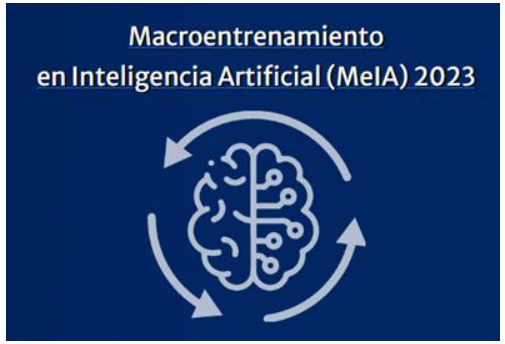


Introducción a la visión por computadora (VxC)

Dr. Jesús Manuel Olivares Ceja

jesus@cic.ipn.mx

junio 2023

EQUIPO DE APOYO:

María Cristina Cerecero Amador, Universidad Veracruzana, Veracruz, México

Meléndez Mendoza Daniela, Universidad Autónoma de Ciudad Juárez, Chihuahua, México

Camargo Porras Ana Sofía, Universitaria Agustiniana Uniagustiniana, Bogotá, Colombia

Centro de Investigación en Computación
Instituto Politécnico Nacional

AGENDA

1 ¿Qué es la Visión por computadora (VxC)?

2 ¿Para qué se quiere la Visión por computadora (VxC)?

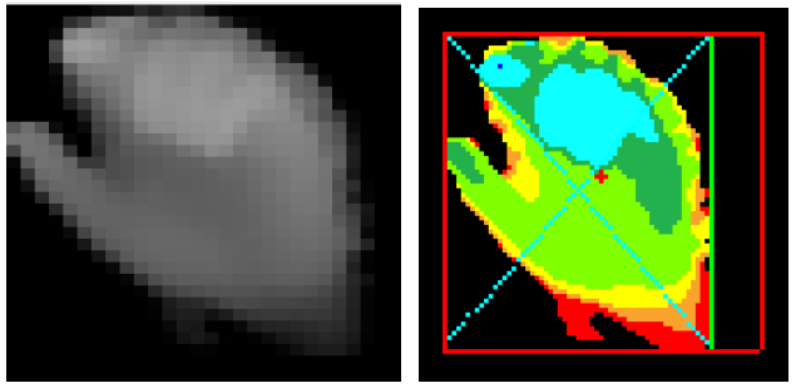
3 Ambiente de desarrollo a utilizar en este taller: GOOGLE COLAB

4 Carga de fotos y vídeos

Conclusiones

OBJETIVO DEL TALLER

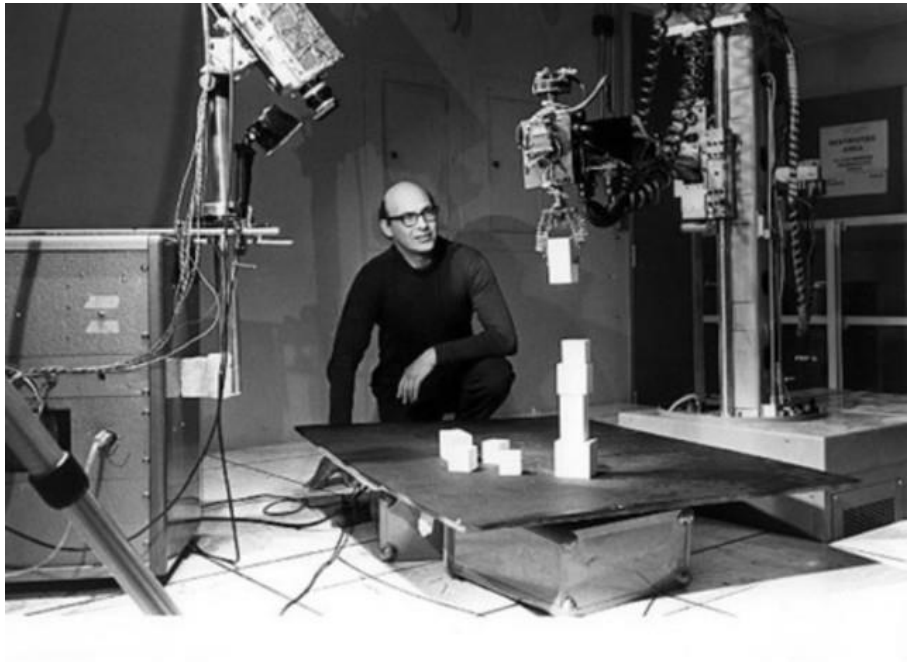
El propósito de este tutorial es que el participante conozca el procesamiento de imágenes y diferentes métodos de reconocimiento de imágenes para su aplicación en aspectos como la seguridad en edificios, ciudades; navegación automática, así como en la medición de objetos.

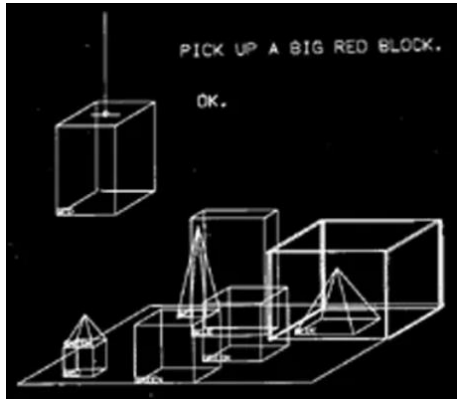


1 ¿Qué es la Visión por computadora (VxC)?

La Visión por Computadora (VxC) es un área de la Inteligencia Artificial que se encarga de procesar imágenes de fotografías y vídeos con el propósito de identificar los elementos contenidos para entender o aplicar la interpretación en la solución de problemas.

El primer trabajo de visión 3D por computadora lo realizó el Mexicano de Oaxaca, Adolfo Guzmán Arenas en 1965 en el MIT. Entre 1968 y 1970 Terry Winograd desarrolló el sistema SHRDLU que distinguía objetos como cubos, pirámides para moverlos en una mesa, esto mediante comandos en lenguaje natural. En 1970 En 1979 Kunihiko Fukushima propuso el sistema de visión por computadora Neocongnitron. En 1980 Yan LeCun propuso las Redes neuronales convolucionales (CNN) que propiciaron el auge de los sistemas de visión por computadora.



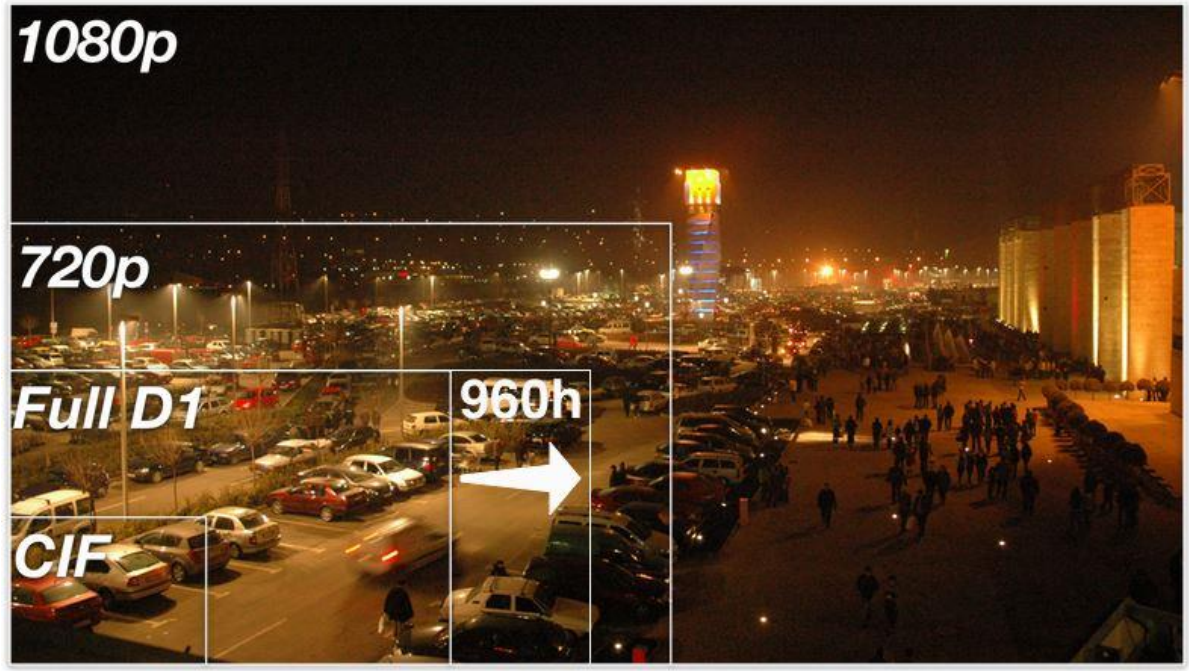


1 ¿Qué es la Visión por computadora (VxC)?

Resolución comparativa de las imágenes utilizadas en VxC

A medida que aumenta la resolución se requiere mayor cantidad de almacenamiento y procesamiento.

CIF (360×240) 4CIF (704×480) D1 (720×480)



Fuente http://www.securitycameraking.com/securityinfo/the-difference-between-cif-and-4cif-resolution/

Ejemplo de imágenes de 4 CIF, 704 x 480 pixeles

Almacenamiento:

Cada imagen ocupa 1.3 Mb.

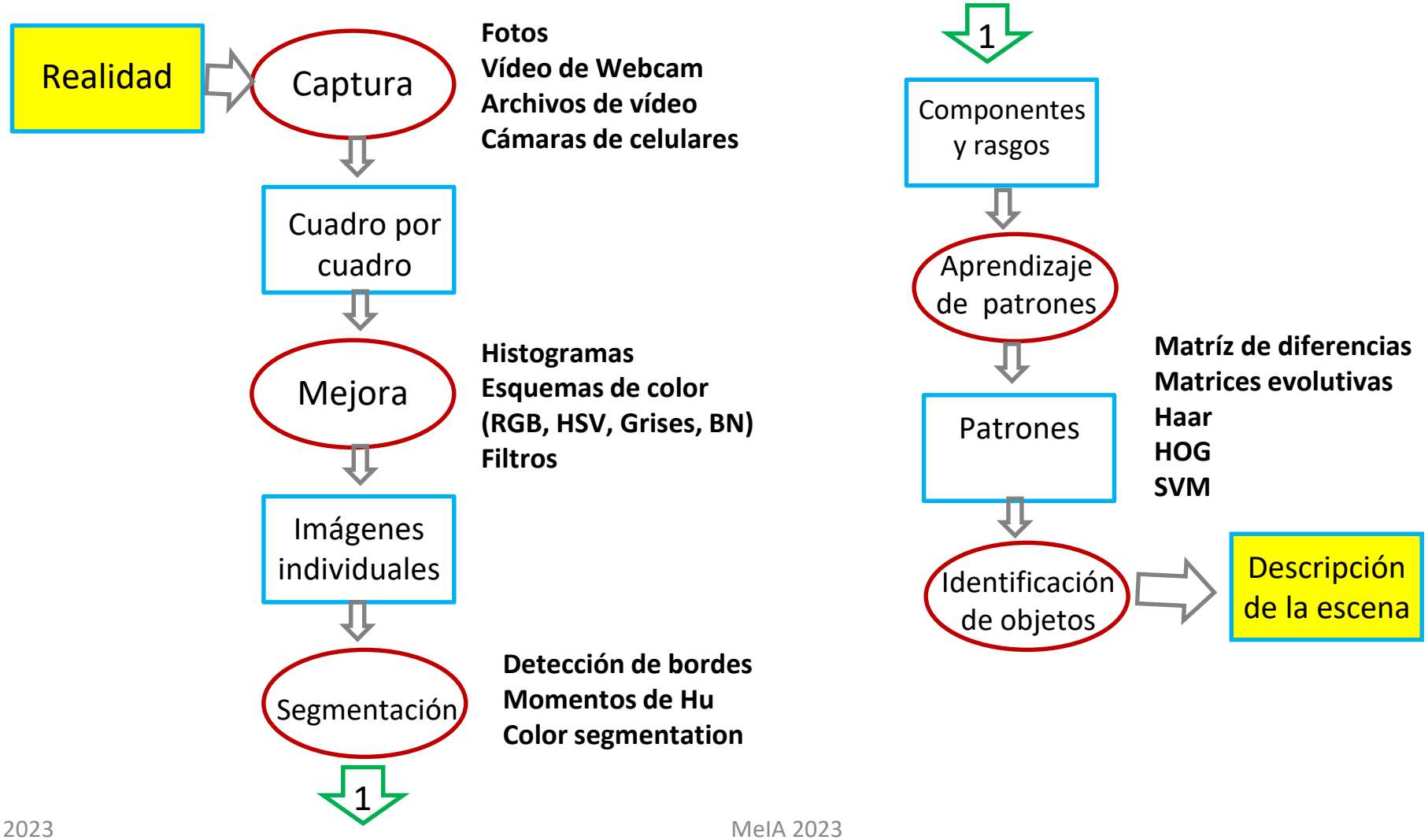
Un vídeo de 30 fps requiere 39 Mb por segundo, en un año requiere **1.14 Pb**

Procesamiento:

En una computadora con CPU de 1 Ghz se pueden procesar 15 fps.

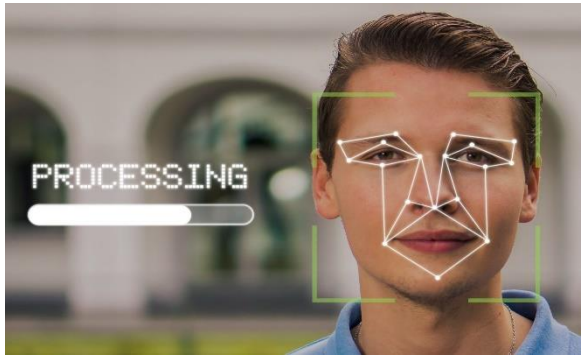
El procesamiento de vídeos de 30 fps requieren procesadores mayores a **3 Ghz**, se sugiere utilizar **GPUs**.

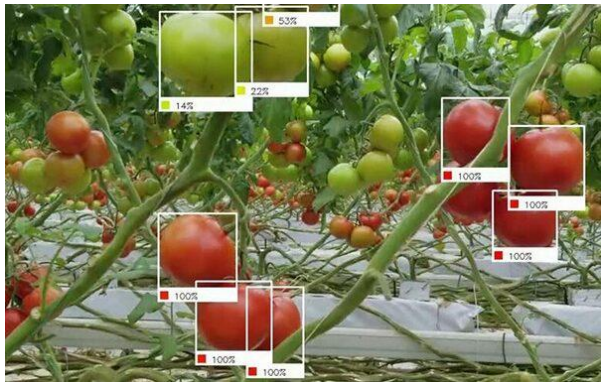
Arquitectura de un sistema de visión artificial

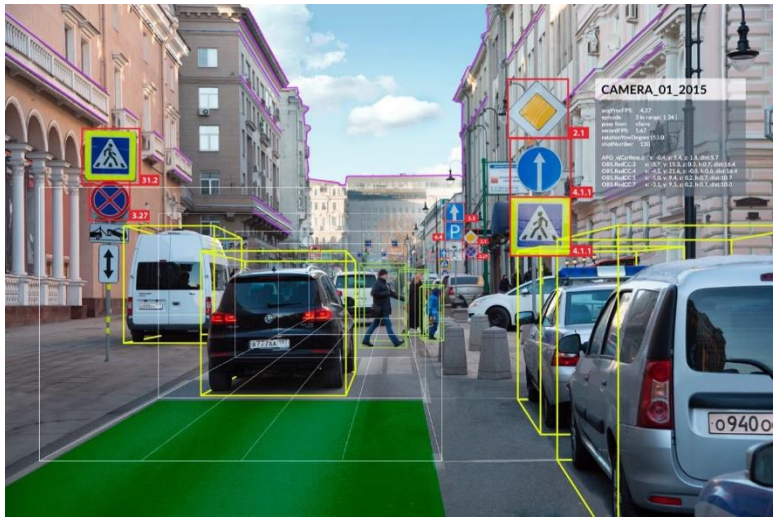


2 ¿Para qué se quiere la Visión por computadora (VxC)?

La Visión por Computadora (VxC) se aplica en diferentes aplicaciones como en el hogar con los robots de servicio, la identificación de personas para ingresar a una casa. En agricultura se utiliza para detectar automáticamente la madurez de fruta para que se realice la cosecha automatizada. En las líneas de producción de la industria permiten el control de calidad reduciendo el tiempo en que se detectan los errores. Se utiliza en las instituciones educativas, investigación, industria espacial, conducción automática, entre otros.







3 AMBIENTE DE DESARROLLO A UTILIZAR EN ESTE TALLER: GOOGLE COLAB

Existen diferentes ambientes y herramientas de desarrollo de aplicaciones, lenguajes de programación, bibliotecas, equipos de procesamiento utilizados en VxC.

4 CARGA DE FOTOS Y VÍDEOS

La Visión por computadora (VxC) requiere de una plataforma de obtención de imágenes, almacenamiento y procesamiento. Se presentan ejemplos que el estudiante debe ejecutar en Colab utilizando su cuenta de Google. Utilicen los archivos de prueba que se proporcionan anexos a esta presentación.

1. Cargando una fotografía

2. Cargando un vídeo

3. Utilizando la webcam

1. Cargando una fotografía

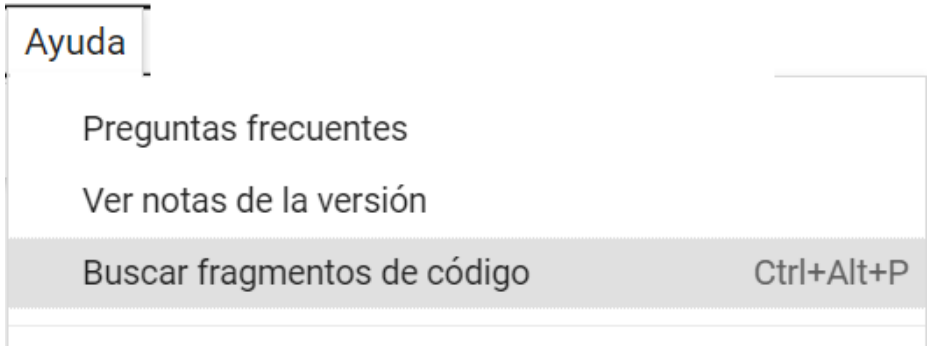
```
import cv2
import matplotlib.pyplot as plt

imagen = cv2.imread("manzana.png")
imagenRGB = cv2.cvtColor(imagen, cv2.COLOR_BGR2RGB)
plt.axis('off')
plt.imshow(imagenRGB)
```

2. Cargando un vídeo

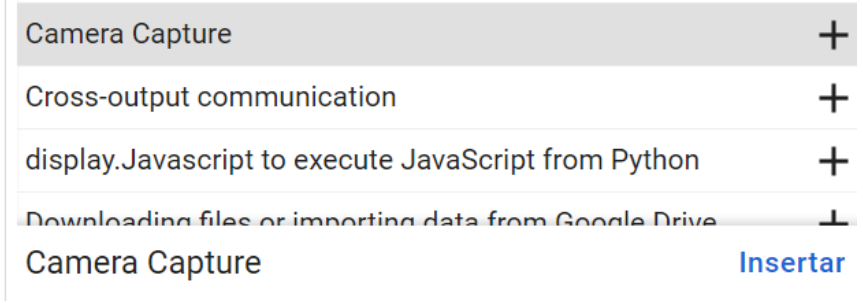
```
import imageio
import matplotlib.pyplot as plt
import matplotlib.animation as animation
from skimage.transform import resize
from IPython.display import HTML
import cv2

def muestraVideo(video):
    figura = plt.figure(figsize=(3,3))
    mov = []
    for i in range(len(video)):
        imagen = video[i]
        img = plt.imshow(imagen, animated=True) # IMAGEN RGB
        plt.axis('off')
        mov.append([img])
    anime = animation.ArtistAnimation(figura, mov, interval=50, repeat_delay=1000)
    plt.close()
    return anime

video =  imageio.mimread('prueba.mp4')
HTML(muestraVideo(video).to_html5_video())
```

3. Utilizando la webcam.

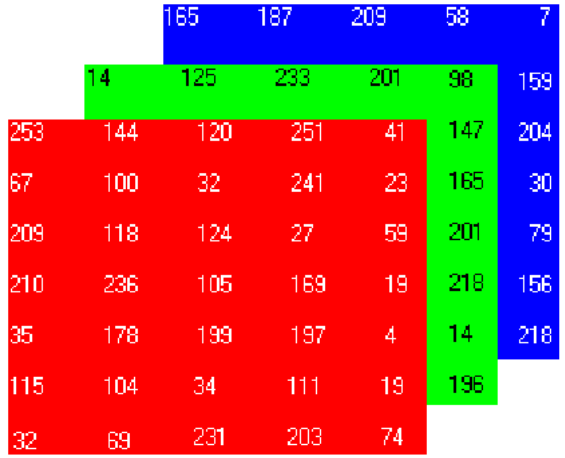
Esta práctica utiliza el código de un snipet de Google Colab, disponible en Ayuda/Buscar fragmentos de cóigo (Ctrl+Alt+P).



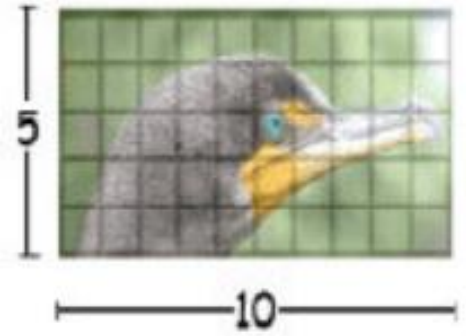
Seleccionar la opción Camera capture y pulsar en la liga de Insert. Ejecutamos ambos fragmentos de código (se requiere tener conectada una Webcam).

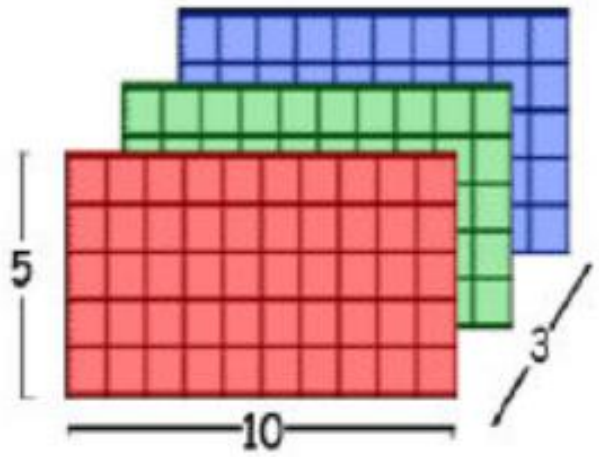


Estructura de una imagen RGB

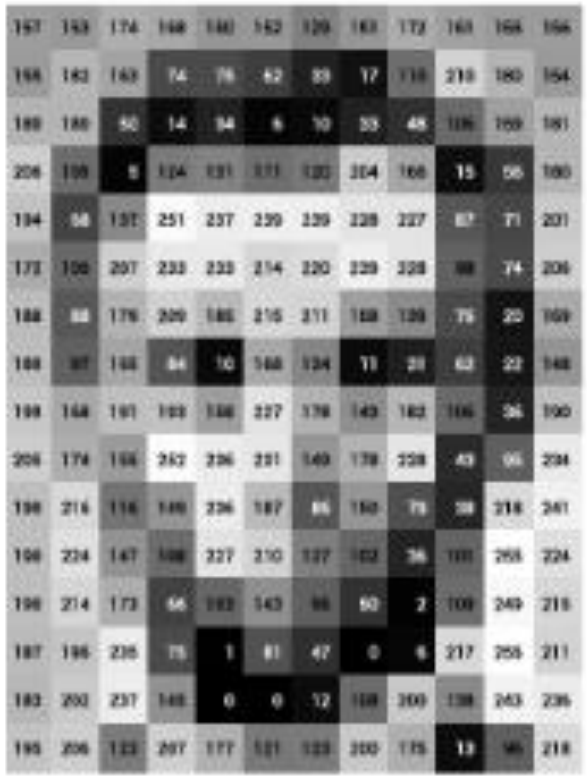



```
import cv2
import matplotlib.pyplot as plt

imagen = cv2.imread("manzana.png")
imagenRGB = cv2.cvtColor(imagen, cv2.COLOR_BGR2RGB) #BGR->RGB
renglones = imagenRGB.shape[0]
columnas = imagenRGB.shape[1]
contador = 0
for ren in range(renglones):
    for col in range(columnas):
        pixel = imagenRGB[ren][col]
        print(contador," R:",pixel[0],", ","G:",pixel[1],", ","B:",pixel[2],", ")
        contador += 1
        if contador > 15: # MUESTRA 15 PIXELES
            break
    if contador > 15: # MUESTRA 15 PIXELES
        break
plt.imshow(imagenRGB)
plt.axis('off')
plt.title("Manzana")
```





Estructura de una imagen en Tonos de gris



```
import cv2
import matplotlib.pyplot as plt

imagen = cv2.imread("manzana.png")
imagenGrises = cv2.cvtColor(imagen, cv2.COLOR_BGR2GRAY) #BGR->Grises
renglones = imagenGrises.shape[0]
columnas = imagenGrises.shape[1]
contador = 0
for ren in range(renglones):
    for col in range(columnas):
        pixel = imagenGrises[ren][col]
        print(contador," G:",pixel)
        contador += 1
        if contador > 15: # 15 PIXELES DEL RENGLON 1
            break
    if contador > 15: # 15 PIXELES DEL RENGLON 1
        break
imagenAjustada = grises = cv2.merge((imagenGrises,imagenGrises,imagenGrises))
plt.imshow(imagenAjustada)
plt.axis('off')
plt.title("Manzana en Tonos de gris")
```

Estructura de un archivo de vídeo

Un vídeo está formado por cuadros por segundo «fps, frame per second». A medida que aumenta el número de cuadros, aumenta la calidad de cada imagen. Cuando se usan pocos cuadros las imágenes se observan borrosas «blur».

.

CONCLUSIONES

Con esta presentación iniciamos el taller de visión por computadora para reconocer objetos y personas.

Hemos seleccionado la plataforma Google Colab para cargar, visualizar y procesar imágenes y vídeos. El propósito es homogeneizar el equipo de procesamiento de las personas en el taller. A medida que aumenta el tamaño de las imágenes y la calidad de los vídeos el requerimiento de memoria y velocidad de procesamiento también se incrementa.

El procesamiento de imágenes tanto de fotografías como de vídeos se realiza en cada fotograma.

Las primeras prácticas nos han permitido cargar imágenes de diferentes fuentes, en la siguiente presentación realizaremos transformaciones a las imágenes.

GRACIAS POR SU ATENCIÓN