Laboratorio 4

Juan Felipe Cerón Uribe
Visión por Computador

21 de febrero de 2019

1. Descripción de las imágenes

Las imágenes utilizadas en este ejercicio son una foto de mi perro (Dante) y una imagen de un pato encontrada en internet: Notará que la cara de el pato está un poco más inclinada que

Figura 1: Foto de Dante.

la de Dante, por esta razón fue necesario girar la foto del pato. Además ajusté los tamaños de ambas fotos dado que tenían diferentes resoluciones. La resolución resultante fue tomada como el promedio de las resoluciones de las dos imágenes.

2. Imagen híbrida

Logré la imagen híbrida como se describe en el enunciado del ejercicio: Apliqué un filtro lowpass a Dante y uno highpass al pato. La imagen resultante presenta el detalle de la

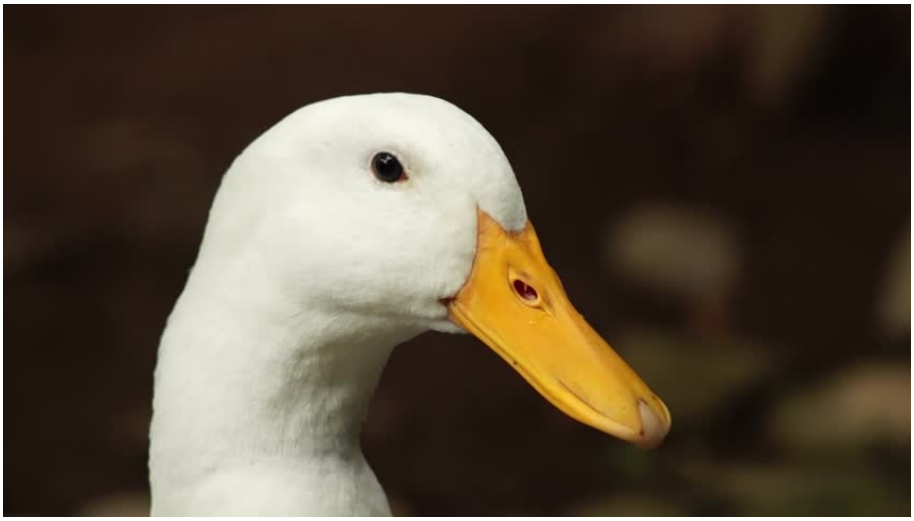

Figura 2: Foto del pato.

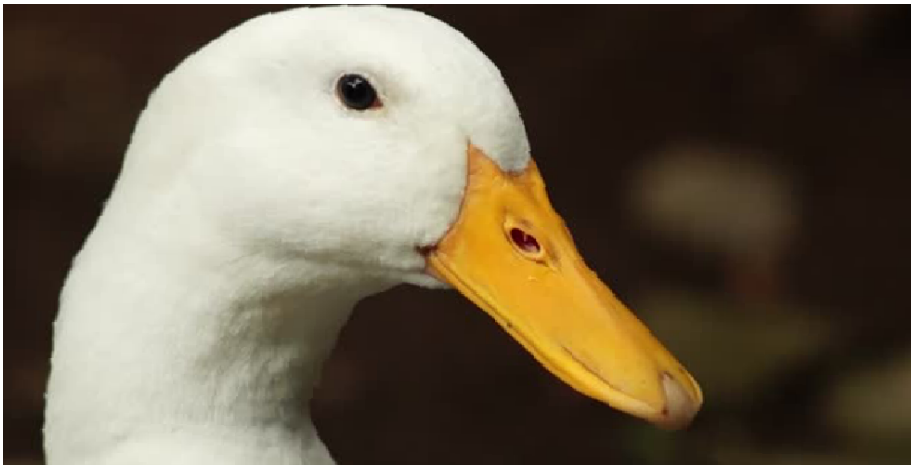

Figura 3: Modificación del pato.

imagen del pato y los rasgos gruesos de la imagen de Dante, de modo que la primera es evidente de cerca y la segunda de lejos.

Podemos visualizar el efecto generado mediante la pirámide gaussiana de la imagen híbrida.

3. Blending

Finalmente seguí el procedimiento sugerido para la creación de una imagen *blended*. Para este fin primero calculé la pirámide gaussiana tanto del pato como de Dante. Posteriormente, como se ve en el siguiente pedazo de código, junté laplacianas de baja resolución de cada

```
lowpass = cv2.GaussianBlur(dante, ksize=(51,51), sigmaX=blurrdan
highpass = pato - cv2.GaussianBlur(pato, ksize=(51,51), sigmaX=b
highpass[highpass < 0] = 0
# imagen hibrida
hibrida = highpass+lowpass
hibrida[hibrida > 255] = 255
hibrida = hibrida.astype('uint8')
plt.imshow(hibrida).write_png('danteduck.jpg')
```

Figura 4: Código de creación de imagen híbrida.

Figura 5: Imagen híbrida.

una de las imágenes. Finalmente partí de esta imagen en un procedimiento de upsampling para obtener la imagen *blended*.

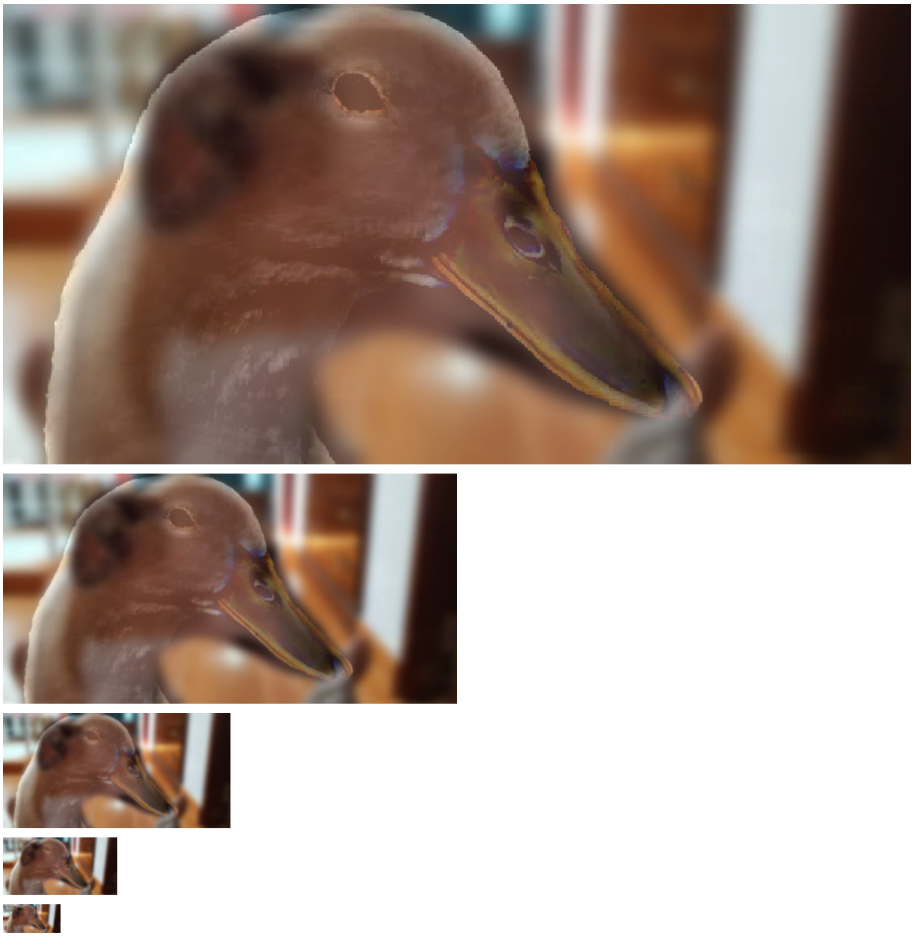

Figura 6: Piramide de la imagen híbrida .

```
# concatenacion de laplacianas
concat = []
for i in range(5):
    concat_i = L_dante[i]
    concat_i[:,0:int(concat_i.shape[1]/2),:] = L_pato[i][:,0:int(concat_i.shape[1]/2),:]
    concat.append(concat_i)

# reconstruccion de imagen blended
blended = concat[4]
for i in range(4):
    blended = cv2.pyrUp(blended)
    if concat[3-i].shape[1]%2 == 1:
        blended = cv2.add(blended[:,0:(blended.shape[1]-1),:],concat[3-i])
    else:
        blended = cv2.add(blended,concat[3-i])
cv2.imwrite('blended.png',blended)
```

Figura 7: Código de *blending* de imagen.

Figura 8: Imagen *blended*.