EE347 Project Report

OBJECTIVES

This report discusses a research project that investigated the use of Fourier optics to correct blurring of images caused by motion. More specifically, linear motion blur of images such as in photographing a speeding car or a shooting star has been considered. We have considered three different cases of linear motion blur and investigated ways of applying inverse filtering techniques to correct the blur. Following are the three cases that we investigated.

1. First we considered the simplest case where we assumed that we have a prior knowledge of the length and angle by which the image has been blurred. Assuming o(x,y) is the original image, b(x,y) and h(x,y) are the blurred and the point spread function respectively, we can relate the two images as follows.

$$B(x, y) = O(x, y)H(x, y)$$

Where the functions with the capital letters represent the Fourier transforms. At first blush it seems like, all we do is divide the blurred image by the Fourier transform of the point spread function H(x, y). This function obviously could have zeros at several spatial frequencies which suggests the solution to the problem is not merely dividing by H(x, y). This problem has been investigated and our solution has been discussed.

2. Secondly, we approached the problem with no prior knowledge of the blurring. For this case, the only information we have is the blurred image. In this case, since the direction and the blurring extent is not known we will have to somehow estimate these values from the degraded image and try to correct the blur using those values. One approach is to use the cepstrum (which is like the spectrum of a spectrum with some adjustments) to periodicity and thus help us determine the extent of blur [1]. We have used this approach to estimate amount of blur. We have tried to solve the problem in two ways. The first method is the direct method which simply estimates the blurring and tries to correct it once. The other method is correcting the blur iteratively which tries to correct the blur by estimating the blur and improving the image for each estimate.

3. Lastly, we investigated the case where there is noise in addition to linear motion blur. We investigated how the situations discussed above in the presence of noise and how we can improve our solutions for cases 2 and 3 when noise is added. Minimization of mean-squared error was one of the major objective criteria we considered when optimizing the restoration of the image.

APPROACH AND TECHNIQUES

a. Blurring and test images

We wrote a matlab code that degrades (blurs) the image. Linear motion blur is equivalent to convolving the image with a rect function. In matlab, we wrote a function that takes a length and an angle and outputs the blurred image. We made use of the "fspecial" function in matlab to create the point spread function (PSF) which is the blurring filter.

b. Unknown blur length and noise free

When the angle or the length of the blur is unknown, it is possible to estimate this angle quite accurately using analysis in the cepstrum domain. To get to the Cepstrum domain, we started by finding the magnitude of the 2-D discrete fourier transform of the original image. Next, we took the logarithm of the frequency spectrum and squared it, which highlighted the most powerful frequencies and diminished the power at DC. Taking the 2-D inverse fourier transform of this space yielded the image in the cepstrum domain, and we were able to use edge detection and the Hough transform to find the most likely blur angles.

Whether we are given the blur angle or we have estimated it, for a given angle we can estimate the blur length in an image. With the image in the cepstrum domain, we rotate the image by the expected blur angle and then take the average of each column. By finding the number of columns between zero crossings, we were able to find the periodicity and estimate the blur length for a given angle with a fairly high accuracy.

We wrote a matlab program that estimates the 20 most likely blur angles and estimates the blur length for each. It then displays each image in the order of it's blur angle likelihood. For images clearly blurred along one angle, this algorithm works very well, but it does require human input to determine whether each result is an improvement or not.

While testing this algorithm with know blur angles, we found that angle estimates very close to

c. noise in the deblurring algorithm

High-frequency or impulse shaped ("salt and pepper") noise causes errors in image representation in the cepstrum domain by adding power to higher frequencies that should not be in the blurred, inherently low-frequency original image. This can cause the Hough transform to miss important edges in the cepstrum domain, and it is difficult to estimate either the blur angle or the blur length. Although applying a low-pass or averaging filter the original image is not advisable in a deblurring algorithm, a non-linear noise-canceling filter such as a median filter will eliminate noise without further blurring the image. We employed a median filter in our blur angle and blur length estimation routines with great success.

RESULTS

a. Known blur length and noise free

In this case since we already know the extent and orientation of blur we could just use those values to undo the blur. This is the equivalent of Inverse filtering and in frequency domain it's equivalent to dividing by the fourier transform of the point spread function which is called optical transfer function (OTF). The only problem that arises in taking this approach is the fact that the OTF could have zeros and we can't divide by zero. Here we should note that the image can not be recovered completely because some of the frequencies in the original image have been zeroed out even in the noise free case [2]. In matlab, we set up a 'for' loop that looks for the zero frequencies in the OTF and sets them to a very small number(in the order of 10^{-6})

This solves the divide by zero problem.

We have attached several images.....

a. unknown blur length and noise free

b. effect of noise in the deblurring algorithm

CONCLUSIONS

SOURCES

[1] cepstrum
http://www.soulinvitation.com/cepstrum/index.html