

# Informe Sesión 1 - Visión por Ordenador

José María Benítez Rodríguez y Antonio Raynel García Bryan

23 de Septiembre 2025

## 1. Introducción

En esta primera sesión del laboratorio de Visión por Ordenador se realizaron diferentes transformaciones básicas sobre imágenes. El objetivo fue familiarizarnos con las operaciones elementales de procesamiento digital, como recorte, cambio a escala de grises, redimensionamiento y rotación, utilizando *OpenCV* en Python.

## 2. Apartado A: Operaciones básicas con imágenes

En este apartado se llevaron a cabo distintas tareas de procesamiento digital de imágenes con el fin de familiarizarnos con las operaciones fundamentales de OpenCV. El flujo de trabajo consistió en:

- **Conversión a escala de grises:** se empleó la función `cv2.cvtColor()` para transformar la imagen desde el espacio de color BGR a escala de grises. Con el atributo `.shape` se verificaron las dimensiones de la imagen y el número de canales antes y después de la conversión.
- **Cambio de tamaño:** utilizando `cv2.resize()` se redimensionó la imagen a un tamaño fijo de  $200 \times 200$  píxeles. Posteriormente se comprobó que el cambio se había aplicado correctamente inspeccionando las nuevas dimensiones.
- **Recorte del centro:** para extraer una región de  $120 \times 160$  píxeles, se calcularon las coordenadas de recorte a partir de las dimensiones originales y del punto central de la imagen. El recorte se realizó mediante operaciones de *slicing* sobre la matriz de píxeles.
- **Rotación:** la rotación de la imagen se implementó combinando `cv2.getRotationMatrix2D()` para calcular la matriz de transformación y `cv2.warpAffine()` para aplicar dicha transformación. Se especificó el centro de la imagen, el ángulo de rotación y un factor de escala igual a 1.0 para mantener el tamaño.

Estas operaciones constituyen la base del procesamiento digital de imágenes, y resultan imprescindibles como paso previo a técnicas más avanzadas de análisis y segmentación.

## 2.1. Imagen original



Figura 1: Imagen original utilizada como entrada.

## 2.2. Recorte (centro de la imagen)



Figura 2: Imagen recortada centrada.

## 2.3. Conversión a escala de grises



Figura 3: Imagen convertida a escala de grises.

## 2.4. Redimensionamiento



Figura 4: Imagen redimensionada.

## 2.5. Rotación

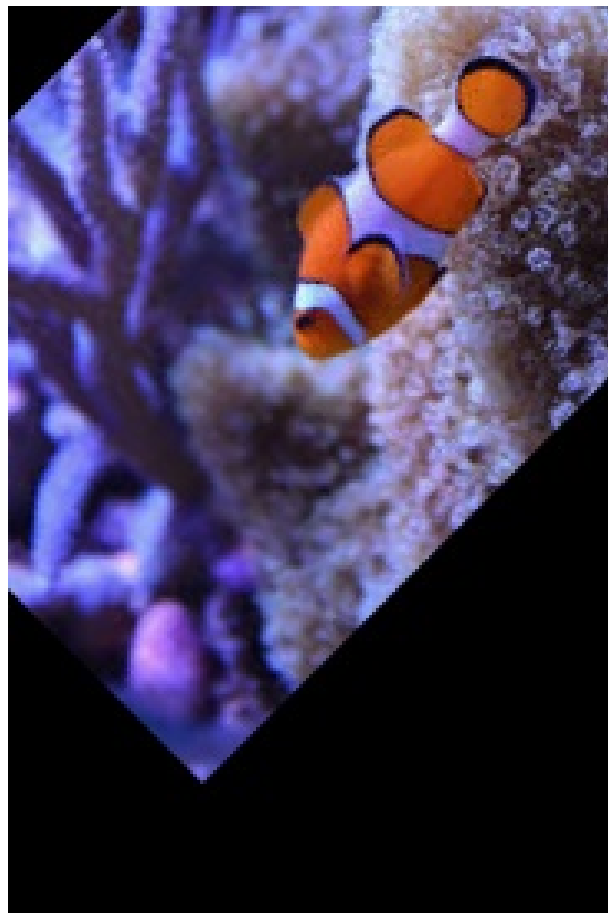


Figura 5: Imagen rotada.

### 3. Conclusiones del apartado A

En este apartado logramos comprender cómo manipular imágenes mediante transformaciones básicas. Estas operaciones son fundamentales para etapas posteriores de la visión por ordenador, donde se aplicarán técnicas más complejas de segmentación, detección y clasificación.

### 4. Apartado B: Segmentación en el espacio HSV

En este apartado se realizó el siguiente procedimiento:

- Se cambió el espacio de color de las imágenes de **BGR** a **HSV**, donde la crominancia e intensidad están separadas. Este cambio puede interpretarse como pasar de un sistema de referencia cartesiano a uno polar, lo que facilita discriminar los colores de interés. Para este paso se utilizó la función `cv2.cvtColor()` de OpenCV. La Figura 4.1 muestra los espacios de color disponibles con los que se trabajará.
- A continuación, se generó una **máscara binaria** para cada imagen. Una máscara es una imagen en blanco y negro en la que los píxeles de interés aparecen en blanco y los que no lo son aparecen en negro. La segmentación se llevó a cabo multiplicando la máscara por la imagen original, obteniendo así únicamente los píxeles de interés.
- Se repitió el mismo proceso para los tonos blancos, ajustando los rangos de valores de **H**, **S** y **V** en la interfaz proporcionada hasta conseguir una buena separación de estas zonas.

### 5. Segmentación por colores en HSV

#### 5.0.1. Conversión a HSV



Figura 6: Imagen convertida al espacio de color HSV.

### 5.0.2. Segmentación del color naranja

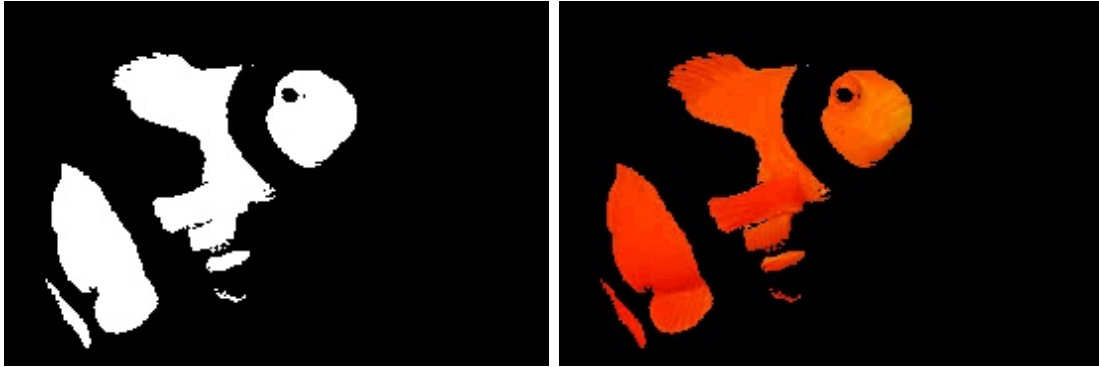


Figura 7: Máscara del color naranja (izquierda) y segmentación resultante (derecha).

### 5.0.3. Segmentación del color blanco

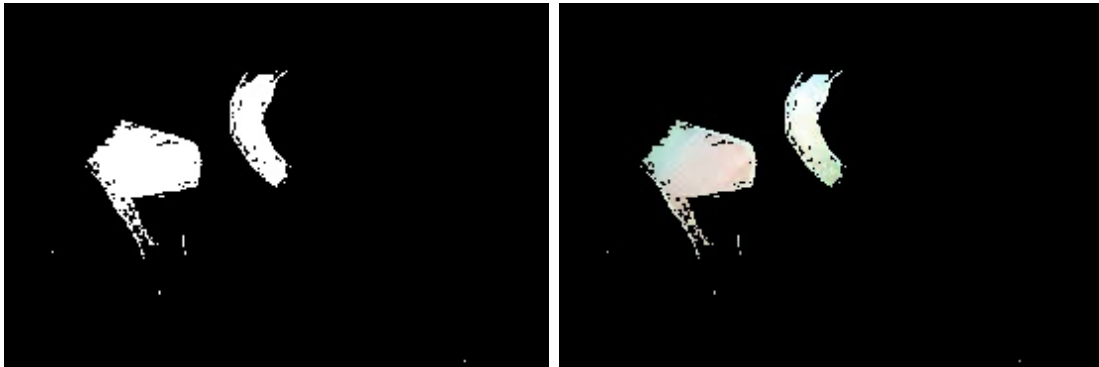


Figura 8: Máscara del color blanco (izquierda) y segmentación correspondiente (derecha).

## 5.1. Segmentación adicional con HSV

Para reforzar el análisis, se realizaron más pruebas de segmentación en distintas imágenes. Nuevamente, se aplicaron umbrales de color en el espacio HSV para obtener máscaras separadas para los tonos naranjas y blancos característicos del pez payaso.

### 5.1.1. Ejemplo con segunda imagen

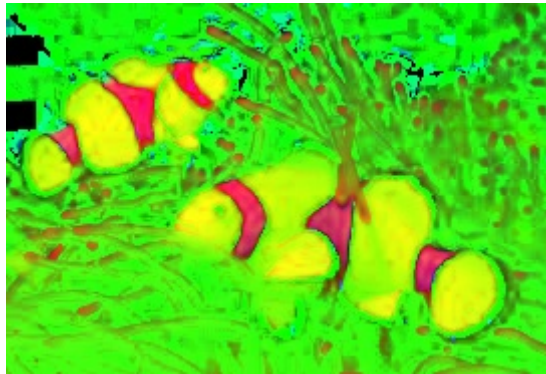


Figura 9: Conversión al espacio HSV de la segunda imagen.

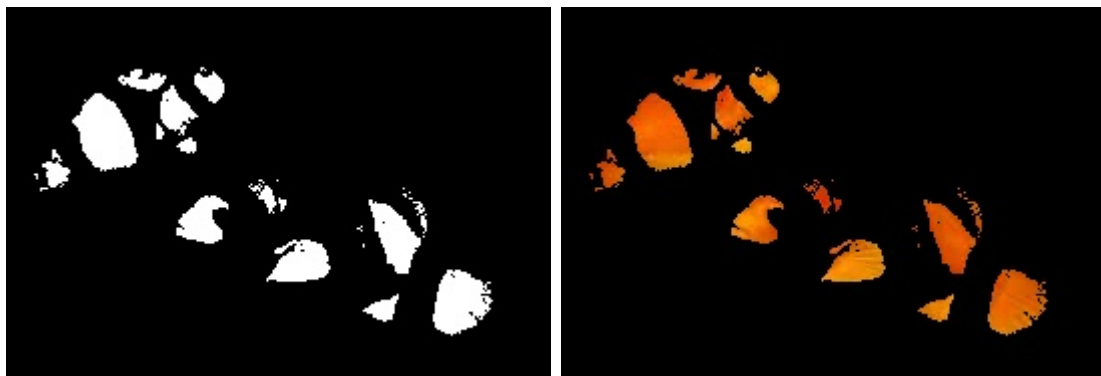


Figura 10: Máscara del color naranja (izquierda) y segmentación del pez (derecha).



Figura 11: Máscara del color blanco (izquierda) y segmentación correspondiente (derecha).

### 5.1.2. Ejemplo con tercera imagen

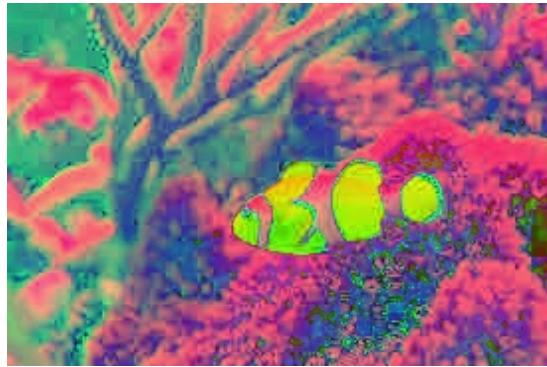


Figura 12: Imagen convertida a HSV.

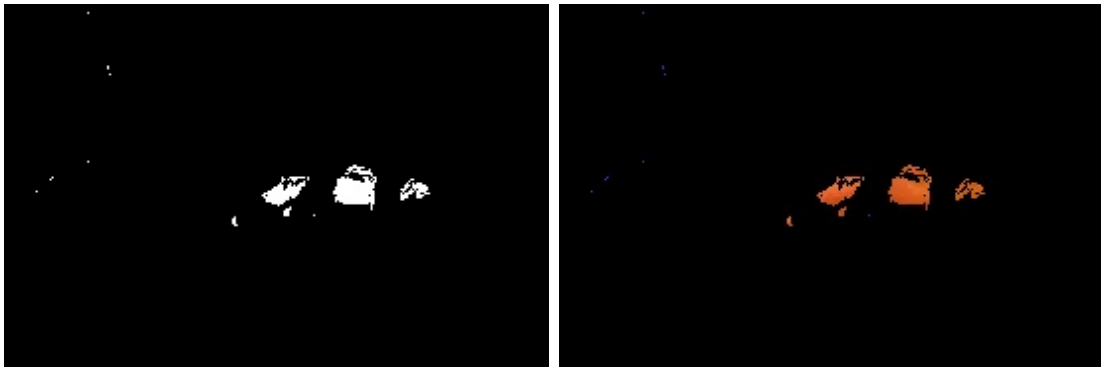


Figura 13: Máscara del color naranja (izquierda) y pez segmentado (derecha).

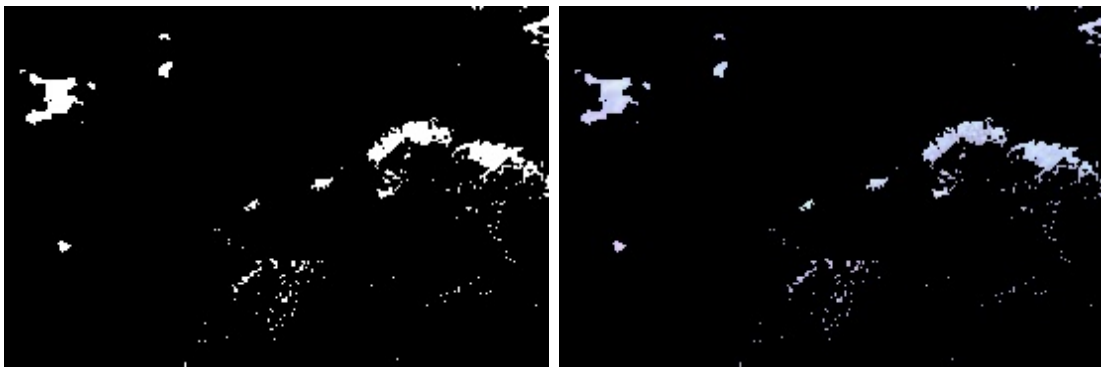


Figura 14: Máscara del color blanco (izquierda) y segmentación final (derecha).

### 5.1.3. Ejemplo con cuarta imagen

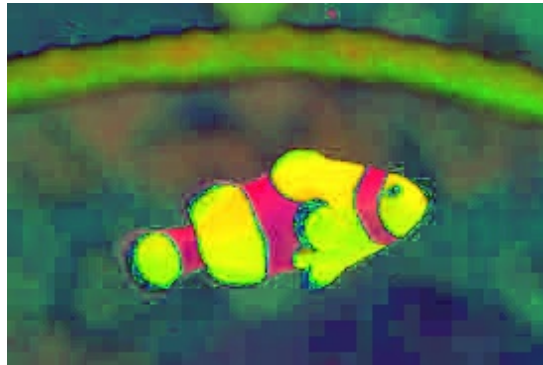


Figura 15: Imagen convertida al espacio HSV.

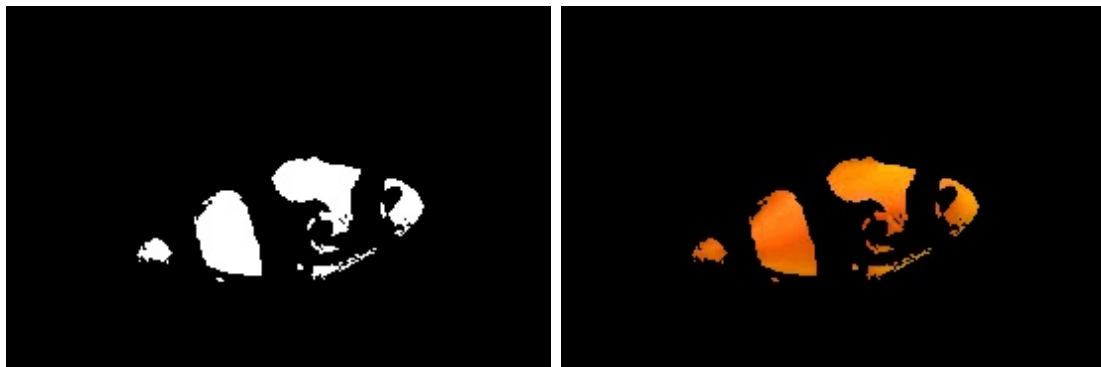


Figura 16: Máscara del color naranja (izquierda) y segmentación del pez (derecha).

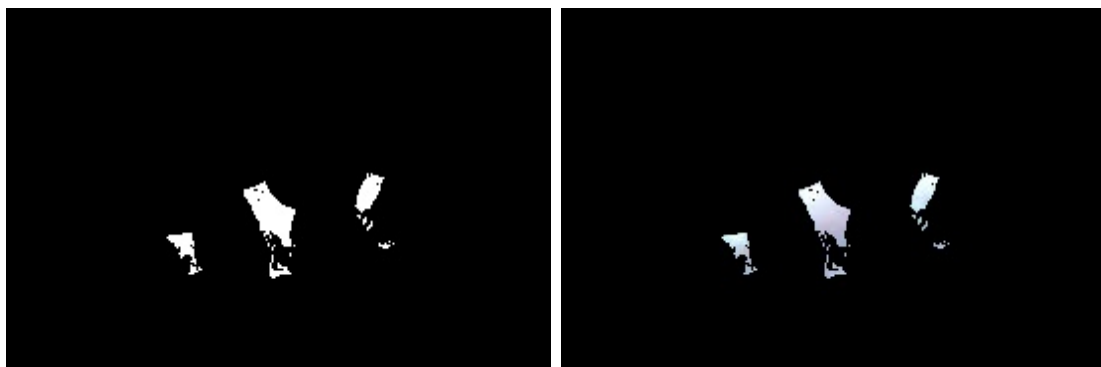


Figura 17: Máscara del color blanco (izquierda) y segmentación correspondiente (derecha).



#### 5.1.4. Ejemplo con quinta imagen

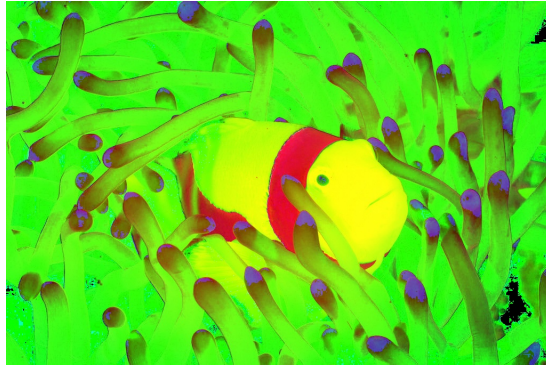


Figura 18: Conversión al espacio HSV de la quinta imagen.



Figura 19: Máscara del color naranja (izquierda) y segmentación final (derecha).

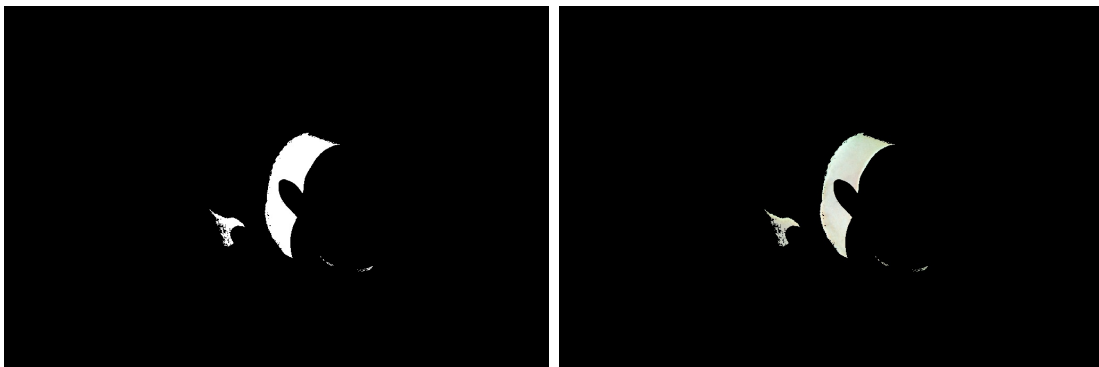


Figura 20: Máscara del color blanco (izquierda) y segmentación final (derecha).

#### 5.1.5. Ejemplo con sexta imagen

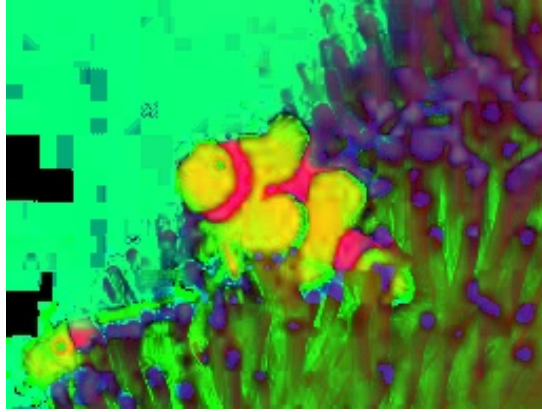


Figura 21: Imagen convertida al espacio HSV de la sexta prueba.

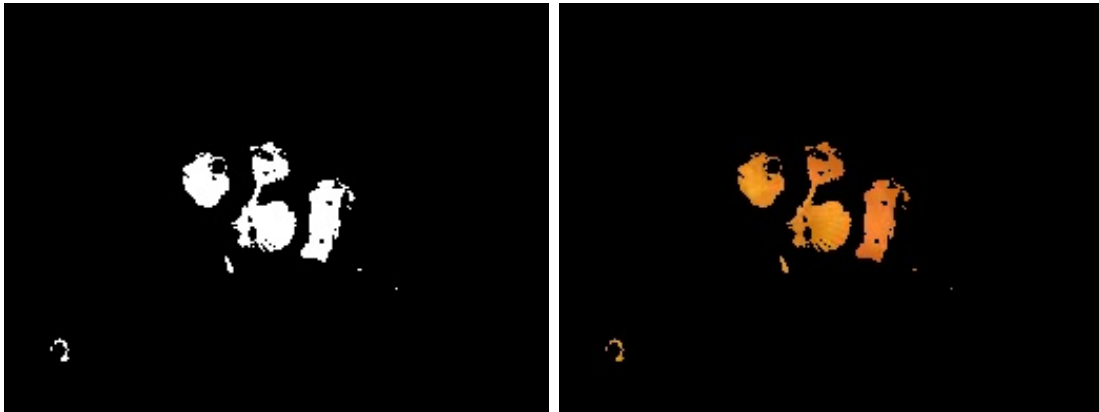


Figura 22: Máscara del color naranja (izquierda) y segmentación del pez (derecha).



Figura 23: Máscara del color blanco correspondiente.

## 6. Segmentación de color y máscaras binarias

Finalmente, se combinaron las máscaras de los colores naranja y blanco para cada imagen, logrando segmentar el cuerpo completo de los peces. Los resultados obtenidos

se compararon con ejemplos de referencia, confirmando la corrección de la segmentación mediante la visualización tanto de las máscaras como de las imágenes segmentadas.

### 6.1. Primer ejemplo



Figura 24: Pez segmentado (izquierda) y su máscara binaria (derecha).

### 6.2. Segundo ejemplo



Figura 25: Segundo pez segmentado y su correspondiente máscara binaria.

### 6.3. Tercer ejemplo

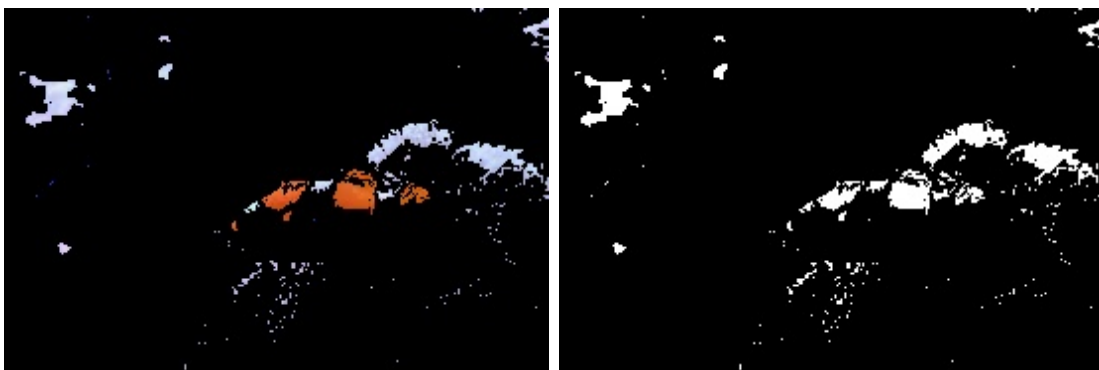


Figura 26: Tercer pez segmentado y máscara binaria.

#### 6.4. Cuarto ejemplo



Figura 27: Cuarto ejemplo de segmentación y máscara generada.

#### 6.5. Quinto ejemplo

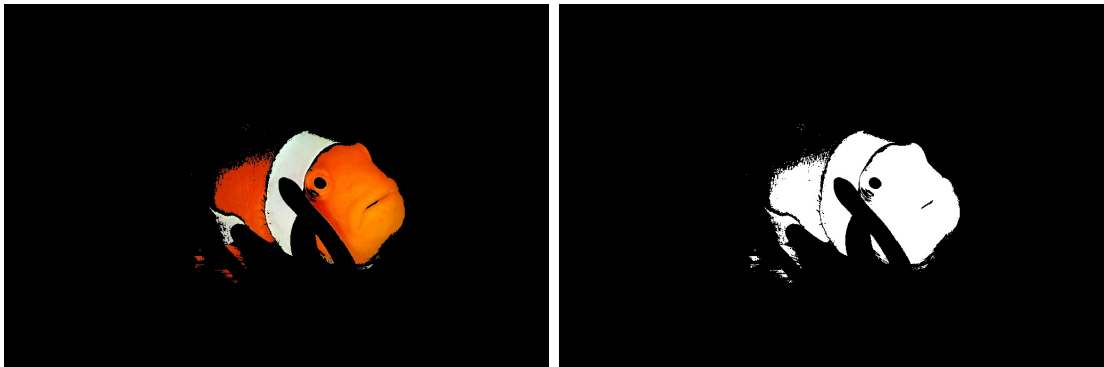


Figura 28: Quinto ejemplo de segmentación y máscara binaria.

#### 6.6. Sexto ejemplo

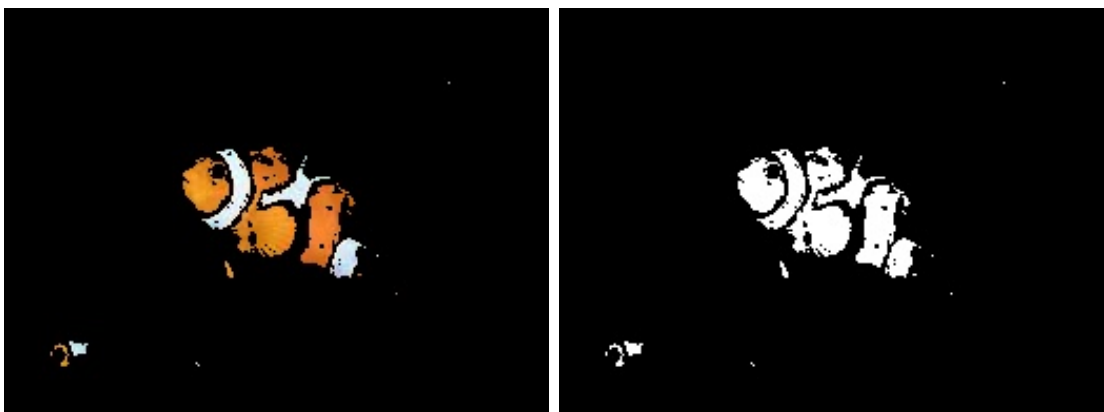


Figura 29: Nuevo pez segmentado (izquierda) y su máscara binaria (derecha).

## 7. Cuestiones acerca del apartado B

### 7.1. Descomposición por color el logotipo de su marca favorita

Se realizó la segmentación por color del escudo de la empresa *Carglass*. Para ello, se aplicó el mismo procedimiento visto en clase: 1) conversión al espacio HSV, 2) generación de máscaras binarias para cada color, 3) segmentación de la imagen original mediante la multiplicación con la máscara, y 4) cálculo del porcentaje que representa cada color respecto al total del escudo.



Figura 30: Escudo original utilizado para la segmentación.



Figura 31: Máscaras binarias generadas para los colores: amarillo, rojo y blanco.

De este modo, se pudo descomponer el escudo en tres colores principales, y calcular el área relativa de cada uno respecto al total.

### 7.2. Carga de imágenes con la función `imageio.imread()` y luego mostradas con el método `show_image()`

Al cargar las imágenes con la función `imageio.imread()` y mostrarlas posteriormente con OpenCV, los colores aparecen alterados. Esto ocurre porque `imageio` carga las imágenes en formato **RGB**, mientras que OpenCV utiliza por defecto el formato **BGR**. Como consecuencia, los canales rojo y azul se interpretan de forma invertida, dando lugar a imágenes donde, por ejemplo, el rojo aparece como azul y viceversa.



Figura 32: Ejemplo del error de visualización al intercambiar los espacios de color RGB y BGR.

La solución consiste en realizar una conversión explícita entre espacios de color mediante la función:

```
cv2.cvtColor(img, cv2.COLOR_RGB2BGR)
```

De esta manera se garantiza que los colores se visualicen correctamente al usar OpenCV.

## 8. Conclusiones del apartado B

El uso de máscaras binarias en el espacio de color HSV permitió separar de forma efectiva al pez payaso de su entorno. Gracias a la definición de rangos cromáticos se pudieron aislar de manera selectiva las zonas naranjas y blancas, comprobando la eficacia del método para identificar regiones específicas del objeto de interés. Este proceso nos permitió comprobar cómo los umbrales de color en el espacio HSV afectan directamente al resultado de la segmentación, y cómo las máscaras binarias generadas facilitan el procesamiento posterior, como el conteo de objetos o la detección de contornos. En conjunto, este procedimiento se mostró robusto en diferentes imágenes y constituye la base para aplicaciones más avanzadas de visión por ordenador, tales como el seguimiento de objetos, el conteo automático o la clasificación.