

EMBEDDED VISION DESIGN 3

BASIC PRINCIPLES

HANDS-ON

JEROEN VEEN



HAN_UNIVERSITY
OF APPLIED SCIENCES

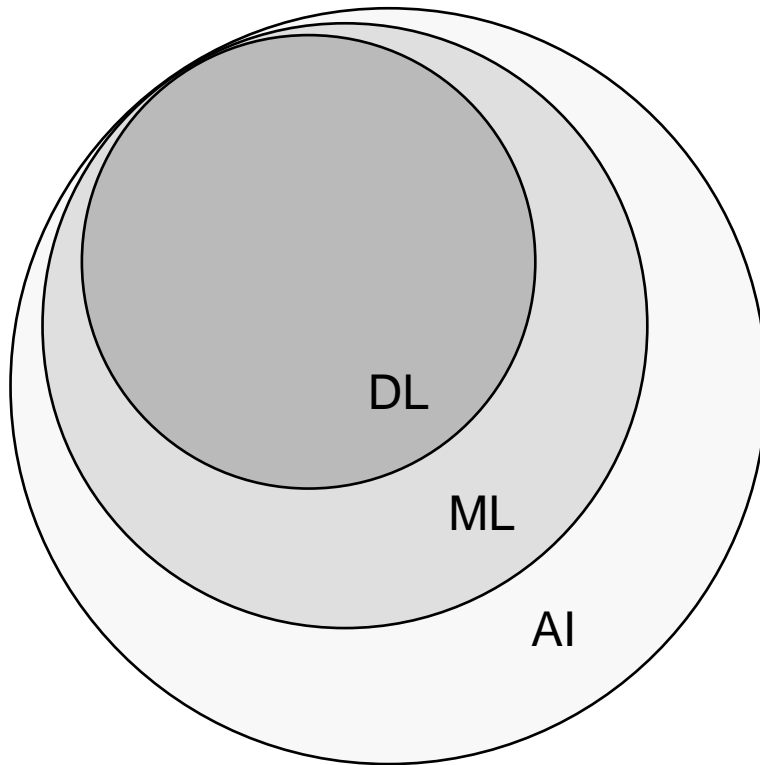
CONTENTS

- Introduction
- Workshop organization
- Assignment introduction
- ML portfolio template
- How to start? Conditioned acquisition and segmentation
- Set-up Raspberry Pi with OpenCV and sci-kit learn

WHAT IS MACHINE LEARNING?

- Human vs machine learning?
- Machines can perform predictive analytics on large amounts of data far faster than humans
- Machines maximize performance on a certain task
Typically function approximations
- Learning does not imply intelligence
if a machine can learn it is not necessarily aware

DEFINING AI, DL & ML

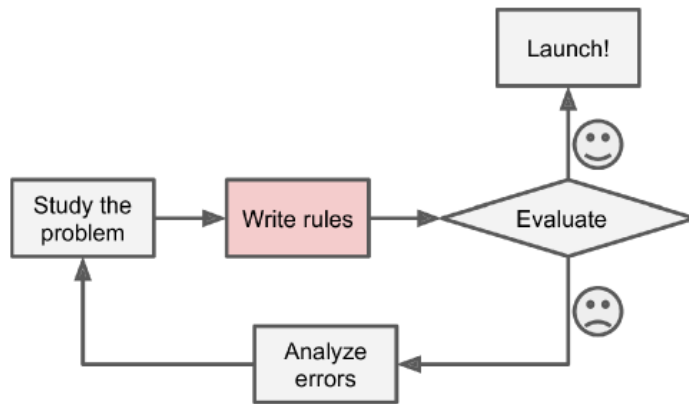


- Strong AI vs Applied AI
- Cognitive replication
- Rational process

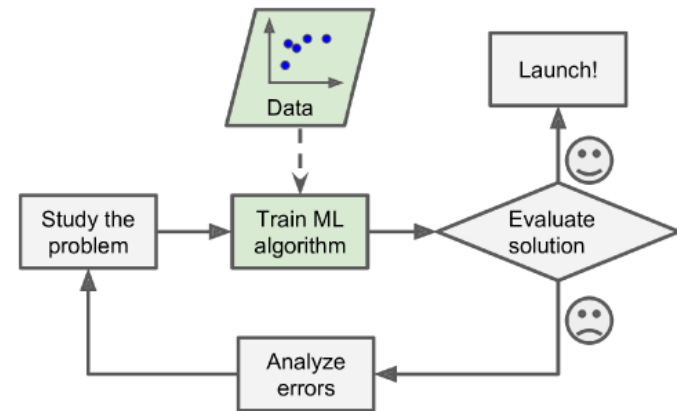
Machine learning

- Performs predictive analysis
- Just fancy math & pattern matching

WHY MACHINE LEARNING?



Traditional approach



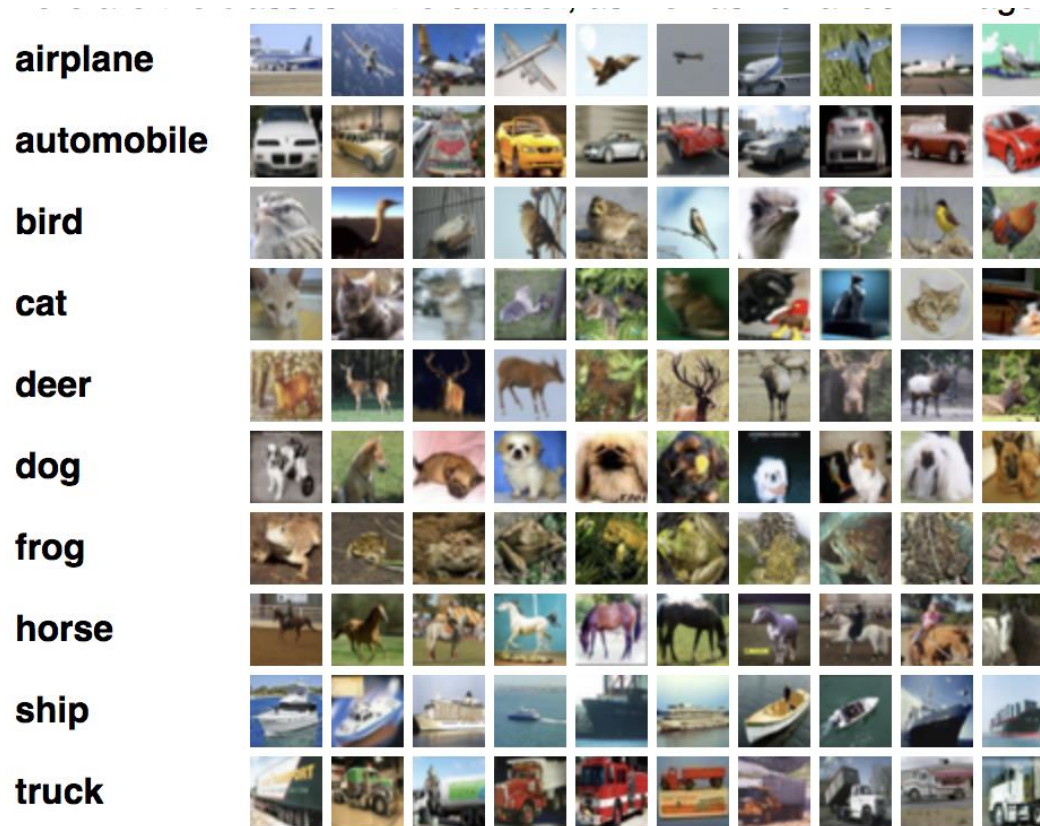
ML approach

Source: Géron, ISBN: 9781492032632

- Tackle problems for which existing solutions require a lot of fine-tuning or long lists of rules
- Deal with fluctuating environments by adapting to new data.
- Getting insights about complex problems and large amounts of data.

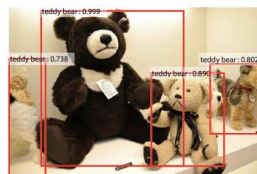
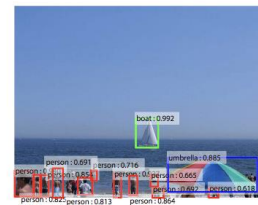
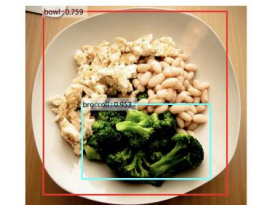
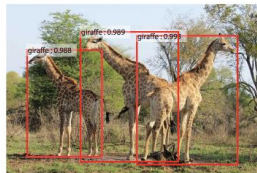
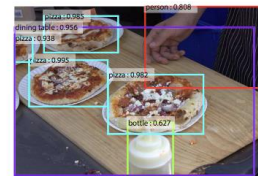
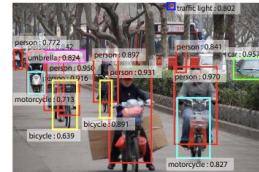
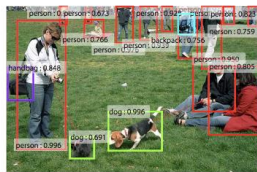
EXAMPLES OF MACHINE LEARNING

- Classification

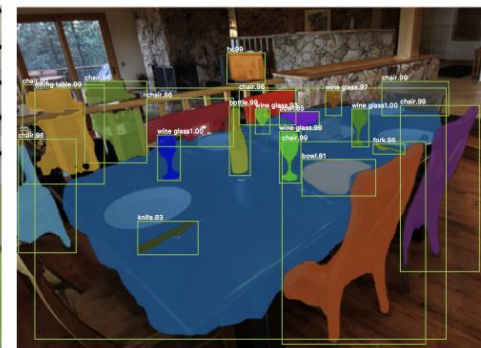
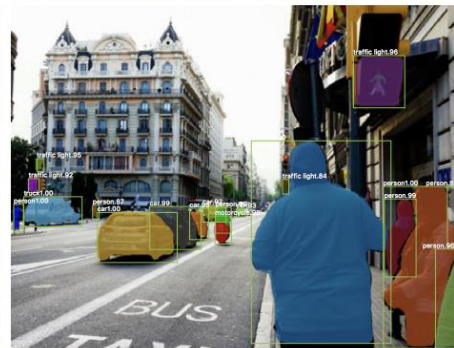
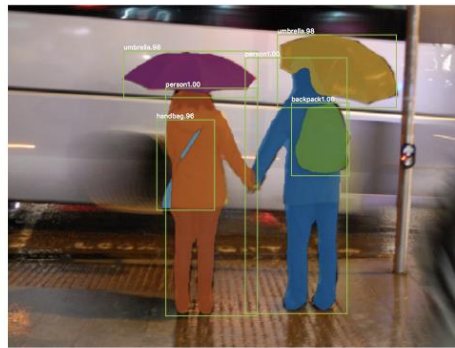
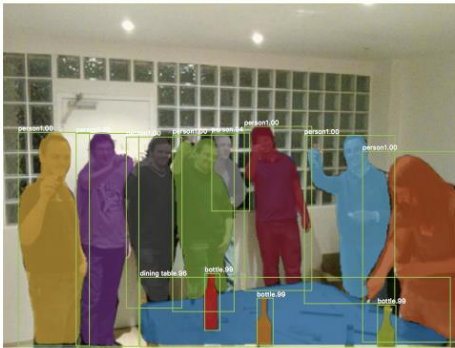


EXAMPLES OF MACHINE LEARNING

- Object detection

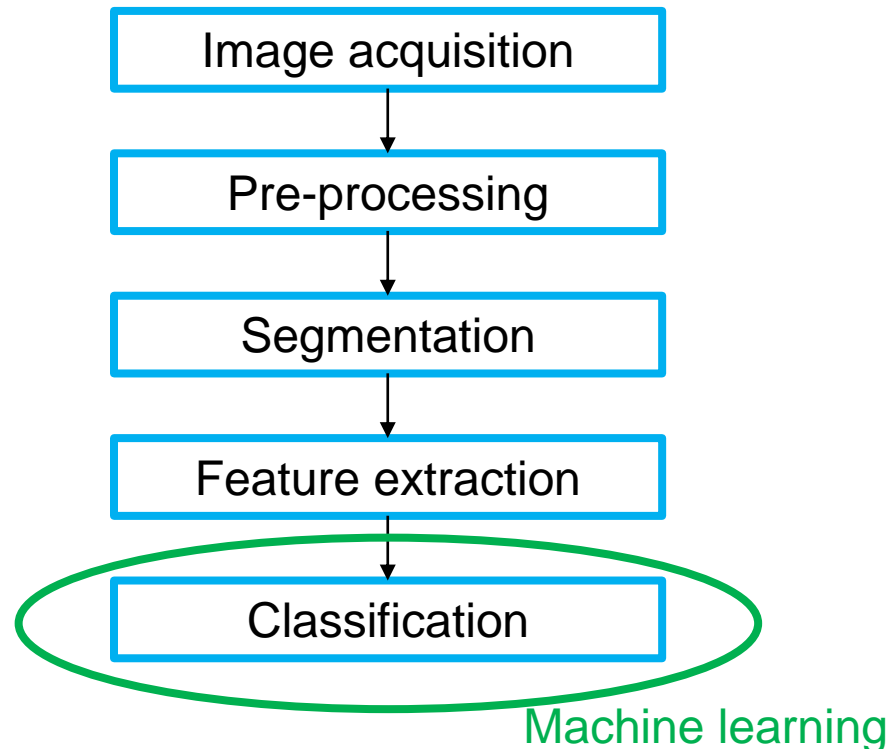


- segmentation



SHORT-CUT TO CLASSIFICATION

- Classical image processing



EVD3 ASSIGNMENTS

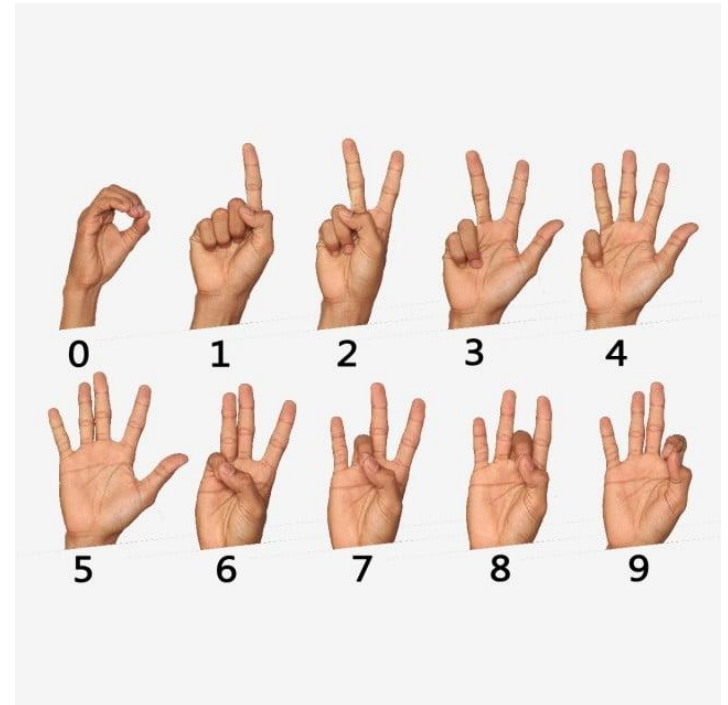
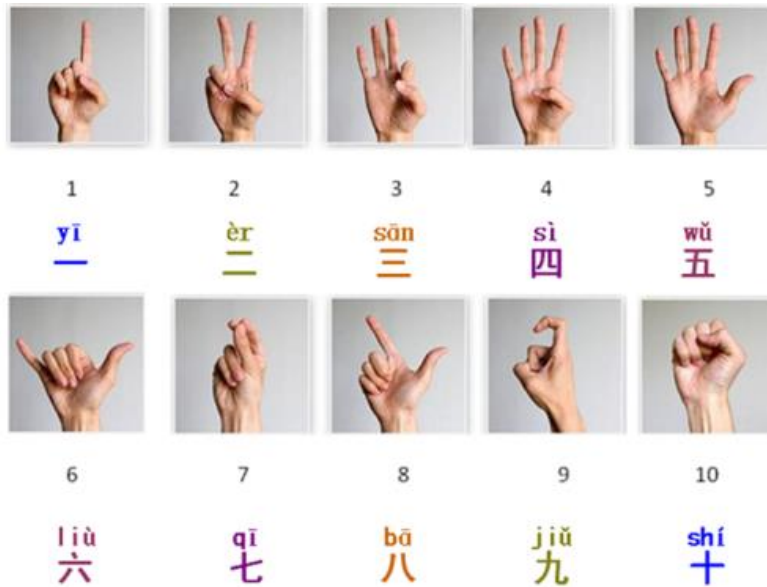
- A project team will consist of 3 students.
- Portfolio building using template
- Deliver intermediate results via HandIN
- Template and schedule on MS Teams

HAND GESTURE CLASSIFICATION

- E.g. sign language, rock-paper-scissors
- Min. 3 classes + unknown
- Pick silhouette gestures
- Alternatively, find a simple case within your main project, e.g. objects in autonomous robot
- Term1: solve with ML
- Term2: solve with DL



EXAMPLES

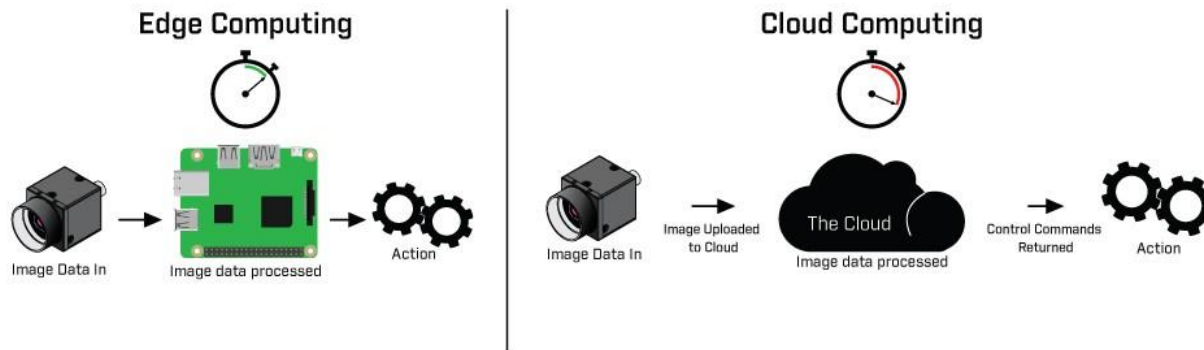


ML PORTFOLIO TEMPLATE

1	INTRODUCTION	3
2	PROBLEM STATEMENT	4
3	DATA ACQUISITION AND EXPLORATION.....	5
4	MODEL SELECTION AND TRAINING	6
5	DEPLOY AND TEST	7
6	CONCLUSION	8
7	REFERENCES	9
	CODE APPENDICES	10

TRAINING AND DEPLOYMENT

- Data exploration, model selection and training on embedded device or PC or even in the cloud (e.g. Google Colaboratory, Microsoft Azure)
- Data acquisition, model deployment and prediction testing on an embedded device, such as Raspberry Pi (sort of...)



CONDITIONED ACQUISITION

- Set-up image acquisition such that segmentation is easy
- Build a script to quickly collect >100 example images per class
- Study quality (and diversity) of your dataset
- Next week we'll discuss an example script

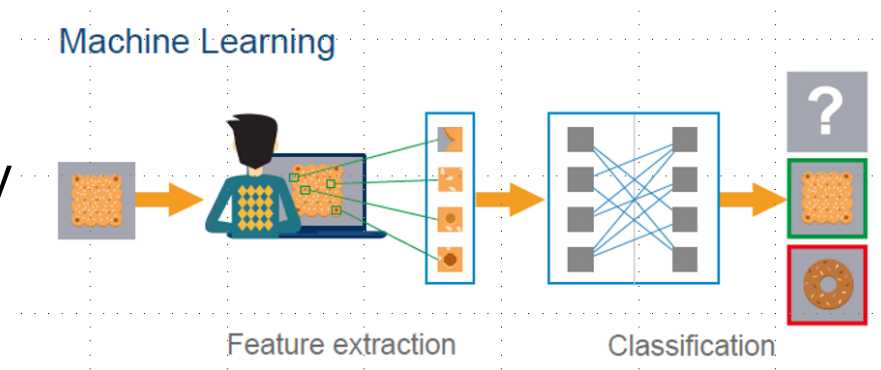


IMAGE ANNOTATION TOOLS

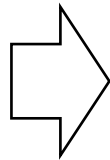
- VGG Image Annotator, LabelImg, OpenLabeler, or ImgLab, or perhaps a commercial tool like
- LabelBox or Supervisely.
- You may also want to consider crowdsourcing platforms such as Amazon Mechanical Turk if you have a very large number of images to annotate.

SIMPLE PREPROCESSING

- Build a script that transforms your images to feature vectors
- Use library functions, e.g. OpenCV



thresholding



feature extraction

Scripting?



PYTHON

- General-purpose language
- Interpreted (no compilation)
- Garbage collected (no memory management)
- Cross-platform: Linux, Mac OS X, Windows
- Weakly typed (duck typing)
- Object-oriented if you want it to
- Python's advantage is its extreme flexibility
- viable end solution for scientific computing, data analysis, plotting
- thanks to lots of efforts from the Open Source community, leading to the availability of mature 3rd-party tools e.g. numpy, scipy, matplotlib, ipython



ZEN OF PYTHON

“Readability counts.”

code is read much more often than it is written

Check out PEP 8 -- Style Guide for Python Code

<https://www.python.org/dev/peps/pep-0008/>

“Explicit is better than implicit.”

“Beautiful is better than ugly.”

“Simple is better than complex.”

...



LEARN PYTHON ONLINE

- <https://www.youtube.com/watch?v=rfscVS0vtbw>
- <https://www.youtube.com/watch?v=sfhhk8m4mcQ>
- <https://docs.python.org/3/tutorial/>
- <https://www.learnpython.org/>
- <https://www.w3schools.com/python/>
- <https://www.tutorialspoint.com/python/index.htm/>
- <https://www.afterhoursprogramming.com/tutorial/python/python-overview/>
- So many tutorials....

We will talk more about this next week, but pls start learning online

WINDOWS INSTALL

- Install the latest version of python3 via <https://www.python.org/downloads/>
- Pip should come with python3, if not install it <https://pip.pypa.io/en/stable/installing/>
- Install the necessary package from the command prompt

`pip install numpy scipy scikit-learn imutils opencv-python`

See e.g. <https://pypi.org/project/opencv-python/>

HEADLESS RASPBERRY PI INSTALL

- Raspberry Pi Imager
<https://www.raspberrypi.org/downloads/>
- To enable SSH, create a file named `ssh` in boot partition
- Via ethernet (e.g. direct) SSH to hostname: `raspberrypi.local`
login: `pi`, password: `raspberry`
- Open `raspi-config`

```
sudo raspi-config
```


and set
 - Interfacing options: enable VNC and camera
 - Advanced options: expand filesystem, memorysplit: 256MB to GPU
- Reboot

```
sudo reboot
```



CONNECT TO “EDUROAM” WI-FI NETWORK

- Open terminal and change /etc/wpa_supplicant/wpa_supplicant.conf

```
network={
    ssid="eduroam"
    scan_ssid=1
    proto=RSN
    key_mgmt=WPA-EAP
    group=CCMP TKIP
    eap=TTLS PEAP
    identity="your_username@han.nl"
    password="your_password"
    phase2="auth=MSCHAPV2"
}
```

- Restart service

```
sudo wpa_supplicant -i wlan0 -c /etc/wpa_supplicant/wpa_supplicant.conf
```

- Check your connection

```
iwconfig
```

<https://www.instructables.com/id/Access-Eduroam-on-a-Raspberry-Pi-in-Cambridge/>



INSTALL OPENCV ON RASPBERRY PI

- <https://www.pyimagesearch.com/2019/09/16/install-opencv-4-on-raspberry-pi-4-and-raspbian-buster/>
- Choose simple pip-install method (take Step#2 and Step #4a)
N.B. pip is the package installer for Python.
You can use pip to install packages from the Python Package Index and other indexes.

- Test and check build information

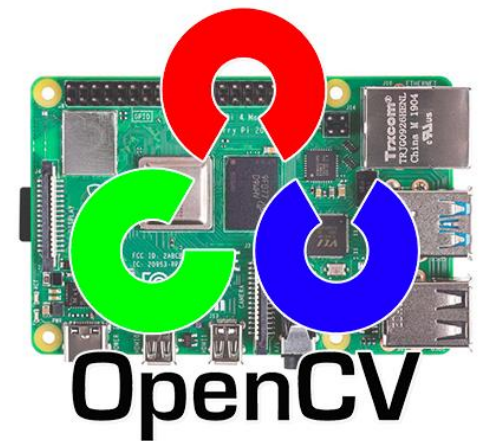
```
python3
>> import cv2
>>> cv2.__version__
>>> print(cv2.getBuildInformation())
```

....

CPU/HW features : VFPV3 NEON

Parallel framework: pthreads

....



ALTERNATIVELY, INSTALL OPENCV 4

- Sept. '21 unfortunately no wheels or pip installer available
- So, you could build from source, see e.g.

<https://qengineering.eu/install-opencv-4.4-on-raspberry-pi-4.html>

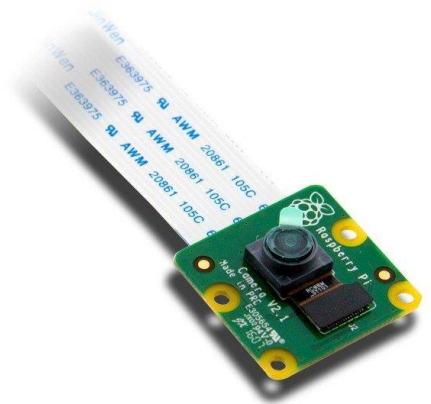
<https://learnopencv.com/build-and-install-opencv-4-for-raspberry-pi/>

RASPBERRY PI CAMERA

- Pure Python interface to the Raspberry Pi camera
<https://picamera.readthedocs.io/en/release-1.13/>

```
sudo pip3 install "picamera[array]"
```

- Newly released: Libcamera
<https://www.raspberrypi.org/documentation/linux/software/libcamera/README.md>



INSTALL MORE PACKAGES

- Mathematics and image processing

```
sudo pip3 install numpy --upgrade
```

```
sudo pip3 install matplotlib
```

```
sudo pip3 install scipy==1.3.3
```

```
sudo pip3 install scikit-image
```

```
sudo pip3 install imutils
```

- Find latest successful builds for Raspbian here:

<https://www.piwheels.org/project/scipy/>

SCIKIT-LEARN

- Finally, the ML package

```
sudo pip3 install scikit-learn
```

- Free software machine learning library for Python
- Features various classification, regression and clustering algorithms
- Designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.



PICK YOUR FAVORITE IDE

- Anaconda
 - Ipython
 - Pycharm
 - Spyder
 - Thonny
 - Atom
 - Visual studio code
 - Jupyter Notebook
 - Colab
-

BUILT-IN DATA TYPES

Text Type:	str
Numeric Types:	int, float, complex
Sequence Types:	list, tuple, range
Mapping Type:	dict
Set Types:	set, frozenset
Boolean Type:	bool
Binary Types:	bytes, bytearray, memoryview

Note that almost *everything* in Python is an object

LISTS

```
my_list = [2, {'dog': ['Rex', 3]}, 'John', 3.14]
```

- Collection which is ordered and changeable (mutable)

Slicing

`x[start:stop:step]`

indices can be negative

- Functions

`len()` gives the length of a list

`sorted()` returns a sorted version of a list

`sum()` does what you might expect

- Methods

`.append()` modifies a list by adding an item to the end

`.pop()` removes and returns the last element of a list

`.index()` searches index of element

FLOW CONTROL

```
fruits = ["apple", "banana", "cherry"]  
for x in fruits:  
    print(x)
```

- Indentation to mark blocks of code
- for iterates on elements from "iterables",
default iterables: arrays, lists, tuples, dictionaries, strings
The for loop specifies: the variable name to use, the set of values to loop over. You use the word "in" to link them together.
- Useful functions
range() returns a sequence of numbers
enumerate() returns item and index
zip() returns combinations
- while loop iterates until some condition is met
- Note: no switch statement
break exits a loop
continue loops around
pass does nothing

FUNCTIONS

```
def my_function():  
    print("Hello from a function")
```

- Functions are defined by def
- Keyword arguments
- Default arguments
- Function recursion is accepted
- Functions Applied to Functions

https://www.w3schools.com/python/python_functions.asp

MODULES

```
from math import log, pi
```

- Collection of variables (a *namespace*, if you like) defined by someone else.
- You can import modules or packages
- E.g. importing math from the standard library
- Other libraries can be easily added
- We can see all the names in math using the built-in function `dir()`.
- Access functions and variables using dot syntax.
- Import module under a shorter alias to save some typing
- `import *` makes all the module's variables directly accessible (without any dotted prefix)
 - but "star imports" can occasionally lead to weird, difficult-to-debug situations.
 - convenient but name collisions possible - avoid if possible
- good compromise is to import only the specific things we'll need from each module

NUMPY

- Fundamental package for computing in Python
- Multidimensional array object
and various derived objects (such as masked arrays and matrices)
- Assortment of routines for fast operations on arrays,
including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete
Fourier transforms, basic linear algebra, basic statistical operations, random
simulation and much more
- Execute operations at near-C speed but with simple code
- Fully supports an object-oriented approach

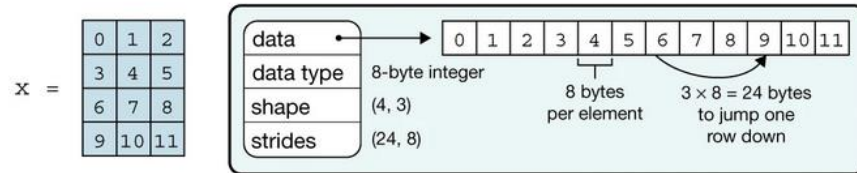
Check out e.g.

- https://www.python-course.eu/numpy_numerical_operations_on_numpy_arrays.php
- <https://towardsdatascience.com/20-numpy-operations-that-every-data-scientist-should-know-fb44bb52bde5>

NDARRAY TYPE

- numpy's ndarray type is specialized for working with multi-dimensional data, e.g.
- Defines its own logic for indexing, allowing us to index by a tuple to specify the index at each dimension, e.g.
- Boolean indexing returns a new array (not a view), e.g.
`y = x[x<2]`
- Overloaded operators, e.g.
`+, -`
- Lots of very useful method, e.g.
`x.shape()`

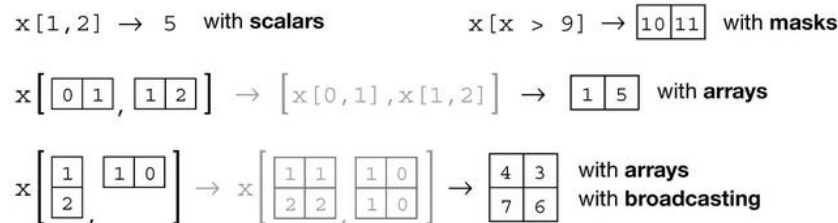
a Data structure



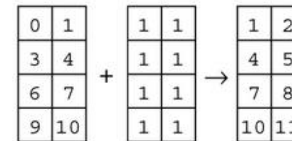
b Indexing (view)



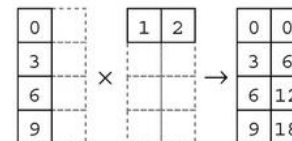
c Indexing (copy)



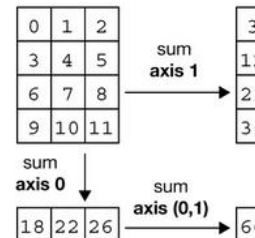
d Vectorization



e Broadcasting



f Reduction



g Example

```
In [1]: import numpy as np
```

```
In [2]: x = np.arange(12)
```

```
In [3]: x = x.reshape(4, 3)
```

```
In [4]: x
```

```
Out[4]:
```

```
array([[ 0,  1,  2],
       [ 3,  4,  5],
       [ 6,  7,  8],
       [ 9, 10, 11]])
```

```
In [5]: np.mean(x, axis=0)
```

```
Out[5]: array([4.5, 5.5, 6.5])
```

```
In [6]: x = x - np.mean(x, axis=0)
```

```
In [7]: x
```

```
Out[7]:
```

```
array([[ -4.5,  -4.5,  -4.5],
       [ -1.5,  -1.5,  -1.5],
       [  1.5,   1.5,   1.5],
       [  4.5,   4.5,   4.5]])
```

From: Array programming with NumPy