitally controlled by an on-board microcontroller. This LCD shutter flutters open and closed according to a pre-determined pattern over the entire exposure period rather than one long exposure. A single exposure effectively defines a temporal box filter that degrades the image and destroys high-frequency information. Since this high-frequency detail is destroyed, image restoration via deconvolution is an ill-posed problem.

Fluttering the camera's shutter during exposure creates a broadband filter that preserves high-frequency details of a moving object. This preservation makes image restoration by way of deconvolution a well-posed problem and the sequence

Figure 2. Digital SLR fitted with LCD Shutter.

used for fluttering can be exploited to accurately deblur a moving object in a photograph. Fluttering the shutter in this fashion determines the Point Spread Function (PSF). Knowledge of the PSF is essential in accurate image restoration. Without a known PSF some form of a PSF estimation algorithm must be employed and may not produce acceptably accurate results.

The 1-D PSF vector is used to find the matrix A (see Appendix) and the least-squares solution to the linear system

$$A * F = F\_blurred$$

is found to recover a close approximation

of the original image F from the blurred image F_blurred.

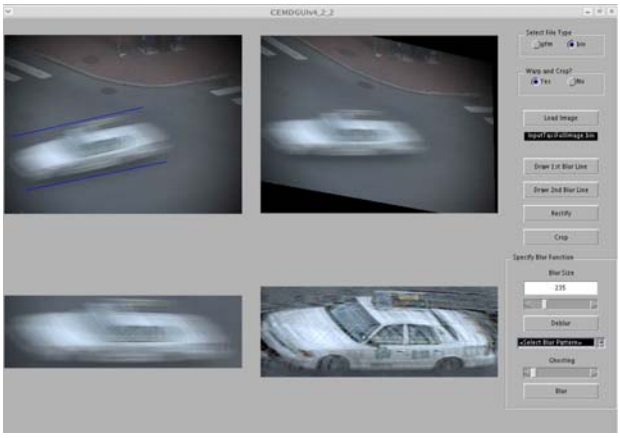

Figure 3. Full restoration of Taxi, including preprocessing; note the blue lines on the top left axes illustrating user-specified motion blur indication lines.

Deblurring with the GUI

Step 1:

The MATLAB GUI takes the blurred image as input – this file should be either of '.bin' or '.pfm' type. The original image is selected and displayed in the GUI.

Step 2:

Agrawal's algorithm assumes horizontal motion blur, so if the actual blur in the image is not horizontal the user must specify two parallel lines on the original image that describe the direction of motion. Lines are drawn on the image illustrating the user's selection.

Step 3:

An affine transformation is applied to the image according to the user-defined lines, which warps the image and causes the motion to be horizontal.

Step 4:

The user specifies a bounding box around the key object in the image that needs to be deblurred; this object is then cropped out of the original image and displayed on a third set of axes within the GUI.

Step 5:

Finally, the user must specify the blur size in pixels. Once all of the preprocessing has taken place the cropped and transformed image can be deblurred and displayed on the fourth set of axes. All of the intermediate steps (save specifying blur size) may be omitted if the input image

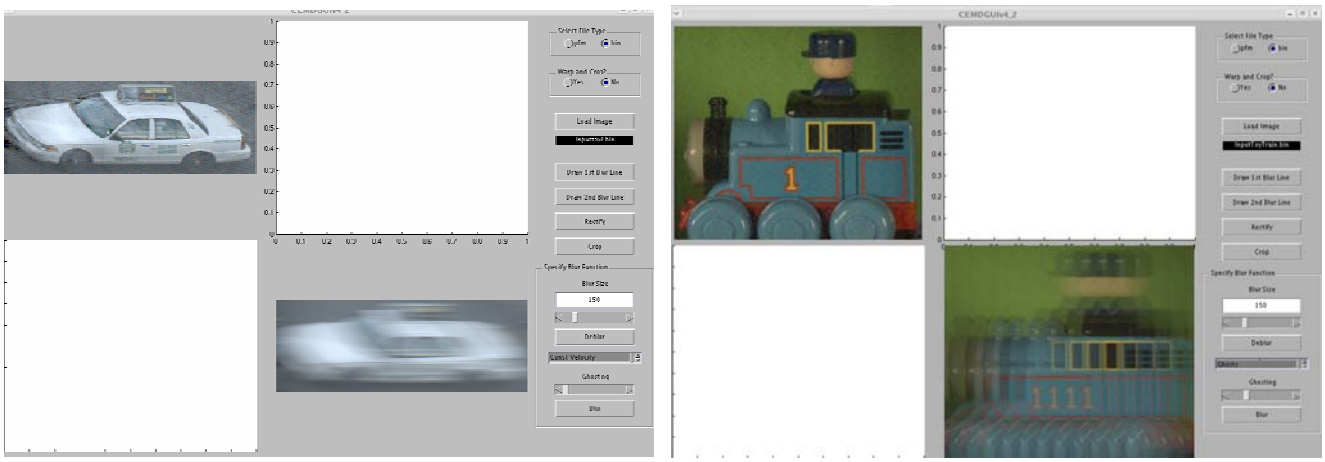

Figure 4. Taxi with Constant Velocity Blur; Train with 4 ghosts.

is already properly cropped and the direction of motion is horizontal.

Novel Blurring with Defined PSF's

Agrawal's algorithm can, in a sense, be applied in reverse to create novel blurring effects that cannot be generated by typical subject or camera motion. His algorithm is easily modified to produce novel blurring effects rather than deblurring. Effects such as a variable number of ghosts, affected constant acceleration in either horizontal direction, constant velocity or some other novel coding are easily generated by simply using different PSF's in the linear system

$$F\_blurred = A * F,$$

where F is the original image and A is a blurring matrix defined by the PSF.

When the algorithm is naïvely applied in reverse some clipping occurs. This is because when the object is being blurred it actually grows in size as it is smeared across the image by convolution. To accommodate this growth the images are padded by k/2, where k is the effective blur size defined by the user within in the GUI, on either side of the image. These synthesized parts are simply repetitions of

the first and last columns of the image, respectively, thus the padding tends to look rather harsh and certainly not aesthetically pleasing. Since the whole basis of creating novel blurs is aesthetics an averaging mask filter is applied to the prosthetic appendages before concatenating them to the image. This softens the appearance, and, when combined with the final blurring alteration, produces a very pleasing image with minimal artifacts; often no visible artifacts are present at all.

Figure 5. Image of a Toy Train expanded for a total blur of 118 pixels.

The GUI for Novel Blurring

Step 1:

Either the final image from the deblurring process or a user-specified image is taken as input and displayed on the first set of axes in the GUI.

Step 2:

The user specifies the type of blur, the desired blur size, and any other applicable parameters.

Step 3:

The image is blurred according to the user-defined parameters and is displayed in the fourth set of axes in the GUI. This final image may be saved in the same format of the original image if the user desires. Results may be saved at the end of either blurring or deblurring.

Figure 6. Composite displaying start to finish restoration and modification of a toy train.

Discussion and Ideas for Future Research

Agrawal's CEMD algorithm clearly has many applications, discussion of which is not within the scope of this paper, and is easily integrated in a MATLAB GUI for user-friendly interaction and use.

The methods employed in CEMD are also effective in creating novel blurring effects on images. Understanding PSF's is key when attempting to generate new effects, but Agrawal's method is fantastically simple, yet effective, making such image modification and PSF development extremely easy.

It is conceivable that in the future the need for user interaction during preprocessing may be entirely eliminated. One potential method would employ gradient analysis in an algorithm that accurately estimates the direction of motion, performs the affine transformation, defines a suitable bounding box and crops the image automatically. It is even likely that the blur size can be estimated as well, thus requiring no user interaction beyond introducing the original image.

Also, only a handful of manually specified PSF's have been generated and tested in producing novel blurring effects. Many other desirable effects may be realizable by combining these methods with others such as edge extraction or any number of image processing techniques commonly used today – such extensions

Figure 7. Deblurred Train; Train with blur lines (constant acceleration); Train with two ghosts – note the two artifacts (one on either side of image) from image expansion.

and applications are limited only by one's imagination.

At this time only one of the blurring functions generates PSF's on the fly, while the others use hard-coded sequences. In the future more complicated algorithms may be written that dynamically produce PSF's defined by certain properties of the specific image being processed. Also, the only blurring effects currently available produce horizontal motion effects. While this provides a solid base from which one can work, these blurring functions may ultimately be made multi-directional, thus facilitating the creation of far more interesting results.

The method of blurring displayed here also has some limitations. Theoretically, novel blurring using a known PSF should be a completely reversible process. However, as discussed above, the image is modified for aesthetic purposes before being blurred. If this modification is not performed the process is perfectly reversible. This is likely an easily solved problem, but as of yet has not been thoroughly researched.

Finally, the MATLAB GUI may be compiled to a Windows- or Linux-based GUI for added portability. Adding support for more than the two included file types may also help to increase portability.

References:

RASKAR, R., AGRAWAL, A., AND TUMBLIN,

J. 2006. Coded Exposure Photography:

Motion Deblurring Using Fluttered Shutter.

In ACM SIGGRAPH.

Appendix:

```
function [A] = motionblurmatrix(CodeSeq,W)

E = CodeSeq(:);
E = E/sum(E(:));
k = size(E,1);

tt = [E(end:-1:1);zeros(W-1,1)];
% Zero-pad sequence
tt = sparse(tt);
% Create Circulant Matrix
A = gallery('circul',tt);
% Keep first W columns out of (W+k-1) cols
A = A(1:W,:);
```