Fundamentals of Photogrammetry in Automatic Object Recognition

A thesis submitted in patrial fulfillment of the requirement

for the degree of Bachelor of Science in

Physics from the College of William and Mary in Virgina,

By

Evin Taft Grano

Accepted for_______________________________________

(Degree Requirement)

Advisor: Dr. Dennis Manos

Williamsburg, Virginia

May 2001

Introduction and Motivation:

The scope of this paper is in the area of physics-based automatic object recognition algorithms research. This includes geometric representations of objects from imagery taken from a digital camera. It also includes radiometric behaviors of objects in imagery. The focus of this study is to concentrate on electro-optical and multi-spectral imagery. This study goes into the detail behind the image space representation of objects and the geometric transformation on objects as part of the imaging process. This will provide the groundwork that will build object-space to image-space transformations (e.g. math models). The research will explain the cost/benefits of detailed object models and resolutions, provide written algorithms and a basic low-level program for spatial deconflections of imagery. This paper will attempt to explain some problems/limitations of this technology and try to solve some of them associated with automatic object recognition. Focusing on distinct non-complex objects (e.g. squares, triangles, and circles), this study has gone into the details behind the image space radiometry of particular objects. Using physics-based principles (with derivations) of Fourier analysis and autocorrelation, this research will determine most probable pixel values to be associated with a particular part of an object. Another focus of this paper is to explore different statistical models in an effort to represent overall object radiometry and determine an appropriate decision scheme for finding the existence of an object in the image.

The purpose of this project is to create a computer program that will be able to correctly identify a known set of objects and their locations by means of spatial modeling and multi-spectral analysis in a digital photograph. The objects that are photographed have diffuse surfaces, which inhibit reflection of bright spots that would wash out the colors. Pictures of objects illuminated by a diffuse source (i.e. fluorescent light) and a point source of light were taken from a known height.

A radiometric modeler was written to perform a multi-spectral analysis of the objects in the picture. The radiometric model includes light reflection, shadowing, and color analysis of the captured images. The object of this modeler is to locate and identify areas of interest in the image to be analyzed by a subsequent spatial model. This multi-step modeler approach is very effective in reducing the analysis time. Preliminary work was restricted to the set of objects of simple distinct shapes such as squares, triangles, and circles, each taken in three basic colors: red, blue, and green. Objects were placed on two backgrounds: white and black to test the significance and effect of shading on the image analysis.

The spatial model constructs the object's shape. This model first locates the center of mass of the object in the picture and by rotating the image finds object points that have local extrema in the distance from the center. These extrema are then pattern analyzed to compare to n-sided polygons until a suitable match is found.

Definitions and Uses:

This research draws on the fields of photogrammetry, image analysis and processing, and machine vision. This section will define some of the nomenclature such as spatial deconflection and radiometry, which are important in understanding image analysis and automatic object recognition. A broad definition of the field of Photogrammetry has been given by Wolf:

> Photogrammetry is defined by the American Society of Photogrammetry as the art, science, and technology of obtaining reliable information about physical objects and the environment through processes of recording, measuring, and interpreting photographic images and patterns of recorded radiant electromagnetic energy and phenomena. [1]

There is a further field of photogrammetry called metric photogrammetry which has the goal of making accurate 3-D measurements using 2-D images. The deconvolution of 3-D points from their projections onto 2-D images is called spatial deconflection. Interpretive photogrammetry has the goal of recognizing and identifying pertinent information in the image using as much information as possible, including spectral and other forms of data analysis. This type of image processing and analysis makes use of the power of scientific computing to do the necessary work. Image processing can be used to clarify an image that has been distorted in capture by the use of different time or frequency domain algorithms, such as Wiener filtering, Fourier decomposition and others. Image Analysis is the technique of interpreting the image for the user. A complete description can be found in standard texts such as Sonka, Hlavac, and Boyle's book *Image Processing, Analysis, and Machine Vision* [2],

> The **spatial resolution** is given by the proximity of image samples in the image plane; **spectral resolution** is given by the bandwidth of the light frequencies captured by the sensor; **radiometric resolution** corresponds to the number of distinguishable gray-scale levels; and **time resolution** is given by the interval between time samples at which images are captured.

All of these "resolutions" will be needed in our analysis. For our purposes, time resolution will mean the use of multiple images of the same scene taken at different angles. This technique is also called “stereoscopy”.

The first use of photogrammetry was in the use of topographical mapping in the use of the U.S. Coast and Geodetic Survey, now called the National Geodetic Survey. With the advent of the satellites, photogrammetry has been used to survey the Earth and the Moon and even other planets. It has been used most extensively for intelligence gathering by agencies like the Nation Reconnaissance Office (NRO), and Central Intelligence Agency (CIA) and others. Image processing and analysis for automatic object recognition is a part of a growing effort toward the development of Machine Vision and Artificial Intelligence for commercial and medical purposes.

Methodology of Digital Images:

The quality of the images is determined by the following criteria: spatial resolution, spectral resolution, radiometric resolution, and time resolution. All of these are determined by the capabilities of the (digital) camera that captures the image. A considerable part of my work was the analysis of the requirements for this research and the specifications of a commercial camera which fit the cost and the delivery time as well as the technical needs of the project. The camera needed to provide several features to which would be indispensable to the project. We will describe some of the motivation for these features below.

Digital cameras capture the color images on a small chip about the size of a postage stamp. This semiconductor chip receives and encodes the image coming through the lens. The kind of chip technology is called a Charge-Coupled Device or CCD. The CCD is the chip which measures the photons hitting the chip. In the digital camera, it is the device which actually captures the image. Resolution and color accuracy has advanced to give a consumer megapixel (and multi-megapixel) cameras. A common analogy that is used is an array of buckets on conveyor belts, raindrops represent these photons of light falling onto the CCD surface and being captured in 'bit buckets' (pixels). The conveyor belts, which empty these buckets, are known as the shift registers, in progressive CCD chips.

The CCD reads the data: one horizontal line, shift the vertical down one pixel, one horizontal line, and repeats. Even though a chip can be listed as being 2 megapixels, this is not entirely correct. The truth is each pixel can in fact capture any color in the visual and sometimes ultraviolet and infrared spectrums. The CCD chip is a monochrome device and therefore requires a color filter. This is usually produced by directly applying dye onto the surface of the CCD chip to produce a colored image. The term “effective pixels" is the actual resolution of the CCD chip in most consumer digital camera. For example the Nikon CP950 reports a CCD resolution of 2.11 megapixels (the total number of pixels on the CCD) but only has an effective resolution of 1.92 megapixels (1600x1200). A large number of digital camera specifications report the CCD resolution and the effective resolution. There are a couple of motivations for this method of resolution construction. In order to create a standard size image, this construction produces a 1600x1200 pixel image with a 4:3 image ratio which incidentally is the same as a computer monitor. On the other hand if the image were taken at full resolution, one would end up with an odd shaped image. Second, the most significant reason for this is the CCD is an analogue device, certain rows or columns of the CCD chip are intentionally dyed black to allow the internal circuitry of the camera to obtain a "black level" or zero light. “Effective pixel count" is therefore classified as the true resolution of the camera, that is, the absolute limit of detail the camera can or could capture. In addition to this tradeoff of resolution and effective resolution, certain cameras use the method of interpolation to increase the resolution of the output image. This is a very common practice in optical analysis to compress the data and then interpolate the data back to its true size.

The filter pixels are arranged in the interleaved pattern of G-R-G-B called a Bayer mosaic pattern, with each pixel capturing a different color. An example of this in a 2-megapixel CCD chip would correspond to 540,000 red pixels, 540,000 blue pixels and 1,080,000 green pixels. An 1800 x 1200 pixel CCD using the Bayer pattern would have:

900 x 600 RED pixels

900 x 600 BLUE pixels

900 x 1200 GREEN pixels

To an observer, there is a larger number of green pixels than other colors due to the sensitivity of the human eye to the luminance of green light. The individual pixels for 24-bit color are calculated by averaging the values of the central pixel and the immediate surrounding pixels.

Other manufacturers have put into practice another filter. Notably, Canon in its PowerShot series of digital cameras (such as the Canon PowerShot Pro70) has painted different patterns of color painted onto the surface of the CCD (dyes) from the Bayer Mosaic Pattern. They used the colors cyan, yellow, green and magenta. The individual color for each pixel is calculated using the interpolation method explained above. The algorithm, which does this calculation, is slightly more complicated. This is a subtractive method of interpolation that was developed by Cannon that adds weighted coefficients to the different colors and takes their difference from a base value. One should note that using a pattern of four colors, each color represents a proportional image of each of the weighted pixel values.

To the human eye, CCD chip technology is accurate with little or no distortion if the sensor array holds many sensors or “buckets”. The problem with this design in scientific computing and accurate image processing is that it distort the image slightly through the interpolation process. This is the problem we would like to avoid selecting the digital camera that does no interpolation.

The camera selected was the Roper-Scientific Photometric “CoolSNAP” Color camera. This is a very high quality camera that is mainly used for scientific purposes and was developed for the biotechnology sector. The CoolSNAP Color camera captures a digital image without the use of pixel interpolation. Unlike the consumer camera, the CoolSNAP Color camera has precisely the same number of red-green-blue sensors. Every time a digital image is captured the camera takes three separate images of the scene, in the spectral scales of red, blue, and green. After these three images are taken and stored, the camera mathematically overlays the images to construct the proper digital image, without distortion, for proper display purity. This permits accurate “3-wavelength” spectral decompositions, which cannot be done in a straightforward manner with a common consumer digital camera. The CoolSNAP Color camera features a variable time and shutter speed. For this project, the shutter speed was fixed at 0.2 milliseconds.

Mathematics:

In this section, we will briefly describe the mathematics and the physics involved in this research project. The primary tool, deconvolution, is familiar to most physicists from the distribution-potential problem of classical electrostatics: Given a charge distribution in a region of space, find the potential at a specific point. What we are attempting to do in this project is to reverse this process. We are given a two-dimensional potential (2D digital image) and we will deconvolve the potential to construct the most-likely charge distribution (or 3D object), which is consistent with those given data. The most useful mathematical tools are the Fourier transform and the Green's Function. In effect, we are replacing a point charge with a pixel value. The brightness and the color will be our "charge" values. Therefore, we have reduced the surface of a 3-D object to a "charge" distribution due the light reflection on the surface of the object. From Jackson [3]'s explanation of charge distribution and the Green's Function, we are searching for the **scalar potential** $\Phi(x)$, which in electrostatics is explained by equation:

$$E = -\Delta\Phi \qquad [1]$$

Now, we can show the scalar potential as given in terms of the "charge" density, r in equation:

$$\Phi(x) = \int \frac{r(x')}{|x - x'|} d^3x' \qquad [2]$$

where the integration is over all "charges" or pixel values in the universe. This equation involves the local discrete or continuous distribution of "charge" with no boundary surfaces, therefore giving rise to this general solution. This is not the purpose of our research because we are inducing a boundary condition by capturing the data at a known distance. We are constraining our "charge" to a finite area with the projected potential on the surface of our captured image. To find the ideal impulse, one may use a linear superposition of such functions (i.e. Fourier Transforms and Green's Functions) to reconstruct the image. The impulse function can be represented as a 2-dimensional Dirac delta function, $\boldsymbol{d}$ (x, y),

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \boldsymbol{d}(x, y)dxdy = 1 \qquad [3]$$

the function $\boldsymbol{d}$ (x, y) = 0 for all x, y $\neq$ 0, and therefore defines a basis set for the image function at each point.

Another important operation needed in image analysis is **Convolution.** The convolution *g* of the two-dimensional functions *f* and *h* (*f* * *h*) is:

$$g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(a, b)h(x-a, y-b)dadb$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x-a, y-b)h(a, b)dadb \qquad [4]$$

$$= (f * h)(x, y) = (h * f)(x, y)$$

The effect of convolution is to 'filter' an image *f* to create a map *g*. Such filters may be created to represent time, frequency (spectral), or spatial resolutions of the camera.

Here the function *f* is an image function on a two-dimensional plane with coordinates *x* and *y*. This allows for the use of two-dimensional Fourier transforms to interpret three-dimensional objects. The two-dimensional Fourier transform of *f* is defined as,

$$F(u,v) = a\frac{1}{(2\boldsymbol{p})^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-2p\,i(xu+yv)}dudy \qquad [5]$$

and has an inverse Fourier transform by,

$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{2pi(xu+yv)}dudy \qquad [6]$$

The variables *(x, y)* denote image coordinates, and the *(u, v)* are the corresponding spatial frequencies. The left side of the equation in the inverse Fourier Transform function *f (x, y)* can be interpreted as a linear combination of simple periodic patterns $e^{2pi(xu+yv)}$. These include the cosine and sine of the real and imaginary components of the pattern. The function *F(u, v)* is a weighted function, which represents the elementary pattern. In discrete expressions, these weight factors are the coefficients of the expression.

One of the main functions of linear analysis in imaging is its use in scene analysis to understand the ordering of patterns in a digital image using autocorrelation and cross-correlation. The statistical theory is too involved to summarize in the scope of this paper, but an explanation can be found in Sonka, Hlavac and Boyle's book, *Image Processing, Analysis, and Machine Vision* [2]. We note that the matching criteria is formulated in terms of a "optimality criterion" using the elements of *f* and *h* as written below:

$$C_2(u,v) = \frac{1}{(\sum_{(i,j)\in V} |\ f(i+u, j+v) - h(i,j)\,|) + 1} \qquad [7]$$

$$C_3(u,v) = \frac{1}{(\sum_{(i,j)\in V} [\ f(i+u, j+v) - h(i,j)]^2) + 1} \qquad [8]$$

This is used to find a matching pattern in a digital image which has been stored as matrix elements.

With the presence of shading and radiometric properties, one can aid in recovering the shape of the object from a 2-D plane (digital color image). This is relatively easy to do with a 2D object on a plain surface, which we selected for this preliminary effort. The most important part of this technique is in edge detection. There are several algorithms that are used to develop the edges of object found in an image. The most widely know of these is Wiener Filtering that can be found in Sonka [2]. This is a technique developed to filter out any white noise in an image and to give clear outlines to objects. Another technique that used in this project was to contrast the different color values, which took advantage of the a-priori knowledge that there were distinctly colored objects in the picture. Such information is not generally available in scene analysis. The pixels that represent an object in the spectral domains can be searched separately. Pixel matches then determine the probability of their association with a target object. The techniques can also be used for any grayscale backgrounds with great success. It also works in developing the edges of multiple objects in a digital picture.

Experimentation and Findings:

The foundation of this project was the digital image taken with the RS CoolSNAP Color digital camera. Pictures were taken of different objects in different arrangements for analysis. The objects selected for this preliminary groundwork were 2-dimensional objects made from construction paper. The primary shapes were squares and circles of two different sizes in each of the colors: red, blue, and green. The digital picture was taken using a tripod at the height of 5 ½ feet. Several pictures were taken of different arrangement of the shapes on a white background (please see **Figures 1, 2, 3, 4**). The software that came with the camera took pictures in a 32-bit Tagged Image File Format

(TIFF) picture format. This is a very clear format for viewing with a digital photo view, but does not lend itself to text base interpolation of the data. In order to do proper analysis, the TIFF pictures were converted to Portable Pixel Map (PPM) with the use of Photoshop. The reason this digital picture format was chosen was because of the ease to which the data is stored. As described in the lecture handout of Park and Rahman [4],

> **Definition 2.2** PPM is the portable pixel map format used to define color images. It consists of an ASCII data header, followed by interleaved color detain either *raw* or *ASCII* format. There are three color bands red, green, blue (RGB). The header consists of:
>
> P6 (or P3)
>
> NumberOfColumns NumberOfRows
>
> NumberOfGreyLevels
>
> #------------
>
> The body of the header is followed by NumberOfColumns x NumberOfRows x NumberOfBands elements of data.

Values are easily read into arrays for manipulation. PhotoShop made the conversion from TIFF to PPM format easy. Once the PPM red, green, blue values were put into an array, the research program was written to interpret the pixel values. Since the picture was taken on a white background, grayscale values that were between 200-255 were the background. When first implemented, this technique was unsuccessful. The main point of failure is in the fact that the camera taking the picture reveals several optical properties that would make this method fail.

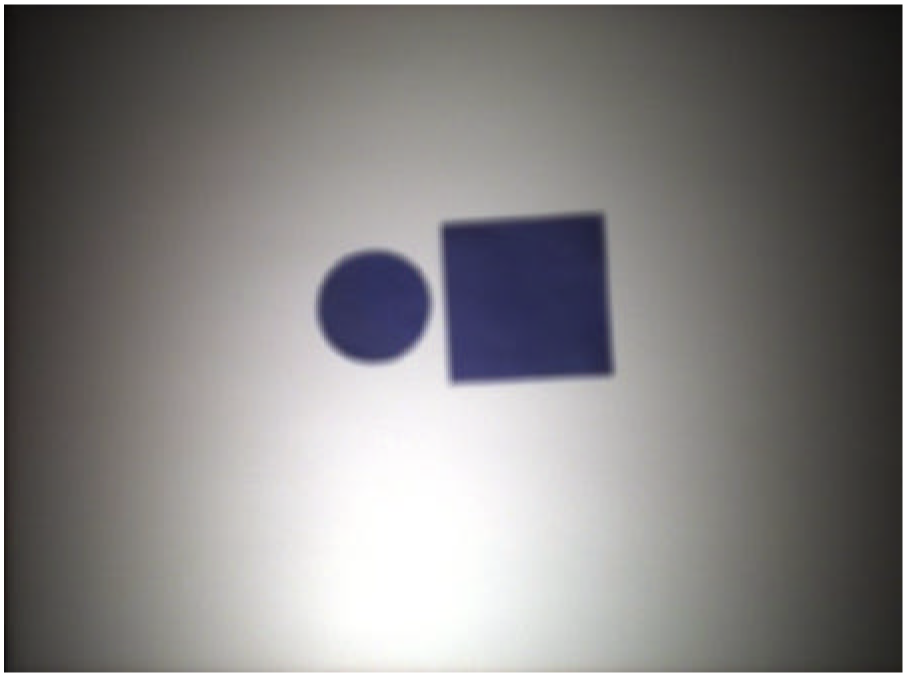

Figure 1:This is a digital image taken with a diffuse light source (ie floursence light). Notice the diffusion pattern that radiates from brightest in the center to darker at the edges

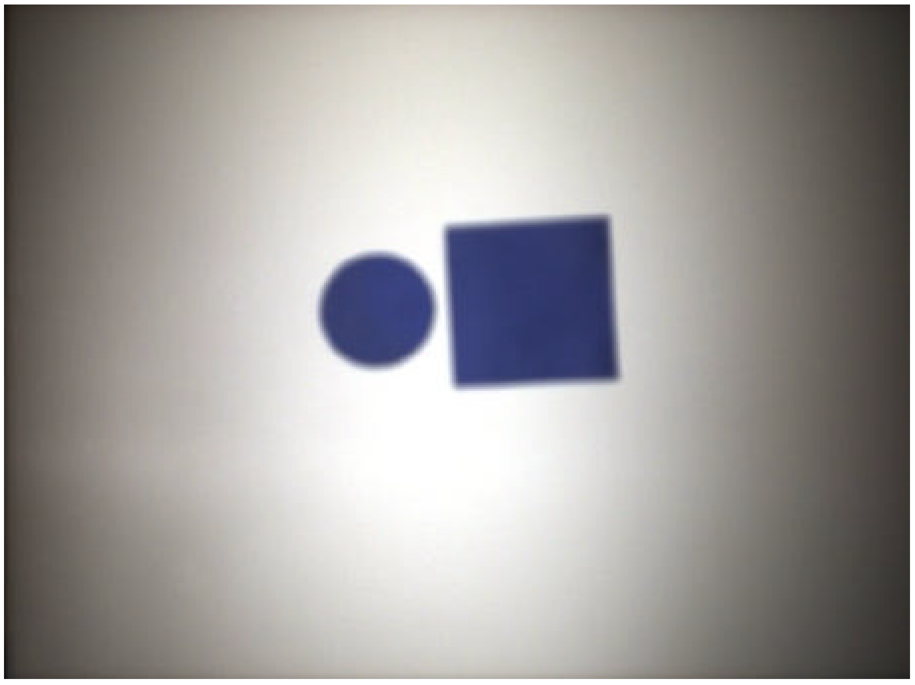

Figure 2:This is a digital image taken with a point source of light. Notice the diffusion pattern is much more apparent (ie brighter at the center/darker at the edges)

Looking at **Figures 1** and **2**, we see the presence of a diffraction pattern. This is caused by the resolution limit of the optical system of the digital camera. Therefore, one must improve the image quality by enhancing the resolving power of the digital camera. This problem is stems from the fact that the image of each pixel captured is never a point, but a distribution of light, or a diffraction pattern. The diffraction pattern limits the resolution and degrades the contrasts present in the data. There is clearly diffraction: one can see fringes radiating from the center to the outer edge of the images. The image in **Figure 2** was taken with a single point source of light at 45 degrees to the left and 75 degrees of elevation at a distance of three feet. The image in **Figure 1**, however, was taken with a diffuse light source that is omni-directional.

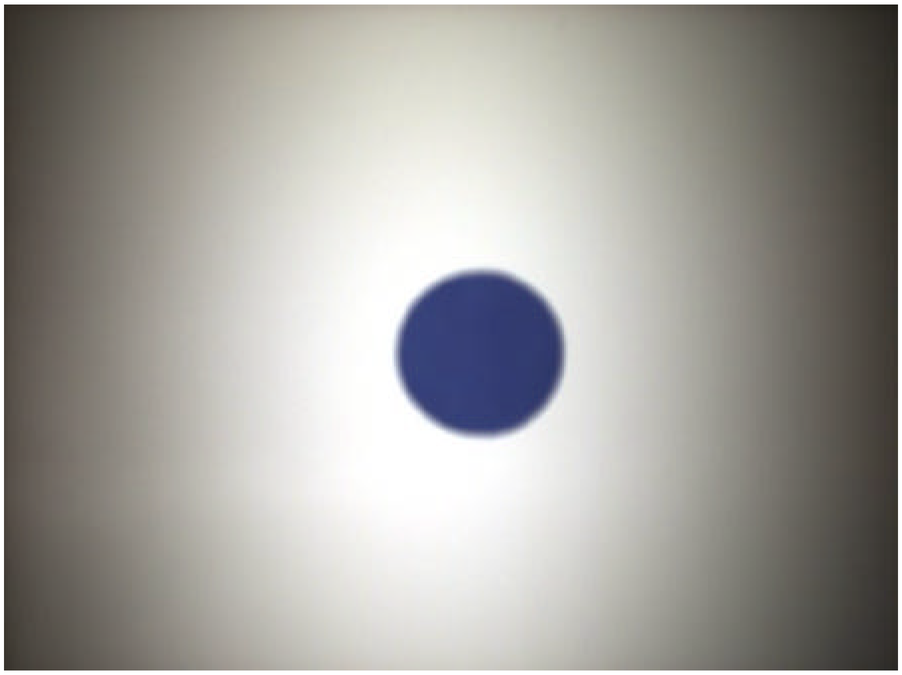

Figure 3:This is a basic example of a digital image which is used in all the calculations

One can see the image in **Figure 2** has a larger diffraction pattern than in **Figure 1**. The larger the diffraction pattern, the more degrading are the effects. In reverse, digital image quality increases if the radiant energy is more concentrated in the central region of the diffraction pattern. Thus, one should suppose that the optimal light or energy distribution must be at or near center of the observation focus. One method of countering this effect is called *amplitude filtering*. This well-known technique that can be found in Tippet [5] involves the action of locally varying the transmission factor of the optical system in an effort to provide a nonuniform illumination of the wavefronts that exit the camera opening. If we assume that the radial intensity is *I(W),* we can represent the energy concentration by the factor of encircled energy:

$$E(W) = \frac{\int_0^W I(W)d(W^2)}{\int_0^\infty I(W)d(W^2)} \qquad [9]$$

Which is the ratio of the energy distributed inside the circle of radius *W* to the total amount of energy in the diffraction pattern. This is called an Airy wave pattern. One of the scopes of this research is to defeat the negative effect of this diffraction pattern without the use of amplitude filtering.

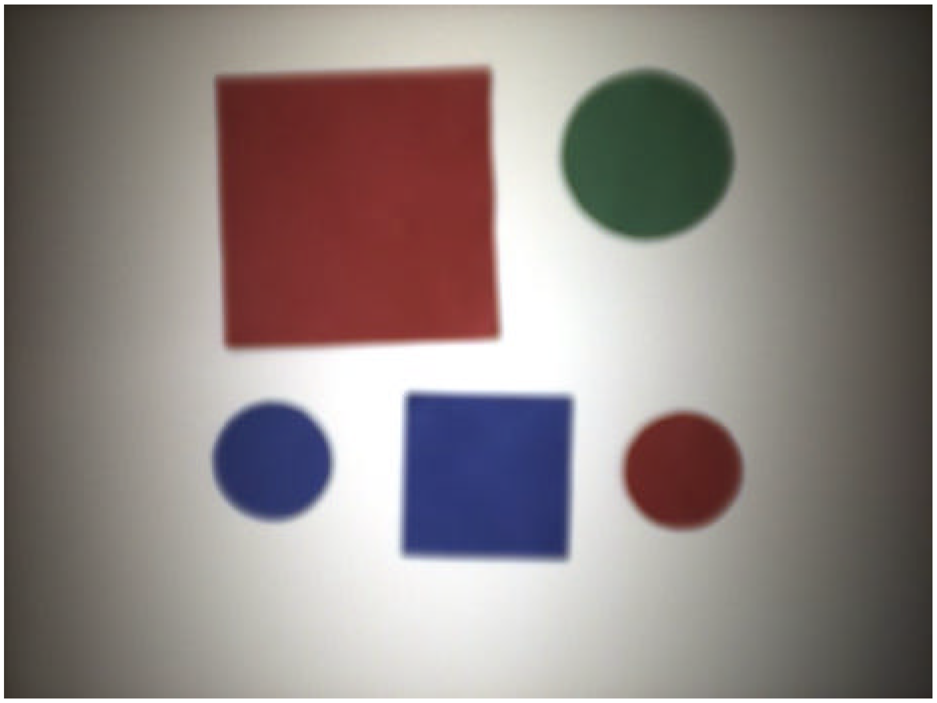

Figure 4:This is another basic digital image with multiple objects

Since we are working with a digital multi-spectral image on a plain background, we must look for changes in the relative values of the RGB color spectrum. When we see a disparity of the color values, we can discover the presence of a possible object in the digital image. Now, the next analysis is to figure out the right range. This is very important figure to assess because it is a trade off between classifying a pixel to the background and defining a definite edge to the object in the picture.

First, this study took a few sample values from the picture where it could be observed whether the pixels were part of the object or part of the background. Here is a sample of the findings taken from **Figure 3**:

X value	Y Value	Red	Green	Blue	Orientation
713	549	73	81	125	Object
100	247	25	27	31	Background
672	473	127	119	150	Object

Table 1: Example of pixels from Fig 1 at position (x, y) with Red, Green, and Blue values on scale 0-255 (i.e. 8-bits per color) and position inside or outside an object in the image.

One can see from the table that each pixel's Red, Green, and Blue values are in the range of 10-30 of each other. First, we calibrated the sensitivity of the program to filter all the pixels whose Red, Green, and Blue values were within 10 of each other. These pixels were considered part of the background. If a pixel's Red, Green, and Blue values were greater than 10, the program marked the pixel as an object pixel, meaning its probable presence in an object in the image. An example of this filtering can be seen in **Figure 5**.

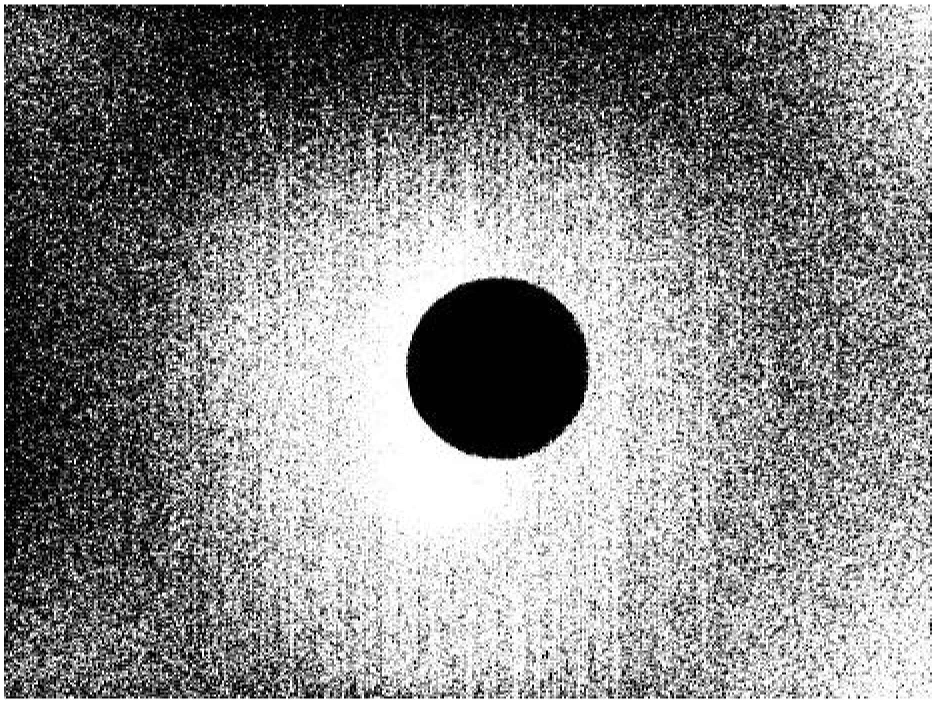

Figure 5:This is a sample output image for Figure 3. All the black pixels indicate a RGB color value that is greater by 10 than the other RGB colors. Poor object recognition.

The resulting image of **Figure 5** shows the lack of sensitivity. Since a 10 color was not sensitive enough, the difference was doubled to 20 and we see the resulting picture in **Figure 6**. The clarity of the image is better, but now we are beginning to see the cost of raising the difference value. The cost to clarity is in the presence of a less defined edge around the outside of the object and an overall reduction in the object. This is caused by the diffraction pattern of the light source and its resultant energy distribution on the image. Through observation and measurement, the optimal difference value is 25-30.

Difference Value	Object Pixels	Ghost Pixels
10	95429	9432
20	94975	1721
30	93433	398
40	89321	102

Table 2: Max Value difference between Red, Green, and Blue values. Object pixels are pixels that are bordered by Object pixels. Ghost Pixels are Object pixels that are not bordered by Object Pixels

This value yielded the lowest number of 'ghost' pixels, which indicate their presence in an object where there is none. The integrity of the shape of the object is maintained as can be seen in **Figure 7**.

Now, an algorithm must be implemented to sharpen the edges and remove the 'ghost' pixels. The following algorithm in psuedocode was innovative, but not very successful.

```
For 0 < x < Number of Columns{
    For 0 < y < Number of Rows{
       If pixelValue(x,y) == White;
          Then search up 5 pixels for a Black pixel;
               search left 5 pixels for a Black pixel;
               search right 5 pixels for a Black pixel;
               search down 5 pixels for a Black pixel;
               If Black pixels are found on more than 2 sides
               Then pixelValue(x,y) == Black;
    }
}
```

This algorithm looks for white pixel values that are surrounded by black pixel values on at least 3 sides. The algorithm conjectures that since at least 3 black pixels around the white pixel it is reasonable to assume that the white pixel should be black. This algorithm worked reasonably well, but did not produce the clarity in output that was desired. This algorithm was not mutually exclusive in its effect on neighboring pixels and causing a wave across the picture producing incorrect output that can be seen in **Figure 9**.

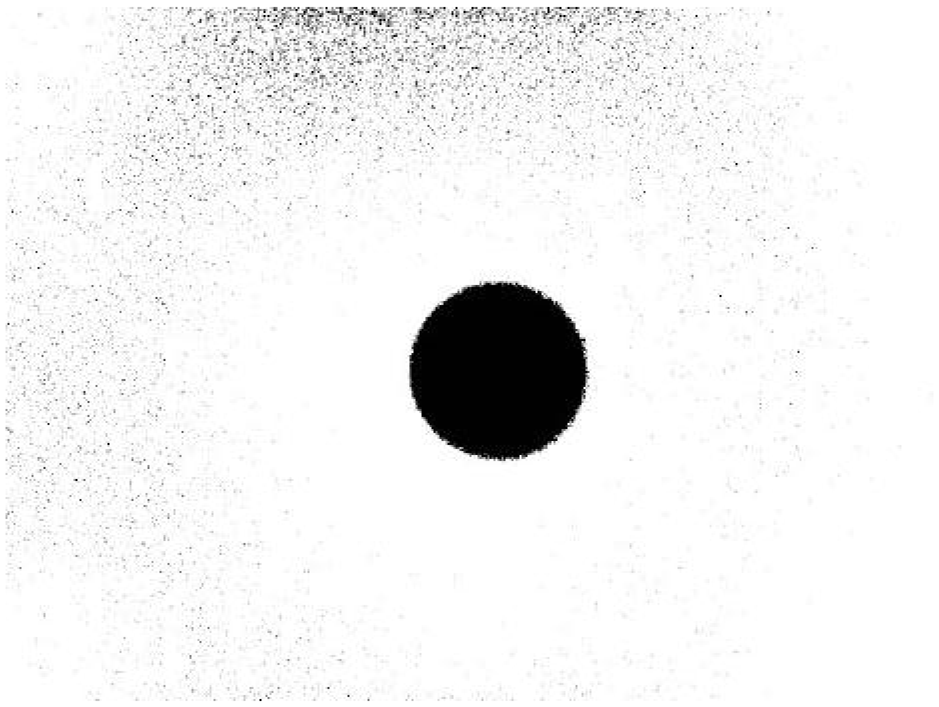

Figure 6:This is a sample output of Figure 3. All the black pixels indicate a RGB color value that is greater by 20 than the other RGB colors. Better than Figure 5, but still poor object recognition.

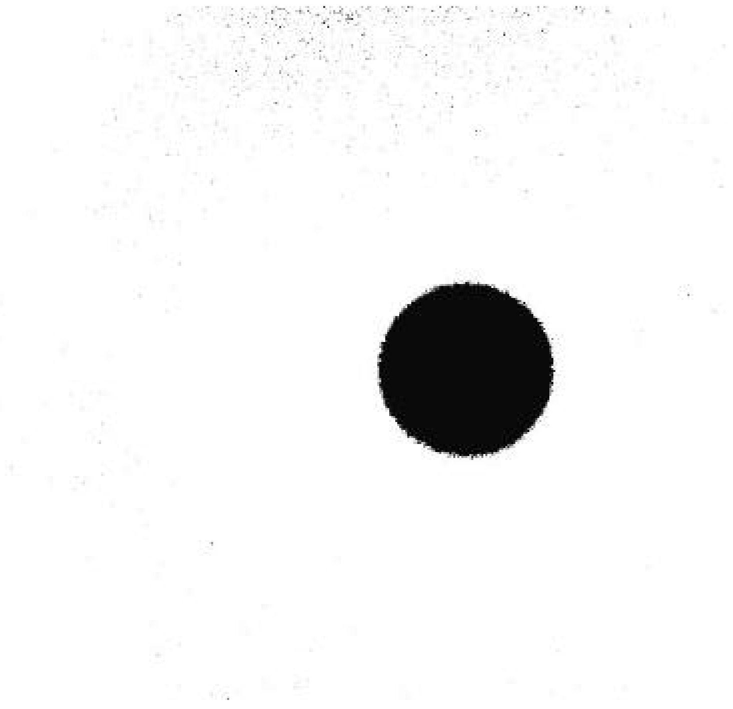

Figure 7: This is a sample output of Figure 3. All the black pixels indicate a RGB color value that is greater by 25-30 than the other RGB colors. This is the optimial RGB color difference.

Another shortcoming of this algorithm is that it only searches in 4 directions. To have a proper idea of the surrounding pixels, one must search in 8 directions:

Pixel (x-1,y-1)	Pixel (x,y-1)	Pixel (x+1,y-1)
Pixel (x-1,y)	Pixel (x,y)	Pixel (x+1,y)
Pixel (x-1,y+1)	Pixel (x,y+1)	Pixel (x+1,y-1)

Table 3: Example of target pixel and the surrounding pixels and their orientation to target pixel

A new algorithm would need to be developed that would search in the eight directions and be mutually exclusive in its interpolation of the color value of each pixel.

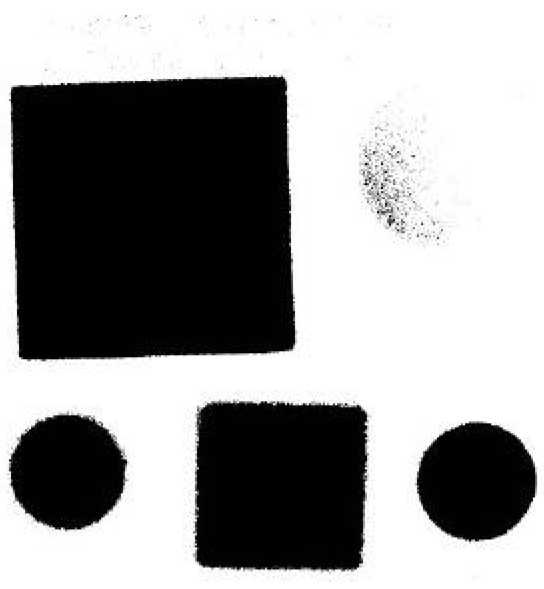

Figure 8: Sample output performed on Figure 4 with the optimal value. Notice the degrading effects of diffusion in the upper right corner with the green circle.

Mutual exclusivity of the pixel was obtained by a method of mapping which is essentially creating a second array of the same size. Thus, correlation of each pixel is mapped to the other array and then mapped back. This second array is used as a holding array that retains the values of the image in matrix form. The algorithm that was used is called an Edge Relaxing Algorithm. This is explained in detail in Sonka on pages 137-142. This algorithm blurs the edges of the object and re-interpolates the edge of an object. This technique also removes the 'ghost' pixels from the digital image. The algorithm is accomplished by adding up all the values of the pixels surrounding the target pixel and averages their value and assigns this average to the target pixel. For Example, with values of 0…255, i.e. Black…White:

255 (white)	1 (black)	1 (black)
255 (white)	96 (dark gray)	1 (black)
255 (white)	1 (black)	1 (black)

Table 4: Example of Edge Relaxing by interpolating the target pixel by the surrounding pixels

If performed multiple times, the object pixels expand the edges outward from black to white. At each pass of the algorithm, a threshold test is perform delegating a pixel value >127 to white and <127 to black. This produced **Figure 10** clearly defining the edges. The next scope of the project was to find the center of mass in the object. Much like a finite charge distribution, the program sums the object pixels and finds the average x-coordinate and average y-coordinate

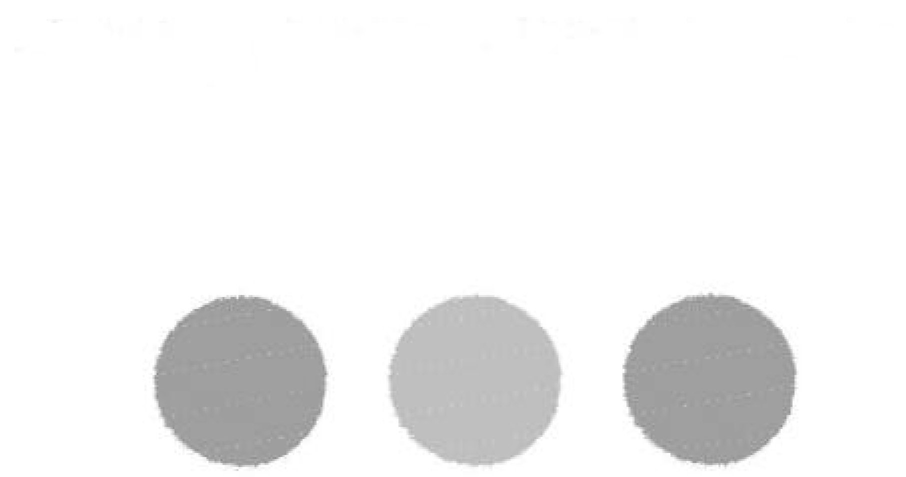

Figure 9:This is the poor output of the the first edge enhancement algorithm performed on Figure 3. This shows the effects of the lack of mutual exclusion.

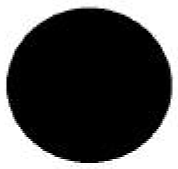

Figure 10:This is sample output of Figure 3 with the edge relaxing algorithm. Notice the clean edge and the absence of black pixels outside the object.

Prospective Developments and Conclusion:

This research produce a tool that solved one of the problems in digital optical data acquisition namely the effect of diffraction on a digital image. The research also provided a tool that can work in varied environmental condition such as varied light source, focused and out-of-focused image processing. All of the algorithms are stable working order. Some of the features were not obtained, such as actual Shape Recognition, but this will be relatively easy to implement with the groundwork that has been laid. The technique that that this study would suggest is to first isolate the edge of the object and take sample reading at varied degrees from the center, following the edge at increments of an angle Theta. At each of these samples, test for a local extrema. The number of extrema that were discovered would be the **n** sides of the polygon. If there were no extremes then the object would be a circle.

This project successfully explains the capture of data and interpretation of waves through a digital camera and some helpful techniques that can be employed in object recognition. This is a very brief synopsis of the research project that is in progress. This project uses a basis to develop a greater background and motivation for the physics and the mathematics that are inherently laden in Automatic Object Recognition. With the advent of great detail in digital imaging, this field is a rapidly growing sector of Electro-Optics and Electro-Dynamics in the private and government sectors.

Bibliography:

[1] Wolf, Paul R., Elements of Photogrammetry, 2nd edition, McGraw-Hill, New York, New York, 1983

[2] Sonka, Milan, & Vaclav Hlavac & Roger Boyle, Image Processing, Analysis, and Machine Vision, 2nd edition, Brooks/Cole Publishing Co., Pacific Grove, CA, 1999

[3] Jackson, J.D., Classical Electrodynamics, 2nd edition, John Wiley & Sons, Inc., New York, New York, 1975

[4] Park, Steven & Zia-ur Rahman, "Lecture Notes on Digital Image Processing", Dept. of Computer Science, The College of William and Mary, Version 2.4, Fall 2000

[5] Tippet, James T., D.A. Berkowitz, L.C. Clapp, C.J. Koester & A. Vanderberg, Jr, Optical and Electro-Optical Information Processing, MIT Press, Cambridge, Massachusetts, 1965

Suggested reading:

[1] Horn, Paul & Berthold Klaus, Robot Vision, [The MIT Electrical Engineering and Computer Science Series], MIT Press, Cambridge, Massechusetts, 1986

Addendum:

The following is the actual code that was used in the project. This is in complete form and can be copied and used over. The right program format of input should be in PPM format and the output will be in PGM format. The suggested viewing should be in PaintShop Pro.

```
//**************************************************************
//  This is my attempt at edge detection. I'm sure that I can do
//  it.  So here it goes.
//
//  Author: Evin Grano
//  Name: edge.cc
//  Language: C++
//  Date: March 20, 2001
//**************************************************************

#include <iomanip.h>
#include <fstream.>
#include <string.h>
typedef char fileName[50];
struct pixel{
  int Red;
  int Blue;
  int Green;
  int edge;
};

int findMax3(int, int, int);
int gotNum(int, int, int, int, int);

int main(void){
  ifstream input;
  ofstream output;
  char tempChar;
  char ppmVersion[2];
  fileName ppmFile;
  fileName pgmFile;
  int  pixelCol;
  int  pixelRow;
  int  pixelScale;  int i, j, maxValue, weightSum, avePixel;
  long sumObjectPixels = 0;
  long sumColVal = 0;
  long sumRowVal = 0;

  cout << "Enter the PPM file to be analyzed: ";
  cin >> ppmFile;
  while ((tempChar != 'y') && (tempChar != 'Y')){
    cout << endl << "Is \"" << ppmFile
       << "\" the correct Filename (Y or N) ";
    cin >> tempChar;
    if ((tempChar == 'N') || (tempChar == 'n')) {
      cout << "Enter the PPM file to be analyzed: ";
      cin >> ppmFile;
    }
  }

  input.open(ppmFile);
  input >> ppmVersion;
  cout << ppmVersion;
  input >> tempChar;
  cout << tempChar;
  if (tempChar == '#'){
    while (tempChar != '\n'){
      input.get(tempChar);
      cout << tempChar;
    }
  }
  cout << "Here...";
  input >> pixelCol >> pixelRow >> pixelScale;
  cout << pixelScale;

  struct pixel picture[pixelCol][pixelRow];
  int holdPic[pixelCol][pixelRow];
cout << endl << "Processing";
  for (i = 0; i < pixelCol; i++){
    for (j = 0; j < pixelRow; j++){
      input >> picture[i][j].Red >> picture[i][j].Green >>
               picture[i][j].Blue;
      maxValue = findMax3(picture[i][j].Red, picture[i][j].Green,
                 picture[i][j].Blue);
      if (((maxValue-picture[i][j].Red)<=30)&&((maxValue
           -picture[i][j].Green)<=30)&&((maxValue
           -picture[i][j].Blue)<=30)){
         picture[i][j].edge = 255;
      } else {
         picture[i][j].edge = 1;
         sumObjectPixels++;
         sumColVal = sumColVal + i;
         sumRowVal = sumRowVal + j;
      }
      holdPic[i][j] = 255;
    }
  }
  input.close();
  cout << endl << "Finished! " << endl;

  /********This is where we clean the image a bit********************/
  for (i = 1; i < pixelCol-2; i++){
    for (j = 1; j < pixelRow-2; j++){
      weightSum = picture[i-1][j-1].edge;
      weightSum = weightSum + picture[i][j-1].edge;
      weightSum = weightSum + picture[i+1][j-1].edge;

      weightSum = weightSum + picture[i-1][j].edge;
      weightSum = weightSum + picture[i+1][j].edge;

      weightSum = weightSum + picture[i-1][j+1].edge;
      weightSum = weightSum + picture[i][j+1].edge;
      weightSum = weightSum + picture[i+1][j+1].edge;

      avePixel = weightSum/8;
      holdPic[i][j] = avePixel;
      //cout << avePixel << " ";
      }
 }
   /******************The output for the analysis***********/
  cout << "Enter the PGM file to be outputed: ";
  cin >> pgmFile;
  tempChar = '0';
  while ((tempChar != 'y') && (tempChar != 'Y')){
    cout << endl << "Is \"" << pgmFile
       << "\" the correct Filename (Y or N) ";
    cin >> tempChar;
    if ((tempChar == 'N') || (tempChar == 'n')) {
      cout << "Enter the PGM file to be outputted: ";
      cin >> pgmFile;
    }
  }
  output.open(pgmFile);
  output << "P2" << endl << "# This is created by Evin\n" << pixelCol
         << " " << pixelRow << endl << pixelScale << endl;
  for (i = 0; i < pixelCol; i++){
    for (j = 0; j < pixelRow; j++){
      output << holdPic[i][j] << " ";
    }
  }

  output.close();
  cout << endl << "Object center: " << sumColVal/sumObjectPixels << " "
       << sumRowVal/sumObjectPixels << endl;
  return(0);
}

int findMax3(int first, int second, int third){
  int max = 0;

  if (first >= second)
    max = first;
  else
    max = second;

  if (max >= third)
    return (max);
  else
    return (third);

  return (max);
}

int gotNum(int first, int second, int third, int fourth, int num) {
  int count = 0;

  if (first){count++;}
  if(second){count++;}
  if(third){count++;}
  if(fourth){count++;}

  if (count >= num)
    return(1);
  else
    return(0);
}

  return (max);
}
```