Nombre completo: Abel Aguilar Chávez
Matricula: A01112847
No de práctica: 1
Fecha de entrega: 29/01/2018

Resumen de la práctica:

La práctica consistió en tres actividades relacionadas a espacios de color. En su primer parte se analizó los valores máximos y mínimos de los componentes RGB de una imagen, respectivamente. Posteriormente, se realizaron transformaciones de formato de la imagen RGB original a escala de grises, CMY, YIQ, YUV, YCbCr, y HSI. Y por último se investigó cómo se representa una imagen indexada, al igual que cómo realizar el indexado de una, para finalizar con el desarrollo práctico de lo anteriormente investigado.

Introducción:

La descripción de un color a partir de las capacidades personales de cada individuo es un método subjetivo, es por ello que nacen los espacios de color para permitir el procesamiento descriptivo de una imagen. Donde una imagen se puede interpretar como un mapa de recuadros, conocidos como pixel, unidad estructural de una imagen que no contiene dimensiones. Cada pixel tiene la función de representar un color, respetando el espacio de color en el cual se está trabajando.

Existen distintos espacios de color, y cada uno expresa atributos distintos de una imagen. Dichos atributos son brillo, matiz, luminosidad, croma, y saturación.

El formato de color más común es RGB (Rojo, verde, y azul por sus siglas en inglés). Formado a partir la adición de cada uno de sus componentes de color.

El espacio de color CMY(Cian, magenta, y amarillo por sus siglas en inglés), a diferencia de la formación de una imagen RGB, este se conforma de forma sustractiva de cada uno de sus componentes de color. En ciertas también a este espacio de color se puede agregar negro como uno de sus componentes, CMY(K) K por la terminación de la palabra negro en inglés, black.

El espacio HSL (Matiz, saturación, y luminosidad por sus siglas en inglés), también se puede encontrar con otros nombres como HSI donde I es intensidad, HSV donde V es valor, HCI donde C es chroma, HVC; Valor y croma, y TSD; matiz, saturación, y oscuridad. La forma en que traduce una imagen RGB es convirtiendo la paleta de colores a un conformación polar, y posteriormente marcando dos vértices, uno superior y otro inferior, que serán los colores blanco y negro. El matiz dirá dirección angular en la paleta de color, la saturación dictará la magnitud del vector, y la intensidad nos dirá que tanto se aproxima a un color blanco, o negro. (Jack, 2009)

Los próximos formatos comparten su función de modular luminosidad y el componente cromático de una imagen. YIQ, distinto de CMY la Y es conocida como luma, I es

referente a la fase por su nombre en inglés (In-phase), y Q a cuadratura de igual forma por su nomenclatura en inglés y es un derivado del espacio YUV utilizado por PAL(Línea de alteración de fase, por sus siglas en inglés) y la NTSC (Comité nacional del sistema televisivo). donde Y se mantiene con la misma terminología, pero sus componentes U y V son coordenadas en un plano donde la paleta de color se ve proyectada. Y de igual forma YCbCr es una versión de escalda del formato YIQ pero con un offset, y es utilizada como estándar del componente de vídeo digital. (Ford, 1998)

Objetivos:

- ***General*:**
  - Conocer las diferentes formas de representar una imagen; espacios de color, formatos de lectura, e indexación de la paleta de color.
- ***Particulares:***
  - Observar los componentes que conforman una imagen en espacio RGB, y como varían dependiendo del formato en el que se lea.
  - Analizar las diferencias de los distintos espacios de color respecto a la imagen en RGB.
  - Comprender la forma en que se indexa una imagen.

Metodología – Resultados- Análisis

En la primer parte de la práctica se eligió una imagen, figura 1-A, a la cual se le realizó un conversión de formato, normalizada y en 8 bits. Para la conversión se utilizó Matlab con el comando de im2double para normalizar, y el comando im2uint8 para obtener la imagen en 8 bits. Los resultados de las transformaciones se pueden observar en las Figuras 1-B y 1-C.
Una vez que se obtuvieron las imágenes con sus formatos respectivos, se obtuvo los valores máximos y mínimos de cada una, los resultados se presentan en la Tabla 1.

Formato	R	G	B
Original	max: 255 min: 0	max: 255 min: 0	max: 255 min: 0
Normalizada	max: 1 min: 0	max: 1 min: 0	max: 1 min: 0
Imagen de 8 bits	max: 255 min: 0	max: 255 min: 0	max: 255 min: 0

Tabla 1. Valores máximos y mínimos de los componentes R, G ,B de cada formato de imagen.

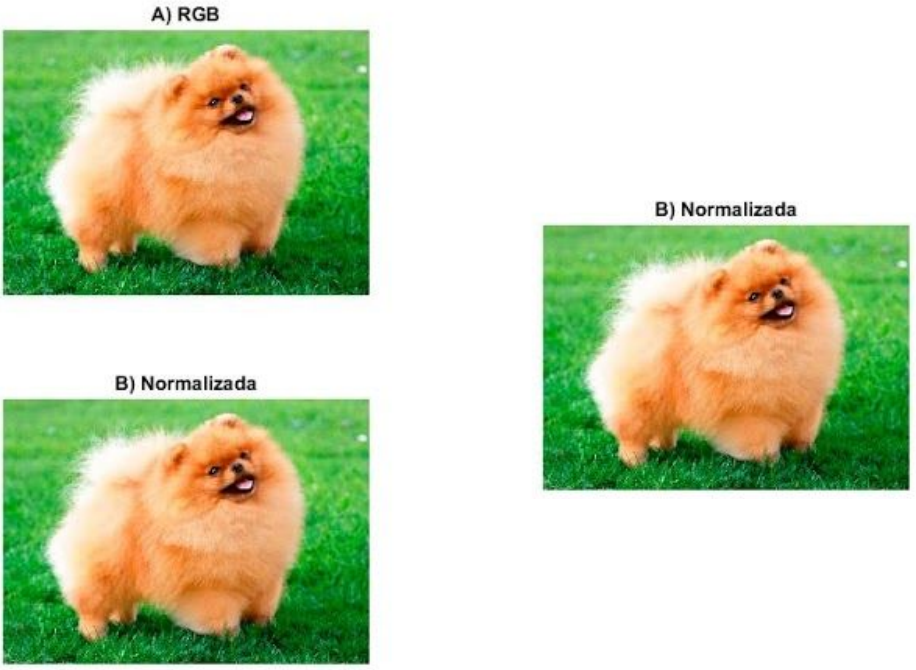

Fig. 1. A) Se presenta la imagen original en formato RGB. B) La imagen original normalizada. C) La imagen original en formato de 8 bits.

Algoritmo: Para leer una imagen y cambiar su formato.

```
Escribir 'Leer imagen'
I = Leer Imagen como RGB
Escribir ' RGB  a Normalizada'
IN = Leer Imagen como double
Escribir 'RGB a 8 Bits'
I_8bits = Leer Imagen como Intager de 8 bits
Escribir 'Encontrar max'
Ancho, Largo = dimensiones de I(:, :, 1)
It = [I, IN, I_8bits]
i = 1
j = 1
i = 1
X = 2
Y = 2
M es un vector
Max es un vector
3_Max es un vector
Repetir por cada IM en It
    Repetir por cada z en rango (1 a 3)
```

```
            A = Leer IM en rangos i, j, z
            B =  Leer IM en rangos i, Y, z
            Repetir hasta que i sea igual a Largo
                Repetir hasta que j sea igual a Ancho o Y sea igual a Ancho
                    Si A > B
                        Y = Y + 1
                        M en posición i = A
                    Si B > A
                        j = j + 1
                        M en posición i = B
            i = i + 1
            Repetir hasta que i sea igual a Largo o X es igual a Largo
                Si M en posición i > M en posición X
                    Max en posición z = M en posición i
                    X = X + 1
                Si M en posición X > M en posición i
                    Max en posición z = M en posición X
                    i = i + 1
        3_Max = Max en posición index de IM
    Repetir por cada IM en It
        Repetir por cada z en rango (1 a 3)
            A = Leer IM en rangos i, j, z
            B =  Leer IM en rangos i, Y, z
            Repetir hasta que i sea igual a Largo
                Repetir hasta que j sea igual a Ancho o Y sea igual a Ancho
                    Si A < B
                        Y = Y + 1
                        M en posición i = A
                    Si B < A
                        j = j + 1
                        M en posición i = B
            i = i + 1
            Repetir hasta que i sea igual a Largo o X es igual a Largo
                Si M en posición i < M en posición X
                    Min en posición z = M en posición i
                    X = X + 1
                Si M en posición X < M en posición i
                    Min en posición z = M en posición X
                    i = i + 1
        3_Min = Min en posición index de IM
    Escribir 3_Min
```

Fin del algoritmo

Análisis de la imagen:
Entre cada formato de imagen se puede ver que hay diferencias claras tales como que 1-C se observa más iluminada que 1-A, de igual forma 1-B se ve más iluminada que 1-A pero no más que 1-C pero está a diferencia de las otras dos tiene como máximo 1 donde 1-A y 1-C tiene como máximo 255 lo cual nos dice que las intensidades en cada pixel fueron de manera proporcional, en un rango de 0 a 1, donde 1 es proporcional a 255.

En la segunda parte de la práctica se realizaron conversiones de la imagen original a distintos espacios de color, los resultados se pueden observar en la figura 2. Y el procedimiento que se realizó se puede observar en el pseudo código, es necesario mencionar que para la transformación a YIQ se utilizó, en Matlab, el comando rgb2ntsc, y que para el espacio YCbCr se utilizó el comando rgb2cbcr.

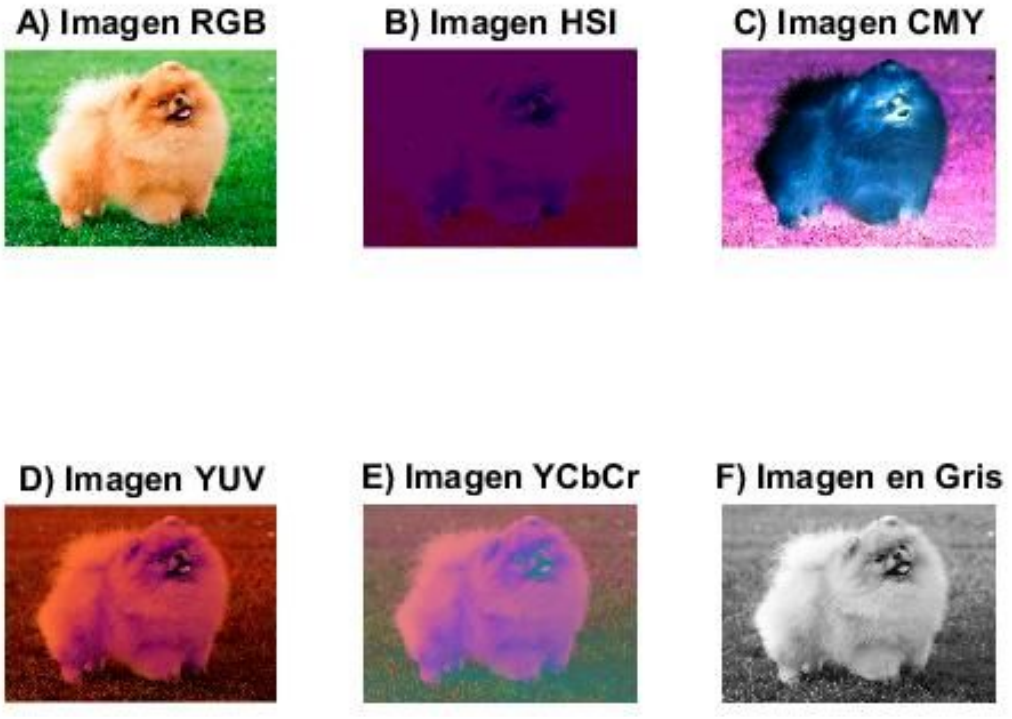

Fig. 2. A) Imagen original en RGB. B) La imagen en el espacio HSI. C) La imagen en el espacio CMY. D) La imagen en el espacio YUV. E) La imagen en el espacio YCbCr. F) La imagen en escala de grises.

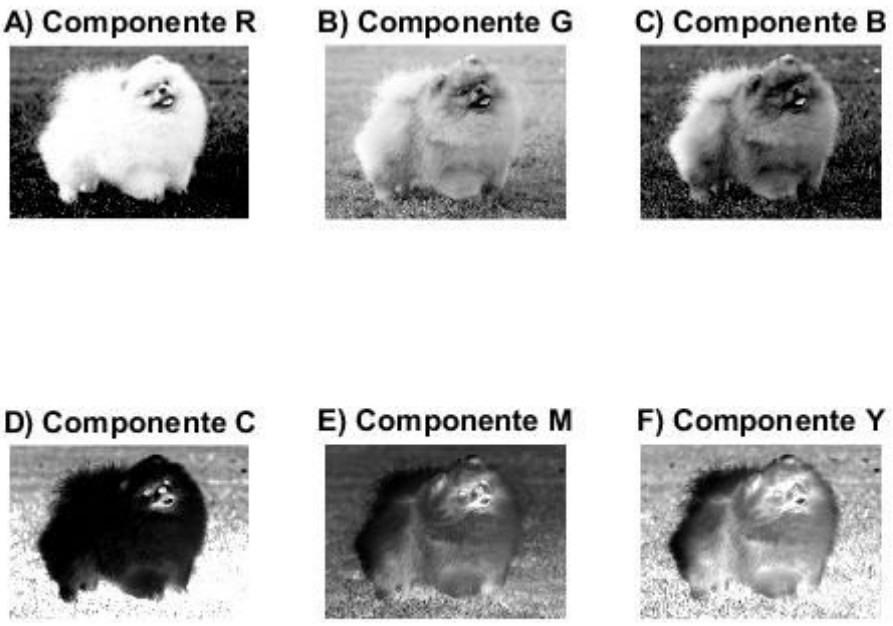

Figura 3. Se presentan los componentes de la imagen original: A) Componente Rojo, B) Componente Verde, C) Componente Azul, D) Componente Cian, E) Componente Magenta, y F) Componente Amarillo.

Algoritmo: Transformación de espacios de color

```
Escribir 'Leer imagen'
I = Leer Imagen como RGB
R  = Leer RGB en posición (:, :, 1)
G  = Leer RGB en posición (:, :, 2)
B  = Leer RGB en posición (:, :, 3)
IN = Leer Imagen como double
Escribir 'Transformar a espacio CMY'
C = 1 - IN(:, :, 1)
M = 1 - IN(:, :, 2)
Y = 1 - IN(:, :, 3)
Concatenar C, M, Y como CMY
Escribir 'Transformar a espacio YIQ'
Convertir I a NTSC como YIQ
Escribir 'Transformar a espacio YUV'
Y = 0.299*R + 0.587*G + 0.114*B
U = - 0.146*R - 0.289*G + 0.425*B
V = 0.615*R - 0.515*G + 0.1*B
Concatenar Y, U, V como YUV
Escribir 'Transformar a espacio YCbCr'
Convertir I a YCbCr como YCbCr
Escribir 'Transformar a espacio HSI'
In = (R + G + B) /3
```

```
a = 0.5*(2*R-G-B)
b = sqrt(3)/2*(G-B)
i = 1, j = 1
Repetir por cada z en rango (1 a 3)
    A = Leer IM en rangos i, j, z
    B =  Leer IM en rangos i, Y, z
    Repetir hasta que i sea igual a Largo
        Repetir hasta que j sea igual a Ancho o Y sea igual a Ancho
            Si A < B
                Y = Y + 1
                M en posición i = A
            Si B < A
                j = j + 1
                M en posición i = B
        i = i + 1n
        Repetir hasta que i sea igual a Largo o X es igual a Largo
            Si M en posición i < M en posición X
                Min en posición z = M en posición i
                X = X + 1
            Si M en posición X < M en posición i
                Min en posición z = M en posición X
                i = i + 1
nz = 1 donde I2 no es igual a 0
S en función de nz = 1 - (Min en función de nz)/(In en función de nz)
S en función de nz negado = 0
H en función de nz = arctan((b en función de nz)/(a en función de
nz))*(180/pi)
H en función de H < 0 = (H en función de H < 0) + 360
H en función nz negado = 0
Concatenar H, S, In como HSI
```

Fin del algoritmo

Análisis de la imagen: El cambio entre cada imagen es evidente. La imagen en el espacio HSI, figura 2-B, se tornó de un color más violeta, de forma homogénea, con toques de magenta y se volvió más difícil distinguir los bordes del perro, por lo que perdió contraste.

En el espacio CMY, figura 2-C, se puede observar la imagen original en negativo, más apreciable en la figura 3, esto se puede inferir debido a que lo que anteriormente era blanco se tornó negro, ya que este espacio actúa de forma sustractiva. También cabe mencionar que ciertos rasgos ahora son visiblemente más fáciles de percibir, en el pelaje.

En la figura 2-D, representando el espacio YUV, se observa la imagen original más apegada a un color rojo, tiene una tonalidad más calurosa, pero con ciertos toques de magenta en las sombras o tonalidades que originalmente eran más obscuras.
En la figura que representa el espacio YCbCr, 2-E, es parecida a la imagen en formato YUV pero con un offset agregado, ya que esta tiene una paleta de color similar pero con una intensidad mayor.
Y la imagen en escala de grises, representando el componente Y del formato YUV, visible en la figura 2-F, tiene más notorios los rasgos del pelaje al igual que se puede observar el contraste entre el fondo y el perro.

Por último se utilizó nuevamente la imagen patrón de la figura, 1-A, para realizar una conversión por indexación de la imagen a una paleta de color. Para ello se utilizó el comando de rgb2ind en Matlab, el cual recibe una imagen y la cantidad de bits con el que se generará la indexación, y regresa la imagen indexada y el mapa de color que le corresponde. La 5 imágenes indexadas se presentan en la figura 4. Y en la figura 5 se ve la imagen original con distintos mapas de colores.

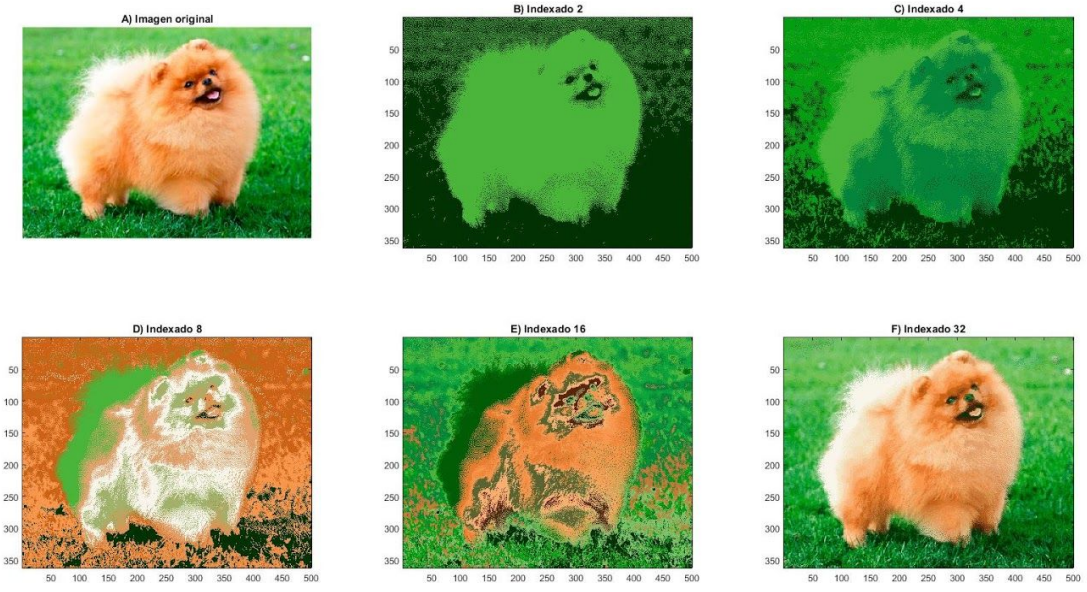

Fig. 4. Se presentan la A) imagen original y su cambio dependiendo del valor indexado: B) Indexado de 2 bits, C) Indexado de 4 bits, D) Indexado de 8 bits, E) Indexado de 16 bits, F) Indexado de 32 bits.

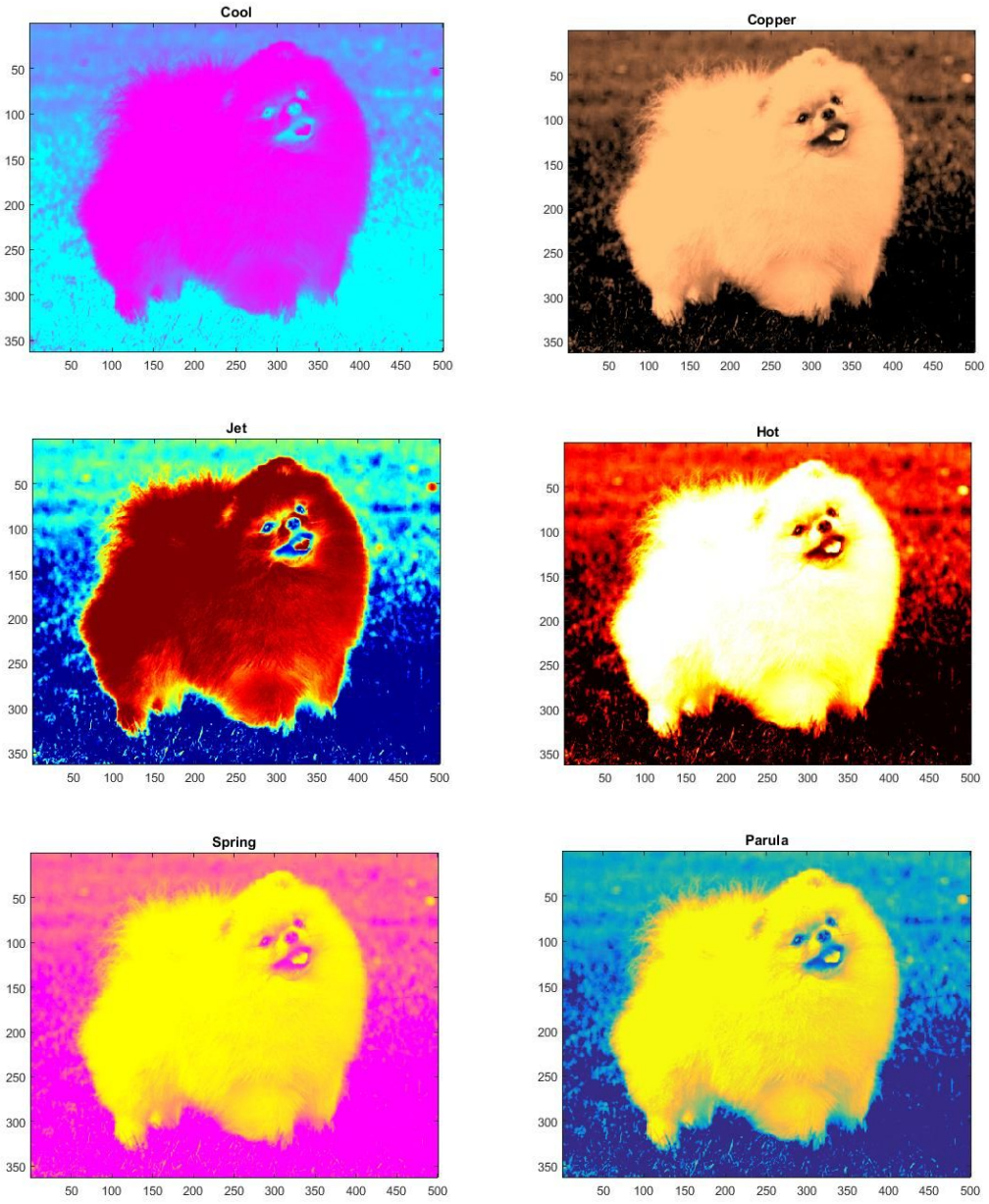

Figura. 5. La imagen original vista en distintos mapas de color. Paleta de color A) Fría, B) Cobre, C) Jet, D) Cálido, E) Primavera, F) Parula.

Algoritmo: Indexado y representación en mapa de color

```
Escribir 'Indexado'
I =Leer Imagen como RGB
Input 'Número de bits para indexar (debe ser un valor de 2^n) ' en Bits
Ind = I./Bits
cmap = colormap(Bits)
desplegar Ind en base a cmap

Escribir 'Cambiar mapa de color'
Input 'Número de bits para indexar (debe ser un valor de 2^n) ' en Bits
Ind = I./Bits
// cmap = colormap('Nombre del mapa de color') ejemplo
cmap = colormap('cool')

desplegar Ind en base a cmap
```

Fin del algoritmo

Análisis de la imagen: Es fácil de notar que en los distintos indexados se pierde cierta cantidad de información, inclusive en el indexado de 32 bits, figura 4-F, que a simple vista parece ser el más apegado a la imagen original, figura 4-A.
Comenzando por la figura 4-B, se perdió el contraste entre la cabeza y el cuerpo del perro, al igual que ya no es visible el peleja como tal, ya sea hebras o distintos colores del mismo, esto es debido a que ahora la imagen consta de 2 índices que dictan que sólo pueden a ver 2 colores dándole una tonalidad verde.
En la figura 4-C se sigue observando una tendencia a verde pero ahora se pueden distinguir sombras en el perro, ya que se han aumentando el número de índices sobre el cual se puede desarrollar la paleta, aún es difícil distinguir el fondo y la cola del perro.
En la figura 4-D, se ve la aparición de un nuevo color en el fondo; este color se asemeja al color del pelaje del perro en la imagen original, pero no se ve exclusivamente en el pelo del perro si no también en el fondo volviendo a un más difícil distinguir entre la silueta del perro y el fondo.
En el indexado de 16 bits, figura 4-E, se perdió el rostro al igual que parte del cuerpo del perro, no se puede comprender visualmente lo que se está observando pero los colores empiezan a asemejarse más a los colores originales aún se encuentran matices de naranja en el fondo pero predomina el verde.
Y por último la imagen indexada por 32 bits, figura 4-F, se aprecia casi en su totalidad el perro, los ojos y la nariz tienen tonalidades de verde todavía, y si se hace un acercamiento a la imagen se puede observar que los bordes se ven con la definición como en la figura 4-A, se ven borrosos.

De igual forma se puede apreciar que distintos mapas de colores pueden afectar en lo que se puede observar de la imagen, se puede perder ciertas texturas como sucede en el peleja en las figuras 5-A),5- B) y 5-D). O bien se puede realzar la diferencia entre el fondo y el perro como en la figura 5-B).

Conclusión

Es necesario conocer las herramientas con las que se cuenta para poder de mejor manera analizar la información que se captura en una imagen, pero también es realizarlo sólo si es requerido ya que es posible perder información en el proceso como se apreció en la última fase de la práctica. Las acciones que se tomen dependen de lo que se quiera observar.

Referencias:

- Adrian Ford, Alan Roberts. (1998). Colour Space Conversion. 25 de enero de 2018, de Poynton Sitio web: https://poynton.ca/PDFs/coloureq.pdf.
- Keith Jack. (2009). Communications Engineering Desk Reference. USA: Elsevier.