In this activity, we investigate on various image types and formats, among which are the basic types and the advanced types. Basic types include binary, grayscale, truecolor, and indexed images, while advanced types include (but not limited to) high dynamic range (HDR) images, multi/hyper-spectral images, 3D images, and videos. Images for each type were obtained online (except for the HDR image and video). Shown below in Fig. 1 are the images used and listed in Table 1 are the properties found from the images.

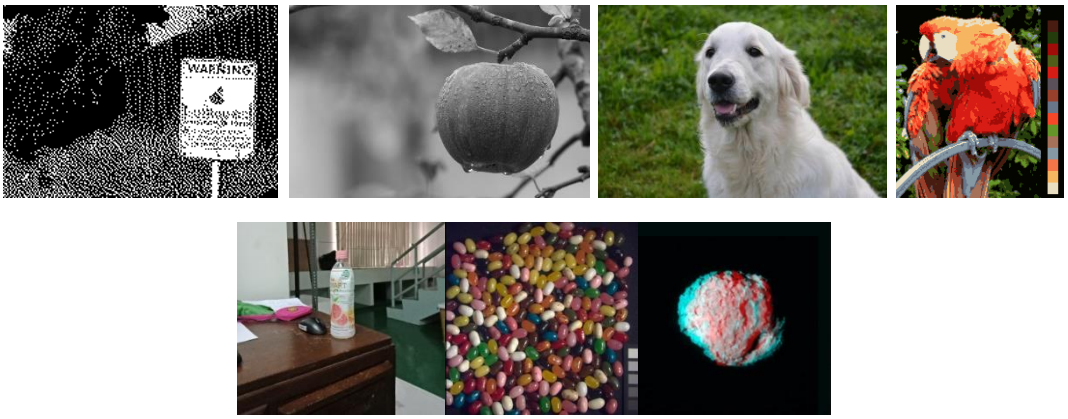

Figure 1. Top, Left to Right: binary image, grayscale image, truecolor image, indexed image. Bottom, Left to Right: HDR image, multispectral image, 3D image.

Table 1. Image properties of obtained images.

Image	Image Type	Width (px)	Height (px)	Horizontal Resolution (dpi)	Vertical Resolution (dpi)	Bit Depth	Size
Binary	PNG	200	140	-	-	1	2.59 KB
Grayscale	JPG	1880	1204	96	96	8	147 KB
Truecolor	JPG	960	640	300	300	24	132 KB
Indexed	PNG	174	200	-	-	8	10.3 KB
HDR	JPG	3456	4608	96	96	24	4.50 MB
Multispectral	BMP	512	512	-	-	24	768 KB
3D	JPG	1024	1024	72	72	24	274 KB

Doing a bit of research about bit depth, it refers to the capacity of an image to store color information, that is, the higher the bit depth, the greater the capacity to store color information. We see this from Table 1: the binary image only has a bit depth of 1 which makes sense because it only displays two values (0 and 1), the grayscale and indexed images have bit depths of 8, while the rest have bit depths of 24 (which may be due to the three image channels RGB)[1]. Ultimately, the size of the image increases when properties such as bit depth, resolution, and dimensions increase.

Below are some additional camera properties found from the truecolor image, HDR image, as well as properties from the video. Also, two more additional properties were present in the HDR image, namely: resolution unit, and color representation.

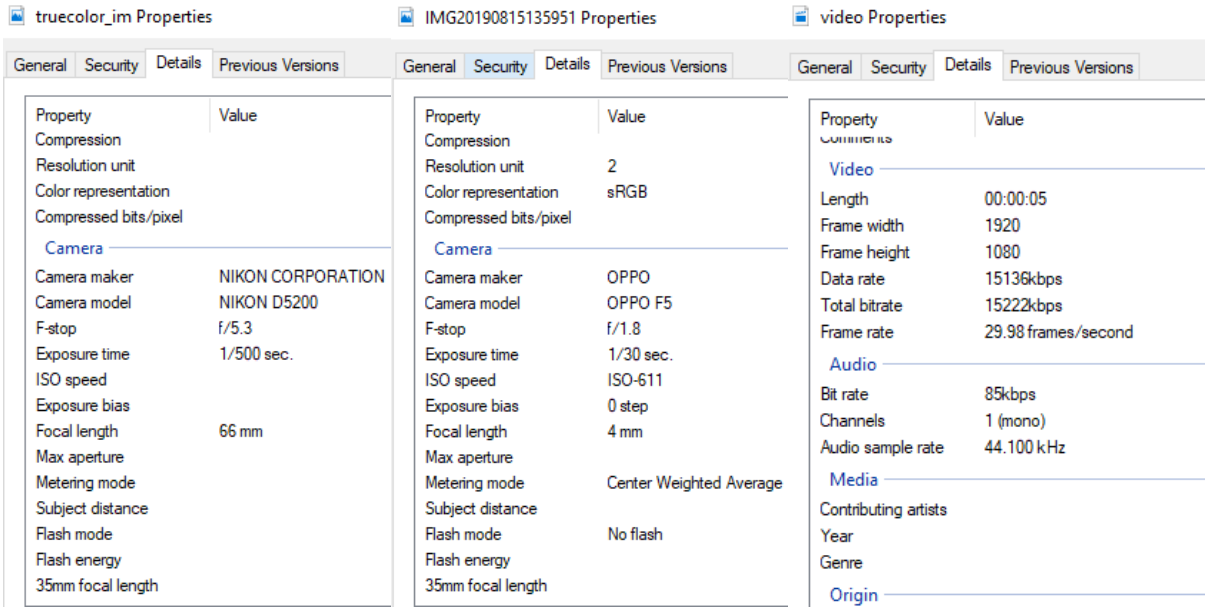

Figure 2. Additional image properties for (left to right): truecolor image, HDR image, video.

Several features of GIMP were also tested on the images, namely: Image Properties, Export, Histogram, and Crop. The Export feature will be useful when images are needed to be saved into different file formats, the Crop feature will be useful when only specific parts of an image is needed. Image Properties shows several image information such as what is shown in Fig. 3. Histogram shows the pixel intensity values (in my case, on all image channels RGB).

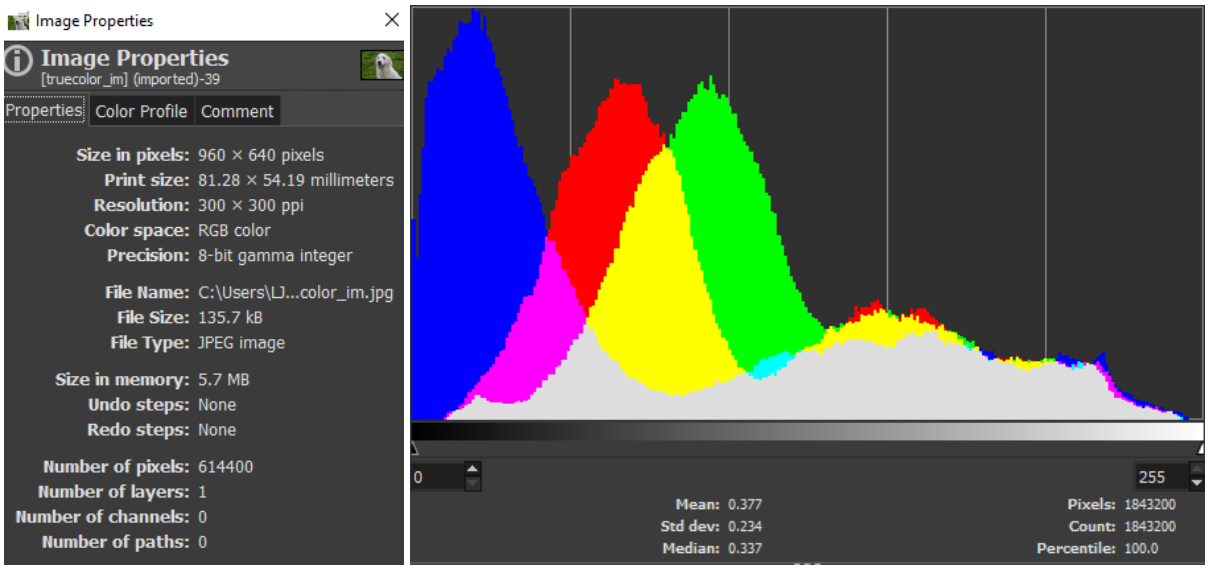

Figure 3. Left: GIMP Image Properties feature. Right: GIMP Histogram feature. Both were done on truecolor image.

Using the Histogram feature on GIMP, one can convert a truecolor image into a binary image by using the Threshold command under Colors tab; such is shown in Fig. 4.

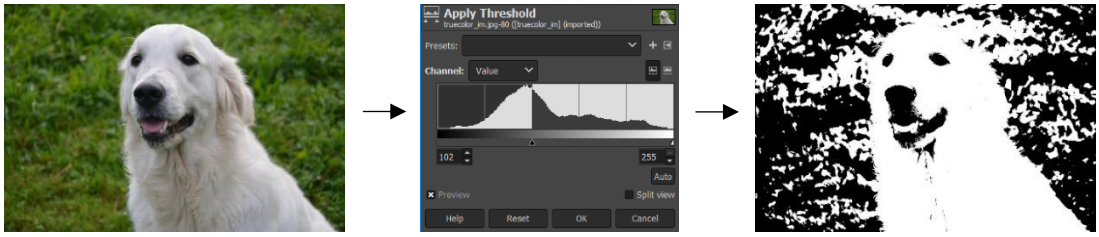

Figure 4. Truecolor image into binary image via thresholding.

Using the available Matlab R2010b at the CSRC Lab PCs, we were to turn our truecolor image into binary, grayscale, and indexed images. To be able to do this, various Matlab commands were used; each command will be discussed below.

To begin, an image **I** is loaded into the Matlab workspace by using the command **imread(I)**, and may be displayed using the command **imshow(I)**. To save a random workspace variable, the function **imwrite(I, filename)** is used.

To turn the truecolor image into binary, the command **im2bw(I, level)** was used. This function takes in image **I** with the threshold **level**. To determine **level**, I used the command **graythresh(I)** which gets the threshold level of a grayscale image. The executed code looked like:

```
I_bw = im2bw(I,graythresh(I));
```

To turn the truecolor image into grayscale, I just used the command **rgb2gray(RGB)**, which takes in truecolor image **RGB** and, as the name implies, converts image into grayscale. The executed code looked like:

```
I_gray = rgb2gray(I);
```

To turn the truecolor image into an indexed image, I used the command **rgb2ind(RGB, Q)** which takes in truecolor image **RGB** with **Q** number of colors (upon our discretion). It produces two outputs, namely: the indexed image and the accompanying colormap. The executed code looked something like this:

```
[I_ind, cmap] = rgb2ind(I, 16);
```

The resulting images are shown in Fig. 5 below for the binary, grayscale, and indexed images:

Figure 5. Left to right: binary image, grayscale image, indexed image.

To further investigate on indexed images, I decided to turn my previous indexed image earlier into an RGB image. I found out that using **`imshow()`** on the indexed image would only show its literal indices; I confirmed this when I checked out its histogram using **`imhist()`**, the histogram showed only 16 lines, the number of indices of the image.

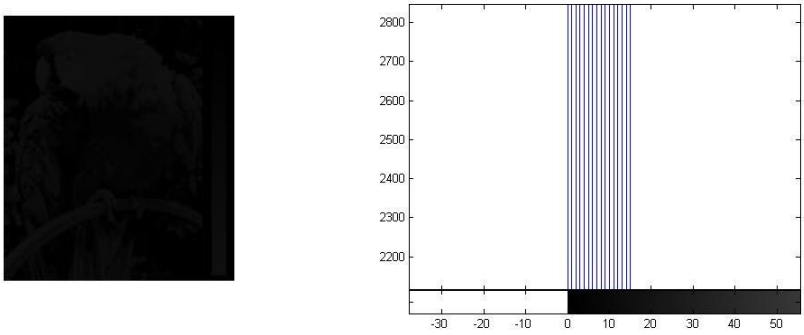

Figure 6. Displayed image of indexed image and its histogram.

To remedy this, I had to figure out the colormap that this image used. For that, I would like to thank my seatmate Andy for cooperating with me to obtain the colormap of the image using the color picker from GIMP (all 16 colors on the palette of the indexed image were subjected to the color picker as shown in Fig. 7).

Figure 7. RGB values of the colors in the colorbar were taken using GIMP

To finally convert the indexed image into truecolor, now that I have the indexed image and the colormap, I used the command **`ind2rgb(X, map)`** where **`X`** is the indexed image and **`map`** is the colormap I just obtained. Shown in Fig. 8 is the RGB image I produced.

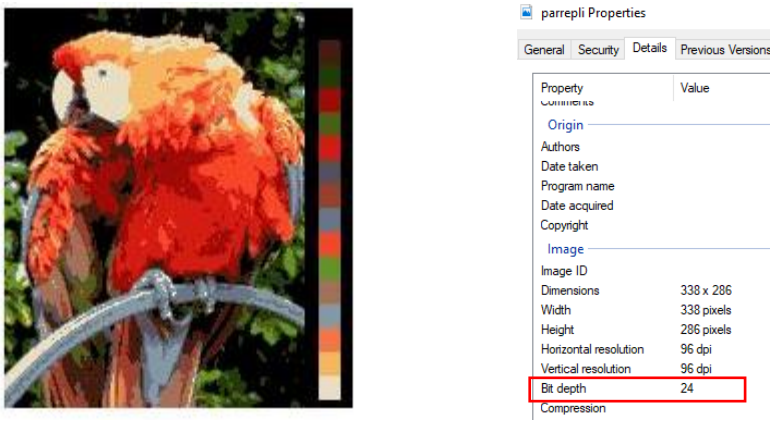

Figure 8. Resulting RGB image from indexed image and colormap, and its image properties.

To confirm if I indeed obtained an RGB image, I went into its image properties and indeed found that it had a bit depth of 24, the same value we saw for truecolor images and such, a while ago. I also tried to use a different colormap just for fun (the colormap I obtained from the truecolor image of the dog) and this is what I got:

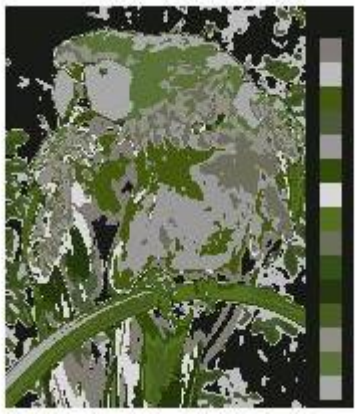

Figure 9. Indexed image with a different colormap.

On different file formats, I researched about JPG, GIF, PNG, and TIFF file formats. The first two (JPG and GIF) are lossy file formats where not all image information are saved and loss of data may occur upon compression. The JPG file format was created by the Joint Photographic Experts Group to which the file was named after. This file format may be used when small file sizes are crucial, and quality is not that much of a priority, e.g. posting images online so they would load quickly on web pages and the like[2]. The GIF file format stands for Graphics Interchange Format and was created by developer Steve Wilhite and his team at tech company CompuServe; GIFs were initially for still images but was further revolutionized by a specific compression algorithm (LZW compression) named after its creators Abraham Lempel, Jacob Ziv, and Terry Welch). Similar to JPG, GIFs are great to use when small file sizes are needed and images are needed to be loaded quickly such as in the internet[3].

On the other hand, PNG and TIFF are lossless file formats. PNG stands for Portable Network Graphics and was first developed in 1995 to compete with the already well-established GIF file format. It soon became capable of supporting a wider spectrum of colors. PNGs are great to use when transparency and quality are needed to be preserved[4]. TIFF stands for Tagged Image Format and was born out of a need going way back to the 1980s; this file format was developed to be used along with document scanners. TIFF files may be useful when formatting of documents need to be preserved across different devices and screens, while still maintaining high quality even when compressed[5].

Overall, I grade myself 12/10 because I was able to produce all required output. Additionally, indexed images kept me interested so I played with them further. And as I was making this paper, I applied what I learned previously and resized images so that I would decrease the total file size of short report.

REFERENCES

[1] - What is bit depth? (n.d.) Retrieved from Tech-Ease: https://etc.usf.edu/techease/win/images/what-is-bit-depth/

[2] – Understanding JPG (2013) Retrieved from Logaster: https://www.logaster.com/blog/what-is-jpg/

[3] – Boissoneault, L. (2017). A brief history of the GIF, from early internet innovation to ubiquitous relic. Retrieved from Smithsonian.com: https://www.smithsonianmag.com/history/brief-history-gif-early-internet-innovation-ubiquitous-relic-180963543/

[4] – Walker, D., Hopping, C. (2019). What is a PNG file? Retrieved from ITPro: https://www.itpro.co.uk/web-browser/30247/what-is-a-png-file

[5] – Massart, R. (2015). The history of TIFF and why you should use it. Retrieved from Peernet: https://www.peernet.com/why-use-tiff-who-uses-tiff/