DIGITAL IMAGE RECOMPRESSION ANALYSIS OF INSTAGRAM

by

ZACHARY DOUGLAS

B.S., University of Colorado Denver, 2015

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirement for the degree of

Master of Science

Recording Arts

2018

This thesis for the Master of Science degree by

Zachary Douglas

Has been approved for the

Recording Arts Program

by

Catalin Grigoras, Chair

Jeff Smith

Lorne Bregitzer

Date: May 12, 2018

Douglas, Zachary (M.S. Recording Arts)

Digital Image Recompression Analysis of Instagram

Thesis directed by Professor Catalin Grigoras

ABSTRACT

Digital images have been used for years in forensic investigations of crimes. With images taken by digital cameras, cell phones, digital video recorders and more, data can be extracted from the metadata of the image in question. The metadata can then be used to aid in connecting said photograph to the device from which it was captured. In addition, further information is sometimes available that can include GPS data, the date and time an image was captured, and digital signatures of any pieces of software with which they may have interacted. While this forensic data has been key to many investigations, recently a trend has been seen of users uploading their photos to social media websites for others to view. While this is a great convenience for users of social media, for forensic examiners it creates some setbacks in the data that is able to be collected from images obtained off of social media sites. In other recent studies of social media sites such as Facebook, it has been found that when images are uploaded to the social media platforms, the images are recompressed and the metadata is wiped (9). This in turn leaves very little original data in tact, which is traditionally relied upon in forensic examinations in order to establish the provenance of an image. Due to this, this thesis researched how images are recompressed when uploaded to the social media platform Instagram. This will allow investigators to determine how to best retrieve an image from Instagram, what to expect from the metadata included in recovered files, how the structure will appear, and some general guidelines for how the software behaves when interacting with original photos of varying resolutions.

The form and content of this abstract are approved. I recommend its publication.

Approved: Catalin Grigoras

DEDICATION

I dedicate this work to my beloved wife Nichole for constantly pushing me to be my best, to work through the hard times, and to persevere in the face of daunting hurdles. I also dedicate this work to my daughter Isla, who is my motivation to provide the best life possible for my family and allow her every opportunity imaginable. I am also grateful for my parents and sister, Bill, Deborah, and Erica, who have always supported my decisions and motivated me to constantly work to push past my boundaries, achieve the goals that I set for myself, and constantly work towards being the best person that I can be.

ACKNOWLEDGEMENTS

I thank my professors Jeff Smith & Catalin Grigoras for their overwhelming support and guidance throughout this process and my education. Prior to entering this program at the University of Colorado Denver, I never imagined that I could one day obtain some of the skills and knowledge that these two have provided me. I could not ask for more dedicated or experienced professionals in the field of digital and multimedia forensics to have learned from, and I am forever grateful for their guidance and the inspiration that they have provided me.

TABLE OF CONTENTS

CHAPTER

LIST OF FIGURES

FIGURE

LIST OF TABLES

TABLE

LIST OF ABBREVIATIONS

API: Application Programming Interface

ADC: Analog to Digital Converter

CCD: Charge Coupled Device

CCIR: Consultative Committee on International Radio

CCITT: International Telegraph and Telephone Consultative Committee

CFA: Color Filter Array

CMOS: Complementary Metal Oxide Conductor

CMYK: Cyan, Magenta, Yellow, Key (Black)

DCT: Discrete Cosine Transform

DIP: Digital Imaging Processor

DSLR: Digital Single Lens Reflex

ISO: International Standard Organization

ITU-R: International Telecommunications Union Recommendation

JFIF: JPEG File Interchange Format

JPEG: Joint Photographic Experts Group

kB: Kilobyte

LCD: Liquid Crystal Display

MSE: Mean Square Error

nM: Nanometer

PEG: Photographic Expert Group

PSNR: Quality Measurement of the Reconstructed Image

QF: Quality Factor

RGB: Red Green Blue

RLC: Run Length Coding

RST Markers: Restart Markers

SOI: Start of Image

SNR: Signal to Noise Ratio

TIFF: Tagged Image File Format

YCbCr: Luminance, Chrominance Blue, Chrominance Red

CHAPTER I

INTRODUCTION

When it comes to forensic sciences, there is a lot of confusion about what is actually entailed in the work. Much of this confusion can be attributed to a lack of education and understanding of the actual practices used by forensic scientists and maybe due to the glamorization that Hollywood has used to portray forensics in the multitude of "CSI" style shows. From these shows, the study of forensics is commonly thought to relate to anything pertaining to crime scenes, laboratory work, or handling evidence (1). While it is true that forensic scientists work with these types of cases, at its root, forensic science, as defined by P. Jones of Purdue University, is applying a specific discipline of science to a court or legal proceeding (1). Due to this, it must be noted that while forensic scientists work on a wide range of cases, at the end of the day they are working to provide clarity to a sequence of events that took place so that evidence can be presented in courts and legal proceedings.

History

Though the term digital imaging in forensics very directly states that this discipline deals in the digital realm, the best practices and methods used have been expanding and evolving since the advent of photography. This dates back to 1839 when Louis-Jacques-Mandé Daguerre first presented his revolutionary invention, photography (2). Through this discovery, which was based on the concepts discovered through the use of the camera obscura, Daguerre launched the world in a new direction for how we would come to capture specific moments in time into a tangible format (2). From this, the profession of forensic imaging was born.

From the advent of photography, investigators quickly saw its potential to be used in the documentation of evidence. One of the earliest recorded instances of this was in 1888 when photographs were used to document crime scenes and the victims that were attributed to Jack the Ripper (1). Today forensic specialists handle the job of photographing and documenting crime scenes and evidence; however, in the early stages of photography there was no such position or specialty. Instead, the police simply contacted the local newspaper photographers who came and took a picture or two of the scene in question (1). Due to this, scenes from investigations during that time frame are unable to be fully recreated or examined due to the lack of documentation. These early cases started the use of photography as a forensic practice through which it evolved over the years, developing into the highly scientific field that it is today.

Capturing Images

Though a lot has changed since the advent of photography, the general principles have remained the same in regards to how cameras function. The primary difference that is seen in modern cameras versus their predecessors, is the digitization of photography. Before this happened, cameras primarily relied on film that had anywhere from one to twelve emulsion layers of light sensitive silver halides (1). This type of film functioned by creating a still shot of the scene that was being viewed through the lens when the picture was taken. The image is captured by the shutter opening and closing for a precise period of time, which allows light to enter the camera and make contact with the film. The amount of light that enters the camera is controlled by the shutter speed and the aperture of the diaphragm. The light that passes through the lens then reacts with the silver halide on the film and creates a still image of the scene that was being viewed, as demonstrated in Figure 1.

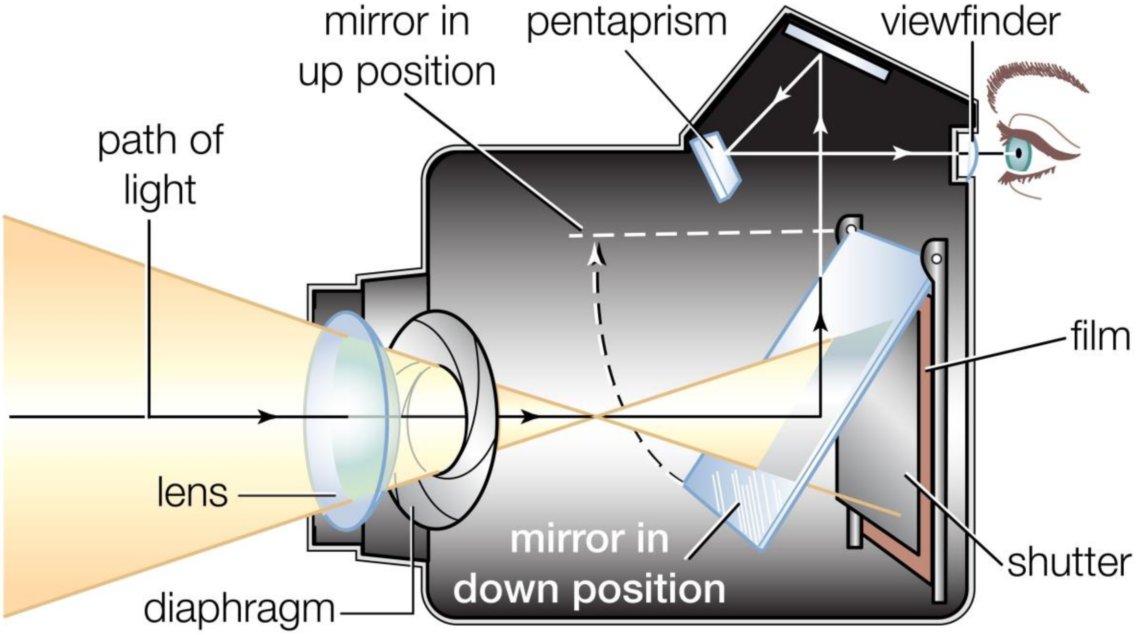


Figure 1: Diagram of a Single-Lens Reflex Camera (13)

The amount of light that is allowed through the aperture and the length of time the shutter is set to remain open will dictate how bright or dark the resulting image will be, and how much movement from the scene viewed will be captured on the film.

Once a picture is taken with a film camera, the film must then be developed. With standard film cameras this entails creating a negative of the image on the film, which can then be used to create multiple prints. This process causes a delay in the availability of the forensic documentation of a particular scene or piece(s) of evidence. To expedite this process, forensic photographers have transitioned to the use of digital cameras, which removes the delay caused by photo development. By doing this, forensic photographers have dramatically changed the time frame in which investigators can view the documentation of the scene of interest. With new Digital Single Lens Reflex (DSLR) cameras, many photographers are now able to have images uploaded in real time as they are taken via Wi-Fi and cellular data connections. This digital connectivity has dramatically streamlined the process of getting the captured images into the hands of investigators immediately, versus the significantly outdated method in which examiners and investigators waited hours, if not days, for the photographs taken on film to be developed, printed, and finally delivered to them.

Digital Cameras

With the advent of digital cameras, a lot has changed with how images are taken and utilized. Though many digital cameras are still very similar in their physical design as their film-based predecessors were, the way in which they function is significantly different. While film cameras were able to simply rely on the light available in any given environment and the reactivity of the silver halide film that was used, today's digital cameras must use electronics to mimic the same process in order to capture the image of interest. To do this, camera manufacturers had to come up with a way for an image to be taken and converted into a digital medium.

To do this, the best way found to turn a scene that is being photographed into a digital image was to break it up into a grid matrix. Each box in the grid matrix, otherwise known as a pixel, is assigned a numerical value that represents a specific color (4). These numerical values tell computers and the camera what colors to display at a specific location in an image. Each box is assigned a number between 0 and 255. In a grayscale image, a value of 0 represents absolute black, whereas a value of 255 represents absolute white. Likewise, in a red, green, blue (RGB) image, 0 represents absolute red, green, or blue, depending on the channel, whereas 255 represents absolute white. By doing this, the electronic processors can parse the data of a digital image into a finite set of data points, making images much more manageable for computers to process rather than if the number of data points was infinite. While this process includes significantly more steps than this to go from taking a picture till you can actually view the resulting image, a basic example of how a photographed image is captured and translated into a digital medium in this manner can be seen in Figures 2 and 3.

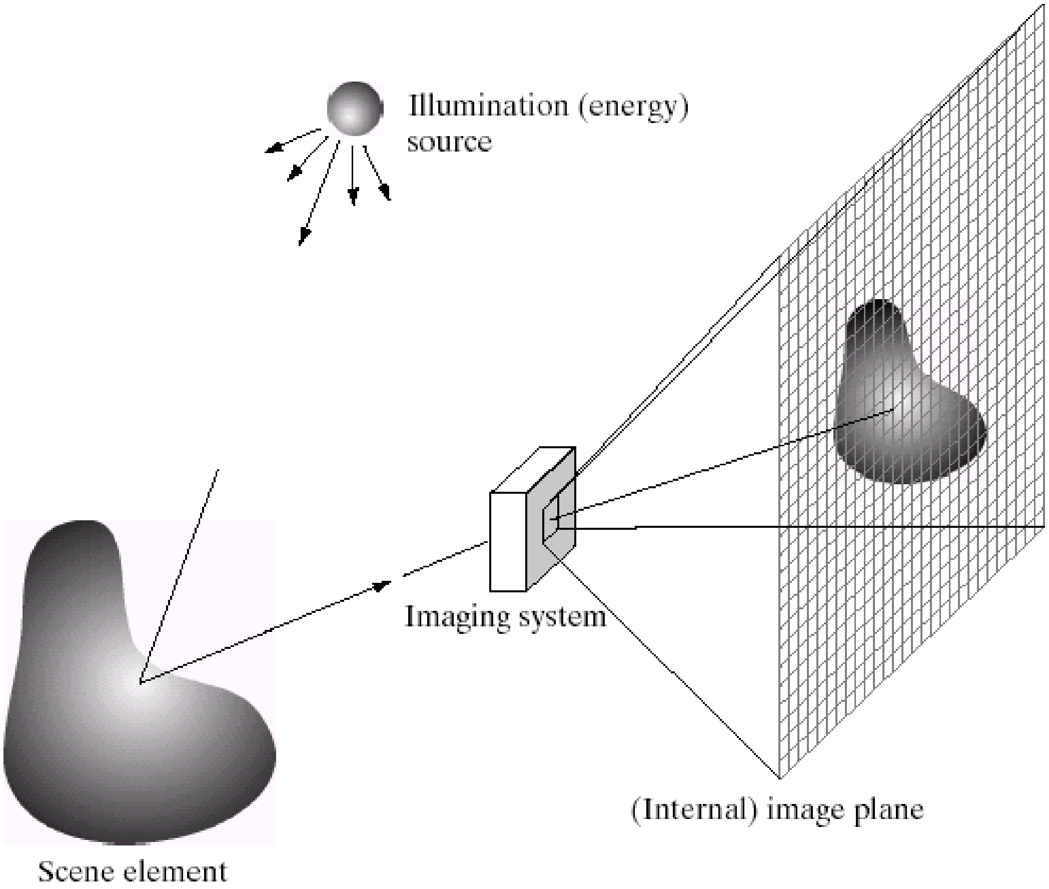
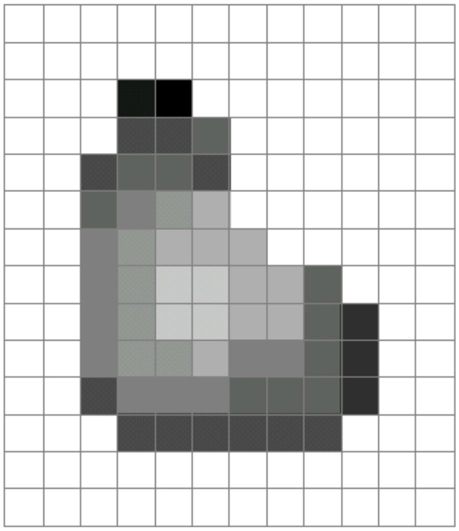


Figure 2: Overview of How a Viewed Scene Becomes a Digital Representation (3)



=

255	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**
255	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**
255	**255**	**255**	**20**	**0**	**255**	**255**	**255**	**255**	**255**	**255**	**255**
255	**255**	**255**	**75**	**75**	**75**	**255**	**255**	**255**	**255**	**255**	**255**
255	**255**	**75**	**95**	**95**	**75**	**255**	**255**	**255**	**255**	**255**	**255**
255	**255**	**96**	**127**	**145**	**175**	**255**	**255**	**255**	**255**	**255**	**255**
255	**255**	**127**	**145**	**175**	**175**	**175**	**255**	**255**	**255**	**255**	**255**
255	**255**	**127**	**145**	**200**	**200**	**175**	**175**	**95**	**255**	**255**	**255**
255	**255**	**127**	**145**	**200**	**200**	**175**	**175**	**95**	**47**	**255**	**255**
255	**255**	**127**	**145**	**145**	**175**	**127**	**127**	**95**	**47**	**255**	**255**
255	**255**	**74**	**127**	**127**	**127**	**95**	**95**	**95**	**47**	**255**	**255**
255	**255**	**255**	**74**	**74**	**74**	**74**	**74**	**74**	**255**	**255**	**255**
255	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**
255	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**

Figure 3: How Colors are Dictated in a Black and White Image (3)

In order to get an image in its final digital format though, many more processes are needed to dictate how an image is captured, such as the following.

- What color profile will be used
- How the image will be converted into the digital realm
- How the image will be digitally processed
- How the resulting digital file will be stored

To do all of this, the processes must start in the analog realm. While it is called digital imaging, this process still follows the same path that was laid out by its analog film predecessor in order to initially capture an image. It is due to this that many digital cameras still have the same basic design as their analog predecessors. Many of the components used still serve the same purpose, even if they function differently. These include the lens, diaphragm, mirror (though the view finder is increasingly becoming replaced by liquid crystal display (LCD) screens), and the recording medium. Instead of film being used as the recording medium though, the light strikes a variety of filters and sensors to capture the image. An example of this and the likeness of a DSLRs design to a traditional film camera can be seen in Figure 4.

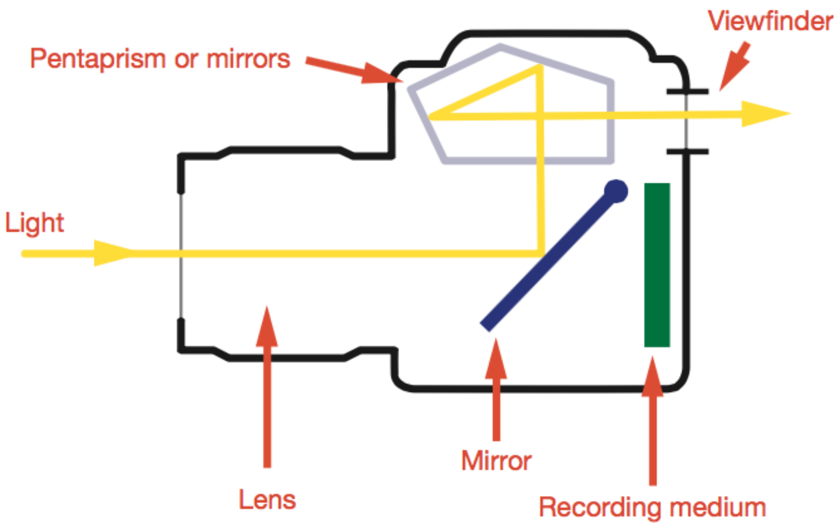


Figure 4: Diagram of a single-lens reflex camera (1)

While this traditional DSLR model followed many of the same design characteristics of traditional film-based cameras, the most commonly used cameras of today have been radically redesigned in order to be carried in the pockets of individuals everywhere. These new cameras are found on nearly every cellular phone that is sold today. While these cameras look nothing like their digital point and shoot, DSLR, or film-based predecessors, they too are still designed with the same basic principles as all other cameras to date. Great care has been taken to miniaturize all of the components and hardware in order to fit the camera into a device that is not only capable of fitting in someone's pocket, but that also has the ability to make and receive phone calls, access the internet, and so much more. An example of cameras that are used in smartphones can be seen in Figures 5 and 6.

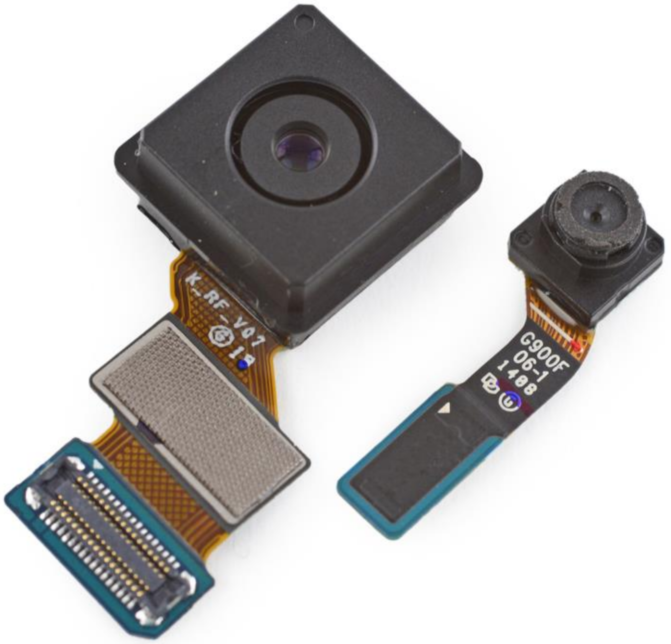


Figure 5: Samsung Galaxy S5 Camera Module (14)

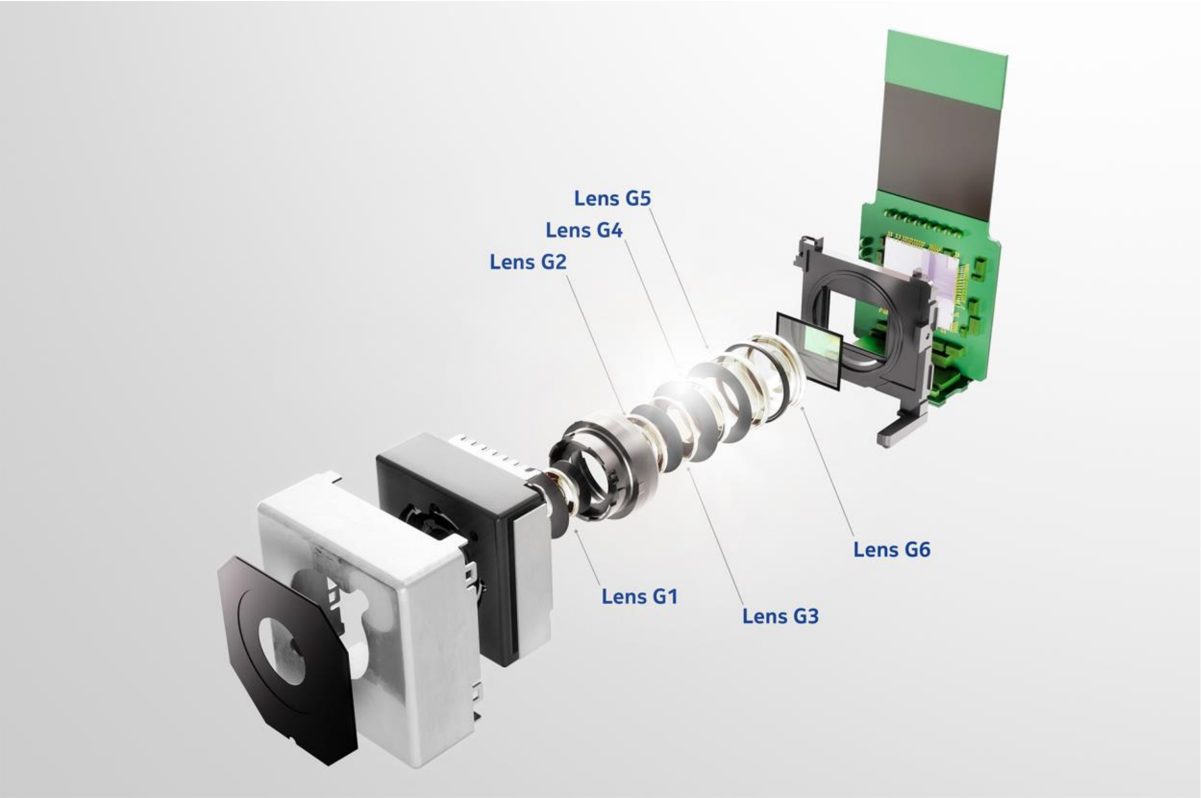


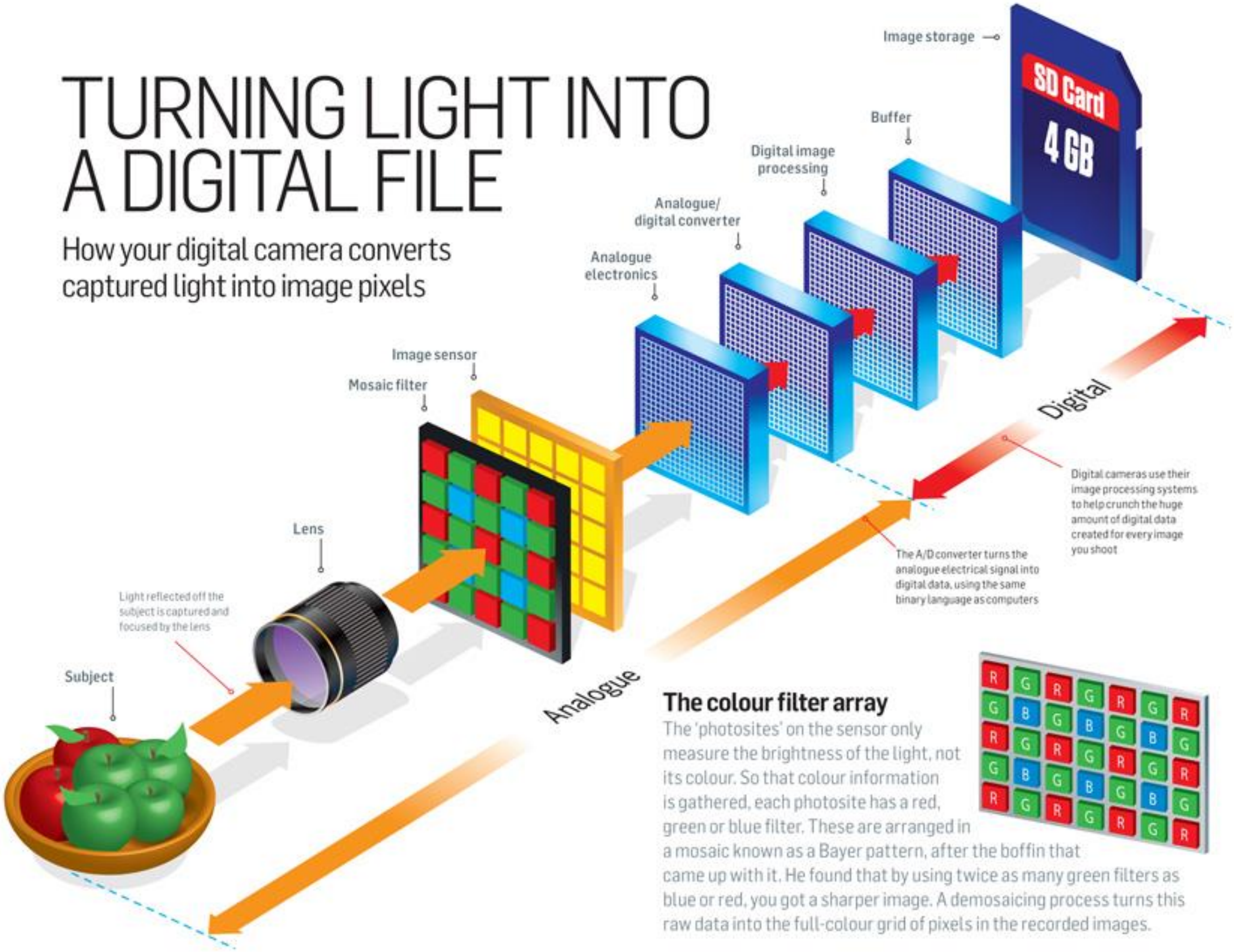
Figure 6: Breakout Construction of Nokia's PureView Camera (14)

Due to the need for miniaturization of all of the components of a camera, there are many limitations that have been addressed and compromises that had to be made in order to scale down the size of every component used. However, due to ever evolving technological advances, the cameras found in smartphones of today have overcome earlier technological limitations and are more widely used than any point and shoot or DSLR cameras. This, combined with the ability to instantly upload images to the Internet, along with favorite photo applications such as Instagram, Facebook, and Twitter, has created an explosion of digital content that is being shared every second of every day.

While digital cameras now come in all different shapes and sizes, such as DSLRs, point and shoot, and camera phones, the design and function of all of these cameras are relatively the same. The differences inherent to each camera lie within the components that are used. Though the process in which a digital image is created is generally the same, differences can come from the following:

- The Lenses
- Type of Color Filter Array (CFA)
- Type of Image Sensor
- Electrical Components
- The Analog to Digital Converter (ADC)
- The Digital Imaging Processor (DIP)
- The Storage Device/Capabilities Utilized

To visualize all of these components that are used in a digital camera, Figure 7 can be referenced which shows the basic order of all of the components of a digital camera.



TURNING LIGHT INTO A DIGITAL FILE

How your digital camera converts captured light into image pixels

Light reflected off the subject is captured and focused by the lens

The A/D converter turns the analogue electrical signal into digital data, using the same binary language as computers

Digital cameras use their image processing systems to help crunch the huge amount of digital data created for every image you shoot

The colour filter array

The 'photosites' on the sensor only measure the brightness of the light, not its colour. So that colour information is gathered, each photosite has a red, green or blue filter. These are arranged in a mosaic known as a Bayer pattern, after the boffin that came up with it. He found that by using twice as many green filters as blue or red, you got a sharper image. A demosaicing process turns this raw data into the full-colour grid of pixels in the recorded images.

Figure 7 Turning Light Into a Digital File (15)

As with any process, the quality of the end results depend on the quality of the components used to reach the final product. With cameras, this is no different. At any given point in the chain of the camera, the quality of each component used will affect the resulting image. It is due to this that no two models of camera are identical or produce the same results. While different models may be similar and produce similar quality images, the components used in them, even if it is just one different component, will result in images that have different characteristics.

When light enters a camera, the first component it passes through is the lens. It is due to this that camera manufacturers spend so much time designing lenses and working to create the best combination of lenses possible. While the term "lens" seems to imply that the lens of a camera is a singular piece of glass, camera lenses are actually made of several layers of glass. By including multiple layers of glass with varying curvatures and thicknesses, camera manufacturers are able to create the clarity and ability to zoom in on objects that consumers have come to expect in cameras today. While it appears that when you zoom in on a scene with a point and shoot camera or a DSLR, it is the extension of the lens housing that is causing the enlarged view, this is not the case. The extension of the lens housing is actually just allowing for the movement of the lenses enclosed within in order for them to move into a position at which they can magnify the scene being viewed. This magnified view can then be focused by adjusting the position of the focusing element, which is located at the rear of the lens housing, as seen in Figure 8.

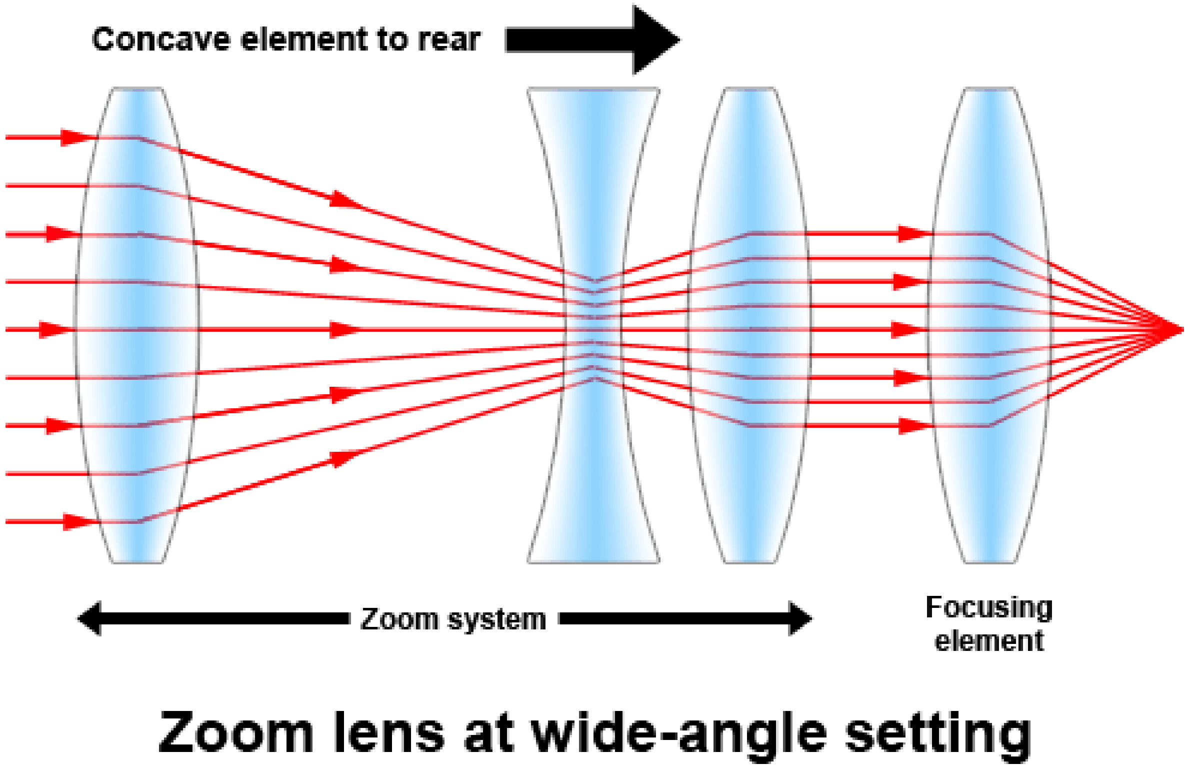


Figure 8: Zoom Lens Design

After the light passes through the lenses, it then hits the CFA. The most popular CFA used today is the Bayer pattern, as seen in Figure 9, though there are a wide variety of other CFA patterns that have been created by various companies. While some companies have chosen to create their own proprietary CFA patterns, many have decided to remain with the Bayer filter, which was named after Dr. Bryce Bayer who worked for the Eastman Kodak Company (17). The Bayer filter is designed on the RGB color scheme, and is composed of red, green, and blue filters that sit on top of every pixel and replace the black and white sensors that would otherwise be in their place, which allows for color images (17). These CFA filters function by filtering the light coming through at specific wave lengths so that each color of light is only able to pass through the CFA at designated points. Of these colored filters though, the grid in a Bayer filter is composed of twice as many green filtered pixels. By doing this, the Bayer filter is able to obtain a sharper perceived image due to the luminance channel being made up primarily of green information (17). An example of how the light passes through these CFA filters in order to create a color image can be seen in Figures 9 and 10.

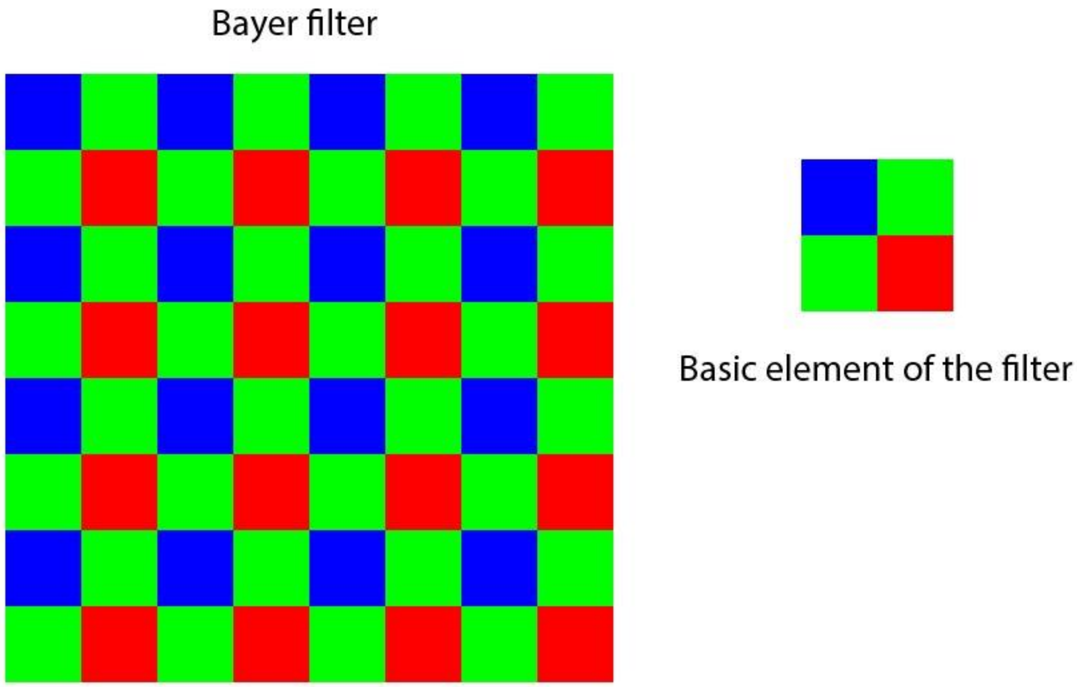


Figure 9: Bayer Color Filter Array (16)

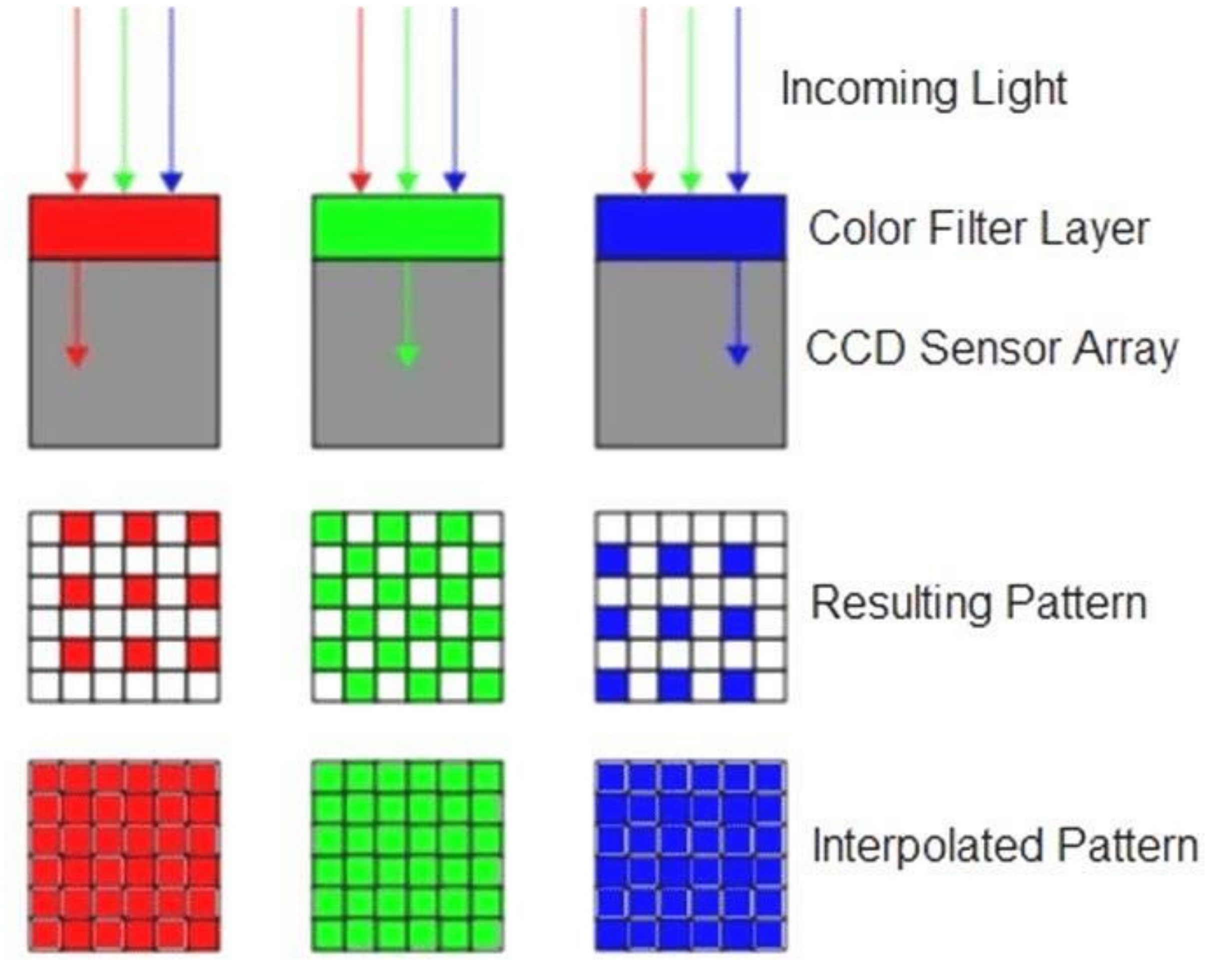
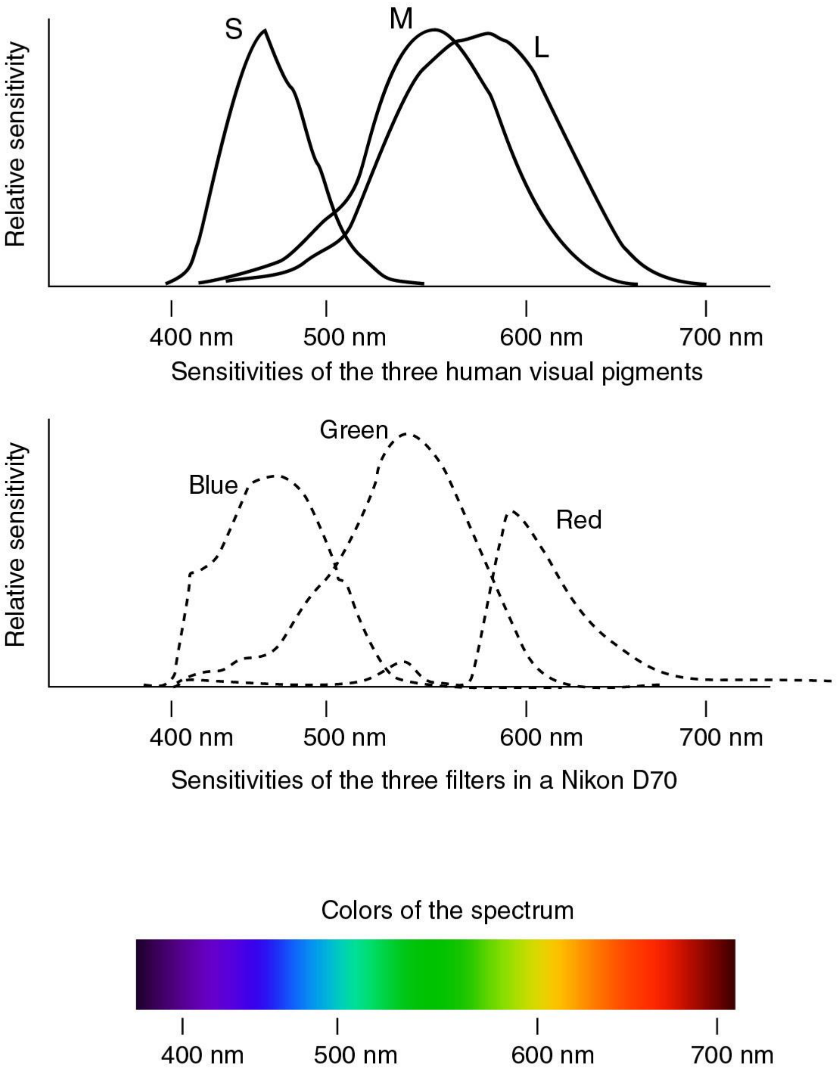


Figure 10: Obtaining the Interpolated RGB Image (18)

It is important to note that the reason 50% of Bayer filters are green is due to the human eyes perception of color. Though the human brain perceives all colors as being represented naturally and proportionally, the human eye is actually more perceptive to colors represented around 550 nanometers (nM) (19). In relation to the visible spectrum of light that can be detected by the human eye, which spans from roughly 380 nM to 700 nM, 550 nM represents the color green (19). Due to this, it was found that the higher percentage of green filters used in Bayer filters created a sharper image than would otherwise be produced if all of the filters were represented in equal proportion. The reason behind this is that since green represents the middle of the visible color spectrum to humans, this spectrum of light also affects the other perceived colors in the ranges both above and below it in the color spectrum. From this, it has been hypothesized that the human eye has evolved to its current state of being most sensitive to the green color spectrum so that the slight changes in color can be distinguished between blue to green and green to red. A visual display showing this correlation can be seen in Figure 11, where “S”, “M”, and “L” refer to Short, Medium, Long wavelengths (19).



Sensitivities of the three human visual pigments

Sensitivities of the three filters in a Nikon D70

Colors of the spectrum

Figure 11: Sensitivities of the Human Eye to Colors in the Visual Spectrum

Once the light has passed through the CFA, the filtered light hits the image sensor, which takes the raw image data and converts it into an image that contains the full color spectrum that is designated by the CFA. In the case of a Bayer CFA, this would mean that a full color RGB image is created, in which all three of the primary colors have their intensities represented within each pixel. It is through this process that the various colors seen with the human eye are recreated in photos, since all colors that can be seen with the human eye are made up through some combination of the colors of Red, Green, and Blue. Likewise, with CFA filters that operate on different wavelengths of light, such as Cyan, Magenta, Yellow, Key (Black) (CMYK), the intensities of light that are recorded blend together in the same fashion in order to create what is being viewed and photographed. An example of these color combinations can be seen in Figure 12.

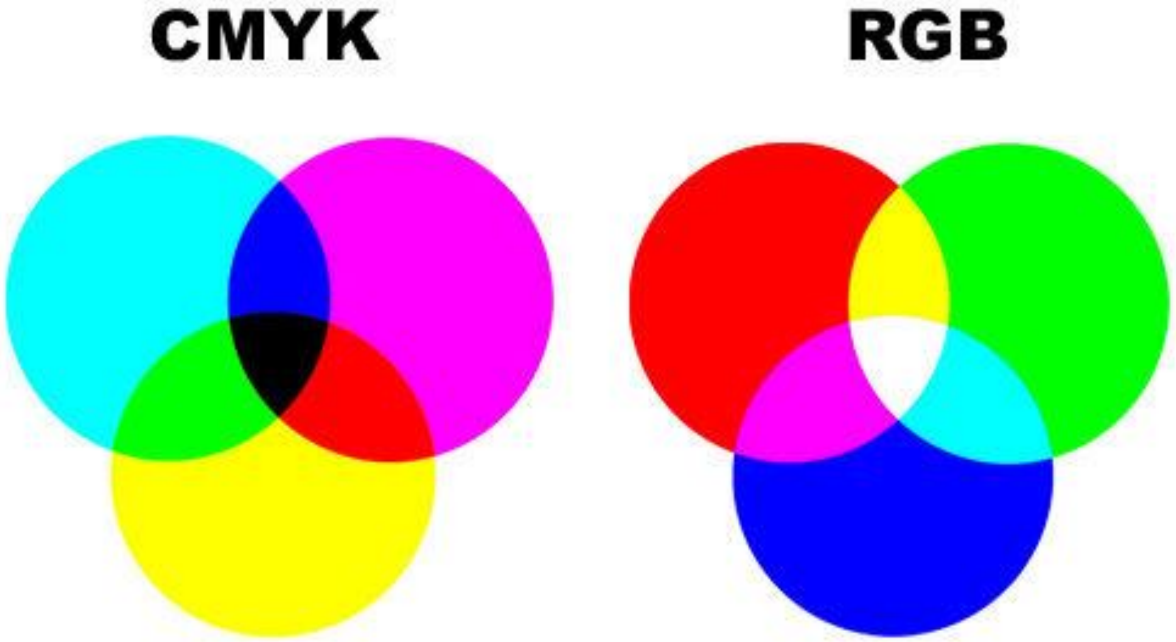


Figure 12: Color Combinations of RGB & CMYK

It is also important to note that there are two primary kinds of imaging sensors available today. The first is a Charged Coupled Device (CCD), and the second is a Complementary Metal Oxide Semiconductor (CMOS). While CCD imaging sensors have been around longer, and thus been further developed, there are pros and cons to both types of sensors that camera manufacturers take into consideration when determining which type of sensor to incorporate into their cameras (20).

The main factor that is considered for consumer-based cameras today is cost. Of the two, CCD sensors are more expensive, which in turn makes them less common in consumer level goods than CMOS sensors. In addition to this, CCD sensors function by measuring the luminosity at each pixel as an electrical charge, which is then transferred from the chip and can be outputted directly as an analog signal, or converted into a digital data signal through a separate chip. CMOS sensors, on the other hand, are made up of a single chip that provides all of the same type of processing, but in a much smaller package. Due to this and their lower cost, CMOS sensors are found in the majority of consumer-level cameras today, including nearly every smartphone, though there are a few exceptions. Outside of cost and size benefits though, CCD sensors tend to outperform CMOS sensors. In terms of Signal to Noise Ratio (SNR), CCD sensors are found to have a much lower SNR than CMOS sensors (20). Due to this, the images derived from cameras with CCD sensors tend to provide a higher image quality due to the lower SNR, which in turn allows them to provide a higher dynamic range. It has also been shown that CCD sensors are more sensitive to light than CMOS sensors, making them more adept in low light situations. Sometimes though, this can be a drawback to CCD sensors when used in bright environments. Overall, there are pros and cons to both CCD and CMOS sensors which have come to dictate which type of devices each sensor is likely to be found in, but both sensors provide the same overall task of converting the light that has been filtered through the CFA into electrons and outputting a signal, either digital or analog, which represents the photographed image.

Once the image sensor has created a full color image from what information it was given by the CFA, the ADC then converts the analog image data into a digital image. Once in the digital realm, the data is then passed on to the DIP, which handles a whole array of functions as assigned by the camera. While no two models of camera are exactly the same, the DIP usually performs functions such as White Balancing, Noise Reduction, Sharpening, and Compression. While these are some of the main functions that are performed by a digital cameras DIP, any feature that is found on the camera to alter, enhance, or manipulate the image being taken is handled by the DIP.

Of these processes, one of the most important functions of the DIP is to process the image and output it into the desired format. Higher end digital cameras, such as a DSLR's, give users the ability to save the images they take as lossless encoded files. This means that when the image is decompressed, no data from the original image is removed or lost. Most cameras though have been designed to compress images as much as possible, while still retaining the highest level of visual quality that can be attained. The reason for this, above anything else, is due to storage and transfer limitations that users are subject to today. Though technology is constantly progressing and providing users with larger, faster, and more compact storage mediums, the need to maximize the storage capabilities of any device is always something that is highly sought after. Also, most users of cameras these days are amateur photographers who are not looking to enlarge their images to hang on walls. Rather, their intentions are to share them online via social media applications such as Instagram.

Due to this, manufacturers are able to use very high levels of compression for digital images in order to maximize storage and transfer capabilities. To achieve such compression levels, many developers over the years have created new types of compression algorithms, all made with the goal of optimizing compression ratios while retaining the most clarity possible. Of all of the compression methods that have been created over the years though, there is one that has stood out and has become the industry standard for handling images. This method is known as JPEG Compression.

CHAPTER II

JPEG COMPRESSION

With the ever-growing use of digitized photos and images that are being shared among various devices and on the Internet, it was essential for developers to create a universal method for compressing images. Not only did developers see a need to minimize the file sizes that were being shared, but they also saw a need for a file type that could be used across various platforms. In today's digitally linked world, if images are not compressed before being shared, many services would not be able to afford to stay in business, even with all of the recent advances in storage and bandwidth capabilities (5). To address this issue, the International Telegraph and Telephone Consultative Committee (CCITT) decided that they should work together and create a joint working group to address the challenges of image compression. Through this joint effort, the Joint Photography Experts Group (JPEG) was established in November of 1986 (6). From this working group came many options and ideas for how to best compress digital images. Then, in 1991, after years of work and research, an open source code for JPEG was released that was based on the standards which had been tried and tested over the years. While this code was further enhanced and the standards solidified in the years to come, this was the birth of JPEG compression, which today has become the most widely used method for compressing digital images.

JPEG File Interchange Format (JFIF)

During this study, it was seen that all of the images that were downloaded from Instagram were recompressed to the JFIF format. While this still adheres to many of the JPEG file format specifications, it has some of its own specific specifications that make it an even more ideal format for being used with images that are being transferred on the internet. Unlike the Tagged Image File Format (TIFF) JPEG specification, the JFIF specification is a very minimal file format that's only purpose is to allow a large amount of applications and digital platforms the ability to transfer JPEG compressed images (7).

In order for a file to be recognized as a JFIF formatted file and not JPEG, the only mandatory requirement in the file structure is that the APP0 marker is located directly after the Start of Image (SOI) marker (7). This marker must be identified by a zero terminated string "JFIF," though the remainder of the APP0 marker can still be used for any other purpose so long as applications that view the file can distinguish between the JFIF APP0 marker (7). If other information is added to the APP0 marker, it must come immediately after the APP0 JFIF string. It can include information such as a one byte extension code that defines if a thumbnail is formatted in any other manner than 24-Bit RGB, or even an organizations name to identify where the file came from (7). If the data added to the APP0 marker is not able to be read or is not recognized by an application, the decoder that is being used will simply skip over the added segment and continue decoding the file in its standard method. In order to help ensure that decoders are able to read the files that are being encoded in the JFIF format, it is also highly recommended that the JPEG baseline process be used in order to ensure that the file is as compatible as possible with applications that support JPEG (7).

For color images, the JFIF format follows the International Telecommunications Union Recommendation (ITU-R) BT.601 (formerly known as Consultative Committee on International Radio (CCIR) Recommendation 601), which specifies that RGB values should be converted to Luminance, Chrominance Blue, Chrominance Red (YCbCr) in order to maximize the range of signal values that are available versus in the RGB spectrum (8). This encoding method uses a ratio of 4:2:2, in which the chrominance channels (Cb and Cr) are half the resolution of the luminance channel (Y) (6), and normalizes the colors so that they use all 256 levels of an 8-bit binary encoded image (7). The formula for this conversion from RGB to YCbCr can be seen in Figure 13.

$$Y = 0.299R + 0.587G + 0.114B$$

$$Cb = - 0.1687\,R - 0.3313\,G + 0.5\,B + 128$$

$$Cr = 0.5\,R - 0.4187\,G - 0.0813\,B + 128$$

Figure 13: RGB to YCbCr Conversion (7)

The values seen here are determined by using the normalized signal values for white, black, saturated primary colors, and complementary colors as seen in Table 1 (8).

Table 1: Normalized signal values (8)

Condition	E_R'□	E_G'□	E_B'□	E_Y'□	E_R'□– E_Y'□	□ E_B'□– E_Y'□
White	1.0	1.0	1.0	1.0	0	0
Black	0	0	0	0	0	0
Red	1.0	0	0	0.299	0.701	-0.299
Green	0	1.0	0	0.587	-0.587	-0.587
Blue	0	0	1.0	0.114	-0.114	0.886
Yellow	1.0	1.0	0	0.886	0.114	-0.886
Cyan	0	1.0	1.0	0.701	-0.701	0.299
Magenta	1.0	0	1.0	0.413	0.587	0.587

When dealing with how to best compress the data contained within an image, the JPEG developers determined that the best way to consolidate the information that was being viewed was to break it up into more manageable sections. Due to the limitations of the human eye, developers were able to exploit the eye's lack of sensitivity to high frequencies in the color spectrum by compressing that color information at a higher ratio than in the lower ranges with nearly no identifiable difference to the human eye (5). It was then determined that the most efficient way to handle all of the data contained within an image when it comes to compression would be to break an image up into many different blocks of pixels to take advantage of the correlations inherent within each box (6). By doing this, developers were able to take significantly larger portions of data at once, much more than would be possible if the compression algorithm looked at pixels individually, and were able to obtain much more favorable results than would be possible if the entire image was compressed at once (5).

For this to work optimally, the process had to be tested to determine the ideal block size. The factor that was considered for this was what block size would be able to find the greatest level of correlation between pixels. If the block size were too small, some correlation between pixels would be missed, whereas if the block size was too large, then correlation between pixels that does not exist might be registered (6). Due to this, it was determined that an 8x8 block size provided optimal results by catching correlations that the 4x4 block size missed and by providing more accurate results with lower processing power than resulted from a 16x16 block size (6).

Compression Process

In order to compress an image, the JPEG format uses four distinct processes as listed here (5).

1. Division of the Image
2. Conversion of the Image Data from the Spatial Domain to the Frequency Domain
3. Quantization

4. Entropy Encoding

For 8-bit grayscale images, this process is completed once and results in a compressed JPEG image. For a 24-bit color image though, this process is repeated three separate times since JPEG treats each layer of a color image as a grayscale representation of that color (5). An example of this process can be seen in Figure 14 (5).

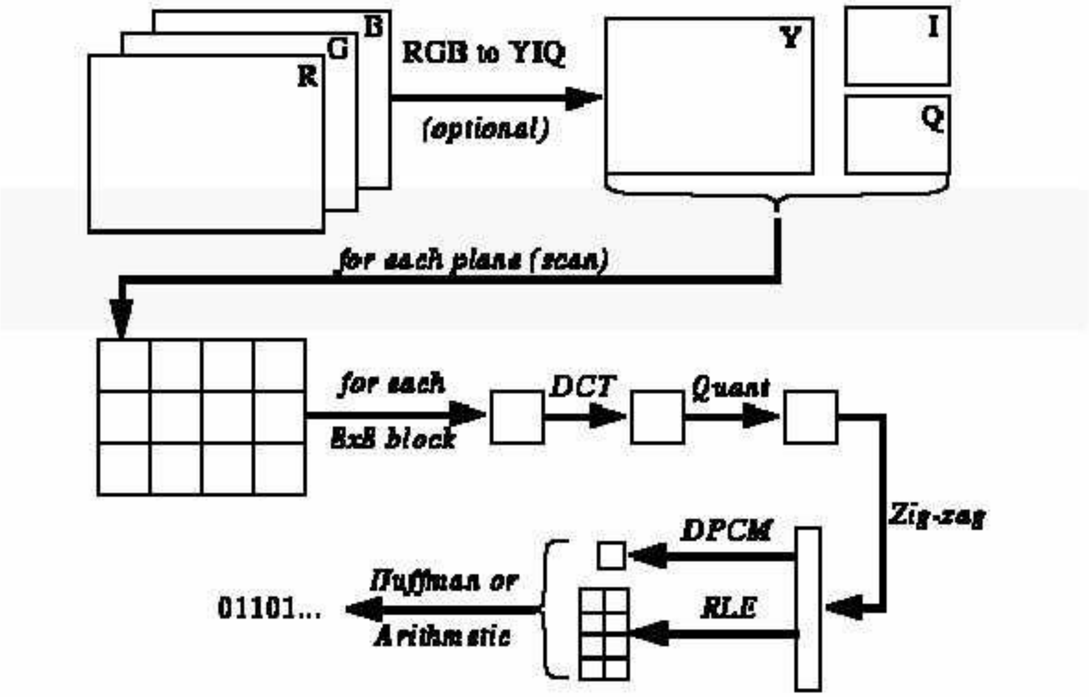


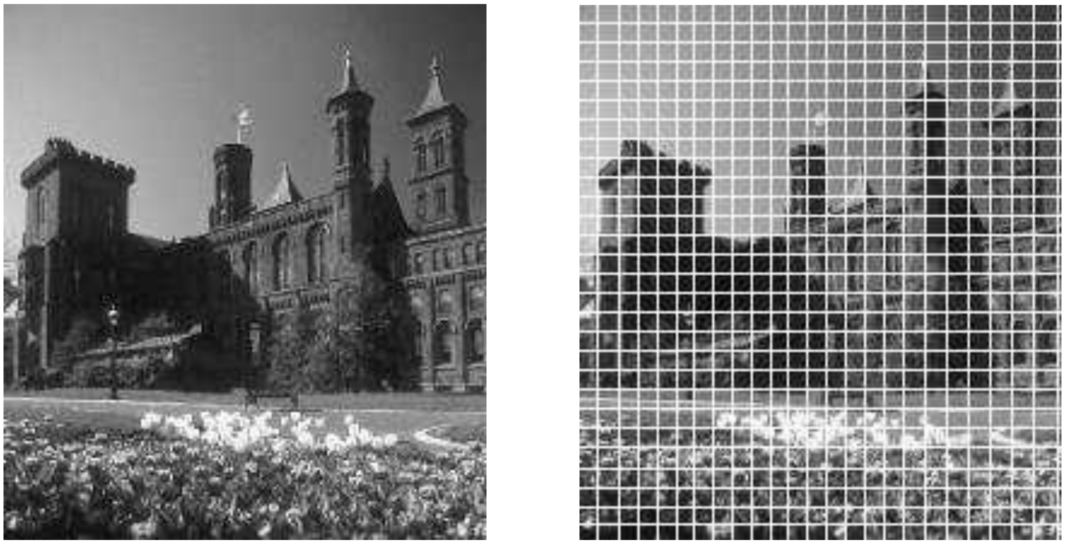
Figure 14: JPEG Compression Order of Operations (5)

One additional factor that determines the end result of a JPEG compressed image is the quality factor that is chosen. Unlike many other compression algorithms, JPEG allows users to chose a quality factor that will determine how heavily the image is compressed. This quality setting ranges from 1-100, where 100 will provide a resulting image with a minimal amount of data lost, whereas a quality factor of 1 will compress the image as heavily as possible and will result in the maximum amount of lost data and heavy blocky artifacts.

Though the option to compress with a quality factor of 100 may seem ideal due to a minimal loss of data, most studies suggest that a quality factor over 90-95 does not improve the visual appearance of the resulting image to the human eye, though it does dramatically increase the size of the resulting file.

Step 1: Division of Image

The first step in the JPEG compression process is the division of the image into 8x8 blocks of pixels, which allows the algorithm to take advantage of the correlation of colors that occur in small sections of an image. This process is initiated in a top-down format, where the first blocks are established at the top left of the image and continue from left to right, top to bottom, until they reach the end of the image at the bottom right (7). If the image being processed has any dimension that does not equal multiples of 8, then the bottom and right side of the image will be padded with extra dummy pixels in order to create full 8x8 blocks throughout the entire image (5). With this completed, every block of 64 pixels can then be processed individually. To help visualize this, Figure 15 below shows an image before it has been broken up into 8x8 blocks and afterwards. It should be noted that these images have been doubled from their original size of 200 pixels by 220 pixels in order to increase the visibility of the 8x8 blocks. Also of note is that while there are lines on the right image to represent the division of the image into 700 blocks of 8x8 pixels, this is a logical process and the lines have only been added for visual clarity.



Before After

Figure 15: Visual Demonstration of the Division of a 200 Pixel x 220 Pixel Image (5)

During this step the color space conversion also occurs (5). This conversion, while optional for standard JPEG compression but mandatory for the JFIF specification, can be done either before or after the image is divided into 8x8 blocks (5). The reason for this conversion is because the YCbCr color space is able to compress more efficiently than the RGB color space, making it ideal to maximize the compression of an image. Also of note is that each pixel of a color image is made up of three bytes, one for each color representation such as RGB or YCbCr. Likewise, when dealing with a grayscale image each pixel is made up of only one byte. Each of these bytes is assigned a value of 0-255, with 0 representing complete black and 255 representing complete white. It is through the designation of the bytes that a pixel is able to represent a specific color depending on how each color channel is represented, or what shade of gray a grayscale image will have (5).

The final step of this first phase of the JPEG compression process is to subtract 128 from each byte from every pixel in the image (5). This changes the values of every byte from 0 to 255, to -128 to 127. By doing this, the algorithm is able to make it so that the average value of every pixel will be closer to zero rather than somewhere around 128, which allows for quicker computations of each 8x8 block within an image by the computer handling the processing (5).

Step 2: Conversion from Spatial Domain to Frequency Domain

In order to optimize the efficiency of the quantization of an image, the values of each pixel are converted from the spatial domain to the frequency domain (5). This is done by applying 2D Discrete Cosine Transform (DCT) to each byte (5)(6). Though DCT is formulated to perform on one-dimensional data, it is able to be used on two-dimensional image blocks by adding a second summing term to the equation in order to make it function in two dimensions (5). The equation used to obtain this result is shown in Figure 16, where C(x)=1/√2 if x=0, while C(x)=1 in all other cases (5).

$$F(u,v)=\frac{1}{4}\cdot C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)\cos\frac{(2x+1)\cdot u\pi}{16}\cos\frac{(2y+1)\cdot v\pi}{16}\right]$$

Figure 16: 2D Discrete Cosine Transform Equation

f(x,y) = 8-bit image value at (x,y)

F(u,v) = New value in the frequency domain

Once transformed into the frequency domain, the JPEG algorithm is then able to distinguish which values are of greater or lesser importance to the image based on the properties of human vision. Since the eyes vision decreases with higher frequencies, the algorithm is able to remove or de-emphasize this data that has now been transformed into the frequency domain in order to reduce the file size of the image in question (5). While doing this dramatically changes how the computer views the image file, to the human eye the image still looks very similar to the original, though the amount of difference will depend on how heavily the image is compressed (5).

Step 3: Quantization

In order for the quantization of the image data to take place, the quality settings must be chosen by the user, or in some cases the manufacturer if the quality setting is predefined and not made adjustable such as on some smartphones. It is at this point that the user or manufacturer will designate the quality factor that was previously described that will determine how lightly or heavily the image data will be compressed. While the options range from 1-100, the numbers that are used by the algorithm are typically arbitrary, though predefined, and will divide the cosine coefficients in the data matrix. Once completed, the resulting coefficients will then be rounded to the nearest integer. A widely accepted formula for this quality measurement is a formula that is referred to as the Quality Measurement of the Reconstructed Image (PSNR) (10). This formula uses the distortion measurement known as the Mean Square Error (MSE) in order to come to the defined quality measurement that can be set by the user (10). The formulas for these calculations are shown in Figure 17 and Figure 18.

$$MSE = \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} [X(i,j) - Y(i,j)]^2$$

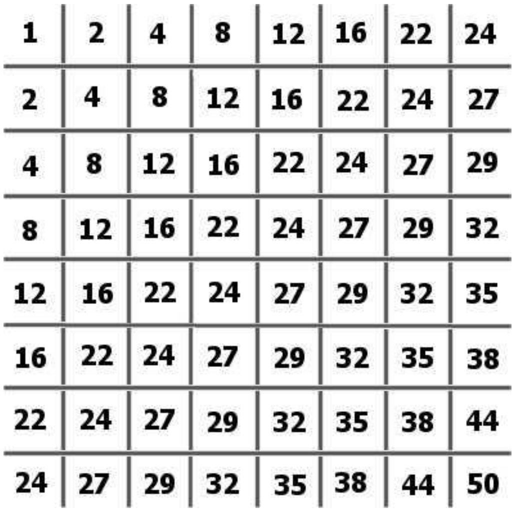
Figure 17: Mean Square Error (MSE) (10)

$$PSNR = 10 \log_{10} \frac{255^2}{MSE} (dB)$$

Figure 18: Quality Measurement of the Reconstructed Image (PSNR) (10)

During this process, there are some trade-offs that are taken into consideration as to how efficiently and accurately the algorithm will process the image data. The smaller the values that are used to divide the frequency data, the more accurately the resulting values will be once rounded to the nearest integer. However, this will result in a minimal compression ratio of the final image. Conversely, the larger the values that are used to divide the frequency data the greater the compression ratio will become as more of the frequency values are rounded to zero (5)(10). It should be noted though that while this results in a greater level of compression, it also results in an increasingly higher level of errors due to the lack of precision of the resulting values that are rounded to the nearest integer.

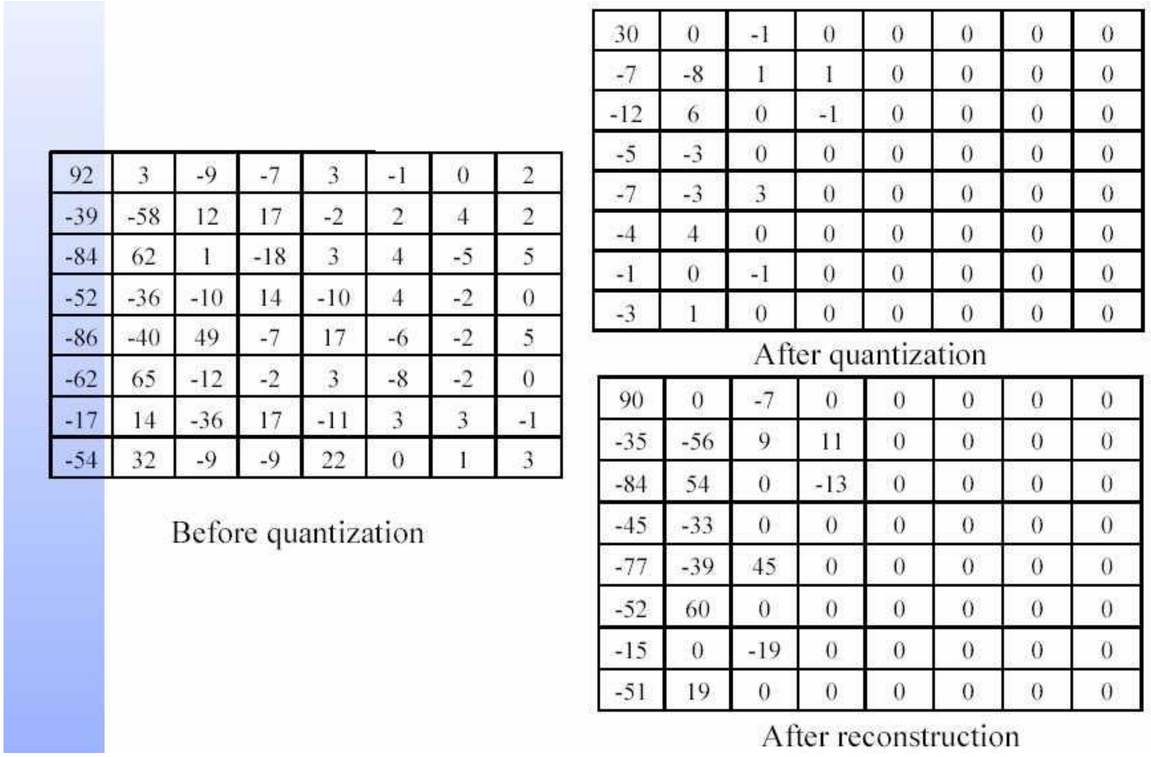
Once the quality factor has been chosen, the correlating constants are then entered into an 8x8 matrix, identical to the matrix of pixels that were established in Step 1, which is called a quantization table (5). It is important to note that for YCbCr images, there will be two quantization tables that will be created, one each for the luminance and chrominance components of the image in question (10). These constants range from 1 to 255, with 1 representing no loss of data for that specific coefficient, and 255 representing a maximum loss of data for that specific coefficient (5). Since the quantization table is the exact size of the pixel matrixes that was previously established, each value in the quantization table will be assigned to the correlating location of the pixel matrix that contains the frequency values. Standard quantization tables usually have values distributed symmetrically through the diagonal axis, from left to right and top to bottom, and have the lowest values distributed in the upper left corner and they gradually grow to the largest values which are distributed in the lower right corner as seen in Figure 19 below (5).



1	2	4	8	12	16	22	24
2	4	8	12	16	22	24	27
4	8	12	16	22	24	27	29
8	12	16	22	24	27	29	32
12	16	22	24	27	29	32	35
16	22	24	27	29	32	35	38
22	24	27	29	32	35	38	44
24	27	29	32	35	38	44	50

Figure 19: Sample Quantization Matrix (5)

Once the quantization table is selected, that table is saved in the metadata of the JPEG file so that it can be referred back to in order to decompress the file. Due to the fact that JPEG compression is a lossy compression, or in other words it does not retain the original state of the binary data, once compressed, the image will never be a duplicate copy of the original. With the quantization table to show how it was compressed though, the file can be uncompressed with a high degree of accuracy, though the high frequency data that was removed during compression will not be able to be recovered, as seen in Figure 20 (5).



92	3	-9	-7	3	-1	0	2
-39	-58	12	17	-2	2	4	2
-84	62	1	-18	3	4	-5	5
-52	-36	-10	14	-10	4	-2	0
-86	-40	49	-7	17	-6	-2	5
-62	65	-12	-2	3	-8	-2	0
-17	14	-36	17	-11	3	3	-1
-54	32	-9	-9	22	0	1	3

Before quantization

30	0	-1	0	0	0	0	0
-7	-8	1	1	0	0	0	0
-12	6	0	-1	0	0	0	0
-5	-3	0	0	0	0	0	0
-7	-3	3	0	0	0	0	0
-4	4	0	0	0	0	0	0
-1	0	-1	0	0	0	0	0
-3	1	0	0	0	0	0	0

After quantization

90	0	-7	0	0	0	0	0
-35	-56	9	11	0	0	0	0
-84	54	0	-13	0	0	0	0
-45	-33	0	0	0	0	0	0
-77	-39	45	0	0	0	0	0
-52	60	0	0	0	0	0	0
-15	0	-19	0	0	0	0	0
-51	19	0	0	0	0	0	0

After reconstruction

Figure 20: Data Matrix Before & After Quantization

The purpose for this division, and the reason for rounding the resulting numbers to their nearest integers, is to establish as much uniformity amongst neighboring pixels as possible. This uniformity will then allow the algorithm to identify correlations that can be leveraged to maximize the compression of the resulting file (9). It is due to this that larger constants in the quantization matrix are used with increasing frequency as the level of compression increases in order to create as many zero level amplitudes as possible. These zero level amplitudes correlate to high frequencies that will be removed, whereas the nonzero amplitudes correlate to low frequencies that will be retained (6). The basic formula for this equation is provided below in Figure 21 (10).

$$F(u,v)_{Quantization} = round\left(\frac{F(u,v)}{Q(u,v)}\right)$$

Figure 21: Quantization Equation

Step 4: Entropy Encoding

In order to take the quantized information and turn it into a more manageable string of data to be compressed and handled, the JPEG formula then lines up all of the data to create a single string by using entropy encoding (5). This type of coding works by reading the values that have been established in the quantized pixel matrix in a zigzag format, which optimizes the time spent to read all of the data contained within. An example of this can be seen in Figure 22 (5).

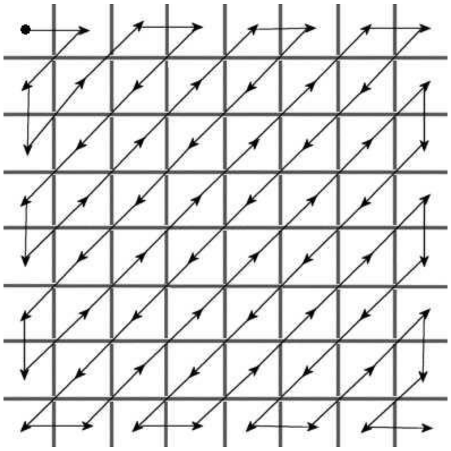


Figure 22: Entropy Encoding Format (5)

Once the data is configured into a single string, a process called Run Length Coding (RLC) can be performed, which was developed by Jørgen Vaaben (9). RLC takes advantage of the large number of zeros present in the string by assigning a "value pair" before each string of zeros. The "value pair" is made up of two numbers which designate how many zeros to skip before the next non-zero amplitude (first number) and how many bits are required to represent the amplitude (second number) (6). Once the entire string is compressed using RLC, an end of block marker is added to the string so that decoders can identify where one block ends and the next begins (5).

After the RLC process is complete, the new data string for each block in encoded using Huffman coding. This functions by scanning the data string to identify frequently and infrequently occurring data. By identifying these trends, Huffman coding is able to assign fewer bits to data that occurs more frequently and more bits to data that occurs less frequently (5). To do this, a Huffman table is created for bits that occur more frequently, which can then be referenced by the decoder in order to distinguish the entire value for the data in question (5). By doing this, the algorithm is able to reduce the final image file size even more while still retaining the pertinent information since it is more efficient to save a regularly recurring value once and simply insert a reference code to identify it rather than allowing that reoccurring piece of data to be repeated many times and enlarge the final file size with every iteration it has. When decoding the file in question, the decoder simply has to reference the metadata of the file where the Huffman tables used to encode the information are stored, and which contain the entire value of bits of data that were supplemented with a unique identifier.

CHAPTER III

STUDY PREPARATION

Image Preparation

In order to ensure bias was mitigated as much as possible for this study, it was determined that the images that were used would all under go the same treatment before they were uploaded, once they were uploaded, and once they were downloaded. Due to this, it was determined that for this initial study of how Instagram recompresses images, that it would be best to use one hundred percent unaltered original photos. It was also determined that due to time and technological constraints, it was necessary to use a small test database with the most common base settings utilized in Instagram.

While there are many more studies published on a multitude of social media sites used today, none have been conducted on Instagram due to the inability to use an Application Programming Interface (API), which at the time of this study could not be developed (11). By using an API, researchers would be able to easily and efficiently upload and download thousands of images in a streamlined fashion in order to obtain a very large data set. This would certainly be the optimal way to conduct a study of this sort because it allows researchers to view very large and overarching trends across the entire data set. However, due to the inability to use such a program for this study, the data set was required to be kept to a minimal size. Due to this limitation, it was deemed that the best course of action was to select two separate data sets that would be used and studied for this project.

The first is the larger of the two data sets in which the goal was to simply gain an understanding of how Instagram handles images of varying dimensions. Due to the ever-widening ranges of image dimensions available across all different makes and models of cameras and smartphones, it was essential to see how Instagram handles all sizes of photos and what the input and output results would be in each case. In order to do this, a test database of 84 images will be used which is comprised of a wide range of image dimensions. In order to ensure there are no discrepancies between the images used for this database, all 84 images will be created using a single host image, and be modified in Photoshop to the requisite dimensions. Due to this, it should be noted that all 84 images used in this test will be configured in portrait orientation, so a similar test with images configured in a landscape orientation could be beneficial in the future.

The second data set was the test images that were going to be used to identify how the file itself was altered by Instagram's recompression process. Due to the limitations described above, it was determined that this data set would simply be comprised of six images taken from three different models of smartphones. While this data set did not take into account any stand alone digital cameras, it was determined that smartphones should be the focus of this study due to their inherent tie to Instagram, which only allows a user to upload images through its mobile platform. Due to this factor, and the fact that there are an estimated 224.3 million smartphones in use in the United States alone, and over 2 billion worldwide, it was determined that for forensic purposes images from smartphones would be the most common images to be retrieved from Instagram (12).

It was then determined that the three models of smartphones to be used would be the Apple iPhone 6s, Samsung Galaxy S7, and the Motorola Droid Turbo II. From these phones, two original images would be selected from each phone and shared with the other two, making for a total of six images on each phone, and a total of eighteen images that would be uploaded, downloaded, and studied.

Image Retrieval

While with many social media platforms it is possible to directly extract or download an image that a user has uploaded, with Instagram a 3rd party software must be utilized in order to obtain posted images. Due to this, it was necessary to research what methods and applications are available to extract images from Instagram on both desktop and mobile applications and to set up a third test to determine how they function. Once a selection of methods and applications was made, these processes/applications were then all utilized in this third test to download/extract the same five images from the Instagram test profile in order to see how they all affected the images that they were tasked with retrieving. In order to compare the results of each method, the metadata, quantization tables, and stream hashes, which are basically a digital "fingerprint" for the image data or the data stream contained within each file, as compared to a traditional file hash which is a "fingerprint" of the file as a whole, that is displayed as a unique alphanumeric string, were studied. For the analysis of the metadata and the quantization tables, Exiftool (21) and JPEGsnoop (22) were used respectively, while ExactFile (23) was used to extract the MD5, SHA1, and SHA256 hashes of the recovered images.

By formulating the analysis in this manner, the study was able to identify how each downloader altered the files it was tasked with retrieving in comparison to the others. In conjunction with this, it was also possible to compare the extracted files with their original counterparts to see what exactly changed and which if any of the methods worked in a similar fashion to each other. By utilizing several different methods for extracting images from Instagram, it was deemed possible to determine the ideal method for image recovery on both mobile and desktop platforms. The results that were determined to provide the most ideal and transparent results would then be used to conduct the rest of the study utilizing the two test databases that had been created. After researching the methods and applications available to extract images from Instagram, the following options were chosen to be compared with one another in order to determine the ideal method for extracting images from Instagram.

- 4K Stogram (Desktop Program)
- Dinsta.com (Desktop Program)
- Download Gram (Desktop Program)
- Page Source Download (Desktop Operation)
- Page Source Download (Mobile Phone Operation)
- InstaSave (Mobile Phone Application)
- SaveGram (Mobile Phone Application)

CHAPTER IV

METHOD

Upload Method

For the first database of images that was used to test how Instagram handles images of varying dimensions, 84 images were created to represent image dimensions seen in various products used today. In order to mitigate any alterations or data that may be inserted into the file of an image by any device used, all 84 images were derived from the same source image, which was reconfigured into the requisite dimensions using Photoshop. By doing this, it was ensured that the structure of each image was identical and the only difference between each image was its varying dimensions and file size, which allowed for an ideal test database where no third party data would skew the results.

To create the second test database of images that would be used for this study and uploaded from three mobile devices, two original images were selected from each mobile device. These images were then directly extracted from each mobile device to a computer, as well as transferred via email and Dropbox to confirm that the resulting images were bit stream duplicates of the originals from each device. Once confirmed that each image was a bit stream duplicate of the original, each image was loaded onto the other two devices from which it did not originate. Once loaded onto each device, these images were once again extracted from each device and hashed to confirm that they were still bit stream duplicates of the originals.

After each device was loaded with its six test images, the images were uploaded to Instagram to the test account (https://www.instagram.com/douglas_thesis_research/) by using the Instagram Mobile App, which was downloaded onto each of the three devices. During this process, the following settings were used on Instagram:

- Upload Quality: Normal
- Cellular Data Use: Default
- Default Cropping
- Filter: Normal
- No Edit or Lux adjustments
- Comments: "Device Name" Upload "Image Name"
  - Ex: iPhone Upload Samsung Image #1

By following this procedure, it was determined that every image that was used for this study would be uploaded in its original and unaltered format. It was also determined that every image would be processed in an identical manner and that each image would be able to be identified once downloaded so that data comparisons could be made.

Extraction Method

In order to extract images from Instagram, the first step will be to determine which tool or method will provide the most forensically viable results in the most efficient manner possible. Since it is known that some tools available today allow for multiple images to be downloaded at once, whereas others only allow for a single image to be downloaded at a time, multiple tools will be tested to compare the data from the resulting images. This will allow any differences in the resulting images to be notated in order to determine which method produces the highest quality image that is the most representative of the original image that was uploaded to Instagram and that provides the most forensically viable data possible for examination.

Once the most ideal tool or method has been established, that tool will be directed to the Instagram user page of interest. In the case of this study, it will be the test account that has been created and loaded with the test images to be used in this study. Once directed to this account, images will then be retrieved in the most efficient manner possible, which will be dependent on the tool used. Ideally, this will be a tool that can download multiple images from a users account in one step due to the need for efficiency when being used in forensic investigations, however, the integrity of the material retrieved is of the utmost importance and can not be compromised at any cost.

Once the images in question have been downloaded, bit stream duplicates will be made in order to ensure that the original data is not corrupted at any point in the investigation. These working copies will then be moved to a working folder from which all of the analysis on the images will be conducted. By doing this, it will ensure that the originally retrieved evidence will not be altered or corrupted. Hash verification of the files will also be made and documented to establish a chain of custody and show that the working files are confirmed to be bit stream duplicates of the evidentiary files.

Analysis Procedure & Tools

In order to analyze the data that was collected, it was determined that the following analysis would be conducted on both the source and collected materials:

- File Hash
- Stream Hash
- Structure Analysis
- Quantization Table Analysis
- Resolution Analysis

- Cropping Analysis

Due to the two different data sets used in this study, it was determined that not all analyses needed to be conducted on both data sets. For the data set used to test how Instagram handles images of varying resolutions, the only objective was to determine the input and output resolutions of the images included in the database. For this reason, the only analyses that were necessary were of the input/output resolutions, the quantization tables used on the output images, and how the output images were cropped, if at all. For the mobile device data set, all facets of the output images were of interest, so a more extensive analysis was performed. This consisted of hashing the input/output files, a structure analysis of the input/output images, quantization table analysis, analysis of input/output resolutions, and an analysis of how each image was cropped.

The first step taken was to document all of the original test images to record their initial states. This included hashing the files and performing a file and structure analysis. The tools used to perform this were JPEGsnoop, Exiftool, and ExactFile. The results of these tests are listed in Table 2.

Table 2: Mobile Phone Input Data
QF = Quality Factor

Image	Hash		Baseline or Progressive	Image Size	QF
Motorola XT1585					
Motorola Image #1	File Hash:	MD5: 50241feffe43fd6ac7c5b5e2814cbc5e	Baseline DCT	Width: 4008	95
	Stream Hash:	MD5: d1e9502e5ce1c3b29015c3ac46f69f0a		Height: 5344	
Motorola Image #5	File Hash:	MD5: 910be2aee7f968b0fff2c8eb35ee5de6	Baseline DCT	Width: 4008	95
	Stream Hash:	MD5: fe3e6e197e84c6d05fdbc955e7e6e251		Height: 5344	
Samsung Image #1	File Hash:	MD5: b6c0ae11bec4a1b5e5b4fb731b8f087c	Baseline DCT	Width: 4032	100
	Stream Hash:	MD5: 2732a4a2ba706811f55c8c3ffb0bc4ed		Height: 3024	
Samsung Image #2	File Hash:	MD5: 5d07288264f816ee9f3bb21a255ece99	Baseline DCT	Width: 4032	100
	Stream Hash:	MD5: 92f95ee8427a9fc6d81eda1d5771491e		Height: 3024	
iPhone Image #1	File Hash:	MD5: 53b24698e50a367349cc9149f2faeffc	Baseline DCT	Width: 3264	96
	Stream Hash:	MD5: 115c8f950a7d339ec12feb0a9fd979a6		Height: 2448	
iPhone Image #2	File Hash:	MD5: a20e7c124fb0234607c8637515de023a	Baseline DCT	Width: 3264	93
	Stream Hash:	MD5: 47bcd32e6693f48c93228bb1f039ca27		Height: 2448	
Samsung S7					
Motorola Image #1	File Hash:	MD5: 50241feffe43fd6ac7c5b5e2814cbc5e	Baseline DCT	Width: 4008	95
	Stream Hash:	MD5: d1e9502e5ce1c3b29015c3ac46f69f0a		Height: 5344	
Motorola Image #5	File Hash:	MD5: 910be2aee7f968b0fff2c8eb35ee5de6	Baseline DCT	Width: 4008	95
	Stream Hash:	MD5: fe3e6e197e84c6d05fdbc955e7e6e251		Height: 5344	
Samsung Image #1	File Hash:	MD5: b6c0ae11bec4a1b5e5b4fb731b8f087c	Baseline DCT	Width: 4032	100
	Stream Hash:	MD5: 2732a4a2ba706811f55c8c3ffb0bc4ed		Height: 3024	
Samsung Image #2	File Hash:	MD5: 5d07288264f816ee9f3bb21a255ece99	Baseline DCT	Width: 4032	100
	Stream Hash:	MD5: 92f95ee8427a9fc6d81eda1d5771491e		Height: 3024	
iPhone Image #1	File Hash:	MD5: 53b24698e50a367349cc9149f2faeffc	Baseline DCT	Width: 3264	96
	Stream Hash:	MD5: 115c8f950a7d339ec12feb0a9fd979a6		Height: 2448	
iPhone Image #2	File Hash:	MD5: a20e7c124fb0234607c8637515de023a	Baseline DCT	Width: 3264	93
	Stream Hash:	MD5: 47bcd32e6693f48c93228bb1f039ca27		Height: 2448	

Table 2: Mobile Phone Input Data (Continued)
QF = Quality Factor

Image	Hash		Baseline or Progressive	Image Size	QF
iPhone 6s					
Motorola Image #1	File Hash:	MD5: 6f64e3f6c0a82df2d9f5a5cc25fcf66e	Baseline DCT	Width: 852	93
	Stream Hash:	MD5: 738a1fad67395eb46f14cf922955a473		Height: 1136	
Motorola Image #5	File Hash:	MD5: 6ef4185d806bc921bfedf12701da6bf1	Baseline DCT	Width: 852	93
	Stream Hash:	MD5: d79fb96062dc6255ffd92a4a501dd497		Height: 1136	
Samsung Image #1	File Hash:	MD5: 35f49758d2a2a559fd491bd50c89502a	Baseline DCT	Width: 852	93
	Stream Hash:	MD5: b365f3a617f02a5b7cb5bae86ca75edc		Height: 1136	
Samsung Image #2	File Hash:	MD5: df71b7dd8f0e3c0ed5fdca957fd8f237	Baseline DCT	Width: 852	93
	Stream Hash:	MD5: dadbc53662eb2debf908419ef3002570		Height: 1136	
iPhone Image #1	File Hash:	MD5: 53b24698e50a367349cc9149f2faeffc	Baseline DCT	Width: 3264	96
	Stream Hash:	MD5: 115c8f950a7d339ec12feb0a9fd979a6		Height: 2448	
iPhone Image #2	File Hash:	MD5: a20e7c124fb0234607c8637515de023a	Baseline DCT	Width: 3264	93
	Stream Hash:	MD5: 47bcd32e6693f48c93228bb1f039ca27		Height: 2448	

With this data collected, it was then possible to determine what changes were made to the images after they were uploaded to Instagram. It is also important to note that the photos loaded onto the iPhone 6s resulted in different file hashes, which indicate that the files were altered from their original state. From this, it can be surmised that since the stream hashes and the quality factors were altered as well, that these images were re-encoded by the iPhone's file system. While this process was attempted in a multitude of ways including a direct file transfer, Dropbox, and Gmail, it was discovered that no matter how the file was loaded onto the iPhone, it was altered by the file system. It is hypothesized that this happens due to iPhones unique file structure in which every image must be named in a specific fashion in order to be read and organized. Due to this, the iPhone will alter any image loaded onto it in order to make it compatible with the phones database system. While this did affect how much data was able to be gathered in regards to how the same images uploaded from different devices would be processed by Instagram, it provided valuable insight into what an image from an android device may look like if it has been loaded onto an Apple device.

CHAPTER V

RESULTS

Image Retrieval Programs

It is not currently possible to directly download a user image from Instagram, either through the mobile application or the website interface. However, there are a number of third party software applications that have been created in order to extract a copy of the image in question. Of the applications available today, five were selected for this study that had the ability to download images on Instagram from either a mobile device or a desktop computer. Along with these applications, a method of extracting the JPEG image from the page source coding on Instagram was also utilized on both a mobile device and a desktop interface.

Of these five applications and two methods used, it was found that six of the seven methods produced bit stream duplicates of each other when downloading the same source image. The six applications and methods that were found to produce identical results were the following:

- Dinsta (Desktop Download)
- DownloadGram (Desktop Download)
- Page Source Download (Desktop Download)
- SaveGram (Mobile Device Download)
- InstaSave (Mobile Device Download)
- Page Source Download (Mobile Device Download)

The one application tested that was found to produce different results was the following:

- 4K Stogram

The results of this comparison are shown in Table 3.

Table 3: Image Retrieval Applications & Methods Results

Image #1		
Downloader	**Hash**	
4K Stogram (Desktop)	File Hash:	MD5: 10a69547cdec18afd5712804c6beea8d
	Stream Hash:	MD5: 256ab6adf9f6cb7296193b5cc5cc359e
Dinsta (Desktop)	File Hash:	MD5: a9b222bc82f7cb793bd05e5b99437cf8
	Stream Hash:	MD5: 256ab6adf9f6cb7296193b5cc5cc359e
DownloadGram (Desktop)	File Hash:	MD5: a9b222bc82f7cb793bd05e5b99437cf8
	Stream Hash:	MD5: 256ab6adf9f6cb7296193b5cc5cc359e
Page Source Download (Desktop)	File Hash:	MD5: a9b222bc82f7cb793bd05e5b99437cf8
	Stream Hash:	MD5: 256ab6adf9f6cb7296193b5cc5cc359e
SaveGram (Mobile Device)	File Hash:	MD5: a9b222bc82f7cb793bd05e5b99437cf8
	Stream Hash:	MD5: 256ab6adf9f6cb7296193b5cc5cc359e
InstaSave (Mobile Device)	File Hash:	MD5: a9b222bc82f7cb793bd05e5b99437cf8
	Stream Hash:	MD5: 256ab6adf9f6cb7296193b5cc5cc359e
Page Source Download (Mobile Device)	File Hash:	MD5: a9b222bc82f7cb793bd05e5b99437cf8
	Stream Hash:	MD5: 256ab6adf9f6cb7296193b5cc5cc359e

Image #2		
Downloader	**Hash**	
4K Stogram (Desktop)	File Hash:	MD5: 6284813d69ab0c8189da676c05618760
	Stream Hash:	MD5: be36d618d038d9f2a55272f1c14b153b
Dinsta (Desktop)	File Hash:	MD5: a451f94331a03b8997c60960ea503162
	Stream Hash:	MD5: be36d618d038d9f2a55272f1c14b153b

Table 3: Image Retrieval Applications & Methods Results (Continued)

Image #2		
DownloadGram (Desktop)	File Hash:	MD5: a451f94331a03b8997c60960ea503162
	Stream Hash:	MD5: be36d618d038d9f2a55272f1c14b153b
Page Source Download (Desktop)	File Hash:	MD5: a451f94331a03b8997c60960ea503162
	Stream Hash:	MD5: be36d618d038d9f2a55272f1c14b153b
SaveGram (Mobile Device)	File Hash:	MD5: a451f94331a03b8997c60960ea503162
	Stream Hash:	MD5: be36d618d038d9f2a55272f1c14b153b
InstaSave (Mobile Device)	File Hash:	MD5: a451f94331a03b8997c60960ea503162
	Stream Hash:	MD5: be36d618d038d9f2a55272f1c14b153b
Page Source Download (Mobile Device)	File Hash:	MD5: a451f94331a03b8997c60960ea503162
	Stream Hash:	MD5: be36d618d038d9f2a55272f1c14b153b

Image #3		
Downloader	**Hash**	
4K Stogram (Desktop)	File Hash:	MD5: 49d63569bb16950e4f02f5718f10669b
	Stream Hash:	MD5: 257a6591f1bedf141ec58a80a391cca9
Dinsta (Desktop)	File Hash:	MD5: 5aa5a2629c1a4428eb91a469ffb9885f
	Stream Hash:	MD5: 257a6591f1bedf141ec58a80a391cca9
DownloadGram (Desktop)	File Hash:	MD5: 5aa5a2629c1a4428eb91a469ffb9885f
	Stream Hash:	MD5: 257a6591f1bedf141ec58a80a391cca9
Page Source Download (Desktop)	File Hash:	MD5: 5aa5a2629c1a4428eb91a469ffb9885f
	Stream Hash:	MD5: 257a6591f1bedf141ec58a80a391cca9
SaveGram (Mobile Device)	File Hash:	MD5: 5aa5a2629c1a4428eb91a469ffb9885f
	Stream Hash:	MD5: 257a6591f1bedf141ec58a80a391cca9

Table 3: Image Retrieval Applications & Methods Results (Continued)

Image #3		
InstaSave (Mobile Device)	File Hash:	MD5: 5aa5a2629c1a4428eb91a469ffb9885f
	Stream Hash:	MD5: 257a6591f1bedf141ec58a80a391cca9
Page Source Download (Mobile Device)	File Hash:	MD5: 5aa5a2629c1a4428eb91a469ffb9885f
	Stream Hash:	MD5: 257a6591f1bedf141ec58a80a391cca9

Image #4		
Downloader	**Hash**	
4K Stogram (Desktop)	File Hash:	MD5: bb7bb82823c48495268bd8f6f0350f5a
	Stream Hash:	MD5: a060b79c4b1db305ba857744b99699a2
Dinsta (Desktop)	File Hash:	MD5: ea863faf15baebbae0b617350c678eec
	Stream Hash:	MD5: a060b79c4b1db305ba857744b99699a2
DownloadGram (Desktop)	File Hash:	MD5: ea863faf15baebbae0b617350c678eec
	Stream Hash:	MD5: a060b79c4b1db305ba857744b99699a2
Page Source Download (Desktop)	File Hash:	MD5: ea863faf15baebbae0b617350c678eec
	Stream Hash:	MD5: a060b79c4b1db305ba857744b99699a2
SaveGram (Mobile Device)	File Hash:	MD5: ea863faf15baebbae0b617350c678eec
	Stream Hash:	MD5: a060b79c4b1db305ba857744b99699a2
InstaSave (Mobile Device)	File Hash:	MD5: ea863faf15baebbae0b617350c678eec
	Stream Hash:	MD5: a060b79c4b1db305ba857744b99699a2
Page Source Download (Mobile Device)	File Hash:	MD5: ea863faf15baebbae0b617350c678eec
	Stream Hash:	MD5: a060b79c4b1db305ba857744b99699a2

Table 3: Image Retrieval Applications & Methods Results (Continued)

Image #5		
Downloader	**Hash**	
4K Stogram (Desktop)	File Hash:	MD5: b6b79d477274a733700a490c539edf2d
	Stream Hash:	MD5: a71167f8f8320d1104524c5754f7c485
Dinsta (Desktop)	File Hash:	MD5: 74a4e6d901c8ddcf12569dc05df4b40e
	Stream Hash:	MD5: a71167f8f8320d1104524c5754f7c485
DownloadGram (Desktop)	File Hash:	MD5: 74a4e6d901c8ddcf12569dc05df4b40e
	Stream Hash:	MD5: a71167f8f8320d1104524c5754f7c485
Page Source Download (Desktop)	File Hash:	MD5: 74a4e6d901c8ddcf12569dc05df4b40e
	Stream Hash:	MD5: a71167f8f8320d1104524c5754f7c485
SaveGram (Mobile Device)	File Hash:	MD5: 74a4e6d901c8ddcf12569dc05df4b40e
	Stream Hash:	MD5: a71167f8f8320d1104524c5754f7c485
InstaSave (Mobile Device)	File Hash:	MD5: 74a4e6d901c8ddcf12569dc05df4b40e
	Stream Hash:	MD5: a71167f8f8320d1104524c5754f7c485
Page Source Download (Mobile Device)	File Hash:	MD5: 74a4e6d901c8ddcf12569dc05df4b40e
	Stream Hash:	MD5: a71167f8f8320d1104524c5754f7c485

While it was found that the six aforementioned methods returned identical results as each other, 4K Stogram was the only program that returned varying results. The differences that were seen between 4K Stogram and the other programs were found within the structure of the obtained files. Unlike the other six downloaders that were tested, 4K Stogram was the only program that downloaded the file in Baseline DCT encoded format, which contains restart markers (RST Markers), whereas all of the other processes downloaded the files in the Progressive DCT encoded format. In addition to this, 4K Stogram was the only program that retained the comments made on photos that were uploaded to Instagram.

This unique feature makes 4K Stogram the optimal choice for forensic analysis. It allows investigators the ability to connect the image directly with a users uploaded image by not only using a hash verification, but by also linking the user's self-populated comment to the image in question. While there was some initial concern about 4K Stogram being the only program to return different results from the rest of the programs/methods tested, it was determined that the only differences between the files downloaded from the other tested programs/methods and 4K Stogram were how the structures differed. These differences were found to be attributed to the additional data 4K Stogram retains and because it reorganized the data into Baseline DCT instead of Progressive DCT. To confirm this, a stream hash was performed on the recovered images from each application/method, which establishes the unique hash associated with solely the image contained in each file, rather than the file as a whole. From this, it was found that the data in each correlating image returned identical stream hashes, thus proving 4K Stogram to be the optimal forensic choice due to its broader scope of retained information while still obtaining an image with the identical structure and data as the rest of the tested methods.

It is also important to note that it was found that the other six downloading applications/methods all extract images in the same manner. The method that is likely used by all of these downloader's works by extracting the JPEG image file from the HTML data embedded in each web page viewed. While each application tested has a slightly different user interface, the coding of each of them utilizes the same HTML data to extract images from webpages. This is why every one of them produced bit stream identical results. Due to this, it is also likely that Instagram saves all of the images uploaded to it as Progressive DCT. Since this is how the file is found to be encoded when accessed directly through the pages HTML source code, it can be surmised that this is the most original state of an image as it would be found on Instagram's servers. However, since it has also been found that the Baseline DCT encoded images retrieved by 4K Stogram have bit stream duplicate stream hash values as the images taken directly from the page source code, it is found that the reorganization performed by 4K Stogram does not alter the forensic viability of the recovered images.

Structure Analysis

Upon the initial tests performed to study the structures of each test file downloaded from Instagram, it was found that each file had it's own semi-unique structure. While each of these structures was unique to their files, a pattern was recognized in the files as to how they were organized. The hypothesis made was that each structure was identical to each other in how all of the data was organized, and that the differences seen in each of the files were due to varying file sizes between images and the additional comment data that was added when the image was uploaded to Instagram. In order to test this hypothesis, the test images were re-uploaded to Instagram in the same fashion as originally done, though instead of labeling each image in the comments section of Instagram, four periods (….) were used. This then ensured that each image had exactly the same amount of data added to the resulting file from the comments notated on the image.

Once this new group of test images was downloaded and examined, it was seen that the structures of the resulting images were all identical. This was even the case for the images uploaded from the iPhone 6s, even though the data within the images were different from that of the their counterparts that were uploaded from the Samsung S7 and the Motorola Droid Turbo 2. As explained previously, the resulting differences in the images that were loaded on to the iPhone were presumed to have been caused by the iPhone’s operating system, which it is hypothesized altered the images in order to catalogue and organize the newly added files. It is important to note though, that while the photos were altered by the iPhone’s operating system, the resulting structure that is implemented by Instagram is identical across all test images used. While the test database of images for this research project was limited in comparison to similar studies of other social media platforms such as in article (11), it is hypothesized that Instagram structures all images in this way that are uploaded to its platform, though further research is certainly needed in order to help validate this hypothesis.

The data extracted from Instagram is listed in Tables 4 and 5 that follow. It should be noted that while it is not possible to identify each image as it relates to the original uploaded image by using its file hash, nor is it possible to identify which device uploaded each image with the exception of images from the iPhone, it is possible to identify each image with its original by using the stream hash, which can then be matched to previously downloaded iterations of the image that have been used in this research. Due to this, each image is labeled with both its file name as it was downloaded and a label denoting which image it originated from. While this is helpful in allowing images to be tracked through this process, it is limited in allowing us to know which phone the bit stream duplicate images came from since the data of each, including how they are labeled, is identical between both the Samsung S7 and the Motorola Droid Turbo 2 that were used. Therefore, all of these images are labeled under the umbrella of “Images Uploaded from Motorola Droid Turbo 2 & Samsung S7” in Table 5.

Table 4: Instagram File Structures

Image #1:	**Image #2:**	**Image #3:**
1632105546742413067_5921277047.jpg	1632105759368477992_5921277047.jpg	1632106299913480929_5921277047.jpg
Structure	**Structure**	**Structure**
Offset 0: M_ID SOIMarker	Offset 0: M_ID SOIMarker	Offset 0: M_ID SOIMarker
Offset 2: APP0	Offset 2: APP0	Offset 2: APP0
Offset 14: APP1	Offset 14: APP1	Offset 14: APP1
Offset E4: APP13	Offset E4: APP13	Offset E4: APP13
Offset 162: DQT dqt[0]	Offset 162: DQT dqt[0]	Offset 162: DQT dqt[0]
Offset 1A7: DQT dqt[1]	Offset 1A7: DQT dqt[1]	Offset 1A7: DQT dqt[1]
Offset 1EC: SOFx sof0	Offset 1EC: SOFx sof0	Offset 1EC: SOFx sof0
Offset 1FF: DHT dht[0]	Offset 1FF: DHT dht[0]	Offset 1FF: DHT dht[0]
Offset 220: DHT dht[1]	Offset 220: DHT dht[1]	Offset 220: DHT dht[1]
Offset 2D7: DHT dht[2]	Offset 2D7: DHT dht[2]	Offset 2D7: DHT dht[2]
Offset 2F8: DHT dht[3]	Offset 2F8: DHT dht[3]	Offset 2F8: DHT dht[3]
Offset 3AF: SOS scanStart	Offset 3AF: SOS scanStart	Offset 3AF: SOS scanStart
Offset 3BA: scanData[113432]	Offset 3BA: scanData[119138]	Offset 3BA: scanData[98270]
Offset 1BED2: M_ID EOIMarker	Offset 1D51C: M_ID EOIMarker	Offset 18398: M_ID EOIMarker

Table 4: Instagram File Structures (Continued)

Image #4: 1632106479454800751_5921277047.jpg	**Image #5:** 1632105927643819587_5921277047.jpg	**Image #6:** 1632106081650347358_5921277047.jpg
Structure Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[98080] Offset 182DA: M_ID EOIMarker	**Structure** Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[163952] Offset 2842A: M_ID EOIMarker	**Structure** Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[136285] Offset 21817: M_ID EOIMarker

Image #7: 1632111872818069831_5921277047.jpg	**Image #8:** 1632111704316194959_5921277047.jpg	**Image #9:** 1632115539042591548_5921277047.jpg
Structure Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[113432] Offset 1BED2: M_ID EOIMarker	**Structure** Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[119138] Offset 1D51C: M_ID EOIMarker	**Structure** Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[98270] Offset 18398: M_ID EOIMarker

Table 4: Instagram File Structures (Continued)

Image #10: 2017-10-2312.24.001632112899164368149_5921277047.jpg **Structure** Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[98080] Offset: 182DA M_ID EOIMarker	**Image #11:** 2017-10-2312.21.181632111538825817751_5921277047.jpg **Structure** Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[163952] Offset 2842A: M_ID EOIMarker	**Image #12:** 2017-10-23 12.20.551632111344352523105_5921277047.jpg **Structure** Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[136285] Offset 21817: M_ID EOIMarker

Image #13: 2017-09-25 17.16.181611966298601205104_5921277047.jpg **Structure** Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[93853] Offset 17257: M_ID EOIMarker	**Image #14:** 2017-09-25 17.15.591611966133546834221_5921277047.jpg **Structure** Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[98692] Offset 1853E: M_ID EOIMarker	**Image #15:** 2017-09-25 17.17.001611966643406420324_5921277047.jpg **Structure** Offset 0: M_ID SOIMarker Offset 2: APP0 Offset 14: APP1 Offset E4: APP13 Offset 162: DQT dqt[0] Offset 1A7: DQT dqt[1] Offset 1EC: SOFx sof0 Offset 1FF: DHT dht[0] Offset 220: DHT dht[1] Offset 2D7: DHT dht[2] Offset 2F8: DHT dht[3] Offset 3AF: SOS scanStart Offset 3BA: scanData[81581] Offset 14267: M_ID EOIMarker

Table 4: Instagram File Structures (Continued)

Image #16: 2017-09-25 17.17.22 1611966829633708294_5921277047.jpg	**Image #17:** 2017-09-25 17.14.40 1611965468749712354_5921277047.jpg	**Image #18:** 2017-09-25 17.15.20 1611965809226573581_5921277047.jpg
Structure	**Structure**	**Structure**
Offset 0: M_ID SOIMarker	Offset 0: M_ID SOIMarker	Offset 0: M_ID SOIMarker
Offset 2: APP0	Offset 2: APP0	Offset 2: APP0
Offset 14: APP1	Offset 14: APP1	Offset 14: APP1
Offset E4: APP13	Offset E4: APP13	Offset E4: APP13
Offset 162: DQT dqt[0]	Offset 162: DQT dqt[0]	Offset 162: DQT dqt[0]
Offset 1A7: DQT dqt[1]	Offset 1A7: DQT dqt[1]	Offset 1A7: DQT dqt[1]
Offset 1EC: SOFx sof0	Offset 1EC: SOFx	Offset 1EC: SOFx sof0
Offset 1FF: DHT dht[0]	Offset 1FF: DHT dht[0]	Offset 1FF: DHT dht[0]
Offset 220: DHT dht[1]	Offset 220: DHT dht[1]	Offset 220: DHT dht[1]
Offset 2D7: DHT dht[2]	Offset 2D7: DHT dht[2]	Offset 2D7: DHT dht[2]
Offset 2F8: DHT dht[3]	Offset 2F8: DHT dht[3]	Offset 2F8: DHT dht[3]
Offset 3AF: SOS scanStart	Offset 3AF: SOS scanStart	Offset 3AF: SOS scanStart
Offset 3BA: scanData[82049]	Offset 3BA: scanData[163509]	Offset 3BA: scanData[132091]
Offset 1443B: M_ID EOIMarker	Offset 2826F: M_ID EOIMarker	Offset 207B5: M_ID EOIMarker

Table 5: Structure Analysis File Info

Images Uploaded from Motorola Droid Turbo 2 & Samsung S7			
Image:	**Quality Factor:**	**MD5 Stream Hash**	**Correlating Image:**
Image #1	70	MD5=256ab6adf9f6cb7296193b5cc5cc359e	Motorola Image #1
Image #2	70	MD5=a71167f8f8320d1104524c5754f7c485	Motorola Image #5
Image #3	70	MD5=9ebd6f64d66410a727999503554adbe8	Samsung Image #1
Image #4	70	MD5=81fa86ce882f56cf0ab0ba469034c361	Samsung Image #2
Image #5	70	MD5=d16f77832a66183ee7de1fb2dc4ca5ea	iPhone Image #1
Image #6	70	MD5=2bb9cc87329d4e3ca977b632c68e3319	iPhone Image #2

Table 5: Structure Analysis File Info (Continued)

Images Uploaded from Motorola Droid Turbo 2 & Samsung S7			
Image:	**Quality Factor:**	**MD5 Stream Hash**	**Correlating Image:**
Image #7	70	MD5=256ab6adf9f6cb7296193b5cc5cc359e	Motorola Image #1
Image #8	70	MD5=a71167f8f8320d1104524c5754f7c485	Motorola Image #5
Image #9	70	MD5=9ebd6f64d66410a727999503554adbe8	Samsung Image #1
Image #10	70	MD5=81fa86ce882f56cf0ab0ba469034c361	Samsung Image #2
Image #11	70	MD5=d16f77832a66183ee7de1fb2dc4ca5ea	iPhone Image #1
Image #12	70	MD5=2bb9cc87329d4e3ca977b632c68e3319	iPhone Image #2
Images Uploaded from iPhone 6s			
Image:	**Quality Factor:**	**MD5 Stream Hash:**	**Correlating Image:**
Image #13	82	MD5=665e30103ded73abdaa2f736b28ac2ec	Motorola Image #1
Image #14	82	MD5=2e8df18489ac663f0d2de0537e921e83	Motorola Image #5
Image #15	82	MD5=595b0990396aad5ca435ef0a73940f45	Samsung Image #1
Image #16	82	MD5=1fde33b0583d87c383d6beb324c9666e	Samsung Image #2
Image #17	72	MD5=2e189d30474dee01b6a59a33de35e42e	iPhone Image #1
Image #18	72	MD5=66a58755098083a7061810226b2b486a	iPhone Image #2

Quantization Tables

After recovering all of the test images that were used in this study, it was found that there were 19 different quantization tables that were utilized by Instagram, and that every image was encoded with Baseline DCT entropy encoding. Of these, 17 were found to have been used on the data set of 84 images that was made up of images of varying dimensions. Of these 17 quantization tables, one of them was also the table found to have been used with the images that were uploaded from both the Motorola Droid Turbo 2 and the Samsung S7. The other two quantization tables that were observed to have been used by Instagram in this study were both used in handling the images that were uploaded from the iPhone 6s.

One specific thing of interest was that the images that were loaded onto the Droid Turbo 2 and the Samsung S7 from the iPhone remained in their original state and transferred as bit stream duplicates as the images that were uploaded from the iPhone. Instagram, however, handled these images differently in how they were compressed and which quantization tables were used. Though it was hypothesized that the same images on different devices should be handled the same way despite which devices uploaded them, this turned out not to be the case. The iPhone images that were uploaded from the Motorola and Samsung devices were recovered with a quality factor of 70.6 Luminance and 70.13 Chrominance, whereas the bit stream duplicate images uploaded from the iPhone were recovered with quality factors of 70.52 Luminance and 70.65 Chrominance. The other finding of interest from this data was that the iPhone images that were uploaded from the Motorola and Samsung devices were handled in an identical manner as the other images uploaded from both the android devices. While this could be seen as a product of Instagrams coding to reduce images larger than 1080x1080 down to the uniform size of 1080x1080 (though discrepancies to this theory have been found) for which the quantization table used would imply, it doesn't explain why the images uploaded from the iPhone were assigned a different quantization table which also compressed their size to 1080x1080. One hypothesis that would require further research to substantiate this is that the Instagram mobile app for iPhone is coded so that it recognizes images that are coming from an iOS device, and as such handles them differently and assigns unique quantization tables to images uploaded from such platforms. If this hypothesis can be substantiated, it would greatly strengthen the forensic implications of the manner in which Instagram handles images from android and iOS devices. This could allow investigators to quickly determine what style of device an image was uploaded from simply by examining the quantization table that was used. Another possibility though, which holds true based on the data collected in this study, as seen in Tables 8 and 9, is that Instagram has multiple quantization tables available for compressing and resizing an image to a specific dimension.

Below in Tables 6 and 7 are the quantization tables and estimated quality factors that were used on the images that were uploaded from the three mobile devices that were utilized in this study. Since multiple images were assigned the same quantization tables, each set of tables used is only listed once and is notated with the images that it is associated with, the device that uploaded said images, and the final size of the resulting images.

Table 6: Quantization Tables Used on Mobile Uploaded Images

Quantization Tables Used On Images Uploaded From Motorola Droid Turbo 2 & Samsung S7	
Luminance	Chrominance
DQT, Row #0: 10 7 6 10 14 24 31 37	DQT, Row #0: 10 11 14 28 59 59 59 59
DQT, Row #1: 7 7 8 11 16 35 36 33	DQT, Row #1: 11 13 16 40 59 59 59 59
DQT, Row #2: 8 8 10 14 24 34 41 34	DQT, Row #2: 14 16 34 59 59 59 59 59
DQT, Row #3: 8 10 13 17 31 52 48 37	DQT, Row #3: 28 40 59 59 59 59 59 59
DQT, Row #4: 11 13 22 34 41 65 62 46	DQT, Row #4: 59 59 59 59 59 59 59 59
DQT, Row #5: 14 21 33 38 49 62 68 55	DQT, Row #5: 59 59 59 59 59 59 59 59
DQT, Row #6: 29 38 47 52 62 73 72 61	DQT, Row #6: 59 59 59 59 59 59 59 59
DQT, Row #7: 43 55 57 59 67 60 62 59	DQT, Row #7: 59 59 59 59 59 59 59 59

Images:	Original Image Dimensions:	Resulting Image Dimensions:
Motorola Image #1	4008x5344	1080x1080
Motorola Image #5	4008x5344	1080x1080
Samsung Image #1	4032x3024	1080x1080
Samsung Image #2	4032x3024	1080x1080
iPhone Image #1	3264x2448	1080x1080
iPhone Image #2	2448x3264	1080x1080

Quantization Tables Used On Images Uploaded From iPhone 6s	
Luminance	Chrominance
DQT, Row #0: 5 5 5 8 11 15 19 23	DQT, Row #0: 6 6 9 16 38 38 38 38
DQT, Row #1: 5 5 5 8 11 15 19 23	DQT, Row #1: 6 9 9 26 38 38 38 38
DQT, Row #2: 5 5 8 11 15 19 23 28	DQT, Row #2: 9 9 21 38 38 38 38 38
DQT, Row #3: 8 8 11 15 19 23 28 32	DQT, Row #3: 16 26 38 38 38 38 38 38
DQT, Row #4: 11 11 15 19 23 28 32 36	DQT, Row #4: 38 38 38 38 38 38 38 38
DQT, Row #5: 15 15 19 23 28 32 36 36	DQT, Row #5: 38 38 38 38 38 38 38 38
DQT, Row #6: 19 19 23 28 32 36 36 36	DQT, Row #6: 38 38 38 38 38 38 38 38
DQT, Row #7: 23 23 28 32 36 36 36 36	DQT, Row #7: 38 38 38 38 38 38 38 38

Table 6: Quantization Tables Used on Mobile Uploaded Images (Continued)

<table>
<tr><th>Images:</th><th>Original Image Dimensions:</th><th>Resulting Image Dimensions:</th></tr>
<tr><td>Motorola Image #1</td><td>852x1136</td><td>852x852</td></tr>
<tr><td>Motorola Image #5</td><td>852x1136</td><td>852x852</td></tr>
<tr><td>Samsung Image #1</td><td>852x1136</td><td>852x852</td></tr>
<tr><td>Samsung Image #2</td><td>852x1136</td><td>852x852</td></tr>
<tr><td colspan="3"></td></tr>
<tr><td>(Luminance)
DQT, Row #0: 7 7 7 12 17 23 30 36
DQT, Row #1: 7 7 7 12 17 23 30 36
DQT, Row #2: 7 7 12 17 23 30 36 43
DQT, Row #3: 12 12 17 23 30 36 43 50
DQT, Row #4: 17 17 23 30 36 43 50 56
DQT, Row #5: 23 23 30 36 43 50 56 56
DQT, Row #6: 30 30 36 43 50 56 56 56
DQT, Row #7: 36 36 43 50 56 56 56 56</td><td colspan="2">(Chrominance)
DQT, Row #0: 9 9 14 25 59 59 59 59
DQT, Row #1: 9 13 13 40 59 59 59 59
DQT, Row #2: 14 13 33 59 59 59 59 59
DQT, Row #3: 25 40 59 59 59 59 59 59
DQT, Row #4: 59 59 59 59 59 59 59 59
DQT, Row #5: 59 59 59 59 59 59 59 59
DQT, Row #6: 59 59 59 59 59 59 59 59
DQT, Row #7: 59 59 59 59 59 59 59 59</td></tr>
<tr><th>Images:</th><th>Original Image Dimensions:</th><th>Resulting Image Dimensions:</th></tr>
<tr><td>iPhone Image #1</td><td>3264x2448</td><td>1080x1075</td></tr>
<tr><td>iPhone Image #2</td><td>2448x3264</td><td>1080x1080</td></tr>
</table>

Table 7: Quality Factors of Mobile Uploaded Images Recovered From Instagram

Quality Factor of Images Uploaded From Motorola Droid Turbo 2 & Samsung S7	
Images:	**Quality Factor**
Motorola Image #1	70
Motorola Image #5	70
Samsung Image #1	70
Samsung Image #2	70
iPhone Image #1	70
iPhone Image #2	70

Quality Factor of Images Uploaded From iPhone 6s	
Images:	**Quality Factor**
Motorola Image #1	82
Motorola Image #5	82
Samsung Image #1	82
Samsung Image #2	82
iPhone Image #1	72
iPhone Image #2	72

Instagram Resizing & Cropping Method

After uploading the dataset of 84 test images of varying dimensions, it was determined that the majority of the time Instagram will crop and resize images into square dimensions. While it was originally hypothesized that Instagram crops every single image into square dimensions, of the 84 test images used 83 came back square, while one came back with the dimensions of 720x900. This image was originally 512x640, so in the recompression process it was up-scaled, though the aspect ratio remained the same between the two iterations of the image. Since this was notated as a potential error, the test was redone with several of the apparent outliers detected from the original results. While there were some different results obtained in this test, the 512x640 image was returned with square dimensions as a 720x720.

It was also found that in the test of images uploaded from mobile devices, that one of the 18 images was also returned with non-square dimensions. This was "iPhone Image #1", uploaded from the iPhone 6s. While the original image was 3264x2448, it was returned as 1080x1075. Being seen as a possible error, all six of the images from the iPhone 6s were re-uploaded to Instagram. When the newly uploaded images were recovered though, it was found that "iPhone Image #1" was resized to the same dimensions of 1080x1075. While it is unknown why this image was handled differently than its bit stream duplicates uploaded from the Samsung S7 and the Motorola Droid Turbo 2, or even from the other iPhone originating image that it was uploaded in conjunction with, the results of this image were returned in identical formats with two consecutive tests, as seen in Tables 8 & 9.

Table 8: Mobile Upload Returned Image Dimensions

Motorola Droid Turbo 2			
Image	**Original Dimensions**	**Returned Dimensions**	**Quality Factor**
Motorola Image #1	4008x5344	1080x1080	70
Motorola Image #5	4008x5344	1080x1080	70
Samsung Image #1	4032x3024	1080x1080	70
Samsung Image #2	4032x3024	1080x1080	70
iPhone Image #1	3264x2448	1080x1080	70
iPhone Image #2	2448x3264	1080x1080	70
Samsung S7			
Image	**Original Dimensions**	**Returned Dimensions**	**Quality Factor**
Motorola Image #1	4008x5344	1080x1080	70
Motorola Image #5	4008x5344	1080x1080	70
Samsung Image #1	4032x3024	1080x1080	70
Samsung Image #2	4032x3024	1080x1080	70
iPhone Image #1	3264x2448	1080x1080	70
iPhone Image #2	2448x3264	1080x1080	70
iPhone 6s			
Image	**Original Dimensions**	**Returned Dimensions**	**Quality Factor**
Motorola Image #1	852x1136	852x852	82
Motorola Image #5	852x1136	852x852	82
Samsung Image #1	852x1136	852x852	82
Samsung Image #2	852x1136	852x852	82
iPhone Image #1	3264x2448	1080x1075	72
iPhone Image #2	2448x3264	1080x1080	72

Table 9: Mobile Upload #2 iPhones Returned Image Dimensions

iPhone 6s			
Image	**Original Dimensions**	**Returned Dimensions**	**Quality Factor**
Motorola Image #1	852x1136	852x852	82
Motorola Image #5	852x1136	852x852	82
Samsung Image #1	852x1136	852x852	82
Samsung Image #2	852x1136	852x852	82
iPhone Image #1	3264x2448	1080x1075	72
iPhone Image #2	2448x3264	1080x1080	72

In addition to Instagram cropping images into square dimensions, it was also found that Instagram will crop the longest dimension of an image, be it the x-axis or the y-axis, the majority of the time. The one discrepancy that was found in this study that contradicts the hypothesis that Instagram will behave this way 100% of the time though was with the test image 1280x800.jpeg. When this image was returned, it had the shorter dimension cropped, in this case the x-axis, and was resized to 1080x1080. With this being the only result with these findings, the original image was re-uploaded with the rest of the outlier data points that were found to determine if this image would be treated the same again. When the results came back with the second test though, it was seen that the image was resized to 800x800, and had only the y-axis cropped, which fit with the original hypothesis of how Instagram would determine which axis to crop in order to make a square image. It should also be taken into consideration though that for this study, no images with square dimensions were used, so it can not be ascertained through this study how Instagram would likely treat an image of such dimensions. As such, this is something that should be researched in future studies.

Another data point that was found that did not fit the hypothesized behavior of Instagram was the return of the test image 480x640.jpg, which came back recompressed to a 1080x1080 image. The question that arose from this is why did Instagram upscale this image and what made it determine to resize the image to 1080x1080, especially since this file was the smallest file dimensionally that was included in the test database. To try and test for some type of error that may have happened during the upload or download process, this image was re-uploaded to Instagram along with the other outliers mentioned above to verify the findings. Once those images were downloaded, it was found that the 480x640.jpg was handled in the same manner as it was the first time. However, as stated before, some of the other data that was re-run came back differently, which begs the question as to how Instagram determines how to process an image.

Generally though, images that are 1344x1792 or larger were seen to be resized down to 1080x1080. Of all of the results that were found in this study, no image returned by Instagram was larger than 1080x1080, so it is hypothesized that 1080x1080 is the largest image size that Instagram will resize a file to. It was also seen that the smallest images returned by Instagram were 480x480. While this was the smallest image size found in this study, the smallest file used in this study measured 480x640, so it is possible that smaller images could be returned in an even smaller scale after being compressed by Instagram. However, after notating that Instagram is capable of up-scaling images, it is hypothesized that 480x480 is the smallest image size that Instagram will return and that anything smaller than that will likely by up-scaled to either 480x480 or larger; though exactly how Instagram determines what size an image will be resized to is yet to be determined with the exception of images larger than 1344x1792.

These findings were a very important discovery due to them showing that while there are some trends that can be seen in how Instagram handles images in general, Instagram does not appear to have a predefined method for handling images with dimensions smaller than 1344x1792, or at least not one which was able to be determined in this study. In a similar study conducted on the recompression of images uploaded to Facebook, it was determined that images smaller than 1536x2048 were left at their original sizes, while images larger than 1536x2048 were resized to retain their same aspect ratio with the largest side measuring 2048 (9). Results such as these found with Facebook allow investigators to determine more about the original size of an image before it was uploaded to a social media site. However, the results that have been obtained thus far for Instagram have not shown this type of regularity or consistency, though it can be seen that images are typically resized to 1080x1080 and cropped into square dimensions as stated above.

The results of the main test, and the secondary test of results that were deemed to be outliers, can be seen in Tables 10 & 11, along with Figures 23-26 which display the dimensions returned during these tests. To aid in the visualization of the change in the dimensions of the images that were processed by Instagram, Figures 27 & 28 have also been added to show the dimensions of the images prior to being uploaded to Instagram. The quantization tables that are referenced in Table 10 & 11 can be found in Appendix A.

Table 10: Instagram Dimensional Resizing Output Data

Original Image Uploaded to Instagram (Width x Height)		Aspect Ratio	Image Downloaded from Instagram (Width x Height)		Quantization Table Used	Image Cropping Description
480	640	1.333	1080	1080	15	Top & Bottom cropped equally
512	640	1.250	720	900	6	Top & Bottom cropped equally
480	768	1.600	640	640	5	Top & Bottom cropped equally
480	800	1.667	544	544	3	Top & Bottom cropped equally
600	800	1.333	540	540	3	Top & Bottom cropped equally
624	832	1.333	480	480	1	Top & Bottom cropped equally
480	848	1.767	480	480	1	Top & Bottom cropped equally
480	854	1.779	624	624	5	Top & Bottom cropped equally
540	960	1.778	600	600	4	Top & Bottom cropped equally
544	960	1.765	480	480	1	Top & Bottom cropped equally
640	960	1.500	480	480	1	Top & Bottom cropped equally
720	960	1.333	512	512	2	Top & Bottom cropped equally
576	1024	1.778	768	768	7	Top & Bottom cropped equally
600	1024	1.707	640	640	5	Top & Bottom cropped equally
640	1024	1.600	600	600	4	Top & Bottom cropped equally
768	1024	1.333	576	576	4	Top & Bottom cropped equally
800	1024	1.280	1024	1024	13	Top & Bottom cropped equally
1024	1024	1.000	800	800	8	Top & Bottom cropped equally
1200	1080	0.900	800	800	8	Top & Bottom cropped equally
832	1120	1.346	768	768	7	Top & Bottom cropped equally
640	1136	1.775	720	720	6	Top & Bottom cropped equally
720	1152	1.600	1024	1024	13	Top & Bottom cropped equally
768	1152	1.500	900	900	11	Top & Bottom cropped equally
864	1152	1.333	864	864	10	Top & Bottom cropped equally
900	1152	1.280	768	768	7	Top & Bottom cropped equally

Table 10: Instagram Dimensional Resizing Output Data (Continued)

Original Image Uploaded to Instagram (Width x Height)		Aspect Ratio	Image Downloaded from Instagram (Width x Height)		Quantization Table Used	Image Cropping Description
720	1280	1.778	640	640	5	Top & Bottom cropped equally
768	1280	1.667	832	832	9	Top & Bottom cropped equally
800	1280	1.600	1080	1080	15	Sides cropped equally
854	1280	1.499	900	900	11	Top & Bottom cropped equally
960	1280	1.333	1080	1080	15	Top & Bottom cropped equally
1024	1280	1.250	720	720	6	Top & Bottom cropped equally
750	1334	1.779	1024	1024	13	Top & Bottom cropped equally
768	1366	1.779	1050	1050	14	Top & Bottom cropped equally
1050	1400	1.333	768	768	7	Top & Bottom cropped equally
900	1440	1.600	854	854	10	Top & Bottom cropped equally
960	1440	1.500	1000	1000	13	Top & Bottom cropped equally
1024	1440	1.406	750	750	7	Top & Bottom cropped equally
1080	1440	1.333	960	960	12	Top & Bottom cropped equally
768	1600	2.083	1080	1080	15	Top & Bottom cropped equally
900	1600	1.778	1080	1080	15	Top & Bottom cropped equally
1024	1600	1.563	1050	1050	14	Top & Bottom cropped equally
1200	1600	1.333	900	900	11	Top & Bottom cropped equally
1280	1600	1.250	768	768	7	Top & Bottom cropped equally
1050	1680	1.600	1024	1024	13	Top & Bottom cropped equally
1000	1776	1.776	960	960	12	Top & Bottom cropped equally
1344	1792	1.333	1080	1080	15	Top & Bottom cropped equally
1440	1800	1.250	1080	1080	15	Top & Bottom cropped equally
1392	1856	1.333	1080	1080	15	Top & Bottom cropped equally
1080	1920	1.778	1080	1080	15	Top & Bottom cropped equally
1200	1920	1.600	1080	1080	15	Top & Bottom cropped equally

Table 10: Instagram Dimensional Resizing Output Data (Continued)

Original Image Uploaded to Instagram (Width x Height)		Aspect Ratio	Image Downloaded from Instagram (Width x Height)		Quantization Table Used	Image Cropping Description
1280	1920	1.500	1080	1080	15	Top & Bottom cropped equally
1400	1920	1.371	1080	1080	15	Top & Bottom cropped equally
1440	1920	1.333	1080	1080	15	Top & Bottom cropped equally
1152	2048	1.778	1080	1080	15	Top & Bottom cropped equally
1280	2048	1.600	1080	1080	15	Top & Bottom cropped equally
1536	2048	1.333	1080	1080	15	Top & Bottom cropped equally
1440	2160	1.500	1080	1080	15	Top & Bottom cropped equally
1440	2304	1.600	1080	1080	15	Top & Bottom cropped equally
1728	2304	1.333	1080	1080	15	Top & Bottom cropped equally
1080	2538	2.350	1080	1080	15	Top & Bottom cropped equally
1080	2560	2.370	1080	1080	15	Top & Bottom cropped equally
1440	2560	1.778	1080	1080	15	Top & Bottom cropped equally
1600	2560	1.600	1080	1080	15	Top & Bottom cropped equally
1700	2560	1.506	1080	1080	15	Top & Bottom cropped equally
1800	2560	1.422	1080	1080	15	Top & Bottom cropped equally
1920	2560	1.333	1080	1080	15	Top & Bottom cropped equally
2048	2560	1.250	1080	1080	15	Top & Bottom cropped equally
2048	2732	1.334	1080	1080	15	Top & Bottom cropped equally
1824	2736	1.500	1080	1080	15	Top & Bottom cropped equally
2100	2800	1.333	1080	1080	15	Top & Bottom cropped equally
900	2880	3.200	1080	1080	15	Top & Bottom cropped equally
1800	2880	1.600	1080	1080	15	Top & Bottom cropped equally
2000	3000	1.500	1080	1080	15	Top & Bottom cropped equally
1800	3200	1.778	1080	1080	15	Top & Bottom cropped equally
2048	3200	1.563	900	900	11	Top & Bottom cropped equally

Table 10: Instagram Dimensional Resizing Output Data (Continued)

Original Image Uploaded to Instagram (Width x Height)		Aspect Ratio	Image Downloaded from Instagram (Width x Height)		Quantization Table Used	Image Cropping Description
2400	3200	1.333	1080	1080	15	Top & Bottom cropped equally
1440	3440	2.389	1080	1080	15	Top & Bottom cropped equally
2160	3840	1.778	1080	1080	15	Top & Bottom cropped equally
2400	3840	1.600	1080	1080	15	Top & Bottom cropped equally
2304	4096	1.778	1080	1080	15	Top & Bottom cropped equally
3072	4096	1.333	480	480	1	Top & Bottom cropped equally
2160	5120	2.370	1080	1080	15	Top & Bottom cropped equally
2880	5120	1.778	1080	1080	15	Top & Bottom cropped equally
3200	5120	1.600	1080	1080	15	Top & Bottom cropped equally

Table 11: Instagram Dimension Resizing Output Data – Outlier Test #2

Original Image Uploaded to Instagram (Width x Height)		Aspec t Ratio	Image Downloaded from Instagram (Width x Height)		Quantization Table Used	Image Cropping Description
480	640	1.333	1080	1080	15	Top & Bottom cropped equally
512	640	1.250	720	720	6	Top & Bottom cropped equally
768	1024	1.333	576	576	4	Top & Bottom cropped equally
1024	1024	1.000	800	800	8	Top & Bottom cropped equally
800	1280	1.600	800	800	16	Top & Bottom cropped equally
1024	1280	1.250	720	720	6	Top & Bottom cropped equally
1200	1600	1.333	900	900	11	Top & Bottom cropped equally
2048	3200	1.563	1080	1080	17	Top & Bottom cropped equally
3072	4096	1.333	480	480	1	Top & Bottom cropped equally

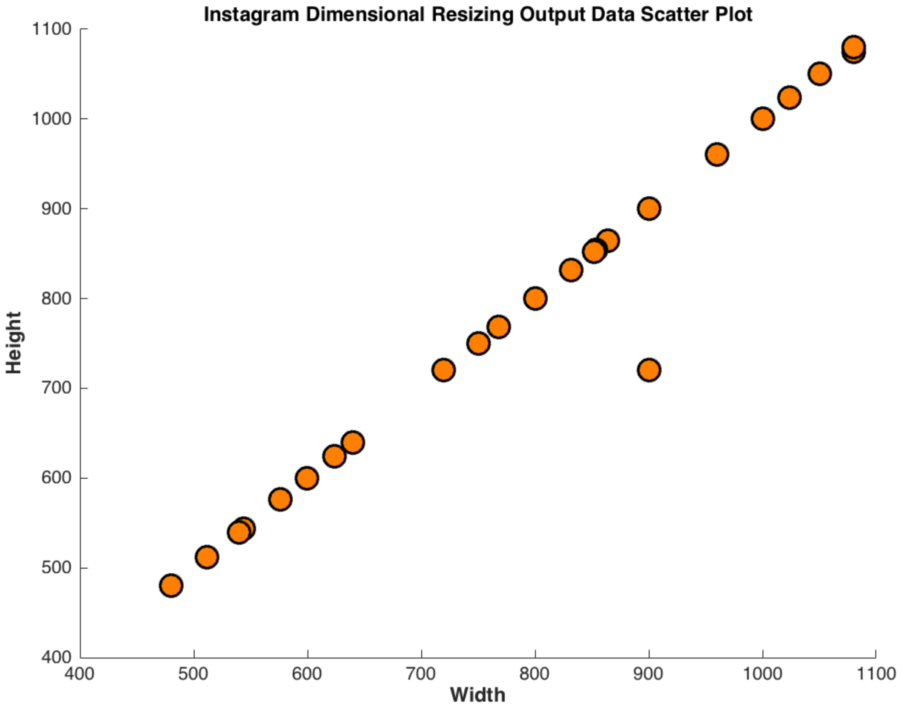


Figure 23: Instagram Dimensional Resizing Output Data Scatter Plot

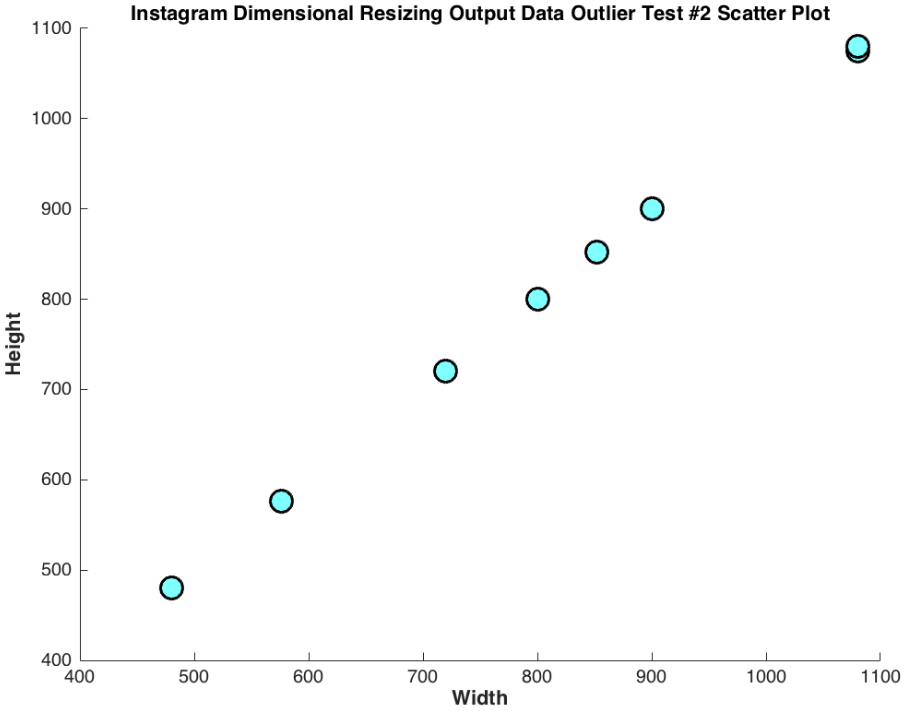


Figure 24: Instagram Dimensional Resizing Output Data Outlier Test #2 Scatter Plot

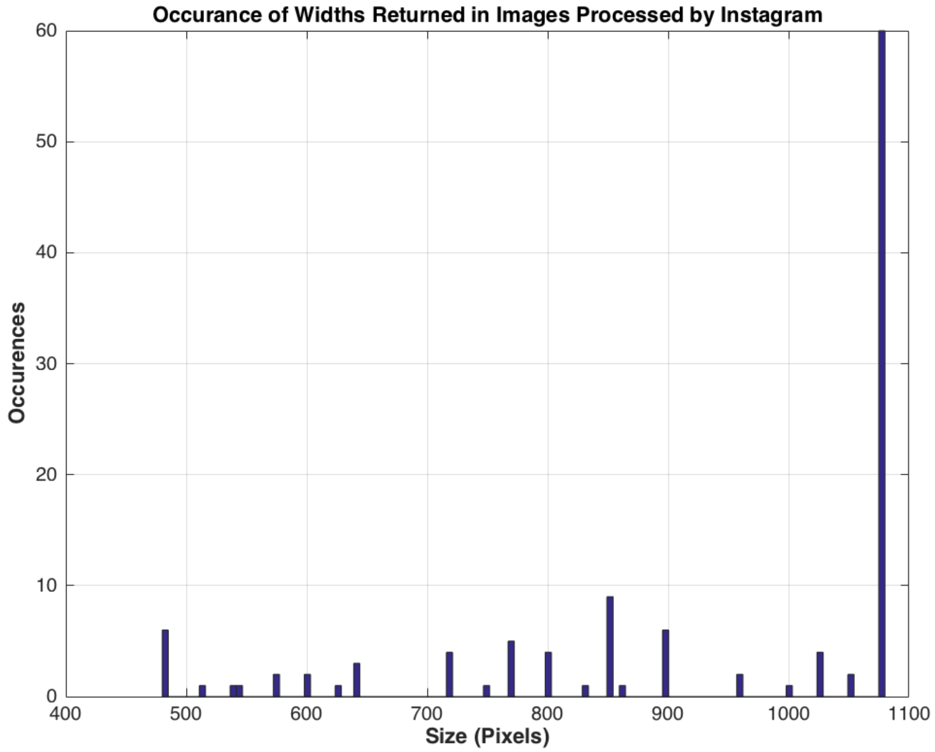


Figure 25: Occurrence of Widths Returned in Images Processed by Instagram

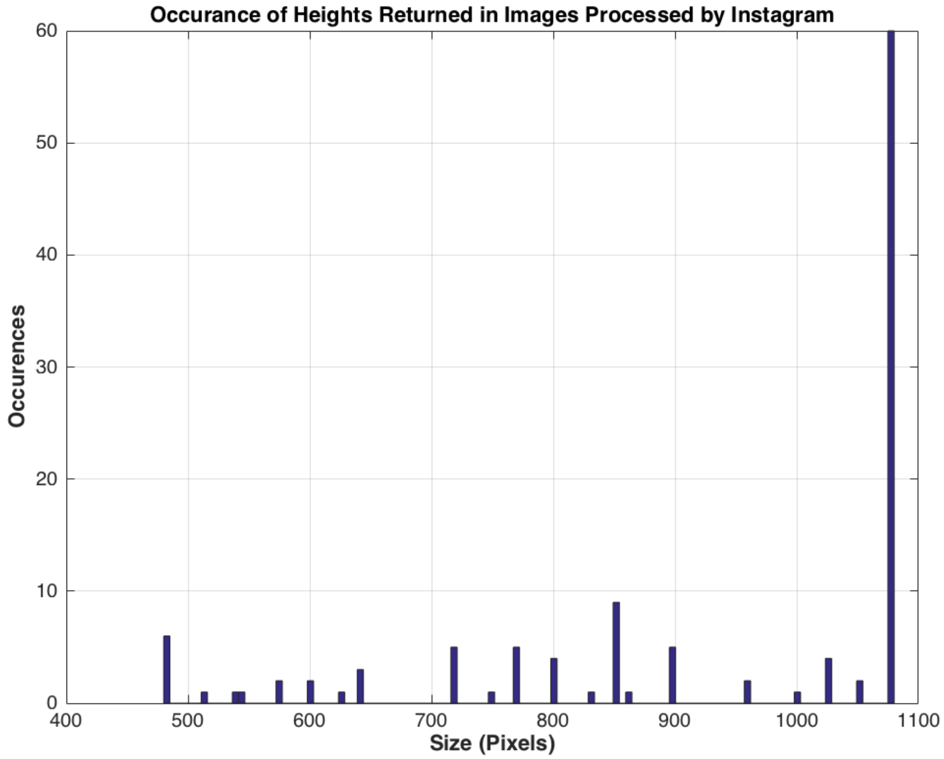


Figure 26: Occurrence of Heights Returned in Images Processed by Instagram

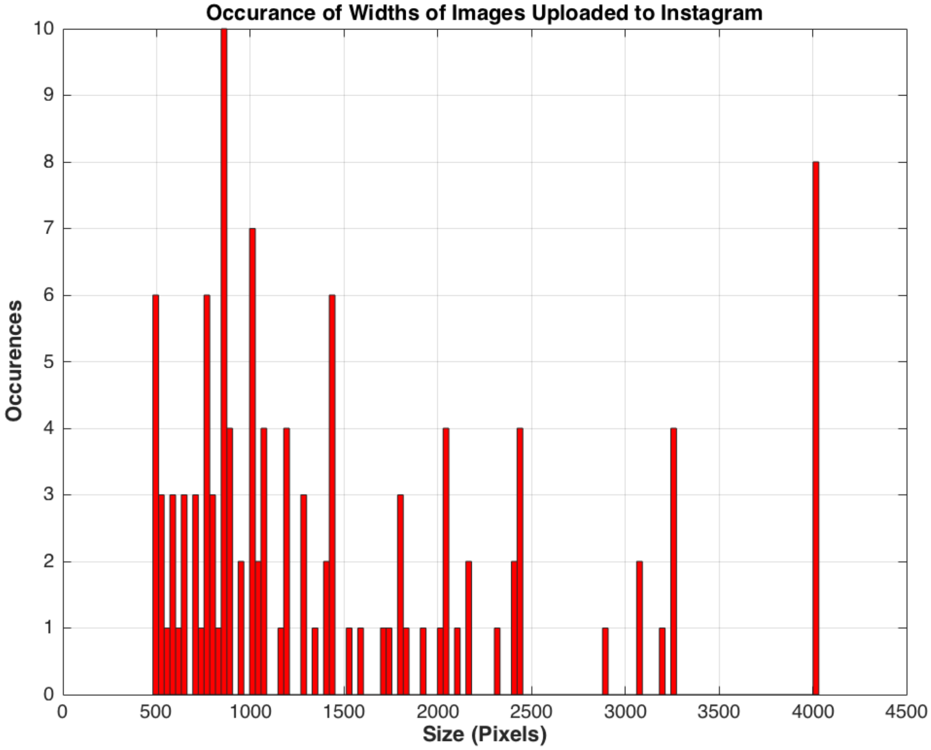
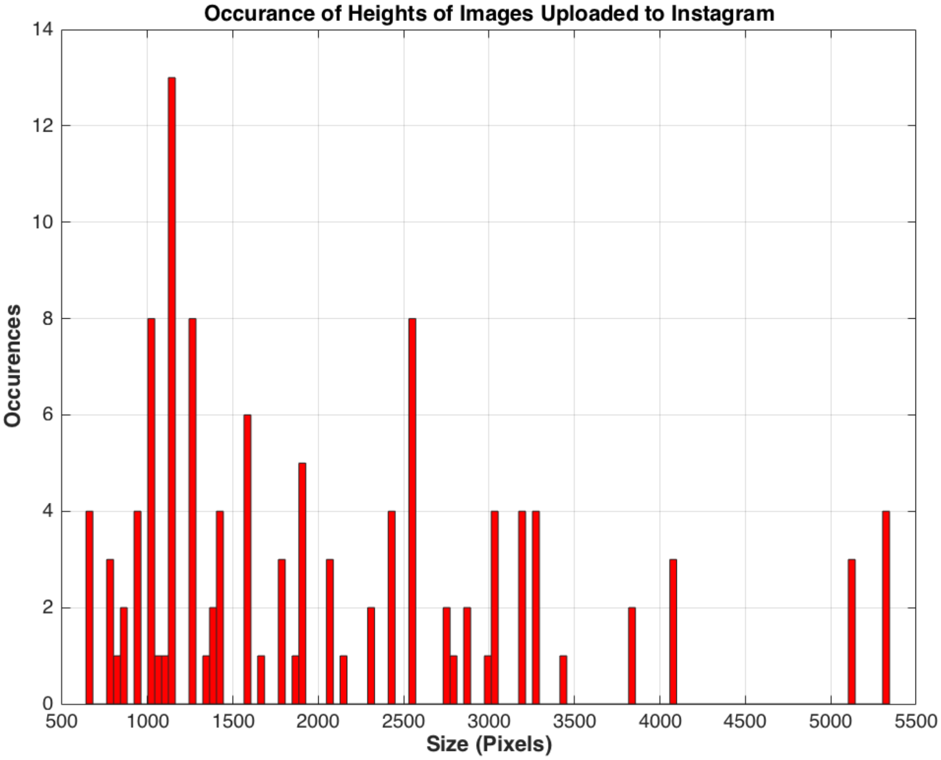


Figure 27: Occurrence of Widths of Images Uploaded to Instagram



28: Occurrence of Heights of Images Uploaded to Instagram

In addition to finding that Instagram crops and resizes images, it was also observed that in some instances images were stretched as well. While this was not observed in all of the images that were used in this study, these findings do indicate that in some situations Instagram will stretch an image in conjunction with resizing and cropping the it. It is hypothesized that this is done to try and correct images that appear to have been warped, though exactly when and how Instagram does this has yet to be determined. An example of this can be seen in Figures 29 & 30, which shows the before and after effects of this observed stretching on the image 800x1280.jpg.

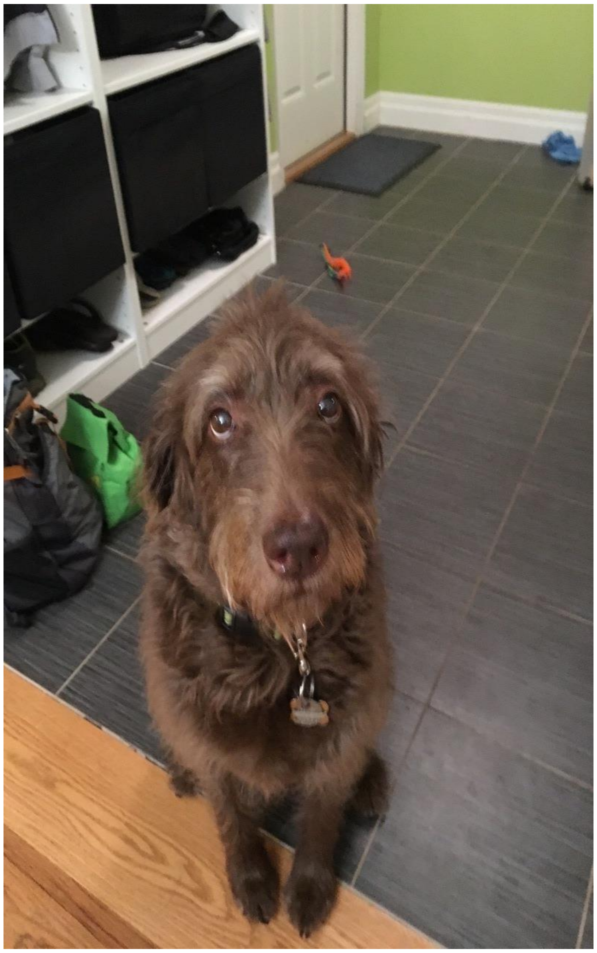


Figure 29: 800x1280.jpg (Original)

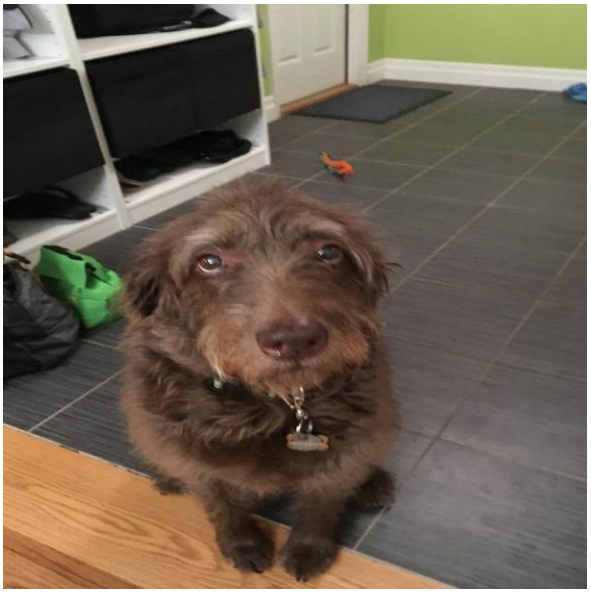


Figure 30: 800x1280.jpg
(Recompressed & Stretched by Instagram)

Established Trends

Trends that have been found include several distinct features that help to identify that an image has been retrieved from Instagram. While there are some inconsistencies in the data that was collected in this study that make it difficult to establish clear patterns in how Instagram handles images of varying sizes, dimensions, and origins, enough data was collected to provide an overview of some of the major trends that can be seen in Instagram.

The first trend that was seen in Instagram was that every image that was returned was returned in a JPEG format. This is very helpful because not only does it narrow down the container that can be expected to be used for an image that was processed through Instagram, but it also gives investigators knowledge of how that container should be organized. By having a basic understanding of the JPEG file format, as explained in Chapter 2, it is possible to know what limitations are inherent in the JPEG file format specifications and what to expect a resulting JPEG encoded file to look like.

In conjunction with this, the most important discovery of this study was found in how the JPEG images that were processed by Instagram are organized. Overall, all JPEG images abide by the same basic structure guidelines as to how the data within each image needs to be organized so that it can be read by various file systems. Through this study though, it was revealed that all images collected from Instagram were returned with identical structures to each other, even when using different software to download images. While the images used in this study consisted of varying dimensions and file sizes, the resulting images all shared identical file structures with the only differences in each file being how much data each image consisted of. It is also important to note that the structures seen in this study that were identical to one another were all extracted by using images that were uploaded with the exact same notations made in the comments section for each image when it was uploaded to Instagram. Therefore, when examining an image to identify if it has a structure of a known origin, an image from Instagram may have a slightly different structure than what is notated in this study simply due to the length of the user-created comments that have been added. If one of the other methods or applications that was addressed in this study is used to extract an image from Instagram though, the structures of any image retrieved in such a fashion will all have identical structures since there will be no added comment data, which is the only reason the structures can vary.

In regards to the dimensions of an image taken from Instagram, it was also found that the majority of the time images would be returned in square dimensions. While it was hypothesized that images would always be returned in square dimensions due to preliminary observations, three images that were returned during this study were found to have rectangular dimensions, as can be seen in Tables 8, 9, 10, and Figures 23 & 24. While this represents only 2.22% of all 135 images that were uploaded and downloaded throughout this study, these irregularities suggest that for an as yet unknown reason images will not necessarily be retrieved with square dimensions 100% of the time. It is important to note though that when one of the images in question (512x640.jpg) was re-uploaded to Instagram and downloaded again with a handful of other images that were returned with suspect data, it was returned with square dimensions, as can be seen in Table 11 and Figure 24. This implies that an error resulted in the return of this image that did not fit the trend found for how Instagram resizes images, but this data can not be ignored and must be noted when examining images that are indicative of having been retrieved from Instagram even if they do not have square dimensions.

Another trend is that when Instagram crops images to square dimensions, it typically does so by cropping the larger axis, while leaving the shorter axis un-cropped. When resizing/cropping an image to 1080x1080, this process is typically done by resizing the image so that the shorter dimension is made equal to 1080, at which point the resulting longer dimension is cropped to equal 1080. This is done by taking the dimensions value, subtracting 1080, and dividing the resulting value by two. This value will then be subtracted from both sides of the longer dimension, which will result in an image that has been resized/cropped to 1080x1080. By cropping images in this manner, Instagram is able to retain the maximum amount of content in each image while still being able to resize them to its desired dimensions. While the majority of the images used in this study where this trend was seen were portrait-oriented images, it is hypothesized that this trend will hold true for landscape-oriented images as well, as was the case in the small number of landscape-oriented images that were included in this study.

There was also one discrepancy found with the theory as to which axis of the image will be cropped. While 134 of the 135 images that were used for this study were consistent with the trend of only the longest side of the image being cropped in order to create an image with square dimensions, there was one image in which the shorter dimension was cropped. When this image was re-uploaded to Instagram though, this image was returned with it's longest dimension cropped while the other sides were left untouched, along with it having been resized to different dimensions. While the discrepancy with the originally uploaded image could be seen as the result of an image with unusual dimensions, the images with widths closest to it both only had the top and bottom of the image cropped, and in addition to this when it was rerun the same image was returned in an expected format. The image in question was 800x1280.jpg, and the images that were not cropped in such a manner but were very close in their dimensions were 768x1280.jpg and 854x1280.jpg. While this could simply be something to do with the image in questions unique dimensions, it is something that should be researched further as the trend that is seen for how Instagram crops images does not hold true 100% of the time. The reasoning for this should be determined so that it can be known when and how Instagram will crop the shorter dimension of an image. For forensic analysis, this is a necessity because it must be determined if any pertinent information may have been cropped from an image after being processed by Instagram, and knowing which dimension was cropped is essential in determining this.

Along with how images are cropped, it was noted that when images of varying sizes are uploaded, that there were some common trends with how the images were compressed and resized. The most notable of these trends is that images uploaded to Instagram that are larger than 1344x1792, tend to be recompressed to 1080x1080. Of the 57 images that were uploaded to Instagram that were originally 1344x1792 or larger, 52 were returned as 1080x1080, with another two being returned as 1080x1075. For this small pool of tested images, this resulted in 91.23% of images sized 1344x1792 or larger being returned as 1080x1080 (94.74% if the two 1080x1075 images are included), which can be seen in Figures 25 and 26. While the small size of this data pool does not allow for as broad of an overview of Instagram's behavior as a whole, it does show a strong trend that is hypothesized to hold true in future studies that images 1344x1792 or larger will be resized to 1080x1080.

CHAPTER VI

FURTHER RESEARCH

Image Types

While it was found that only JPEG images were returned from Instagram, every image that was uploaded to Instagram in this study was a JPEG file to begin with. Due to this, further research needs to be conducted with a wide variety of image formats in order to verify that those images too are recompressed to JPEG file specifications. It is hypothesized that the results will show that JPEG images are the only file types returned by Instagram, but this was outside the purview of this study and as such was not looked into.

Structure

Though all of the structures returned from Instagram in regards to this study were identical, this study was limited in the size of the data pool that was used. Due to this a larger study needs to be performed on Instagram purely in regards to structure analysis. This should focus on finding if the structure found in this study can be attributed to all images uploaded to Instagram from any type of mobile device and image type, or whether there are instances when a different structure could be returned that was simply not addressed in this study.

In addition to this, it must also be studied as to exactly how a file's structure will differ depending on how long of a comment a user leaves on an image or if they use special characters. While the structures all returned in this study were the same, they were only identical when taken from images that were uploaded with identical comments made on each image. In order to make this data as useful and transparent as possible for examiners who are trying to study exactly what to look for in an image taken from Instagram, a table should be created showing which markers will be affected by comment data. Also, the table should note how far apart the other markers should be spaced depending on the content entered in the comments on an image. Further, it is important to note that the comment data will only affect the structures of returned images when they are retrieved using 4K Stogram, since it is the only known application that retrieves the comment data in addition to the image. Due to this, it would also be pertinent to perform a structure analysis study using a different method other than 4K Stogram. This analysis could then identify what the structure of images will be returned as in every other instance since all other extraction methods found in this study produce bit stream identical results.

Image Dimensions

Although it was hypothesized in this study, and as a trend overall with Instagram, that all images would be returned with square dimensions, three images were not. Whether this was due to a computing error or some coding in Instagram that makes this a possibility in certain circumstances, these irregularities need to be further researched to determine how and why they happened. Some of the questions that should be addressed in regards to how Instagram determines the dimensions of images are as follows:

- Is Instagram able and designed to return images in oblong dimensions?

  If so:

  - Why are more non-square images not returned from Instagram?
  - What paradigms does Instagram follow when deciding whether to resize an image into square or oblong dimensions?
  - Are the results found repeatable and reproducible?

  If not:

    - Can evidence of errors be found in the uploading or downloading process that would result in an image being returned in an oblong format?
    - Can evidence be found of an error within Instagram that causes the program to return such results periodically?

Along with these questions, it should also be noted that very few images were used in this study that had larger widths than heights. Due to this, it would advisable to conduct a study in which the majority of images used are configured in a landscape format in order to see if the trends found in this study still hold true.

Cropping

While most images in this study were returned with the sides of the longest axis cropped equally, one image was returned with the shorter axis cropped instead. While this does not fit the paradigm that has been seen that only the longest dimension will be cropped, it should be further researched to see if this is something that Instagram does with images of certain dimensions, or if this was an error. Following are some questions that should be addressed in regards to how Instagram crops images.

- ▪ Can Instagram intentionally crop the shorter dimension of an image? If so:
    - In what circumstance does Instagram crop the shorter dimension of an image?
    - Are the results found to be repeatable and reproducible across a large data pool of images with the same characteristics?

- Why did this happen on only one of the images that was used in this study?
    - In addition to this, why was the same image cropped differently when the test was repeated?

If not:

- Were the results found in this study potentially the result of an error caused by the uploading or downloading process?
- Were the results found in this study potentially an error caused by Instagram?

It should also be noted that since the majority of the images used in this study were in a portrait format, it needs to be studied whether the hypothesis that Instagram crops the longer of the two dimensions hold true for landscape images in a study with a larger database of images that are formatted in this orientation.

Stretching

Though stretching was not a common trend seen throughout this study, in some of the more heavily warped images that were resized for the Resizing and Cropping Test, this was found. Due to this, it must be further investigated to try and ascertain when Instagram will stretch an image and what regularities can be found with this procedure. To do this, it is recommended that a test be conducted which utilizes a large pool of resized images that have been resized to odd dimensions which alter the appearance of the image compared to the original. New images should then be created with the same dimensions and aspect ratios as the test images that were resized, which would be comprised of a white background and a precise grid of squares. By doing this, once all of the images are uploaded and downloaded after being recompressed, the grids that match up with each image can then measured to determine on the pixel level, how an image has been stretched. This could also be done by simple overlaying a precise grid on an image, but the grid must be fully distinguishable from the background in order to facilitate precise measurements of any stretching detected.

Instagram Settings

Although the most common upload settings were used in this study, there are other setting options available for uploading images on Instagram. Due to this, further research needs to be conducted utilizing these alternate settings to determine what difference they make in the resulting image after being processed by Instagram. The settings that need to be tested include the following:

Upload Quality: Basic

- Cellular Data Use: Use Less data
- Uploading Multiple Images at Once
- Uploading Images in Their Original Aspect Ratio
- Lux Adjustments
- Use of Filters
- Under "Edit" Option
    - Adjust
    - Brightness
    - Contrast
    - Structure
    - Tilt Shift
    - Warmth
    - Saturation
    - Color
    - Fade
    - Highlights
    - Shadows
    - Vignette
    - Sharpen

Along with this, it has also been observed that some images found on Instagram are not in square dimensions. While the conclusion of this thesis has been that Instagram crops images into square dimensions the majority of the time, this is only the case when uploading an image to Instagram in the "Standard" fashion. There is also the option to upload an image with its original aspect ratio by pinching the picture on the original image selection screen, which allows a user to upload an image without it being cropped into square dimensions. Since this is a feature that was not commonly observed though, this option was not tested in this study. Due to this, further research is needed to determine how images uploaded with their original aspect ratio are treated in comparison to the results obtained in this study.

Instagram's Compression Algorithms

Though this study found some clear trends in how Instagram recompresses images, like all digital platforms updates are made to the way that software functions. Due to this, future studies need to be done in the same manner as this study was conducted in order to ascertain if, or more likely when, Instagram changes how it processes images that are loaded onto its platform. Due to the ever-evolving nature of technology, it cannot be expected that the results found in this study will dictate how Instagram will function for the rest of its days. Instead, regular tests should be conducted on Instagram to try and ascertain when changes have occurred and to keep this data as up to date and current as possible. While it will not always be possible to know exactly when an update has been made to Instagram, regular and consistent testing of the platform should be able to keep close track of how the platform is functioning. In addition to this, it may be possible to determine how frequently Instagram updates its recompression methodologies, which would allow a schedule for reoccurring tests to be established in order to keep data on the social media platform as relevant as possible.

CHAPTER VII

CONCLUSION

After reviewing all of the data that was obtained throughout this study, it has been found that there are some common trends and tendencies that can be expected to be found when looking at images that have been recovered from Instagram. Of these, the most notable finding is that every image recovered from Instagram comes back with the same structure. Forensically, this is an extremely valuable finding due to the ability to determine whether an image has been recovered from Instagram, or if it is an original and authentic image. While if an image is found to have been recovered from Instagram, it is not possible to determine provenance of the image from before Instagram recompressed it, this structure analysis does at least allow investigators to know that the image they are dealing with is not the original image.

It was also found that while there is a multitude of ways to extract images from Instagram, the application 4K Stogram provides the most forensically viable data with images being recovered. The data that is extracted with this application that is not obtainable with any of the other methods tested is the recovered comment text data that is entered by the profiles user/uploader. With this additional data, investigators can show a clear link to the originating image on Instagram not only through a hash verification, but also through the connected user comment data that is then embedded in the recovered image. This is also extremely useful due to the volatility of images uploaded by users, due to the users ability to edit or delete comments from images, or the images in their entirety, which would leave investigators without definitive evidence showing the text originally associated with an image. Another aspect of 4K Stogram that was found to be extremely advantageous for the use in forensic investigations is the ability to very quickly extract every image from a users account and have them saved in a specified file with one step, whereas other methods require downloading images individually.

When looking at the trends found with image dimensions that are returned, it was seen that the majority of images that are originally larger than 1344x1792 were returned as 1080x1080. In conjunction with this, it was found that images were returned in square formats over 97% of the time. While one of the instances in which an image was returned in a non-square format was hypothesized to be an error of unknown origin since it was returned in a square format upon a subsequent test with the same image, it must still be noted that there can be instances in which images will be returned from Instagram with non-square dimensions. Along with this, it was also observed that when Instagram crops images, it generally only crops the longer axis of the image. While there was one instance found in which an image had its shorter axis cropped, this is a trend that appears to be fairly consistent in images handled by Instagram, especially since when the image in question was retested it was returned with the longer axis cropped. In addition, it was also observed that there is the ability to upload images to Instagram with their original aspect ratios, rather than defaulting to having the images cropped into square dimensions. While this feature was not utilized or examined during this study, it should be kept in mind by investigators that the dimensions of an image alone cannot prove or disprove that an image was recovered from Instagram.

In addition to the trends that were found with how images are handled when recompressed by Instagram, almost equally important were the findings in this study of the inconsistencies that were seen. While these inconsistencies do not necessarily help investigators in determining whether an image was recovered from Instagram, they do give some clear questions for future research into how Instagram processes images.

Due to this, it is vitally important that further research be conducted on the behaviors of Instagram in order to further validate the findings of this study, answer the questions that have arisen, and to try and establish as clear and concise guidelines as possible for investigators looking into Instagram. While this study did find some very useful data for forensic analysis of images that are known to have come from Instagram, or are suspected to have, there were many questions uncovered in the process of this study that need to be looked into further in order to obtain a clearer picture of how Instagram behaves in specific situations.

BIBLIOGRAPHY

1. Jones, P. *Digital Evidence/Photography and Digital Imaging - Digital Imaging and Photography: An Overview*. Encyclopedia of Forensic Sciences, Second Edition. Purdue University, West Lafayette, IN, USA. 2013.

2. Daniel, M. *Daguerre (1787–1851) and the Invention of Photography*. Department of Photographs, The Metropolitan Museum of Art. October 2004.

3. Snavely, N. *CS6670: Computer Vision*.

4. Ledesma, S. *A Proposed Framework for Image Enhancement*. University of Colorado. 2015

5. Kashyap, N., Modi, J. *JPEG Image Code Format*. 140.429

6. Ebrahimi, T. *JPEG at 25: Still Going Strong*. IEEE Computer Society. April-June 2017. Industry and Standards.

7. Hamilton, E. *JPEG File Interchange Format, Version 1.02*. C-Cube Microsystems. September 1, 1992.

8. *Recommendation ITU-R BT.601-5, Section 11B: Digital Television*. 1995

9. Pippin, M. *Digital Image Recompression Analysis: Facebook*. University of Colorado Denver. 2016.

10. Raid, A., Khedr, W., El-dosuky, M., Ahmed, W. *Jpeg Compression Using Discrete Cosine Transform – A Survey*. Mansoura University, Faculty of Computer Science & Information. Zagazig University, Faculty of Science. International Journal of Computer Science & Engineering Survey. Volume 5, Number 2. April 2014.

11. Amerini, I., Becarelli, R., Caldelli, R. *Image Origin Classification Based on Social Network Provenance*. IEEE Transactions on Information Forensics and Security. Volume 12, Number 6. June 2017.

12. *Number of Smartphone Users in the U.S. 2010-2022*. Statista DMO. 2017. https://www.statista.com/statistics/201182/forecast-of-smartphone-users-in-the-us/

13. Masoner, L. *Camera Body*. The Spruce. 2/18/2017. https://www.thespruce.com/what-to-know-about-camera-body-2688630

14. Schiesser, T. *Know Your Smartphone: A Guide To Camera Hardware*. TechSpot. July 28, 2014. https://www.techspot.com/guides/850-smartphone-camera-hardware/ . Jan 24, 2017.

15. *Photography Cheat Sheets*. Delview Secondary School. Media Arts Department. Jan 25, 2018. https://delviewmedia.weebly.com/photography-cheat-sheets.html

16. *Everything You Need But Don't Want to Know About Digital Imaging.* XInstruments. 2018. http://x-instruments.com/knowledge-base/digital-imaging-glossary.html

17. Guttosch, R. *Investigation of Color Aliasing of High Spatial Frequencies and Edges for Bayer-Pattern Sensors and Foveon X3 Direct Image Sensors*. Foveon, Inc.

18. Selek, M. *A New Autofocusing Method Based on Brightness and Contrast for Color Cameras*. Advances in Electrical and Computer Engineering. January 2016.

19. Schleif, R. *Sensing Violet: The Human Eye and Digital Cameras.* Johns Hopkins Krieger School of Arts & Sciences. Department of Biology. Accessed 1/15/18. http://pages.jh.edu/~rschlei1/Photographic/violet/violet.html

20. Kähler, C.,Hain, R., Tropea, C. *Comparison of CCD, CMOS and Intensified Cameras.* Exp Fluids. 42:403–411. January 13, 2007.

21. Exiftool. Version 5.18.2. 7/28/2016. https://sourceforge.net/projects/exiftool/

22. JPEGsnoop. Version 1.7.5. 2015. https://www.impulseadventure.com/photo/jpeg-snoop.html?ver=1.7.5

23. ExactFile. Version 1.0.0.15 BETA. 2009. http://www.exactfile.com/

APPENDIX

A:

1	**Luminance**	**Chrominance**
(480x480)		
	DQT, Row #0: 4 3 3 4 6 10 13 16	DQT, Row #0: 4 5 6 12 26 26 26 26
	DQT, Row #1: 3 3 4 5 7 15 16 14	DQT, Row #1: 5 5 7 17 26 26 26 26
	DQT, Row #2: 4 3 4 6 10 15 18 15	DQT, Row #2: 6 7 15 26 26 26 26 26
	DQT, Row #3: 4 4 6 8 13 23 21 16	DQT, Row #3: 12 17 26 26 26 26 26 26
	DQT, Row #4: 5 6 10 15 18 28 27 20	DQT, Row #4: 26 26 26 26 26 26 26 26
	DQT, Row #5: 6 9 14 17 21 27 29 24	DQT, Row #5: 26 26 26 26 26 26 26 26
	DQT, Row #6: 13 17 20 23 27 31 31 26	DQT, Row #6: 26 26 26 26 26 26 26 26
	DQT, Row #7: 19 24 25 25 29 26 27 26	DQT, Row #7: 26 26 26 26 26 26 26 26

2	**Luminance**	**Chrominance**
(512x512)		
	DQT, Row #0: 4 3 3 4 7 11 14 17	DQT, Row #0: 5 5 7 13 28 28 28 28
	DQT, Row #1: 3 3 4 5 7 16 17 15	DQT, Row #1: 5 6 7 18 28 28 28 28
	DQT, Row #2: 4 4 4 7 11 16 19 16	DQT, Row #2: 7 7 16 28 28 28 28 28
	DQT, Row #3: 4 5 6 8 14 24 22 17	DQT, Row #3: 13 18 28 28 28 28 28 28
	DQT, Row #4: 5 6 10 16 19 31 29 22	DQT, Row #4: 28 28 28 28 28 28 28 28
	DQT, Row #5: 7 10 15 18 23 29 32 26	DQT, Row #5: 28 28 28 28 28 28 28 28
	DQT, Row #6: 14 18 22 24 29 34 34 28	DQT, Row #6: 28 28 28 28 28 28 28 28
	DQT, Row #7: 20 26 27 27 31 28 29 28	DQT, Row #7: 28 28 28 28 28 28 28 28

3	**Luminance**	**Chrominance**
(540x540)		
(544x544)	DQT, Row #0: 5 3 3 5 7 12 15 18	DQT, Row #0: 5 5 7 14 30 30 30 30
	DQT, Row #1: 4 4 4 6 8 17 18 17	DQT, Row #1: 5 6 8 20 30 30 30 30
	DQT, Row #2: 4 4 5 7 12 17 21 17	DQT, Row #2: 7 8 17 30 30 30 30 30
	DQT, Row #3: 4 5 7 9 15 26 24 19	DQT, Row #3: 14 20 30 30 30 30 30 30
	DQT, Row #4: 5 7 11 17 20 33 31 23	DQT, Row #4: 30 30 30 30 30 30 30 30
	DQT, Row #5: 7 11 17 19 24 31 34 28	DQT, Row #5: 30 30 30 30 30 30 30 30
	DQT, Row #6: 15 19 23 26 31 36 36 30	DQT, Row #6: 30 30 30 30 30 30 30 30
	DQT, Row #7: 22 28 29 29 34 30 31 30	DQT, Row #7: 30 30 30 30 30 30 30 30

4	**Luminance**	**Chrominance**
(576x576)		
(600x600)	DQT, Row #0: 5 4 3 5 8 13 16 20	DQT, Row #0: 5 6 8 15 32 32 32 32
	DQT, Row #1: 4 4 4 6 8 19 19 18	DQT, Row #1: 6 7 8 21 32 32 32 32
	DQT, Row #2: 4 4 5 8 13 18 22 18	DQT, Row #2: 8 8 18 32 32 32 32 32
	DQT, Row #3: 4 5 7 9 16 28 26 20	DQT, Row #3: 15 21 32 32 32 32 32 32
	DQT, Row #4: 6 7 12 18 22 35 33 25	DQT, Row #4: 32 32 32 32 32 32 32 32
	DQT, Row #5: 8 11 18 20 26 33 36 29	DQT, Row #5: 32 32 32 32 32 32 32 32
	DQT, Row #6: 16 20 25 28 33 39 38 32	DQT, Row #6: 32 32 32 32 32 32 32 32
	DQT, Row #7: 23 29 30 31 36 32 33 32	DQT, Row #7: 32 32 32 32 32 32 32 32

5	**Luminance**	**Chrominance**
(624x624)		
(640x640)	DQT, Row #0: 5 4 3 5 8 14 17 21	DQT, Row #0: 6 6 8 16 34 34 34 34
	DQT, Row #1: 4 4 5 6 9 20 20 19	DQT, Row #1: 6 7 9 22 34 34 34 34
	DQT, Row #2: 5 4 5 8 14 19 23 19	DQT, Row #2: 8 9 19 34 34 34 34 34
	DQT, Row #3: 5 6 7 10 17 30 27 21	DQT, Row #3: 16 22 34 34 34 34 34 34
	DQT, Row #4: 6 7 13 19 23 37 35 26	DQT, Row #4: 34 34 34 34 34 34 34 34
	DQT, Row #5: 8 12 19 22 28 35 38 31	DQT, Row #5: 34 34 34 34 34 34 34 34
	DQT, Row #6: 17 22 27 30 35 41 41 34	DQT, Row #6: 34 34 34 34 34 34 34 34
	DQT, Row #7: 24 31 32 33 38 34 35 34	DQT, Row #7: 34 34 34 34 34 34 34 34

6	**Luminance**	**Chrominance**
(720x720)		
(720x900)	DQT, Row #0: 6 4 4 6 10 16 20 24	DQT, Row #0: 7 7 10 19 40 40 40 40
	DQT, Row #1: 5 5 6 8 10 23 24 22	DQT, Row #1: 7 8 10 26 40 40 40 40
	DQT, Row #2: 6 5 6 10 16 23 28 22	DQT, Row #2: 10 10 22 40 40 40 40 40
	DQT, Row #3: 6 7 9 12 20 35 32 25	DQT, Row #3: 19 26 40 40 40 40 40 40
	DQT, Row #4: 7 9 15 22 27 44 41 31	DQT, Row #4: 40 40 40 40 40 40 40 40
	DQT, Row #5: 10 14 22 26 32 42 45 37	DQT, Row #5: 40 40 40 40 40 40 40 40
	DQT, Row #6: 20 26 31 35 41 48 48 40	DQT, Row #6: 40 40 40 40 40 40 40 40
	DQT, Row #7: 29 37 38 39 45 40 41 40	DQT, Row #7: 40 40 40 40 40 40 40 40

7	**Luminance**	**Chrominance**
(750x750)		
(768x768)	DQT, Row #0: 7 5 4 7 10 17 21 26	DQT, Row #0: 7 8 10 20 42 42 42 42
	DQT, Row #1: 5 5 6 8 11 24 25 23	DQT, Row #1: 8 9 11 28 42 42 42 42
	DQT, Row #2: 6 5 7 10 17 24 29 24	DQT, Row #2: 10 11 24 42 42 42 42 42
	DQT, Row #3: 6 7 9 12 21 37 34 26	DQT, Row #3: 20 28 42 42 42 42 42 42
	DQT, Row #4: 8 9 16 24 29 46 43 32	DQT, Row #4: 42 42 42 42 42 42 42 42
	DQT, Row #5: 10 15 23 27 34 44 47 39	DQT, Row #5: 42 42 42 42 42 42 42 42
	DQT, Row #6: 21 27 33 37 43 51 50 42	DQT, Row #6: 42 42 42 42 42 42 42 42
	DQT, Row #7: 30 39 40 41 47 42 43 42	DQT, Row #7: 42 42 42 42 42 42 42 42

8	**Luminance**	**Chrominance**
(800x800)		
	DQT, Row #0: 7 5 4 7 11 18 22 27	DQT, Row #0: 7 8 11 21 44 44 44 44
	DQT, Row #1: 5 5 6 8 11 26 26 24	DQT, Row #1: 8 9 11 29 44 44 44 44
	DQT, Row #2: 6 6 7 11 18 25 30 25	DQT, Row #2: 11 11 25 44 44 44 44 44
	DQT, Row #3: 6 7 10 13 22 38 35 27	DQT, Row #3: 21 29 44 44 44 44 44 44
	DQT, Row #4: 8 10 16 25 30 48 45 34	DQT, Row #4: 44 44 44 44 44 44 44 44
	DQT, Row #5: 11 15 24 28 36 46 50 40	DQT, Row #5: 44 44 44 44 44 44 44 44
	DQT, Row #6: 22 28 34 38 45 53 53 44	DQT, Row #6: 44 44 44 44 44 44 44 44
	DQT, Row #7: 32 40 42 43 49 44 45 44	DQT, Row #7: 44 44 44 44 44 44 44 44

9	**Luminance**	**Chrominance**
(832x832)		
	DQT, Row #0: 7 5 5 7 11 18 23 28	DQT, Row #0: 8 8 11 22 46 46 46 46
	DQT, Row #1: 6 6 6 9 12 27 28 25	DQT, Row #1: 8 10 12 30 46 46 46 46
	DQT, Row #2: 6 6 7 11 18 26 32 26	DQT, Row #2: 11 12 26 46 46 46 46 46
	DQT, Row #3: 6 8 10 13 23 40 37 29	DQT, Row #3: 22 30 46 46 46 46 46 46
	DQT, Row #4: 8 10 17 26 31 50 47 35	DQT, Row #4: 46 46 46 46 46 46 46 46
	DQT, Row #5: 11 16 25 29 37 48 52 42	DQT, Row #5: 46 46 46 46 46 46 46 46
	DQT, Row #6: 23 29 36 40 47 56 55 46	DQT, Row #6: 46 46 46 46 46 46 46 46
	DQT, Row #7: 33 42 44 45 52 46 47 46	DQT, Row #7: 46 46 46 46 46 46 46 46

10 (854X854) (864X864)	**Luminance**	**Chrominance**
	DQT, Row #0: 8 5 5 8 12 19 24 29	DQT, Row #0: 8 9 12 23 48 48 48 48
	DQT, Row #1: 6 6 7 9 12 28 29 26	DQT, Row #1: 9 10 12 32 48 48 48 48
	DQT, Row #2: 7 6 8 12 19 27 33 27	DQT, Row #2: 12 12 27 48 48 48 48 48
	DQT, Row #3: 7 8 11 14 24 42 38 30	DQT, Row #3: 23 32 48 48 48 48 48 48
	DQT, Row #4: 9 11 18 27 33 52 49 37	DQT, Row #4: 48 48 48 48 48 48 48 48
	DQT, Row #5: 12 17 26 31 39 50 54 44	DQT, Row #5: 48 48 48 48 48 48 48 48
	DQT, Row #6: 24 31 37 42 49 58 58 48	DQT, Row #6: 48 48 48 48 48 48 48 48
	DQT, Row #7: 35 44 46 47 54 48 49 48	DQT, Row #7: 48 48 48 48 48 48 48 48

11 (900X900)	**Luminance**	**Chrominance**
	DQT, Row #0: 8 6 5 8 12 20 26 31	DQT, Row #0: 9 9 12 24 50 50 50 50
	DQT, Row #1: 6 6 7 10 13 29 30 28	DQT, Row #1: 9 11 13 33 50 50 50 50
	DQT, Row #2: 7 7 8 12 20 29 35 28	DQT, Row #2: 12 13 28 50 50 50 50 50
	DQT, Row #3: 7 9 11 15 26 44 40 31	DQT, Row #3: 24 33 50 50 50 50 50 50
	DQT, Row #4: 9 11 19 28 34 55 52 39	DQT, Row #4: 50 50 50 50 50 50 50 50
	DQT, Row #5: 12 18 28 32 41 52 57 46	DQT, Row #5: 50 50 50 50 50 50 50 50
	DQT, Row #6: 25 32 39 44 52 61 60 51	DQT, Row #6: 50 50 50 50 50 50 50 50
	DQT, Row #7: 36 46 48 49 56 50 52 50	DQT, Row #7: 50 50 50 50 50 50 50 50

12 (960X960)	**Luminance**	**Chrominance**
	DQT, Row #0: 9 6 5 9 13 22 28 33	DQT, Row #0: 9 10 13 25 53 53 53 53
	DQT, Row #1: 6 6 8 10 14 31 32 30	DQT, Row #1: 10 11 14 36 53 53 53 53
	DQT, Row #2: 8 7 9 13 22 31 37 30	DQT, Row #2: 13 14 30 53 53 53 53 53
	DQT, Row #3: 8 9 12 16 28 47 43 33	DQT, Row #3: 25 36 53 53 53 53 53 53
	DQT, Row #4: 10 12 20 30 37 59 56 42	DQT, Row #4: 53 53 53 53 53 53 53 53
	DQT, Row #5: 13 19 30 35 44 56 61 50	DQT, Row #5: 53 53 53 53 53 53 53 53
	DQT, Row #6: 26 35 42 47 56 65 65 55	DQT, Row #6: 53 53 53 53 53 53 53 53
	DQT, Row #7: 39 50 51 53 60 54 56 53	DQT, Row #7: 53 53 53 53 53 53 53 53

13	**Luminance**	**Chrominance**
(1000X1000)		
(1024X1024)	DQT, Row #0: 9 6 6 9 13 22 29 34	DQT, Row #0: 10 10 13 26 55 55 55 55
	DQT, Row #1: 7 7 8 11 15 32 34 31	DQT, Row #1: 10 12 15 37 55 55 55 55
	DQT, Row #2: 8 7 9 13 22 32 39 31	DQT, Row #2: 13 15 31 55 55 55 55 55
	DQT, Row #3: 8 10 12 16 29 49 45 35	DQT, Row #3: 26 37 55 55 55 55 55 55
	DQT, Row #4: 10 12 21 31 38 61 58 43	DQT, Row #4: 55 55 55 55 55 55 55 55
	DQT, Row #5: 13 20 31 36 45 58 63 52	DQT, Row #5: 55 55 55 55 55 55 55 55
	DQT, Row #6: 27 36 44 49 58 68 67 57	DQT, Row #6: 55 55 55 55 55 55 55 55
	DQT, Row #7: 40 52 53 55 63 56 58 55	DQT, Row #7: 55 55 55 55 55 55 55 55

14	**Luminance**	**Chrominance**
(1050x1050)		
	DQT, Row #0: 9 6 6 9 14 23 30 35	DQT, Row #0: 10 10 14 27 57 57 57 57
	DQT, Row #1: 7 7 8 11 15 34 35 32	DQT, Row #1: 10 12 15 38 57 57 57 57
	DQT, Row #2: 8 8 9 14 23 33 40 32	DQT, Row #2: 14 15 32 57 57 57 57 57
	DQT, Row #3: 8 10 13 17 30 50 46 36	DQT, Row #3: 27 38 57 57 57 57 57 57
	DQT, Row #4: 10 13 21 32 39 63 60 45	DQT, Row #4: 57 57 57 57 57 57 57 57
	DQT, Row #5: 14 20 32 37 47 60 66 53	DQT, Row #5: 57 57 57 57 57 57 57 57
	DQT, Row #6: 28 37 45 50 60 70 70 59	DQT, Row #6: 57 57 57 57 57 57 57 57
	DQT, Row #7: 42 53 55 57 65 58 60 57	DQT, Row #7: 57 57 57 57 57 57 57 57

15	**Luminance**	**Chrominance**
(1080x1080)		
	DQT, Row #0: 10 7 6 10 14 24 31 37	DQT, Row #0: 10 11 14 28 59 59 59 59
	DQT, Row #1: 7 7 8 11 16 35 36 33	DQT, Row #1: 11 13 16 40 59 59 59 59
	DQT, Row #2: 8 8 10 14 24 34 41 34	DQT, Row #2: 14 16 34 59 59 59 59 59
	DQT, Row #3: 8 10 13 17 31 52 48 37	DQT, Row #3: 28 40 59 59 59 59 59 59
	DQT, Row #4: 11 13 22 34 41 65 62 46	DQT, Row #4: 59 59 59 59 59 59 59 59
	DQT, Row #5: 14 21 33 38 49 62 68 55	DQT, Row #5: 59 59 59 59 59 59 59 59
	DQT, Row #6: 29 38 47 52 62 73 72 61	DQT, Row #6: 59 59 59 59 59 59 59 59
	DQT, Row #7: 43 55 57 59 67 60 62 59	DQT, Row #7: 59 59 59 59 59 59 59 59

16	**Luminance**	**Chrominance**
(800x800) Original: 1280x800 (Upload #2)	DQT, Row #0: 10 10 10 11 15 22 34 51 DQT, Row #1: 10 10 12 16 20 24 32 45 DQT, Row #2: 10 12 14 19 26 37 55 81 DQT, Row #3: 11 16 19 24 32 44 64 94 DQT, Row #4: 15 20 26 32 41 56 79 113 DQT, Row #5: 22 24 37 44 56 74 101 143 DQT, Row #6: 34 32 55 64 79 101 136 187 DQT, Row #7: 51 45 81 94 113 143 187 251	DQT, Row #0: 10 10 10 11 15 22 34 51 DQT, Row #1: 10 10 12 16 20 24 32 45 DQT, Row #2: 10 12 14 19 26 37 55 81 DQT, Row #3: 11 16 19 24 32 44 64 94 DQT, Row #4: 15 20 26 32 41 56 79 113 DQT, Row #5: 22 24 37 44 56 74 101 143 DQT, Row #6: 34 32 55 64 79 101 136 187 DQT, Row #7: 51 45 81 94 113 143 187 251

17	**Luminance**	**Chrominance**
(1080x1080) Original: 3200x2048 (Upload #2)	DQT, Row #0: 10 10 10 11 15 22 34 51 DQT, Row #1: 10 10 12 16 20 24 32 45 DQT, Row #2: 10 12 14 19 26 37 55 81 DQT, Row #3: 11 16 19 24 32 44 64 94 DQT, Row #4: 15 20 26 32 41 56 79 113 DQT, Row #5: 22 24 37 44 56 74 101 143 DQT, Row #6: 34 32 55 64 79 101 136 187 DQT, Row #7: 51 45 81 94 113 143 187 251	DQT, Row #0: 10 10 10 11 15 22 34 51 DQT, Row #1: 10 10 12 16 20 24 32 45 DQT, Row #2: 10 12 14 19 26 37 55 81 DQT, Row #3: 11 16 19 24 32 44 64 94 DQT, Row #4: 15 20 26 32 41 56 79 113 DQT, Row #5: 22 24 37 44 56 74 101 143 DQT, Row #6: 34 32 55 64 79 101 136 187 DQT, Row #7: 51 45 81 94 113 143 187 251