FORENSIC ANALYSIS OF DIGITAL IMAGE TAMPERING

THESIS

Jonathan R. Sturak

AFIT/GIA/ENG/04-01

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

AFIT/GIA/ENG/04-01

FORENSIC ANALYSIS OF DIGITAL IMAGE TAMPERING

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Information Assurance

Jonathan R. Sturak, B.S.

December 2004

AFIT/GIA/ENG/04-01

FORENSIC ANALYSIS OF DIGITAL IMAGE TAMPERING

Jonathan R. Sturak, B.S.

Approved:

1 DEC 04

Dr. Gilbert L. Peterson — date
Committee Chairman

Richard A Raines — 1 Dec 04

Dr. Richard A. Raines — date
Committee Member

Henry B. Potoczny — 1 Dec. 2004

Dr. Henry B. Potoczny — date
Committee Member

Table of Contents

Page

List of Figures

List of Tables

Page

AFIT/GIA/ENG/04-01

Abstract

The use of digital photography has increased over the past few years, a trend which opens the door for new and creative ways to forge images. The manipulation of images through forgery influences the perception an observer has of the depicted scene, potentially resulting in ill consequences if created with malicious intentions. This poses a need to verify the authenticity of images originating from unknown sources in absence of any prior digital watermarking or authentication technique. This research explores the holes left by existing research; specifically, the ability to detect image forgeries created using multiple image sources and specialized methods tailored to the popular JPEG image format. In an effort to meet these goals, this thesis presents four methods to detect image tampering based on fundamental image attributes common to any forgery. These include discrepancies in 1) lighting and 2) brightness levels, 3) underlying edge inconsistencies, and 4) anomalies in JPEG compression blocks. Overall, these methods proved encouraging in detecting image forgeries with an observed accuracy of 60% in a completely blind experiment containing a mixture of 15 authentic and forged images.

FORENSIC ANALYSIS OF DIGITAL IMAGE TAMPERING

Chapter 1 : Introduction

1.1 Research Introduction

The progression of the digital information age has evolved to replace technologies with state-of-the-art digital counterparts. The music and video display industries provide two examples of this evolution. Audio has progressed from analog audio tapes and records to Compact Discs and MP3s. Video displays have advanced from the analog Cathode Ray Tube (CRT) to the digital Liquid Crystal Display (LCD). The change of photography from requiring smelly chemicals and darkroom tricks to manipulate images has given way to the digital era. With the move to the world of Megapixels, a new door opens to the dark-side of image counterfeiting and forgeries. Gone are the days of needing to create "trick shots" with an analog camera or careful chemical preparation in the darkroom. Today, manipulating an image involves simply using tools available in the *digital* darkroom, such as Adobe Photoshop or Macromedia Fireworks. With these new techniques easily available to the masses via an inexpensive PC, the need exists to verify the authenticity of a digital image because of our increased reliance on digital media.

Two examples of the importance of digital image authentication are witnessed in the news media we rely on to provide accurate information and the courtroom where someone's fate may depend on the authenticity of a digital image as evidence. This thesis explores these issues with emphasis on creating tools to aid in the detection of digital image tampering for JPEG compressed images.

1.2 Background

A digital image is fundamentally composed of a series of “pixels,” a word derived from combining “picture” and “element” [20]. By coloring and brightening these individual pixels, a digital picture emerges. At face value, a digital image is nothing more than a slew of pixels set in some logical state. Three 8-bit numbers represent most color images with each octet corresponding to the amount of red, green, and blue a pixel embodies. A grayscale image typically contains a sole 8-bit number to signify the amount of gray in a pixel. In addition to the color depth an image contains, the number of pixels, or “resolution,” is an additional image attribute. Common notation for an image’s resolution is “M x N” where M represents the number of horizontal pixels and N represents the number of vertical pixels. [20] Common examples include “800 x 600” or “2048 x 1536”. The total number of pixels in a particular digital image is calculated by multiplying both horizontal and vertical numbers. With the example “2048 x 1536,” there are 3,145,728 total pixels representing this image. Accordingly, this would be the resolution of a digital image produced by a 3.2 MegaPixel digital camera.

While the color depth and number of pixels represent a digital image, images are further classified by the particular image format chosen to store the image. Common image formats include BMP, TIFF, and JPEG. Each has its own pros and cons when choosing to represent a digital image. The selection of one format over another depends on the particular application of the digital image. One must consider file size, application on the web, and image quality. Image formats such as BMP and TIFF use a lossless compression scheme. That is, they do not discard any information in the compression process, thus emphasizing quality over a smaller file size. However, the JPEG format uses lossy compression which sacrifices image quality for file size. Lossy compressed images discard pixels that should not overly degrade image quality based on a configurable Quality Factor. These four formats are common in the digital image community but by no means represent the entire range of digital image formats used in computing. Chapter 2 discusses some new compression formats recently announced and provides some insight into the underlying schemes used by digital image formats.

The attributes of a digital image, including color depth, resolution, and image format, form a basis for someone to perform manipulation to the perceived view from a digital image. This leads into a discussion about the problem that the research in this thesis attempts to investigate.

1.3 Problem Statement

Digital images provide a new way to represent pictures and scenes that only film and a darkroom could supply before. This new way to capture and store images opens a door to malicious individuals wishing to forge or otherwise manipulate original authentic images. Since digital photography is improving and becoming more widely used by the average photographer, a need exists to provide countermeasures against malicious forgers. The media that we rely on is an example of the increasing need to verify an image's authenticity. In the spring of 2004, several photographs emerged over media channels which depicted abuse against Iraqi detainees by several U.S. and British soldiers [13]. Much debate ensued concerning the authenticity of these photographs. In early May 2004 a British soldier was arrested for producing a forged photograph depicting detainee abuse, but not before a British tabloid newspaper ran the picture on the cover of one of its issues [4]. The old adage “don’t believe everything you hear” is becoming “don’t believe everything you see.”

The example in Figure 1.1 shows two digital images. The left image was printed by several news sources in an article about a mysterious giant-sized “hogzilla” [19]. While the authenticity of the image is unknown, with very little skill a “forged” version was digitally created using the computer software Adobe Photoshop. It is very hard, if impossible, for the human eye to detect digital manipulation at face value. This is just one example of the need for a tool to aid in the detection of digital image tampering. The research in this thesis attempts to address this need and provide some insight into this challenging problem.

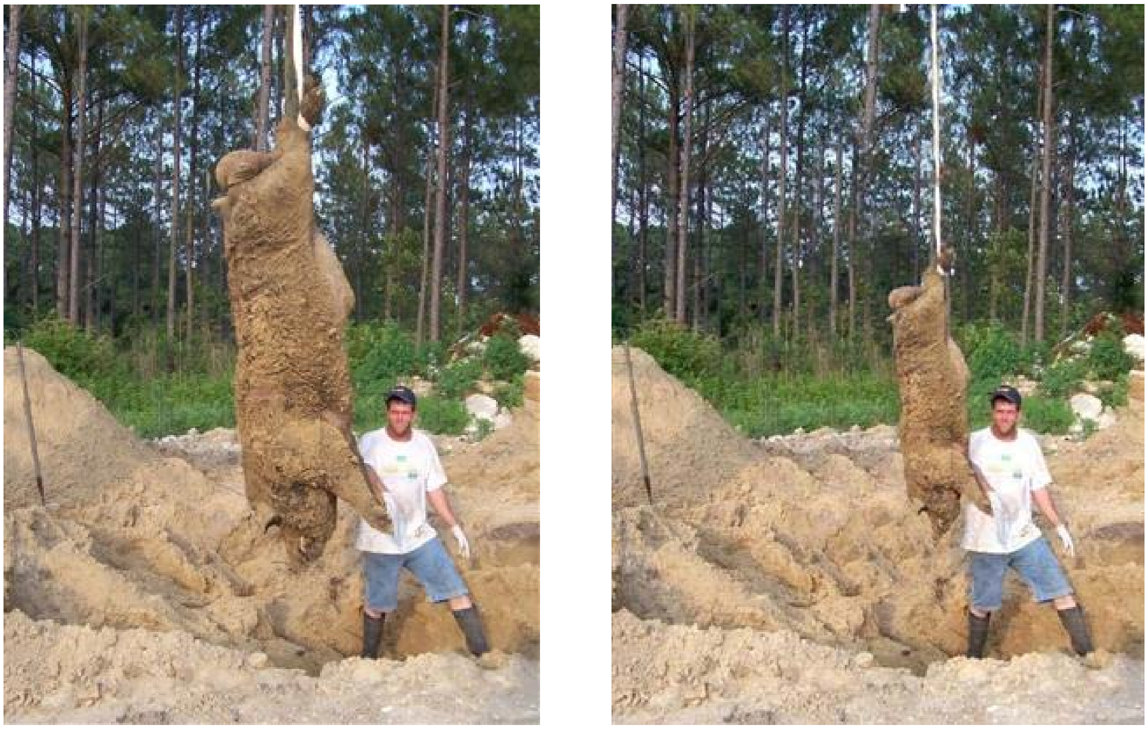


Image as Printed in San Jose Mercury News [19]

Digitally Manipulated Image

Figure 1.1 – Example of Digital Forgery

1.4 Research Focus

The focal point of this research is to survey the research community with respect to the detection of digital image forgeries. Additionally, this thesis extends the current state of the art with a new tool to detect image forgeries where previous methods fail. This area of image authentication is very broad due to the vast number of image formats, image resolutions, ways to create digital forgeries, and conceivable approaches to detect image tampering. As Chapter 2 discusses, researchers in the image processing community have developed several techniques to detect image forgeries [9] [12]. Much time and effort has gone into analyzing uncompressed images but current techniques return dismal success in detecting one of the most common digital image formats, JPEG [9]. With that in mind, the research presented here attempts to tailor methods toward the JPEG format as well as incorporate all image formats where possible. Many approaches exist in an effort to detect image tampering but the best place to start is to build upon the already known.

1.5 Research Approach

Digital images offer many attributes for a tamper detection algorithm to take advantage of, specifically the color and brightness of individual pixels as well as an image's resolution and format. These properties allow for analysis and comparison between the fundamentals of digital forgeries in an effort to develop an algorithm for detecting image tampering. This thesis focuses on images saved in the JPEG format, therefore a complete dissection of this compression scheme is discussed to determine what information can be gathered about a digital forgery saved in this format. Other fundamental properties of any digital forgery are used to develop additional detection techniques. This analysis will be the type of methodology used when conducting experiments in this thesis.

1.6 Summary

The digital age is among us and the evolution of digital photography is common place for photo gurus and the average photographer alike. With the increase in capturing and storing images in digital format, a new and uncharted door is open to the world of digital tampering. What took clever photography and extensive time in the darkroom can now be accomplished with the *digital* darkroom, consisting of a digital camera, a Personal Computer, and image manipulation software in seconds.

This thesis investigates image forgeries created digitally by surveying the current research performed in this area. The overall goal is to develop a new tamper detection tool which further extends the current methods and techniques available to a forensic analyst. Chapter 2 of this thesis includes a discussion of the current research community and presents prerequisite information for the design of tamper detection tools. This leads into Chapter 3, which discusses the methodology of new detection approaches as well as an experiment testing these newly proposed tamper detection methods. Finally, Chapter 4 presents the results of this experiment with Chapter 5 containing concluding remarks.

Chapter 2 : Literature Review

2.1 Introduction

This chapter introduces the techniques and methods currently available in the area of digital image forgery detection. A survey of the current research is presented as well as an analysis of the current techniques and methods available to detect image tampering. This area of research is relatively new and only a few sources exist that directly relate to the detection of image forgeries, therefore techniques are presented that apply to general digital image processing, but show promise in the detection of digital forgeries. Finally, image processing techniques are presented that will pave the way for Chapter 3, which deals with the methodology of an experimental design for image forgery detection.

2.2 Digital Watermarking

A discussion of image authentication techniques is not complete without first introducing the main method of proving image ownership, which is digital watermarking [7]. In digital watermarking, a desired image is combined with a watermark to form a watermarked image. This watermark may be *visible* or *invisible* to the naked eye. Figure 2.1 illustrates an example of *visible* watermarking. Here, a watermark is embedded into the host image, forming the watermarked image with a silhouette of the watermark clearly visible. This technique is useful when displaying a company logo or to show ownership of the image.

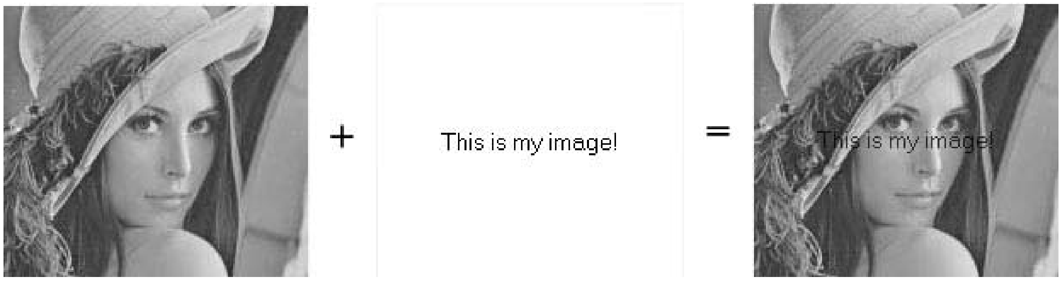


Figure 2.1 – Example of *visible* watermark using AiS Watermark Pictures Protector

A second form of watermarking exists in which the watermark is embedded but is "invisible" to the naked eye. This is useful for the author of the image to put his or her signature on it for security or anti-tamper reasons. Figure 2.2 shows an example of this.

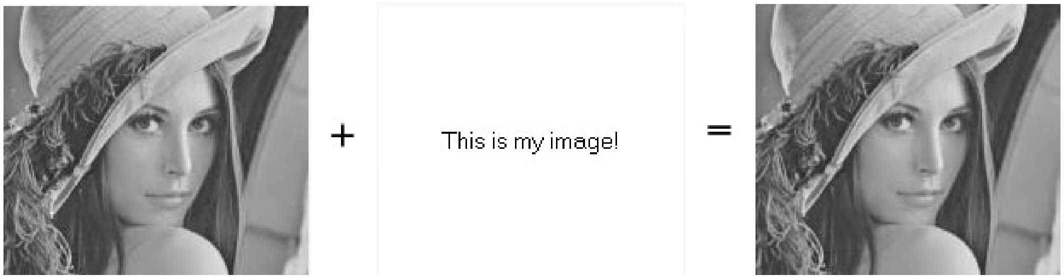


Figure 2.2 – Example of *invisible* watermark using Steganography Software *F5*

In this example, the original image and the watermarked image are visibility identical and the human eye generally can not see a difference. The existence of the watermark can usually only be determined using an extraction and detection algorithm that complements the embedding algorithm.

Digital watermarking applications used by the government, private industry, and for personal protection are ownership assertion, digital "fingerprinting," copy prevention or control, fraud and tamper detection, and ID card security [7]. *Invisible* watermarking also has some other benefits which take advantage of the fact that the watermark is not visible to the human eye. These include copyright protection and image tracking. The use of *invisible* watermarking helps guard against the increasing threat of passport fraud by embedding unique personal information into a government issued passport [7]. These areas of digital watermarking are increasingly important to implement in today's digital world, but the situation still exists in which an image's authenticity needs verification without relying on a watermarking scheme.

2.3 Unknown Image Origin

With techniques available to protect an original image from tampering, the reverse scenario raises concern of verifying the authenticity of an image of unknown origin. This is an increasingly important issue as digital cameras come down in price and ease of use of powerful image processing software, i.e. Adobe Photoshop and GIMP (GNU Image Manipulation Program), become more widely available [15]. In fact, GIMP is freely available on the web and is a viable alternative to Adobe Photoshop. Most of the image manipulations discussed in this thesis can be performed using GIMP. With increasing opportunities and ease to digitally manipulate images, the research community has its work cut out.

The state of the art in research in digital image forensics currently focuses on digital watermarking and variations of this, as previously discussed. Research conducted on image authentication in the absence of any digital watermarking scheme is still in its infancy stages [9] [12]. Therefore, this thesis explores this topic. Unknown origin images fall into 2 classes, *copy-move* & *copy-create*. The reason for distinguishing classes of image forgeries is because various image processing techniques exist that are better suited for each class as a whole.

The first class of image forgeries includes images tampered by means of copying one area within an image and pasting it onto another. A useful name for this class is *copy-move* forgeries. Figure 2.3 illustrates an example of this type. Here, copied parts of the foliage cover and mask the truck in such a way which completely masks it.

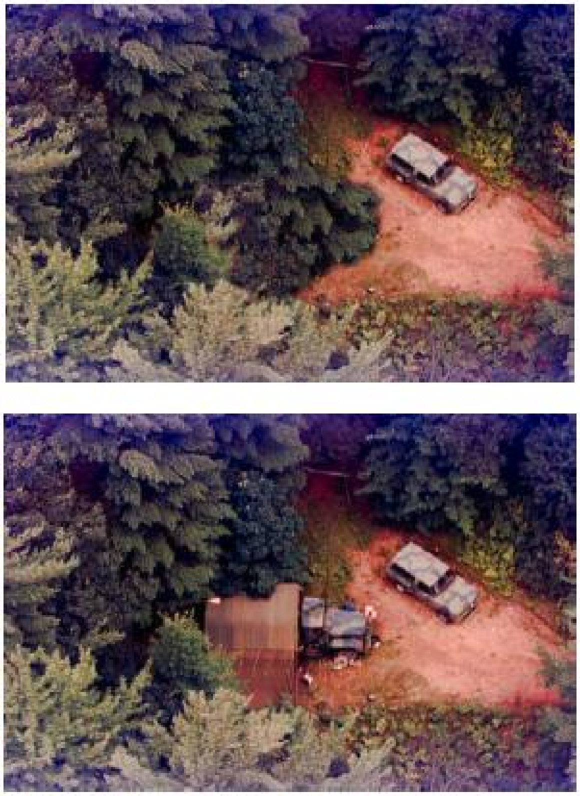


Figure 2.3 – Example of *copy-move* image forgery [12]

The second class of forged images deals with creating the forgery by using more than just the single image for copying and pasting. This is done by taking one or more images and copying and pasting from various areas within each to form a forged image. The image processing community formally refers to this type of image as an image “composition,” which is defined as the “digitally manipulated combination of at least two source images to produce an integrated result” [6]. The name for these types of images, in context of this thesis, is *copy-create* forgeries. Figure 2.4 shows an example of this.

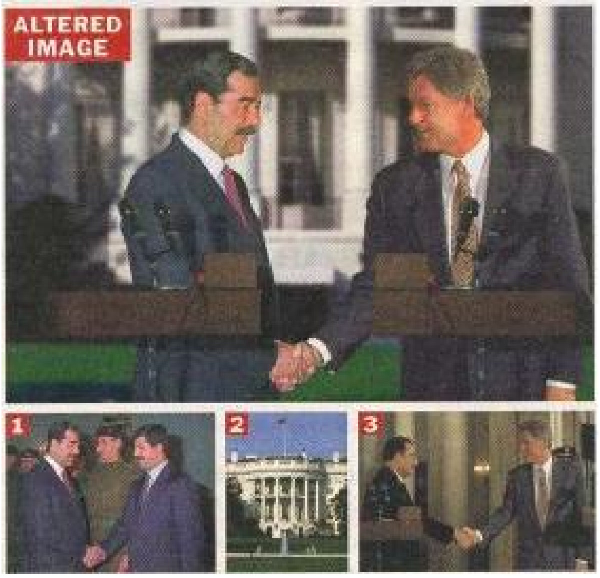


Figure 2.4 – Example of image forgery created from several sources [12]

In this example, 3 pictures are taken from various sources and merged together to form a forged image. Current image manipulation software can create forged images, such as this, by a person with moderate skill. Various techniques such as enlarging the White House and creating the podiums are used to strengthen the credibility of the image.

Forgeries can and usually contain various combinations of the above *copy-move* and *copy-create* techniques. Forgeries can also use image manipulation software to change the color or size of objects within the image to make it more believable. For example, an image forger makes use of the “smudge” tool to change the copied portion slightly. Features available in most digital toolkits, such as “airbrush” or “sketch/skew,” are applied to an image in order to change the color or orientation of its contents. Figure 2.5 illustrates an example of this.

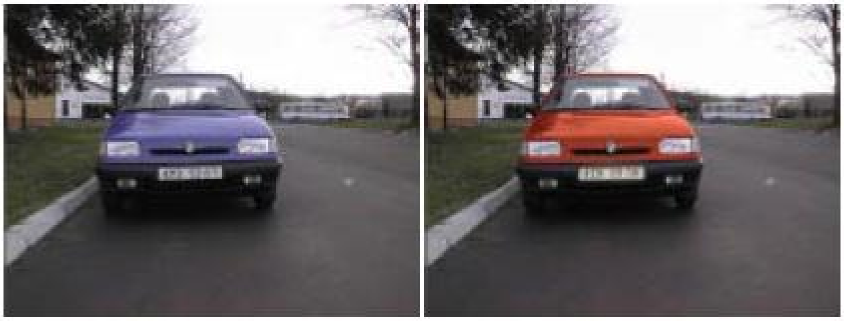

Original Forged

Figure 2.5 – Example of image forgery using image manipulation toolkit [17]

The original image here is the car on the left with blue paint. By using image manipulation software a forger uses the “fill” tool to modify the original image creating a red car instead.

The human eye attempts to detect image forgeries from these two classes by first determining if the scene depicted in the image portrays something believable. A person’s expectation of an image is sometimes the best detection method in determining if an image is forged. If an image appears real or comes from a reliable source, not much effort to determine its authenticity is usually exerted. However, if an image is suspected of tampering because it either came from an unreliable source or appears unnatural, its authenticity is scrutinized more. The human eye usually picks up on c*opy-create* forgeries easily. This is because this type of forgery consists of several images, each of which may have different lighting, color patterns, quality, or shadows. In general, the eye first attempts to scan the image for these anomalies when determining if the image appears to be forged. On the other hand, the human eye usually has much more trouble detecting *copy-move* forgeries. This is because the forged area consists of parts from within the same image, thus containing consistent lighting and color patterns. Again, the human eye attempts to look for abnormal areas in the image that appear tampered. With these observable facts, a computer aided by various image processing techniques is the best approach to aid an investigator in detecting digital image tampering.

2.4.1 Edge Detection using first-order operators

Edge detection algorithms, a classical image processing technique, have been analyzed against a number of forged test images [17]. Lukas analyzed these first since edge detection algorithms are a fundamental application to image processing. The edges of an image are extremely significant in many applications since they provide information about the location of objects and their texture, size, and shape. This concept is of interest in forgery detection because image tampering introduces hidden anomalies often associated with a double edge around the tampered objects. This phenomenon occurs because the blurring of space around the tampered objects, in conjunction with the actual edge, forms a double or “ghost” edge.

An edge is defined as areas in the image where the intensity of pixels moves from a low value to a high value or vice versa [18]. This leads into an analysis of first-order operators and their power at detecting discontinuities. First-order operators detect points in the image that are discontinuous by calculating a function of the image which uses first-order derivatives. There are various convolution masks used in image processing and some have already been used to analyze forged digital images. Previous images were analyzed using the Roberts, Sobel, and Prewitt masks [17]. The Sobel mask is more receptive to edges that are diagonal in nature rather than horizontal or vertical. The Roberts mask is more susceptible to noise than the other masks while Prewitt is better at horizontal and vertical edges. [18]

The following formula computes the convolution of an image [17]:

$$h_{x,y} = \sum_{i=d}^{n} \sum_{j=d}^{n} g_{i,j} \, f_{x+i,y+j},$$

where $d = \frac{-(s-1)}{2}$ and $n = \frac{s}{2}$, g is a convolution mask of size s x s, and f is the image function.

The following are the masks described above and used for the variable *g*.

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

Roberts' mask

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Sobel mask

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Prewitt mask

2.4.2 Edge Detection using second-order operators

First-order operators are a good fundamental technique to use in image processing and forgery detection, but second-order operators offer a distinct approach in the detection of image forgeries. Second-order operators provide an alternative method at detecting what is considered an edge, which allows for more robustness. This is true because second-order operators provide much better edge localization based on how they calculate the edge. Instead of calculating an edge several pixels wide, and thus posing the problem of determining the center of an edge, second-order operators attempt to guard against this [18]. Second-order operators use Laplacian and Gaussian functions to calculate the convolutions of the image in question. These techniques are robust against various image degradations, i.e. noise, because of the Gaussian function [17]. Marr and Hildreth posed this technique which looks for zero-crossings after convolution with the Laplacian and the Gaussian functions. The Marr edge detector first performs Gaussian smoothing before convolving the image with the Laplacian function [18].

An example of a Marr edge detector of order *5 x 5* is given below [17]:

$$\begin{bmatrix} 0 & 3 & 6 & 3 & 0 \\ 3 & 15 & 0 & 15 & 3 \\ 6 & 0 & -108 & 0 & 6 \\ 3 & 15 & 0 & 15 & 3 \\ 0 & 3 & 6 & 3 & 0 \end{bmatrix}$$

Marr edge detector of order 5 x 5

This mask provides symmetry both horizontally and vertically. This is due to the symmetry of the Gaussian function which enables equal balance across portions of the image being filtered. The power of edge detection permits the possibility of detecting hidden discontinuities, which might be prevalent in image forgeries [17]. The Marr edge detector follows similar symmetry for larger size matrices of higher order. The next subsection presents a different, but equally interesting, image processing approach dealing with frequency analysis.

2.4.3 Spectral Analysis

Spectral analysis approaches utilize the power of Discrete Fourier Transforms (DFTs) and their ability to detect brightness and intensity levels of an image. The following formula is used to compute the DFT of a given image [17]:

$$F_{x,y} = \frac{1}{\sqrt{MN}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f_{m,n} e^{-2\pi i(\frac{xm}{M}+\frac{y\mathrm{n}}{N})},$$

where f is the image of size M x N represented as a brightness function of each pixel.

Lukas analyzed some preliminary test images using the power of DFTs [17]. This technique allows one to see areas of the image that may be manipulated, by looking for the natural logarithm of the amplitude in high frequencies of the image. Since a digital image can be treated as a two dimensional signal, tampering with an area of an image introduces anomalies in the frequency of this signal. If a local maximum in the high frequency range is present when performing spectral analysis, the image may be a victim of an image forgery. [17]

Farid and Popescu extend Lukas's spectral analysis approach by presenting an encouraging process which detects image forgeries based on the observed effects of re-sampling an image [9]. Their respective method differs from Lukas' in that it concentrates on pre-processing and filtering the image in an attempt to gain high detection accuracy. Fully analyzing the forgery process and its effect on the victim image enabled Farid and Popescu to develop a fully customizable method.

Forged images that are the result of merging two or more host images together usually requires that at least one image be cropped, resized, or rescaled. This manipulation leads to underlying changes in the statistical nature of the image, which spectral analysis captures. By calculating the Fourier transform of suspected areas of manipulation in the image, one looks for a periodic pattern suggesting that an area has been re-sampled. [9] To further explain this technique as well as the expected results, the following figures provide an example. Figures 2.6 and 2.7 show an image of a car's license plate illustrating both an authentic and forged version.

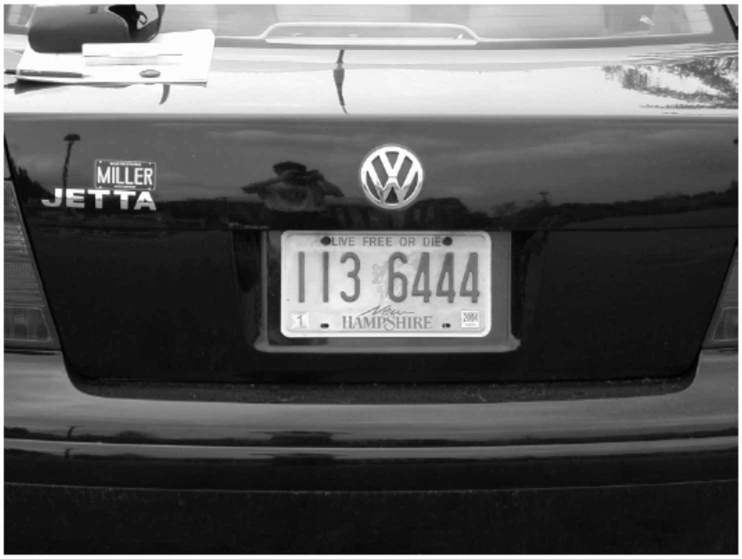


Figure 2.6 – Original Image of a car's license plate [9]

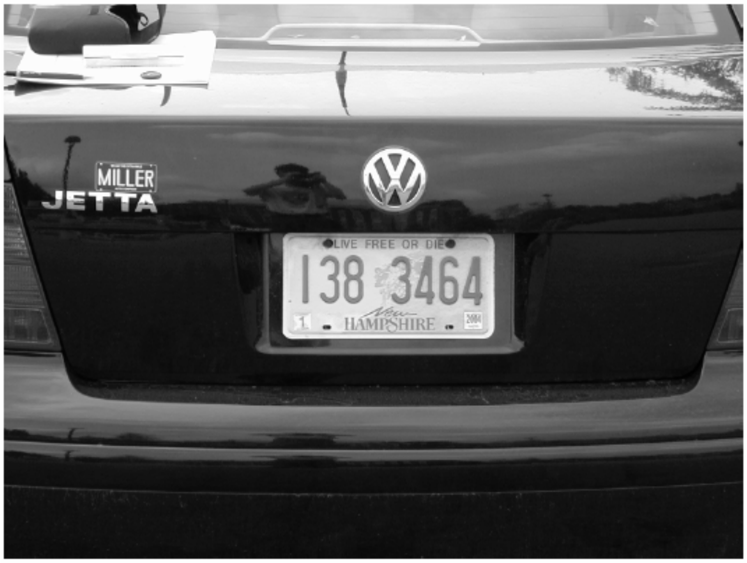


Figure 2.7 – Forged Image of a car’s license plate [9]

Figure 2.8 represents a high-pass filtered “probability map” of the forgery (Figure 2.7). A probability algorithm using Bayes’ rule produces this version of the image. In this algorithm, a correlation between pixel neighbors is estimated against several periodic samples. The motivation for the use of this map in Fourier transform analysis is that it removes the low frequency information from the image which may return false positives. In the forgery detection algorithm, areas of this probability map are blocked off and used for comparison. One blocked area should encompass the suspected tampered portion and the other blocked area should cover an assumed authentic region. [9] Figure 2.8 depicts this probability map as well as the two blocked areas used for analysis, with the license plate suspected of tampering.

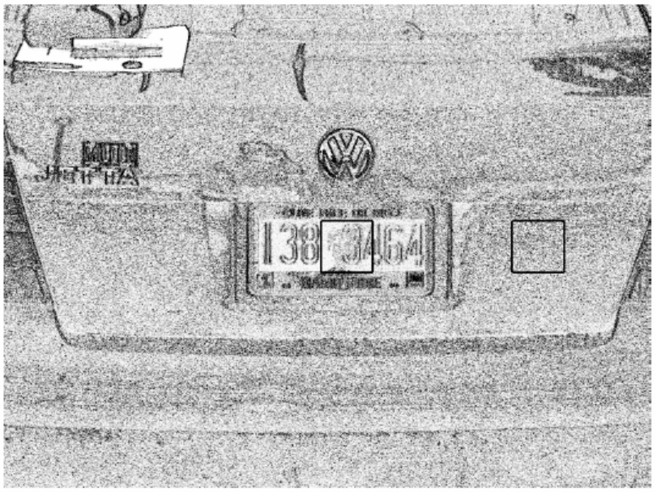
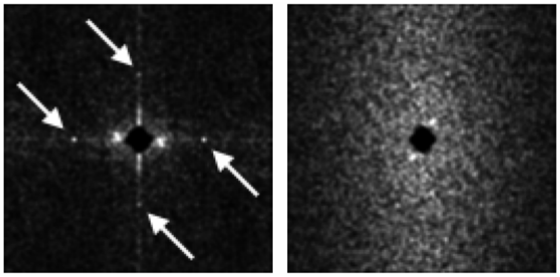


Figure 2.8 – Probability map of Figure 2.7 [9]

The following Figure 2.9 shows the results of the Fourier transform when performed on the areas blocked off in Figure 2.8. The left portion, which is that of the tampered license plate, shows a periodic pattern (spikes in the Fourier transform) while the right portion, which is an authentic part of the trunk, yields an evenly distributed result. The periodic pattern shown in the license plate suggests that the area has been re-sampled and thus been tampered with.



Forged Area Authentic Area

Figure 2.9 – Result of Fourier transform on blocked areas of Figure 2.8 [9]

This technique using Fourier transform analysis has been found to work best on uncompressed images, i.e. TIFF. Images saved in the lossy JPEG format exhibit much lower detection accuracy with Quality Factors of 97/100 and lower. When a JPEG image has been saved using a Quality Factor of 90/100 or lower, detection becomes an extremely hit or miss occurrence. The introduction of noise and the periodic block pattern of the JPEG compression algorithm are the suspected reasons for this difficulty. [9] As the Quality Factor goes down, the above two observable facts increase, thus causing Fourier transform analysis to become less reliable. It should be noted that most JPEG images are set to a Quality Factor of approximately 80/100 for optimal high quality, with medium to low quality images using much lower Quality Factors. Section 2.6.4 explores the JPEG compression scheme and the proposed methods of tamper detection in JPEG images.

It is worth discussing other spectral analysis techniques dealing with signal and image processing, namely the Wavelet Transform. Unlike the Cosine and Fourier Domain, Wavelets encompass both frequency and time information of a signal. The Sine wave, which is the basis of Fourier analysis, and the Cosine wave both exhibit a smooth and predictable pattern, while Wavelet analysis breaks up the original signal into a scaled and shifted version focusing on trends and peaks in the signal. This uniqueness allows for an alternative method to examine signals. [3]

While spectral analysis techniques, in general, exhibit distinctive power at breaking down and analyzing images, which are nothing more than two dimensional signals, they do have limitations in detecting image forgeries. These include only having high detection accuracy on uncompressed images while exhibiting poor detection precision on compressed images (i.e. JPEG) with minimal compression [9]. Section 2.5 further discusses the correctness of an example using spectral analysis techniques.

Wavelets are also used to form new compression schemes for digital images. While the JPEG standard, using Discrete Cosine Transforms (DCTs), is the most popular and widely used format on the web and by digital cameras [2], the Discrete Wavelet Transform (DWT) is currently being researched and forms the basis for the JPEG2000 format. DWT compression in digital images provides a new and unique approach at obtaining images with smaller files sizes and at the same time having better quality. While the International Standards Organization has finalized the JPEG2000 DWT format in late December 1999 [14], it is not widely supported in web browsers, digital cameras, and image manipulation software [1]. The JPEG DCT standard is still the most widely used and supported medium for digital images [2].

2.4.4 Exhaustive Search for detection of *copy-move* images

When analyzing an image to determine if an area has been copied and pasted onto another, an intuitive suggestion is to perform an exhaustive search to determine a pattern or like areas. Fridrich discussed and analyzed this technique [12]. Overlaying each circularly shifted position of the grayscale converted image and comparing it with the original yields the areas copied and pasted, albeit computationally slow. Figure 2.10 illustrates one case of a test image and a circular shift.

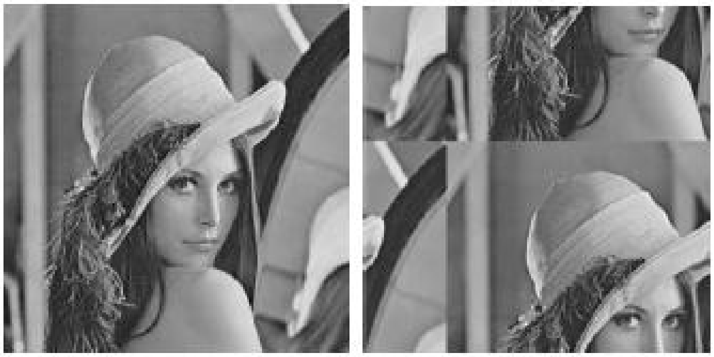


Figure 2.10 – Test image and a circularly shifted case [12]

Using the exhaustive search technique the following differences are calculated [12]:

$$\left|x_{i\,j} - x_{i+k} \operatorname{mod}(M)\, j + l \operatorname{mod}(N)\right|, k = 0,1,\ldots, M-1, l = 0,1,\ldots, N-1 \text{ for all } i \text{ and } j.$$

where x$_{i\,j}$ is the pixel grayscale image of size *M x N* at position *i*,*j*. While it is possible to cut down computational complexity by a factor of 4 due to redundancy of the shifted and original image, it is still too taxing to implement in reality as computational complexity rapidly increases with image size.

2.4.5 Block matching of *copy-move* images

Section 2.4.4 discusses a technique that employs a brute force exhaustive search to determine similar areas. Although computationally complex, this technique shows promise at detecting copied areas. A variation of this uses a *B x B* block of pixels, which represents the minimal size considered for a match [12]. This block is first placed in the upper left hand corner and moves one pixel at a time right and then down. It continues until the *B x B* block reaches the bottom right corner. There exists a total of (*M*-*B*+1)(*N*-*B*+1) positions for the block. The pixels are extracted by columns in each block position and placed into a matrix. The matrix will have B^2 columns and (M-B+1)(N-B+1) rows. The matrix is then searched with respect to rows that are the same but correspond to different areas of the image, thus suggesting that the portion of the image has been copied from one location to the other. This technique follows a running time proportional to the desired size of the *B x B* block. The block size also dictates the desired accuracy of the image in question. This technique is encouraging at detecting *copy-move* forgeries, but when looking at JPEG images one must realize that because of the lossy compression much of the exact matches will no longer be present. A BMP or TIFF image would be suitable for this technique, but the method presented in the next section is more favorable for a larger group of image formats.

2.4.6 Robust matching of *copy-move* images

When an image is saved in the JPEG format, the exact matching technique loses its power. In order to utilize the block matching technique, blocks are matched based on their representation consisting of quantized Discrete Cosine Transform (DCT) coefficients [12]. A user specifies a value, *Q*, to determine the sensitivity of the block match. This is equivalent to the Quality Factor of the JPEG compression. In this method, the same technique is used which creates a matrix from *B x B* blocks. The difference being the storage of computed DCT coefficients instead of pixel values. Rows in the matrix are then sorted and if a match occurs the position of each is recorded. To cut down on similarities between close blocks, a shift-vector counter, *C*, is calculated. First the shift vector, *s*, must be calculated. If two position match, (i_1, j_1) and (i_2, j_2), their relative location is determined and stored in *s*. The following equation represents this idea:

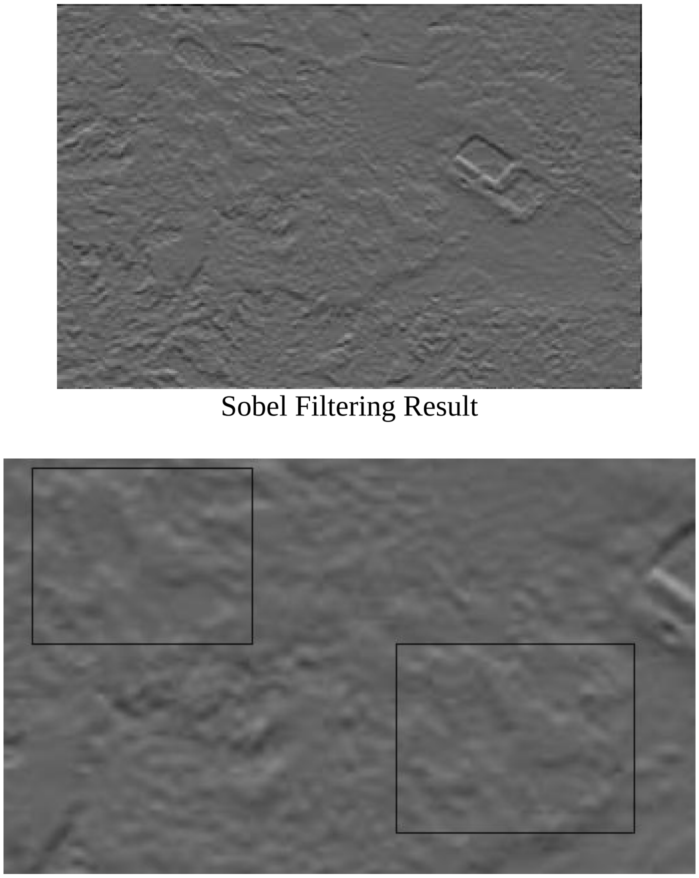
$$s = (s_1, s_2) = (i_1 - j_1, i_2 - j_2)$$

This pair must be normalized because –*s* and *s* both characterize the same shift. Once *s* is calculated, a counter, $C(s_1,s_2)$, is kept to keep track of appropriate locations in the image suited as a *copy-move* candidate. This value starts at zero and is incremented whenever a block match occurs. After going through the sorted rows of the matrix, the result is a set of shift vectors, $\mathcal{S}$. Based on a user-defined threshold, *T*, the areas in the image that correspond to the shift vectors within *T* depict a result of where copied portions may exist. This threshold, *T,* is set to allow for more fine detail in the determination of a *copy-move* segment, although a possibility exists in detecting more false-positives. [12]

2.5 Correctness and Performance of the presented detection methods

This section discusses the results of using the various forgery detection techniques presented in Sections 2.4.1 – 2.4.6 on the forged image displayed in Figure 2.3. These methods are the current state of the art in image forgery detection and show the progress of the research community in recent years.

The first technique analyzed in this chapter was image convolution masks discussed in Sections 2.4.1 and 2.4.2. First-order and second-order operators form the basis of image edge detection, which is a fundamental image processing task. Figure 2.11 presents the results of performing the Sobel convolution mask on the forged Figure 2.3. The tampered portion, in this example, has been magnified for better analysis. While a similar pattern arises in this magnified portion, as witnessed in the blocked regions, no firm conclusion signifies that image tampering has occurred. “Off the shelf” convolution masks are not ideal to detect image tampering because they lack the ability to make a solid conclusion in regard to whether an image has been tampered with. They may be good to use in extending other more conclusive methods, but the several test images analyzed by Lukas [17], as well as the example given in Figure 2.11, show the shortcomings of standard convolution masks. It is wise to look next at the results of methods based on spectral analysis.

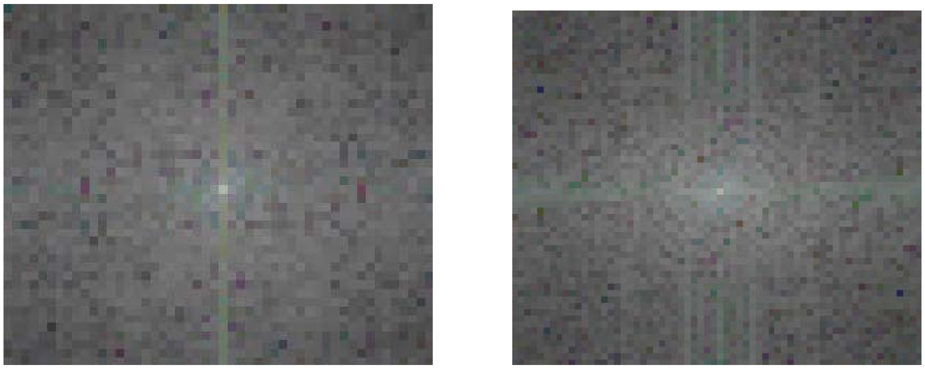


Sobel Filtering Result

Magnification of forged area with copied areas marked

Figure 2.11 – Sobel convolution filtering of forged Figure 2.3

Section 2.4.3 discusses the use of spectral analysis in determining if an image is forged. To keep with the consistency of the results, Figure 2.12 includes the resulting magnitude blocks when taking the Fourier transform of the forged area and an assumed authentic area of Figure 2.3. The results here show that there exists more variation in magnitude when analyzing the tampered portion, but nothing that raises a definitive red flag. This image was small in file size and resolution and thus results in an increase in Fourier transform magnitude based on this fact alone. The color or brightness pattern of the tested area could also affect the results of this method. Since the spectral analysis technique discussed in Section 2.4.3 requires an image in an uncompressed format with a high resolution to be highly accurate [9], this technique returns dismal results when used to analyze the forgery in Figure 2.3.



Authentic Area

Forged Area

Figure 2.12 – Fourier transform method applied to forged Figure 2.3

Section 2.4.4 presents an exhaustive search method to detect image tampering commonly found in *copy-move* image forgeries. This technique, in concept, should be successful at finding the tampered portion. Fridrich performed a test of this method and had good results albeit very long computational time [12]. This fact ultimately caused this method to be abandoned and thus not analyzed here.

In analyzing the proposed methods to detect *copy-move* images, specifically from Sections 2.4.5 and 2.4.6, the Exact and Robust Matching Techniques were found to be very promising [12]. Figure 2.13 shows the results of performing the Exact Matching Technique on Figure 2.3. The suspect areas that have been determined to be matching are colored white, while all other areas are colored black. It appears that an image manipulation tool, such as “smudge”, has been used to try to hide the manipulator’s tracks. As stated before, this technique would only be good for image formats that do not use randomized lossy compression. The popular and widely used JPEG format would fail when using the Exact Matching technique. The next paragraph covers the correctness and accuracy of the Robust Matching Technique.

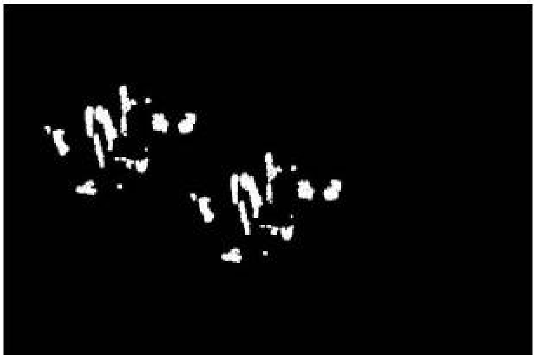


Figure 2.13 – Exact Match technique of Figure 2.3 using *B* = 4 [12]

The results of using the Robust Matching Technique are very promising with regard to the few test images analyzed [12]. Similar to the Exact Match Technique, the areas determined to be duplicate copies are shaded with a color that corresponds to the different shift vectors. Everything else not matched is colored black. Figure 2.14 shows the results of performing the Robust Matching Technique on Figure 2.3. As with the Exact Match Technique, the results are very promising.

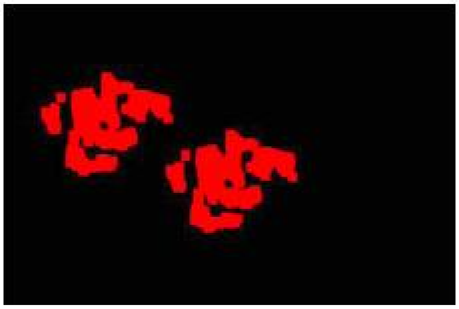


Figure 2.14 – Robust Match technique of Figure 2.3 using *B* = 16 [12]

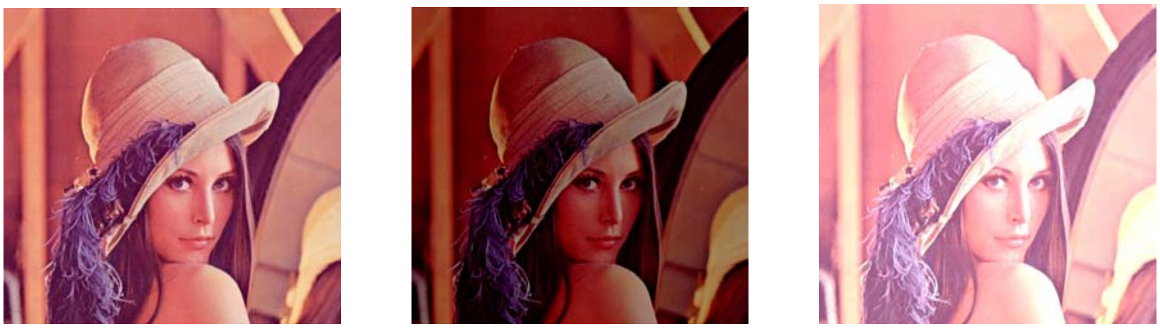
Overall, the Robust Match Technique is worthy of much praise in detecting *copy-move* image forgeries. While several of the test images exhibited small areas of false positives, it is still an excellent technique to use as a baseline in the detection of *copy-move* forgeries. A false positive is common on flat backgrounds that contain very similar color and texture patterns, such as the sky. Therefore, human examination is obviously necessary to interpret the results of any algorithm designed to detect image forgeries [12]. The methods presented here focus mainly on the detection of *copy-move* forgeries saved in any image format as well as *copy-create* forgeries saved in uncompressed formats, i.e. TIFF. Much work still needs to be performed with respect to *copy-create* forgeries saved in the very common and widely used JPEG image format.

2.6 Other Image Processing Techniques to Investigate

Section 2.5 discussed previously proposed forgery detection methods and their correctness at detecting various types of image forgeries. Several other methods in image processing should be further investigated to determine which are better suited at detecting image tampering. These methods include an analysis of the Luminance and HSV (Hue-Saturation-Value**)** intensity levels of an image. Also, various custom filtering masks should be investigated to capture their flexibility in filtering an image using customizable parameters. Finally, in-depth analysis of the JPEG compression algorithm is a viable research path since it is the foundation of detecting “hidden” information about an image not easily detected by the human eye.

2.6.1 Detection of tampering based on analysis of Luminance levels

The luminance of an image is the measurement of the perceived brightness levels [20]. Intuitively, if two images are taken from different cameras with different lighting, some sort of discrepancy may occur in those areas which were copied and pasted. In particular, analyzing areas in a forged image which are approximately the same distance away from the lens but have different luminance levels. This analysis is heavily dependant on the skill level of the person creating the forgery and the resources available to perform the manipulation. Newer versions of image processing software make it easy for even a novice user to create forgeries based on automated “auto-brightness” adjustments. Figure 2.15(b,c) shows the original test image in Figure 2.15(a) with luminance levels at both extremes on the scale. The 2.15(b) image has a low level of luminance while the 2.15(c) image has a much higher level.



(a) (b) (c)

Figure 2.15 – Example of changes in luminance levels

2.6.2 Detection of tampering based on Hue-Saturation-Value (HSV) levels

As in the previous section dealing with the luminance of an image, an analysis method based on the Hue-Saturation-Value (HSV) levels of an image follows. The Hue of a color is best described as the “tint” [20]. Saturation or “shade” is defined as the level of how pure or intense a color is [20]. Value is the level of brightness (luminance) of a color or how light or dark it is [20]. Intuitively, if an area or areas throughout an image are copied and pasted from different sources, the color and brightness, as captured from each respective camera, may be slightly different. Thorough analysis of HSV levels helps to determine this. Figure 2.16 shows an example of changes in HSV levels of Figure 2.15(a).

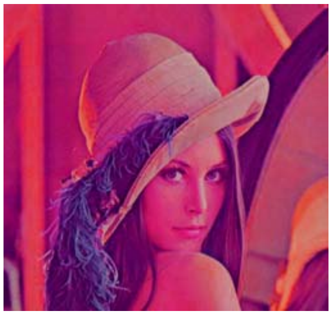


Figure 2.16 – Example of a change in HSV levels to Figure 2.16(a)

2.6.3 Detection of tampering based on alternative filtering masks

As discussed in Section 2.4.1, Lukas has looked at several edge detectors based on the Sobel and Prewitt masks. This method of filtering an image is formally classified as pixel-group or spatial domain filtering. While one reason to perform this type of processing is to detect edges, as presented in Section 2.4.1, other interesting information can also be gathered from an image, such as the low or high pass filtered version. These filtering methods give an alternative way to view an image and therefore may uncover small anomalies introduced from image tampering. The power to create customized masks may prove to be of some interest in detecting image forgeries, or providing further validation that one has occurred.

Spatial domain filtering deals with calculating a pixel value based upon its surrounding pixels. This type of "pixel group" processing provides a way to show trends in an image, such as brightness levels across particular areas [5]. In the *3 x 3* case, every pixel is evaluated with its eight neighboring ones. Below is an abstract representation of each pixel and its eight neighbors:

$$\begin{matrix} (x_{i\text{-}1,j\text{-}1}) & (x_{i\text{-}1,j}) & (x_{i\text{-}1,j+1}) \\ (x_{i,j\text{-}1}) & (x_{i,j}) & (x_{i,j+1}) \\ (x_{i+1,j\text{-}1}) & (x_{i+1,j}) & (x_{i+1,j+1}) \end{matrix}$$

where $x_{i,j}$ is the pixel at location ***i,j*** in image *X* and the rest of the letters represent $x_{i,j}$'s eight neighbors. The integer values of each pixel are extracted and manipulated with a *convolution kernel*. Formally, the values obtained from pixel $x_{i,j}$ and its eight neighbors are multiplied by their respective convolution kernel coefficients and then the summation over all nine is taken. Finally, this value is then divided by the total number of elements summed. This returned number is now the new value for the pixel $x_{i,j}$. This same technique is applied to every pixel in the image, with all pixels eventually assuming the representation $x_{i,j}$. Care is taken at the image boundaries to only use those pixels that would fall within the image. Below is a depiction of the convolution kernel, which maintains consistency throughout the entire filtering process:

$$\begin{matrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{matrix}$$

The following is a representation of the summation of pixels $x_{i-1,j-1}$ through $x_{i+1,j+1}$ with the respective convolution kernel:

$$\textit{Output pixel } x_{i,j} = [\,(\ x_{i-1,j-1}(k_{11}) + x_{i-1,j}(k_{12}) + x_{i-1,j+1}(k_{13}) + x_{i,j-1}(k_{21}) +$$

$$x_{i,j}(k_{22}) + x_{i,j+1}(k_{23}) + x_{i+1,j-1}(k_{31}) + x_{i+1,j}(k_{32}) + x_{i+1,j+1}(k_{33})\ )\ /\ 9\,]$$

Intuitively, the result of the above operation emphasizes the trends in an image, particularly abrupt pixel variability as witnessed in edges and, more importantly, tampered areas. This is because a pixel's eight neighbors is averaged and used to determine its new value. Conversely, with processing the whole image together, effectively a block size equal to the size of the complete image, the power to see any trends or suspicious areas may be lost. This is due to the weighted average approach used by spatial domain processing [5]. Block Based Processing with a relative block size to a single pixel could lend clues or provide further justification that a particular area in question is victim to image manipulation.

2.6.4 Detection of tampering based on the JPEG compression scheme

When an image is broken into sub-parts or equal squares to perform processing, the classification “Block Based Processing” is warranted. This technique is similar to that described in section 2.6.3, but the difference is that each block is considered a separate sub-image. This method is analogous to a recursive type process, with the sub-processing resembling a “divide and conquer” approach. Block Based Processing is useful because the calculations performed are influenced by only the information present in that particular block.

Block Based Processing is important in image processing, specifically image compression. The compression standard set forth by the International Standards Organization (ISO) and International Electro-Technical Commission (IEC) of Joint Photographic Expert Group (JPEG) images uses a Discrete Cosine Transform (DCT) scheme [21]. The DCT domain is used to convert a signal into coefficient values with the ability to perform truncating and rounding operations, thus allowing compression of this signal to take place. The JPEG compression process starts by calculating the DCT of each unique *8 x 8* blocks, B_{kl} , in the image based on the following formula [11]:

$$D_{ij} = \sum_{k,l=0}^{7} a_{kl}(i, j)B_{kl},$$

where $a_{kl}(i, j) = \frac{1}{4} w(k)w(l)\cos\frac{\pi}{16}k(2i+1)\cos\frac{\pi}{16}k(2j+1)$ and $w(k) = \frac{1}{\sqrt{2}}$ for $k = 0$

and $w(k) = 1$ otherwise.

Matrix ***D***, which contains 64 DCT coefficients, is then quantized using a quantization matrix ***Q*** [11]:

$$D_{ij} = round\left(\frac{D_{ij}}{Q_{ij}}\right), i, j \in \{0,1,2,3,4,5,6,7\}$$

The quantized coefficients, D_{ij}, are then arranged in a zigzag order, encoded using the Huffman Algorithm, and inserted into what makes up the JPEG file [11]. Decomposition works similarly just in reverse order. By rounding the ratio above, an integer value is obtained and thus allows an image to be compressed. A threshold is set to determine what integer values should effectively be discarded. The parts to be discarded are carefully calculated based on a "Quality Factor", which is a reference number between 0 and 100 [8]. The higher the Quality Factor, the less compressed and the better quality the image is. A trade-off between file size and image quality is always necessary in this type of lossy compression.

A JPEG image can either be color or grayscale. The above operations encode pixel values that are usually in the 0 to 255 range (8-bit). In the case of grayscale images, a sole 8-bit number represents the level of gray in each pixel. Color images use similar boundaries but include three 8-bit numbers, one for the Red, Green, and Blue channels. This allows for the creation of a 24-bit color image. [20] The analysis in this section works for all types of JPEG images and the various forensics approaches apply regardless of the color type.

Whenever an image is heavily compressed using the JPEG scheme, a distinct phenomenon occurs. The *8 x 8* blocks, resulting from the DCT function and subsequent information loss, become easily noticeable. Figure 2.17 shows an example of this occurrence. Here, the image in Figure 2.15(a) is compressed very heavily and then enlarged to show the effects of the compression. The blocks are easily distinguishable in this image and show the effects of DCT compression.

With a somewhat predictable scheme used by the JPEG compression algorithm, the analysis of an image with respect to this scheme may show promise in detecting image tampering. JPEG compression forms a type of "fingerprint" that may indicate alteration.

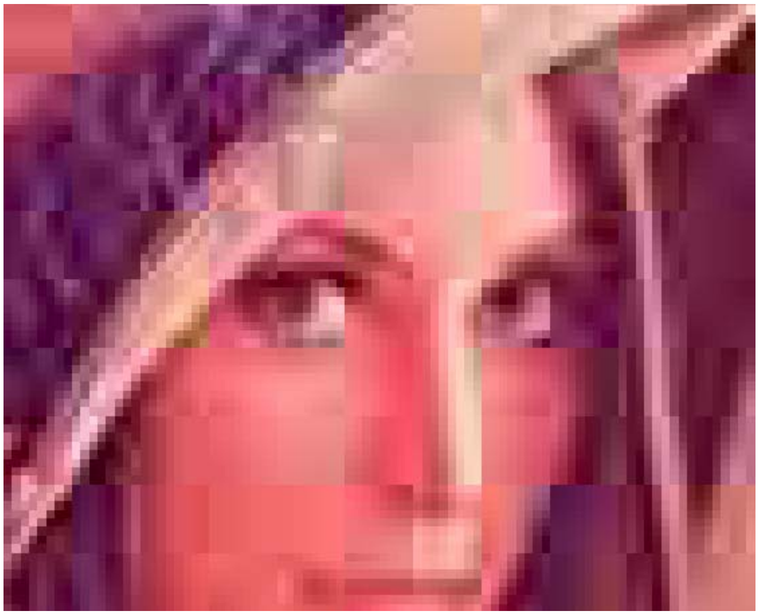


Figure 2.17 – Magnified Portion of Figure 2.15(a) after Heavy Compression

If two images are used to create a forgery, it is likely that both have different levels of compression, specifically the "Quality Factor" discussed previously may be different in both cases. Also, it is likely that resizing, rotating, or cropping was performed on the tampered portion to ensure it blends in with the rest of the image. Therefore, the compression algorithm may leave behind some possible clues. Figure 2.18 depicts an example of the above conjecture. Here, the higher compressed (QF = 5) image from Figure 2.17 (Image A) and the better quality (QF > 70) original from Figure 2.15(a) (Image B) are merged together to form forged Image C. This manipulation was accomplished by simply performing a copy and paste operation. Image A was positioned accurately over Image B, as displayed in the circled area, and then returned to the normal magnification. The result at normal magnification is almost indistinguishable to the human eye. The different levels of compression present should be noted, specifically that seen in the woman's eyes. Her left eye was part of Image B, while the copied portion, Image A, contains her right eye at much lower quality. When looking at the resulting Image C, one would not think anything is suspicious unless prior knowledge of tampering was known. This simple simulation shows the power of attempting to do an analysis of the compression levels used in a JPEG image.

A technique has been previously used in determining if a BMP image in raw format, i.e. one without any compression, has been previously JPEG compressed [8]. By breaking up an image into disjoint *8 x 8* blocks, analysis can be performed to determine if a "fingerprint" exists that will signify that the image has, in fact, been previously JPEG compressed. An intuitive approach is to calculate sample differences from within a block and again at the blocks boundaries [8]. Figure 2.19 shows an abstract representation of an *8 x 8* block with the pixel values marked used in calculations.

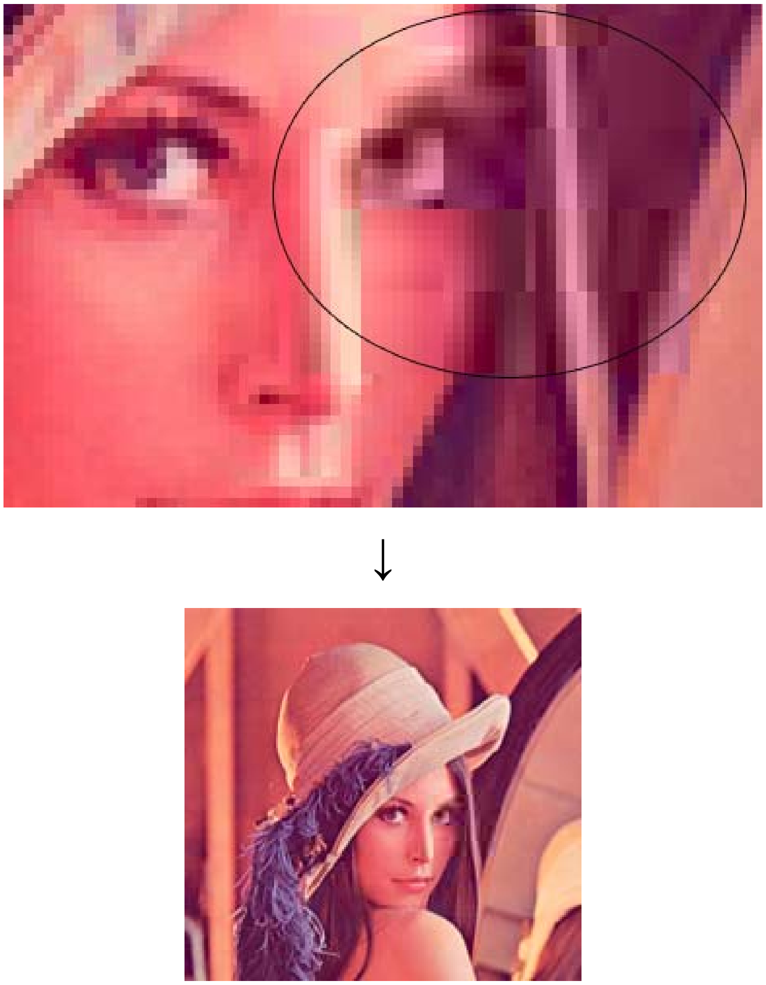


Figure 2.18 – Example of forgery from two different images with different compression

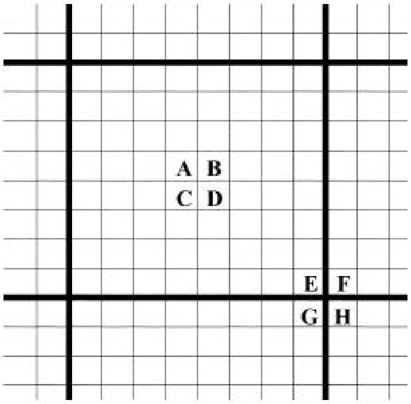


Figure 2.19 – Abstract representation of an *8 x 8* block used by the JPEG algorithm [8]

Solving the following equations calculates the differences [8]:

$$Z'(i,j) = \left|A - B - C + D\right| \qquad Z''(i,j) = \left|E - F - G + H\right|$$

Finally, the histograms of Z' and Z'' are computed. The resulting information is analyzed to look for a discrepancy in pixel patterns. If there appears to be differing histogram results over multiple blocks, it is determined that the image has been previously compressed. Respective histograms that are extremely similar over multiple blocks warrant an image that has not been previously compressed. [8]

Further analysis of JPEG images exist which build upon the previous paragraph. This includes the estimation of the primary quantization table from an image that has been JPEG compressed twice [11]. By again analyzing each *8 x 8* block of an image, statistical determination can be made whether an image has been double compressed. The key here is to understand what occurs when an image has been compressed twice, and then take advantage this phenomenon. When an image is compressed for the first time, corresponding pixels are the result of rounded integers. When the second compression occurs, these rounded values are used again to compute with the second quantization table, Q^2. By analyzing the histograms of these quantized coefficients, an attempt is made to find a pattern which leads back to the original quantization table, Q^1. This technique is useful at blindly detecting images that have been watermarked [11]. Most watermarking programs take a “cover image,” insert hidden information, and then save the image again, hence yielding a double compression. Estimating the primary quantization table assists in determining the watermark used.

The methods discussed in this subsection deal with performing analysis of an image with respect to JPEG compression. Much information can be determined from this type of analysis and could be promising at its ability to detect image manipulations. It is possible for an image tampering expert to perfect a technique to create near flawless forgeries, concentrating on covering their tracks of “hidden” attributes of an image, such as JPEG compression blocks. But this area is still worthwhile and should be investigated further.

2.7 Summary

This chapter discussed the current state of research in terms of digital image forensics. While digital watermarking has been the method of choice to safe-guard one’s images from manipulation and to secure a copyrightable image, it has been difficult to determine if an image of unknown origin is authentic. Several techniques exist that touch the surface of the subject. These hold some sound results, as previously discussed, but further analysis is needed to determine the best and most efficient method to detect image forgeries. The Exact and Robust Matching algorithm to detect *copy-move* image forgeries shows potential as a tool already exists to detect this type of tampering [12]. But the areas of *copy-create* forgeries is in need of more research. First and Second Order convolution filters as well as preliminary spectral analysis approaches analyzed by Lukas returned discouraging results [17]. The recent results of Farid and Popescu take spectral analysis approaches further by devising a useful tool for detecting image forgeries. As with all of the techniques presented, close human interpretation is needed and there appears to be no “silver bullet” in terms of a detection scheme. Various methods available in the image processing toolkit will need to be applied to this area with results closely scrutinized. An interesting approach that requires more investigation is one that looks at the JPEG compression scheme of an image. Even though a forgery may appear to be flawless to the human eye, small underlying details of the JPEG “fingerprint” could be its Achilles’ heal.

Chapter 3 : Methodology

3.1 Introduction

This chapter describes the methodology used in determining which techniques are best capable at detecting *copy-create* image forgeries. Chapter 2 presented the background on several approaches, including methods based on HSV and Luminance level analysis, various filtering masks, and analysis based on the JPEG compression scheme. This chapter describes the Luminance, HSV, Custom High-Pass Filtering, and JPEG Block methods and presents an experimental design of a test-bed of digital image forgeries. The scope and suspected system boundaries are also discussed.

3.2.1 Methods Based on Hue-Saturation-Value (HSV) and Luminance Levels

Sections 2.6.1 and 2.6.2 presented a description of the Luminance and HSV levels of a digital image. These properties of an image are influenced by the amount and intensity of light when the photograph was taken, the particular camera's representation of color, and any post-processing performed by image manipulation software. In a forged image, these properties may all slightly differ in and around tampered areas. This is because of the multiple images used to create the forgery, each most likely originating from different cameras and environments.

While it is necessary for human interpretation when analyzing the results of an image tamper detection algorithm, there are techniques and methods to assist. A color or grayscale image can be converted to a binary image based on some configurable threshold. A binary image is one in which a pixel is either “on” (filled with black) or “off” (empty or filled with white) [20]. The determination of whether a pixel gets an “on” or “off” value is based on a threshold. In the context of this section, the threshold is the luminance level of an image.

The luminance level threshold of an image is represented as a decimal number between 0.00 and 1.00. If the threshold value is 0.75, for example, a pixel is assigned as on (or black) if its luminance level is equal to or less than 0.75. An image is returned all black if the threshold level is set to 1.00 or returned all white if the threshold level is set to 0.00. Determining an appropriate threshold for testing is heavily dependent on the type of image being analyzed. A value of approximately 0.50 is a good starting point with subsequent tests performed in both directions. The ultimate goal is to look for results depicting an area of suspected tampering, which are witnessed by unnatural or abnormal luminance levels in an alleged area. Figure 3.1 shows the binary counterpart of Figure 3.2(a) based on a luminance threshold of 0.30.

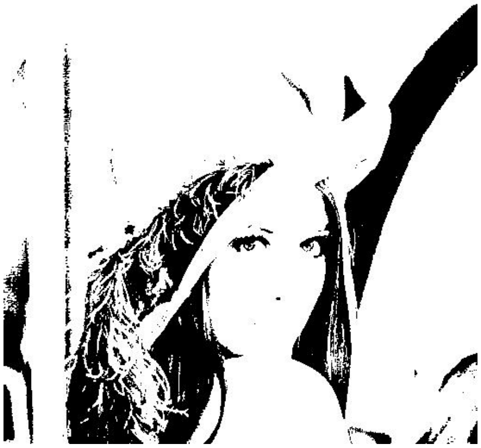
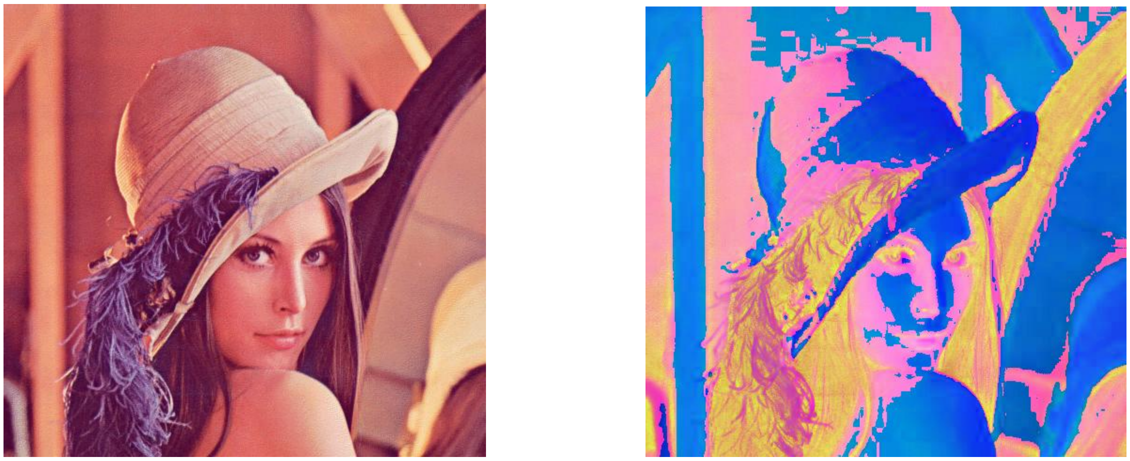


Figure 3.1 – Binary Counterpart of Figure 3.2(a) with Luminance Threshold 0.30

Further analysis of color images may be performed with respect to Hue-Saturation-Value (HSV) levels. The standard RGB (Red-Green-Blue) color-space of a digital image represents each color a pixel takes on by an amount of the Red, Green, and Blue components. This is the standard way that color digital images are represented on the computer screen. In the HSV color-space, the amount of Hue, Saturation, and Value represents each color. As discussed in Section 2.6.2, the Hue of a color is described as the "tint", the Saturation or "shade" is defined as the level of how pure or intense a color is, and finally the Value is the level of brightness (luminance) of a color or how light or dark it is. [20] Figure 3.2(a) and 3.2(b) show an image in the standard RGB color-space and its counterpart in the HSV color-space, respectively.



(a) RGB (b) HSV

Figure 3.2 – Images in the Red-Green-Blue and Hue-Saturation-Value color-spaces

The color-space and luminance levels of an image provide a unique method of analysis. Discrepancies and other anomalies in these attributes indicate image tampering has taken place. To exhibit a scenario explaining what to look for in both the luminance and HSV methods, the following examples are presented. Figure 3.3 depicts the binary image after performing a luminance level analysis on the forged Figure 3.10 with a threshold of 0.60. In the magnification presented here, the tampered area shows an abnormal pattern with respect to the rest of the image.

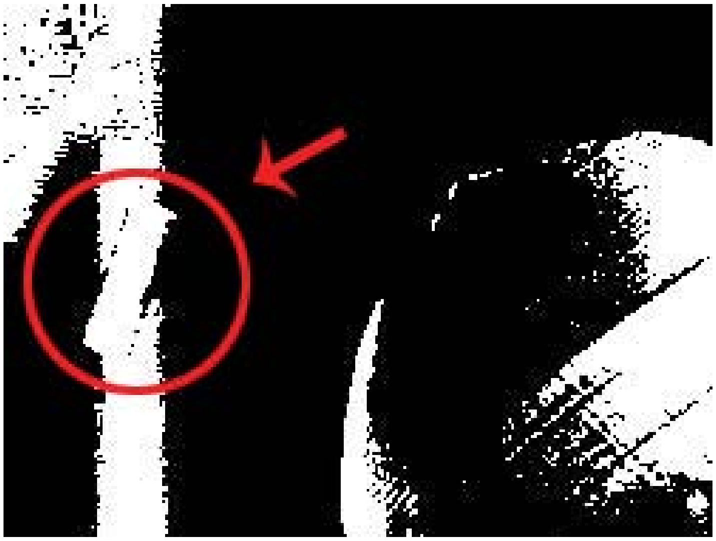


Figure 3.3 – Result of Performing Luminance Level Threshold 0.60 on Forged Figure 3.10

Additionally, Figure 3.4 shows the results of a similar test performed on Figure 3.10 using the HSV color-space. Again, the magnified area in this figure illustrates the tampered portion by showing an uneven color pattern and shape compared with the surrounding area. The abnormal color “bleeding” also indicates some form of tampering has occurred.

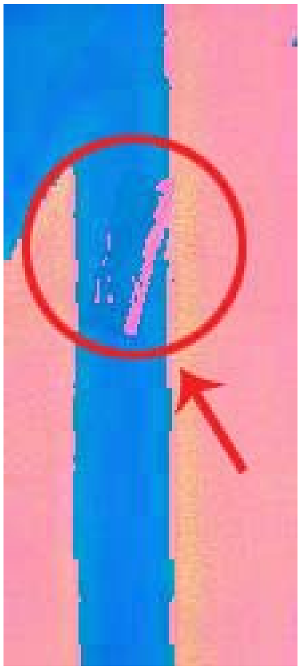


Figure 3.4 – Result of Converting Forged Figure 3.10 into HSV color-space

The results of these methods attempt to analyze an image using various color and brightness components. An equally important method for tamper detection lies in the results of filtering an image, discussed in the following subsection.

3.2.2 Methods Based on Alternative Filtering Masks

Section 2.4.1 and 2.4.2 presented several filtering methods analyzed by Lukas [17]. These include filtering based on the Roberts', Sobel, Prewitt, and Marr masks. These methods have been limited in their detection of image forgeries, as discussed in Section 2.5. Although these masks provide a foundation for image filtering, the use of a custom mask would allow for better tailoring to the detection of image tampering. Section 2.6.3 discussed the fundamentals of a filtering mask and the configurability it possesses. By using various convolution kernels, emphasis is placed on a particular image's attributes, such as edges or distinct contrast. The detection of anomalies caused by image tampering is the ultimate goal of a good convolution kernel. The method presented here uses *3 x 3* block size, as described in Section 2.6.3, because of its power to capture the trends in an image without introducing too much pixel variation found in a larger block size. Evidence of image tampering arises in areas where double edges and other abnormal patterns exist. A good filtering method magnifies these irregular edges. In an effort to match these requirements, the following convolution kernel is presented.

$$\begin{bmatrix} -1 & -2 & -1 \\ -2 & 12 & -2 \\ -1 & -2 & -1 \end{bmatrix}$$

Custom Convolution Mask

The weight of 12 is placed on the center pixel along with all other neighbors' weights summing to -12. This effectively filters out all areas in the image that are statistically similar and only shows those that are very different. These different areas most likely arise from prominent edges, but may also surface in locations that are victim to image tampering. Since the resulting image is very dark in nature, alternative viewing modes enable easier analysis in print or on screen. These include inverting the resulting image which makes darker pixels lighter and vice versa. It should be noted that the image in question is first converted to grayscale before performing the convolution filtering. This should have no effect on the statistical data of the original image. Figure 3.5 shows the result of using the custom filter just described on the image in Figure 3.2(a).

This method produces an extremely high pass filtered version of the original image. Due to this fact, Figure 3.5 and Figure 3.6 were inverted for readability in print. Tampered areas surface easily in this high pass filtered version. Therefore, it is wise to perform analysis of portions within the image that are noisy or contain “hidden” edges, paying close attention to double or “ghost” edges and uneven patterns on suspected areas. Figure 3.6 shows this filtering method on Figure 3.10. In this example, the magnified top left portion shows the tampered area which exhibits a distinctive abnormal pattern in comparison with the surrounding area. An alternative method of detecting forgeries in JPEG images lie in the “fingerprint” of the JPEG standard. The next subsection discusses this concept.

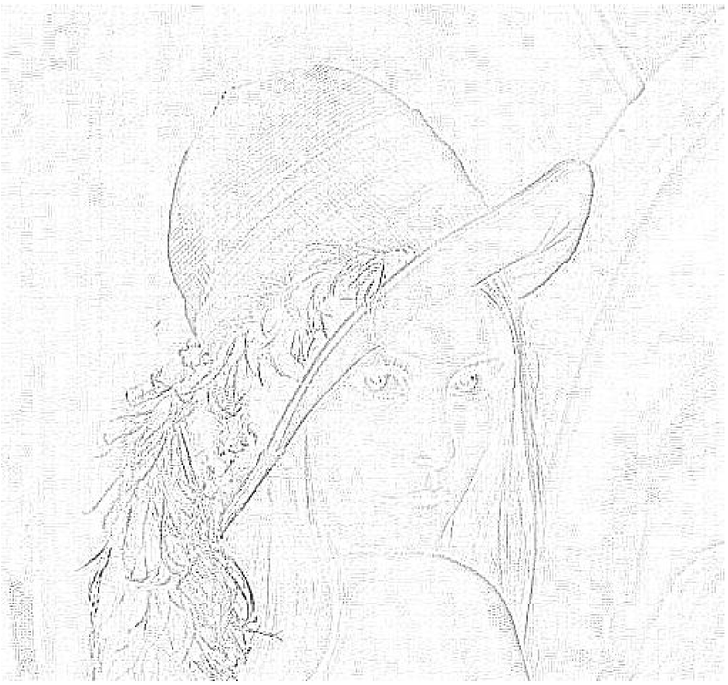


Figure 3.5 – Inverted Result of Performing Custom Filter Mask on Figure 3.2(a)

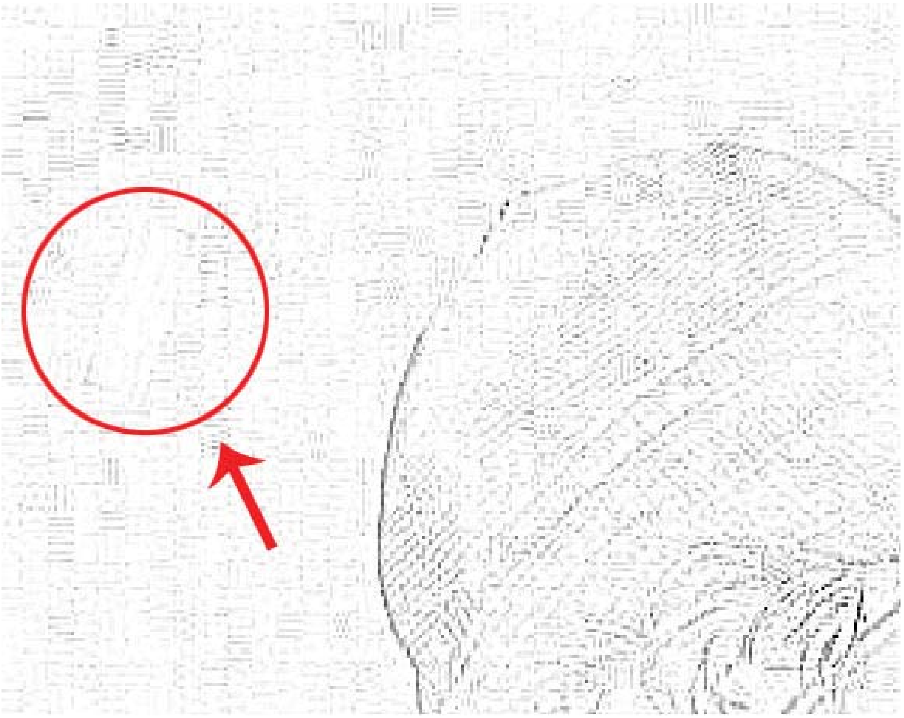


Figure 3.6 – Inverted Result of Performing Custom Filter Mask on Forged Figure 3.10

3.2.3 Methods Based on JPEG compression analysis

Every image that has been JPEG compressed goes through a series of steps transforming a raw image into one that is compressed. Based on the compression level desired, this image can have a small file size or be of higher quality. As discussed in Section 2.6.4, an analysis of an image may be performed with respect to the JPEG compression standard. An image compressed using this standard is broken up into disjoint *8 x 8* blocks, which the compression calculation uses. These blocks effectively form a "fingerprint" of the image.

Creating an image forgery from two or more JPEG images may cause statistical discrepancies across the image as a whole. When an image is altered, it may be composed of several pieces of other images which are cropped, scaled, and rotated to make the forged image's authenticity more believable. Also, these pieces may have originated from images that have previously been JPEG compressed with differing Quality Factors. These operations introduce small anomalies in the statistical data of the newly created forged JPEG image. Therefore, this proposed technique to detect tampering analyzes a JPEG image with respect to the *8 x 8* blocks used by the JPEG compression scheme. Performing statistical calculations on the boundaries of these blocks builds upon the technique presented by Fan and Queiroz [8] for detecting prior JPEG compression in a BMP image. Figure 3.7 shows an abstract representation of an *8 x 8* block of pixels in a JPEG image with letters representing interested pixel values.

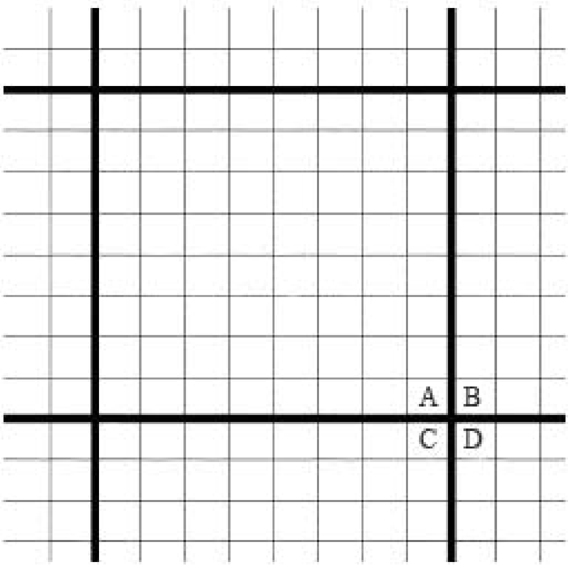


Figure 3.7 – Abstract representation of an *8 x 8* block used by JPEG compression

The following result is calculated based on the numerical pixel values at locations A, B, C, and D in every *8 x 8* block, (i, j), which makes up the image (see Figure 3.7):

$$R(i, j) = \left| A - B - C + D \right|$$

Each $R(i, j)$ value effectively represents the degree of pixel variation present between an *8 x 8* block and its 3 neighbors, as depicted in Figure 3.7. Intuitively, an image that is heavily compressed should result in different statistical data across the entire image as compared to one that is less compressed. Figure 3.8 illustrates a magnified portion of an image which is heavily compressed (QF = 5) and Figure 3.9 shows the same portion but from a much lesser compressed image (QF > 70). The *8 x 8* blocks present in both images show how the calculated $R(i, j)$ values will be dependent on the Quality Factor used when saving an image in the JPEG format.

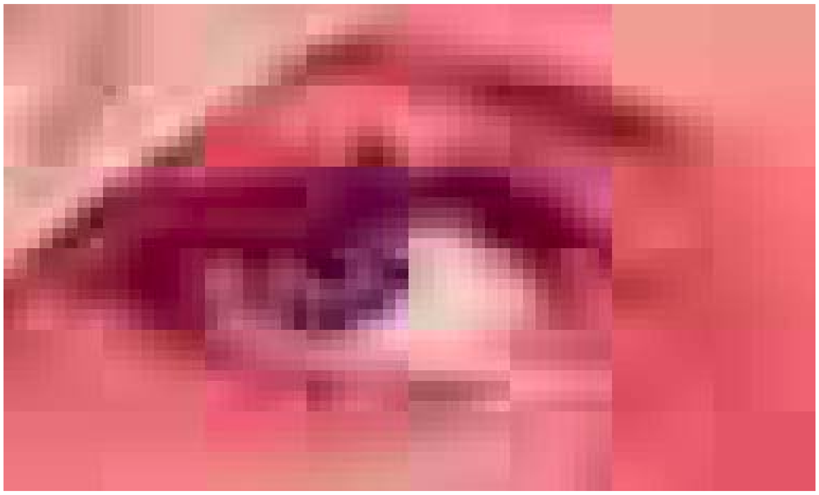


Figure 3.8 – Magnified portion of heavily compressed JPEG image depicting *8 x 8* blocks

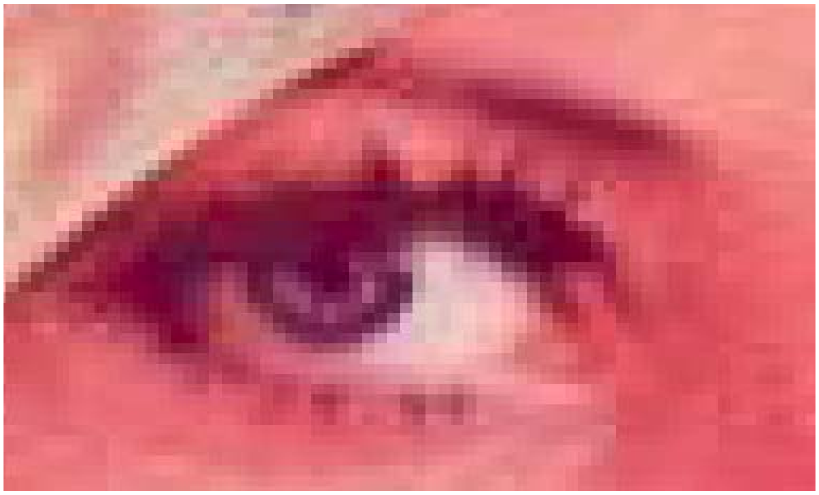


Figure 3.9 – Magnified portion of lesser compressed JPEG image depicting *8 x 8* blocks

Creating a forged JPEG image from portions of two other JPEG images, with different Quality Factors or having been cropped, rotated, or rescaled, introduces anomalies in the average $R(i, j)$ values across the image. By examining this type of tampering at magnification, these anomalies are easily distinguished by the human eye. Figure 3.10 shows a simulation of an image forgery with tampering performed on the top left portion. In this example, a segment was copied from a different JPEG image with a lower Quality Factor, pasted onto the host image, and that portion was then reshaped and resized to simulate a forgery. The magnified portion of the tampered area shows remnants of the original image it was pasted from and its *8 x 8* blocks, which have become distorted from tampering.

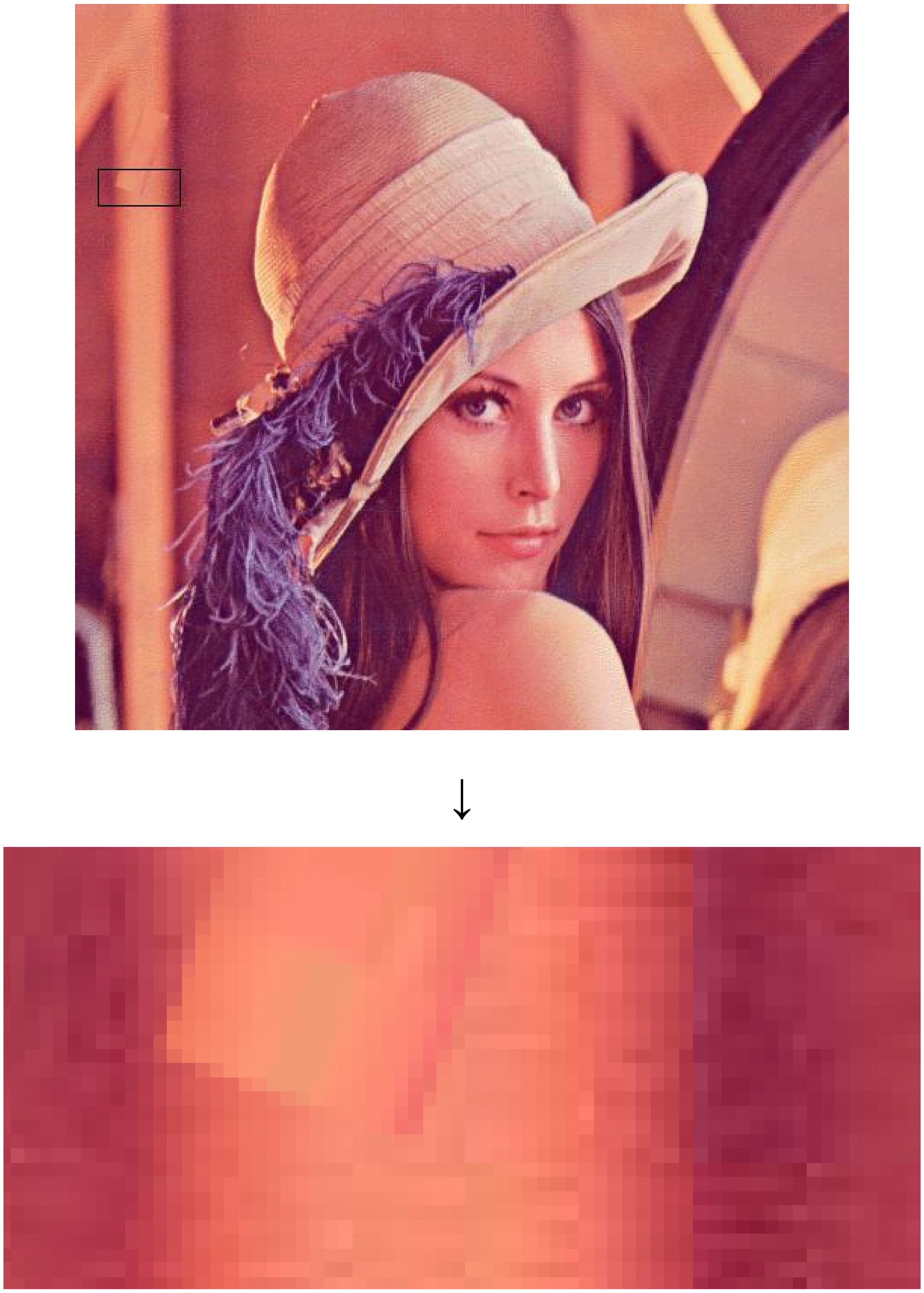


Figure 3.10 – Simulation of Image Forgery with portion of tampered area magnified

The simulated image forgery in Figure 3.10 shows that the $R(i, j)$ value obtained by analyzing the *8 x 8* pixel blocks of a forged image may show some statistical anomalies in the tampered areas. To capture some statistical data of the suspected image forgery, all $R(i, j)$ values can be calculated for each *8 x 8* block and then analyzed by some threshold value, *t*, producing a binary image. This threshold value and subsequent binary image is similar in nature to that discussed in Section 3.2.1. The difference here is that all 64 pixels in each *8 x 8* block will be either entirely white or black based on the value obtained from the following formula:

$$D_{right}(i, j) = \left|R(i, j) - R(i, j+1)\right| \qquad D_{bottom}(i, j) = \left|R(i, j) - R(i+1, j)\right| \qquad (1)$$

Here, $D_{right}(i, j)$ equals the difference between a block's $R(i, j)$ value and its direct neighbor to the right's value, which is represented as $R(i, j+1)$. $D_{bottom}(i, j)$ is equal to the difference between a block's $R(i, j)$ value and its direct neighbor to the bottom's value, which is represented as $R(i+1, j)$. Blocks at the far right and bottom edge of an image get $D_{right}(i, j)$ and $D_{bottom}(i, j)$ values equal to zero as this should not be an area of interest for tamper detection. Therefore, they are effectively ignored. Once all $D_{right}(i, j)$ and $D_{bottom}(i, j)$ values are calculated, the threshold value, *t*, is used to set all blocks equal to white if their $D_{right}(i, j)$ or $D_{bottom}(i, j)$ value is equal to or greater than *t*. Otherwise it is set to black. Emphasis should be placed on the "or" condition from "$D_{right}(i, j)$ or $D_{bottom}(i, j)$" in the previous sentence. Much testing has gone into determining which blocks should be used in the comparison of their respective $R(i, j)$ differences. Combinations of block differences as well as changing the above conditional "or" to an "and" were used for comparison purposes. From testing, there was not much difference in the overall result when using the various block combinations. Ultimately, the decision was made to use the proposed method which calculates the difference in $R(i, j)$ values of a block with its right and bottom neighbors (1). A white block is returned if at least one comparison was within the bounds of the threshold. By using this "or" comparison, increased emphasis is witnessed on the tampered area, if one exists. Figure 3.11 provides a comprehensive algorithm describing this method.

```
Block_Analysis (image, t)
    divide image into disjoint 8 x 8 compression blocks ( i , j )
    for each 8 x 8 JPEG compression block ( i , j ) within bounds
        $R(i, j) = |A - B - C + D|$ where A = pixel value ( 8*i , 8*j ), B = pixel
            value ( 8*i , [8*j] + 1 ), C = pixel value ( [8*i] + 1 , 8*j ), D =
            pixel value ( [8*i] + 1 , [8*j] + 1 )
    for each 8 x 8 JPEG compression block ( i , j ) within bounds
        $D_{right}(i, j) = |R(i, j) - R(i, j+1)|$
        $D_{bottom}(i, j) = |R(i, j) - R(i+1, j)|$
    for each 8 x 8 JPEG compression block ( i , j ) within bounds
        if ( $D_{right}(i, j) \geq t$ ) OR ( $D_{bottom}(i, j) \geq t$ )
            set all pixel values in ( i , j ) to white
        else
            set all pixel values in ( i , j ) to black
end Block_Analysis
```

Figure 3.11 – Algorithm for JPEG Block Technique

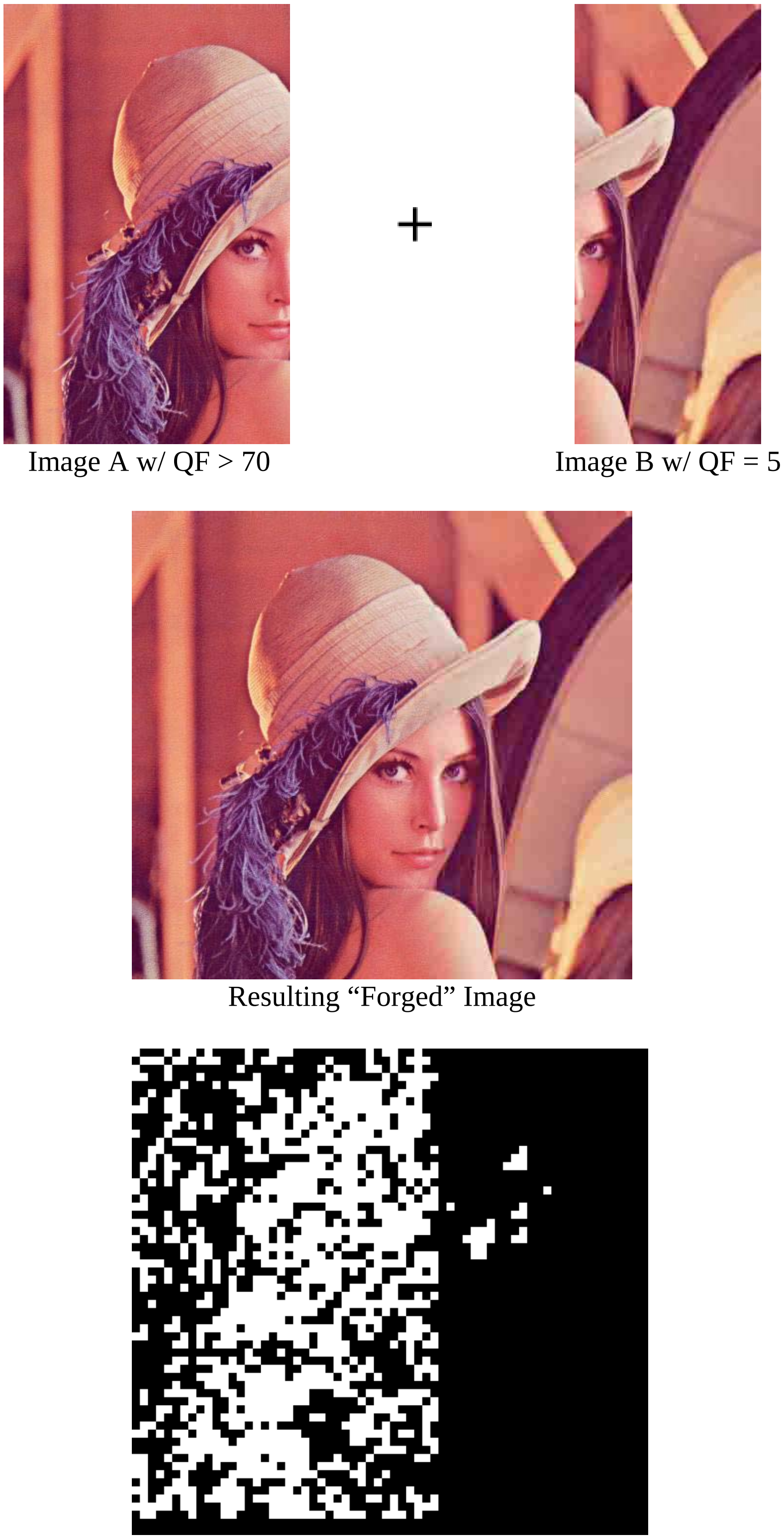


Image A w/ QF > 70

Image B w/ QF = 5

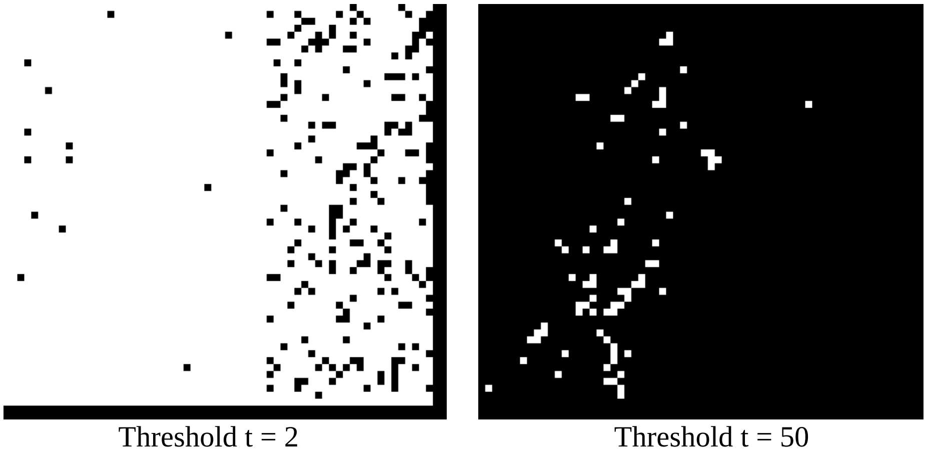
Resulting “Forged” Image

Binary Image Result with Threshold *t* = 15

Figure 3.12 – “Forged” Image with Result from the JPEG Block Technique

Figure 3.12 provides an illustration of this proposed JPEG Block Technique. In this example, two exact copies produced the forged image, with one copy being of higher quality (QF > 70) and the other being more heavily compressed (QF = 5). The left portion of the forged image is composed of the better quality donor image and the right portion is made of the other more heavily compressed donor image. This figure shows the resulting binary image using a threshold, *t*, equal to 15. The binary result of this proposed block analysis technique has uncovered a definitive pattern in the differing compression levels of the two merged images. This is a good example of how the naked eye is fooled by the authenticity of a forged image. It is difficult to detect any means of image tampering at face-value in this example, but the "fingerprint" of the JPEG compression scheme leaves much to analyze at the pixel level. Section 3.4 discusses an in-depth experiment which investigates the correctness and performance of this technique.

The threshold value, *t*, allows for a user-defined way to analyze an image with respect to the variability in each *8 x 8* block, as previously described. This approach gives the opportunity to allow for a fine-tuned analysis of all JPEG images of different sizes, resolutions, and Quality Factors. It is extremely difficult to develop a method to detect image tampering that does not require some human configurability and interpretation. Thus, setting a user-defined threshold value, *t*, and subsequent binary image analysis is required for maximum robustness with this method. Determining an appropriate threshold value for a given image requires the user to perform several experiments. As the sign of image tampering may come from both threshold value extremes, several tests should be completed to narrow down any firm clues that highlight tampered areas (or that conclude none are tampered). Figure 3.13 shows two resulting binary images of the forgery in Figure 3.12 with threshold value, *t*, equal to the values 2 and 50. When *t* is equal to the value 2, it is more difficult to see a pattern of the image tampering. Most blocks are colored white since each block in the resulting binary image is colored white if the degree of variability is greater than or equal to the value 2. On the other extreme, when *t* is equal to the value 50, most blocks are colored black and a definitive pattern still can not firmly be concluded from the resulting binary image.



Threshold t = 2

Threshold t = 50

Figure 3.13 – Result from the JPEG Block Technique of the forged image in Figure 3.12

An approach to take in determining a proper threshold is to start out with a threshold value equal to 50. The result should then be analyzed with further testing performed using threshold values in increments of 5 or 10. Each test should look for distinctive patterns in the binary image or be focused on the results of areas that have been suspected of tampering. As the threshold value is increased, the resulting white pixels usually focus on areas of image tampering, if any exist. This is because high levels of JPEG block variability are usually seen in areas with prominent edges or that have been digitally tampered with. The tampered areas usually show higher JPEG block variability than legitimate areas, such as edges, and thus a higher threshold value will usually focus in on these tampered areas. When a pattern emerges from performing the JPEG Block Technique, further testing using the other proposed methods should be performed with the results of the suspected area analyzed closely.

Finally, an analysis of the results using a threshold value under 15 should be performed to determine if this returns any definitive patterns of image tampering. The tampered area in an image may have come from an original image with a higher or lower JPEG Quality Factor than the other parts of the image, therefore a pattern of white or black blocks may result from this. Thus, analyzing results using a wide range of threshold values is the best approach when using this technique. Figure 3.12 is a good example of this. Using the JPEG Block Technique with a lower threshold value, in this case, increases the emphasis on the tampering. Other threshold values, witnessed in Figure 3.13, return results that do not show the tampering well.

3.3 System Boundaries

The proposed techniques to detect digital image tampering, presented in Sections 3.2.1 through 3.2.3, have several limits in the scope of this study. All of the methods presented are tailored to the JPEG image format. This format was chosen because of its popularity on the web and with digital cameras [2]. Other image formats such as PNG (Portable Network Graphics) and BMP (Windows Bitmap) work with the proposed Luminance and HSV methods as well as with the alternative filtering masks. Obviously, the JPEG Block technique only accurately works with images saved in the JPEG format.

The best way to detect image tampering, in general, is to perform a wide range of tests on the image in question. The methodology presented in this chapter attempts to accomplish this by including a broad range of detection techniques. Table 3.1 includes a listing of the proposed detection methods and those image formats that work with each.

Forgery Detection Method	**RGB Color**	**Grayscale**	**JPEG**	**PNG**	**BMP**
Luminance Level Threshold	X	X	X	X	X
HSV Color-Space	X		X	X	X
Custom High-Pass Filter	X	X	X	X	X
JPEG Block Technique	X	X	X		

Table 3.1 – Proposed Detection Methods and Tested Image Format

This chapter focuses on methods to detect digital forgeries created from multiple images. Chapter 2 defines this classification as *copy-create* image forgeries. Images that result from portions copied and moved within the same image to "cover-up" something are classified as *copy-move* forgeries. Fridrich has looked extensively at this class and has designed a very promising tool to detect this type of image forgery [12]. Therefore, the experimental design and analysis herein focuses on *copy-create* image forgeries.

A crafty individual, who wants to perfect an image forgery, with time not a factor, can usually give any detection method trouble. This includes transforming an image from digital to analog back to digital (i.e. taking a 35mm picture of the computer screen displaying the forgery and then scanning the developed picture back into the computer) or spending copious amounts of time sculpting individual pixels via image manipulation software. If image tampering occurs in an uncompressed image and then that image is converted to the JPEG image format, the JPEG Block Technique will fail to capture evidence of tampering. This conversion process destroys all proof of tampering since the original tampering does not affect any JPEG blocks. Additionally, any image tampering performed on an image prior to an image size reduction will eliminate detectable anomalies for the custom filtering mask.

By and large, boundaries will always exist in detecting image tampering. Some assumptions need to be made about countermeasures a forger used when crafting an image forgery. The methods presented in this Chapter are well suited to detect varying skill levels of a person performing image manipulations, but given enough time and resources there are ways to give any detection algorithm difficulty in accurately deciding if an image is authentic.

3.4 Experimental Design

Sections 3.2.1 through 3.2.3 present several methods to detect image tampering. These include techniques based on the Luminance and HSV levels of an image, a custom image filtering mask, and a JPEG block analysis technique. Testing for the accuracy of these methods includes performing all of the techniques on a series of test images. Appendix B contains these test images, which includes the image, a description of the tampering, and the attributes of that image (i.e. resolution, file size, etc.). The correctness and accuracy of the JPEG Block Technique is suspected to rely heavily on how the image forgery was actually composed. Therefore, specific testing is performed on this method using forgeries composed of differing Quality Factors.

Appendix B includes images that represent the fundamental image tampering typically seen in the real-world. Image forgeries that are typically crafted by someone with novice skills are included, i.e. not adjusting brightness or size of the tampered area. On the other hand, image forgeries that are typically performed by someone with intermediate to expert skill level are included. Various image sizes and resolutions are included for maximum scope of testing. This array of image forgeries allows for analysis of when each method fails, which Chapter 4 discusses. Finally, a test image containing an *invisible* watermark using LSB steganography is included to determine what effects a watermark has on the validity of each proposed detection method. Section 2.2 discusses watermarking and provides examples of the various watermarking techniques. Table 3.2 sums up these described experiments.

A major factor in determining the correctness and accuracy of a method is to perform a test representing a broad set of image forgeries. While it is impossible to predict how or where image tampering will be performed, the examples tested will exhibit fundamental tampering techniques used in a typical forgery. As previously discussed, these include cropping, rescaling, and rotating the tampered area as well as using pieces of varying JPEG Quality Factors to make the forgery. While these are assumptions in the way image forgeries are created, they are justified by evidence witnessed in typical image tampering.

The images analyzed in the above experiment are included to stress the strengths and weaknesses of each forgery detection method. Subsequently, the tampered portion in these images is known prior to performing each experiment and thus makes each test somewhat subjective. Therefore, an independent party provided 15 JPEG test images which allows for an unbiased experiment with no preconceived knowledge of image tampering. This mixture of forged and authentic images contains various types of image forgeries commonly found on the web and other suspect sources. Overall, this type of experiment verifies a percentage of correctly identified images in a situation similar to one witnessed in the real-world. Table 3.2 includes the tests conducted with a description for each of the images in the blind image test-bed. The images in this test-bed represent expert level forgeries.

Experiment	**Detection Method**	**Metric**
(1) Luminance Level Method Test	Luminance Level	Measure perceived variability in luminance levels in test image. Identity and document areas of luminance discrepancy, if any exist.
(2) HSV Level Method Test	HSV	Measure perceived variability in HSV levels in test image. Identity and document areas of color and/or light discrepancies, if any exist.
(3) Custom Filtering Method Test	Custom Filtering	Measure perceived object edges in test image. Identity and document areas containing double or “ghost” edges, if any exists.
(4) JPEG Block Method Test	JPEG Block	Measure variability of JPEG Blocks in test image. Identity and document areas of block concentration signaling tampering, if any exists.
(5) Quality Factor Test	JPEG Block	Count number of flagged white JPEG Blocks in each Quality Factor test image.
(6) Blind Image Test	All	Execute each detection method on test image to identity and document abnormal areas, if any exist.

Table 3.2 – Design of Experiments to Test Image Forgery Detection Methods

Each experiment described in Table 3.2 hopes to provide evidence to decide if an image is authentic or forged. The image in question is analyzed to the fullest potential by each method. To be as objective as possible in collecting a meaningful metric, a strict evaluation technique is followed which the next section presents. Appendix C includes the resulting images of Experiments 1 through 5.

3.5 Evaluation Technique

The experimental design presented in Section 3.4 lays out the tests performed on the proposed methods to detect image tampering. As previously mentioned, the results of any forgery detection scheme require some sort of human interpretation. The methods proposed in this chapter are no different. Therefore, to validate the results of the experiments, close analysis with respect to the tested image is performed. The Luminance and HSV techniques require examining the areas suspected of tampering for discrepancies in brightness or color pattern compared to other areas in the image. Performing image filtering using the custom filtering mask described in Section 3.2.2 requires an analysis of the outputted image for areas that contain abnormal or inconsistent edges. Finally, the JPEG Block Technique is performed using various threshold values, discussed in Section 3.2.3, and the subsequent results examined for distinctive patterns of image tampering. Again, when a need is present to verify an image's authenticity, it is wise to perform all of these methods because each has a unique approach at detecting image forgeries.

To determine when each of the proposed techniques fails, each method is performed against varying types of image forgeries, as discussed in Section 3.4. Attention is placed on the fundamental concepts of each technique with using observable facts about each to discuss respective shortcomings in Chapter 4.

Finally, the blind image test requires an objective approach with time devoted to fully analyze each image. A spiral approach is used to exhaust the powers of each method to make a conclusion about each image. Chapter 4 includes analysis of this experiment as well as the percentage correctly identified.

3.6 Summary

This chapter outlined the methodology used in detecting digital image tampering from images with unknown origin. While it is difficult to predict exactly how a malicious person will forge an image, a wide range of techniques have been presented to account for tamper methods. The detection system proposed includes methods based on Luminance and HSV levels of an image, a custom filtering mask, and an analysis of JPEG compression blocks. An experiment testing these methods has been set-up that will help in determining the accuracy and correctness of the proposed tamper detection techniques, as well as when each fails. It has been conjectured that these methods will help in detecting various types of image forgeries, but one has to acknowledge that no “silver-bullet” exists to account for every type of forgery imaginable. To wrap up testing, an independent experiment is presented to help analyze the correctness of this system of techniques at accurately identifying blind images as authentic or forged.

Chapter 4 : Results

4.1 Introduction

This chapter provides a discussion of the results of each experiment described in Section 3.4. The accuracy and correctness of each technique is examined in detail as well as an analysis when each technique fails. Appendix C includes the resulting images from each experiment which the sections in this chapter use for reference. A table is also presented in each section which summarizes the accuracy of each technique with respect to the test images.

4.2.1 Analysis of the Luminance Level Technique

The Luminance Level Technique presented in Chapter 3 analyzes an image with regard to a threshold of a pixel's luminance level. Appendix C includes resulting images of this technique when performed on the test-bed of forgeries displayed in Appendix B. Table 4.1 provides a summary of these results.

Two of the nine test images return results that exhibit uncertain signs of tampering. Forgery B.4 depicts the character "Yoda" merged into a background image of a field on-looking a lake. The tampered area in this example is small and contains similar luminance levels as the background image. Obviously this image is forged, but the Luminance Level Technique is not well suited to capture signs of this type of forgery. Additionally, Forgery B.6 is an extremely small and heavily compressed image. Limited pixel information is present and therefore this technique fails to pick up on any signs of digital tampering.

Five of the nine test images show signs of possible image tampering. These warrant additional testing with the other tamper detection techniques. Forgeries B.1, B.5, B.WM.Wno, and B.WM.Wyes show the tampered area having signs of uneven or unnatural luminance levels compared to nearby or similar areas. Here, the light source from the forged image is inconsistent with the light source of the host image resulting in the Luminance Level Technique capturing the tampered area. In actuality, someone wishing to create a good forgery usually takes care to adjust the luminance levels of the forged image. Forgery B.7 raises suspicion based on “hidden” discontinuities present in the tampered area. The results of this test more easily capture the signs of “touching up” and “blending” this area into the background image. Figure 4.1 demonstrates this by providing a magnified view of the results of the tampered area.

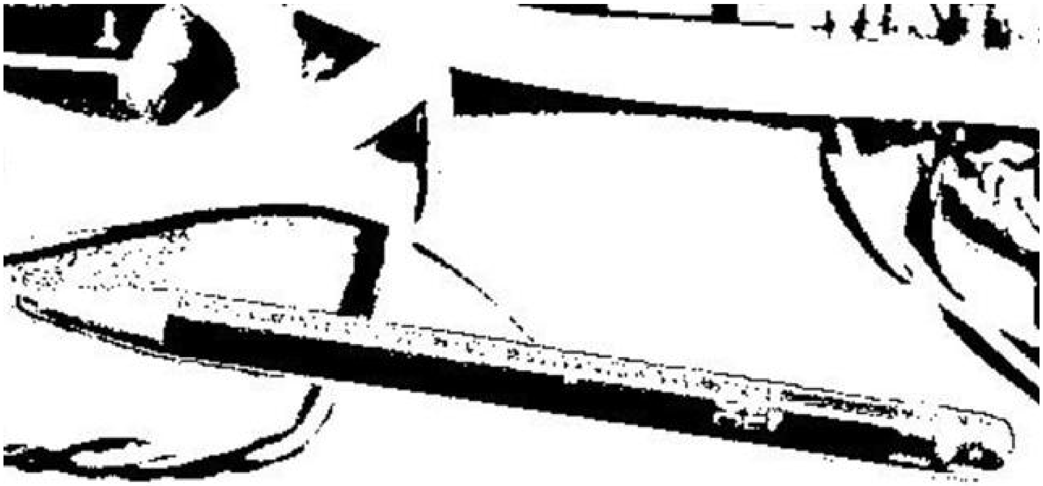


Figure 4.1 – Magnified Portion of the results of Forgery B.7

Forgery	Inconclusive Signs of Tampering	Possible Signs of Tampering (should run other techniques)	Definitive Signs of Tampering
B.1		X	
B.2			X
B.3			X
B.4	X		
B.5		X	
B.6	X		
B.7		X	
B.WMno		X	
B.WMyes		X	

Table 4.1 – Summary of the Results of the Luminance Level Technique

The result of executing the Luminance Level Technique on Forgeries B.2 and B.3 return definitive signs of image tampering. In Forgery B.2, the man depicted in the picture has obviously been tampered with. The original image before testing shows no shadow cast by the man as well as a differing light source. The results from performing the Luminance Level Technique confirm that, given a threshold equal to 0.25, the man's image is extremely bright and abnormal compared to the road and grass surrounding it. Therefore, this example validates suspicion of image tampering. Additionally, Forgery B.3 contains two shadows which include very different brightness levels. Since the light source in this image is coming from the same location, both shadows should be relatively similar. Since this is not the case, the Luminance Level Techniques confirms that this image is a forgery.

A test of the Luminance Level Technique between a watermarked image (Forgery B.WMyes) and one that is visibly identical but without a watermark (Forgery B.WMno) yields nearly identical results. Since an *invisible* watermark is embedded into a cover image to be indistinguishable with the original non-watermarked image, the luminance level of each pixel is barely changed, if at all. This phenomenon confirms itself in the results of this test.

Taken as a whole, the method discussed in this section proves to offer some unique and fundamental results when testing an image for digital tampering. While most cases in this particular experiment result in poor detection of image tampering or require further testing using one of the other techniques, this method may provide, in general, further justification that an image is forged. Nevertheless, most skilled persons adjust and blend luminance levels of the tampered area to make the resulting forgery more authentic, thus causing the Luminance Level Technique discussed here to fall short. This leads into an analysis of the other proposed techniques.

4.2.2 Analysis of the HSV Level Technique

Chapter 3 describes a method to detect image tampering based on analysis of the image in the Hue-Saturation-Value (HSV) color-space. This method allows for an alternative view of the image in a color-space that may uncover a tampered area. Appendix C includes the results of this method when performed on the test-bed of images in Appendix B. Table 4.2 summarizes the results.

As Table 4.2 suggests, this method returns average results in detecting image forgeries as five of the nine forged images result in indecisive signs of tampering.

Forgeries B.4, B.6, B.7, B.WMNo, and B.WMYes show no conclusive signs that tampering has occurred. Specifically, the results of these forged areas have HSV color patterns that could, in fact, be from authentic areas. Forgery B.WMno in Appendix C provides an explicit example of this. Here, similar HSV colors represent the monster in the water as well as the other areas in the image, such as the left cow. Although the result of the HSV method on this image returns no definitive signs of tampering, the monster is obviously forged from another image.

Forgery	**Inconclusive Signs of Tampering**	**Possible Signs of Tampering (should run other techniques)**	**Definitive Signs of Tampering**
B.1		X	
B.2		X	
B.3			X
B.4	X		
B.5		X	
B.6	X		
B.7	X		
B.WMno	X		
B.WMyes	X		

Table 4.2 – Summary of the Results of the HSV Level Technique

Three of the nine test images hint at signs of being forged. While HSV levels in the tampered portions of Forgeries B.1, B.2, and B.5 raise some suspicion of tampering, other tests must be performed to make a solid conclusion. The HSV color patterns in the tampered portions of these images are uneven or abnormal when compared to other areas in the image. Forgery B.2 in Appendix C depicts a particular example of this. The image of the man contains HSV colors which look unusual when compared to other portions of the image. This could be because the light source is different from the original image containing the man with the background image used to house the forgery. Conversely, since there is no other persons depicted in this image to compare against, the HSV pattern of the man's image may be specific to a person's clothing and thus be authentic. In reality, performing further tests using the other methods is wise to make a more solid conclusion about this image.

The HSV Technique confirms that Forgery B.3 exhibits definitive signs of tampering. This image depicts two shadows cast by two different objects. Similar to the discussion in the Section 4.2.1, both shadows should be relatively similar since each is cast from the same light source. As the results of this technique suggest, both shadows are not consistent and therefore raise suspicion. This is an example well suited to the HSV Technique because it involves discrepancies in shadows and brightness.

Comparing the results of the watermarked (Forgery B.WMyes) and non-watermarked (Forgery B.WMno) images, the Luminance Level Technique produces a similar result. Both images have near identical HSV color patterns and levels. This result is not unusual because of the fact that an *invisible* watermark makes very little, if any, changes to the appearance of image.

Overall, the method presented in this section returns average results when used to detect image tampering. There are a few cases in which some suspicion is raised of an image being tampered with though further testing must be performed using the other methods to make a solid conclusion. The only instance that this method was highly accurate was in the situation that brightness or color differences exist in the tampered region. In the end, HSV Level analysis may help provide additional justification of an image forgery and is therefore still a good alternative method to include in an image forensic toolkit. This paves the way into an analysis of the next method dealing with custom high-pass filtering.

4.2.3 Analysis of the Custom High-Pass Filtering Technique

The previous two sections discuss the accuracy and correctness of the Luminance and HSV Level Techniques. This follows into a discussion of the Custom High-Pass Filtering Technique presented in Chapter 3. The image results of this method when performed on the test-bed of images in Appendix B are included in Appendix C. Table 4.3 provides a summary of the results.

Two of the nine results after performing the Custom Filtering Technique do not show any definitive signs of being forged. Specifically, these are Fogeries B.2 and B.6 in Appendix C. The outcome of the tampered area containing the man in Forgery B.2 does not have any “hidden” discontinuities or other abnormal signs of manipulation. Additionally, Forgery B.6 contains very little pixel information as it has been heavily compressed and has a very low resolution; thus its Custom Filtering result contains many anomalies and fuzzy areas. This makes the result of the forged portion not distinct in comparison with the rest of the image.

Forgery	Inconclusive Signs of Tampering	Possible Signs of Tampering (should run other techniques)	Definitive Signs of Tampering
B.1		X	
B.2	X		
B.3		X	
B.4			X
B.5		X	
B.6	X		
B.7		X	
B.WMno		X	
B.WMyes		X	

Table 4.3 – Summary of the Results of the Custom Filtering Technique

The majority of the results of the Custom Filtering Technique fall into the middle category, which shows possible signs of tampering. In particular, Forgeries B.1, B.3, B.5, B.7, B.WMno, and B.WMyes result in areas that have discontinuities or other irregularities. The result of Forgery B.3 provides a specific example. The forged shadow of the lighthouse in this picture results in an outline of "ghost" edges and other anomalies in portions of the "touched-up" white fence. Other areas of the fence do not exhibit this type of behavior and therefore raises suspicion of this region. Performing further tests using the other methods is necessary to verify this area is victim to image tampering.

Forgery B.4 is the only image to result in definitive signs of image tampering. The figure of "Yoda" returns results that are very distinct compared to the rest of the resulting image. An extremely thick outline of double edges draws "Yoda" and suggests that some re-sampling or other manipulation has occurred to merge this character into the background image. This technique is the justification needed to make a solid conclusion about the image that the previous two methods lack.

As suspected, the Custom Image Filtering Technique yields near identical results when performed on both the watermarked (Forgery B.WMyes) and non-watermarked (Forgery B.WMno) images. Again, since the *invisible* watermark alters the appearance of the host image very slightly its result when performing any type of filtering should be near identical to the original non-watermarked result.

In summary, filtering an image with the Custom filtering mask presented in Chapter 3 produces promising results. While it did fail in an extreme case dealing with a heavily compressed image, the results, for the most part, picked up on the tampered portion in the image. Most cases provide assurance in deciding if image tampering occurred, but warrant further analysis using the other proposed detection techniques. This method paves the way for further analysis based on the JPEG compression standard, which the next section discusses.

4.2.4 Analysis of the JPEG Block Technique

Section 3.2.3 discusses a technique that captures the “fingerprint” of the JPEG compression standard and uses it to detect image tampering. Appendix C includes the results of this method when applied to the test images found in Appendix B. Table 4.4 summarizes the results.

Forgery	Inconclusive Signs of Tampering	Possible Signs of Tampering (should run other techniques)	Definitive Signs of Tampering
B.1			X
B.2			X
B.3	X		
B.4			X
B.5			X
B.6	X		
B.7			X
B.WMno			X
B.WMyes			X

Table 4.4 – Summary of the Results of the JPEG Block Technique

The outcome of this technique is very promising. The test-bed in Appendix B includes a sample of the broad range of image forgeries typically found in the real-world. Forgery B.6 is included in this test-bed to represent a heavily compressed and low resolution example. When performed on this image, the JPEG Block Technique returns inconclusive signs of image tampering. This method fails because of the limited pixel information and the inadequate number of *8 x 8* compression blocks to analyze. Additionally, Forgery B.3 exhibits inconclusive results of forgery because the tampering in this image deals strictly with brightness differences. This type of forgery causes the JPEG Block Technique difficulties. While Appendix C shows the inconclusive results of both images, it does not represent the overall crux of this technique's results.

The remainder of the test images return definitive signs of image tampering when using the JPEG Block Technique for analysis. This method captures the forged area after using various threshold values for testing. As discussed in Chapter 3, testing is best performed with a threshold value equal to 50 with adjustments to follow in increments of 5 – 10 in both directions. Appendix C provides results using the most appropriate threshold value unique to each image. While several return noisy white patterns throughout the image when using a threshold value too low, after some tweaking the forgery becomes prevalent. Forgery B.7 provides an example of this phenomenon. Since this image contains large numbers of distinct edges and abrupt pattern changes, it returns a large number of noisy white patterns when using a low threshold value. Figure 4.2 shows the results of the JPEG Block Technique using a threshold value of 50. In this figure, a decisive pattern of the tampered area is not easily discovered because of the excessive noise returned from the prominent edges. The best threshold value in this case is 75, which the final result in Appendix C includes. As Chapter 3 discusses, the remaining white blocks returned when using an increasingly larger threshold value are usually the result of some form of image tampering. The larger threshold value effectively filters out the false positives caused by edges since tampering with an area on the image usually causes greater variability in the JPEG blocks. Consequently, if no pattern arises using different threshold values, the image is most likely authentic or requires analysis of the results of the other methods.

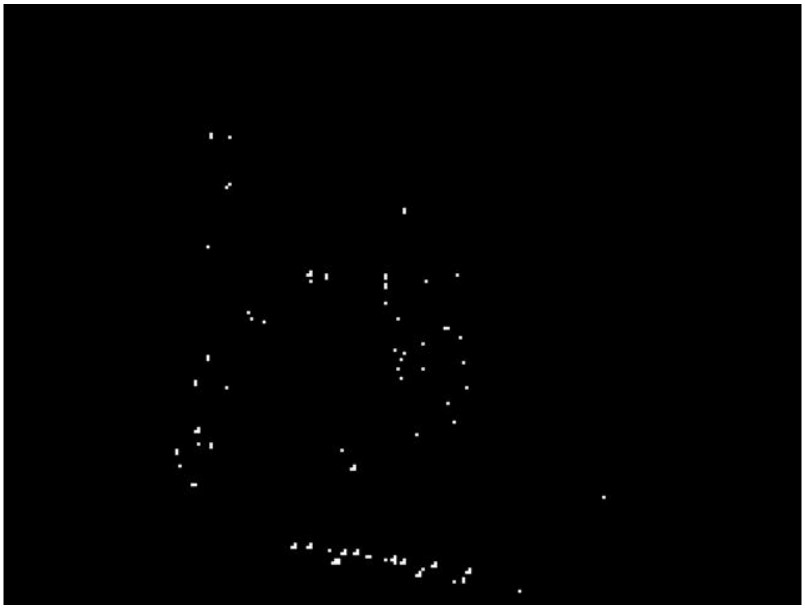


Figure 4.2 – JPEG Block Technique on Forgery B.7 w/ threshold 50

Sections 4.2.1 – 4.2.3 had a common consensus that the presence of a digital watermark did not have noticeable effect on the results of each test. The overall result of the JPEG Block Technique on both the watermarked (Forgery B.WMyes) and non-watermarked (Forgery B.WMno) images yield very similar results. It is interesting to note that these two resulting images are not exactly the same. Both contain slight variations in flagged *8 x 8* pixel blocks. The reason for this is because the watermarking algorithm resaves the image in the JPEG format once it embeds the watermark. This resaving causes the execution of the whole JPEG compression process with a second quantization table, Q^2, thus resulting in a slightly different copy of the image with different DCT information. Visually they are near identical images, but each *8 x 8* pixel block is modified slightly and therefore returns faintly different results when performing the JPEG Block Technique. However, both results still provide evidence of image tampering.

Additionally, a unique test performed on the JPEG Block Technique attempts to capture the results of creating a forgery using two JPEG images with differing Quality Factors. As Chapter 3 discusses, it is suspected that the greater the original Quality Factor difference is between merged images, the more distinctive the results from the JPEG Block Technique. Forgeries B.QF.100, B.QF.90, and B.QF.75 return similar results with the forged area revealing 4 or 5 white blocks. At the extreme end of the scale, the JPEG Block Technique returns 9 white blocks when performed on Forgery B.QF.0. With consistent threshold values used in this test, the data does support the hypothesis. While the results of this technique still return signs of image tampering for all levels of Quality Factor differences, the greater the difference does cause the JPEG Block Technique to return more positive signs of image tampering.

Overall, the JPEG Block Technique shows promise when used to test an image for tampering. Seven of the nine test images return results with definitive signs of image manipulation. The main factor for trouble with Forgery B.3 was that the tampering involved only changing an area’s brightness and shadow levels. The other image with poor results is the product of heavy compression and major resizing. This image only has a file size of 4.74 KB, therefore contains extremely narrow pixel information. If an image used for testing is small, heavily compressed, or been damaged or partially corrupted, chances are that this technique will have a hard time determining a tampered area. The other techniques analyzed in this chapter are alternative methods for testing an image for tampering, and for max robustness these other methods should be performed in conjunction with the JPEG Block Technique discussed here. A multilayered approach is the best practice one should follow when deciding if an image is forged or authentic.

As a side note, when testing an image for tampering, a low threshold value may provide the best evidence of digital tampering. A pattern of black blocks may be the indicator to look for. While the images in Appendix C use higher threshold values to reveal the tampered areas, other images may fear better if analyzed with a smaller threshold value, such as Figure 3.12. Thus, testing an image using a broad range of threshold values is the best policy.

4.3 Results of Blind Test

Each of the previous experiments analyzed images with previously known tampered portions. While these images are used for each experiment to stress the strengths and weaknesses of each method, they do open up debate about objectiveness. This is why a unique experiment is performed which includes a blind test of a mixture of 15 authentic and forged images. Overall, 6 of the 15 test images were found to be incorrectly identified. This included 2 of 15 identified as false positive and 4 of 15 identified as false negatives. Therefore, an overall observed accuracy of this experiment is 60% with a 13.33% false positive result and 26.67% false negative result.

The results of this experiment raise some important points about performing the proposed methods to detect image tampering. When performing each technique on an image of unknown origin, some subjective analysis is required of each method’s result. In the case of JPEG images with low Quality Factors, one has to determine if a flagged area is due to actual image tampering or if the high compression introduced the distortion.

Many images found on the web are heavily compressed and therefore this fact needs to be taken into consideration when analyzing the results of each method. Also, it is wise to get a second opinion of each result to aid in the decision making process. This helps to interpret the results of each method as well as lend another's perspective about the depicted scene in the image. For example, if the image in question portrays an aircraft flying in the air, it is beneficial to have the opinion of an aircraft expert aid in the decision process. One needs to ultimately determine if portions of the depicted scene existed when the image was taken or if they have been digitally altered by some other means. This experiment overall proved to be interesting and found a respectable accuracy percentage compared to deciding an image without the help of any detection methods.

4.4 Summary

This chapter presented an analysis of the results obtained from performing the experiments described in Chapter 3. The four methods discussed to detect image tampering were scrutinized to determine where each failed or succeeded at detecting the image forgeries in Appendix B. After analysis, the Luminance and HSV Level methods proved to be helpful when used in conjunction with the other methods. Each was by no means an end-all solution to detect locations of image tampering, if any exist. The Custom Filtering method verified that it was more successful than the previous two methods, but yielded results that required further testing. Finally, the JPEG Block Technique confirmed its robust ability to detect the broad range of image forgeries presented in Appendix B. In all but one of the results, definitive signs of image tampering could be concluded based in this technique. While this method is encouraging at detecting image forgeries, it should not be used by itself. Each method demonstrates strengths and weaknesses and is best suited to be used in conjunction with the other three. The only image forgery that gave each method trouble at deciding a firm conclusion was one that was heavily compressed and shrunk in resolution. By and large, the techniques work best when a higher resolution of pixels represents the image but in the real-world this is not always the case. Overall, the methods analyzed by this chapter prove to be hopeful in raising the bar on detecting image forgeries.

Chapter 5 : Conclusion

5.1 Summary

The research presented in this thesis analyzed the area of image forensics relating to the detection of digital image tampering. The current research in the detection of digital image tampering focused on several subclasses of image forgeries. Therefore, this thesis' research helped to move toward the broader goal of deciding if any given image is forged or, in fact, authentic. The goal of this research was to take the currently available image forgery detection methods and focus on where each method lacked. Detection of *copy-move* forgeries as well as image forgeries in uncompressed formats were the two areas where a promising detection tool already existed. Thus, the goal of this research focused on *copy-create* image forgeries in addition to a method tailored to the lossy JPEG image format. Subsequently, the other three methods developed in this thesis work on any digital image due to their specialization in fundamental attributes of any digital image, such as color or brightness analysis. To conclude the research of this thesis, a thorough experiment and blind test was performed to test the overall detection accuracy of these four methods.

5.2 Conclusions of Research

The research performed in this thesis ended with the development and testing of four forgery detection techniques. Each method focuses on image attributes with small anomalies in addition to other discrepancies introduced by tampering. Overall, a detection accuracy of 60% was observed when performing a blind experiment containing an unknown mixture of 15 authentic and forged JPEG images. Detection accuracy was found to be heavily dependent on the amount of time spent analyzing the results of each method as well as any pre-existing tampering knowledge of the image in question.

The research of this thesis concludes that no one technique is best suited to detect every given image forgery. Much uniqueness lies in the creativity and effort of the forger and thus there are an infinite number of possibilities to create, alter, and digitally manipulate any given image. Also, the accuracy of a detection method is influenced by the amount of compression, and subsequent recompression, as well as the file size of the image in question. Testing has concluded that the amount of false positives introduced into a given image increases as the resolution and file size decreases. This phenomenon most influences the JPEG Block Technique. Overall, these methods prove to be beneficial to the research community and hope to spark the ideas of new and unique forgery detection methods.

5.3 Recommendations for Future Research

Digital image forensics is a research area which is in its infancy stages. While most research emphasizes on digital watermarking and other ways to prevent tampering from occurring, the area explored by this thesis looks at the situation when an image's authenticity needs to be verified in absence of any prior watermarking technique. Designing new detection methods, in addition to the four discussed here, is a viable extension of this thesis. An analysis of other fundamental image attributes may help to improve detection accuracy as well as increase robustness to new and creative ways in digitally manipulating an image. The reassuring characteristic of this research area is the ability for a researcher to be creative in designing new detection methods.

5.4 Closing Remarks

In conclusion, the detection of image tampering relies on one very big assumption; the tampering performed by a forger introduces some detectable anomaly. This can be some inconsistent color or brightness pattern, abnormal edge, or other small discrepancy introduced as a by-product of image tampering. Cases in which an individual spends copious amounts of time sculpting each individual pixel to ensure a fully believable forgery are instances in which any forgery detection method would have difficulty detecting a fraud. However, this thesis explores best practices in the detection process and recommends an inclusive layered approach. An image viewed originating from an unknown source is sometimes the only instance one has of a digitally captured scene. Is the depicted scene actually authentic? As the Greek Philosopher Plato (427 – 347 B.C.) once said, “Science is nothing but perception.”

Appendix A: MATLAB Source Code

```
% Implementation of Luminance Level Technique
function LuminanceLevelTechnique(imagearg,thresh)

    % Reads in image
    image = imread(imagearg);
    % Threshold and return Binary Image
    image = im2bw(image,thresh);
    % Displays resulting Binary Image
    imshow(image);


%_______________________________________________________________________


% Implementation of HSV Technique
function HSVTechnique(imagearg)


    % Reads in image
    image = imread(imagearg);

    % Checks if image is in color format, if not it errors
    [m,n,z] = size(image);
    if(z == 1)

        errordlg('Image must be Color for this Technique');

    else

        % Converts to HSV color-space
        image = rgb2hsv(image);
        % Displays resulting Image
        imshow(image);

    end


%_______________________________________________________________________


% Implementation of Custom High Pass Filtering method
function CustomHighPassFiltering(imagearg)

    image = imread(imagearg);

    [m,n,z] = size(image);

     % Checks if image is in grayscale format, if not it is converted
    if(z == 3)
        image=double(rgb2gray(image));
```

```
else
    image=double(image);
end

imageresult = image;


% Filters Image using mask:
% [-1  -2  -1]
% [-2  12  -2]
% [-1  -2  -1]
  j = 1;
w = waitbar(0);
  while(j < m)
    % Wait bar to let user know ETA
    waitbar(j/m,w, 'Please Wait...');
        i = 1;
        while(i < n)
        % Counter in special cases where mask is not with 9 pixels
        % (i.e. edges)
        counter = 0;

        % Below is used to check to see if subscripts are within
        % bounds
        % of image to prevent errors
        j_minus_1 = (j-1);
        i_minus_1 = (i-1);
        j_plus_1 = (j+1);
        i_plus_1 = (i+1);

        % Top-left pixel ( -1 in mask )
        if(( j_minus_1 < 1) || (i_minus_1 < 1))
            image1 = 0;
        else
            image1 = (image( (j-1), (i-1) ))*-1;
            counter = counter + 1;
        end

        % Top pixel ( -2 in mask )
        if( ( j_minus_1 < 1) )
            image2 = 0;
        else
            image2 = (image( (j-1), i ))*-2;
            counter = counter + 1;
        end

        % Top-right pixel ( -1 in mask )
        if( ( j_minus_1 < 1) || ( i_plus_1 < 1) )
            image3 = 0;
        else
            image3 = (image( (j-1), (i+1) ))*-1;
            counter = counter + 1;
        end

        % Left-pixel ( -2 in mask )
```

```
if( ( i_minus_1 < 1) )
    image4 = 0;
else
    image4 = (image( j, (i-1) ))*-2;
    counter = counter + 1;
end

% Center pixel ( 12 in mask )
image5 = (image( j, i ))*12;
counter = counter + 1;

% Right pixel ( -2 in mask )
if( ( i_plus_1 < 1) )
    image6 = 0;
else
    image6 = (image( j, (i+1) ))*-2;
    counter = counter + 1;
end

% Bottom-left pixel ( -1 in mask )
if(( j_plus_1 < 1) || (i_minus_1 < 1))
    image7 = 0;
else
    image7 = (image( (j+1), (i-1) ))*-1;
    counter = counter + 1;
end

% Bottom pixel ( -2 in mask )
if( ( j_plus_1 < 1) )
    image8 = 0;
else
    image8 = (image( (j+1), i ))*-2;
    counter = counter + 1;
end

% Bottom-right pixel ( -1 in mask )
if(( j_plus_1 < 1) || (i_plus_1 < 1))
    image9 = 0;
else
    image9 = (image( (j+1), (i+1) ))*-1;
    counter = counter + 1;
end

% Summation of values
value = (image1 + image2 + image3 + image4 + image5 + ...
         image6 + image7 + image8 + image9);

% Divide by number of elements summed
value = value / counter;


imageresult( j, i ) = value;


      i=i+1;
```

```
            end

            j=j+1;

    end

    % Close Wait bar
    close(w);

    % Coverts image back to 8bit unsigned integer(required for MATLAB
    % to
    % interpret matrix as an image
    imageresult = uint8(imageresult);

    imshow(imageresult);


%_____________________________________________________________________


% Implementation of JPEG Block Technique
function JPEGBlockTechnique(imagearg,thresh)

    image = imread(imagearg);

    % Checks if image is in grayscale format, if not it is converted
    [m,n,z] = size(image);
    if(z == 3)
        image=double(rgb2gray(image));
    else
        image=double(image);
    end

    % Calculates differences at 8x8 block edges
      j = 1;
    w = waitbar(0);
      while((8*j) < m)
        % Wait bar to let user know ETA
        waitbar((8*j)/(3*m),w, 'Please Wait...');
            i = 1;
            while((8*i) < n)

                  value = image( (8*j),(8*i) ) - ...
                    image( (8*j), ((8*i)+1) ) - ...
                    image( ((8*j)+1), (8*i) ) + ...
                    image( ((8*j)+1), ((8*i)+1) );

            % Sets all 64 pixels in block equal calculated value
            a=((8*(j-1))+1);
                  while(a <= (8*j))

                        b=((8*(i-1))+1);
                        while(b <= (8*i))

                              image(a,b) = value;
```

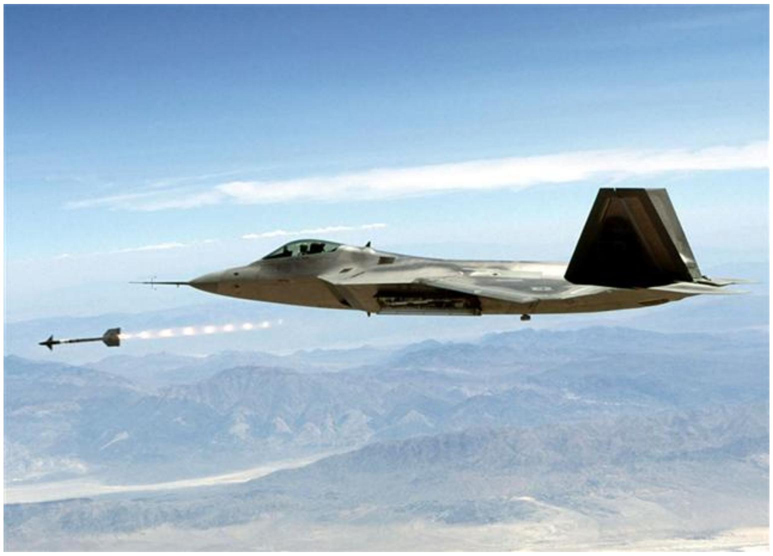
```
                          b=b+1;

                    end

                    a=a+1;

              end

              i=i+1;
        end

        j=j+1;

end

% Used for overall Wait bar
cnt = (8*j);

% Checks differences and thresholds them (left to right OR up and
% down)
j = 1;
  while((8*j) < m)
    % Update Wait bar
    waitbar((cnt + (8*j))/(3*m),w, 'Please Wait...');
        i=1;
        while((8*i) < n)

             difflr = abs(image( (8*j),(8*i) ) - ...
                      image( (8*j), ((8*i)+1) ));
             diffud = abs(image( (8*j),(8*i) ) - ...
                      image( ((8*j)+1), (8*i) ));

             if((difflr >= thresh) || (diffud >= thresh))

                 %sets all 64 pixels in block to white (255)
                 a=((8*(j-1))+1);
                       while(a <= (8*j))

                            b=((8*(i-1))+1);
                            while(b <= (8*i))

                                  image(a,b) = 255;

                                  b=b+1;

                    end

                    a=a+1;

                      end

            end

              i=i+1;
        end

        j=j+1;

end
% Used for overall Wait bar
cnt = ((2*8)*j);

% Sets all nonwhite blocks equal to 0
j = 1;
  while((8*j) < m)
    % Update Wait bar
    waitbar((cnt + (8*j))/(3*m),w, 'Please Wait...');
        i = 1;
        while((8*i) < n)

          if(image( (8*j),(8*i) ) ~= 255 )

            a=((8*(j-1))+1);
                  while(a <= (8*j))

                    b=((8*(i-1))+1);
                    while(b <= (8*i))

                          image(a,b) = 0;

                          b=b+1;

                    end

                    a=a+1;

                  end

        end

              i=i+1;
        end

        j=j+1;

  end

% Cleans up right border
  i = 1;
  while((8*i) < n)
        i=i+1;
  end

i=i-1;

a = 1;
while(a <= m)

    b=1;
    while((8*(i-1)+b) <= n)
        image(a, (8*(i-1)+b)) = 0;

        b=b+1;

    end
    a=a+1;

end

% Sets j to bottom pixel row for next loop
j=1;
  while((8*j) < m)
        j=j+1;
  end

% Cleans up next to last row
a = 1;
while(a <= n)

    b=1;
    while((8*(j-2)+b) <= m)
        image((8*(j-2)+b), a) = 0;

        b=b+1;

    end
    a=a+1;
end

% Cleans up last row
a = 1;
while(a <= n)

    b=1;
    while((8*(j-1)+b) <= m)
        image((8*(j-1)+b), a) = 0;

        b=b+1;

    end
    a=a+1;

end
% Close Wait bar
close(w);

% Coverts image back to 8bit unsigned integer(required for MATLAB
% to
% interpret matrix as an image
image = uint8(image);

imshow(image);
```

Appendix B: Images Used for Experiments

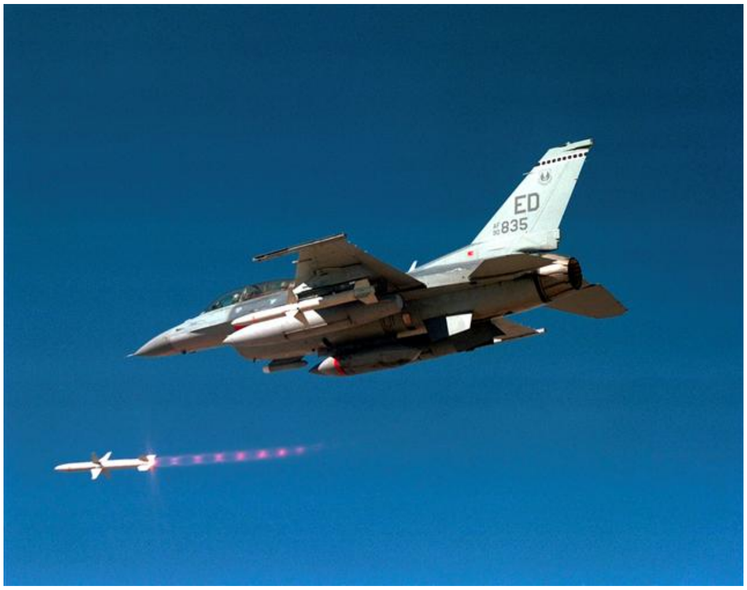
This Appendix includes the images used to test the image forgery techniques discussed in Chapter 3. A listing of the image format, resolution, and file size are included below the picture. A short description of the image forgery is also included. Assume the source of an image came from writer's digital camera (Fuji FinePix A303) unless otherwise noted.

Image Forgery B.1

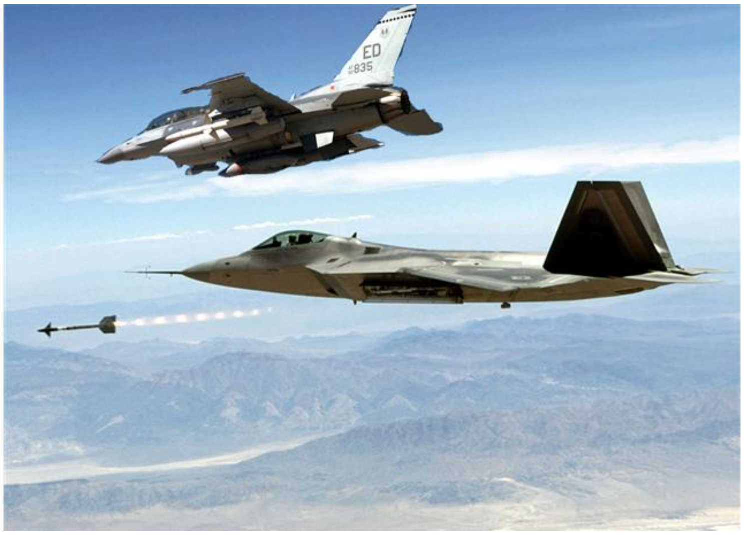
Description: Two digital pictures of different aircraft are taken and merged together to form a forged image.



Original JPEG Image – 1200 x 860 – 113 KB
Source: http://www.usu.edu/afrotc/pics.htm



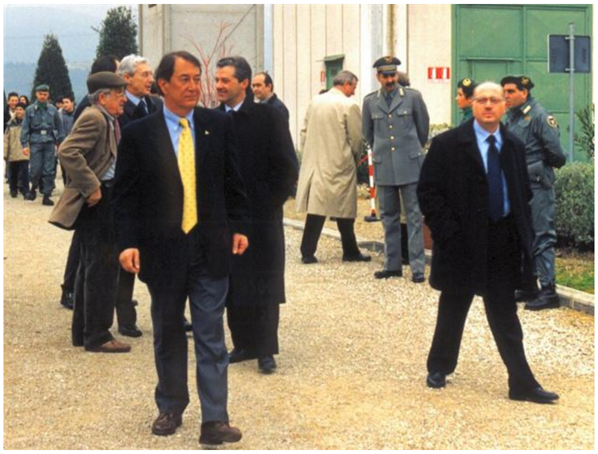
Original JPEG Image – 1273 x 1000 – 405 KB
Source: http://www.usu.edu/afrotc/pics.htm



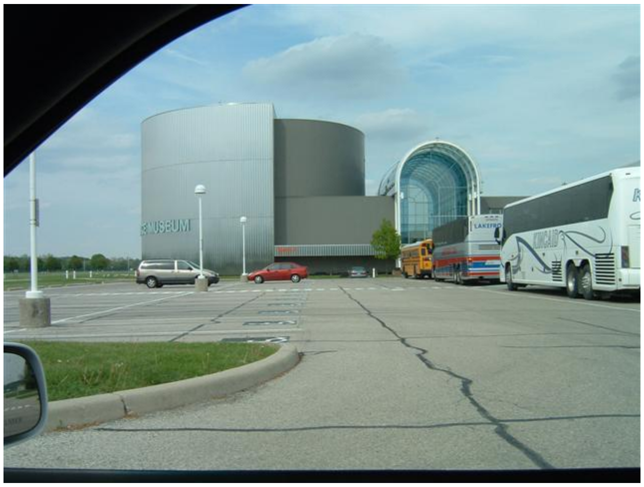
Forged JPEG Image – 1200 x 860 – 128 KB

Image Forgery B.2

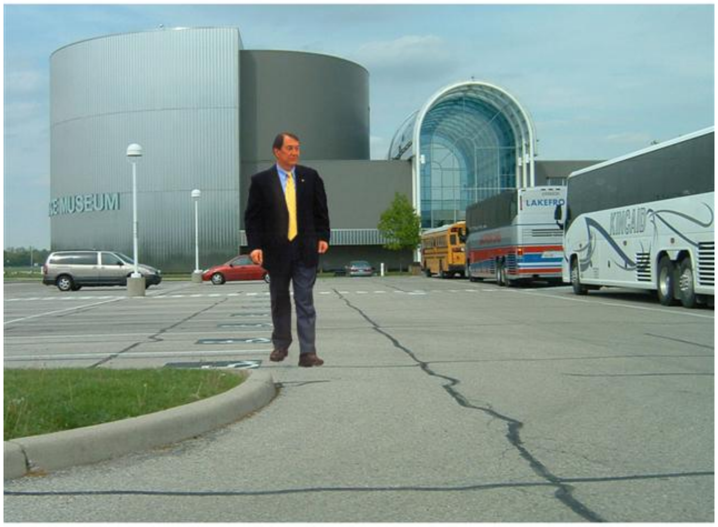
Description: A man from a digital image containing various people is taken and pasted into an image of a parking lot.



Original JPEG Image – 800 x 600 – 120 KB
Source: Google Image Search



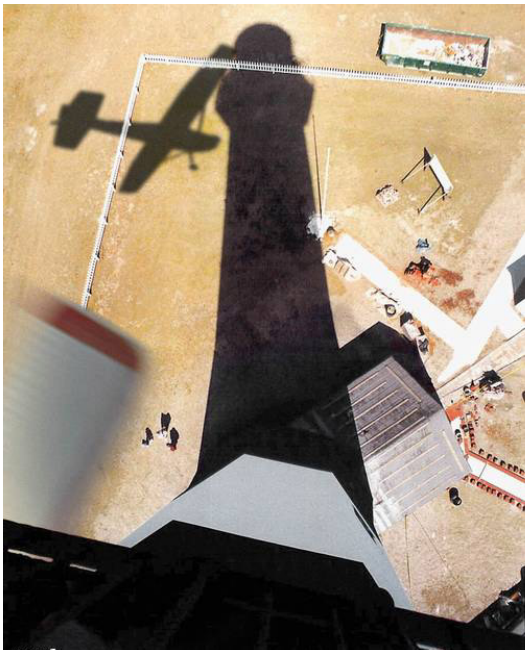
Original JPEG Image – 2048 x 1536 – 1.18 MB



Forged JPEG Image – 1600 x 1200 – 289 KB

Image Forgery B.3

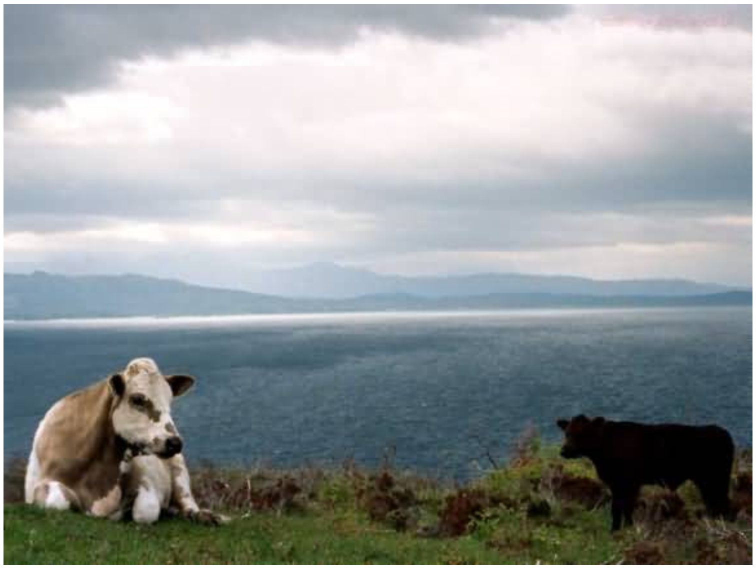
Description: A forged image depicts the impending crash of a plane into a lighthouse. The shadow of the lighthouse was digitally added to this picture by darkening the ground area with image manipulation software.



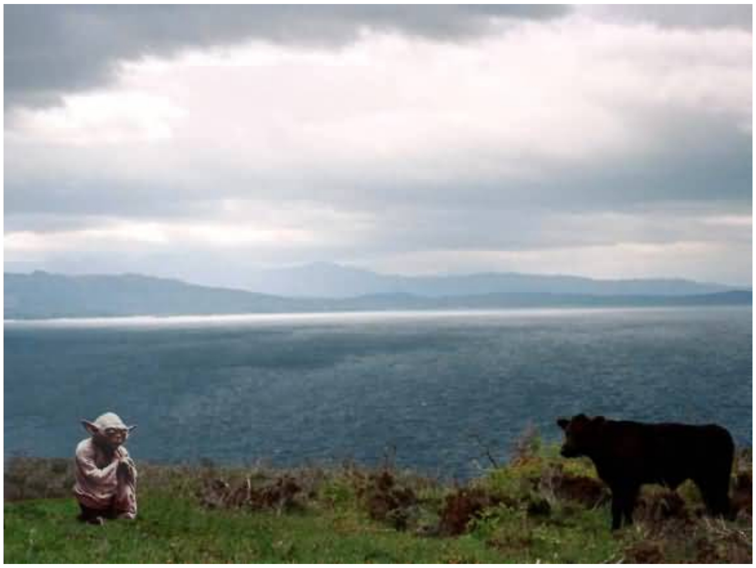
Forged JPEG Image – 500 x 620 – 82.3 KB
Source: web

Image Forgery B.4

Description: The host image for this forgery is one that shows two cows in a grassy area with water in the background. The forged image is the original host image with the cow on the left removed and the character "Yoda" put in its place.



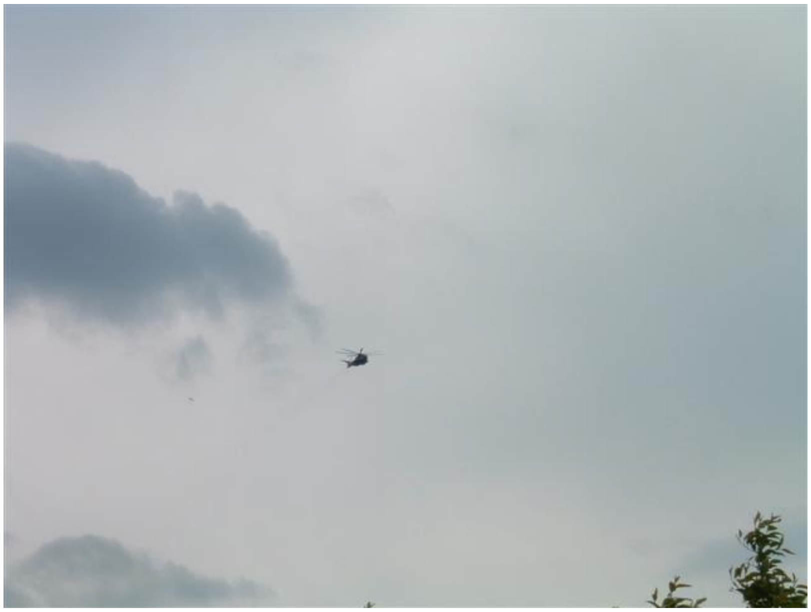
Original JPEG Image – 580 x 435 – 17.4 KB
Source: http://www.morrice.info/galleries/manipulated_cows.html



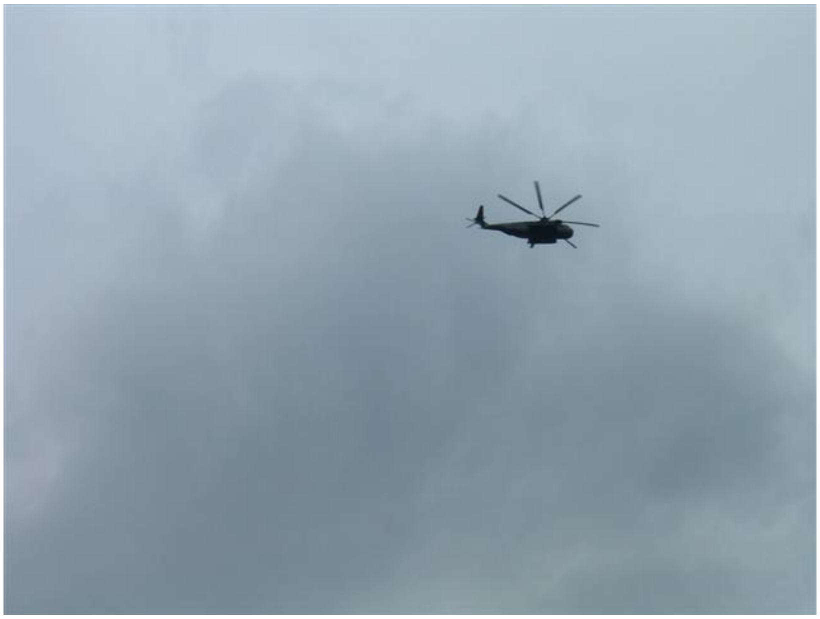
Forged JPEG Image – 580 x 435 – 16.1 KB
Source: http://www.morrice.info/galleries/manipulated_cows.html

Image Forgery B.5

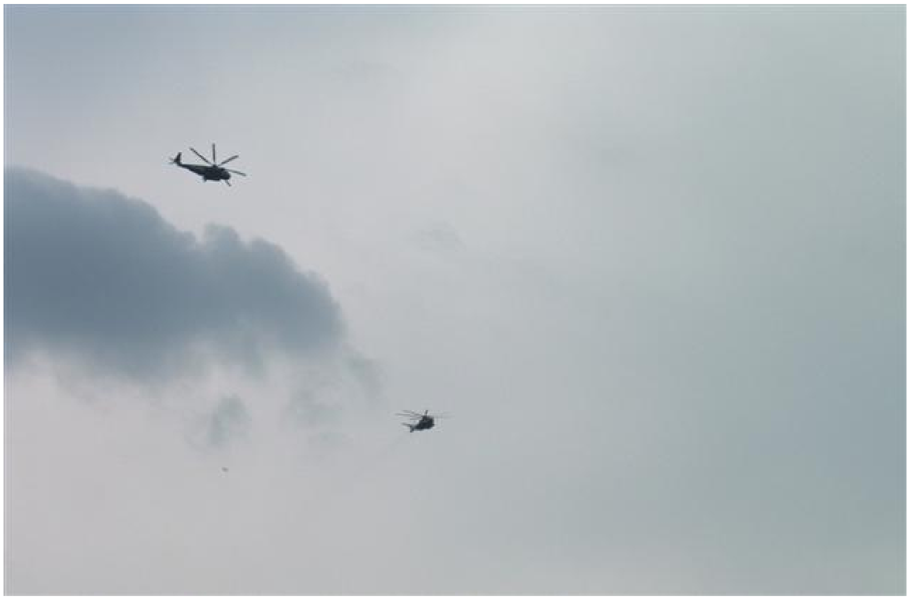
Description: Two images depicting a helicopter in the sky are taken and merged to create a forgery containing both helicopters.



Original JPEG Image – 2048 x 1536 – 0.99 MB



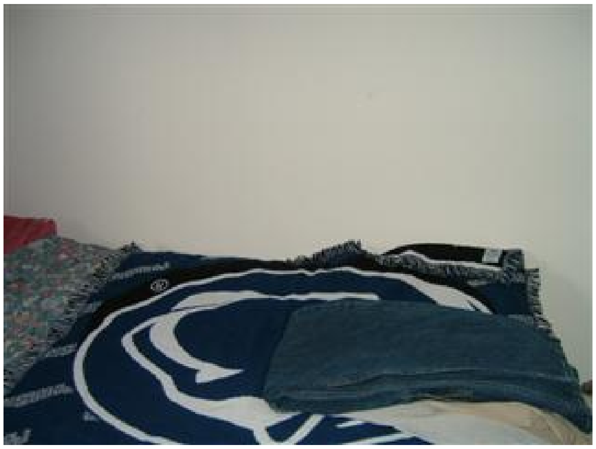
Original JPEG Image – 2048 x 1536 – 1.01 MB



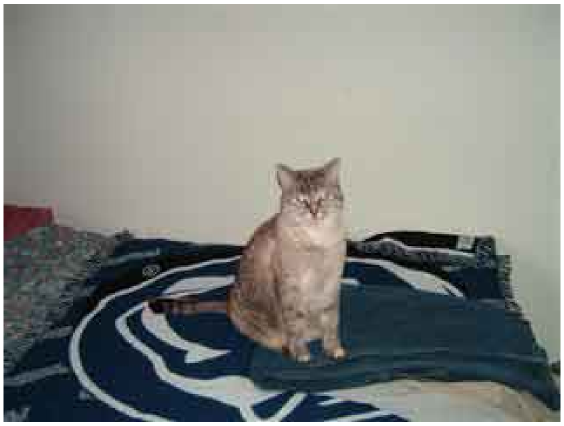
Forged JPEG Image – 1946 x 1278 – 536 KB

Image Forgery B.6

Description: This is an example of an image that has a very small resolution and file size, thus containing limited pixel information. A picture of a bed is shown to be the original with a cat being placed on it to form a forgery.



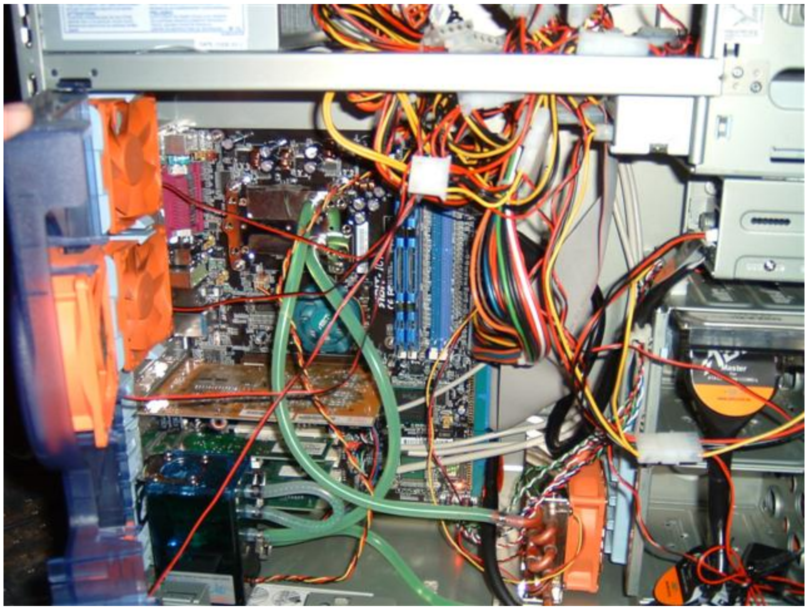
Original JPEG Image – 320 x 240 – 9.17 KB



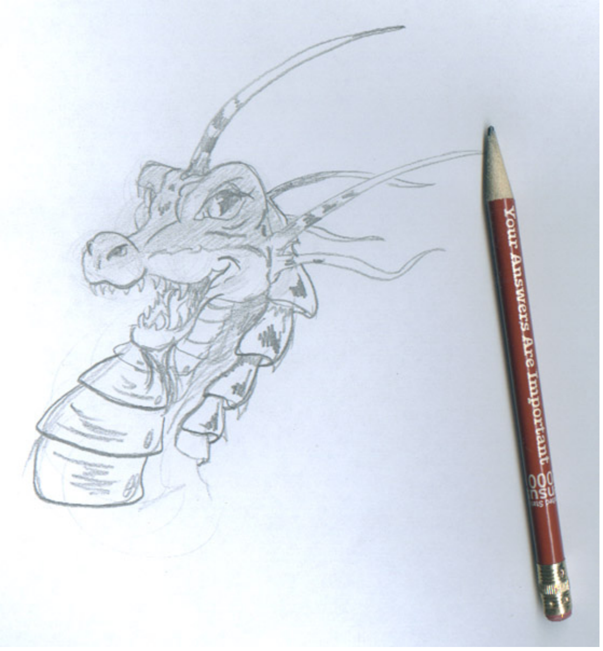
Forged JPEG Image – 320 x 240 – 4.74 KB

Image Forgery B.7

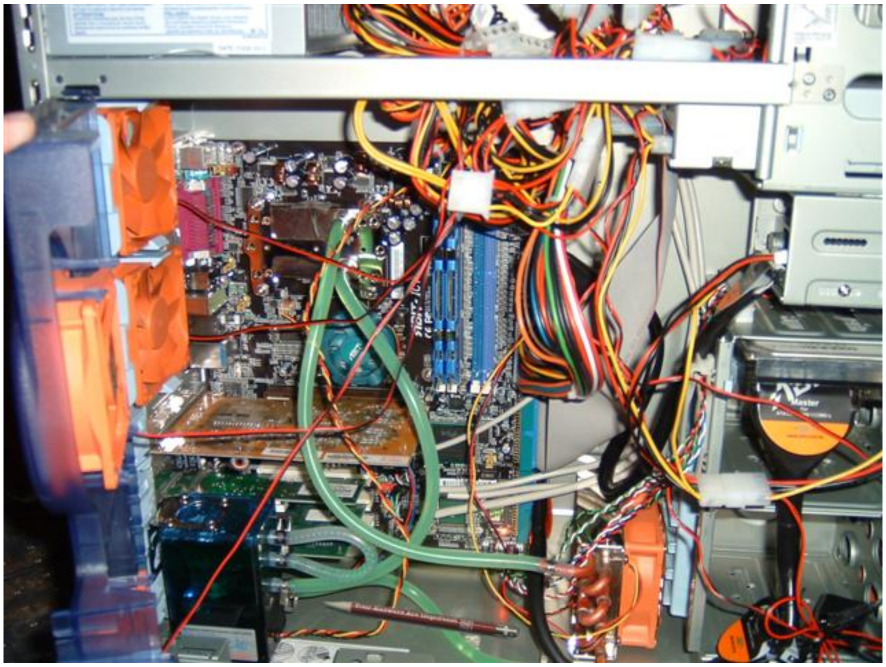
Description: This example is used to test how the proposed forgery detection techniques hold up against an image with a lot of varying colors and edges. In this example, a picture showing the inside of a computer case is used as a host image. A pencil is taken and pasted into the host image to make it appear as if it is lying in the bottom of the case.



Original JPEG Image – 2048 x 1536 – 1.23 MB



Original JPEG Image – 556 x 600 – 62 KB
Source: http://dragonneo.com/malathar/rough/behirhead-malsketch-pencil-r.jpg

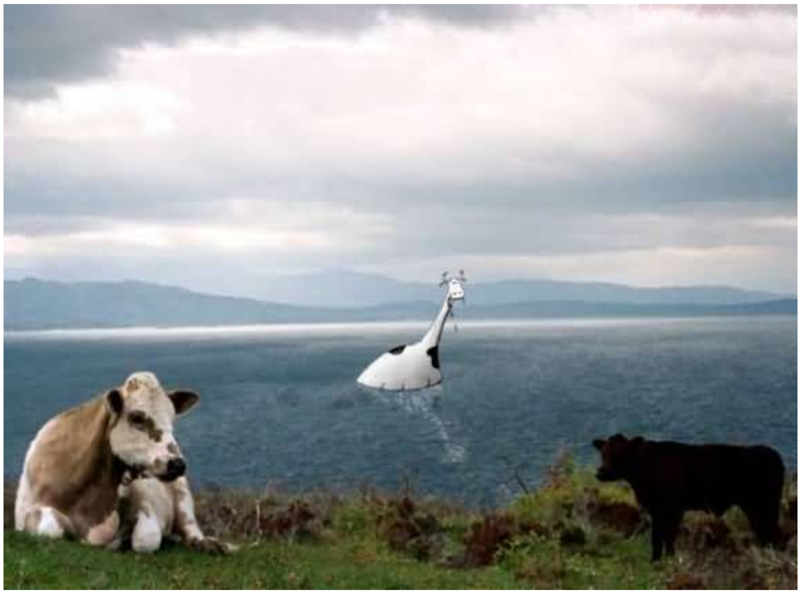


Forged JPEG Image – 2048 x 1536 – 709 KB

Image Forgery B.WM

Description: This example is presented to determine what effects an *invisible* watermark has on the results of each detection method. The host image is similar to that in Forgery B.4 with a scene depicting two cows on a grassy area with water in the background. This original image is forged with a monster placed on the water. The text used in the watermarking process is given below with the resulting forged image containing the watermark also presented. The software used to embed the hidden watermark is Steganography Software *F5* version 11+ discussed in Section 2.2.

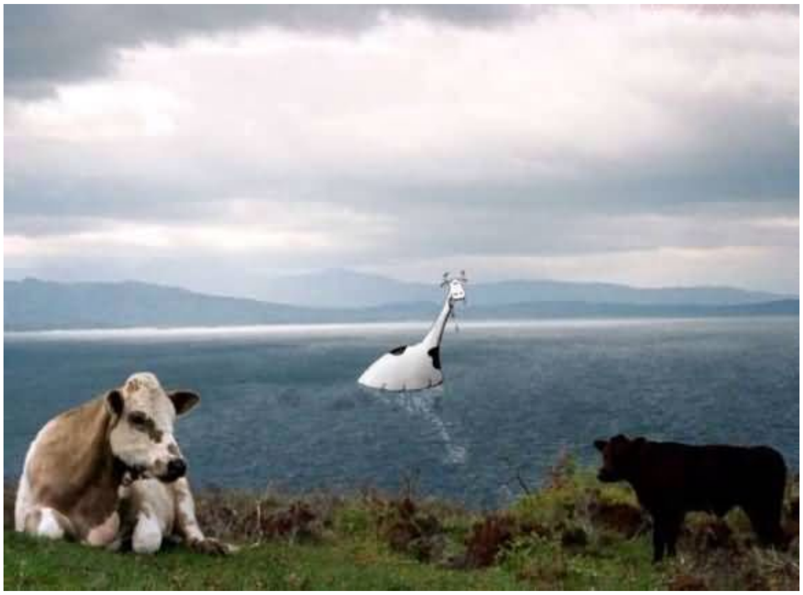
Original JPEG Image – 580 x 435 – 17.4 KB
Source: http://www.morrice.info/galleries/manipulated_cows.html



Forged JPEG Image **without** Watermark – 580 x 427 – 17.1 KB
Source: http://www.morrice.info/galleries/manipulated_cows.html

Text used for watermarking:

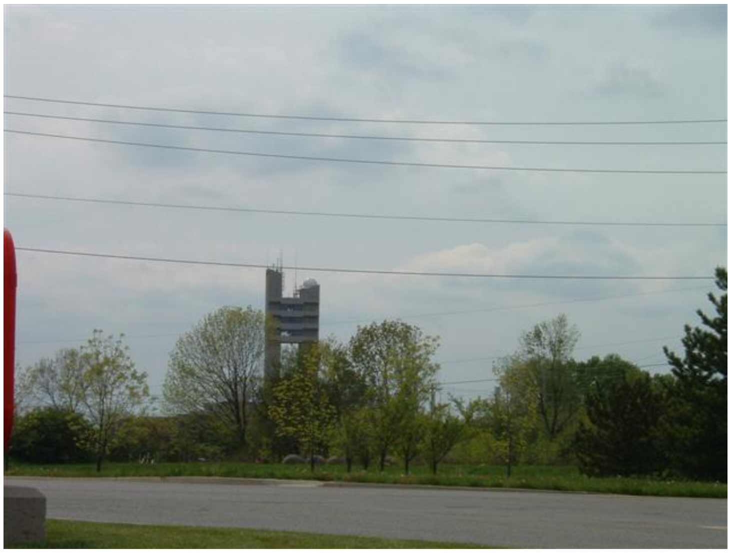
This is text used to embed into the forged image! It is used to simulate an invisible watermark!



Forged JPEG Image **with** Watermark – 580 x 427 – 21.4 KB

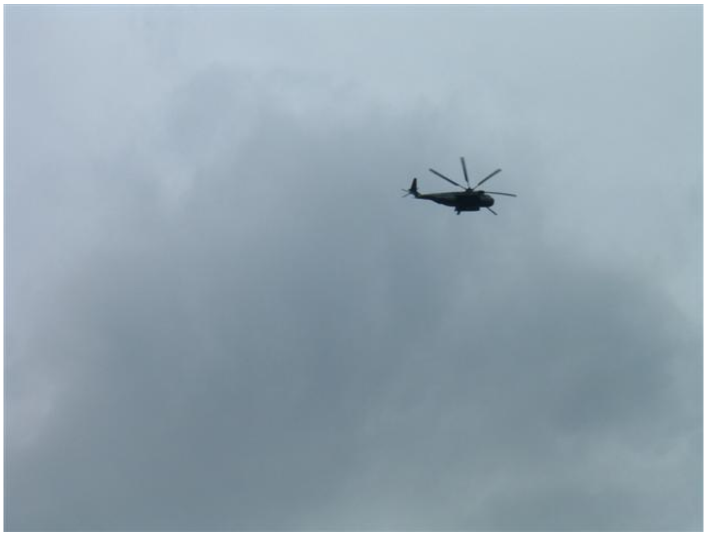
Image Forgery B.QF

The following is a set of images made up of pieces of another image with varying JPEG Quality Factors. The following image is the original test image:

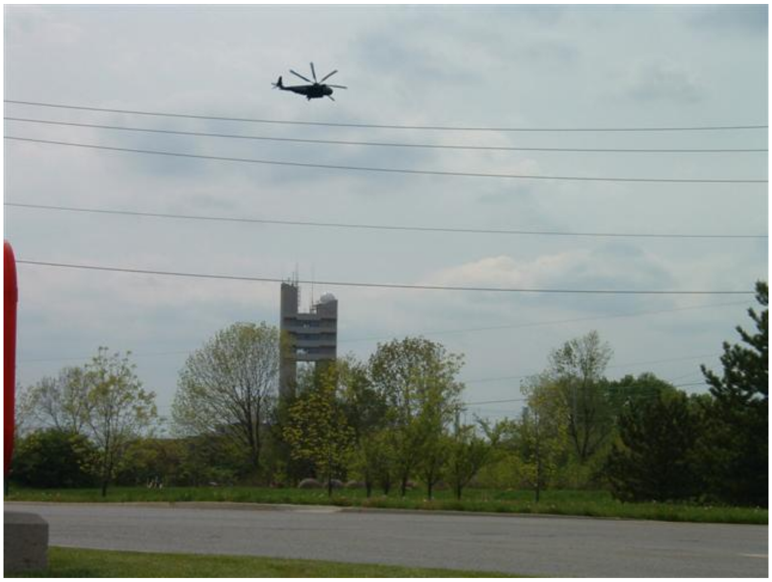


Original JPEG Image – 2048 x 1536 – 1.04 MB

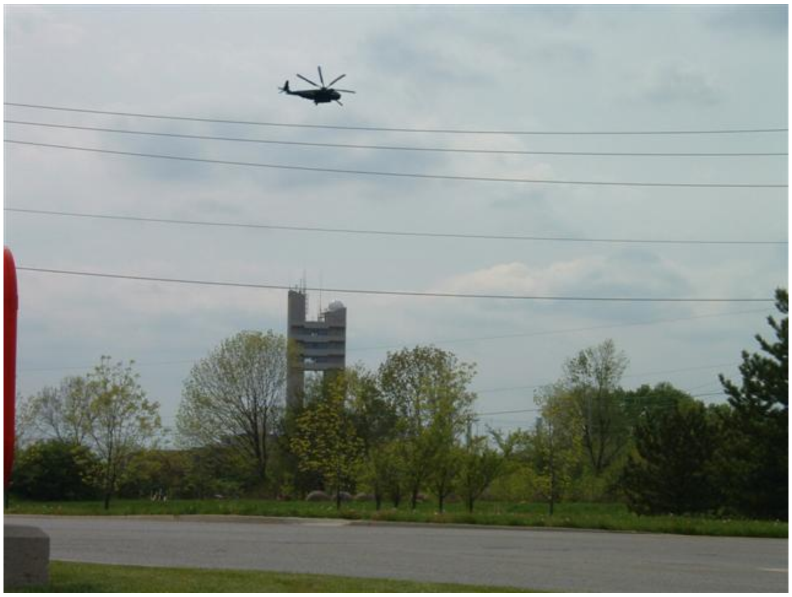
The helicopter from the following image was saved with various Quality Factors and then used to create the forgery.



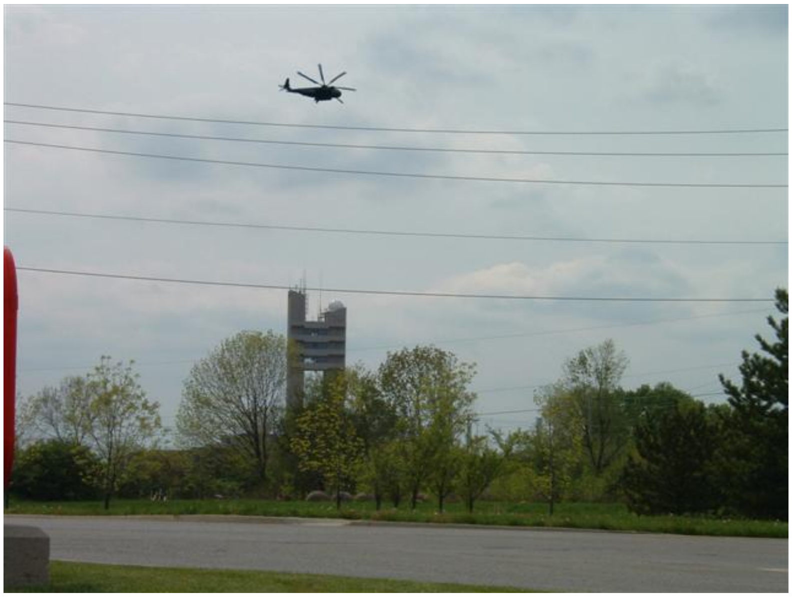
Original JPEG Image – 2048 x 1536 – 1.01 MB



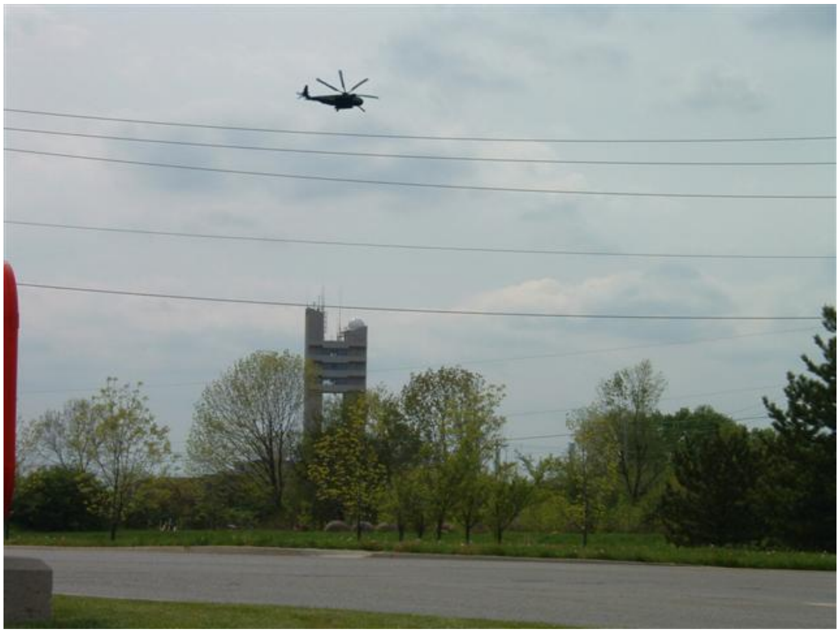
Forged JPEG Image with Helicopter’s QF = **100** – 2048 x 1536 – 936 KB



Forged JPEG Image with Helicopter’s QF = **90** – 2048 x 1536 – 936 KB



Forged JPEG Image with Helicopter’s QF = **75** – 2048 x 1536 – 936 KB

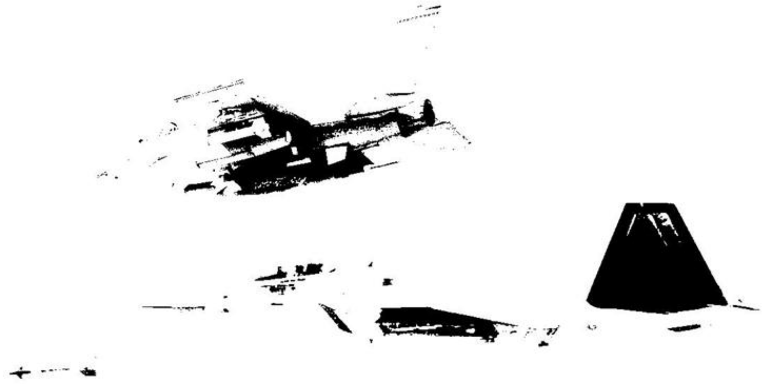


Forged JPEG Image with Helicopter’s QF = **0** – 2048 x 1536 – 936 KB

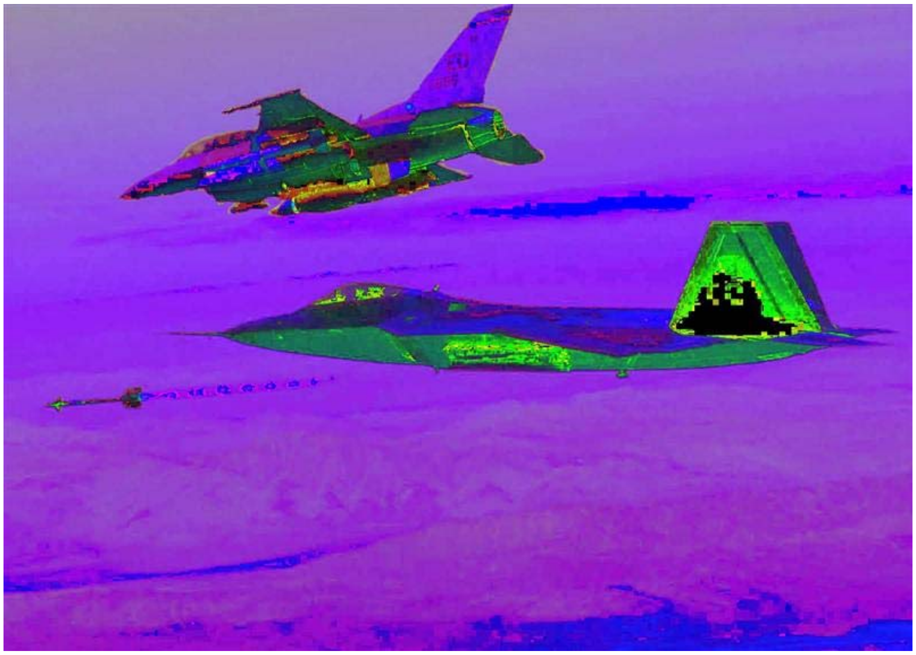
Appendix C: Image Results of Proposed Detection Techniques

This Appendix includes the image results of performing the proposed detection techniques on the test images from Appendix B. Given below the image is a description of the technique along with the threshold value used in the Luminance and JPEG Block Techniques.

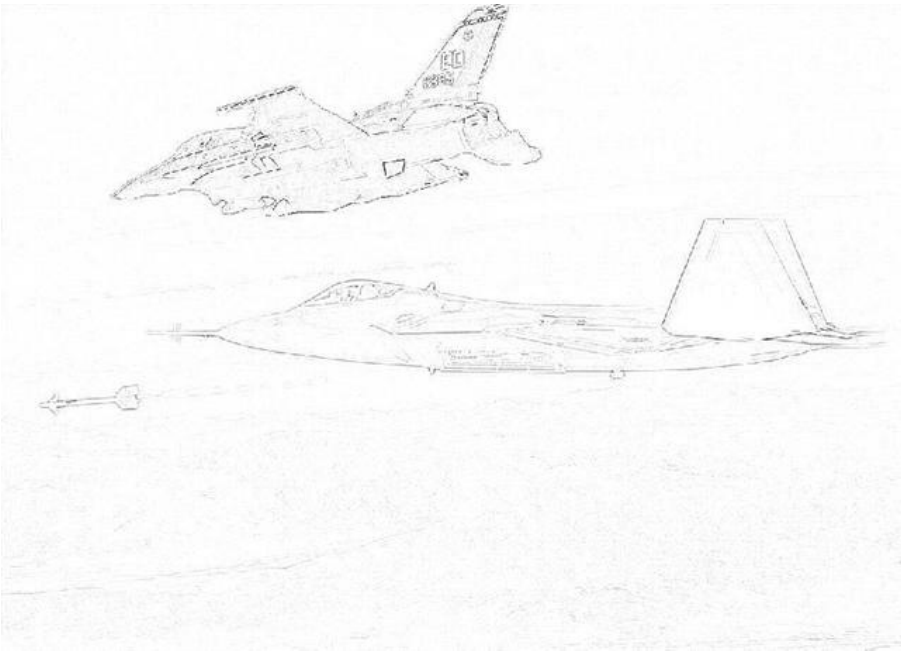
Results of Image Forgery B.1



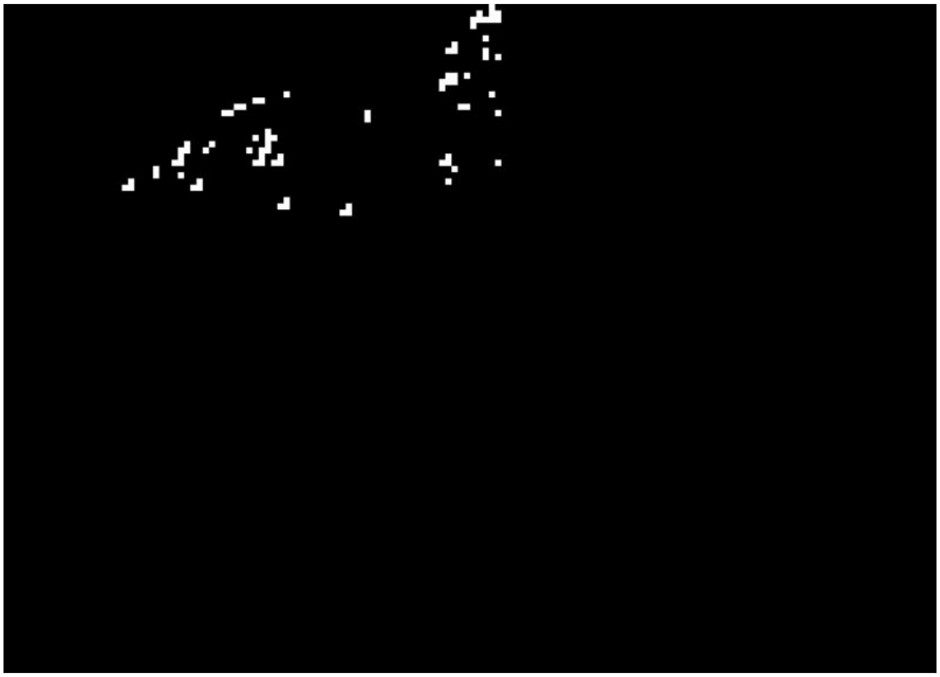
Luminance Level Technique w/ threshold 0.2



HSV Level Technique

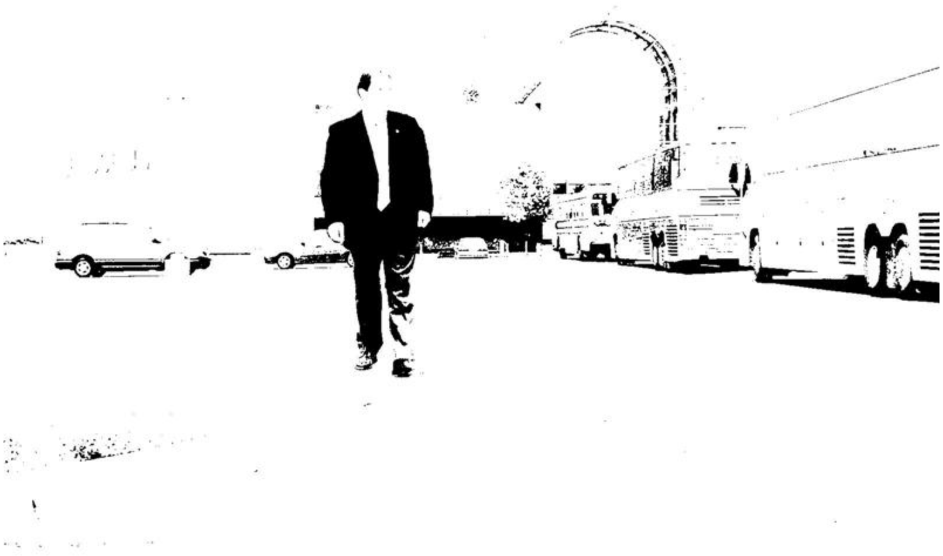


Custom Filtering Technique (inverted for readability)

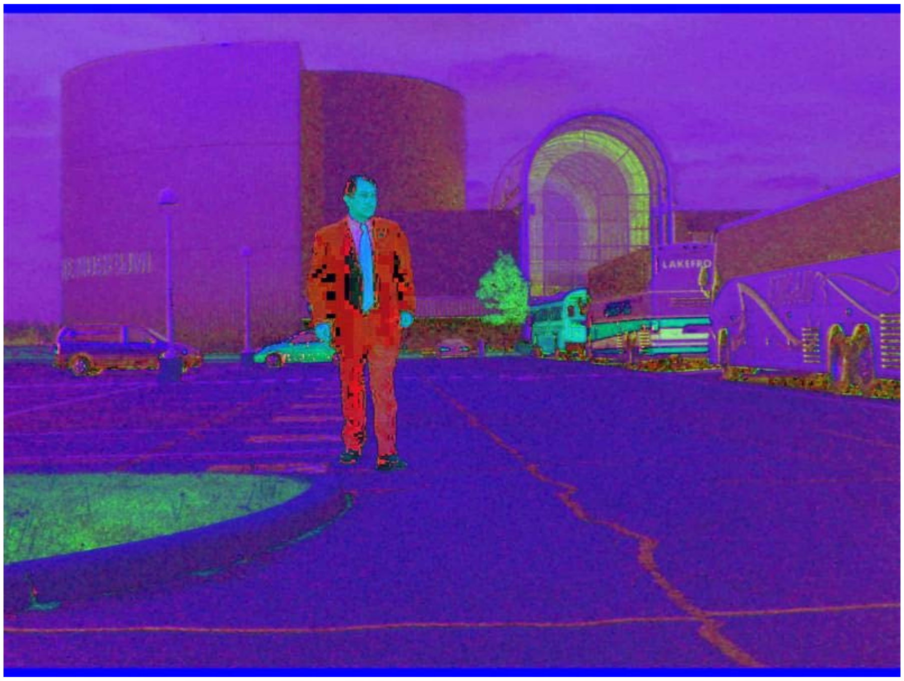


JPEG Block Technique w/ threshold 55

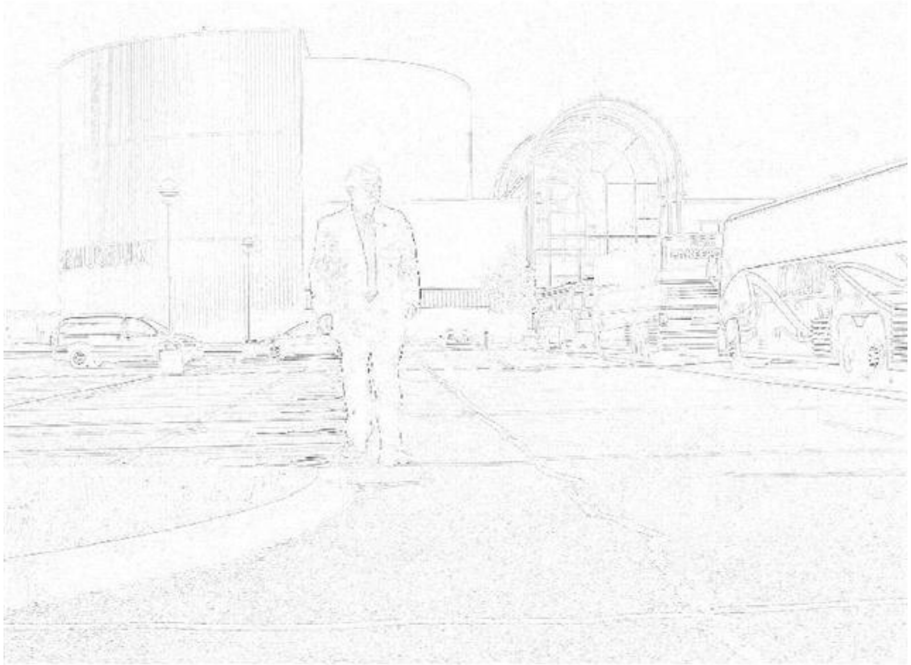
Results of Image Forgery B.2



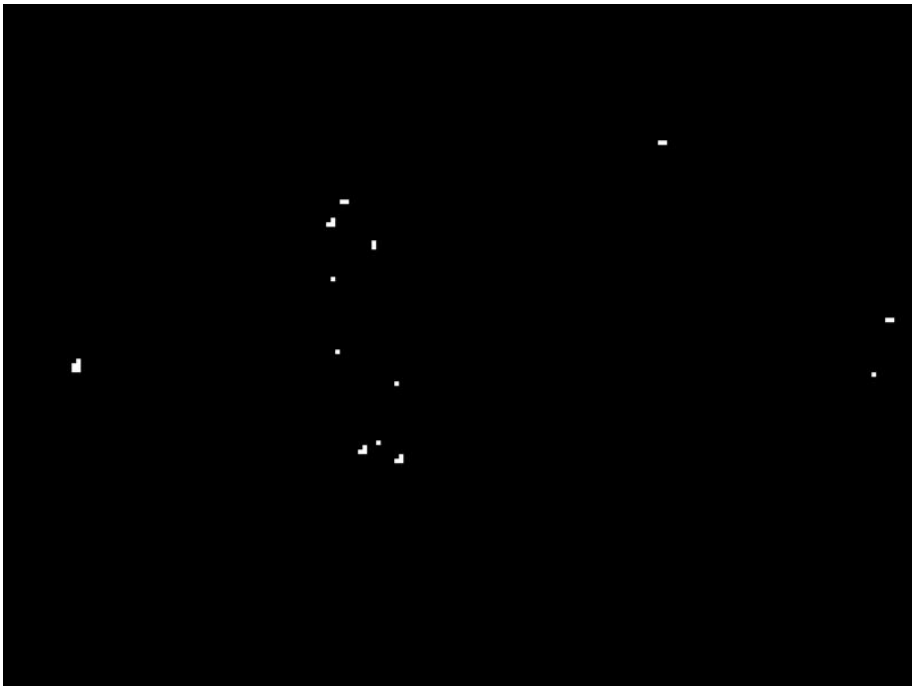
Luminance Level Technique w/ threshold 0.25



HSV Level Technique

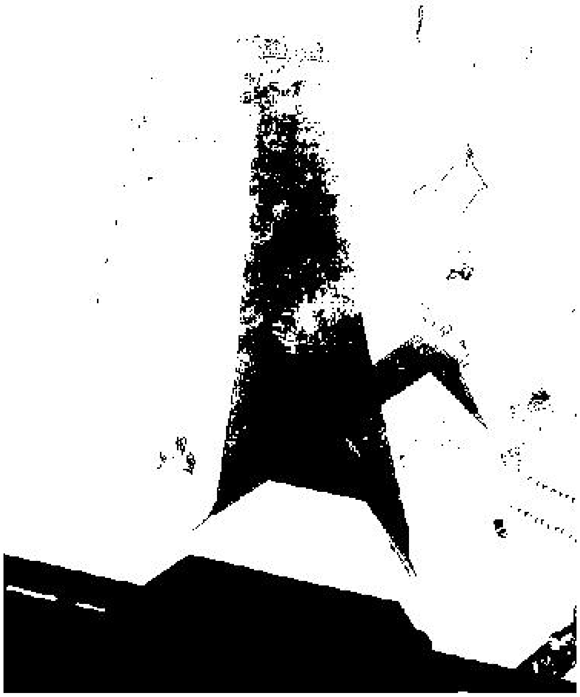


Custom Filtering Technique (inverted for readability)

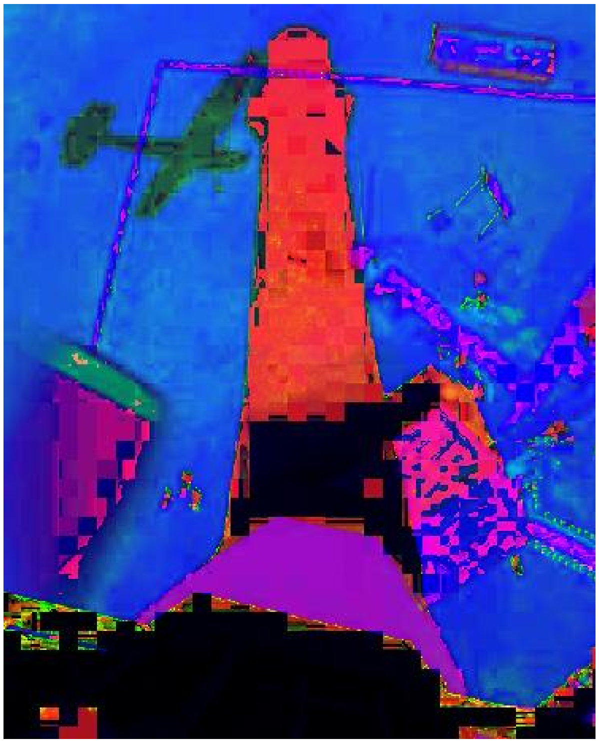


JPEG Block Technique w/ threshold 55

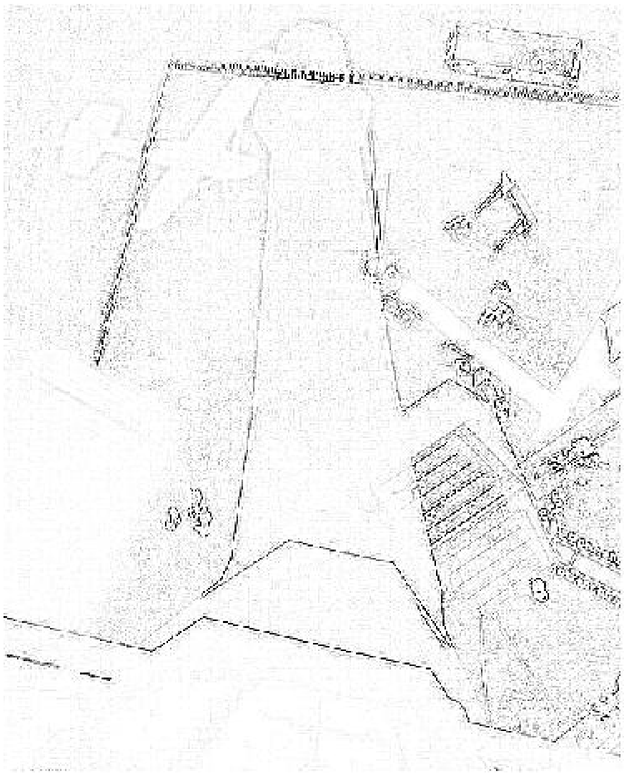
Results of Image Forgery B.3



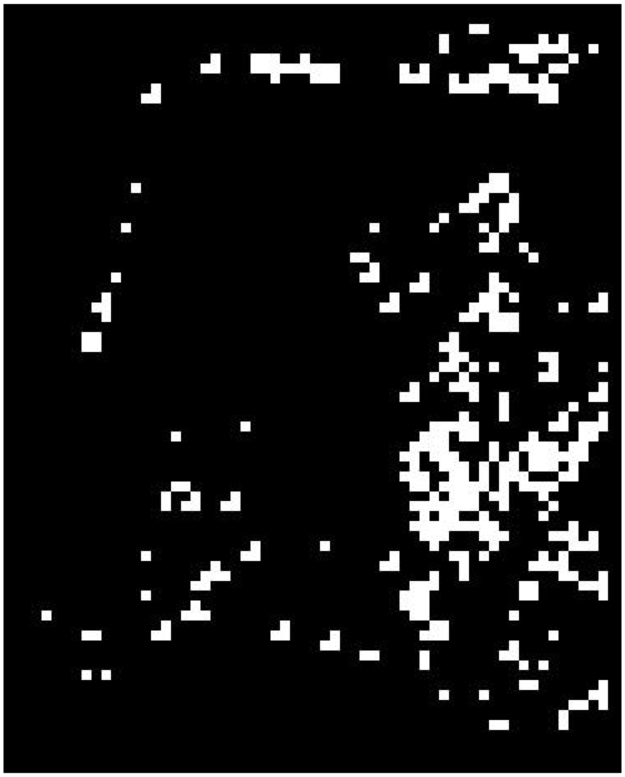
Luminance Level Technique w/ threshold 0.20



HSV Level Technique



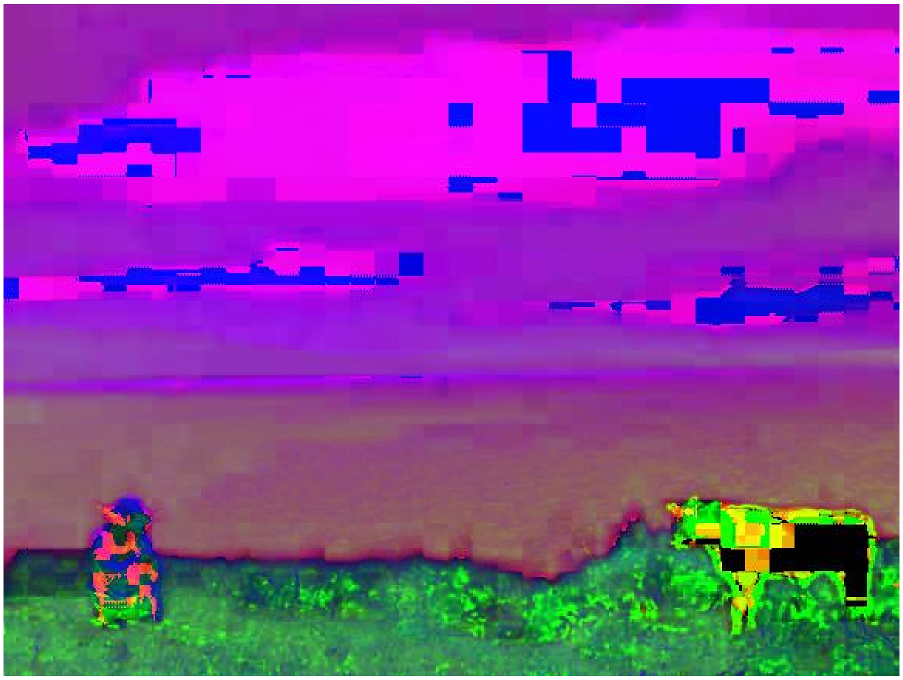
Custom Filtering Technique (inverted for readability)



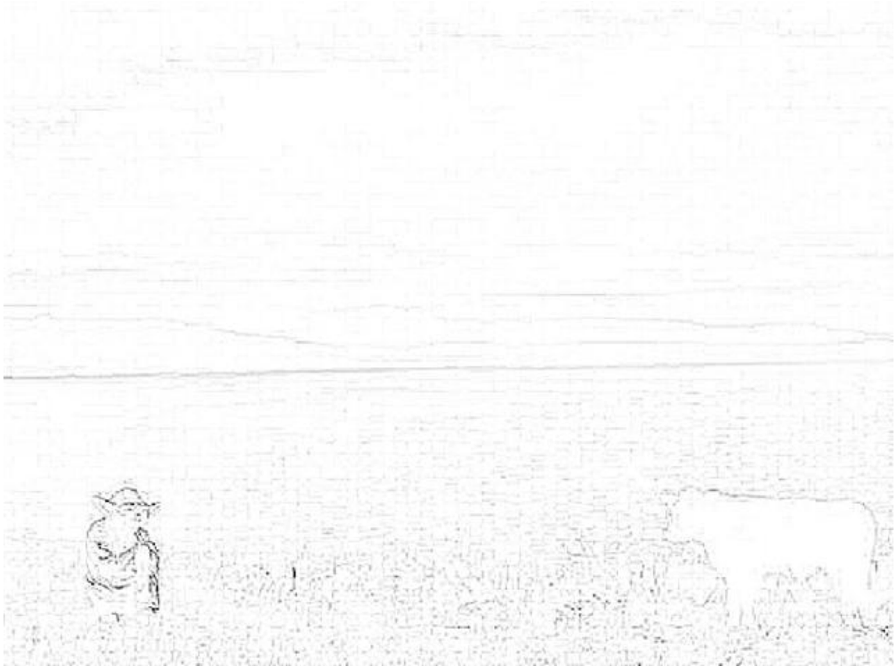
JPEG Block Technique w/ threshold 50

Results of Image Forgery B.4

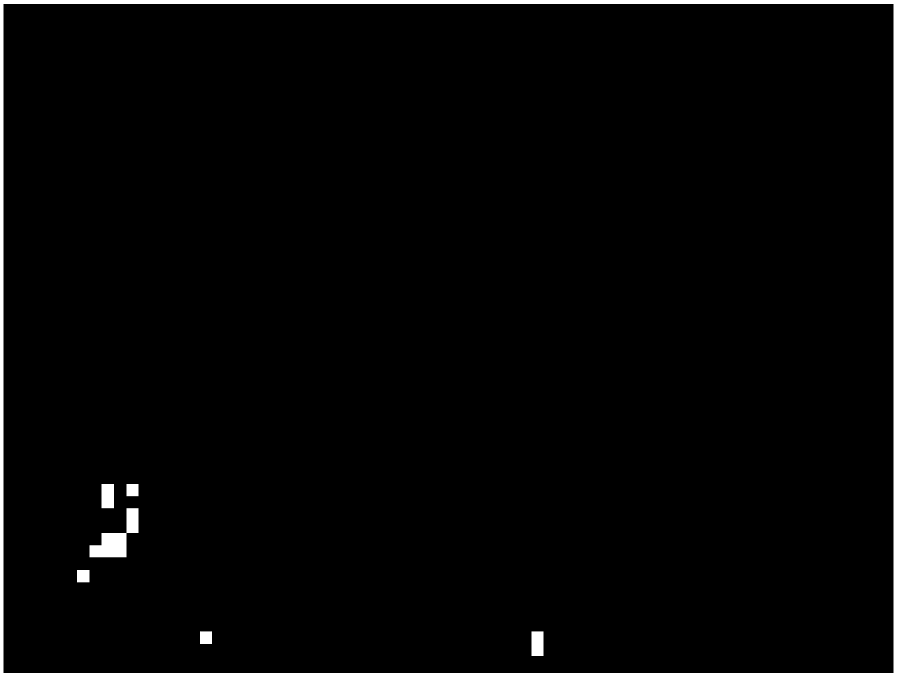
Luminance Level Technique w/ threshold 0.15



HSV Level Technique

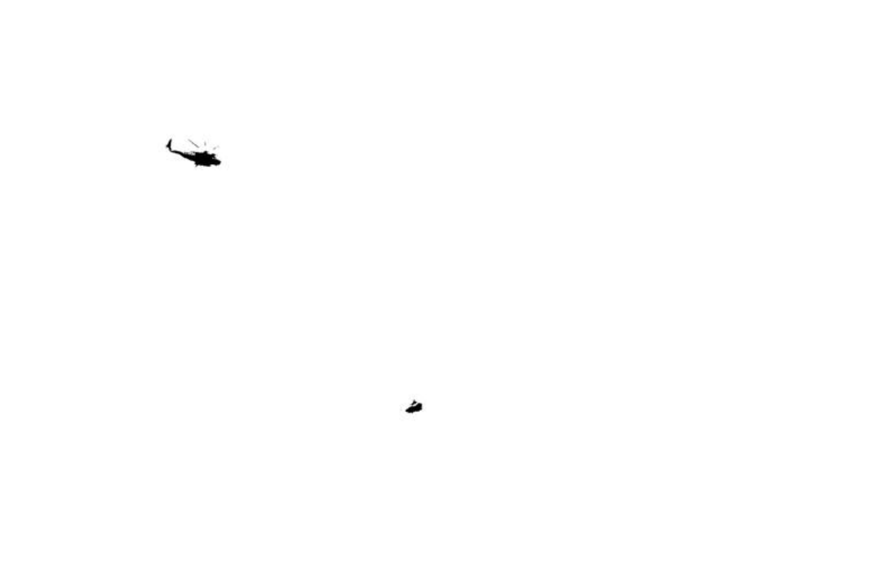


Custom Filtering Technique (inverted for readability)

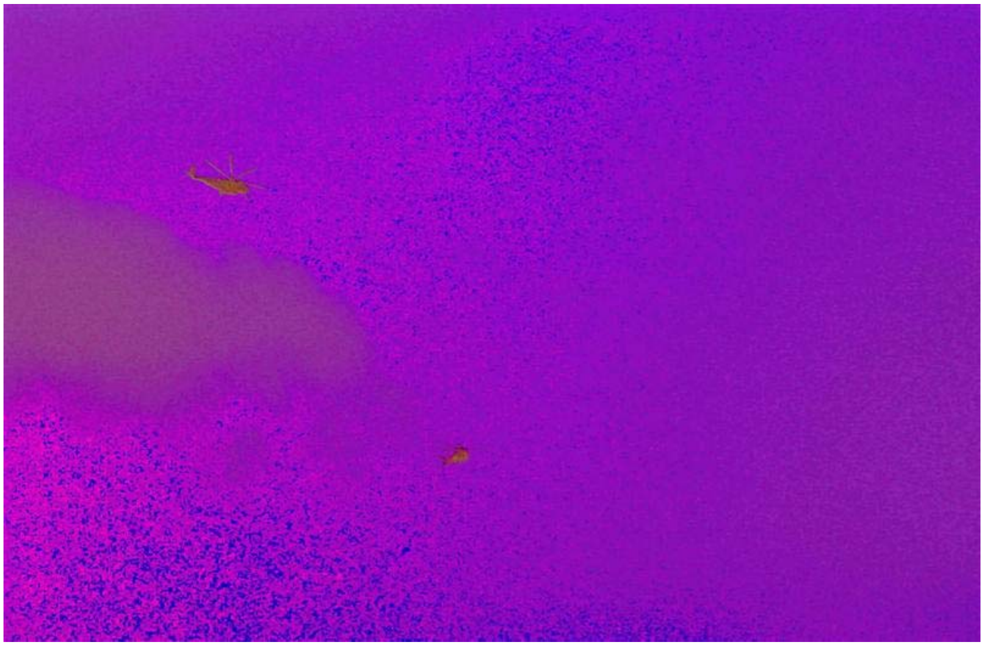


JPEG Block Technique w/ threshold 70

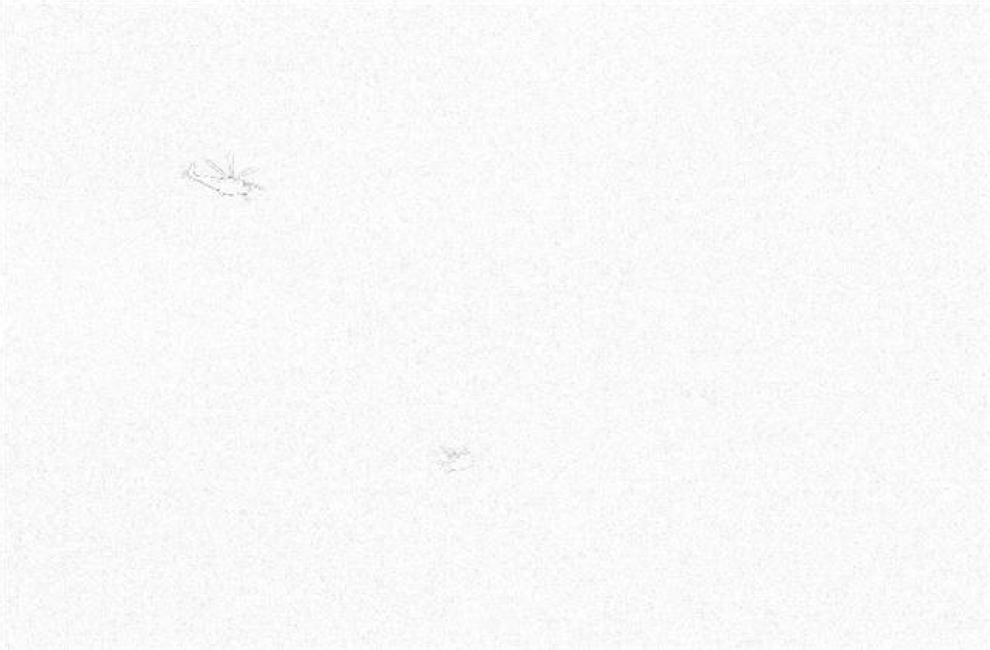
Results of Image Forgery B.5



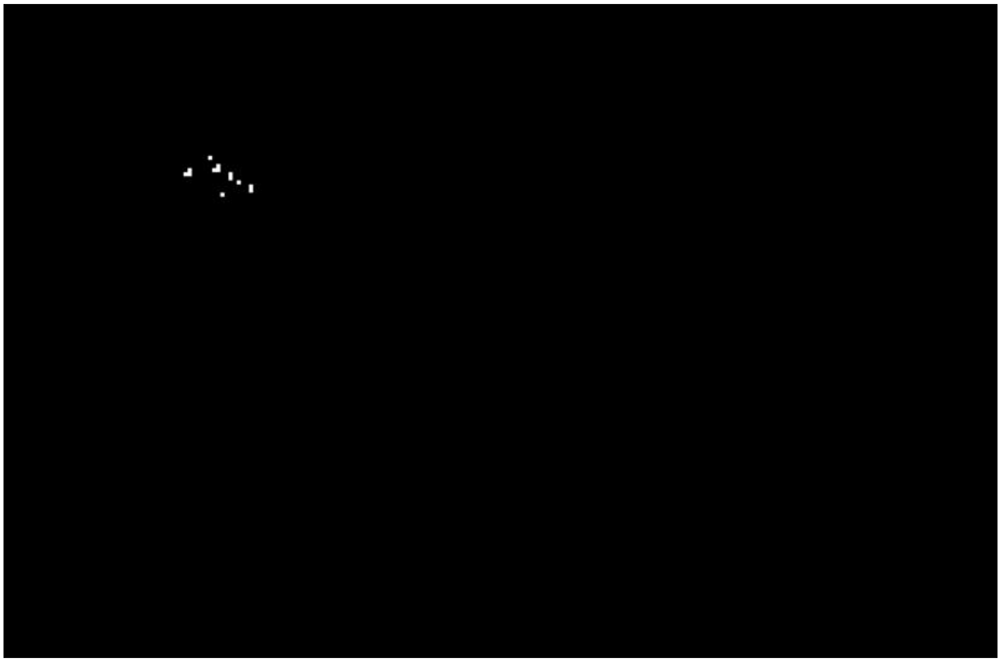
Luminance Level Technique w/ threshold 0.25



HSV Level Technique

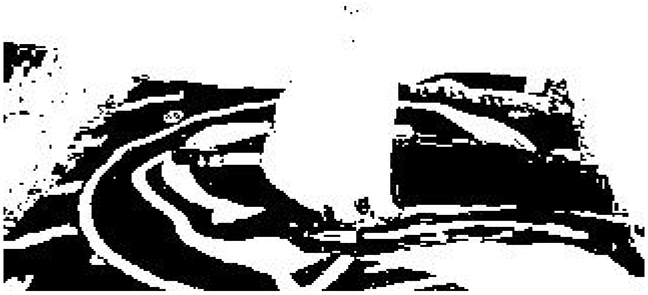


Custom Filtering Technique (inverted for readability)

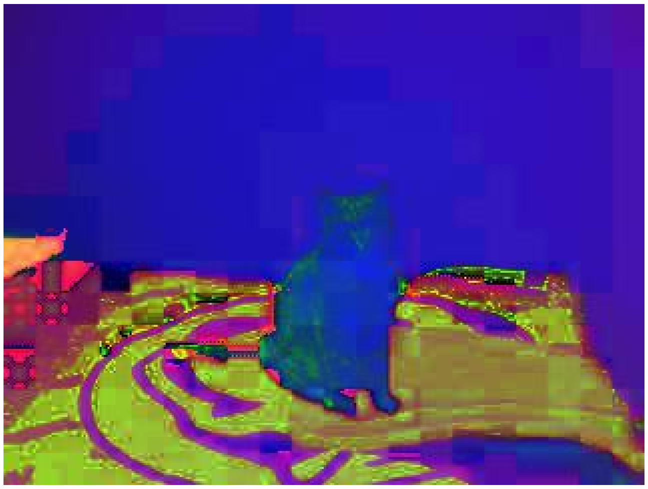


JPEG Block Technique w/ threshold 30

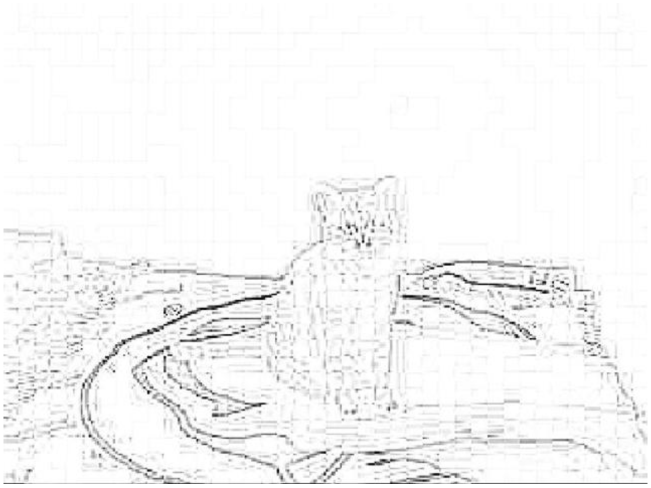
Results of Image Forgery B.6



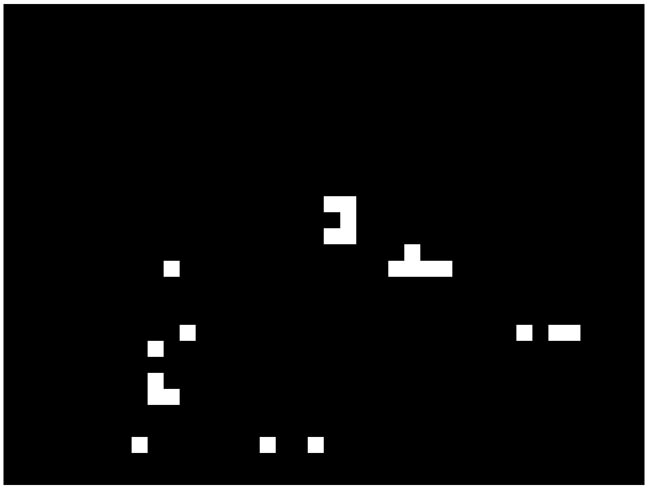
Luminance Level Technique w/ threshold 0.2



HSV Level Technique

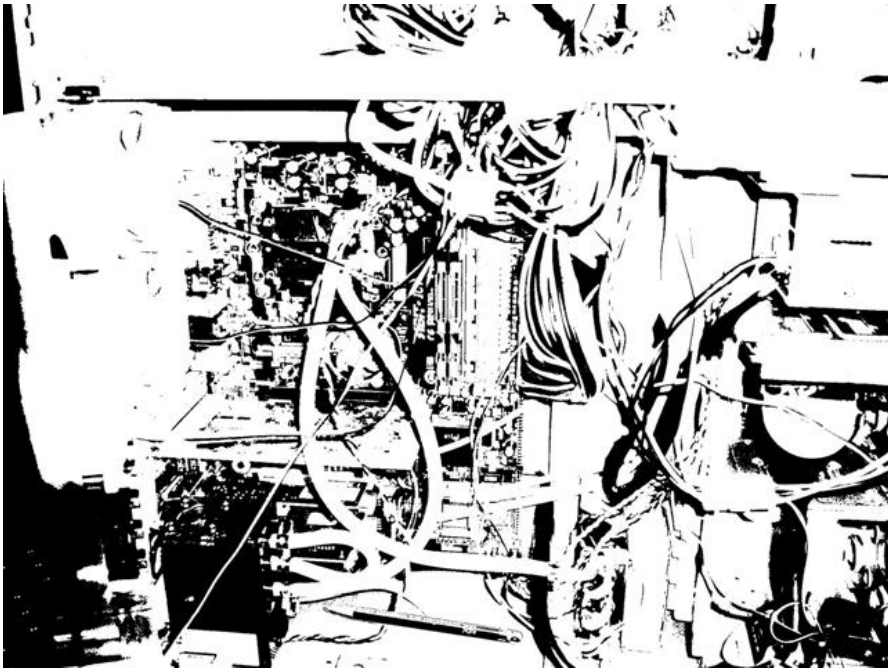


Custom Filtering Technique (inverted for readability)

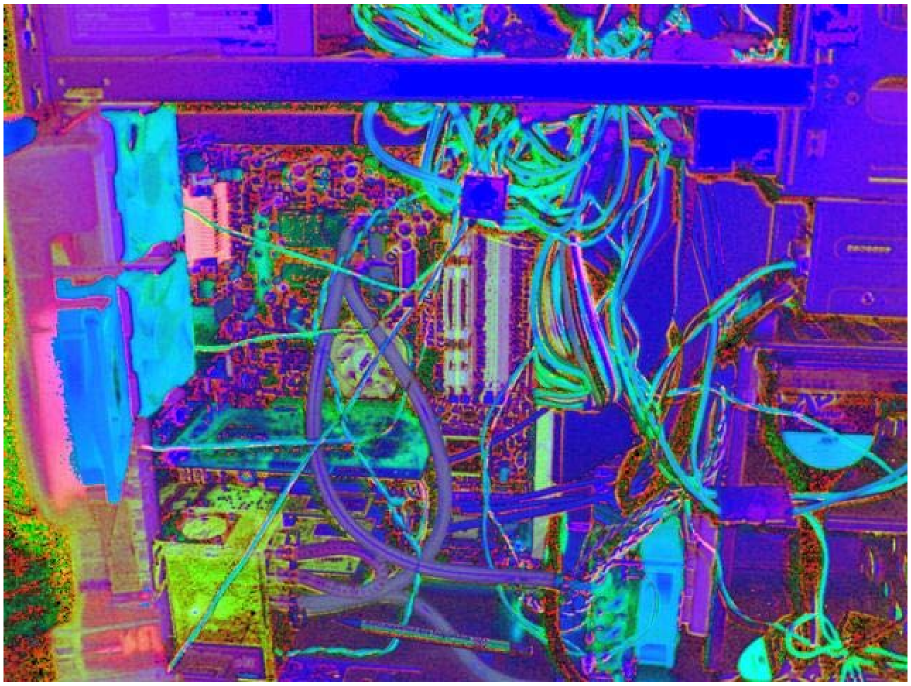


JPEG Block Technique w/ threshold 90

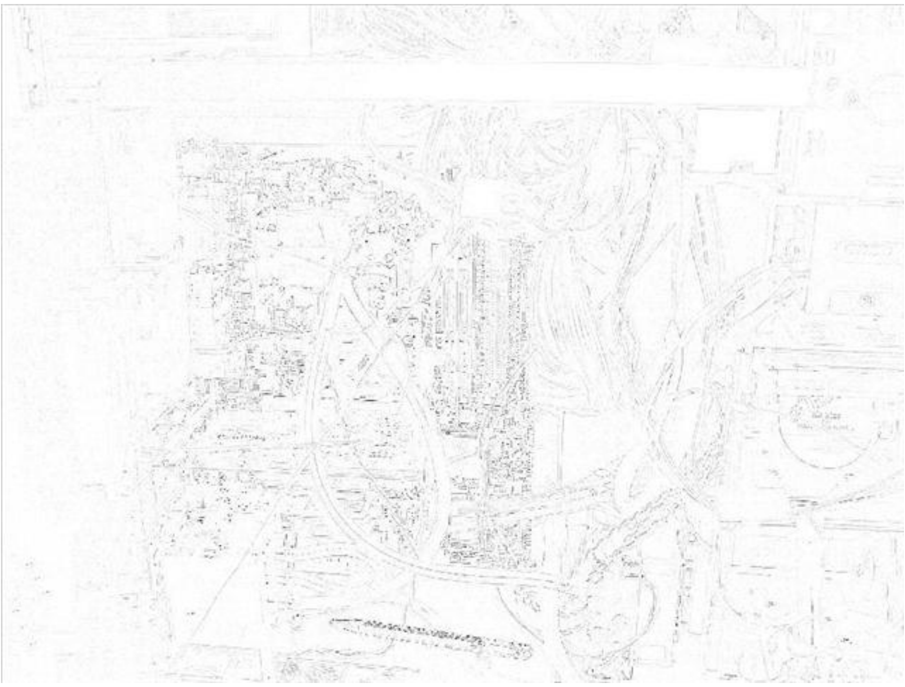
Results of Image Forgery B.7



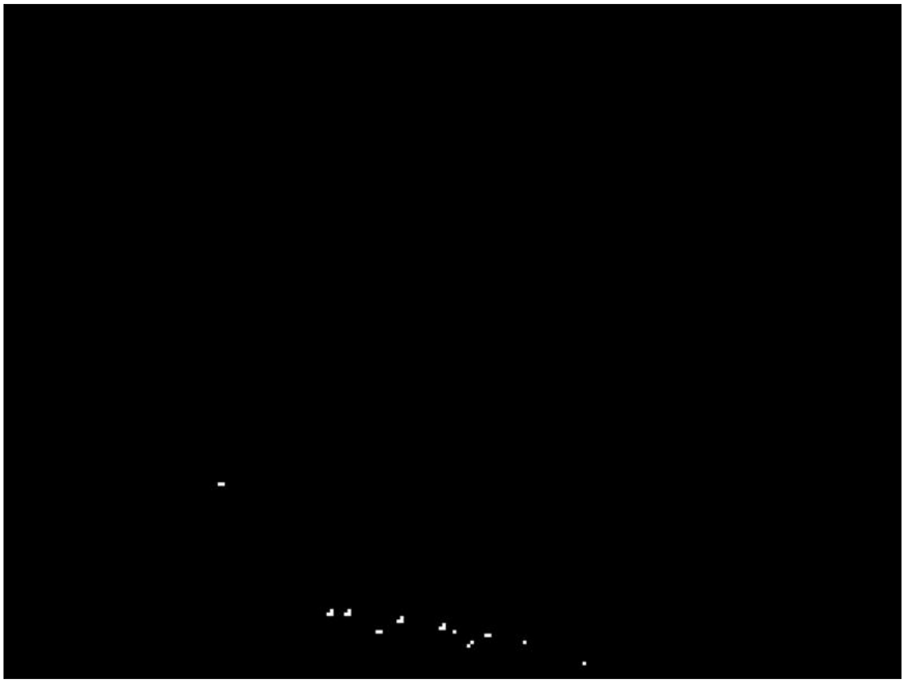
Luminance Level Technique w/ threshold 0.3



HSV Level Technique

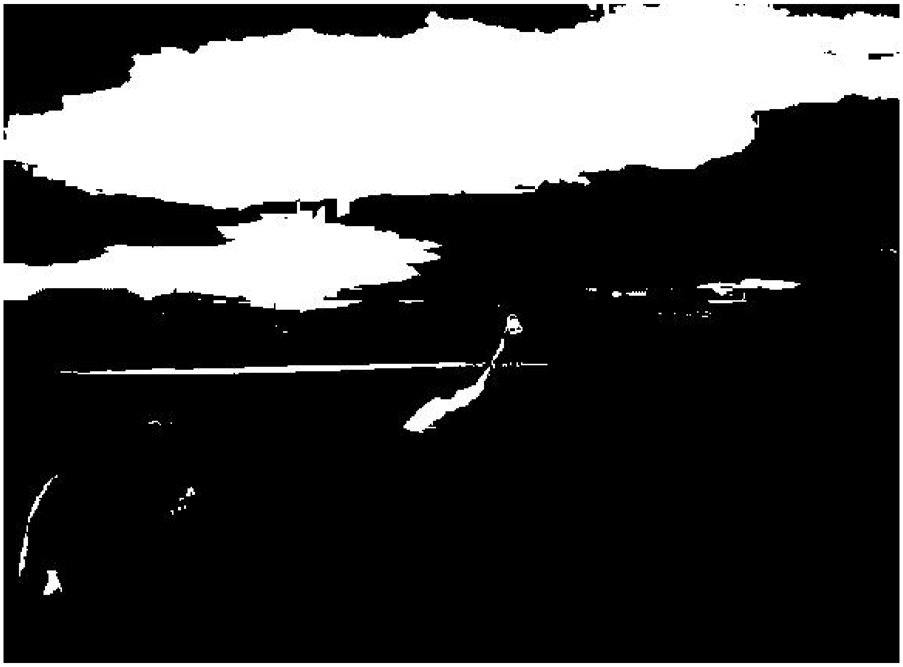


Custom Filtering Technique (inverted for readability)

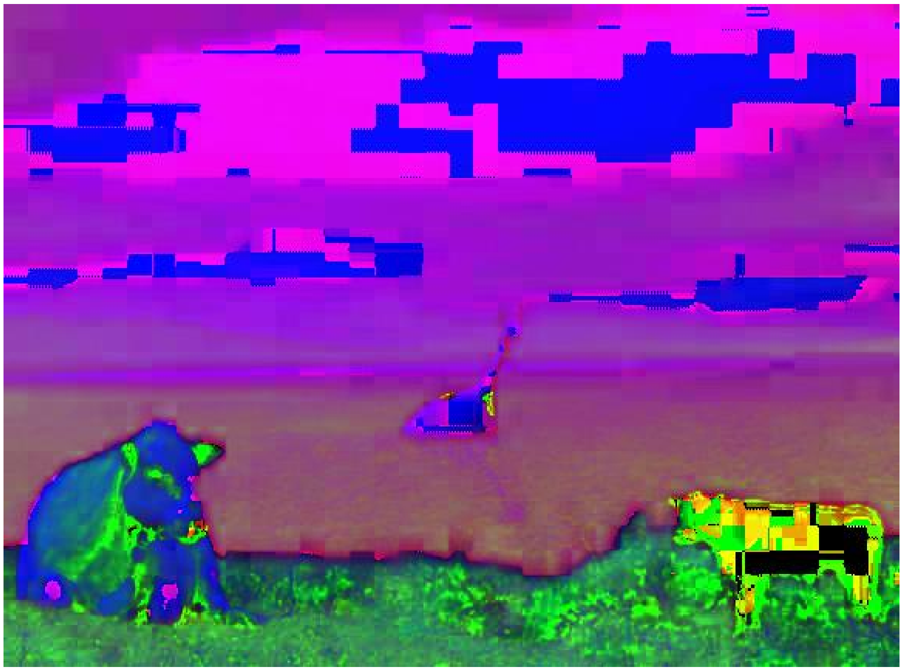


JPEG Block Technique w/ threshold 75

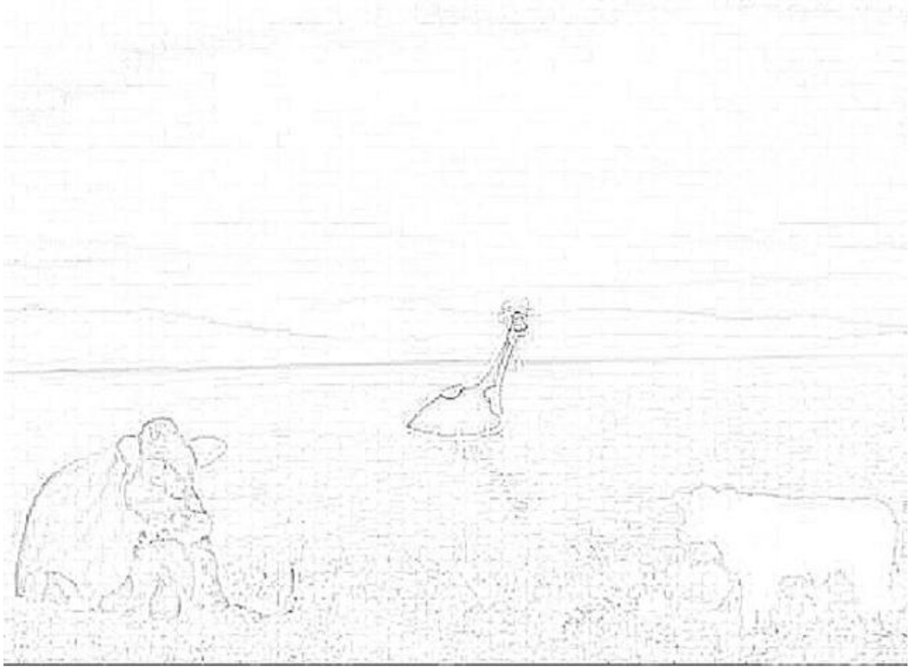
Results of Image Forgery B.WMno



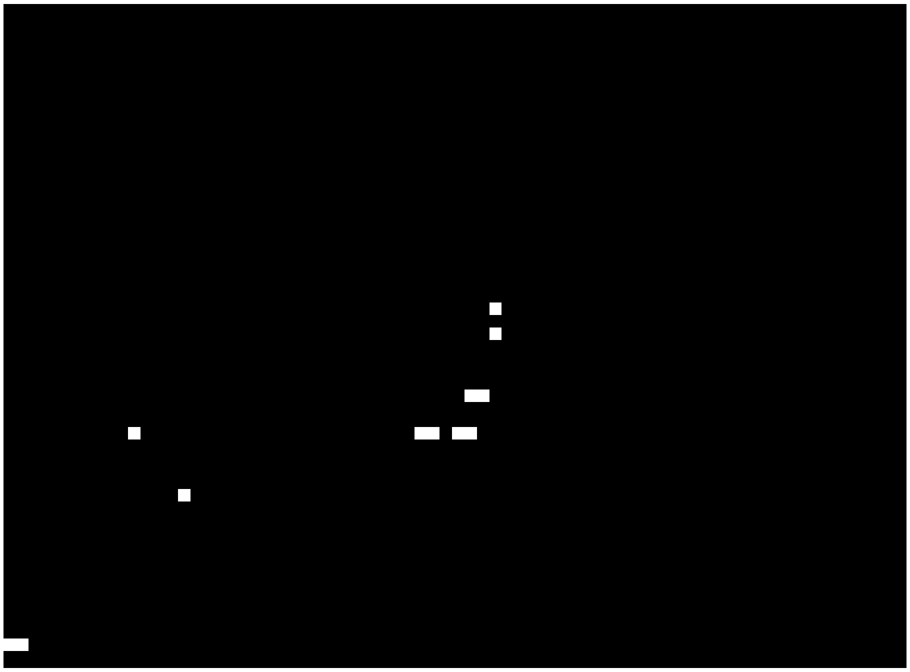
Luminance Level Technique w/ threshold 0.85



HSV Level Technique

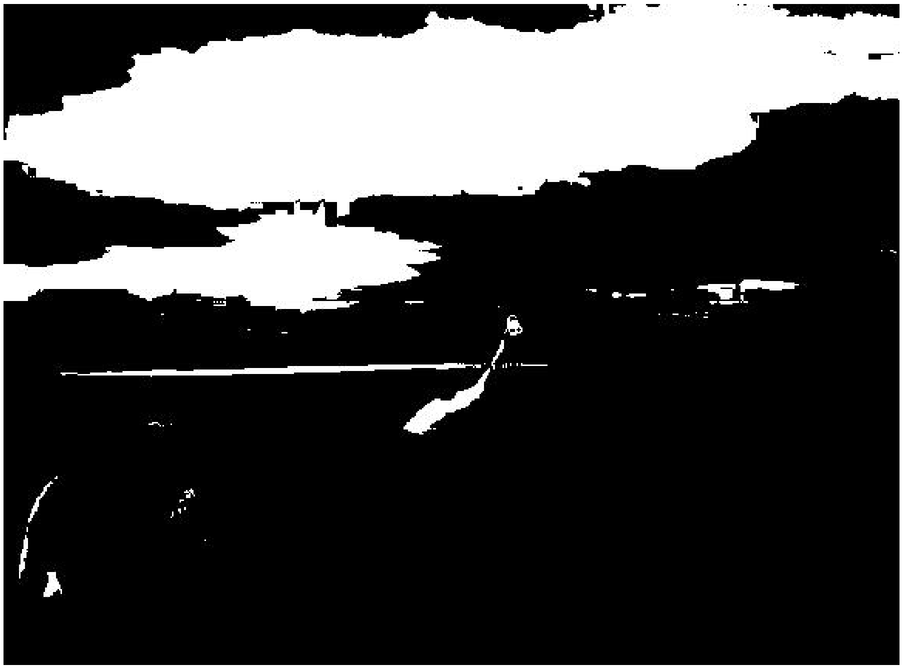


Custom Filtering Technique (inverted for readability)

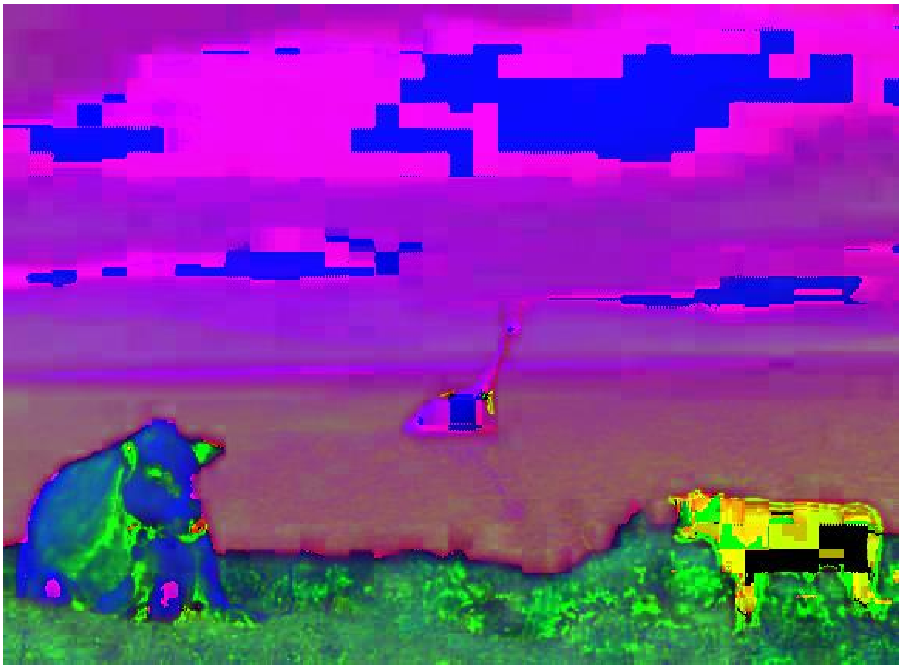


JPEG Block Technique w/ threshold 50

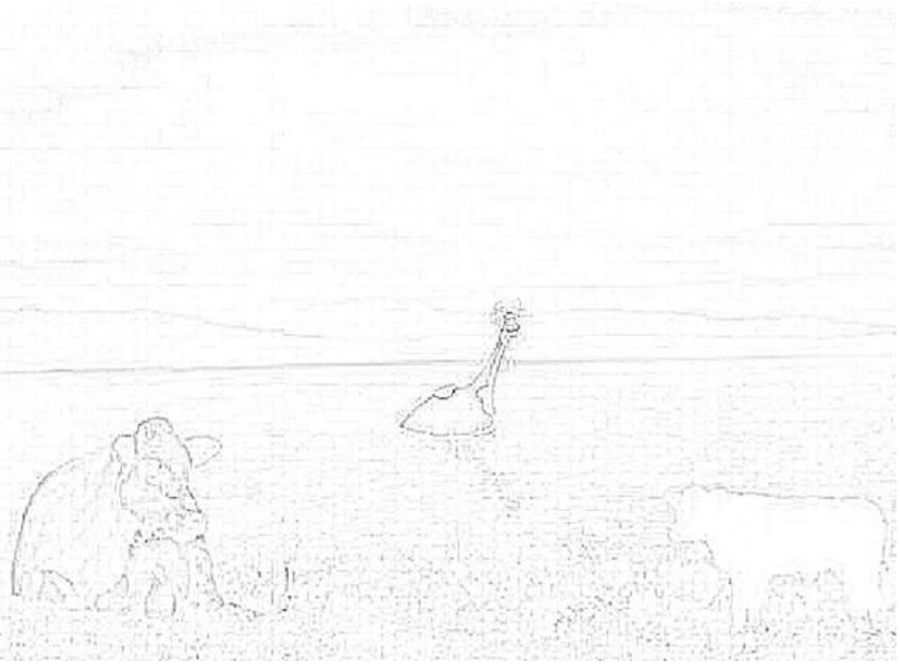
Results of Image Forgery B.WMyes



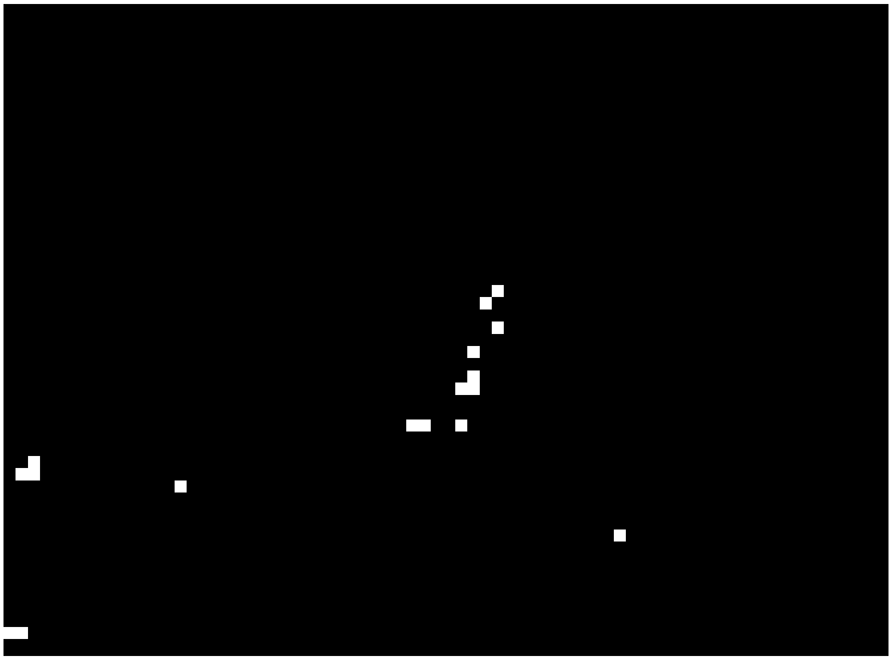
Luminance Level Technique w/ threshold 0.85



HSV Level Technique



Custom Filtering Technique (inverted for readability)

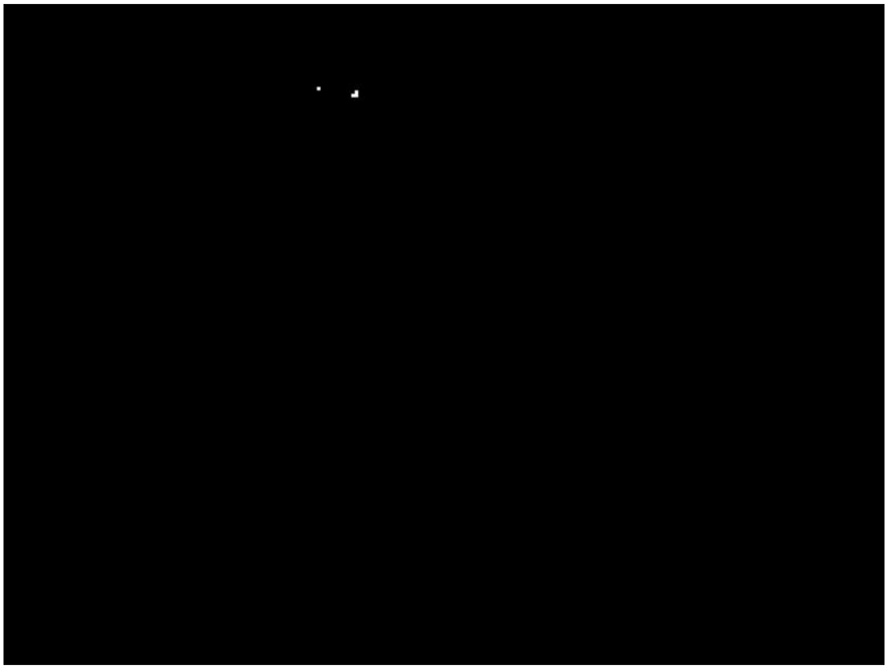


JPEG Block Technique w/ threshold 50

Results of Image Forgery B.QF.100

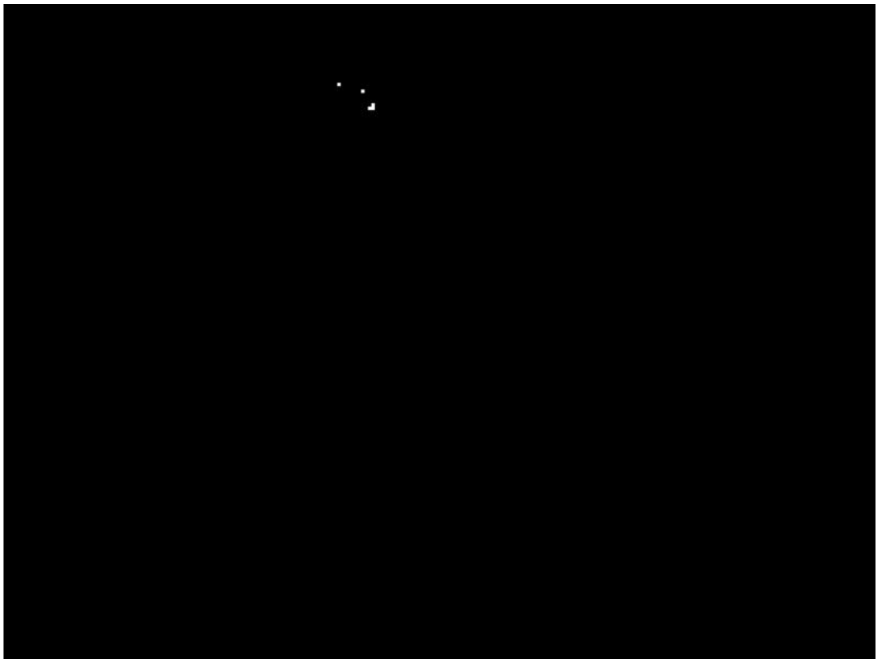
JPEG Block Technique w/ threshold 50

Results of Image Forgery B.QF.90



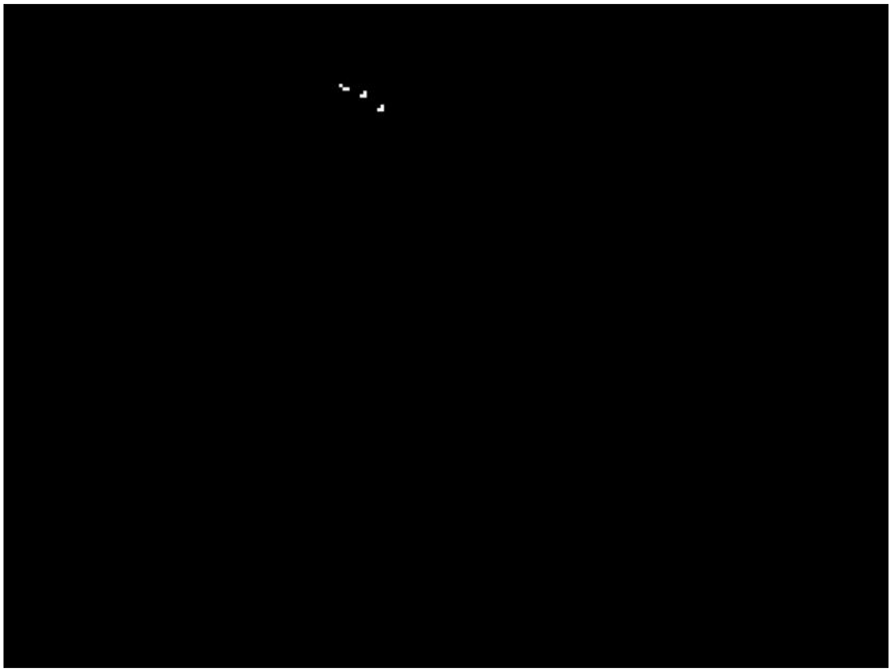
JPEG Block Technique w/ threshold 50

Results of Image Forgery B.QF.75



JPEG Block Technique w/ threshold 50

Results of Image Forgery B.QF.0



JPEG Block Technique w/ threshold 50

Bibliography

1. "JPEG2000," *TheFreeDicitionarydotcom*. August 18, 2004. http://encyclopedia.thefreedictionary.com/JPEG%202000.

2. "Myths & Facts about JPEG," *Graphics Software at About.com*. August 18, 2004. http://graphicssoft.about.com/library/weekly/aa0104jpegmyths.htm.

3. "What is Wavelet Analysis?" *Wavelet Toolbox Documentation*. MATLAB version 6.5. The MathWorks, Inc. 2002.

4. Associated Press, "Britain Says Soldier Held in Photo Probe." *Newsday* May 18, 2004. http://www.newsday.com/news/nationworld/world/wire/sns-ap-britain-prisoner-abuse,0,4827144.story?coll=sns-ap-world-headlines.

5. Baxes, G. A., *Digital Image Processing : Principles and Applications.* New York: John Wiley & Sons, Inc, 1994.

6. Brinkmann, R., *The Art and Science of Digital Compositing*. San Diego: Academic Press, 1999.

7. Chandramouli, R., N. Memon, and M. Rabbani, *Encyclopedia of Imaging Science and Technology: Digital Watermarking*. J. Hornak, Editor. John Wiley, October 2001.

8. Fan, Z. and R. L. de Queiroz, "Identification of Bitmap Compression History: JPEG Detection and Quantizer Estimation," *IEEE Transactions on Image Processing*, Vol. 12, No. 2: 230-235, February 2003.

9. Farid, H. and A. Popescu, "Exposing Digital Forgeries by Detecting Traces of Re-sampling." *Proceedings of the IEEE Transactions on Signal Processing.* (In Press). 2004.

10. Fridrich, J., R. Du, and M. Goljan, "Steganalysis Based on JPEG Compatibility," Special session on Theortical and Practical Issues in Digital Watermarking and Data Hiding, *Multimedia Systems and Applications IV*. Pp. 275-280. Denver, CO, August 2001.

11. Fridrich, J. and J. Lukas, "Estimation of Primary Quantization Matrix in Double Compressed JPEG Images." *Proceedings of DFRWS 2003*. Cleveland, OH, August 2003.

12. Fridrich, J., J. Lukas, and D. Soukal, "Detection of Copy-Move Forgery in Digital Images." *Proceedings of DFRWS 2003*. Cleveland, OH, August 2003.

13. Guggenheim, K. “New Prison Abuse Photos Outrage Lawmakers.” *Phillyburbs* May 13, 2004. http://www.phillyburbs.com/pb-dyn/news/27-05132004-299158.html.

14. Johnson, R. C. “JPEG2000 Wavelet Compression Spec Approved,” August 18, 2004. http://www.us.design-reuse.com/news/news1917.html.

15. Katzenbeisser, S. and F. Petitcolas, A.P. *Information Hiding – techniques for steganography and digital watermarking*. Boston: Artech House, 2000.

16. Klasen, L. *Image Sequence Analysis of Complex Objects*. PhD dissertation. Linkoping University, Sweden, 2002.

17. Lukas, J. “Digital Image Authentication Using Image Filtering Techniques.” *Proceedings of 15th Conference of Scientific computing "Algoritmy 2000"*, Vysoke Tatry-Podbanske, Slovakia, September 2000.

18. Luong C. M. “Introduction to Computer Vision and Image Processing,” Department of Pattern Recognition and Knowledge Engineering, Institute of Information Technology, Hanoi, Vietnam, May 4, 2004. http://www.netnam.vn/unescocourse/computervision/computer.htm.

19. Minor, E. (Associated Press) “Hogzilla! Huge pig has small town talking, but is it a hoax?” *San Jose Mercury News,* July 29, 2004. sec. 3A.

20. Sachs, J. *Digital Image Basics*. Digital Light & Color. 1999.

21. Saha, S. “Image Compression – from DCT to Wavelets: A Review,” May 28, 2004 http://www.acm.org/crossroads/xrds6-3/sahaimgcoding.html.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* 01-12-2004	**2. REPORT TYPE** Master's Thesis	**3. DATES COVERED** *(From - To)* March 2004 - December 2004

4. TITLE AND SUBTITLE
FORENSIC ANALYSIS OF DIGITAL IMAGE TAMPERING

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Sturak, Jonathan R.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Building 640
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GIA/ENG/04-01

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Mr. Scott F. Adams
Air Force Research Laboratory
Information Directorate/IFEC (AFMC)
32 Brooks Road
Rome, NY 13441-4114 Phone: (315) 330-4104

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The use of digital photography has increased over the past few years, a trend which opens the door for new and creative ways to forge images. The manipulation of images through forgery influences the perception an observer has of the depicted scene, potentially resulting in ill consequences if created with malicious intentions. This poses a need to verify the authenticity of images originating from unknown sources in absence of any prior digital watermarking or authentication technique. This research explores the holes left by existing research; specifically, the ability to detect image forgeries created using multiple image sources and specialized methods tailored to the popular JPEG image format. In an effort to meet these goals, this thesis presents four methods to detect image tampering based on fundamental image attributes common to any forgery. These include discrepancies in 1) lighting and 2) brightness levels, 3) underlying edge inconsistencies, and 4) anomalies in JPEG compression blocks. Overall, these methods proved encouraging in detecting image forgeries with an observed accuracy of 60% in a completely blind experiment containing a mixture of 15 authentic and forged images.

15. SUBJECT TERMS
Image Forgery, Authentication, Image Tampering, JPEG

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Gilbert L. Peterson, AFIT/ENG
U	U	U	UU	138	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-6565, ext 4281 (gilbert.peterson@afit.edu)