Exploring image types and formats, and their functions

Andrea Rica Advincula | advinculaar@gmail.com | 2015-04544

In this age of technology, images have been used for a wide variety of reasons ranging from leisure to scientific study. This activity allows us to know about the different types of images, and how to be able to manipulate them for their designated uses. The types of images are divided into two parts: the basic, and advanced types. Figure 1 shows us samples of each of the 7 types excluding for videos, which are impossible to be presented on a document.

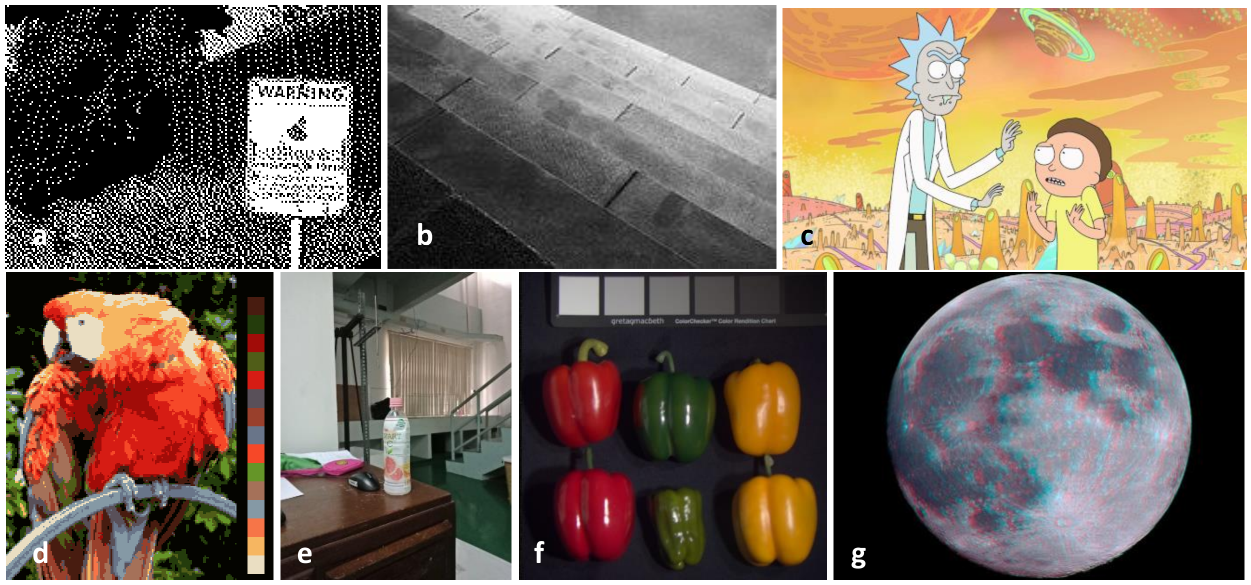


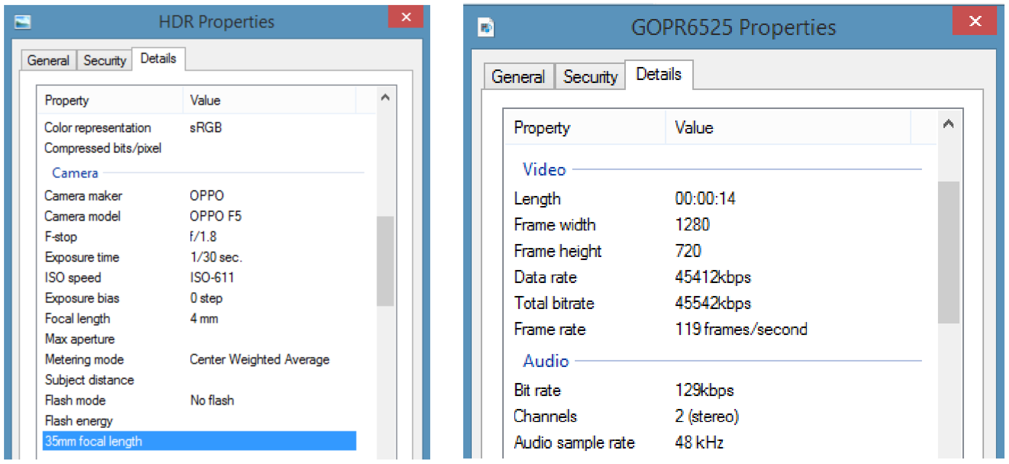
Figure 1. Images of different types downloaded from the internet a) binary image [1] **b)** grayscale image [1] **c)** truecolor image [2] **d)** indexed image [1] **e)** high dynamic range (HDR) image **f)** multispectral image [3] **g)** 3D image [4].

FileName	Type	Size	Dimensions	Width	Height	Bit depth	Resolution
binary (Fig. 1a)	PNG	2.59 KB	200 x 140	200 pixels	140 pixels	1	
grayscale (Fig. 1b)	JPEG	8.56 KB	220 x 147	220	147	8	96 dpi
truecolor_im (Fig. 1c)	JPEG	425 KB	1000 x 563	1000	563	24	72
indexed_im (Fig. 1d)	PNG	10.3 KB	174 x 200	174	200	8	-
HDR (Fig. 1e)	JPEG	4.50 MB	3456 x 4608	3456	4608	24	96
multispectral (Fig. 1f)	Bitmap	768 KB	512x512	512	512	24	-
3D (Fig. 1g)	JPEG	231 KB	1024 x 768	1024	768	24	96
GOPR6525	MP4	76 MB	-	1280	720	-	-

Table 1. List of details of the collected images.

The distinct difference in properties for the images is their Bit depth. A binary image has a bit depth of 1, because it only contains the values of 0, and 1 corresponding to black, and white respectively. A grayscale image has an 8-bit depth which allows 256 different shades of gray [1]. A truecolor image, as well as the HDR, multispectral, and 3D image has a 24-bit depth which allows 3 layers of 256 different shades of red, green, and blue. The combination of these layers forms the image. Similar to a grayscale image, an indexed image has an 8-bit depth because the RGB values used on the image are stored as indexes. The colors on the image is now limited to the indexes defined. Some images with higher bit depth could present opacity in an alpha channel on top of the RGB values it contains [5]. When an image is taken by a camera the properties of the camera shows up in its properties. Fig. 2 (left) are the properties of the HDR image taken on my Oppo cellular phone. This HDR image has a color representation of sRGB or Standard RGB. This is a specification for the range of colors a screen display, in this case, my cellular phone [6]. Since a video, are sets of images in temporal space embedded with audio, it has properties that define the image and audio in time (Fig. 2).

Noticeably, the file types in Table 1 are also varying. These are either lossy or lossless image compression, which in its name is self-explanatory. The lossy type does not save all information, but a human observer would not notice this loss. The JPEG is a lossy format and is the usual file type in cellular phones, and digital cameras where each specific detail is not entirely necessary. The GIF is a bitmap image format, and also a lossy format. However, it could support a temporal image with no audio. The lossless type saves all information which is best for scientific analysis like x-rays. The PNG is a lossless format, it can support, 1-bit, 8-bit, and even higher bits that contains an alpha layer for opacity. The TIFF format started as an attempt to a common format for desktop scanners which only handles 1-bit. However, today it can fully support color images [1]. These formats were created by teams of developer and were up against others who were developing their own compressions. These formats rose to be the most popular ones.



Property	Value
Color representation	sRGB
Compressed bits/pixel	
Camera	
Camera maker	OPPO
Camera model	OPPO F5
F-stop	f/1.8
Exposure time	1/30 sec.
ISO speed	ISO-611
Exposure bias	0 step
Focal length	4 mm
Max aperture	
Metering mode	Center Weighted Average
Subject distance	
Flash mode	No flash
Flash energy	
35mm focal length	

Property	Value
Video	
Length	00:00:14
Frame width	1280
Frame height	720
Data rate	45412kbps
Total bitrate	45542kbps
Frame rate	119 frames/second
Audio	
Bit rate	129kbps
Channels	2 (stereo)
Audio sample rate	48 kHz

Figure 2. Screenshot of other properties displayed for the HDR image (left), and the temporal images/ video (right).

To be able to manipulate images, we can use a free image editing software. I am using GIMP because its properties, and tools are straightforward. Fig. 3 shows us the Image Properties, and Histogram window of the 'truecolor_im.jpg' file (Fig. 1c). The Image Properties list downs the information the image file contains, like the ones in Table 1. The Histogram in Fig. 3 are the RGB values found in the image, however, you could isolate the values for every layer by choosing the desired layer in the dropdown list on the upper left corner. You could also use the crop feature, that removes areas from the image. It is usually using a rectangular selection box but can also be a free form selection. You can choose to only keep your selected area or take it away. After manipulating your image, saving the file would save it as a GIMP-editable file and not as an image. You therefore use the 'Export' feature which allows you to save your image into the filetype you want.

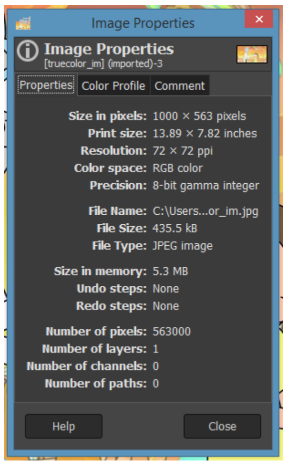
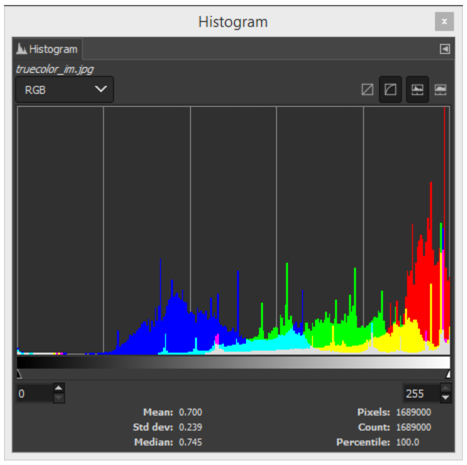


Image Properties

Image Properties
[truecolor_im] (imported)-3

Properties | Color Profile | Comment

Size in pixels:	1000 × 563 pixels
Print size:	13.89 × 7.82 inches
Resolution:	72 × 72 ppi
Color space:	RGB color
Precision:	8-bit gamma integer
File Name:	C:\Users...or_im.jpg
File Size:	435.5 kB
File Type:	JPEG image
Size in memory:	5.3 MB
Undo steps:	None
Redo steps:	None
Number of pixels:	563000
Number of layers:	1
Number of channels:	0
Number of paths:	0

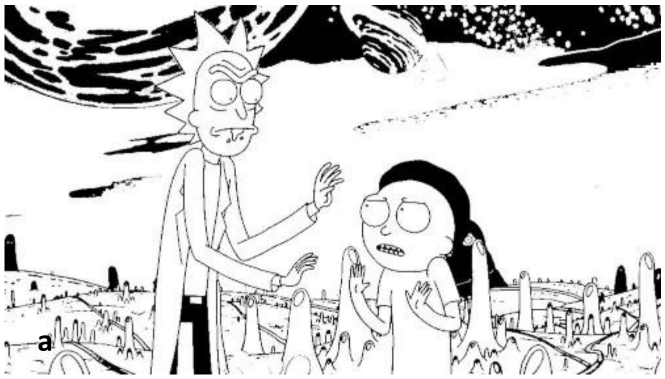


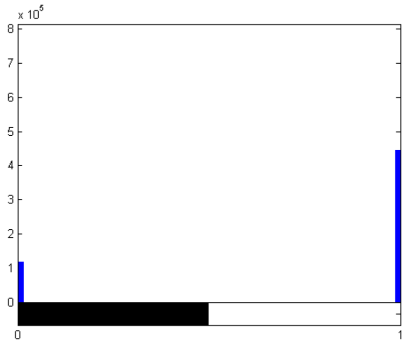
Mean:	0.700	**Pixels:**	1689000
Std dev:	0.239	**Count:**	1689000
Median:	0.745	**Percentile:**	100.0

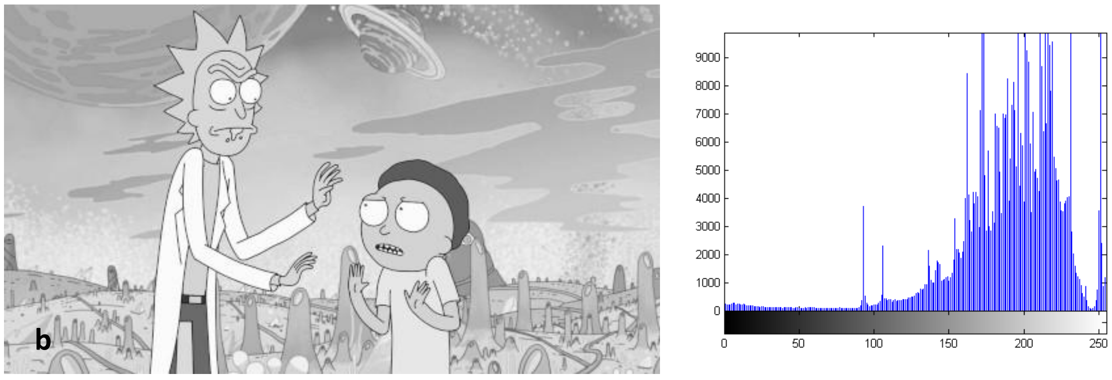
Figure 3. Screenshot of the Image Properties (left), and Histogram (right) features of GIMP.

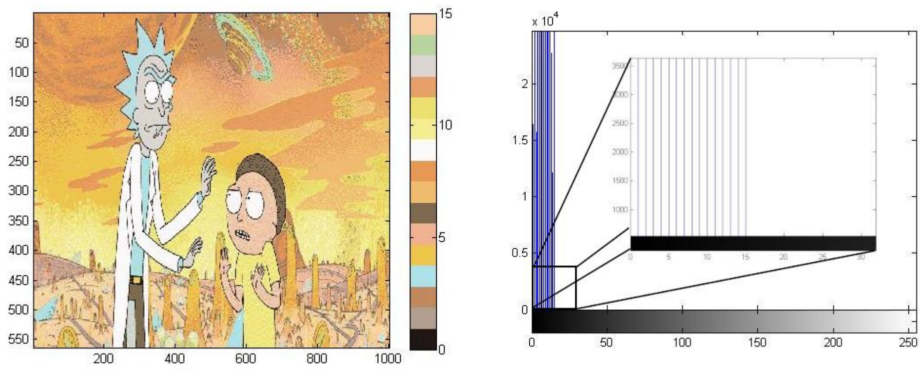
Other than GIMP, you could also manipulate images via programming. I used MatLab because I am familiar with its image processing syntax. I converted 'truecolor_im.jpg' to a binary, grayscale, and indexed image using the functions im2bw, rgb2gray, and rgb2ind respectively. The resulting images are shown in Fig. 4 along with its corresponding histogram. Fig. 4a is a binary image and it shows in its histogram that there are only two values: 0 and 1 corresponding to the only two color it uses. Fig. 4b, the grayscale image, shows a histogram of a variety of gray shades. Lastly, and the most interesting for me is the indexed image, wherein its histogram shows 16 discrete bars that correspond to the 16 colors indexed. These 16 colors are the only colors used in the image. I have displayed its color palette beside its resulting image. The function rgb2ind outputs two data sets, the image and its colormap. The colormap is its list of RGB values stored in each index. The exported images, though coming from the same original image have decreasing file sizes from the original image, the indexed, the grayscale, and the smallest file size, the binary image.

Image | Histogram







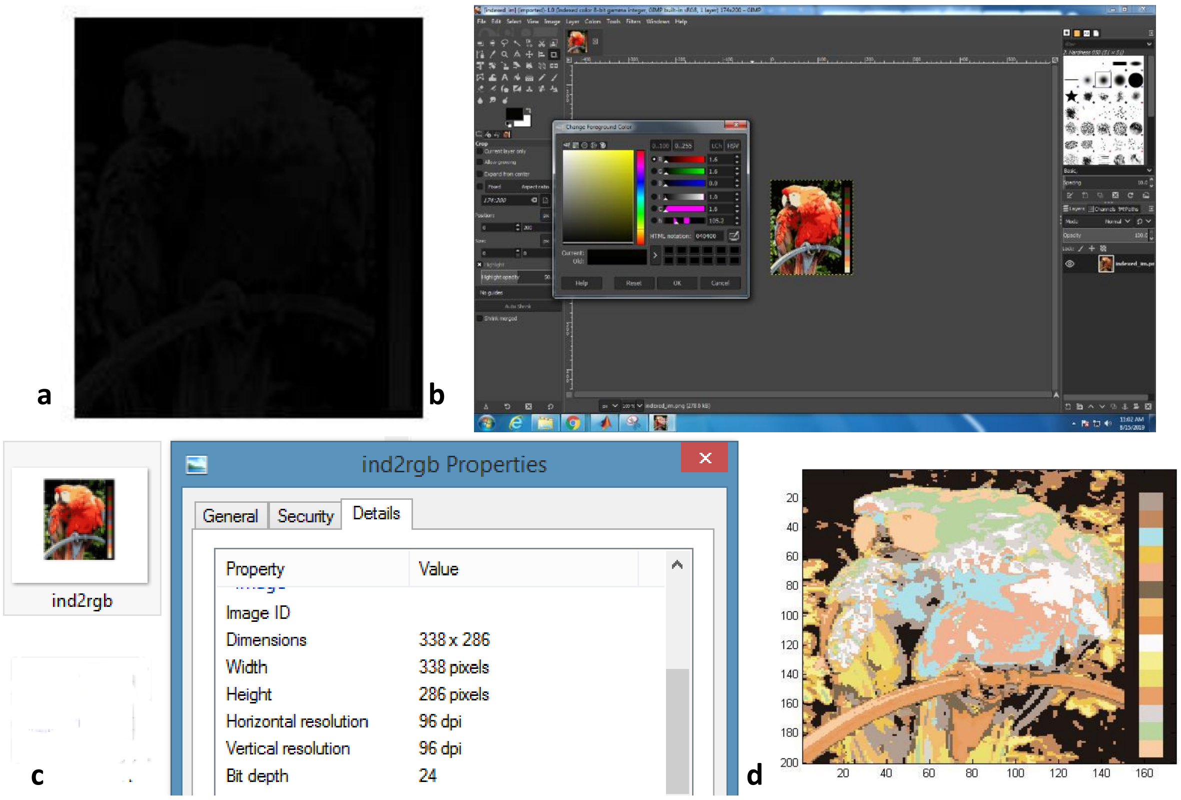


c

Figure 4. Resulting images after converting the truecolor image sample to a a) binary image, **b)** grayscale image, and **c)** indexed image.

Since, I was so fascinated by the indexed image, as a challenge I wanted to try converting the indexed image ‘indexed_im.png’ (Fig. 1d) into an RGB image. However, when I opened the image in MatLab using imshow it gave me a dark image with shadows of the indexed image (Fig. 5a). I realize this was because the image only contains integers from 0 to 15 corresponding to its index. To be able to convert the image to RGB, I needed the colormap of the indexed image, which I had no idea where to get. With the help of my seatmate, LJ, we brainstormed ideas on how to extract the colormap. It took us 30 min. of trying different methods and researching before he suggested to use GIMP’s color picker to get the palette’s RGB values (Fig. 5b). The RGB values in GIMP were in a range of 0-100, so we divided the values by 100 to be MatLab compatible. And, VOILA! We were able to convert the indexed image to an RGB type (Fig. 5c). I was having fun with all the experimenting, I even tried to input the colormap from Fig. 4c and the resulting parrot has become crazy (Fig. 5d).

I give myself 12/10 for this activity because I was able to accomplish all requirements and tried experimenting from the things I have learned. Moreover, I was able to apply the previous lesson on scaling the image to reduce the file size of a document. If I could, I would definitely give myself +3 bonus points for an enjoyable job well done. In total, 15/10.



ind2rgb Properties

General Security Details

Property	Value
Image ID	
Dimensions	338 x 286
Width	338 pixels
Height	286 pixels
Horizontal resolution	96 dpi
Vertical resolution	96 dpi
Bit depth	24

Figure 5. Converting an indexed image to RGB. a) indexed image w/o colormap **b)** extracting colormap manually from GIMP color picker **c)** successful conversion from 8-bit to 24-bit **d)** image applied with Fig. 4c colormap

Sources

1. https://en.wikipedia.org/
2. https://variety.com/2019/tv/news/rick-and-morty-season-4-premiere-date-1203215745/
3. http://www.cs.columbia.edu/CAVE/databases/multispectral/food_and_drinks/
4. https://apod.nasa.gov/apod/ap070602.html
5. http://www.libpng.org/pub/png/book/chapter08.html
6. https://techterms.com/definition/srgb