Universidad Autónoma de Yucatán

Maestría en Ciencias de la Computación

Visión Computacional

TCP 1

Autor: Mario Herrera Almira

28 de marzo del 2023

Descripción de lo hecho:

En este trabajo se realizó una clasificación en cascada de una imagen en formato CIE-Lab, operando solamente sobre las componentes “a” y “b” de cada pixel. Para el componente de luminosidad se utilizaron los valores sin alterar de la imagen original.

Para empezar este ejercicio lo primero que se hace es cargar la imagen en formato BGR, dividir cada pixel entre 255 para obtener valores entre 0 y 1 y luego convertir la imagen a formato CIE-Lab. Adicionalmente se crear una matriz (llamada “clases”) que tiene el mismo tamaño que la matriz que representa a la imagen. Esta se utiliza para llevar el control de a qué clase pertenece cada pixel. Inicialmente esta matriz se inicializa aleatoriamente con las clases 1 y 2.

Luego se comienza a utilizar los algoritmos de distancia Euclidiana y distancia de Mahalanobis para reacomodar los pixeles y posicionarlos en la clase que menor distancia arroje. Inicialmente se deja correr el algoritmo con la distancia Euclidiana hasta que converja y luego se utiliza la distancia de Mahalanobis. Para poder calcular estas distancias primero se calcula la media y la covariancia de cada clase, esto se lleva a cabo mediante una función proporcionada por el profesor que recibe la imagen en formato CIE-Lab y una máscara que es una matriz del mismo tamaño que la imagen que tiene valores de 1 en los pixeles que se van a analizar para calcular la media y la covarianza, el resto de posiciones en la matriz tienen valor igual a 0.

Al analizar si los pixeles caen en una clase u otra se tiene en cuenta que si todos los pixeles se agrupan en una sola clase significa que ya no se puede seguir subdividiendo esa rama del árbol por lo que se dejan de añadir nuevas clases por esa rama.

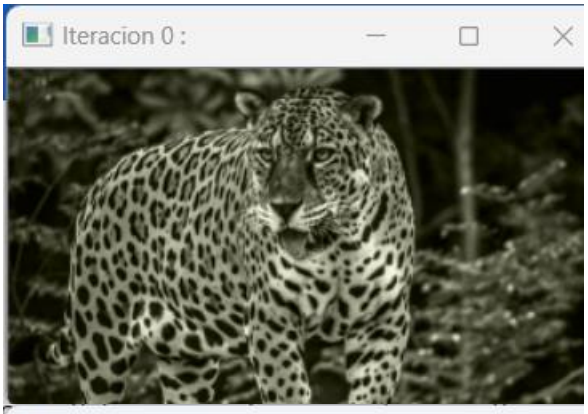
Cada vez que se logra completar un nivel del árbol con éxito se guarda una copia de la clasificación en cascada hasta el momento para poder mostrar al final como la imagen se va pareciendo poco a poco a la imagen original. Luego de realizar la copia se observa cuáles clases se pueden seguir subdividiendo, se generan las etiquetas nuevas para expandir el árbol y se inicializan de manera aleatoria los pixeles dentro de las clases nuevas que le correspondan.

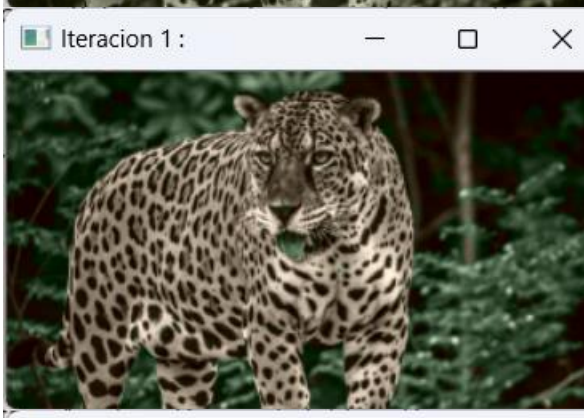
A partir de este punto se repite todo el proceso hasta que ya no sea posible seguir expandiendo ninguna clase, en ese momento se detiene el algoritmo y se muestra en pantalla la secuencia de imágenes que demuestran como la imagen va cambiando en cada iteración, pareciéndose cada vez más a la imagen original.

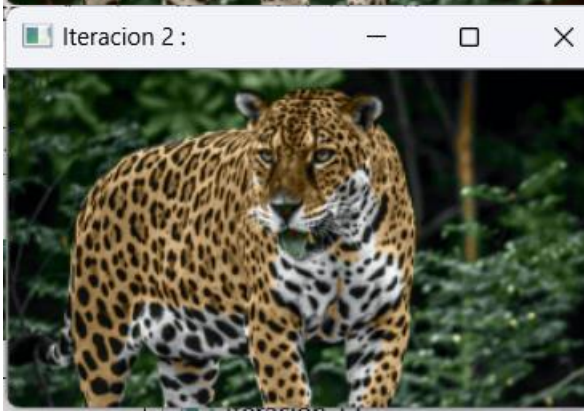
Ejemplo de resultados obtenidos

Original

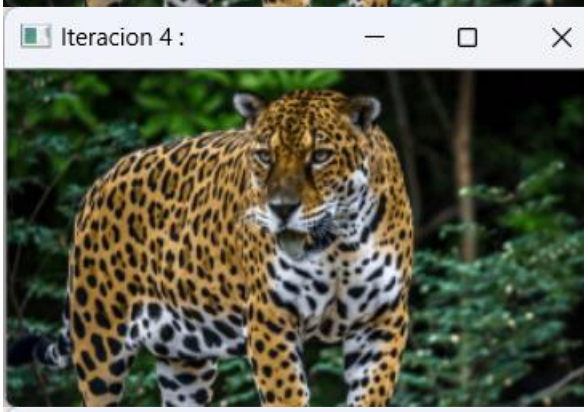
Evolución de la clasificación

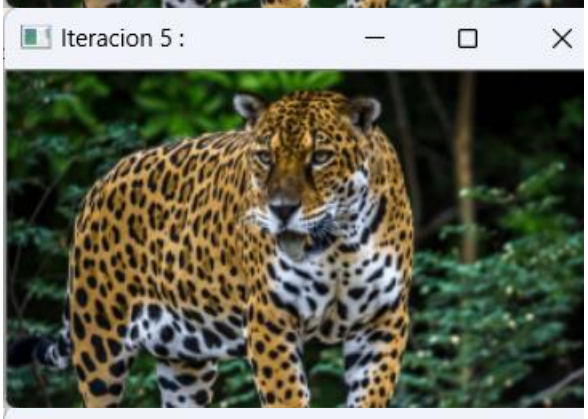


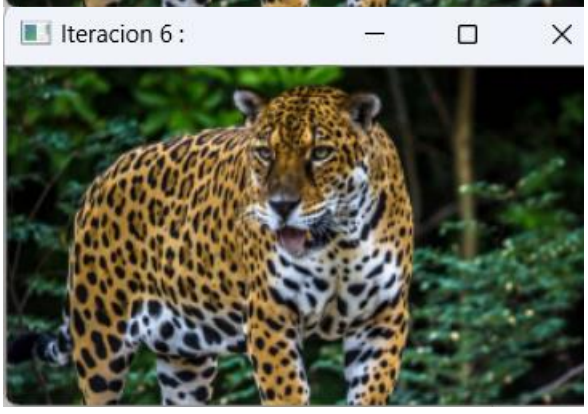





Iteracion 3 :

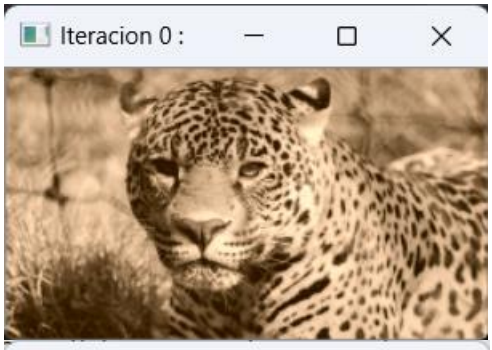
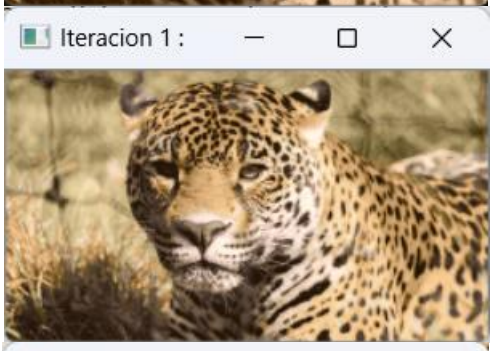
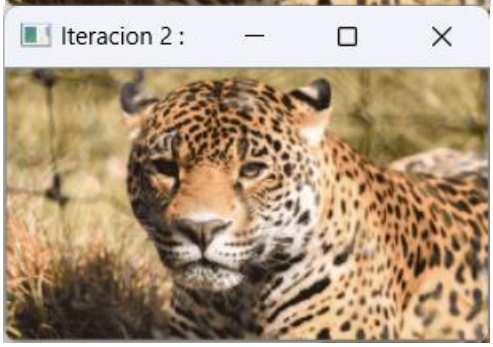

Iteracion 4 :

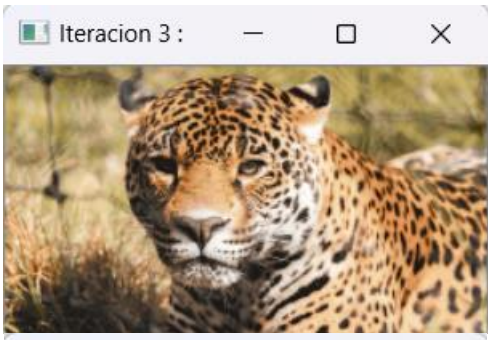

Iteracion 5 :

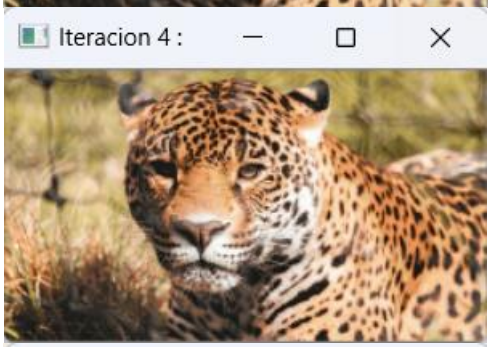

Iteracion 6 :

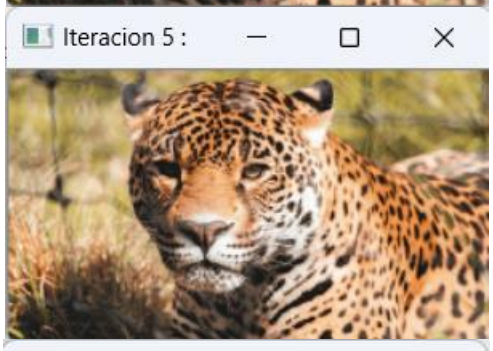
La mayor dificultad que se encontró durante este trabajo fue que luego de que la imagen se encontrara casi idéntica a la original comenzaba a distorsionarse y a poner colores de manera incorrecta. Luego de mucha búsqueda se encontró que el problema estaba en que se calculaban muchas clases que no se deberían calcular porque su media y matriz de covarianza eran iguales a cero. El problema fue detectado y corregido satisfactoriamente.

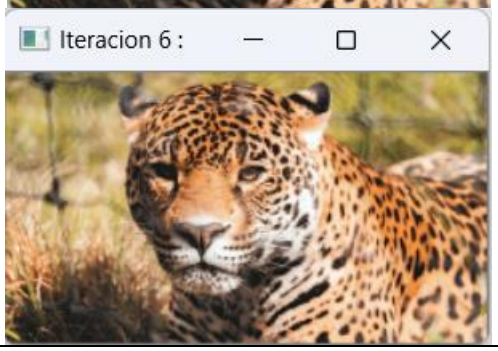
A continuación, se muestran más ejemplos donde el algoritmo logra segmentar correctamente las imágenes:

Original	**Evolución de la clasificación**
	
	
	


Iteracion 3 :

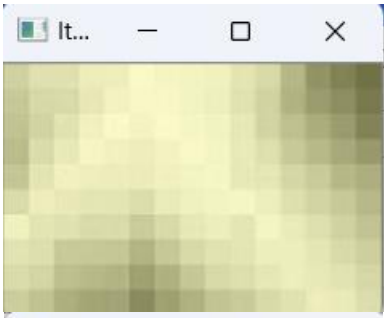

Iteracion 4 :

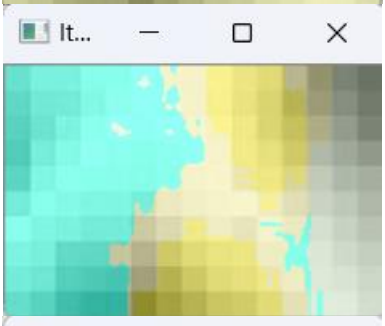

Iteracion 5 :

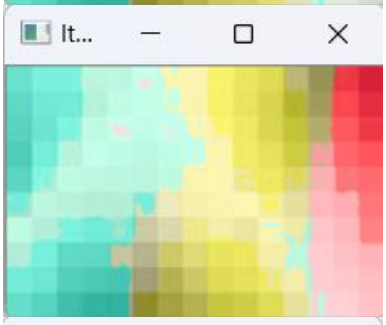

Iteracion 6 :

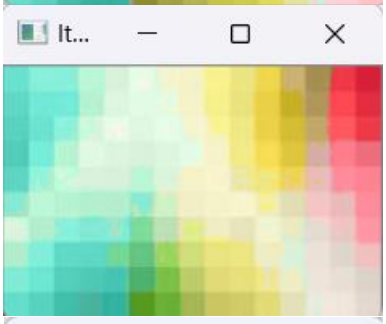
Original

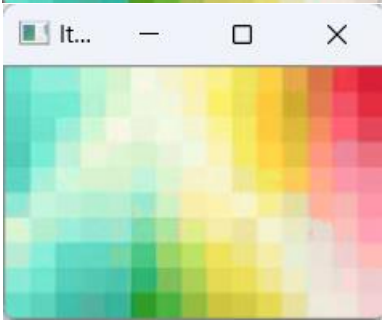
Evolución de la clasificación

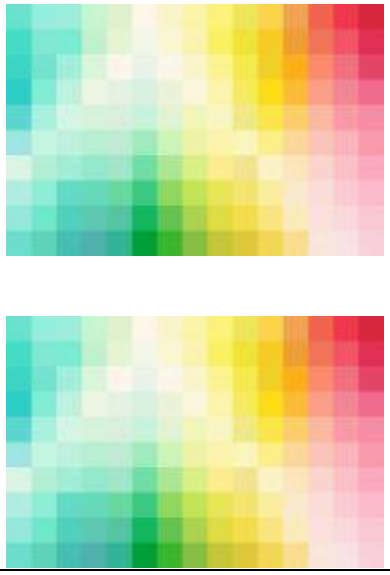


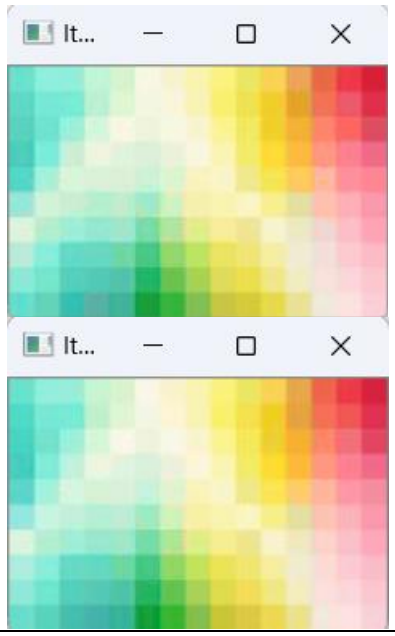












Original

Evolución de la clasificación