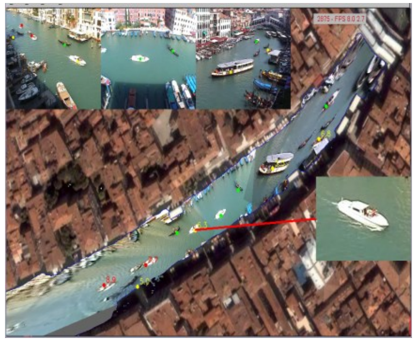
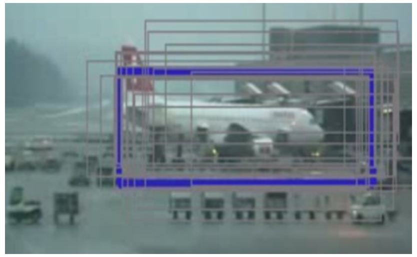
UNIVERSITÀ DEGLI STUDI DELLA BASILICATA

Corso di Sistemi Informativi
A.A. 2018/19

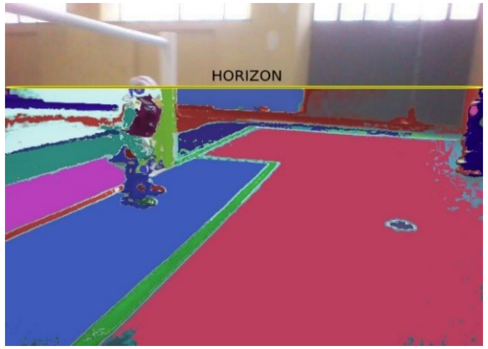
Docente
Domenico Daniele Bloisi

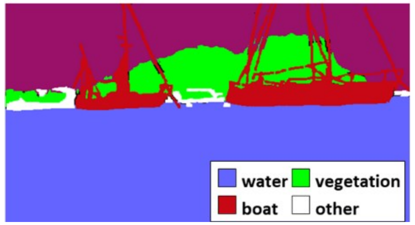


OpenCV (Python)



Aprile 2019





OpenCV

- OpenCV (Open Source Computer Vision Library) è una libreria software open source per la computer vision e il machine learning
- Distribuita con licensa BSD (è possibile utilizzarla per fini commerciali)
- Più di 2500 algoritmi disponibili
- Più di 47000 utenti nella community
- Più di 14 milioni di download

OpenCV

- Può essere utilizzata con C++, Python, Java e MATLAB
- Può essere installata su Windows, Linux, Android e Mac OS
- Dispone di interface per CUDA e OpenCL
- Viene usata da Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota

OpenCV - storia

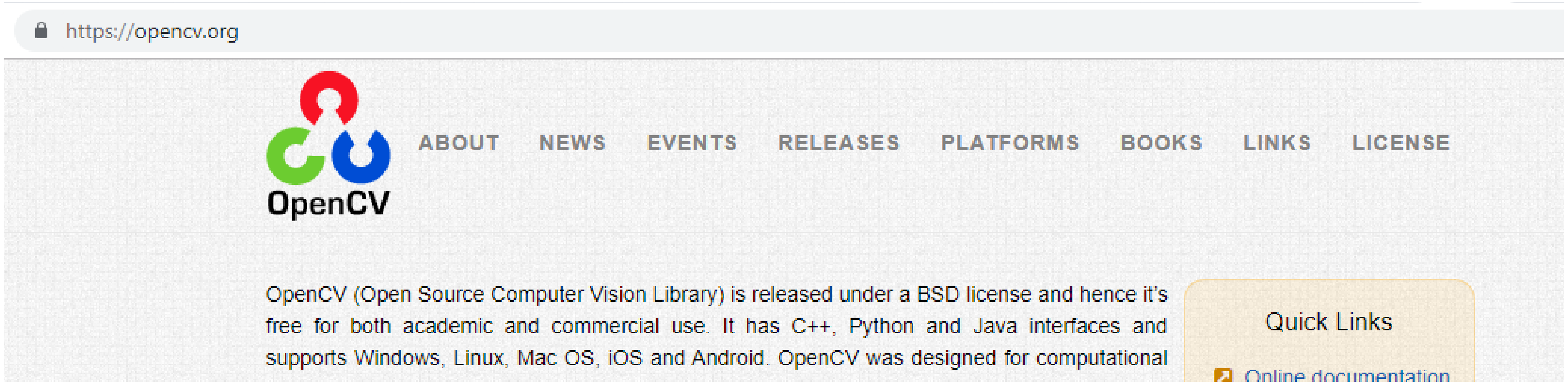
- OpenCV was started at Intel in 1999 by **Gary Bradsky**, and the first release came out in 2000. **Vadim Pisarevsky** joined Gary Bradsky to manage Intel's Russian software OpenCV team.

- In 2005, OpenCV was used on Stanley, the vehicle that won the 2005 DARPA Grand Challenge.

- Later, its active development continued under the support of Willow Garage with Gary Bradsky and Vadim Pisarevsky leading the project.

OpenCV - links

- Home: https://opencv.org/
- Documentatation: https://docs.opencv.org/
- Q&A forum: http://answers.opencv.org
- GitHub: https://github.com/opencv/



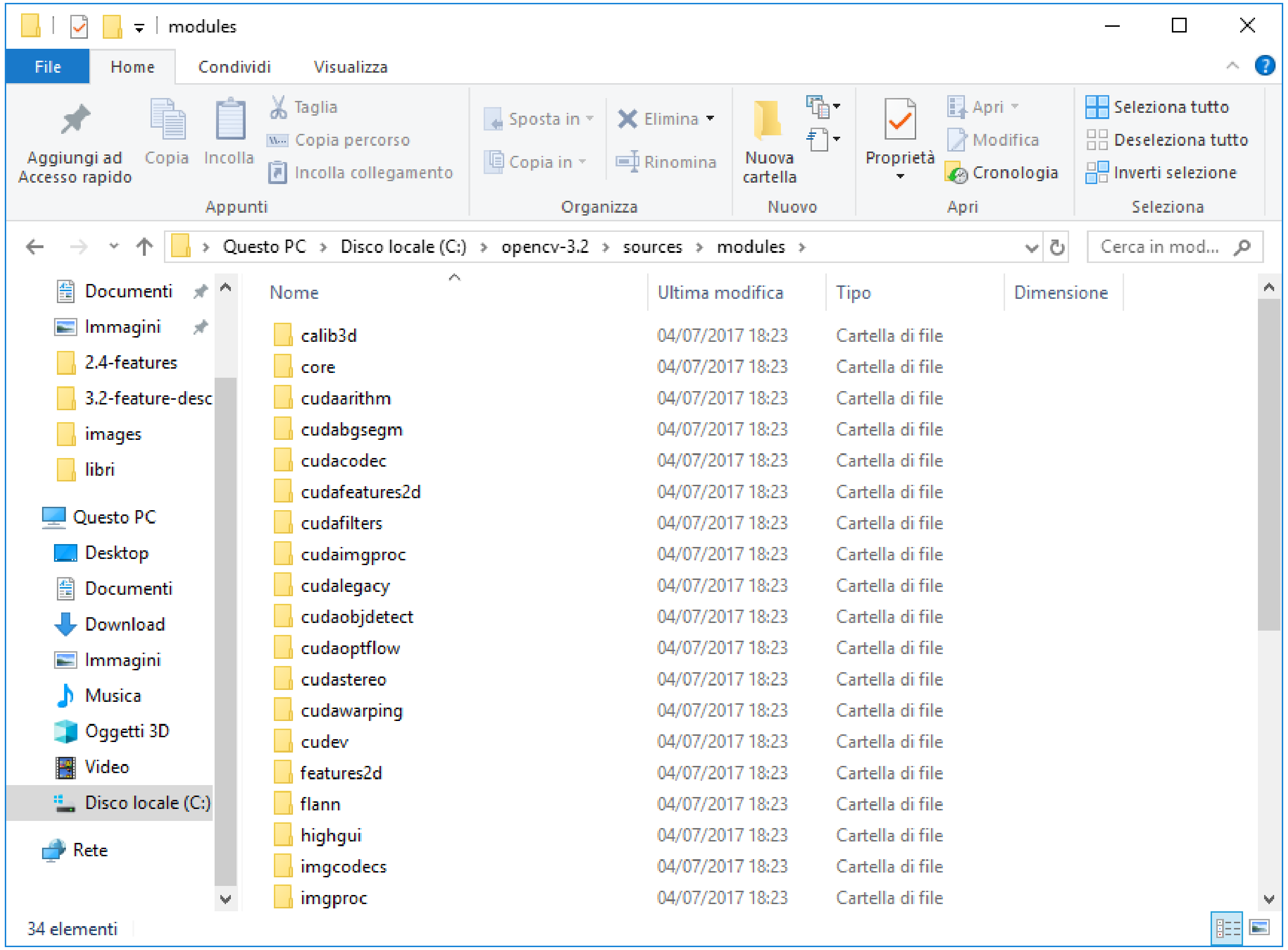
OpenCV (Open Source Computer Vision Library) is released under a BSD license and hence it's free for both academic and commercial use. It has C++, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational

OpenCV - moduli

OpenCV ha una struttura modulare

I principali moduli sono:

- core
- imgproc
- video
- calib3d
- features2d
- objdetect
- highgui



Questo PC › Disco locale (C:) › opencv-3.2 › sources › modules ›

Nome	Ultima modifica	Tipo	Dimensione
calib3d	04/07/2017 18:23	Cartella di file	
core	04/07/2017 18:23	Cartella di file	
cudaarithm	04/07/2017 18:23	Cartella di file	
cudabgsegm	04/07/2017 18:23	Cartella di file	
cudacodec	04/07/2017 18:23	Cartella di file	
cudafeatures2d	04/07/2017 18:23	Cartella di file	
cudafilters	04/07/2017 18:23	Cartella di file	
cudaimgproc	04/07/2017 18:23	Cartella di file	
cudalegacy	04/07/2017 18:23	Cartella di file	
cudaobjdetect	04/07/2017 18:23	Cartella di file	
cudaoptflow	04/07/2017 18:23	Cartella di file	
cudastereo	04/07/2017 18:23	Cartella di file	
cudawarping	04/07/2017 18:23	Cartella di file	
cudev	04/07/2017 18:23	Cartella di file	
features2d	04/07/2017 18:23	Cartella di file	
flann	04/07/2017 18:23	Cartella di file	
highgui	04/07/2017 18:23	Cartella di file	
imgcodecs	04/07/2017 18:23	Cartella di file	
imgproc	04/07/2017 18:23	Cartella di file	

OpenCV – core e imgproc

Core functionality (**core**)
A compact module defining basic data structures, including the dense multi-dimensional array Mat and basic functions used by all other modules.

Image Processing (**imgproc**)
An image processing module that includes linear and non-linear image filtering, geometrical image transformations (resize, affine and perspective warping, generic table-based remapping), color space conversion, histograms, and so on.

OpenCV – video e calib3d

Video Analysis (**video**)
A video analysis module that includes motion estimation, background subtraction, and object tracking algorithms.

Camera Calibration and 3D Reconstruction (**calib3d**)
Basic multiple-view geometry algorithms, single and stereo camera calibration, object pose estimation, stereo correspondence algorithms, and elements of 3D reconstruction.

OpenCV – features2d e objdetect

2D Features Framework (**features2d**)

Salient feature detectors, descriptors, and descriptor matchers.

Object Detection (**objdetect**)

Detection of objects and instances of the predefined classes (for example, faces, eyes, mugs, people, cars, and so on).

OpenCV – highgui e videoio

High-level GUI (**highgui**)
an easy-to-use interface to simple UI capabilities.

Video I/O (**videoio**)
An easy-to-use interface to video capturing and video codecs.

OpenCV – Python

- Python is slower compared to C++ or C. Python is built for its simplicity, portability and moreover, creativity where users need to worry only about their algorithm, not programming troubles.

- Python-OpenCV is just a wrapper around the original C/C++ code. It is normally used for combining best features of both the languages. Performance of C/C++ & Simplicity of Python.

- So when you call a function in OpenCV from Python, what actually run is underlying C/C++ source.

- Performance penalty is < 4%

Source: Mašinska vizija

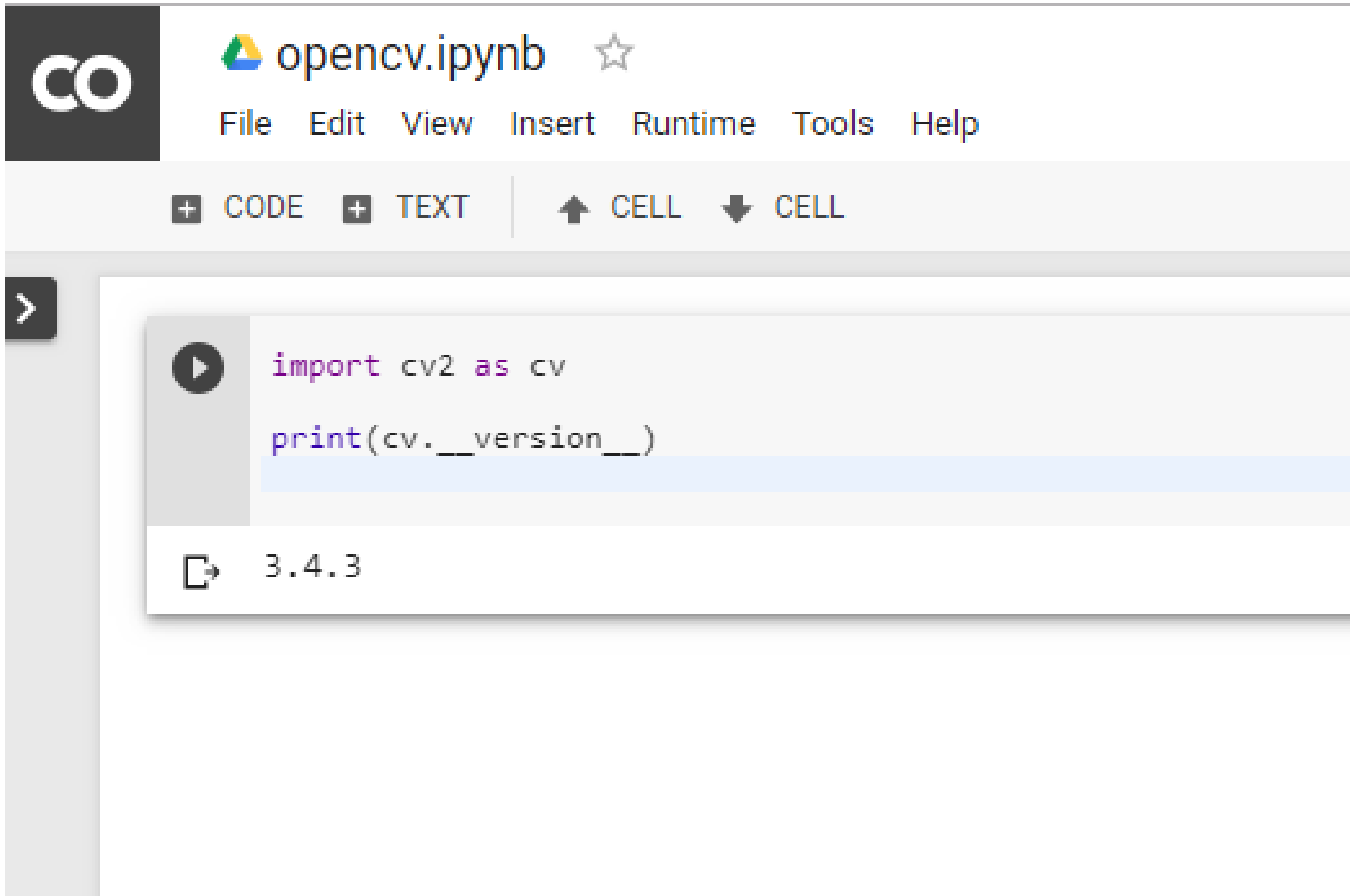
OpenCV Timeline

Version	Released	Reason	Lifetime
pre 1.0	2000 (first alpha)	-	6 years
1.0	2006 (ChangeLog)	maturity	3 years
2.0	2009 (ChangeLog)	C++ API	>3 years
3.0	2014	several (next level maturity, ...)	
4.0	**Nov. 2018**	better DNN support	

https://github.com/opencv/opencv/wiki/images/OpenCV3_0_CVPR_2014.pptx

OpenCV in Colab

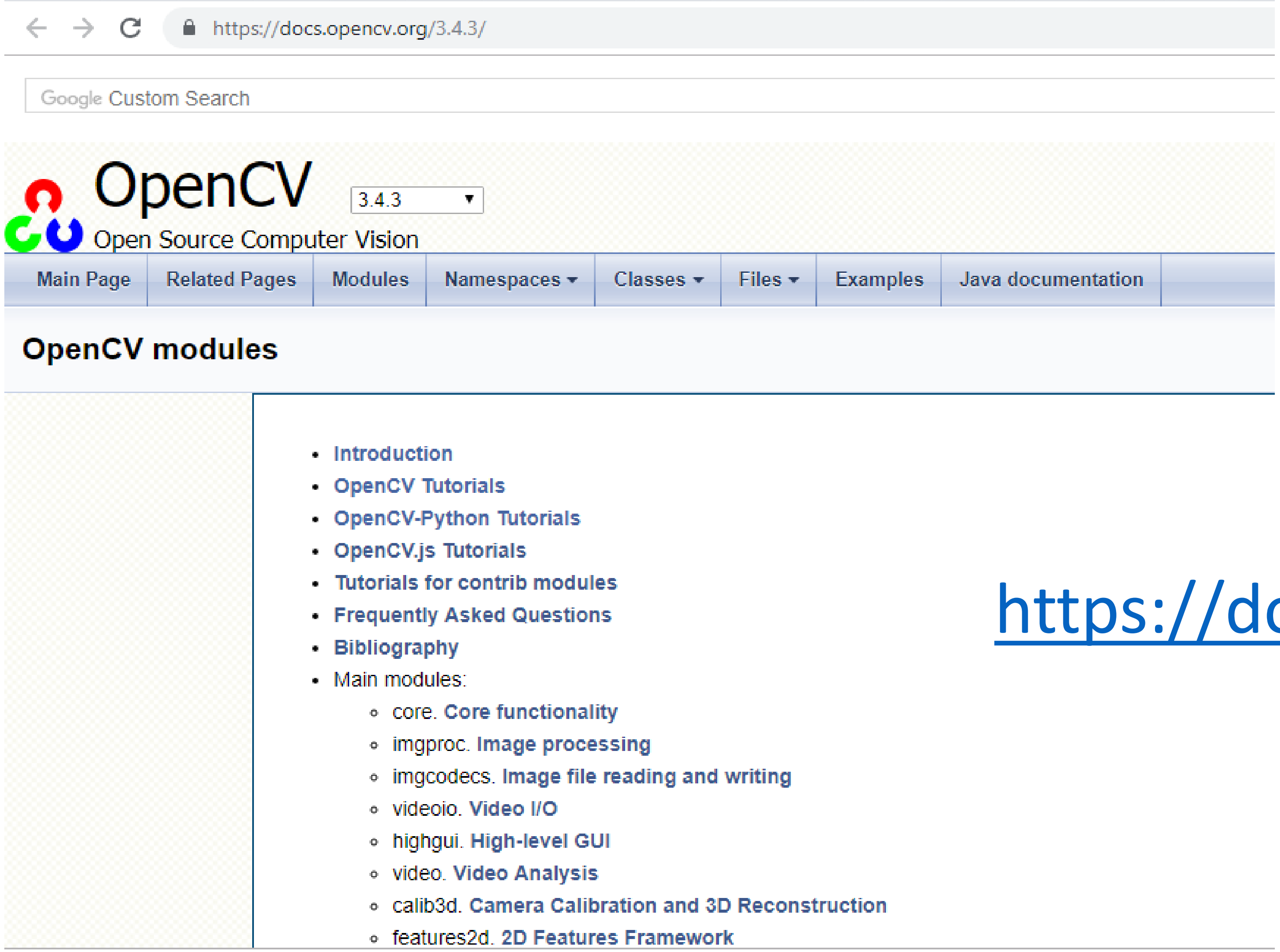
La versione di OpenCV attualmente disponibile in Google Colab è la 3.4.3



```
import cv2 as cv
print(cv.__version__)
```

```
3.4.3
```

OpenCV 3.4.3 docs



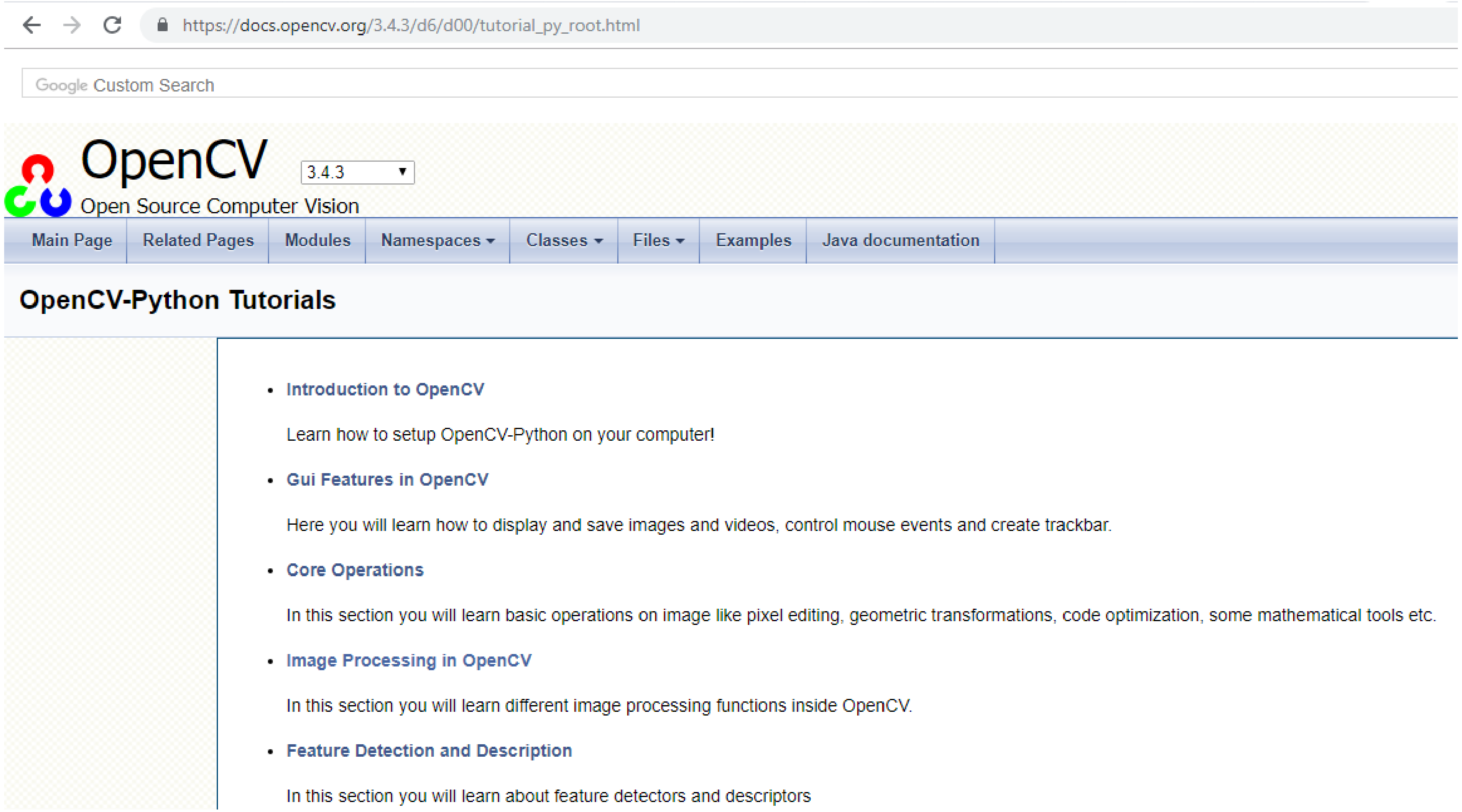
OpenCV modules

- Introduction
- OpenCV Tutorials
- OpenCV-Python Tutorials
- OpenCV.js Tutorials
- Tutorials for contrib modules
- Frequently Asked Questions
- Bibliography
- Main modules:
  - core. Core functionality
  - imgproc. Image processing
  - imgcodecs. Image file reading and writing
  - videoio. Video I/O
  - highgui. High-level GUI
  - video. Video Analysis
  - calib3d. Camera Calibration and 3D Reconstruction
  - features2d. 2D Features Framework

https://docs.opencv.org/3.4.3/

OpenCV-Python Tutorials

OpenCV fornisce una serie di tutorial specifici per Python che possono essere utilizzati per imparare ad utilizzare la libreria attraverso esempi pratici



OpenCV-Python Tutorials

- Introduction to OpenCV

  Learn how to setup OpenCV-Python on your computer!

- Gui Features in OpenCV

  Here you will learn how to display and save images and videos, control mouse events and create trackbar.

- Core Operations

  In this section you will learn basic operations on image like pixel editing, geometric transformations, code optimization, some mathematical tools etc.

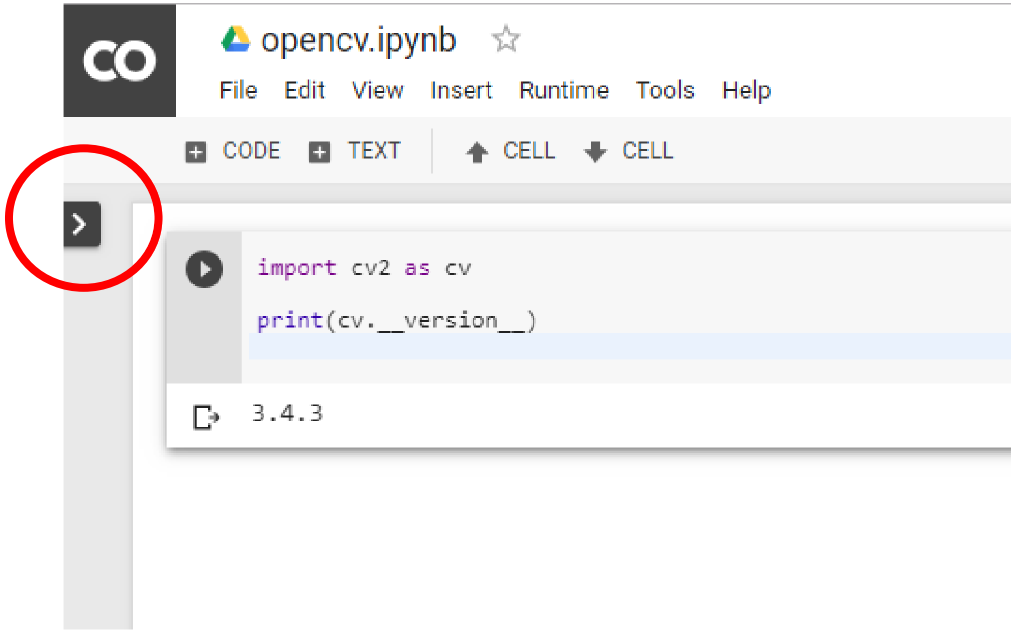
- Image Processing in OpenCV

  In this section you will learn different image processing functions inside OpenCV.

- Feature Detection and Description

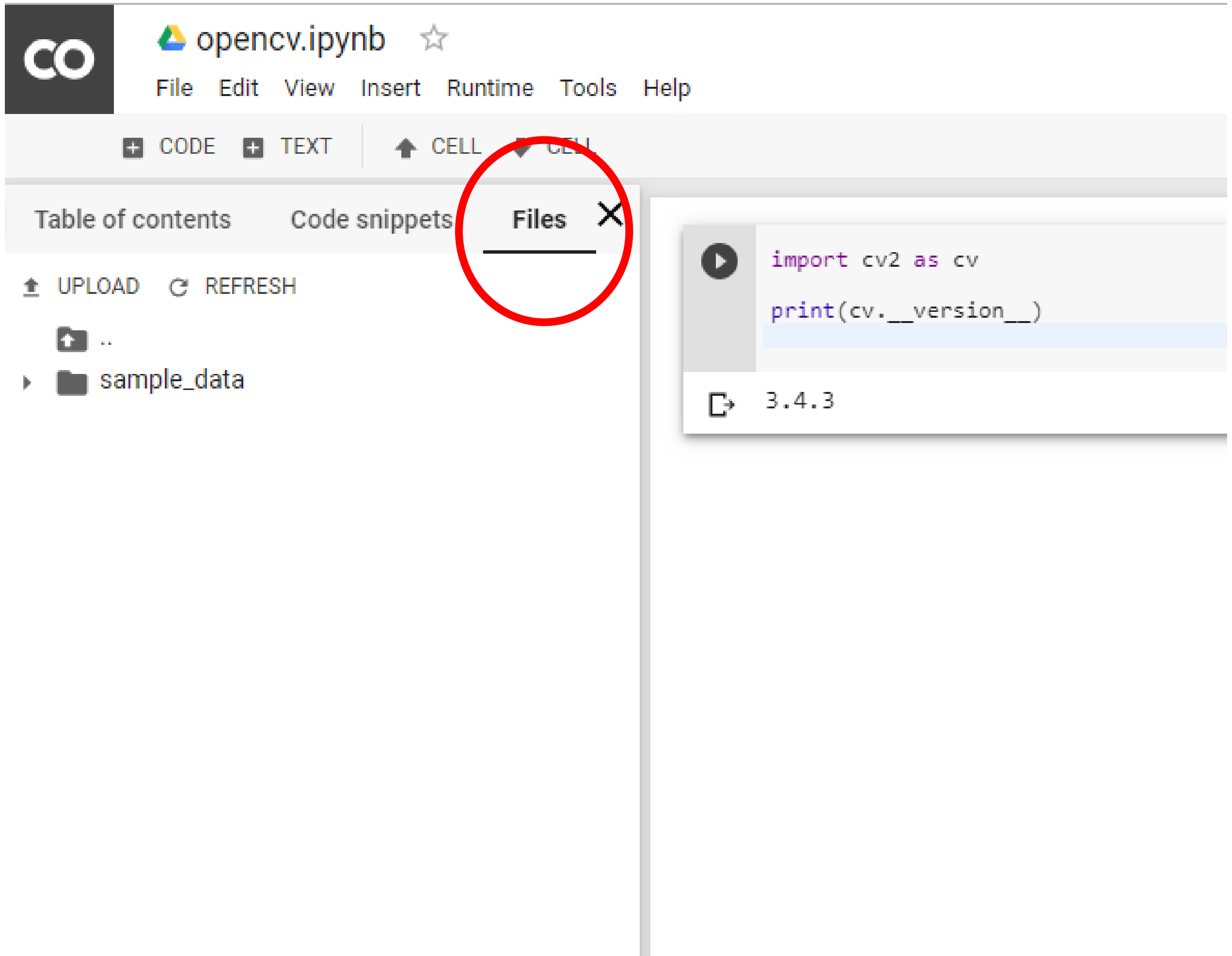
  In this section you will learn about feature detectors and descriptors

Load an image in Colab



```
import cv2 as cv
print(cv.__version__)
```

```
3.4.3
```

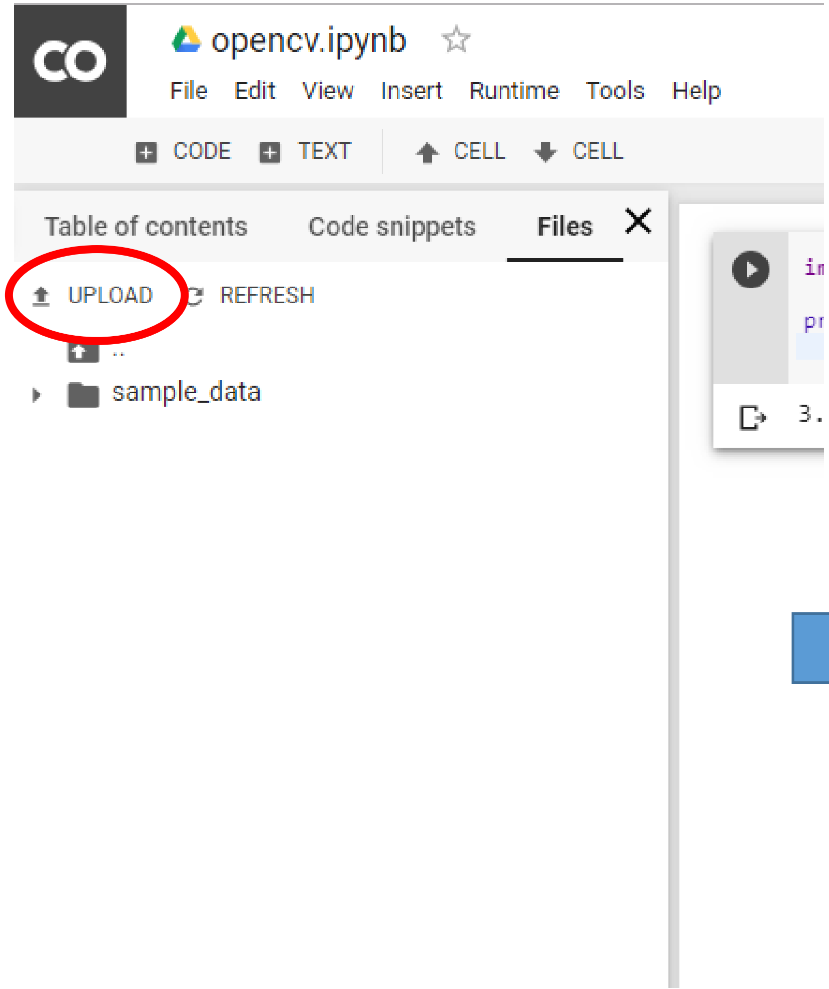


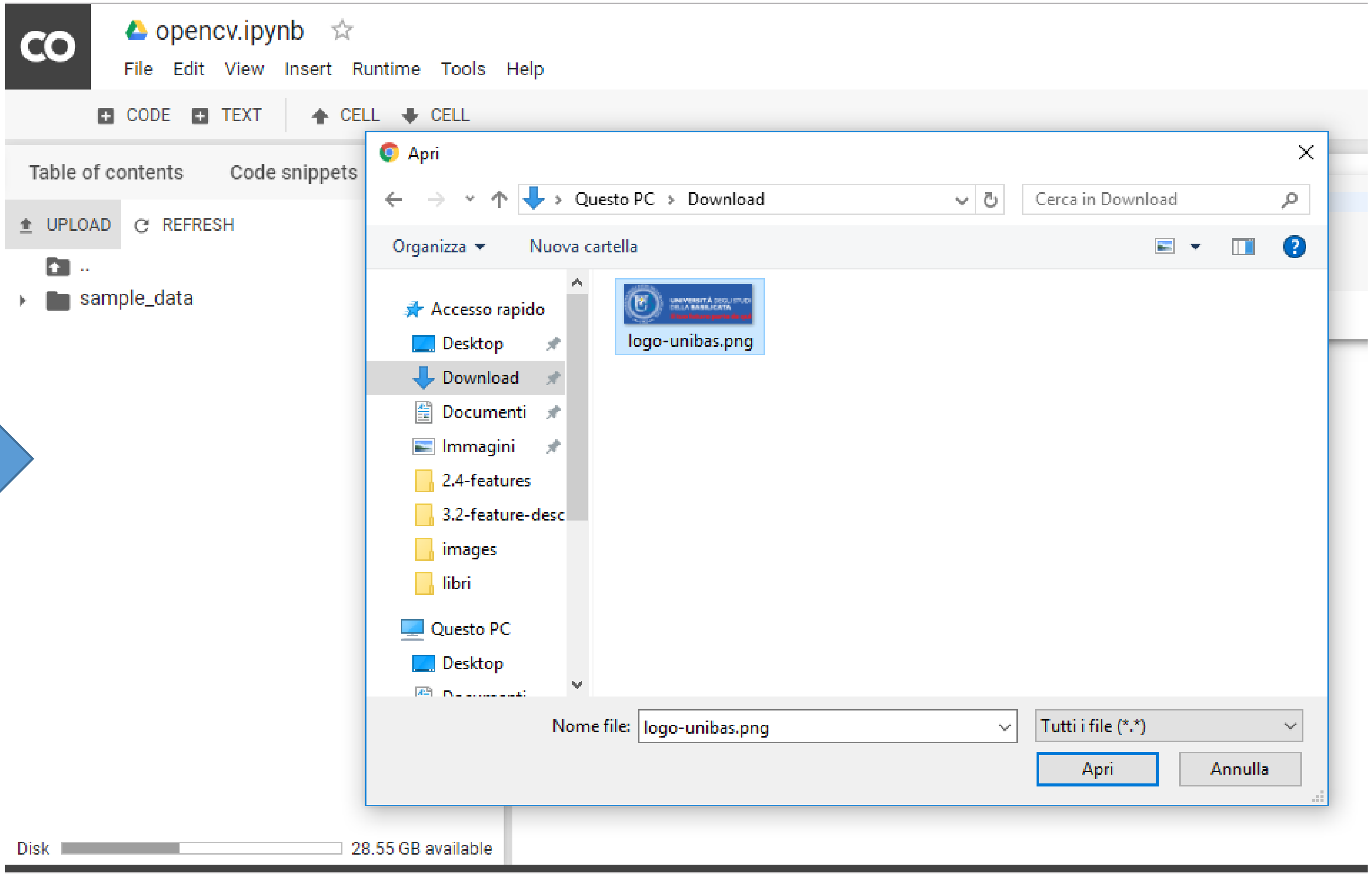
```
import cv2 as cv
print(cv.__version__)
```

```
3.4.3
```

Load an image in Colab

http://portale.unibas.it/contents/instance1/images/logo-unibas.png

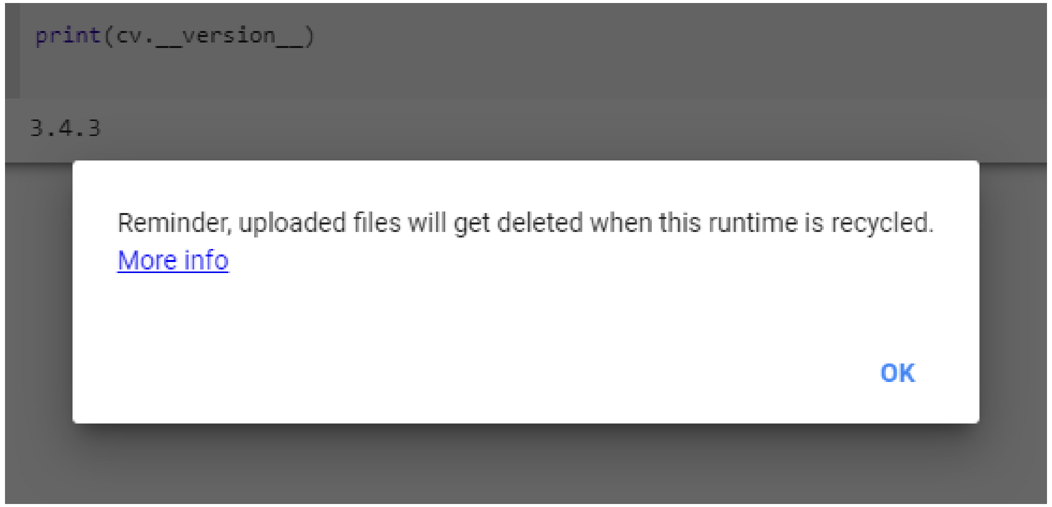




Nome file: logo-unibas.png

Tutti i file (*.*)

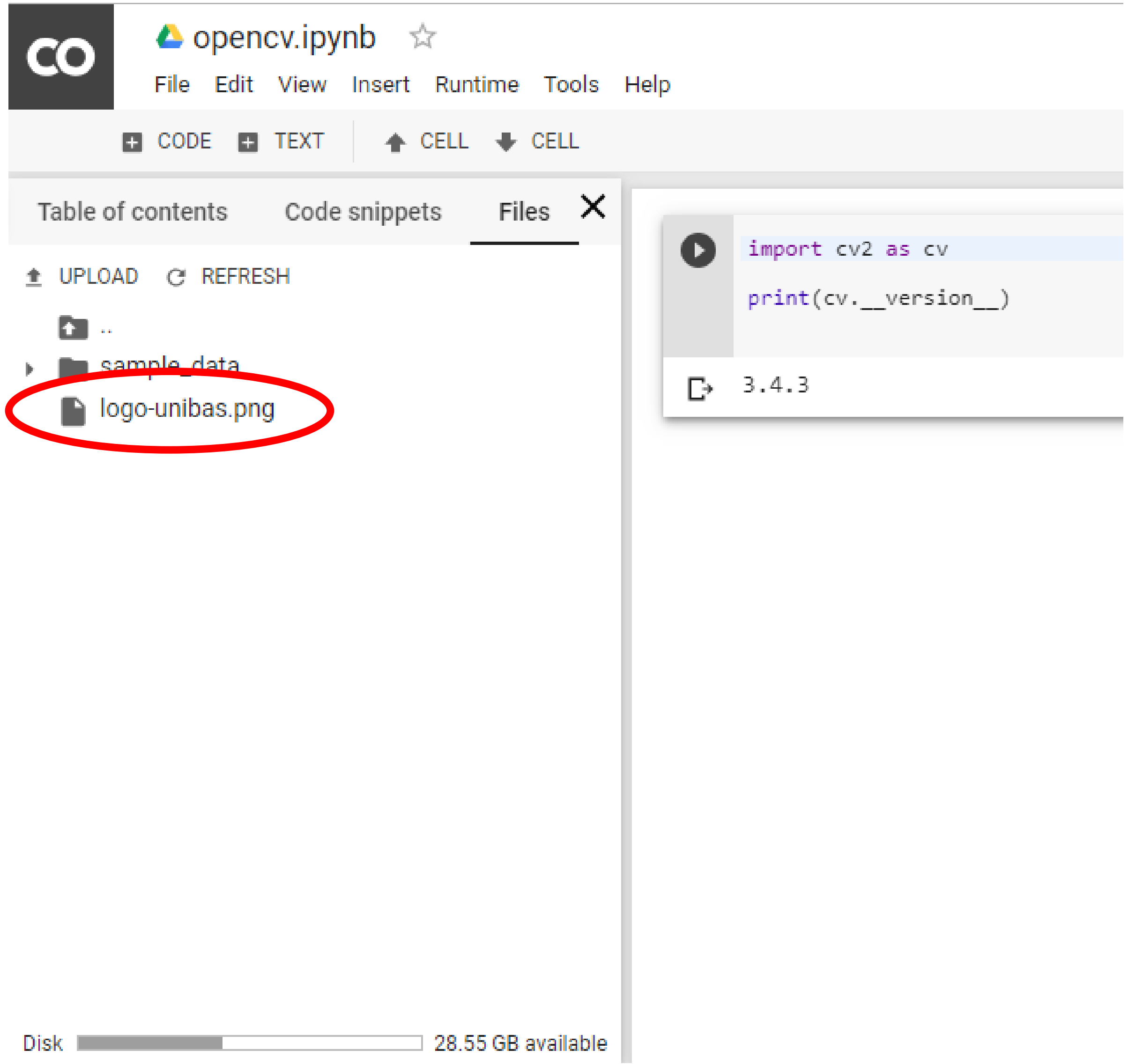
Load an image in Colab



```
print(cv.__version__)
```

```
3.4.3
```

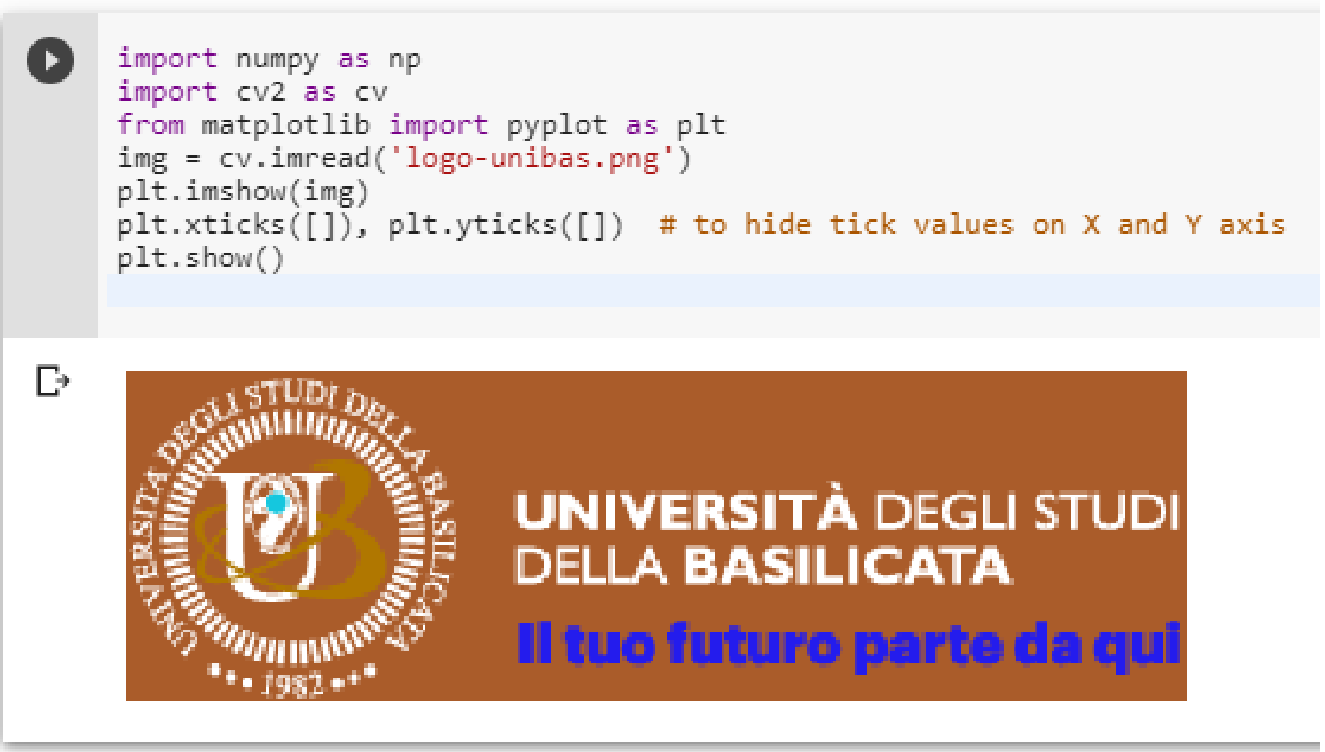
Reminder, uploaded files will get deleted when this runtime is recycled.
More info



```
import cv2 as cv
print(cv.__version__)
```

```
3.4.3
```

Read an image

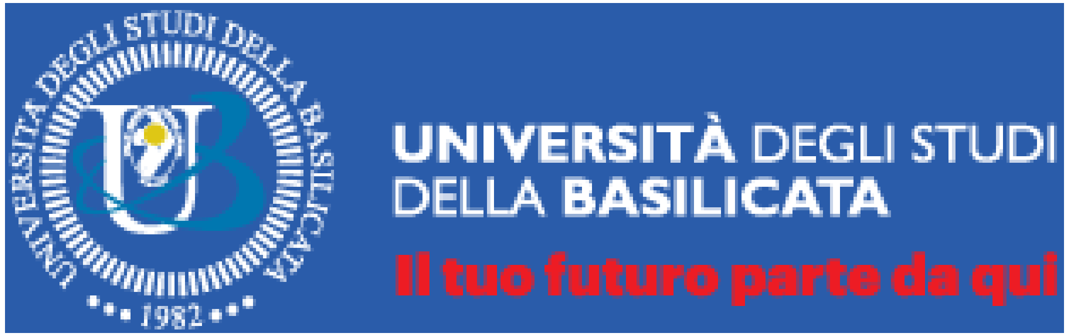


```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png')
plt.imshow(img)
plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis
plt.show()
```

warning

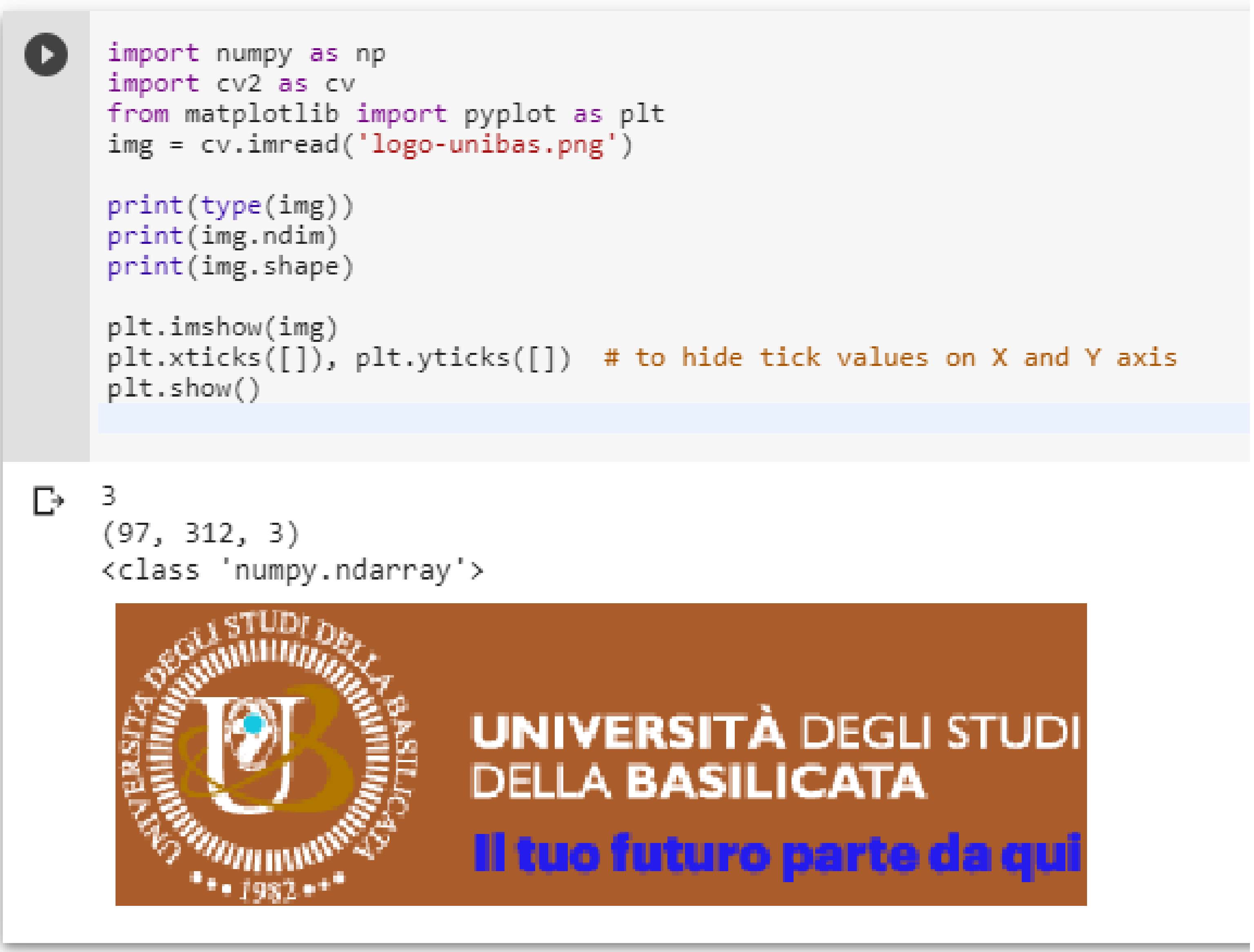
Color image loaded by OpenCV is in BGR mode. But Matplotlib displays in RGB mode. So color images will not be displayed correctly in Matplotlib if image is read with OpenCV.

Source image



Images are NumPy arrays

Images in OpenCV-Python are NumPy arrays

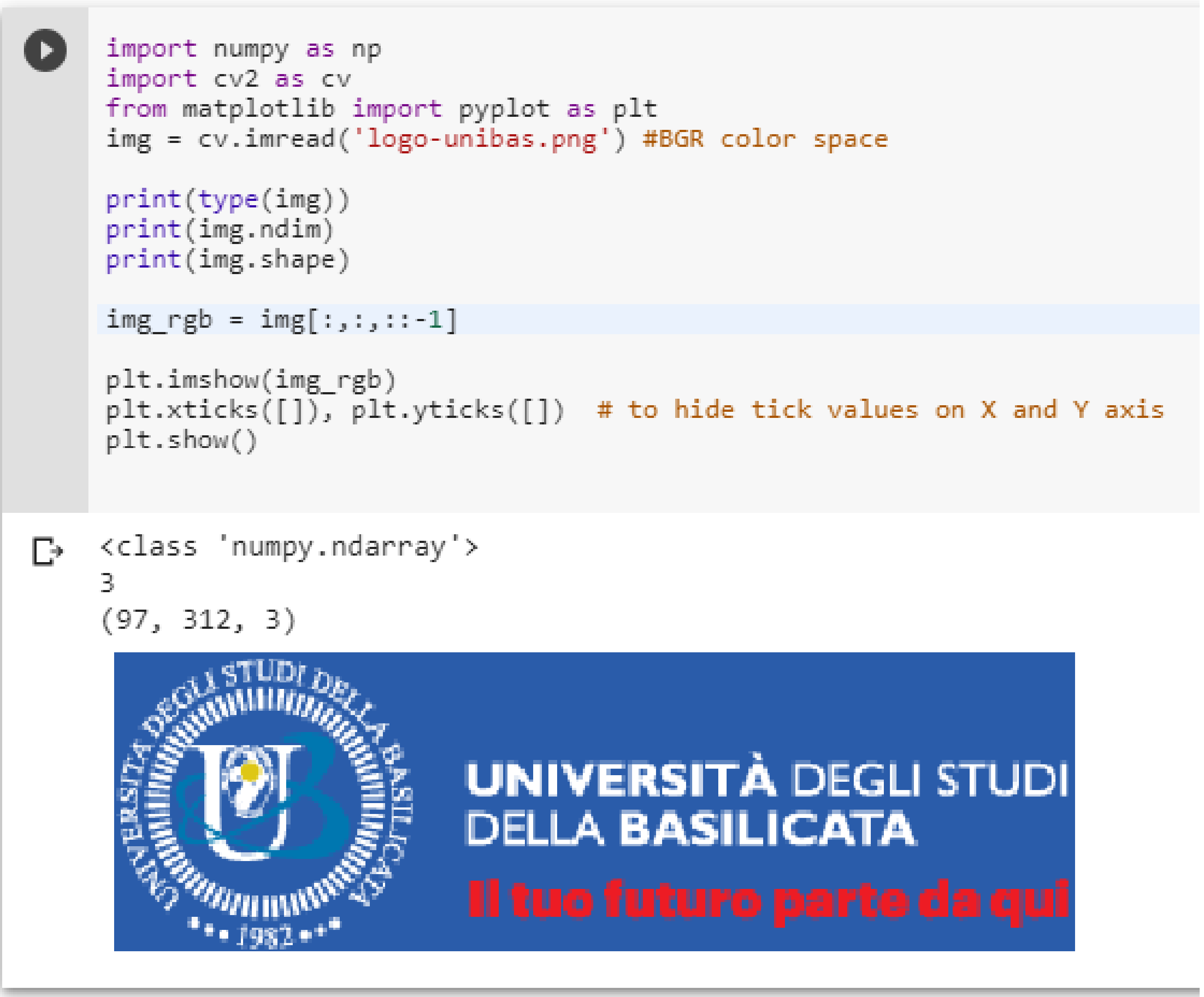


```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png')

print(type(img))
print(img.ndim)
print(img.shape)

plt.imshow(img)
plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis
plt.show()
```

```
3
(97, 312, 3)
<class 'numpy.ndarray'>
```

UNIVERSITAS STUDIORUM DEGLI STUDI DELLA BASILICATA
1982
UNIVERSITÀ DEGLI STUDI
DELLA BASILICATA
Il tuo futuro parte da qui

RGB visualization in Matplotlib

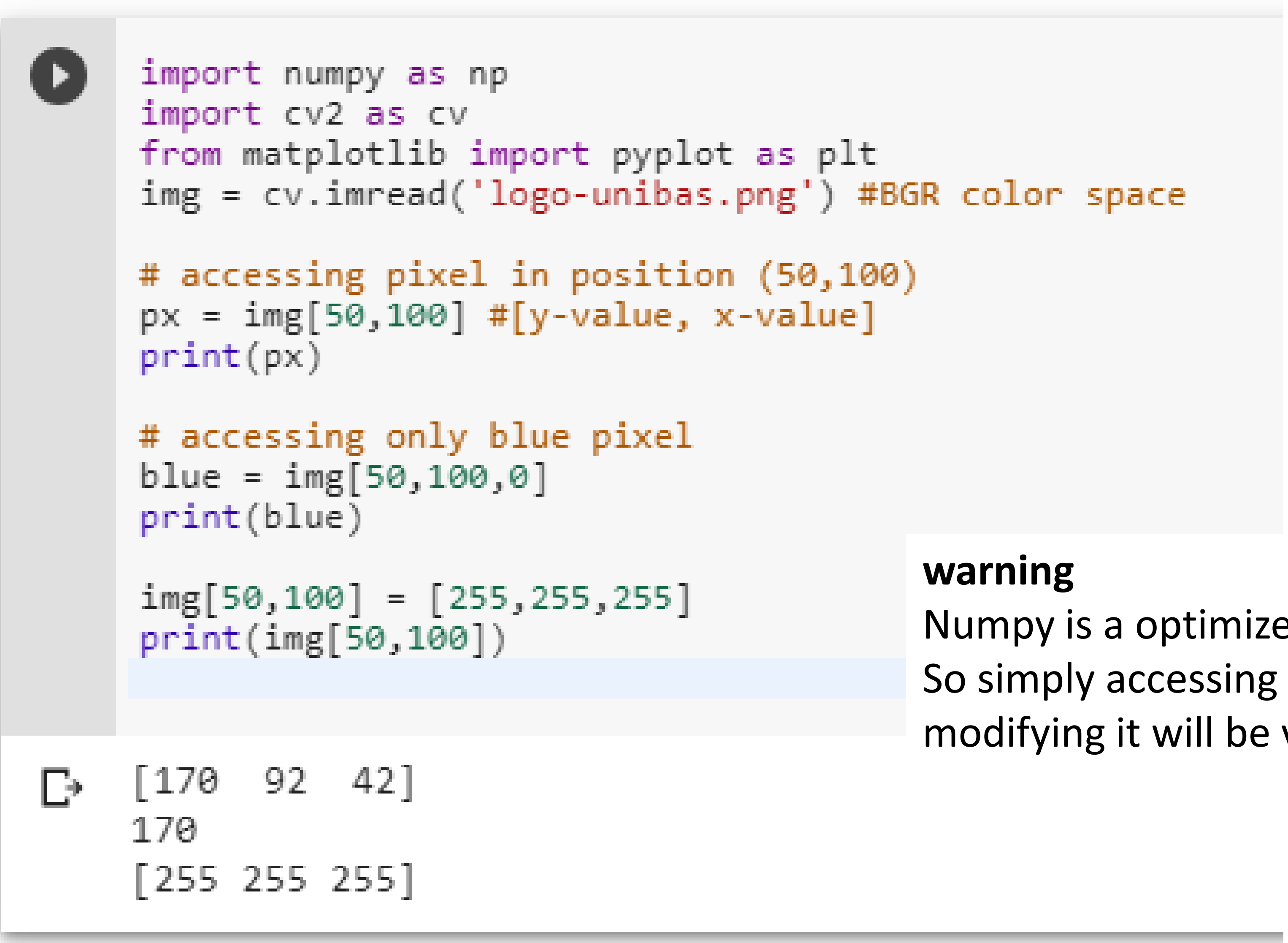


```python
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png') #BGR color space

print(type(img))
print(img.ndim)
print(img.shape)

img_rgb = img[:,:,::-1]

plt.imshow(img_rgb)
plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis
plt.show()
```

```
<class 'numpy.ndarray'>
3
(97, 312, 3)
```

UNIVERSITÀ DEGLI STUDI DELLA BASILICATA
UNIVERSITÀ DEGLI STUDI
DELLA BASILICATA
Il tuo futuro parte da qui
1982

Accessing and Modifying pixel values



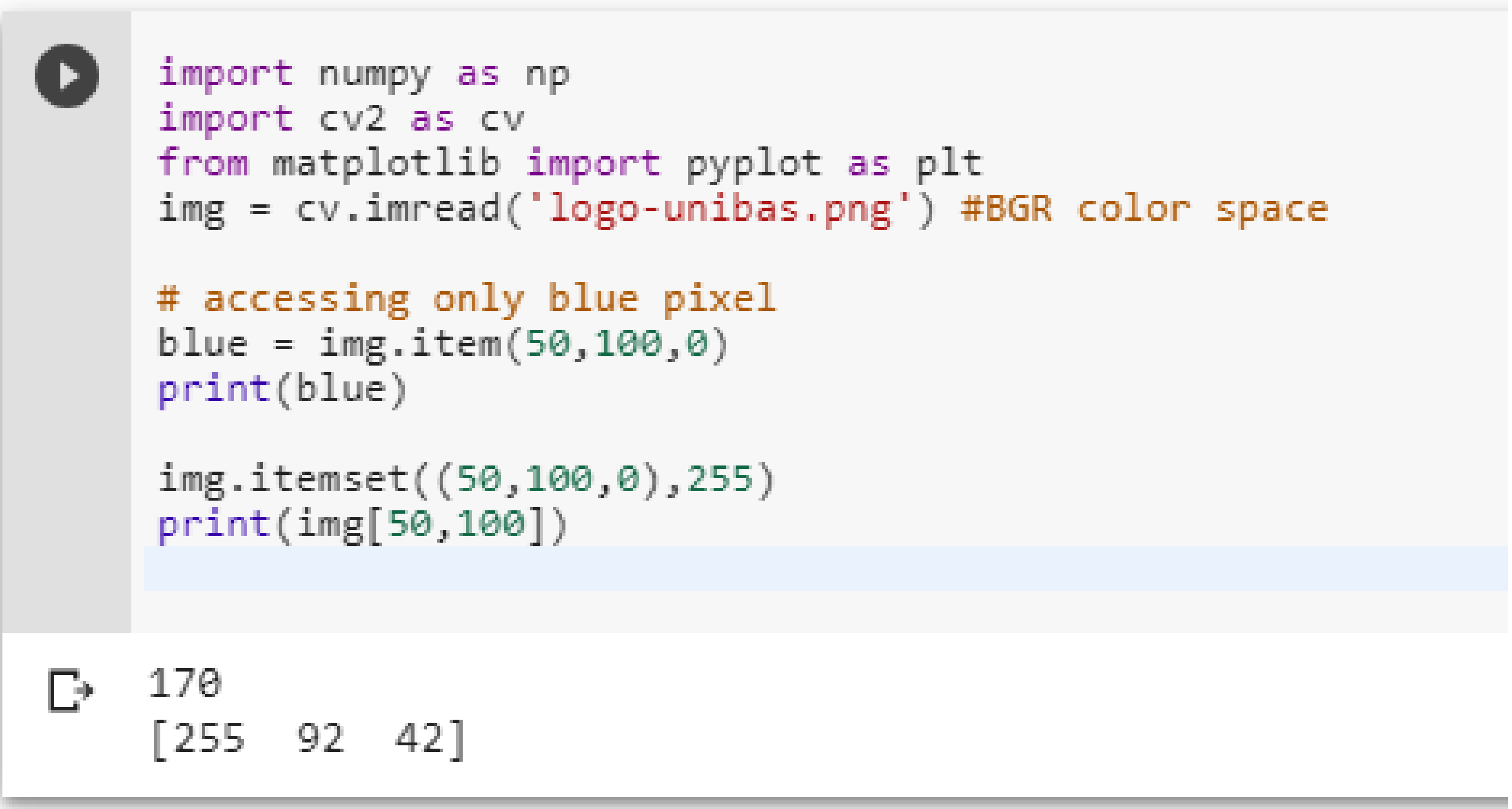
```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png') #BGR color space

# accessing pixel in position (50,100)
px = img[50,100] #[y-value, x-value]
print(px)

# accessing only blue pixel
blue = img[50,100,0]
print(blue)

img[50,100] = [255,255,255]
print(img[50,100])
```

```
[170  92  42]
170
[255 255 255]
```

warning

Numpy is a optimized library for fast array calculations. So simply accessing each and every pixel values and modifying it will be very slow and it is discouraged.

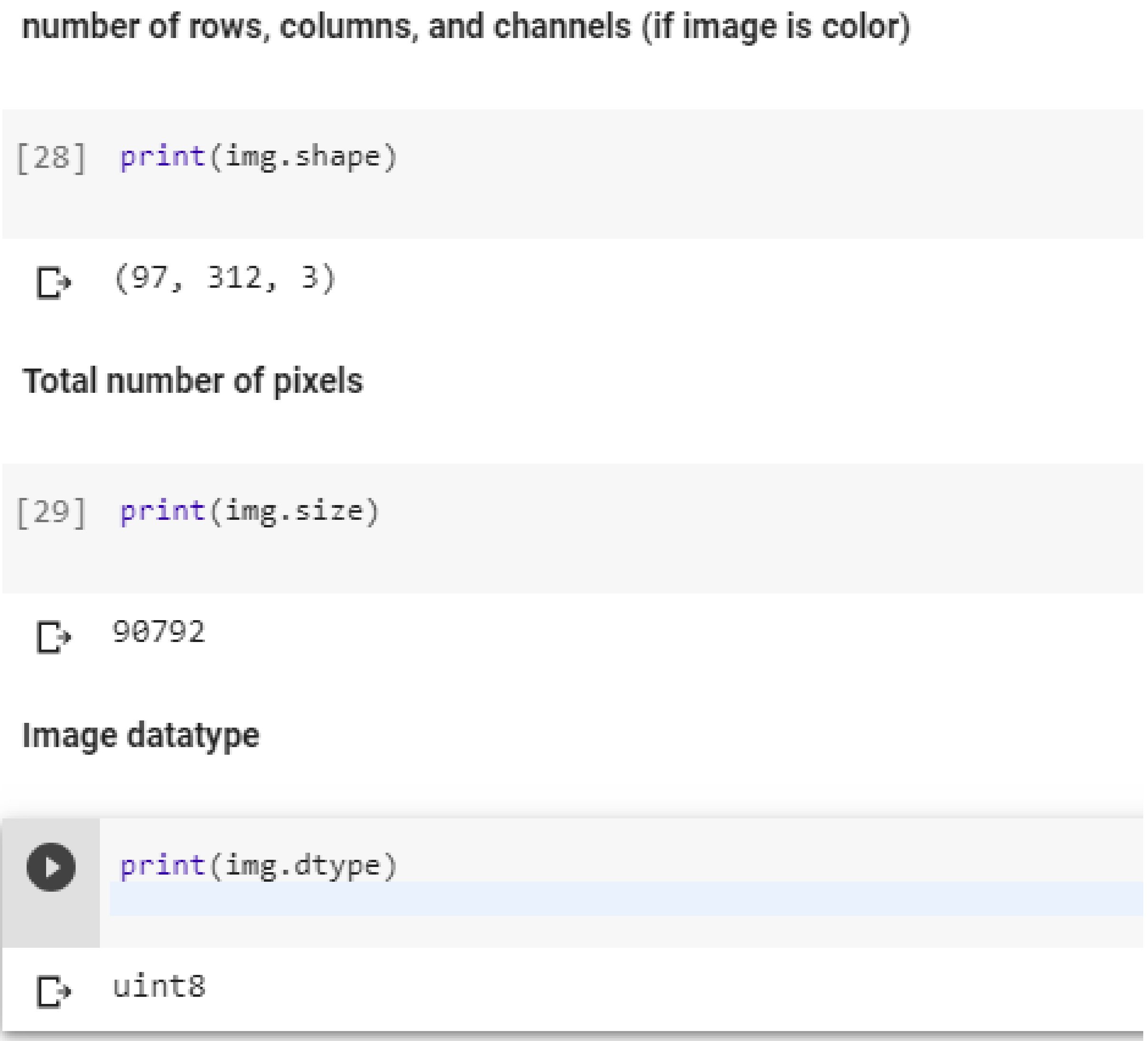
item e itemset



```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png') #BGR color space

# accessing only blue pixel
blue = img.item(50,100,0)
print(blue)

img.itemset((50,100,0),255)
print(img[50,100])
```

```
170
[255  92  42]
```

Accessing Image Properties



number of rows, columns, and channels (if image is color)

```
[28] print(img.shape)
```

```
(97, 312, 3)
```

Total number of pixels

```
[29] print(img.size)
```

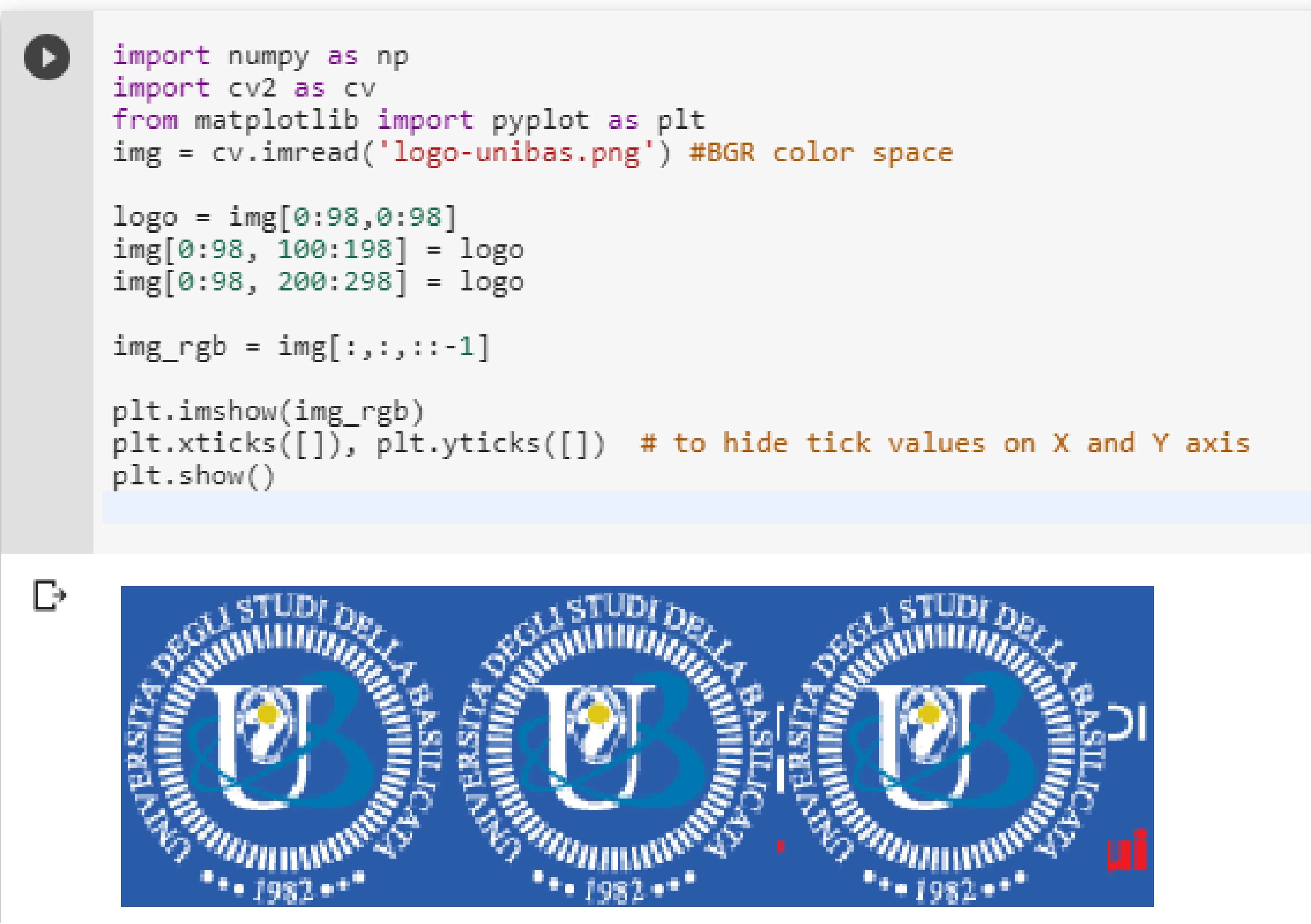
```
90792
```

Image datatype

```
print(img.dtype)
```

```
uint8
```

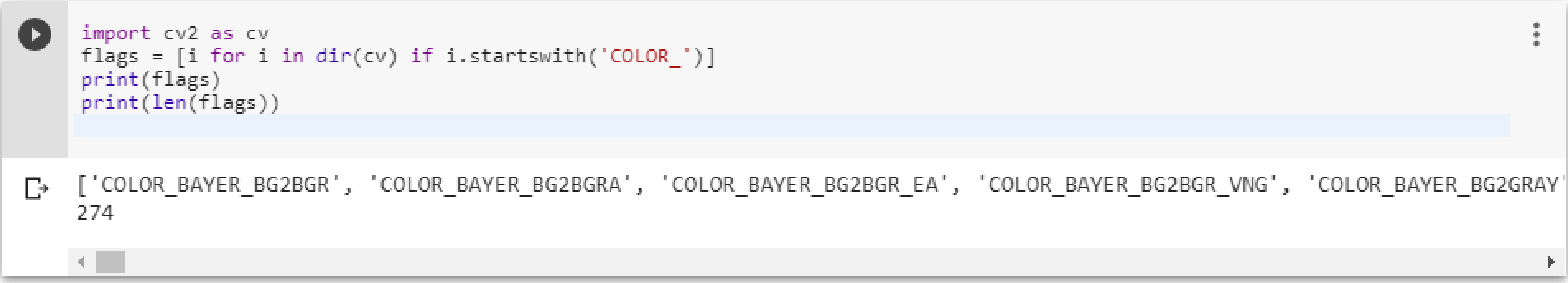
Image ROI



```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png') #BGR color space

logo = img[0:98,0:98]
img[0:98, 100:198] = logo
img[0:98, 200:298] = logo

img_rgb = img[:,:,::-1]

plt.imshow(img_rgb)
plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis
plt.show()
```

Changing Color-space

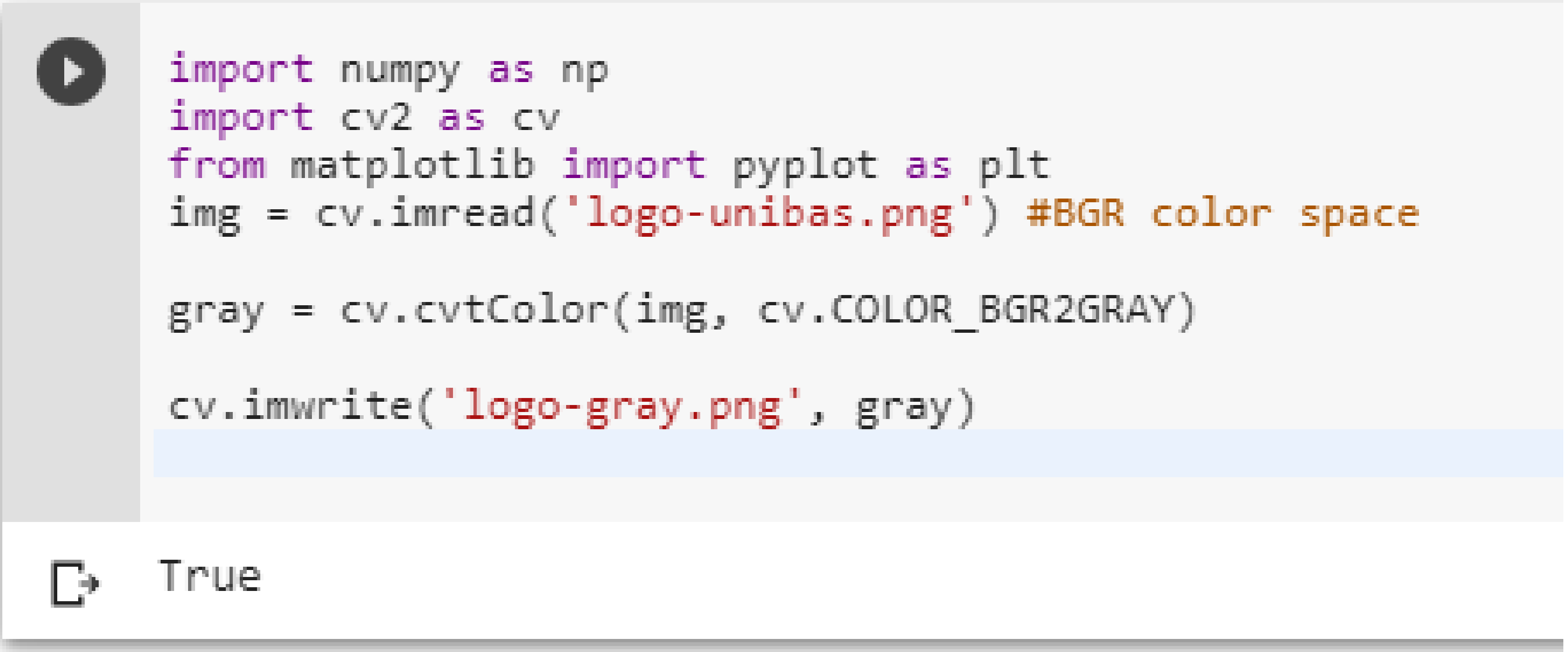
There are more than 150 color-space conversion methods available in OpenCV.



```
import cv2 as cv
flags = [i for i in dir(cv) if i.startswith('COLOR_')]
print(flags)
print(len(flags))
```

```
['COLOR_BAYER_BG2BGR', 'COLOR_BAYER_BG2BGRA', 'COLOR_BAYER_BG2BGR_EA', 'COLOR_BAYER_BG2BGR_VNG', 'COLOR_BAYER_BG2GRAY'
274
```

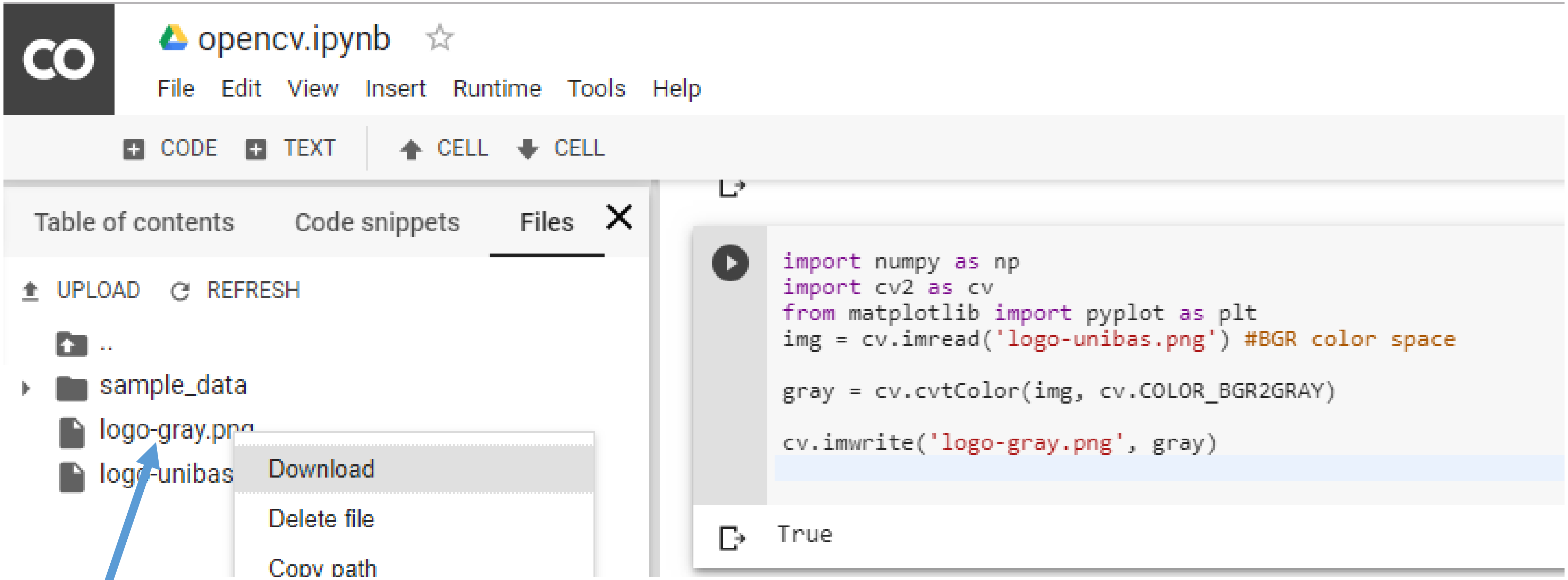
Changing Color-space



```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png') #BGR color space

gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)

cv.imwrite('logo-gray.png', gray)
```

```
True
```

Grayscale conversion



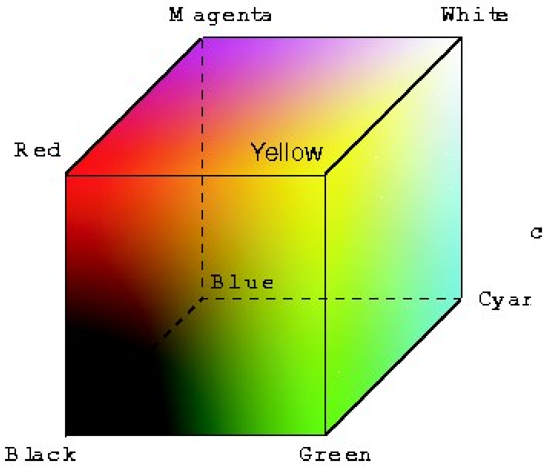
```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png') #BGR color space

gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)

cv.imwrite('logo-gray.png', gray)
```

```
True
```

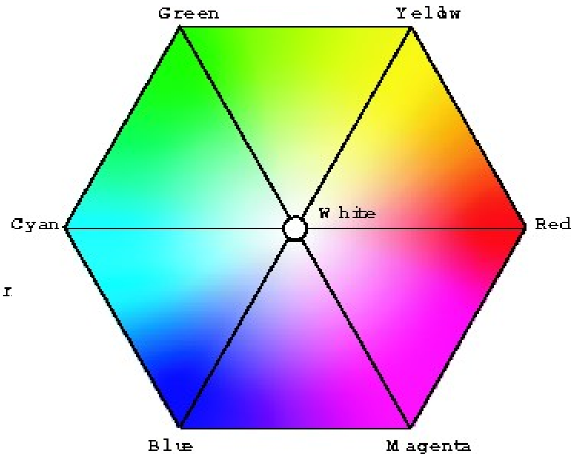
tasto destro del mouse

UNIVERSITÀ DEGLI STUDI DELLA BASILICATA
UNIVERSITÀ DEGLI STUDI DELLA BASILICATA
1982
Il tuo futuro parte da qui

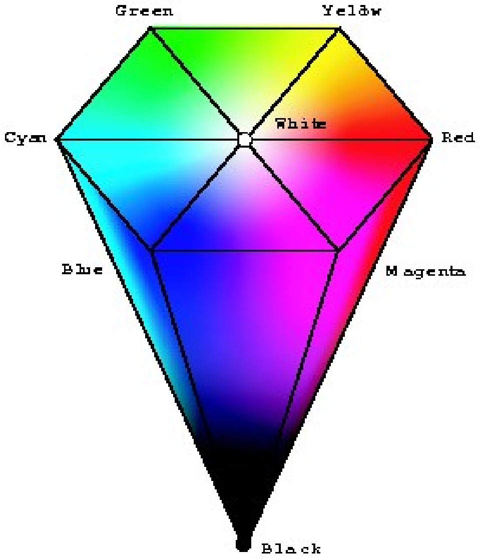
HSV color-space



RGB cube



HSV top view

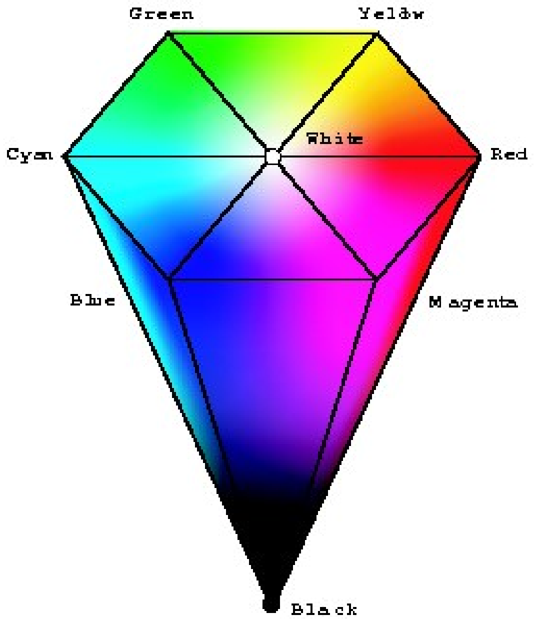


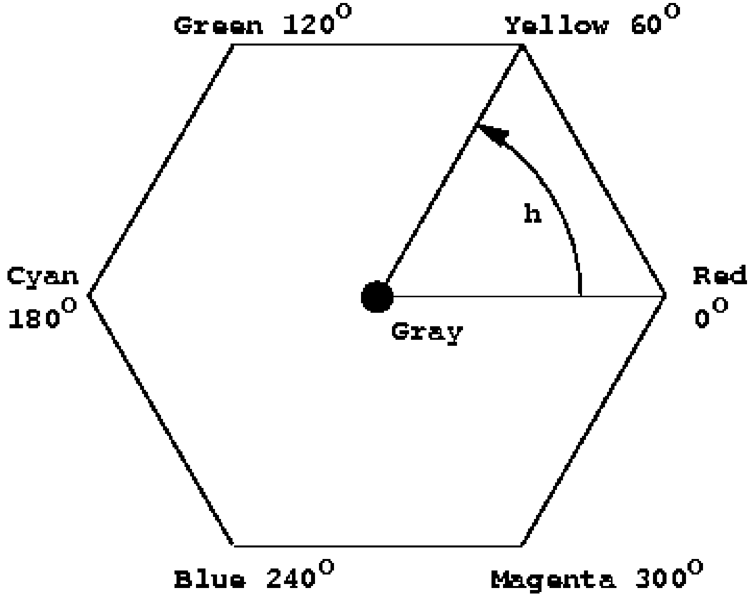
HSV cone

HSV is a projection of the RGB space

Source: Donald House

Hue



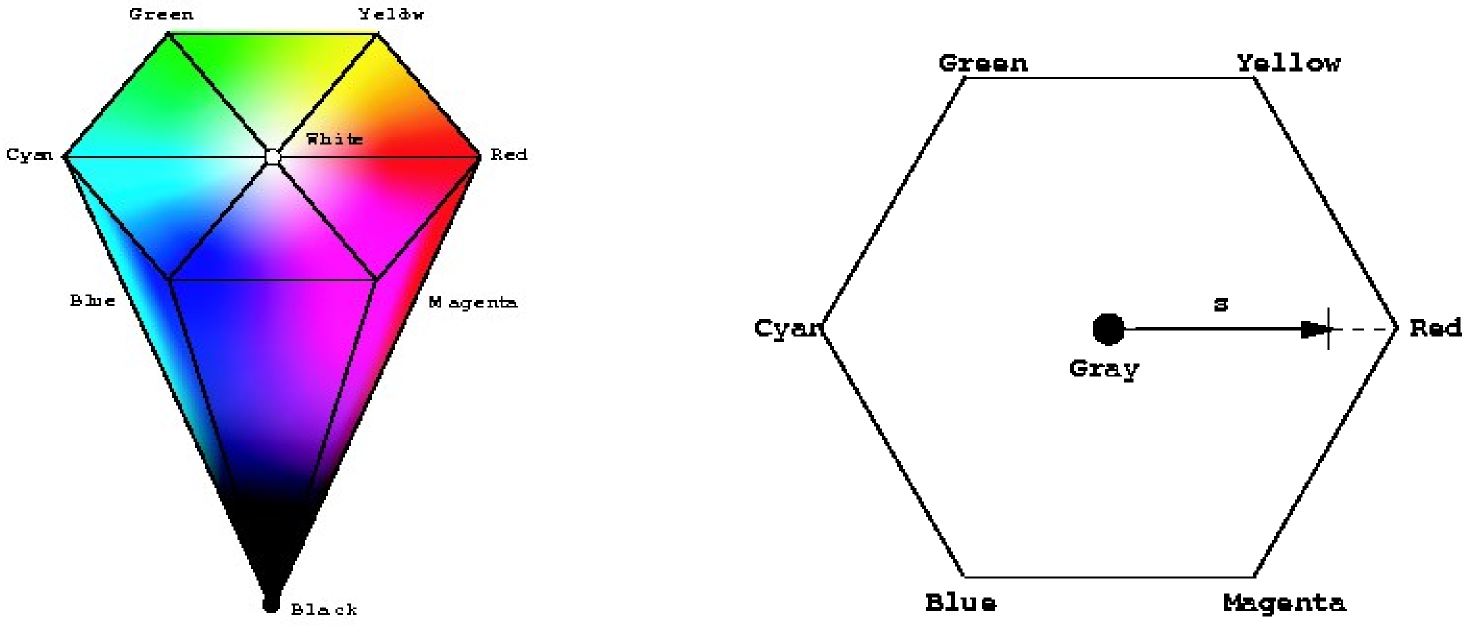


Hue, an angular measure (0 … 360)

Hue range is [0,179] in OpenCV

Source: Donald House

Saturation

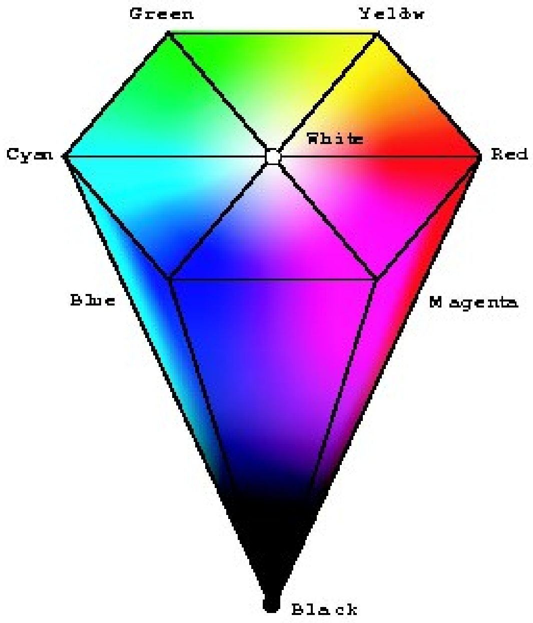


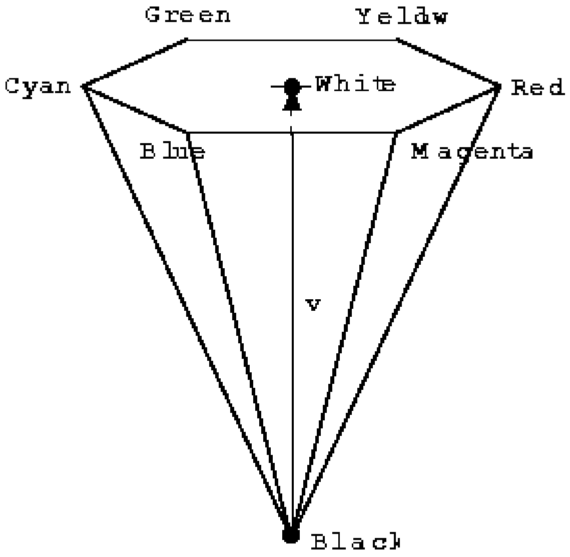
Saturation, a fractional measure (0.0 … 1.0)

Saturation range is [0,255] in OpenCV

Source: Donald House

Value



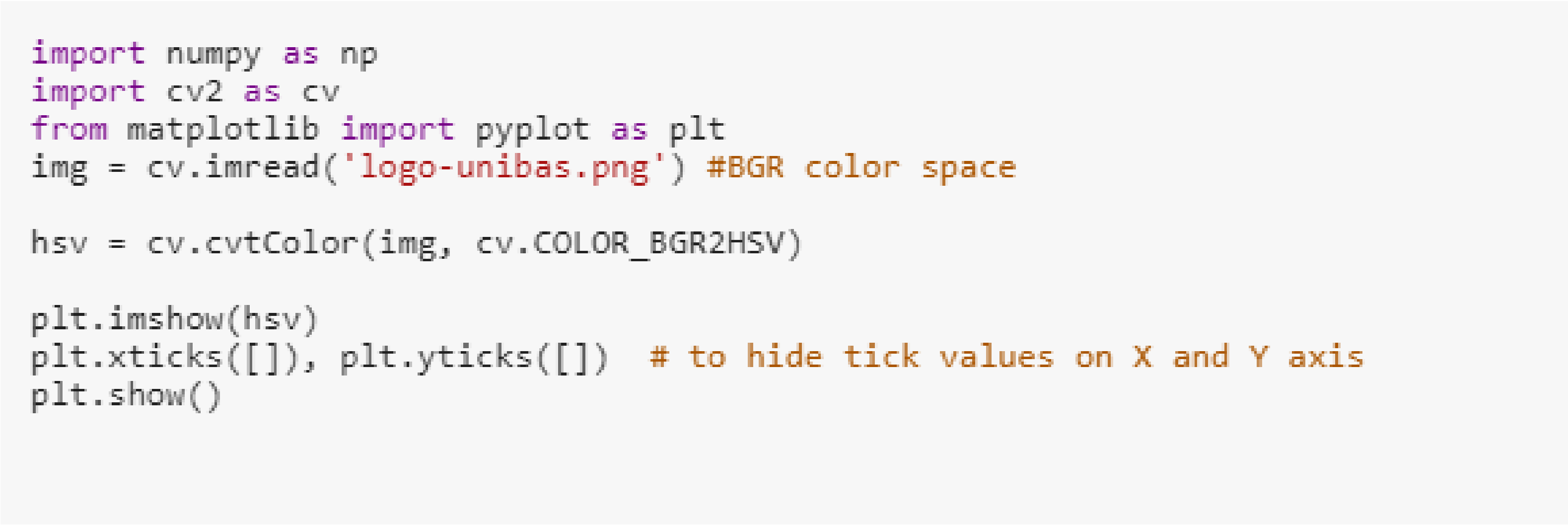


Value, a fractional measure (0.0 … 1.0)

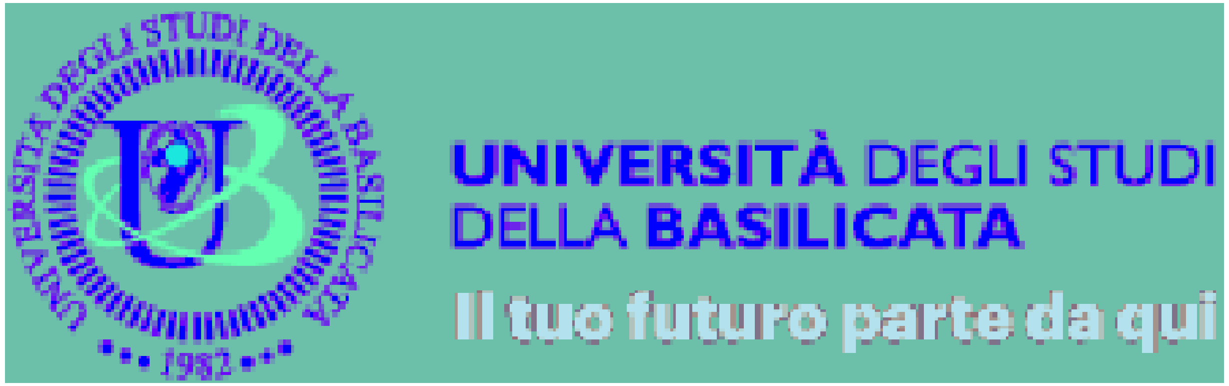
Value range is [0,255] in OpenCV

Source: Donald House

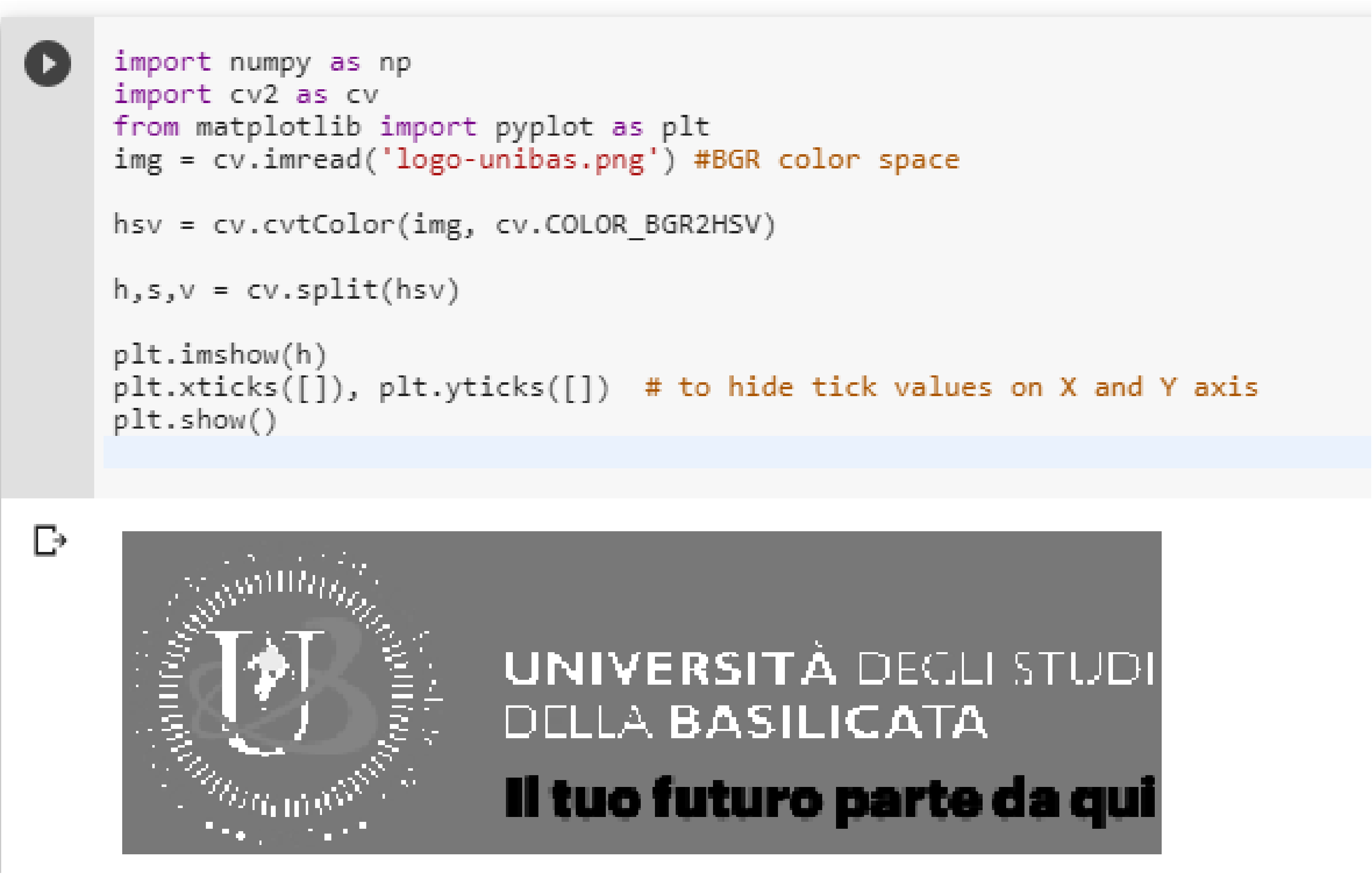
HSV conversion



```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png') #BGR color space

hsv = cv.cvtColor(img, cv.COLOR_BGR2HSV)

plt.imshow(hsv)
plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis
plt.show()
```

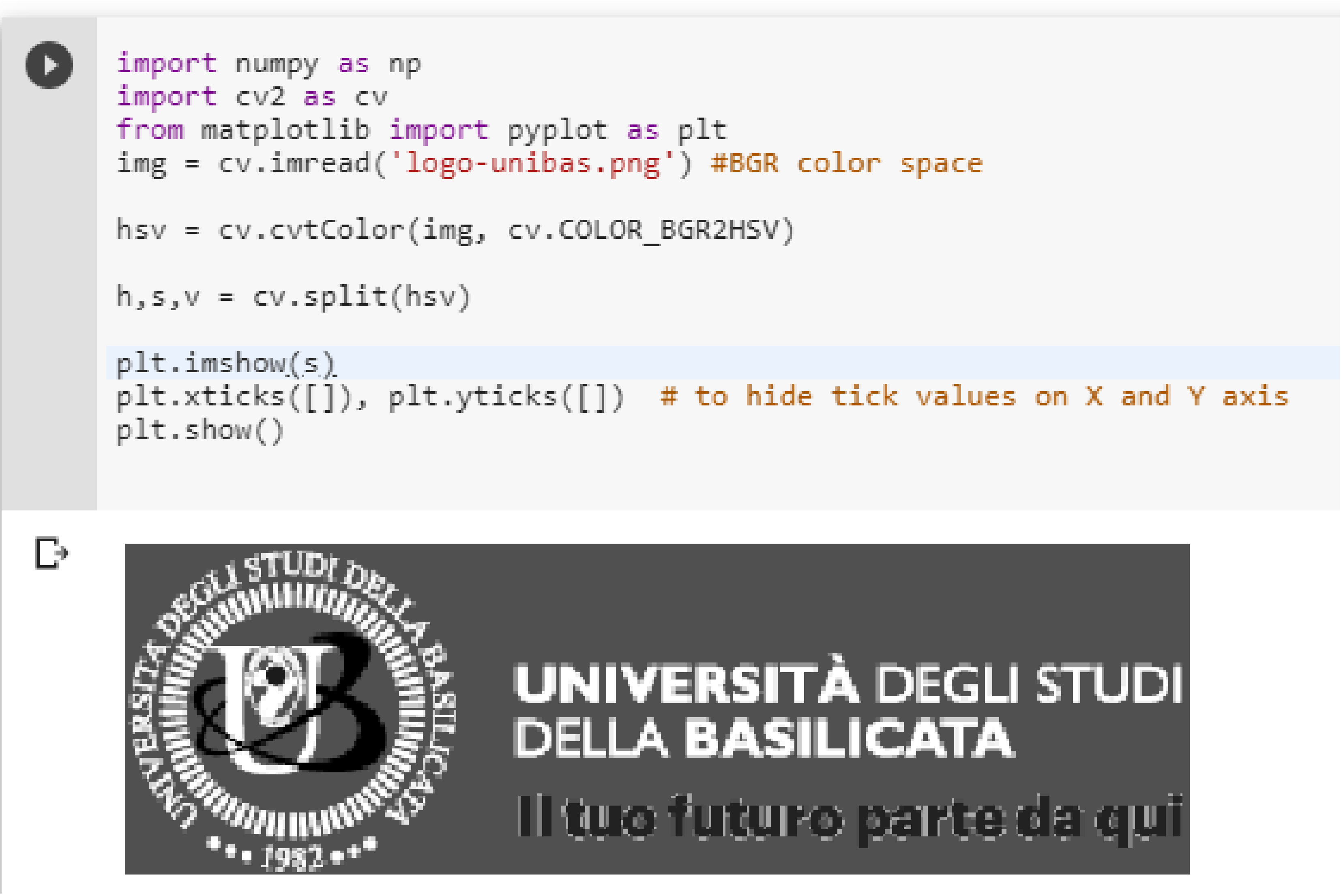

UNIVERSITÀ DEGLI STUDI DELLA BASILICATA
1982
UNIVERSITÀ DEGLI STUDI
DELLA BASILICATA
Il tuo futuro parte da qui

Split



```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png') #BGR color space

hsv = cv.cvtColor(img, cv.COLOR_BGR2HSV)

h,s,v = cv.split(hsv)

plt.imshow(h)
plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis
plt.show()
```

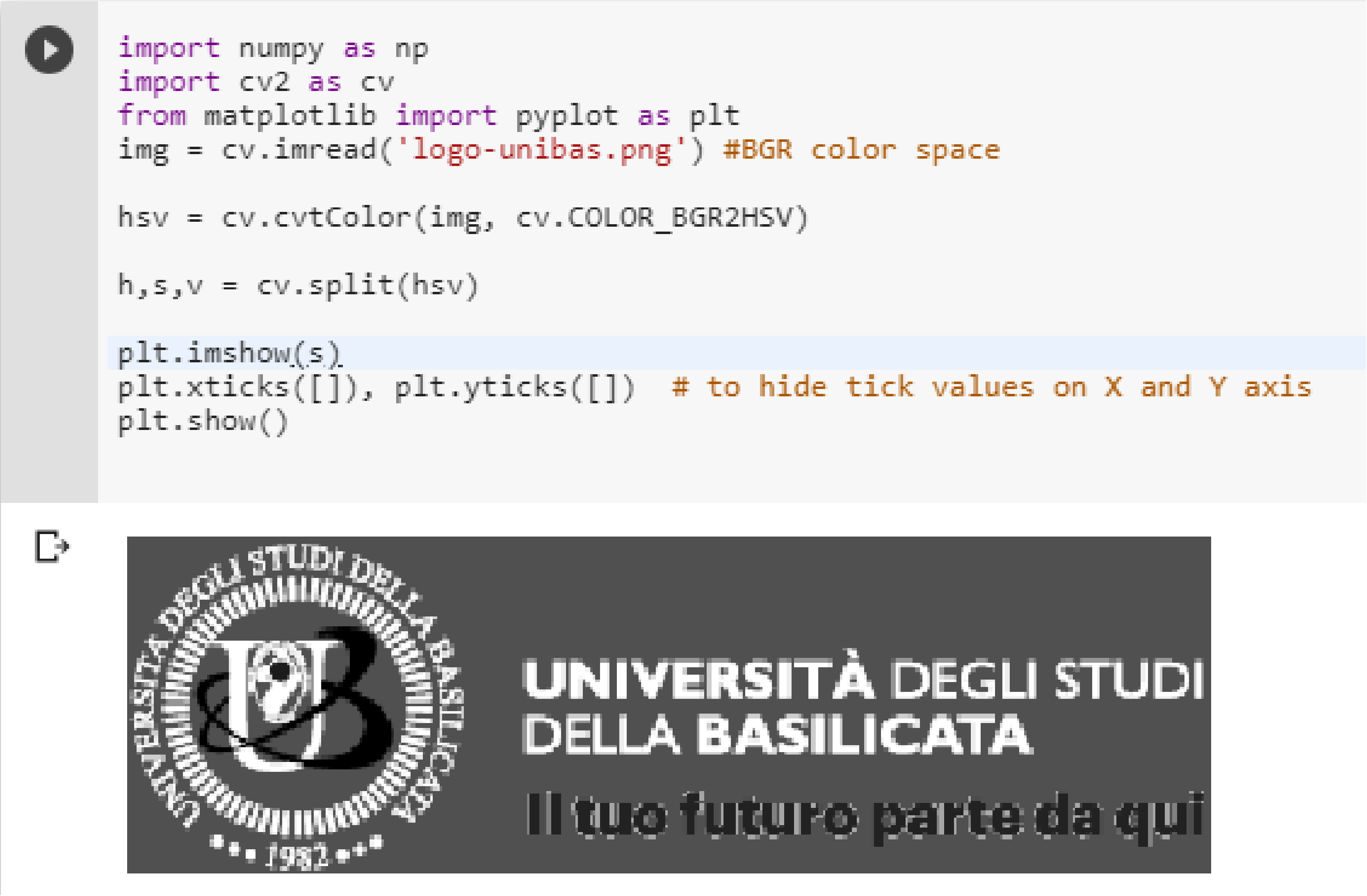
Split – Saturation channel



```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png') #BGR color space

hsv = cv.cvtColor(img, cv.COLOR_BGR2HSV)

h,s,v = cv.split(hsv)

plt.imshow(s)
plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis
plt.show()
```

UNIVERSITÀ DEGLI STUDI DELLA BASILICATA
1982
UNIVERSITÀ DEGLI STUDI
DELLA BASILICATA
Il tuo futuro parte da qui

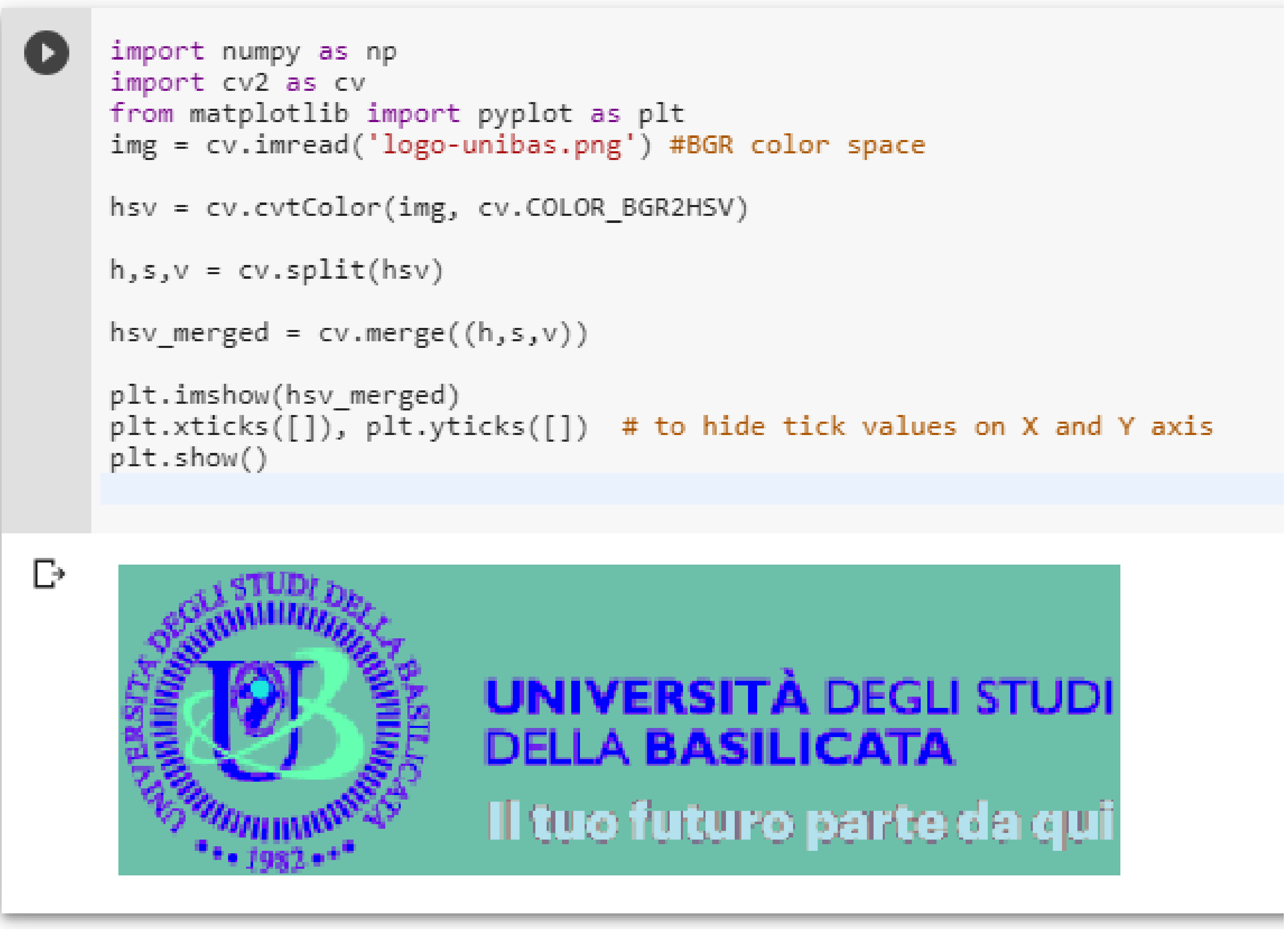
Split – Value channel



```python
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png') #BGR color space

hsv = cv.cvtColor(img, cv.COLOR_BGR2HSV)

h,s,v = cv.split(hsv)

plt.imshow(s)
plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis
plt.show()
```

UNIVERSITÀ DEGLI STUDI DELLA BASILICATA
1982
UNIVERSITÀ DEGLI STUDI
DELLA BASILICATA
Il tuo futuro parte da qui

Merge

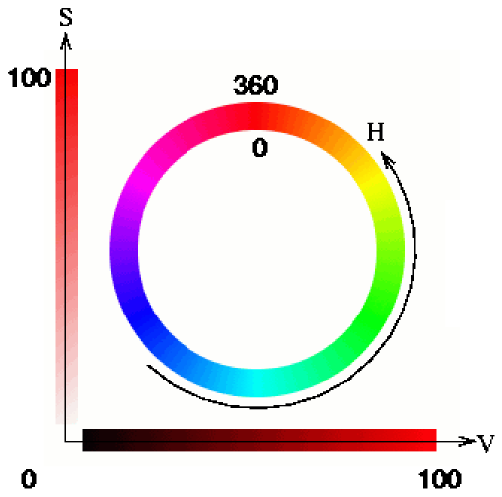


```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
img = cv.imread('logo-unibas.png') #BGR color space

hsv = cv.cvtColor(img, cv.COLOR_BGR2HSV)

h,s,v = cv.split(hsv)

hsv_merged = cv.merge((h,s,v))

plt.imshow(hsv_merged)
plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis
plt.show()
```

UNIVERSITÀ DEGLI STUDI
DELLA BASILICATA
Il tuo futuro parte da qui
UNIVERSITÀ DEGLI STUDI DELLA BASILICATA
1982

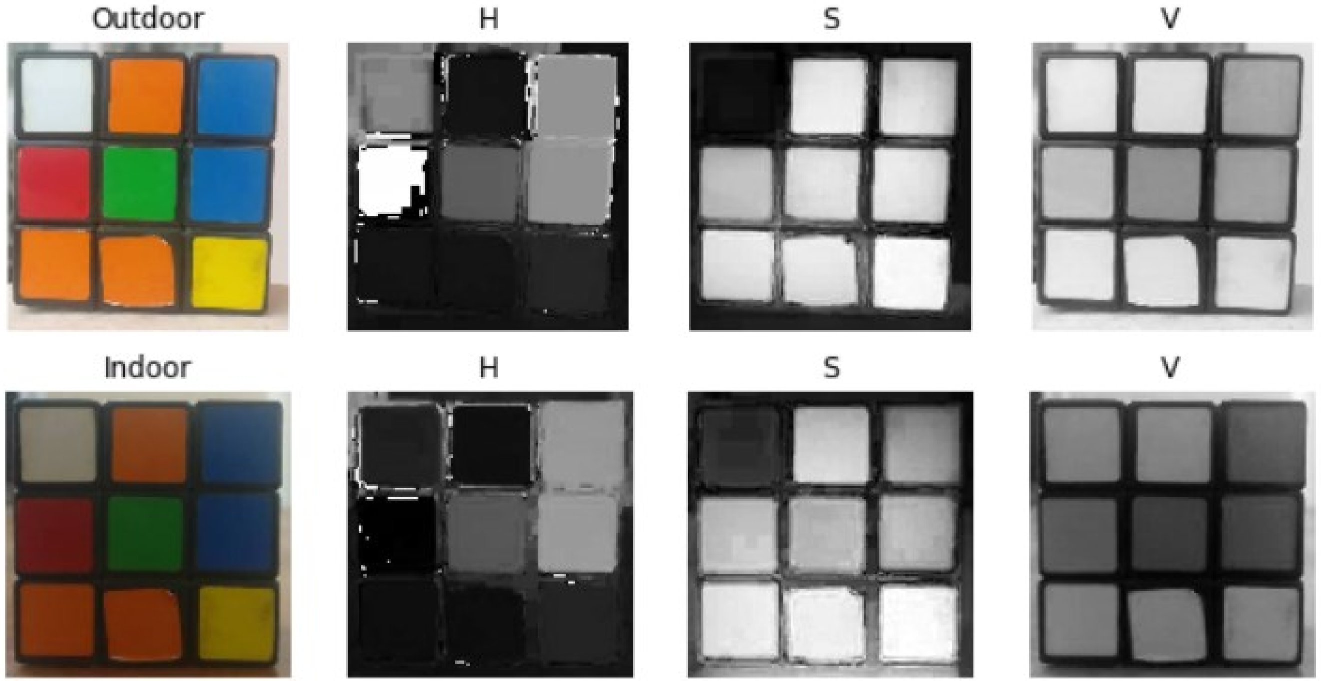
HSV color space

http://tinf2.vub.ac.be/~dvermeir/manual/gimp/Grokking-the-GIMP-v1.0/node51.html



The HSV color space has the following three components

1. H – Hue (Dominant Wavelength)
2. S – Saturation (Purity/shades of the color)
3. V – Value (Intensity)

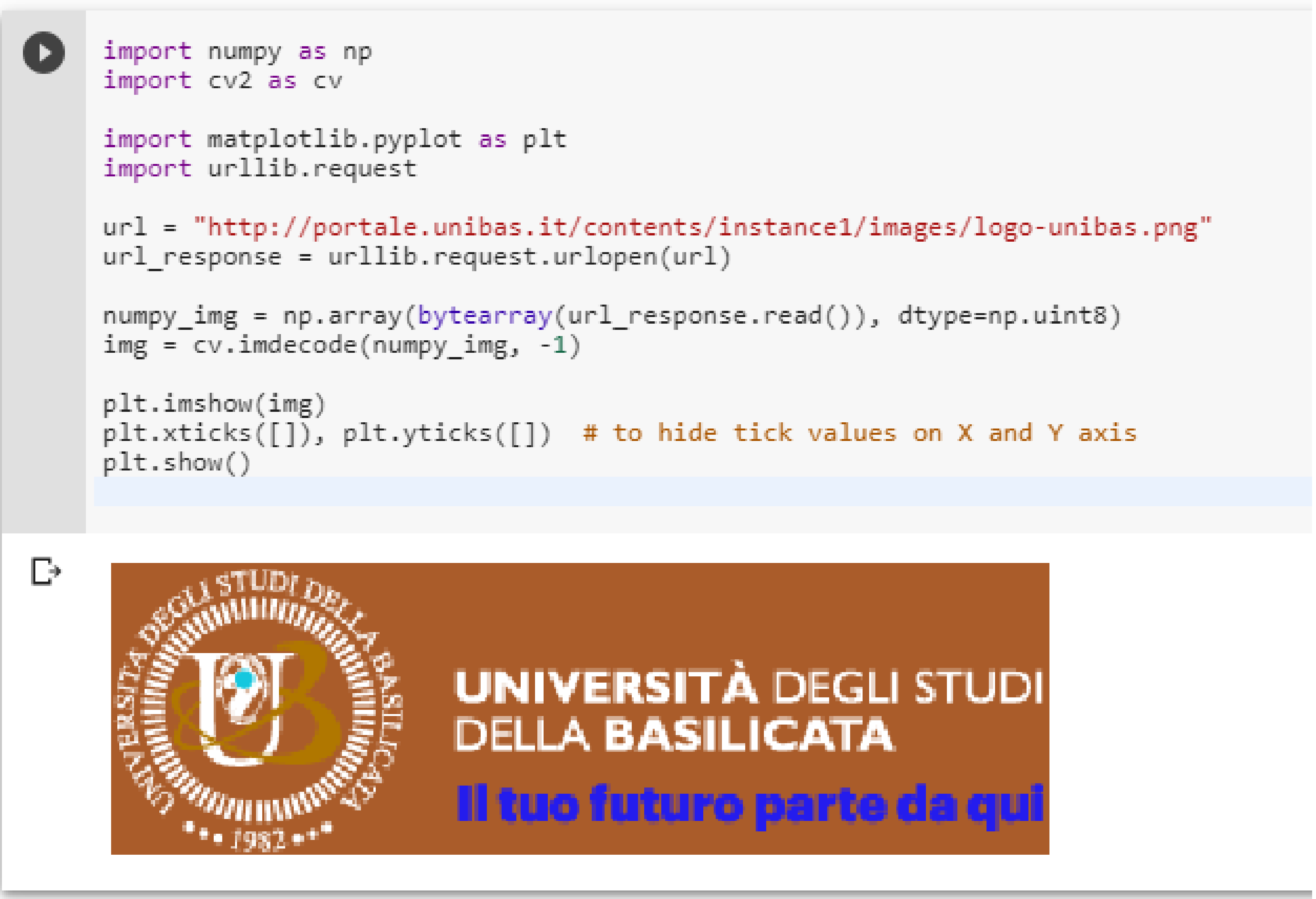


Observations

- The H Component is very similar in both the images which indicates the color information is intact even under illumination changes
- The S component is also very similar in both images
- The V Component captures the amount of light falling on it thus it changes due to illumination changes

https://www.learnopencv.com/color-spaces-in-opencv-cpp-python/

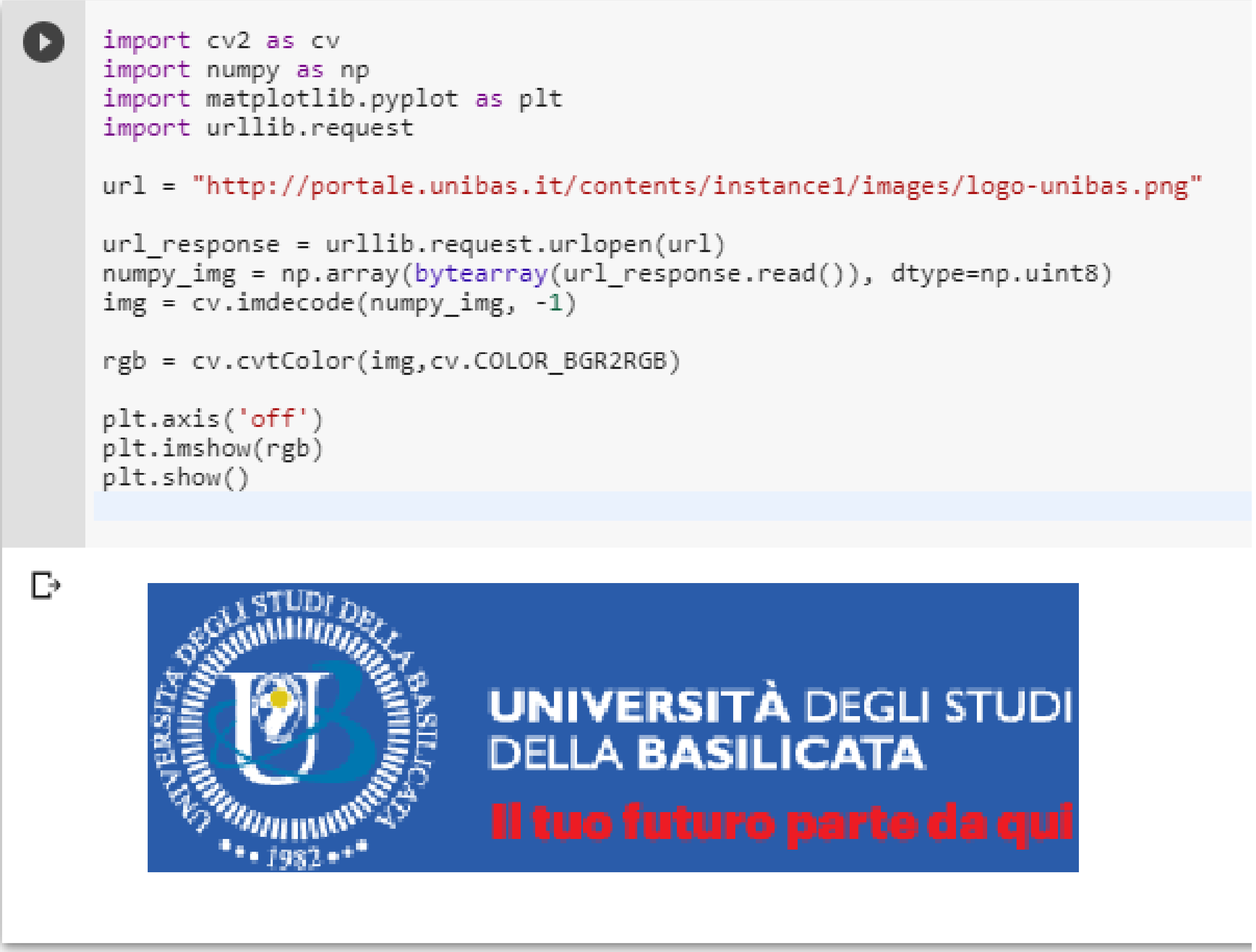
Read an image from URL



```python
import numpy as np
import cv2 as cv

import matplotlib.pyplot as plt
import urllib.request

url = "http://portale.unibas.it/contents/instance1/images/logo-unibas.png"
url_response = urllib.request.urlopen(url)

numpy_img = np.array(bytearray(url_response.read()), dtype=np.uint8)
img = cv.imdecode(numpy_img, -1)

plt.imshow(img)
plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis
plt.show()
```

UNIVERSITÀ DEGLI STUDI DELLA BASILICATA
1982
UNIVERSITÀ DEGLI STUDI
DELLA BASILICATA
Il tuo futuro parte da qui

Read an image from URL



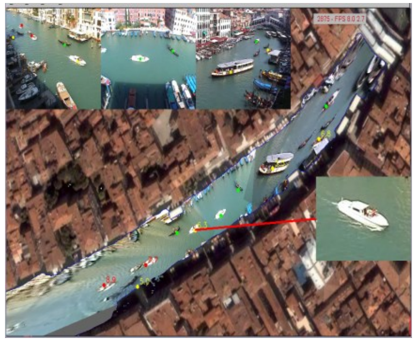
```
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt
import urllib.request

url = "http://portale.unibas.it/contents/instance1/images/logo-unibas.png"

url_response = urllib.request.urlopen(url)
numpy_img = np.array(bytearray(url_response.read()), dtype=np.uint8)
img = cv.imdecode(numpy_img, -1)

rgb = cv.cvtColor(img,cv.COLOR_BGR2RGB)

plt.axis('off')
plt.imshow(rgb)
plt.show()
```

UNIVERSITÀ DEGLI STUDI DELLA BASILICATA
1982
UNIVERSITÀ DEGLI STUDI
DELLA BASILICATA
Il tuo futuro parte da qui

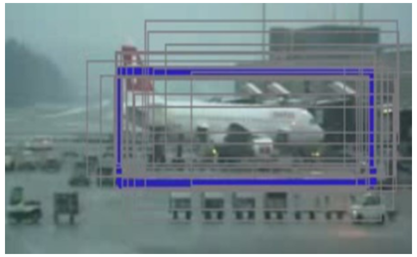
UNIVERSITÀ DEGLI STUDI DELLA BASILICATA

Corso di Sistemi Informativi
A.A. 2018/19

Docente
Domenico Daniele Bloisi



OpenCV (Python)



Aprile 2019