Prácticas de Visión por Computador
Grupo 2

Clase 1: Presentación de las prácticas, guía de instalación y Bonus (P0)

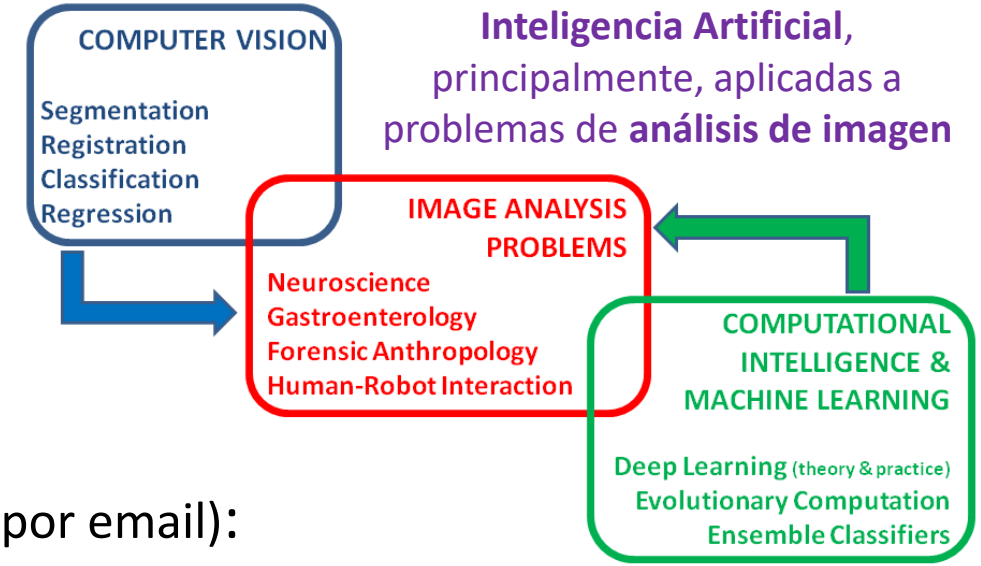
Pablo Mesejo
Universidad de Granada
Departamento de Ciencias de la Computación e Inteligencia Artificial

UNIVERSIDAD DE GRANADA

Profesor

- Pablo Mesejo Santiago
- Website: www.ugr.es/~pmesejo
- Email: pmesejo@go.ugr.es
- Tutorías (concertar cita previamente por email):
  - Oficialmente: Martes y Miércoles de 10:00 a 13:00
  - Atiendo fuera de estos horarios sin problema

Investigador en técnicas de **Inteligencia Artificial**, principalmente, aplicadas a problemas de **análisis de imagen**



Objetivo

- **QUE APRENDÁIS** (Y APROBÉIS)
  - No quiero que nadie apruebe sin aprender
  - Pero tampoco quiero que nadie aprenda y suspenda

- El objetivo es doble: aprender y aprobar
  - Lo primero suele implicar lo segundo

Objetivo (y 2)

- Mi objetivo es mejorar como profesor
  - No dudéis en darme ***feedback*** **para mejorar**
    - Tengo muy en cuenta vuestros comentarios y consejos, tanto a nivel de impartir las clases como de organizar las asignaturas
  - Estoy aquí para ayudaros en todo lo que necesitéis
  - Acordaos de cubrir las **encuestas de evaluación de la calidad docente**!
    - El curso pasado (2020-21): 9 (de 22) estudiantes evaluaron mi calidad docente. Nota media: 4.89/5.00

Dudas

- No dudéis en preguntar y solicitar toda la información que necesitéis.

  Todos somos ignorantes. Lo que pasa que ignoramos cosas diferentes.

- Aprovechad las horas de clase!

- Fuera de horas de clase:
  - emplead el email indicado (pmesejo@go.ugr.es) para consultarme dudas offline
  - emplead el email indicado (pmesejo@go.ugr.es) para solicitarme tutorías (recomendable enviarme también las dudas por email, porque en ocasiones es muy rápido resolverlas y no tenéis que esperar a clase o tutoría)

- Dudas de teoría al profesor de teoría (y de prácticas a vuestro profesor de prácticas).
  - Es él quien os evaluará de esa parte, y preguntarle a él será más efectivo.

TFG/TFM/Tesis

- No dudéis en contactar conmigo si tenéis interés en hacer el TFG, TFM o la Tesis en
  - Aprendizaje Automático
  - Visión por Computador
  - Análisis de Imágenes Biomédicas
  - Inteligencia Computacional (*Soft Computing*)

Sobre este curso (1)

- Esta clase no es un curso sobre Python ni sobre OpenCV ni sobre Keras…
  - No tenéis que convertiros en expertos en ninguna de estas librerías ni dominar todas las funcionalidades
  - No se evaluará la elegancia/calidad del código
    - Al margen de que el código entregado tiene que poder ejecutarse sin problemas y resolver el problema indicado
- Se consideran meras herramientas para resolver problemas de Visión por Computador

Sobre este curso (2)

- Asistencia a prácticas no obligatoria
- Todo el material de la asignatura lo tenéis en PRADO
- Con cada práctica se entrega (en PRADO):
  - Un informe en PDF, en donde se expone el desarrollo de la práctica y se muestran y discuten los resultados obtenidos → hay que introducir imágenes y análisis
  - Código desarrollado en Python.

Sobre este curso (y 3)

Planificación aproximada de las prácticas de la asignatura:

- **P0 -** Introducción a OpenCV: 3 ptos (Septiembre)
- **P1 -** Filtrado y Detección de regiones: 8 ptos (Octubre)
- **P2 -** Redes neuronales convolucionales: 8 ptos (Noviembre)
- **P3 -** Detección de puntos relevantes y Construcción de panoramas: 10 ptos (Diciembre)
- **Proyecto Final:** 31 ptos (Enero)

- Todas las prácticas son individuales, menos el proyecto (que es en parejas).

¿Qué es la visión por computador?

- Rama de la IA que trata los problemas de **percepción visual**
  - Ciencia que se ocupa de **la interpretación automática de las imágenes**
- En el libro de Russell & Norvig (Cap. 1) se mencionan explícitamente 6 disciplinas principales dentro de la IA:
  - Procesamiento de lenguaje natural
  - Representación del conocimiento
  - Razonamiento automático
  - Aprendizaje automático
  - **Visión por computador**
  - Robótica

¿Qué es la visión por computador?

- Computer Vision vs
  - Computer Graphics (síntesis digital de contenido visual)

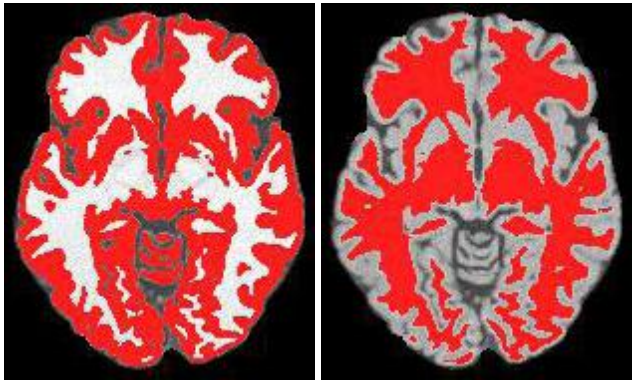
Imagen extraída de
https://upload.wikimedia.org/wikipedia/commons/5/5f/Utah_teapot_simple_2.png

  - Image Processing (procesado de imágenes a bajo nivel)

Imagen extraída de
https://es.mathworks.com/help/images/ref/imgaussfilt.html

Aplicaciones de visión por computador

Animación extraída de https://analyticsindiamag.com/what-is-the-difference-between-computer-vision-and-image-processing/



Aplicaciones de visión por computador

Imagen extraída de https://es.mathworks.com/discovery/image-registration.html

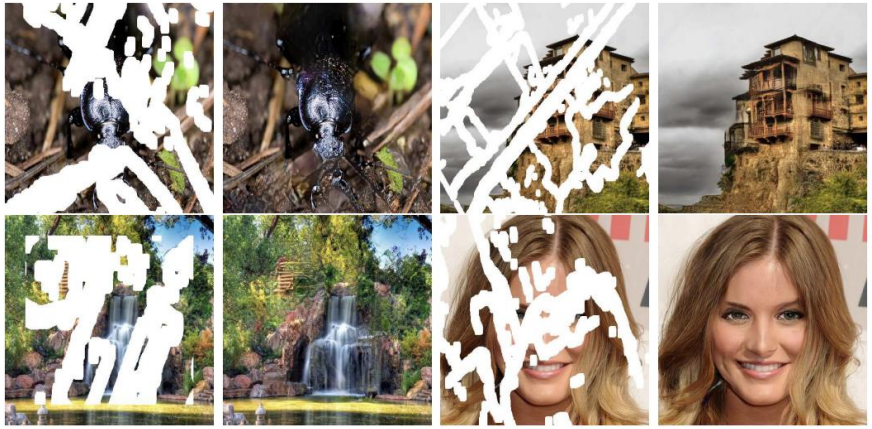
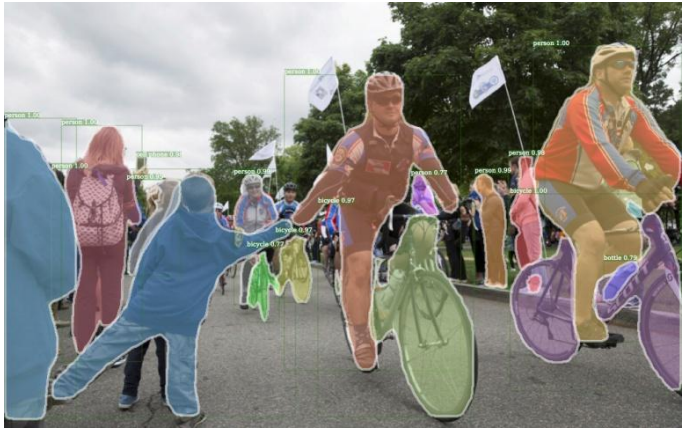


Imagen extraída de “Image Inpainting for Irregular Holes Using Partial Convolutions” (Liu et al., 2018)



Mask R-CNN output from https://github.com/facebookresearch/Detectron

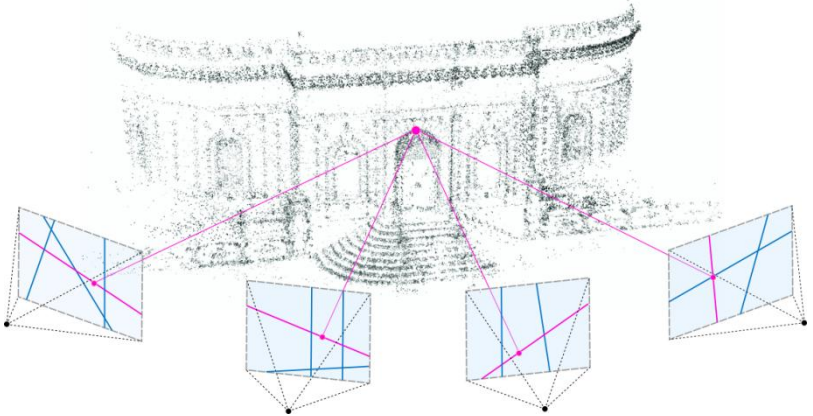


Imagen extraída de https://www.cvg.ethz.ch/research/privacy-preserving-sfm/

Aplicaciones de visión por computador

Aplicaciones de visión por computador

Guía de Instalación

- Instalar Anaconda/Spyder y Python 3.[7-8]
- pip install opencv-python
- pip install tensorflow
- pip install keras
- Comprobar la instalación y revisar/aprender Python/OpenCV
  - https://docs.opencv.org/4.5.3/d6/d00/tutorial_py_root.html

Práctica 0 (Bonus)

- Dirigida a familiarizarse con el uso de OpenCV y Python en el ordenador portátil y en la herramienta “Colab” de Google.
- 5 ejercicios básicos de lectura, visualización y manipulación de imágenes.
- Fecha límite: 30 de Septiembre (a través de PRADO)

Práctica 0 (Bonus)

Objetivo: iniciarse en el procesado y manipulación de imágenes.

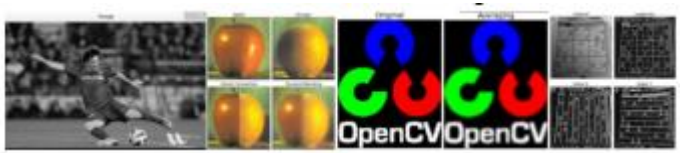
1. **Leer una imagen de fichero**, y mostrarla tanto en escala de grises como en color, empleando un *flag*.

2. **Visualizar una matriz arbitraria** de números reales, sea monobanda o tribanda, es decir, con un único canal (como pueden ser las imágenes en escala de grises o blanco y negro) o con varios canales (como las imágenes RGB o cualquier otra representación). La idea es normalizar dichas matrices al intervalo [0,1].

   Nota: Tened en cuenta que OpenCV almacena las imágenes en orden BGR en lugar de RGB. Esto influye si uno lee, por ejemplo, las imágenes con OpenCV y las muestra con matplotlib.

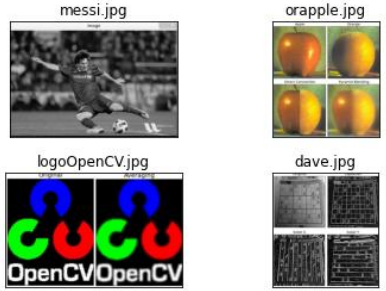
Práctica 0 (Bonus)

3. **Combinar varias imágenes en una sola.**
   ¿Qué pasa si tienen distintos tamaños o número de canales?



4. **Modificar los píxeles de una imagen**, dada una lista de posiciones.

5. **Visualizar las imágenes dentro de la misma ventana** junto con el título correspondiente.



Prácticas de Visión por Computador
Grupo 2

Fundamentos de Python para manipulación de imágenes

Pablo Mesejo
Universidad de Granada
Departamento de Ciencias de la Computación e Inteligencia Artificial

UNIVERSIDAD DE GRANADA

Lectura de Imágenes

```
filename = "images/orapple.jpg"

import matplotlib.image as mpimg
image1 = mpimg.imread(filename)

from PIL import Image
image2 = Image.open(filename)

import cv2 as cv
image3 = cv.imread(filename)
```

Python Imaging Library (PIL)

Lectura de Imágenes (y 2)

```
image1.shape
Out[1]: (535, 500, 3)

image2.shape
AttributeError: 'JpegImageFile' object has no attribute
   'shape'
import numpy as np
im = np.asarray(image2)
im.shape
Out[1]: (535, 500, 3)

image3.shape
Out[1]: (535, 500, 3)
```

matplotlib

Python Imaging Library (PIL)

OpenCV

Visualización de Imágenes

```
from IPython.display import Image
Image(filename)
```

IPython
Interactive Computing

```
import matplotlib.pyplot as plt
plt.imshow(image1)
```

matplotlib

```
image2.show()
```

Python Imaging Library (PIL)

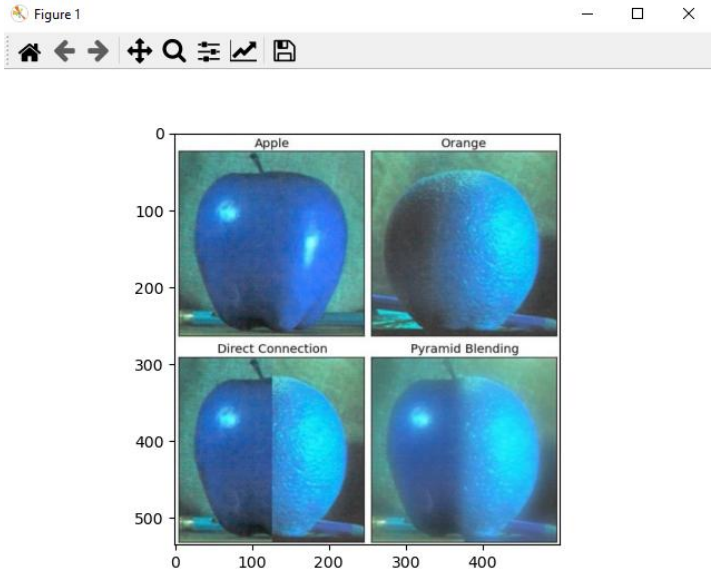
```
cv.imshow('Titulo Imagen',image3)
```

OpenCV

Ojo con los detalles

OpenCV almacena las imágenes en BGR en lugar de RGB. Si leemos con *OpenCV* y visualizamos con *matplotlib* debemos tener cuidado.

```
filename = "images/orapple.jpg"
import cv2 as cv
image = cv.imread(filename)
import matplotlib.pyplot as plt
plt.imshow(image)
```



Ojo con los detalles

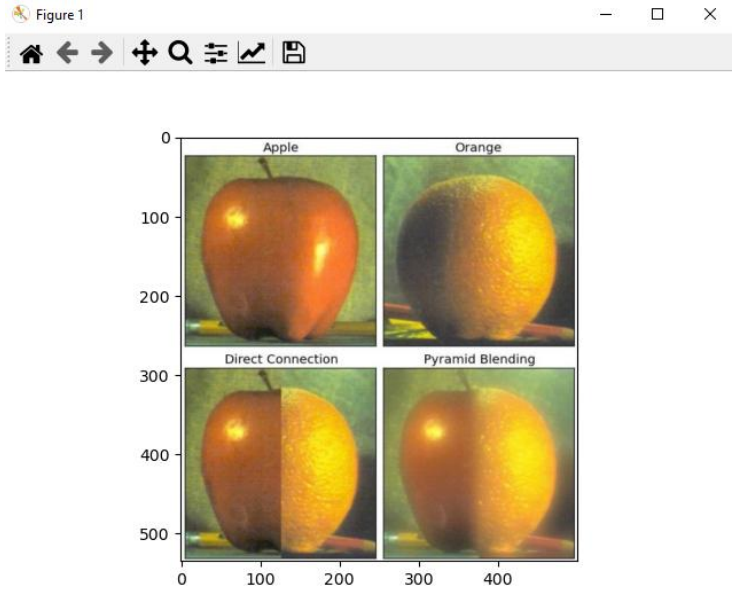
Debemos reordenar los canales para mostrarlos correctamente

Una alternativa es:

```
plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
```

Otra alternativa es:

```
plt.imshow(image[:,:,::-1])
```



Ojo con los detalles

cv2.imshow en Google Colab puede dar problemas

```
import cv2

img = cv2.imread('./drive/My Drive/Colab Notebooks/orapple.jpg')
cv2.imshow('Imagen',img)
```

```
---------------------------------------------------------------------------
DisabledFunctionError                     Traceback (most recent call last)
<ipython-input-2-ebc0249a3f51> in <module>()
      2
      3 img = cv2.imread('./drive/My Drive/Colab Notebooks/orapple.jpg')
----> 4 cv2.imshow('Imagen',img)

/usr/local/lib/python3.6/dist-packages/google/colab/_import_hooks/_cv2.py in wrapped(*args, **kwargs)
     50   def wrapped(*args, **kwargs):
     51     if not os.environ.get(env_var, False):
---> 52       raise DisabledFunctionError(message, name or func.__name__)
     53     return func(*args, **kwargs)
     54

DisabledFunctionError: cv2.imshow() is disabled in Colab, because it causes Jupyter sessions
to crash; see https://github.com/jupyter/notebook/issues/3935.
As a substitution, consider using
  from google.colab.patches import cv2_imshow
```

Ojo con los detalles

cv2.imshow en Google Colab puede dar problemas

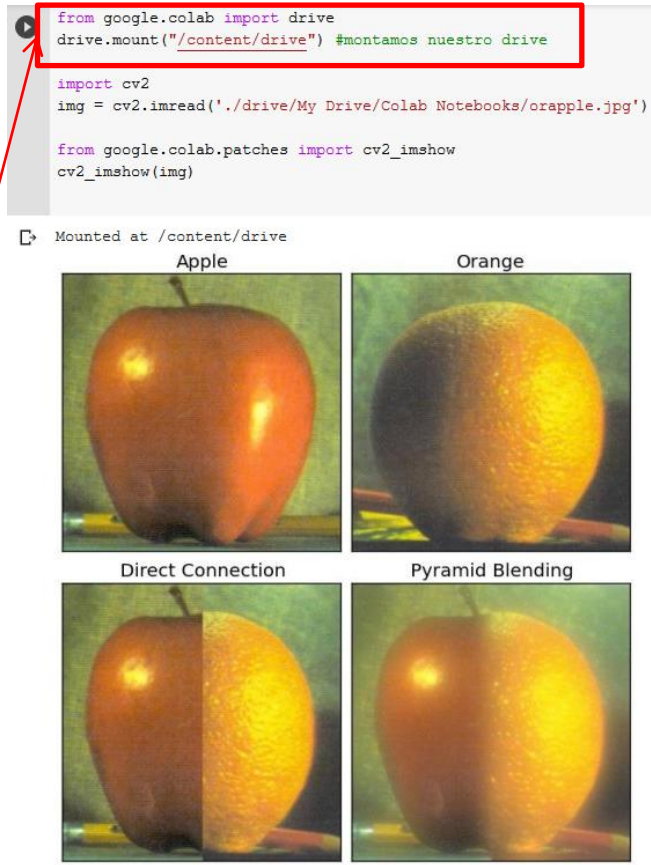
Una alternativa:

```
from google.colab.patches import cv2_imshow
cv2_imshow(img)
```

Otra alternativa:

```
import matplotlib.pyplot as plt
plt.imshow(img[:,:,::-1])
```

Nota: acordaos de montar vuestro Drive en Colab para acceder a vuestros datos

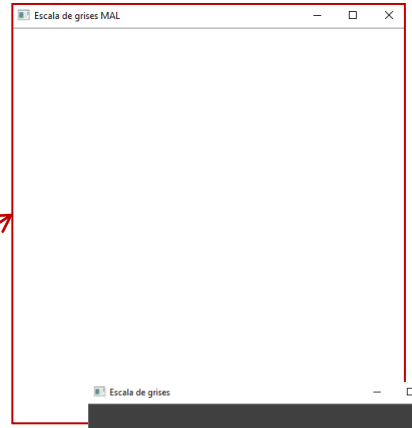


Ojo con los detalles

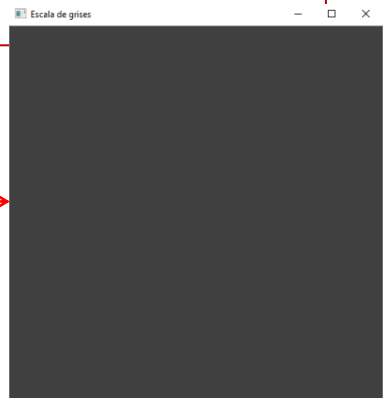
Si queréis visualizar una matriz (p.ej. RGB o escala de grises) con *OpenCV*, acordaos de que esté en el rango [0,1]

```
img = np.ones([500,500])*64
cv.imshow("Escala de grises MAL",img)
```

Satura a blanco (es decir, a 1) todo valor mayor que 1




```
cv.imshow("Escala de grises",img/255)
```



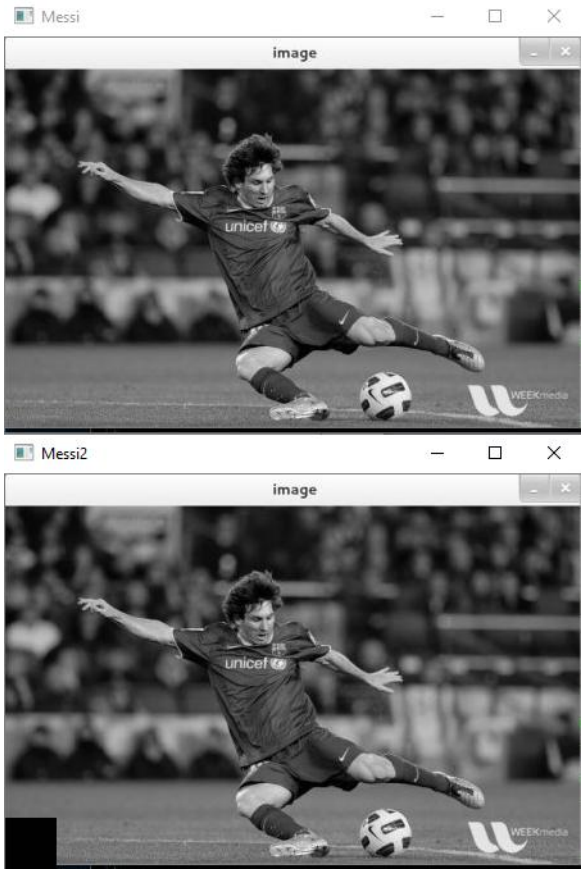
Nota: o también podéis usar matplotlib

```
plt.imshow(img, cmap='gray', vmin=0, vmax=255)
```

Ojo con los detalles

- Debemos tener cuidado…
  - ¿por qué el código siguiente no saca un cuadrado blanco?

```
import cv2
lado = 40
img = cv2.imread('images/messi.jpg')
img2 = img.copy()
img2[img2.shape[0]-lado:img2.shape[0],0:lado,:] = [1,1,1]
cv2.imshow('Messi', img)
cv2.imshow('Messi2', img2)
```



Ojo con los detalles

- Porque OpenCV leyó la imagen como una matriz de enteros (y 1 es casi negro)

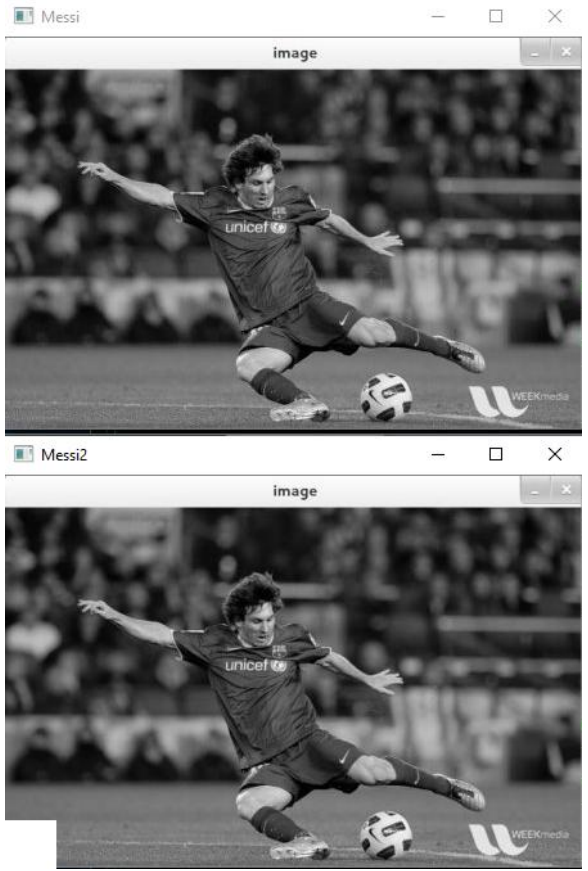
Nomb	Tipo	Tamaño	Valor
img	Array of uint8	(308, 450, 3)	[[[181 181 181]
img2	Array of uint8	(308, 450, 3)	[[[181 181 181]
lado	int	1	40

- Una forma de solucionarlo:

```
import cv2
lado = 40
img = cv2.imread('images/messi.jpg')/255
img2 = img.copy()
img2[img2.shape[0]-lado:img2.shape[0],0:lado,:] =
    [1,1,1]
cv2.imshow('Messi', img)
cv2.imshow('Messi2', img2)
```

Filas de la imagen/matriz. Eje Y

Columnas de la imagen/matriz. Eje X



Ojo con los detalles

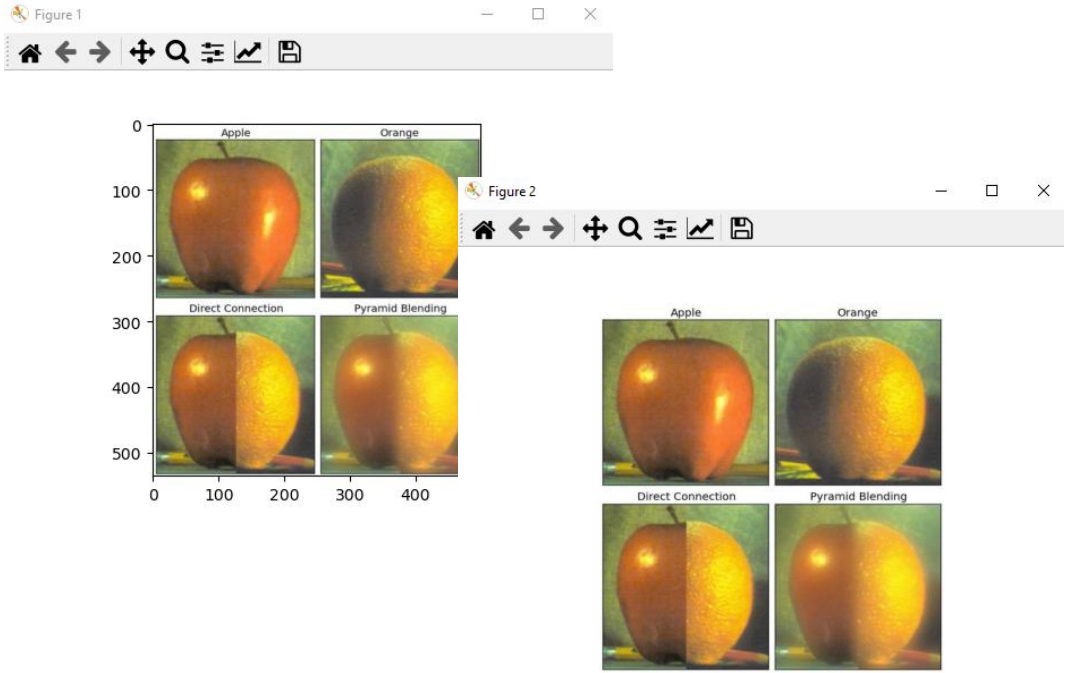
Si queréis mostrar varias figuras simultáneamente con *matplotlib*, acordaos de usar **`plt.figure()`**, o se sobreescribirán

```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

filename = "images/orapple.jpg"
image = mpimg.imread(filename)

plt.figure(1)
plt.imshow(image)

plt.figure(2)
plt.axis("off")
plt.imshow(image)
```



Enlaces de interés

- OpenCV-Python:
  - https://docs.opencv.org/4.5.3/d6/d00/tutorial_py_root.html
- Google Colab:
  - https://colab.research.google.com/notebooks/intro.ipynb
  - https://colab.research.google.com/notebooks/basic_features_overview.ipynb
  - https://colab.research.google.com/drive/1RWGmqoEQdeyh5TssoGtsXsFk8hbLGtWp
- Jupyter Notebook:
  - https://www.dataquest.io/blog/jupyter-notebook-tutorial/