Image Deblurring Techniques

Vincent DeVito
Computer Systems Lab
2009-2010

Abstract

In the world of photography and machine vision, blurry images can spell disaster. They can ruin an otherwise perfect photo or make it impossible for a computer to recognize the image or certain components of it for processing. The best way to counter this without taking another, clearer picture is to utilize deconvolution techniques to remove as much blur as possible. My plan is to first design a program that takes an image, blurs it using a known blur kernel, then deblurs it to reproduce the original image. After that I will attempt to create a program to determine the blur kernel of a naturally blurred image.

Background

In my research I have found various methods of blind and non-blind image deconvolution. One paper discussed comparing a blurry, correct intensity image with a sharp, noisy image to produce a proper, deblurred output image with few artifacts. Another paper discussed an algorithm they developed to estimate the blur kernel and use that to deblur the image from just a single photograph. Various deconvolution algorithms already exist, and it is the other component, the blur kernel, that requires further research. The more accurately the blur kernel can be estimated, the more accurate and clear the output image will be.

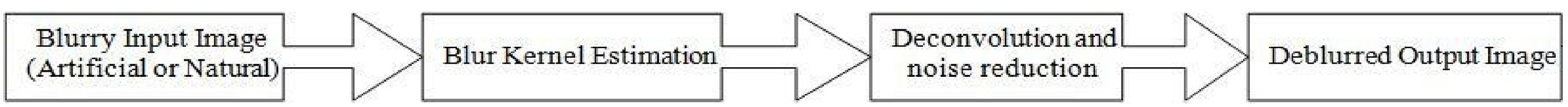

Figure 1. A graphical representation of the layout of my project design

Methods

The convolution and deconvolution process heavily rely upon the Fourier transform (Figures 3 & 4). The 2D Fourier transform converts images from the spatial domain to the frequency domain with complex values. This makes convolution and deconvolution simple, since they are just a matter of point multiplication or division, respectively, of the transformed image's pixel values with the transformed blur kernel's pixel values. From there, the inverse Fourier Transform (Figure 5) converts the convolution/deconvoluted image back to the spatial domain.

$$F(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n) e^{-j2\pi\left(x\frac{m}{M}+y\frac{n}{N}\right)}$$

Figure 3. 2D Fourier Transform

$$P(k,b) = \frac{1}{N}\sum_{a=0}^{N-1} f(a,b)\ e^{-\iota 2\pi\frac{ka}{N}}$$

$$F(k,l) = \frac{1}{N}\sum_{b=0}^{N-1} P(k,b)\ e^{-\iota 2\pi\frac{lb}{N}}$$

Figure 4. Separated Fast Fourier Transform

$$f(m,n) = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} F(x,y) e^{j2\pi\left(x\frac{m}{M}+y\frac{n}{N}\right)}$$

Figure 5.2D Inverse Fourier Transform

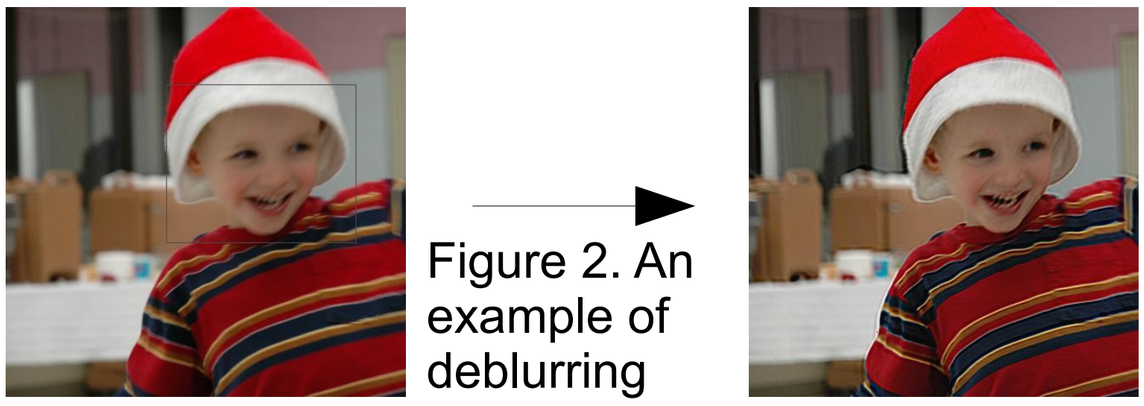

Figure 2. An example of deblurring

Results

My results so far are limited and inconsistent. My program to test transforming using the FFT and IFFT produces inconsistent results, varying from correct, to inverted colors, to distorted and pixelated (Figure 6). The next step is to troubleshoot this program and then move on to convolution, then deconvolution, then the largest part, which is blind blur kernel estimation.

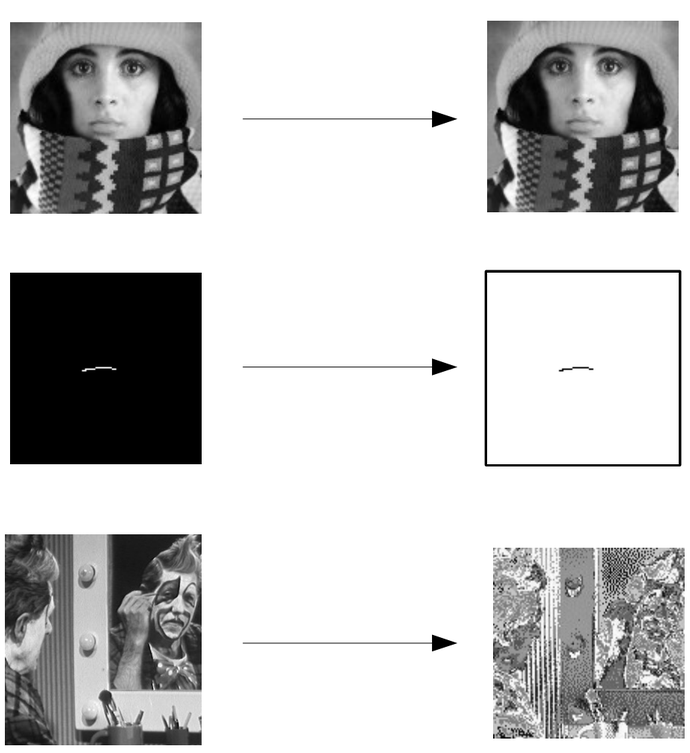

Figure 6. Various output combinations of my current FFT and IFFT code.