Visión Por Computadora I - TPI

Abril 2024

FABIÁN ANDRÉS URIBE GUERRA
Estudiante

.UBAfiuba
FACULTAD DE INGENIERÍA

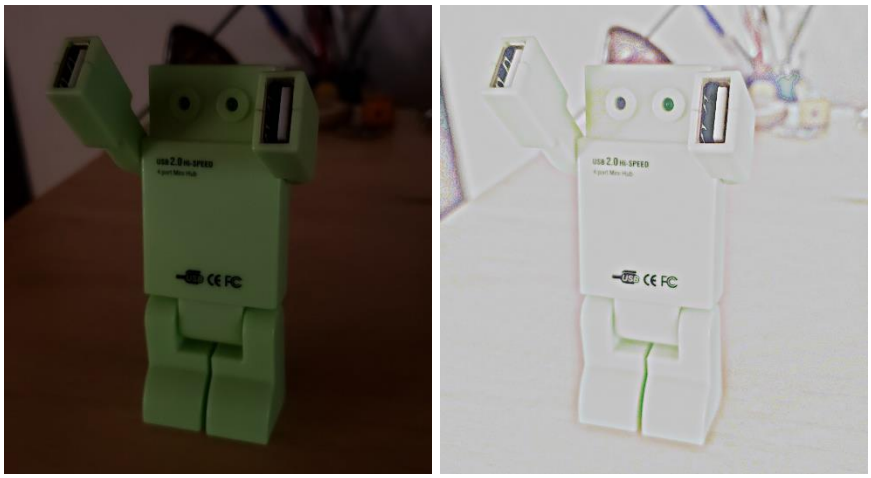
FABIAN URIBE GUERRA

▪ **Parte 1.1**

Implementar el algoritmo de pasaje a coordenadas cromáticas para librarnos de las variaciones de contraste.

1.1.1 CoordCrom_1.png:

1.1.2 CoordCrom_3.png:



1.1.3 CoordCrom_2.png:

FABIAN URIBE GUERRA

▪ **Parte 1.2**

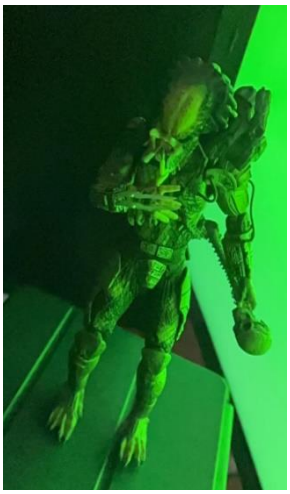
Implementar el algoritmo ***White Patch*** para librarnos de las diferencias de color de iluminación.

1.2.1 wp_green2.jpg:


▪ **Resultados (wp_green2)**

El resultado luego de aplicar el algoritmo arroja una imagen con mayor contraste. Sin embargo, la sobreexposición al verde es demasiado evidente, el modelo no pudo detectar correctamente la zona blanca y no se percibe una mayor diferencia respecto a la original por lo que se infiere que la calibración de colores no fue correcta.

1.2.2 wp_green.png:




▪ **Resultados (wp_green)**

El resultado luego de aplicar el algoritmo arroja una imagen con mayor contraste, mejor diferenciación de blancos y la imagen se ve más neutral. El alíen ha tenido un rebalanceo de azul y rojo mucho más efectivo y en base a la original se puede inferir que le fue más fácil al algoritmo identificar el área más brillante de la imagen.

1.2.3. wp_red.png:


▪ **Resultados (wp_red)**

El resultado luego de aplicar el algoritmo arroja una imagen con mayor contraste y nitidez. Aparentemente al algoritmo le ha resultado más conveniente adaptar el azul y verde en la imagen tomando el fondo rojo. Al igual que la anterior imagen se observó una mejor diferenciación de blancos y la imagen se ve más neutral.

FABIAN URIBE GUERRA

Parte 1.2
Implementar el algoritmo White Patch para librarnos de las diferencias de color de iluminación.

1.2.4 test_red.png

▪ **Resultados (test_red)**

El resultado luego de aplicar el algoritmo arroja unas imágenes con las manos con un color un poco más oscuro al esperado. Es de esperar ya que se ha agregado más azul y verde para equilibrar. Sin embargo, considero que el resultado es el esperado ya que se aprecia de mejor forma el color de la piel de las manos.

1.2.5 test_green.png:

▪ **Resultados (test_green)**

El resultado luego de aplicar el algoritmo arroja unas imágenes con las manos con un color un poco más claro que el anterior. Al tener que compensar con rojo y azul entiendo que el modelo ha logrado arrojar un mejor resultado. (Mejor resultado)

1.2.6 test_blue.png:

▪ **Resultados (test_blue)**

El resultado luego de aplicar el algoritmo arroja unas imágenes con las manos con un color más rojizo, lo que permite inferir que el algoritmo ha rebalanceado en mayor proporción el rojo por encima del verde y el azul. Tener en cuenta que el brillo de la imagen original es menor y el blanco es más complejo de detectar comparado con la imagen roja y verde.

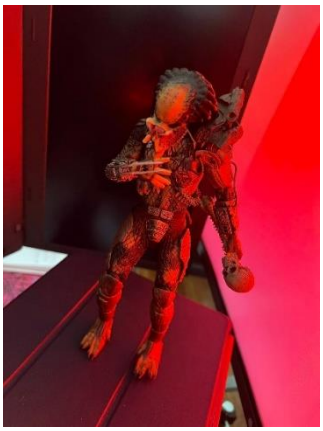
▪ Parte 1.2
Implementar el algoritmo White Patch para librarnos de las diferencias de color de iluminación.

1.2.7 wp_blue.jpg

▪ Resultados (wp_blue)

El resultado luego de aplicar el algoritmo arroja una imagen sin mayores cambios. Al igual que con el ejercicio de las manos, con el azul le es más complejo encontrar un punto blanco de referencia y el rebalanceo de rojo y verde no tiene demasiado impacto en el resultado final.

1.2.8 wp_red2.jpg



▪ Resultados (wp_red2)

En comparación con la anterior imagen roja, al esta haber sido tomada a mayor distancia el balanceo de RGB para mejorar la imagen no tiene un mismo impacto comparado con la imagen anterior. Sin embargo, al algoritmo le favorece las características de brillo y blancos en esta imagen.

FABIAN URIBE GUERRA

▪ **Parte 2.1**

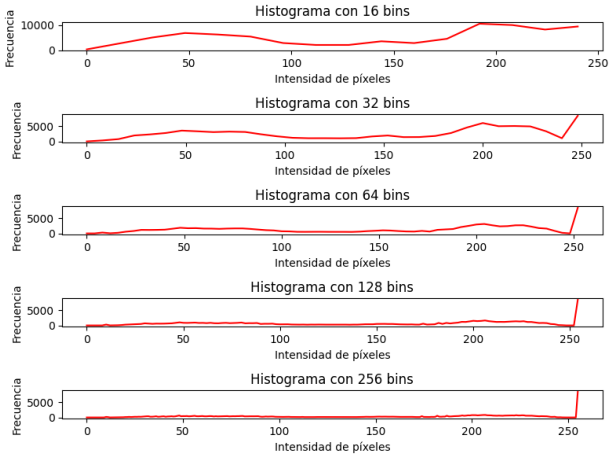
Para las imágenes img1_tp.png y img2_tp.png leerlas con OpenCV en escala de grisas y visualizarlas.

2.1.1 img1_tp.png:

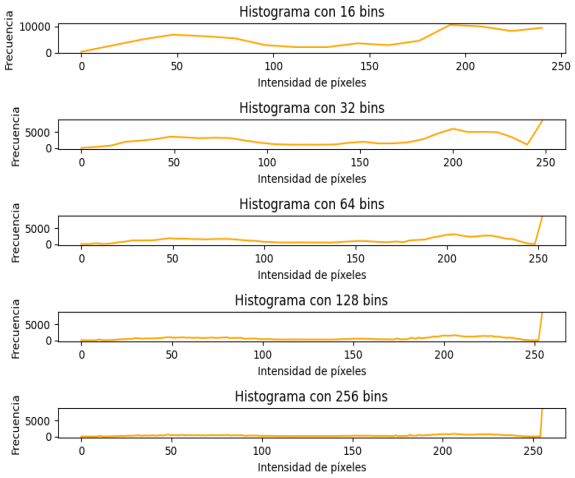
2.1.2 img2_tp.png:

Elija el número de bins que crea conveniente y grafique su histograma, compare los histogramas entre sí. Explicar lo que se observa, si tuviera que entrenar un modelo de clasificación/detección de imágenes, considera que puede ser de utilidad tomar como 'features' a los histogramas?

2.1.1 img1_tp.png:



2.1.2 img2_tp.png:



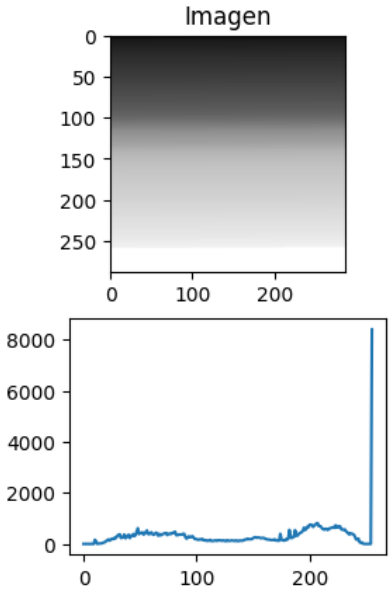
En base a los resultados podemos inferir que el histograma de la img1_tp.png tiene una distribución más uniforme en el rango de tonos lo cual es esperado debido a que es una imagen con gradaciones homogéneas. Por su parte, el histograma de la img2_tp muestra una mayor concentración en los colores más claros. Lo cual tiene sentido al considerar que la imagen es una flor cuyos pétalos son blancos. Ahora, respecto a la pregunta si considero que es de utilidad tomar como 'features' a los histogramas puedo comentar que si ya que son útiles para inferir la distribución y frecuencia de los tonos presentes en la imagen lo cual permite diferenciar el tipo de imagen y es de utilidad para mejorar la precisión de los modelos de clasificación que se pretendan usar. A continuación se muestra a mas detalle los resultados:

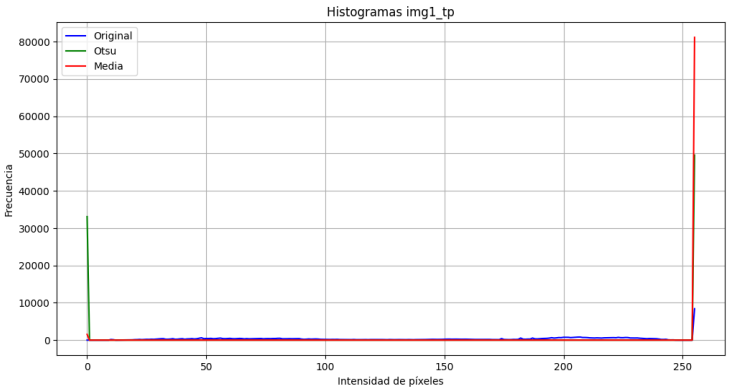
FABIAN URIBE GUERRA

▪ **Parte 2.1**

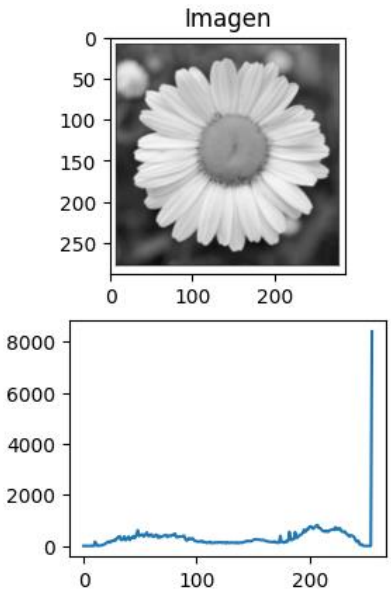
Elija el número de bins que crea conveniente y grafique su histograma, compare los histogramas entre sí. Explicar lo que se observa, si tuviera que entrenar un modelo de clasificación/detección de imágenes, considera que puede ser de utilidad tomar como 'features' a los histogramas?

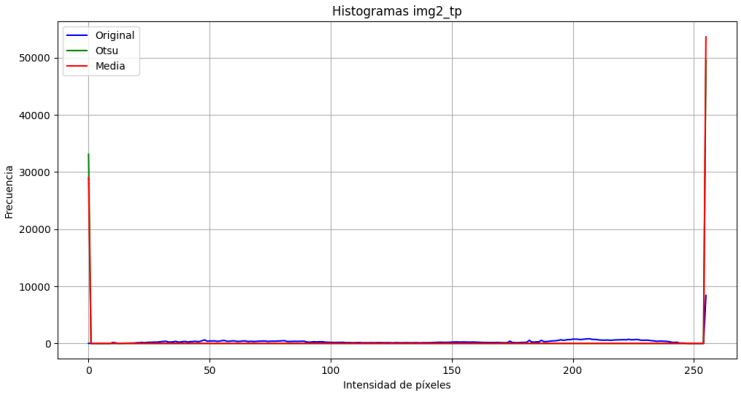
2.1.1 img1_tp.png:





2.1.2 img2_tp.png:

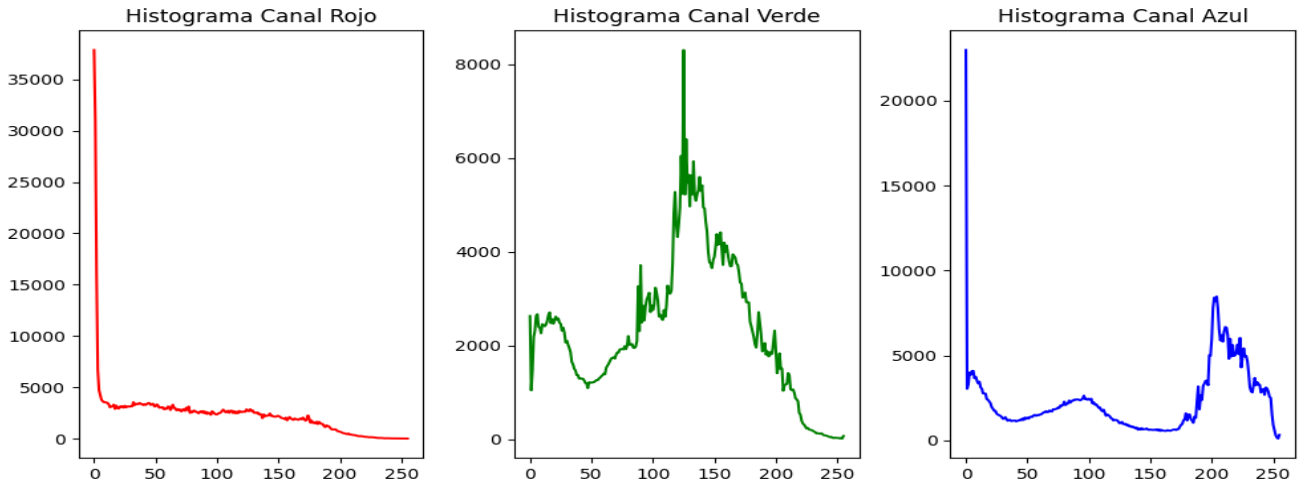


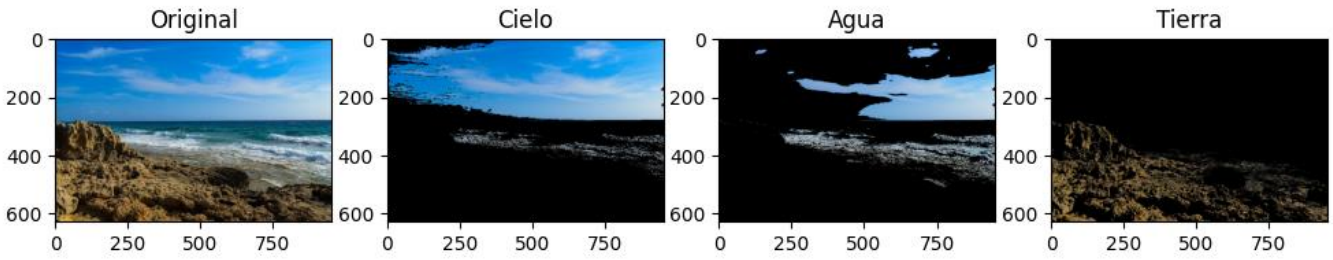


FABIAN URIBE GUERRA

▪ **Parte 3.1**

Para la imagen segmentacion.png analice el histograma de los canales RGB. Segmente algunos de los elementos presentes en la imagen (agua, cielo, tierra) y muestre, aplicando máscaras, las regiones en imágenes separadas





Para poder desarrollar este punto se extrajo el histograma de cada canal (RGB) y se encontró que el canal rojo presenta una distribución de intensidad más baja a lo largo de todo el rango por que se podría inferir que no hay una presencia dominante de tonos rojos en la imagen. Por su parte, el canal verde mantiene una forma gaussiana con picos en el rango medio lo que sugiere una mayor cantidad de verde en la imagen. Finalmente, el canal azul tiene unos '*spikes*' en el lado derecho lo cual infiere la presencia de áreas de cielo y del mar.

Umbrales de la máscara:

```
# Rango de azules (agua)
lower_blue = np.array([100, 150, 150])  # Ajuste para tonos claros de azul
upper_blue = np.array([200, 250, 255])  # Blanco para capturar espuma de mar

# Rango de verdes (cielo)
lower_green = np.array([0, 50, 200])
upper_green = np.array([200, 250, 255])

# Rango de marrones (tierra)
lower_brown = np.array([100, 80, 0])     # Ajuste para arena
upper_brown = np.array([200, 180, 100])  # Ajuste para tonos más oscuros de marrón

# Aplicar mascaras para segmentar los elementos
mask_blue = cv2.inRange(img_rgb, lower_blue, upper_blue)
mask_green = cv2.inRange(img_rgb, lower_green, upper_green)
mask_brown = cv2.inRange(img_rgb, lower_brown, upper_brown)

# Aplicar las mascaras a la imagen original
segmented_blue = cv2.bitwise_and(img_rgb, img_rgb, mask=mask_blue)
segmented_green = cv2.bitwise_and(img_rgb, img_rgb, mask=mask_green)
segmented_brown = cv2.bitwise_and(img_rgb, img_rgb, mask=mask_brown)
```