



ALGORITMO 3

VISION ROBOTICA
INGENIERIA EN ROBOTICA

Alumno: Carlos Omar Rodriguez Vazquez

Introducción

El procesamiento de imágenes es una disciplina fundamental en el ámbito de la visión por computadora y la robótica, ya que permite extraer información significativa de las imágenes para su posterior análisis y toma de decisiones. En este contexto, el presente reporte tiene como objetivo abordar diversas técnicas y algoritmos para el procesamiento básico de imágenes utilizando Python y la biblioteca OpenCV. Se explorarán conceptos clave como la conversión a escala de grises, la binarización, el cálculo del negativo de una imagen y la rotación de imágenes. Estas técnicas son fundamentales para la implementación de sistemas de visión artificial en aplicaciones robóticas y otros campos relacionados.

Marco Teorico

Conversión escala de grises

Para realizar operaciones de procesamiento de imágenes en Python utilizando la biblioteca OpenCV, es esencial comprender varios conceptos fundamentales. Uno de estos conceptos es la conversión de una imagen a escala de grises. Esto se logra mediante el uso de la función cv2.cvtColor(), que toma una imagen de entrada (im) y utiliza el flag cv2.COLOR_BGR2GRAY para convertirla a una representación de escala de grises (ig).

```
ig = cv.cvtColor(im, cv.COLOR_BGR2GRAY)
```

Conversión a blanco y negro

Un proceso crucial en el procesamiento de imágenes es la conversión a blanco y negro. Para ello, se utiliza la función cv2.threshold(), que toma la imagen en escala de grises (ig) y un umbral (T) como parámetros. Esta función asigna un valor binario a cada píxel según si su intensidad es mayor o menor que el umbral especificado, resultando en una imagen binaria (ibn) donde los píxeles se clasifican como blanco o negro. Además, se puede realizar una conversión binaria adicional utilizando una operación de asignación de valores. Por ejemplo, el código ib[ig > 100] = 255 asigna el valor 255 a todos los píxeles en la imagen binaria ib3 cuya intensidad en la imagen en escala de grises ig sea mayor que 100.

```
#Metodo OpenCV
(T, ibn) = cv2.threshold(ig, 100, 255, cv2.THRESH_BINARY)

#Otro metodo
ib[ig > umbral] = 255
```

Cálculo del negativo de una imagen

Calcular el negativo de una imagen en escala de grises implica invertir los valores de intensidad de los píxeles. Este proceso se realiza en Python aplicando la operación de sustracción entre el valor máximo de intensidad y el valor de intensidad de cada píxel en la imagen original en escala de grises.

```
ng = 255 - ig
```

Rotación de una imagen

La rotación de una imagen se puede realizar con la función cv2.rotate(), que toma la imagen original y un ángulo como parámetros. Para rotar la imagen 90 grados en sentido de las manecillas del reloj, se utiliza un ángulo positivo; para rotarla en contra de las manecillas del reloj, se utiliza un ángulo negativo; y para una rotación de 180 grados, se utiliza un ángulo de 180. Esta función devuelve la imagen rotada según el ángulo especificado.

```
#Rotar 90 grados en el sentido de las manecillas del reloj.

im_rotated = cv.rotate(im, cv.ROTATE_90_CLOCKWISE)

#Rotar 90 grados al contrario de las manecillas del reloj.

im_rotated = cv.rotate(im, cv.ROTATE_90_COUNTERCLOCKWISE)

#Rotar 180 grados.

im_rotated = cv.rotate(im, cv.ROTATE_180)
```

Ejercicios

- Ejercicio 1

```
im = cv.imread('im1.jpg')
ig = cv.cvtColor(im, cv.COLOR_BGR2GRAY)
ng = 255 - ig

cv.imshow('escala_de_grises', ig)
cv.imshow('negativo', ng)
cv.imshow('original_image', im)
cv.waitKey(0)
```



Figura 1: Imagenes del ejercicio 1. (a) Imagen original (b) Imagen en escala de grises (c) Negativo de la imagen.

- Ejercicio 2

```
im2 = cv.imread('im2.png')
ig2 = cv.cvtColor(im2, cv.COLOR_BGR2GRAY)
ng2 = 255 - ig2

cv.imshow('escala_de_grises', ig2)
cv.imshow('negativo', ng2)
cv.imshow('original_image', im2)
cv.waitKey(0)
```



Figura 2: Imagenes del ejercicio 1. (a) Imagen original (b) Imagen en escala de grises (c) Negativo de la imagen.

- Ejercicio 3

```
im3 = cv.imread('im3.png')
ig3 = cv.cvtColor(im3, cv.COLOR_BGR2GRAY)
ib3 = np.zeros_like(ig3)
ib3[ig3 > 100] = 255

cv.imshow('escala_de_grises', ig3)
cv.imshow('blanco_y_negro', ib3)
cv.imshow('original_image', im3)
cv.waitKey(0)
```

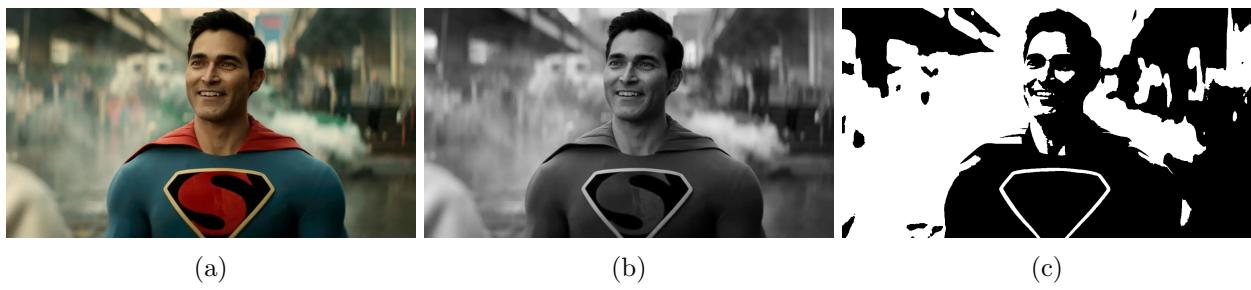


Figura 3: Imagenes del ejercicio 3. (a) Imagen original (b) Imagen en escala de grises (c) Imagen en blando y negro.

- Ejercicio 4

```

im4 = cv.imread('im4.jpg')
ig4 = cv.cvtColor(im4, cv.COLOR_BGR2GRAY)
ib4 = np.zeros_like(ig4)
ib4[ig4 > 100] = 255
ng4 = 255 - ig4

cv.imshow('escala_de_grises', ig4)
cv.imshow('blanco_y_negro', ib4)
cv.imshow('negativo', ng4)
cv.imshow('original_image', im4)
cv.waitKey(0)

```

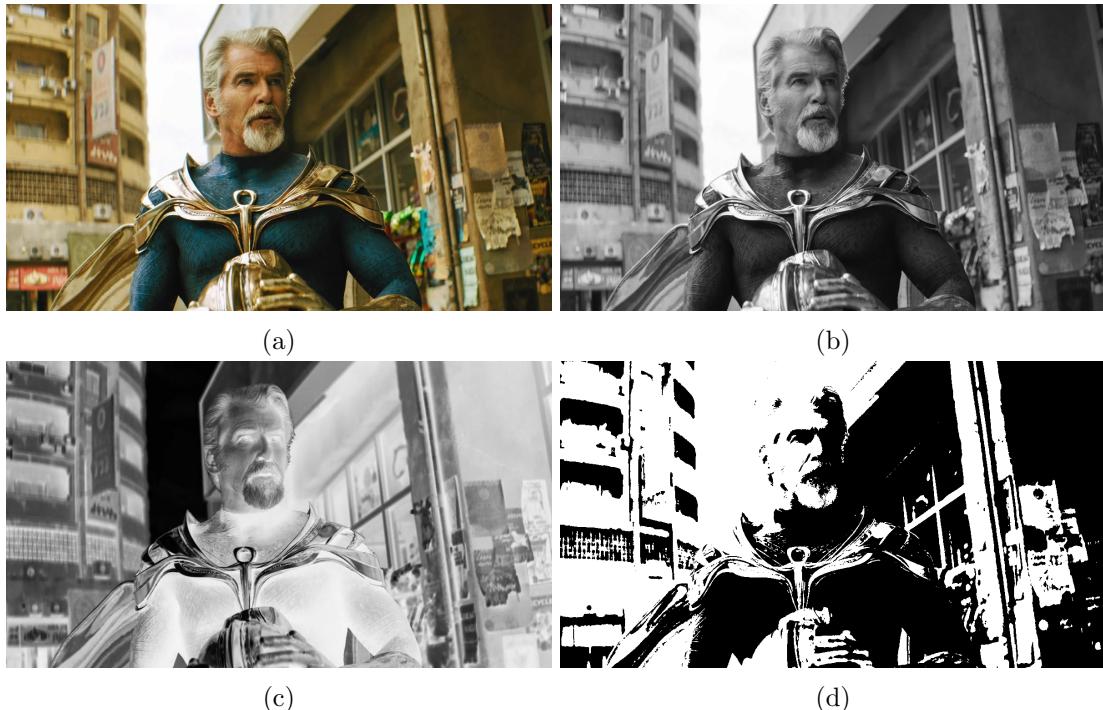


Figura 4: Imagenes del ejercicio 4. (a) Imagen original (b) Imagen en escala de grises (c) Negativo de la imagen (d) Imagen en blando y negro.

- Ejercicio 5

```

im5 = cv.imread('im5.png')
im5R = cv.rotate(im5, cv.ROTATE_90_CLOCKWISE)
ig5 = cv.cvtColor(im5R, cv.COLOR_BGR2GRAY)
ng5 = 255 - ig5

cv.imshow('imagen_rotada', im5R)
cv.imshow('escala_de_grises', ig5)
cv.imshow('blanco_y_negro', ng5)
cv.imshow('original_image', im5)
cv.waitKey(0)

```

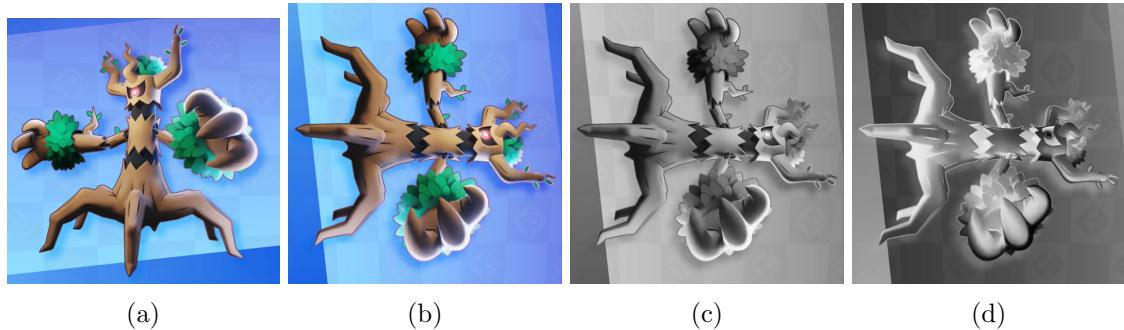


Figura 5: Imagenes del ejercicio 5. (a) Imagen original (b) Imagen rotada 90 grados en sentido de las manecillas del reloj (c) Imagen en escala de grises (d) Imagen en blanco y negro.

- Ejercicio 6

```

im6 = cv.imread('im6.jpg')
im6R = cv.rotate(im6, cv.ROTATE_90_COUNTERCLOCKWISE)
ig6 = cv.cvtColor(im6R, cv.COLOR_BGR2GRAY)
ng6 = 255 - ig6

cv.imshow('imagen_rotada', im6R)
cv.imshow('escala_de_grises', ig6)
cv.imshow('negativo', ng6)
cv.imshow('original_image', im6)
cv.waitKey(0)

```

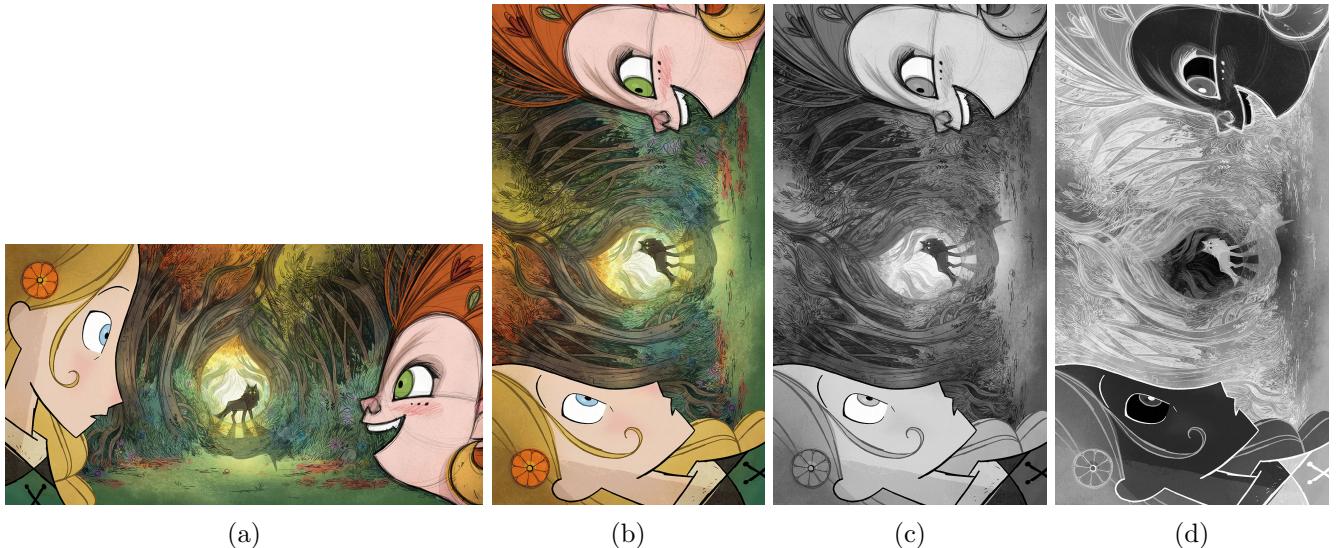


Figura 6: Imagenes del ejercicio 6. (a) Imagen original (b) Imagen rotada 90 grados en contra de las manecillas del reloj (c) Imagen en escala de grises (d) Negativo de la imagen.

- Ejercicio 7

```
im7 = cv.imread('im7.jpeg')
im7R = cv.rotate(im7, cv.ROTATE_180)
ig7 = cv.cvtColor(im7R, cv.COLOR_BGR2GRAY)
ng7 = 255 - ig7

cv.imshow('imagen_rotada', im7R)
cv.imshow('escala_de_grises', ig7)
cv.imshow('negativo', ng7)
cv.imshow('original_image', im7)
cv.waitKey(0)
```

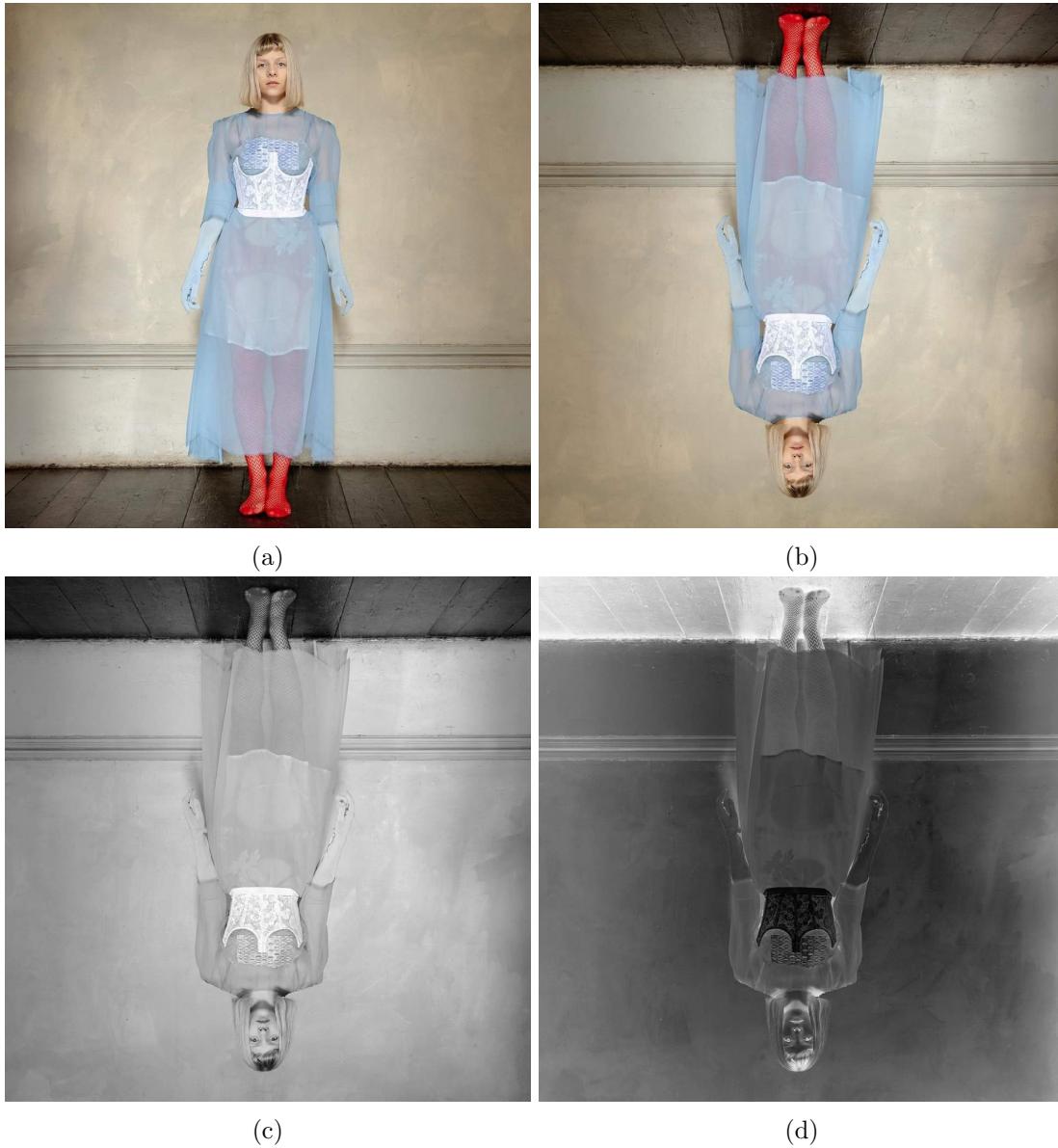


Figura 7: Imagenes del ejercicio 7. (a) Imagen original (b) Imagen rotada 180 grados (c) Imagen en escala de grises (d) Negatigo de la imagen.

Conclusión

En conclusión, el procesamiento de imágenes es una herramienta poderosa en el campo de la visión por computadora y la robótica, que permite analizar y comprender la información visual de manera eficiente. En este reporte, hemos explorado varios conceptos y técnicas fundamentales para el procesamiento básico de imágenes utilizando Python y OpenCV. Desde la conversión a escala de grises hasta la rotación de imágenes, cada técnica desempeña un papel crucial en el análisis y la manipulación de imágenes en aplicaciones prácticas. Continuar explorando y comprendiendo estos conceptos es esencial para desarrollar sistemas de visión artificial más avanzados y eficaces en el futuro.