01 May 1994

Image Processing Techniques and Implementations in Software for use with a CCD Camera

Clarence Franklin Jr.

Image Processing Techniques and Implementations in Software for use with a CCD Camera

Clarence Franklin, Jr.
Department of Computer Science, University of Missouri-Rolla
Rolla, Missouri 65401

ABSTRACT

The concept of image processing is used whenever there is a reference to digital images. By random and natural errors introduced into the image during collection and transmission, the images are distorted. Image processing techniques are used to eliminate these error before viewing of the images. For this experiment, the images provided by the ST-6 CCD Camera System will be used in the development and implementation of image processing techniques into a software package.

INTRODUCTION

Digital images have become a fact of everyday life. They are used in a wide range of applications, spanning the range from satellite images to digitized photographs to spectroscopy profiles and beyond. During the process of collecting and transmitting, natural and random errors are introduced into these images. For instance, noise can be introduced into the image when it is transmitted, through the air or over a line, to some receiver. Another instance could be overexposure of the image during the time when the image is collected (too much light was on the object whose image was being taken or too much light entered the camera). These errors, which are often unavoidable, can distort the image. A means to correct these errors is then needed; this is the purpose of digital image processing.

EQUIPMENT

The choice of images to be processed for this experiment are supplied by the Model ST-6 Professional CCD Imaging Camera [1]. This choice is partly attributed to the wide range of applications this camera can be used to perform. The camera, since it can take long exposures (several hours in length), is ideal for taking astronomical images (since long exposures are needed to capture faint light from distant stars), yet can take a profile of a light sample supplied via optical cable (which is unlike a traditional image). Also, CCD (charge-coupled device) technology is commonly used in light weight home video cameras, so the techniques used to process ST-6's digital images can be used to process other digital images as well.

To understand how the images are collected for processing, a brief explanation of the ST-6 system is warranted. The ST-6 Camera consists of a camera head, a CPU (central processing unit) box, a

wall transformer and necessary cables & software. The camera head contains the CCD, the thermoelectric cooler and Analog-to-Digital (A/D) converter. During an exposure, the CCD, which is an array of optical detectors, converts incoming photons into electrons and stores the charge until later. Each element, called a pixel, does this operation individually, thus obtaining the digital image information. Later, the readout electronics use the 16 bit Analog-to-Digital converter to convert the stored charge into a digital number from 0 to 65535 and transmits the number to the CPU, which is the display, analysis and storage device for the images. If the photographer wishes to view the image at a later time, the image can be saved as a special "image" file which the software recognizes at the later date.

The choice of programming language to write the software to process the stored images in was Borland C++ [2]. Borland C++ is very versatile and supports many of the graphics functions needed for image display and processing. Also, the dual MS-DOS/Windows package, that was purchased for this experiment, allows for programming in a MS-DOS environment and a Windows based environment.

DIGITAL IMAGE STORAGE AND PROCESSING TECHNIQUES

I. Tagged Image File Format (TIFF) [3]

The file format chosen to store image files for this experiment was the Tagged Image File Format (file extension .TIF). This choice was made due to the several advantages of this file structure. Many graphics file formats follow a standard structure of placing a fixed-length header (which contains important information about the image) at the beginning of the file, followed by a fixed-length image data area. This approach makes the file very inflexible. If any changes are made to the data contained in the file, the whole file must be reprocessed and rewritten to a new file. A TIFF file takes a different approach which can be seen by examining the file structure.

First, a single 8 byte header appears at the beginning of the file (not a 2K or larger header). This header contains 3 pieces of information: 1) the type of machine the file was made with (Intel or Motorola), 2) the version number of the file format and 3) the pointer to the first image file directory (IFD). The image file directory contains all of the information about the image not found in the pixel data (such as, imagewidth, imagelength, Xresolution, etc ...) and pointers to the pixel data. Each entry in the IFD is 12 bytes in length (except for the 1st entry which contains the number of entries in the IFD) and contains a tag & pointer/data. If the data that the tag refers can be placed in the 12 bytes accompanying the tag, the data is placed in the directory entry; otherwise, the data is placed elsewhere in the file and a pointer to the data is placed in the directory entry. A list of tags that can be used in the IFD is offered in [Table 1]. At the end of the

IFD, is one more entry that points to the next IFD (the entry is zero if there if not another IFD in the file).

The flexibility of this file format allows for a dynamic pixel data space, a dynamic "header information" space and multiple images per file. Plus, if any changes are made to the file, only the information that is updated is changed in the file (restructuring of the whole file is not necessary). Barring the added complexity of this file format to software design, the TIFF file structure is ideal for image storage and processing.

II. Conversion of ST-6 Files to TIFF Files

The TIFF file structure may be ideal for image storage, unfortunately the ST-6 CCD Camera saves its files in its own format (file extension .ST6). The files are in a fixed format with a 2K header and a fixed pixel data space. A sample of the header of a ST-6 image file is offered in [Figure 1]. To be able to use the information in the ST-6 image files with our software, they must first be converted to TIFF image files. This can be done easily by taking out of the ST-6 header only the information needed and placing it in the new TIFF file using tags. Then the pixel data is transferred from the ST-6 file over to the new TIFF file according to the TIFF data format. After these steps are performed, the old ST-6 file can be discarded and all of the image processing can be performed on the new TIFF file. This conversion is necessary to prevent extra code from being added to the original software package. Instead, file conversion code can be introduced externally.

III. Gray-Scale Modification [4]

Gray-scale modification is a simple and effective way of modifying an image's dynamic range or contrast. For instance, if an image has been taken with too much light in the background, gray-scale modification can increase the shadowy regions of the image, giving the image more clarity. The technique is described below:

$$n_x = \textit{input intensity} \qquad n_y = \textit{output intensity}$$

These values can be related to each other through the transformation:

$$n_y = T[n_x]$$

Given all of the input intensities and the output intensities, a histogram of the image can be produced. The histogram function, denoted below, represents the number of pixels that have a specific intensity as a function of the intensity variable n_x :

$$p(n_x)$$

Due to the simplistic nature of the transformation, it is possible that the operator viewing the image could choose a new transformation by looking at the processed image and the histogram of the image. Or, the operator could use the following automatic transformation to modify the image's gray-scale:

$$p_d(n_y)$$

Represents the desired histogram of the output image (a given set of values), which normally has a maximum around the middle of the dynamic range and decays as the intensity increases or decreases.

$p'(n_x)$, $p'_d(n_y)$ *represent cumulative histograms given by*:

$$p'(n_x) = \sum_{k=0}^{n_x} p(k) = p'(n_x - 1) + p(n_x)$$

$$p'_d(n_y) = \sum_{k=0}^{n_y} p_d(k) = p'_d(n_y - 1) + p_d(n_y)$$

Then we can find n_x and n_y for:

$$n_y = T[n_x]$$

Such that:

$$p'_d(n_y) \quad , \quad p'(n_x)$$

are closest together.

IV. Median Filtering [5]

Median filtering is a nonlinear process useful in reducing impulsive or salt-and-pepper noise. It is also useful in preserving edges in an image while reducing random noise. The technique is described below:

0	0	0
0	X	0
0	0	0

A window slides across the image. In each window, is contained X = the pixel to be processed and 0 = the surrounding pixels. In each window position, the surrounding pixels are averaged together and replace the pixel that is being

processed. Then the window moves on to the next position, unit the whole image has been covered.

This type of filtering has an advantage over other types of filtering, such as low pass filtering and high pass filtering. Median filtering will smooth the image, removing the noise, but will preserve discontinuities in a step function and can smooth a few pixels whose values differ from their surroundings without affecting the other pixels. Thus, the median filter eliminates the blurred effect of low and high pass filtering.

V. Thresholding [6]

Thresholding is a type of contrast manipulation, however unlike the previous techniques, it is not designed to enhance the image contrast. Instead, thresholding divides an image into segments by a single gray level threshold. For instance, if an image that contained dark areas and light areas, thresholding would darken the pixels that fell below the threshold value and lighten the pixels that fell above the threshold value. This techniques sharpens specific areas of an image and removes the extraneous areas at the same time.

VI. Other Image Processing Techniques

Other image processing techniques that were considered to be included in this software package were:

1. Color Enhancement [7]: adding color to a black and white image to enhance the present detail of the image.
2. Registration [8]: the overlaying of multiple images and having the frames line up correctly with each other.
3. RGB Color Images: allowing input of three images (1 Red, 1 Blue and 1 Green), registering the images into one picture and based on this information produce a color image.
4. Maximum Entropy [9]: using a probability distribution function that has the maximum entropy allowed by the present information in the image, to recover the information that is missing from the image.

CONCLUSION

Multitudes of image processing techniques have been discovered and more are on the way with the creation of new imaging techniques. Not all techniques work for all images. For instance, a low pass filter will remove the low noise from an image, but if the image clarity is sufficient before the filter is applied, the filter will blur the image instead of enhancing the image. So, careful consideration has to be used to determine which techniques should be used for a particular application.

I would like to conclude by stating that my research participation in this experiment has been more of a learning experience, than that of producing a product. I have learned

several image processing techniques over the period of this experiment. However, I compounded the task of learning a new language (Borland C++), experimenting with the special functions of the language, picking up some of its finer aspects, learning image processing techniques and implementing those techniques in a new language. The task proved to be too much to do in the time period I had allotted myself. Although, a foundation of doing further research in this area has been made, of which I plan to continue and extend.

With the continuing integration of computers into the mainstream of our everyday lives, new techniques in image processing are becoming more and more valuable. Research in this area should be continued.

ACKNOWLEDGEMENTS

The author would like to thank the following: Mr. Christopher P. Smith for introduction of the project concept and providing of the image data files; Dr. John L. Schmitt for providing project specifications on equipment and software design; and finally Dr. Fikret Ercal for continued guidance in the area of Computer Science.

REFERENCES

1. Model ST-6 Professional CCD Imaging Camera, operations manual, Santa Barbara Instrument Group, No Date Provided.

2. Borland C++, computer software, Borland International, Inc., 1992.

3. Alfred Poor, "Lab Notes: Looking at the TIFF Specification From the Inside" PC Magazine 17 December 1991: 371-376..

4. J. S. Lim, ed. M. P. Ekstrom, "Image Enhancement", Digital Image Processing Techniques (Academic Press, Ontario, Fl., 1984), pp. 5 - 11.

5. J. S. Lim, ed. M. P. Ekstrom, "Image Enhancement", Digital Image Processing Techniques (Academic Press, Ontario, Fl., 1984), pp. 25 - 31.

6. R. A. Schowengerdt, Techniques for Image Processing and Classification in Remote Sensing, (Academic Press, Ontario, Fl., 1983), pp. 68-71.

7. J. G. Moik, Digital Processing of Remotely Sensed Images, (N.A.S.A., Washington, DC, 1980), pp. 141-158.

8. J. G. Moik, Digital Processing of Remotely Sensed Images, (N.A.S.A., Washington, DC, 1980), pp. 187-198.

9. R. M. Haralick, et al., ed. H. Stark, "The Principle of Maximum Entropy in Image Recovery", Image Recovery: Theory and Application, (Academic Press, Ontario, Fl., 1987), pp. 157-193.

Tag (decimal)	Hexidecimal	Tag Name
254	FE	NewSubfileType
255	FF	SubfileType
256	100	ImageWidth
257	101	ImageLength
258	102	BitsPerSample
259	103	Compression
262	106	PhotometricInterpretation
263	107	Tresholding
264	108	CellWidth
265	109	CellLength
266	10A	FillOrder
269	10D	DocumentName
270	10E	ImageDescription
271	10F	Make
272	110	Model
273	111	StripOffsets
274	112	Orientation
277	115	SamplesPerPixel
278	116	RowsPerStrip
279	117	StripByteCounts
280	118	MinSampleValue
281	119	MaxSampleValue
282	11A	XResolution
283	11B	YResolution
284	11C	PlanarConfiguration
285	11D	PageName
286	11E	XPosition
287	11F	YPosition
288	121	FreeOffsets

Table 1. Tagged Image File Format (possible tags for IFD entries)

Tag (decimal)	Hexidecimal	Tag Name
289	121	FreeByteCounts
290	122	GrayResponseUnit
291	123	GrayResponseCurve
292	124	Group3Options
293	125	Group4Options
296	128	ResolutionUnit
297	129	PageNumber
301	12D	ColorResponseCurves
305	131	Software
306	132	DateTime
315	13B	Artist
316	13C	HostComputer
317	13D	Predictor
318	13E	WhitePoint
319	13F	PrimaryChromaticities
320	140	ColorMap

Table 1. Tagged Image File Format (possible tags for IFD entries) (continued)

```
ST-6 Compressed Image or ST-6 Image
File_version = 2
Data_version = 1
Exposure = xxx
Focal_length = xx.xxx
Aperture = xx.xxx
Response_factor = xx.xxx
Note = xxxxxxxx
Background = xxx
Range = xxx
Height = xxx
Width = xxx
Date = xx/xx/xx
Time = xx:xx:xx
Exposure_state = xx
Temperature = xx.xx
Number_of_exposures = xx
Each_exposure = xx
History = xxxxxx
Observer = xxxxxxxx
End
```

Figure 1. Header format of a ST-6 image file.