Abstract—Previously, Raskar proposed a method of transforming the frequency response of a traditional shutter through the use of coded exposure. This technique shaped the frequency spectrum to a broadband signal that preserved higher frequency information and thus made the deconvolution process more well posed. This paper presents a generalization of the motion blur kernel he used. Instead of requiring a fixed deconvolution kernel determined by the binary code used to capture the image, this kernel can be created corresponding to a blur of any size in any direction.

Simulation results show that this 2-D filter exhibits traits predicted by Raskar, and application of this kernel to deconvolve real images has shown that it can produce sharper but noisier results.

I. INTRODUCTION

The purpose of this paper is to build off the framework presented in [1]. Raskar’s original paper was intended as a proof of concept, showing that new methods of image formation can increase the amount information captured in a single photograph. His technique shaped the frequency spectrum to a broadband signal that preserved higher frequency information and thus made the deconvolution process more well posed. This paper presents a generalization of the motion blur kernel he used. Instead of requiring a fixed deconvolution kernel determined by the binary code used to capture the image, this kernel can be created corresponding to a blur of any size in any direction.

II. BACKGROUND

Why does one take a picture? There can be many reasons, but in essence, the objective is to record a freeze frame of time in order to capture a subjective or objective reality. The classic camera model is based on this idea. Over the duration of the image capture process, the objects in the scene are assumed to be static. In reality, objects are constantly moving and changing, which implies that an ideal camera needs to capture an image instantaneously. In other words, the shutter needs to be open for only an instant. In practice, the period of the shutter, \( T \), is proportional to the Signal to Noise Ratio, SNR. The longer the shutter is open, the better the SNR. Therefore, an inherent trade off exists between SNR and satisfying the 0th order model of motion assumption.

The effect of leaving the shutter open longer is motion blur. Motion blur can be modeled as

\[
\begin{align*}
\mathbf{r}(x, y) &= (s \ast h)(x, y) \\
\end{align*}
\]

where \( s(x, y) \) is the moving object, \( h(x, y) \) is a continuous motion blur kernel, and \( \ast \) denotes the two-dimensional convolution. Often noise is introduced during image formation, thus a more accurate model is

\[
\mathbf{r}(x, y) = (s \ast h)(x, y) + \eta(x, y)
\]

Given that we have acquired \( r(x, y) \), a common question is if we can recover \( s(x, y) \) from \( r(x, y) \)?

Ramesh Raskar investigated this question in the journal paper, “Coded Exposure Photography: Motion Deblurring using Flutter Shutter” [1]. In general, deconvolution of a motion blurred image is hard. The problem lies in the shape of the traditional shutter window, see Figure 1. The rectangular shape of this temporal filter leads to a sinc frequency response. A sinc frequency response is problematic due to its infinite number of zero crossings. During convolution, frequency information of the original signal is lost at the zero crossings. The position of zero crossings relative to the origin is inversely proportional to the shutter period, \( T \). For this reason, as \( T \to \infty \) all of the frequency content of \( r(x, y) \) is lost to the zeros. Visually this is interpreted to mean a complete smearing of the image in the direction of motion. Since every pixel in a line is smeared with every other pixel of that line, there is no way to recover the original image. On the other hand, as \( T \to 0 \) the filter \( h(x, y) \) approaches the delta function, \( \delta(x, y) \). The delta function is the identity map for convolution; thus all of the information of \( s(x, y) \) is persevered in the motion blur convolution. From this perspective, it can be seen that the zero crossings of \( h \) cause deconvolution to be ill-posed.

![Fig. 1. Application of a coded sequence to a traditional shutter to obtain a coded shutter](image)

Raskar et al. proposed a means of manipulating the frequency response of the motion blur kernel to allow the deconvolution problem to become well-posed, by removing the zeros. The change is simple. Instead of leaving the shutter open continuously for \( T \) seconds, the shutter is “fluttered” between open and close. This is similar to multiplying a traditional shutter by a sequence of 1’s and 0’s, where 1’s
correspond to open and 0’s correspond to closed. With the deconvolution well-posed, linear inversion techniques are then able to recover the original image. Thus, image formation using coded exposure can account for 1st order motion. This is a significant change to a foundational assumption of photography.

A. Code Selection

The key difference from a traditional image and a coded exposure image is the embedded code. Given a code of length \( m \), where each bit can be a 1 or 0, the number of possible codes is \( 2^m \). In order to determine the optimal code, [1] used two criteria.

1) Maximize[Min(Frequency Response)]
2) Minimize[Varience(Frequency Response)]

The first criteria corresponds to the erasure of zeros from the frequency response. The second criteria corresponds to the desire for robustness, in that a small mis-estimation will not have amplified effects in the output. The code implemented in [1] had a constant length \( m = 52 \) and equals

101000111000010100001100111101011001010011

This code was chosen from a set of \( 3 \times 10^6 \) candidate codes, based on the provided criteria. This code is used to capture the real data used later.

B. Implementation

Only two components need to be changed in a typical camera system to implement coded exposure. First, images needed to be taken with a coded exposure. The implementation in [1] and for the real data in this paper used an 8 megapixel Canon Pro 1 with an attached PIC micro-controller which controlled the camera and a ferro-electric shutter placed over the lens. Second, the recorded image must be deconvolved. Because of the assumption of a 1st order model of motion, Raskar implemented all deconvolution through 1-D linear inversion. For motion at an angle, the blurred image is rotated and resized to fit the binary blurring kernel. The general configuration can be seen in Figure 2.

![Diagram of code selection](image_url)

The current implementation requires multiple uses of MATLAB’s imrotate and imresize commands. These commands require interpolation, thereby introducing additional error to the system. By cutting out the rotation and resizing the quality of the output can be improved and the mathematical link connecting the recorded and recovered images can be more easily seen.

III. Related Work

A. Coded Aperture

Coded image formation is not new. X-ray imaging, which is unable to focus images with a lens, has been using coded image formation since the 80's [2]. Instead of a coded exposure, X-ray imaging uses a technique called coded aperture. Ideally, X-ray imaging would like to utilize a pinhole camera to perfectly image incoming rays, similar to convolution with a delta function. But, in practice, a pinhole camera does not allow enough light through the aperture onto the sensors. This means that the SNR will be low. Coded aperture is a systematic arrangement of “pinhole” blocks that act similar to shifted pinholes. If the pattern is chosen correctly, the shape of small holes is arbitrary. The end result is a complex checkerboard-like pattern, which allows significantly more light through than a single pinhole. The advantage being a larger SNR. The disadvantage of coded aperture is that the input from each “pinhole” creates a shifted replica of the original image on the output. This superposition needs to be reversed in order to attain a meaningful image. But, given the proper coded aperture this deconvolution problem is well posed.

Coded exposure is analogous to this technique. Coded exposure is a manipulation of the temporal filter, while coded aperture is a manipulation of the spatial filter. In both cases the desired filter is a delta function, but due to practical SNR constraints, a complex code of pseudo-delta functions are arranged to increase the SNR and allow for simple deconvolution of the complex filter. The main difference is the domain of the filter being manipulated, temporal or spatial.

B. Short-Exposure Imaging

Another technique very close to coded exposure is short-exposure imaging. Coded exposure can be thought of as the superposition of many short exposure images. In this way, short-exposure imaging is similar to pinhole imaging, with the same issue of low SNR. New techniques allow special cameras to improve their SNR, but this often requires very ideal settings and expensive equipment. Another inherent disadvantage of short-exposure imaging is storage requirement. To replicate coded exposure, an average of 26 images would need to be taken, stored, and processed.

C. Smarter Cameras

In general, coded exposure is part of a general trend towards smarter cameras. Classical film cameras require numerous choices on the part of the user: focal length, flash, shutter speed, storage medium, etc. Let the set of all parameters that can be manipulated define a vector specifying one camera choice. Next consider the span of all such vectors ranging over all possible parameter values. We can imagine this as a multi-dimensional box defining the scope of classical cameras [3]. Through the use of computational photography techniques such as coded exposure, “smart” cameras break outside of the box.