Camera based navigation system

For blind and visually impaired people

Team Members

- Islam Mohamed Kamal
- Karam Nasreddin Abulhassan
- Mahmoud Ahmed Ibrahim
- Mohamed Fareed Gad
- Romany Tawfeek Aziz
- Mohamed Saber El Sayed

Supervised by

- Dr. Mostafa Salah
- Dr. Ahmed Farghal

Introduction

Problem Statement

- **According to the 2012 estimates** by the **World Health Organization**, nearly **285 million** people **of all ages** worldwide are visually impaired.
- Visually impaired persons **face difficulties in performing daily activities.**
- They are **exposed to hazards** (indoor/outdoor0) moving in the surrounding, **interacting with the environment.**

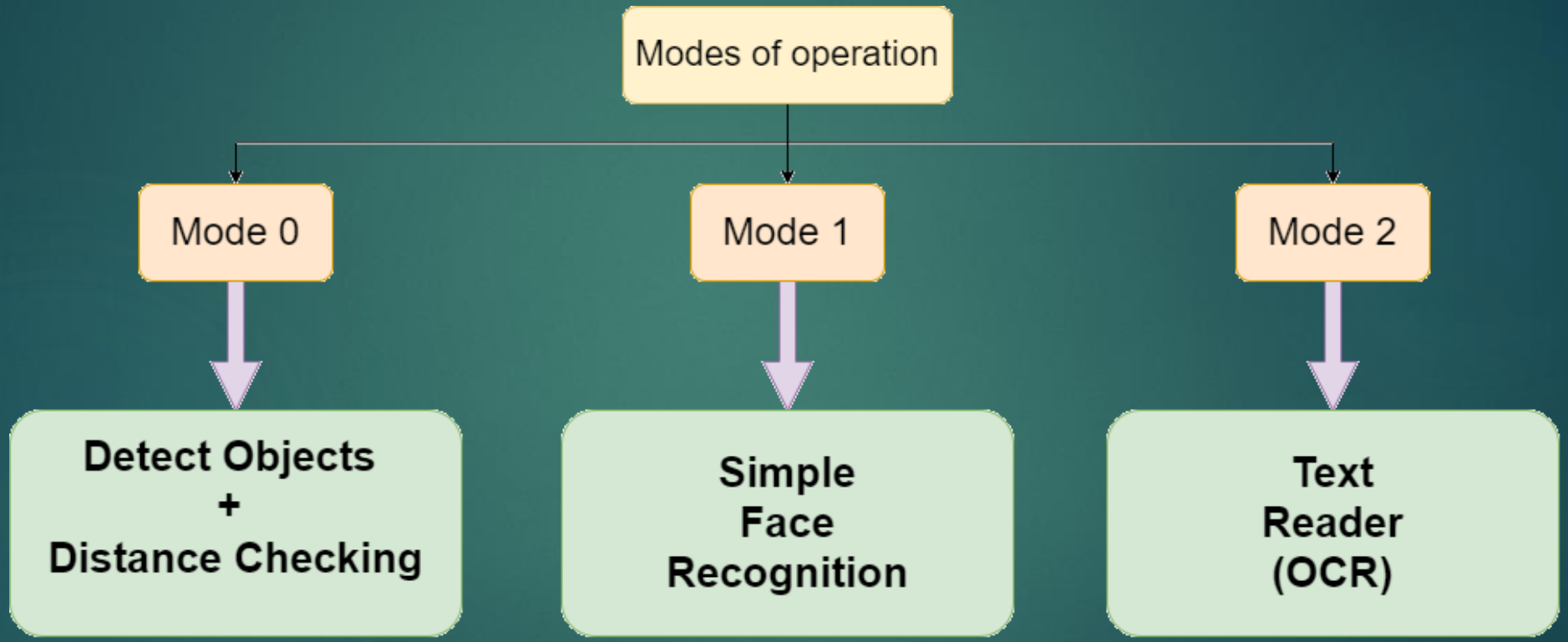
Available HW & SW Solutions

TapTapSee
Camera for the Blind
GET

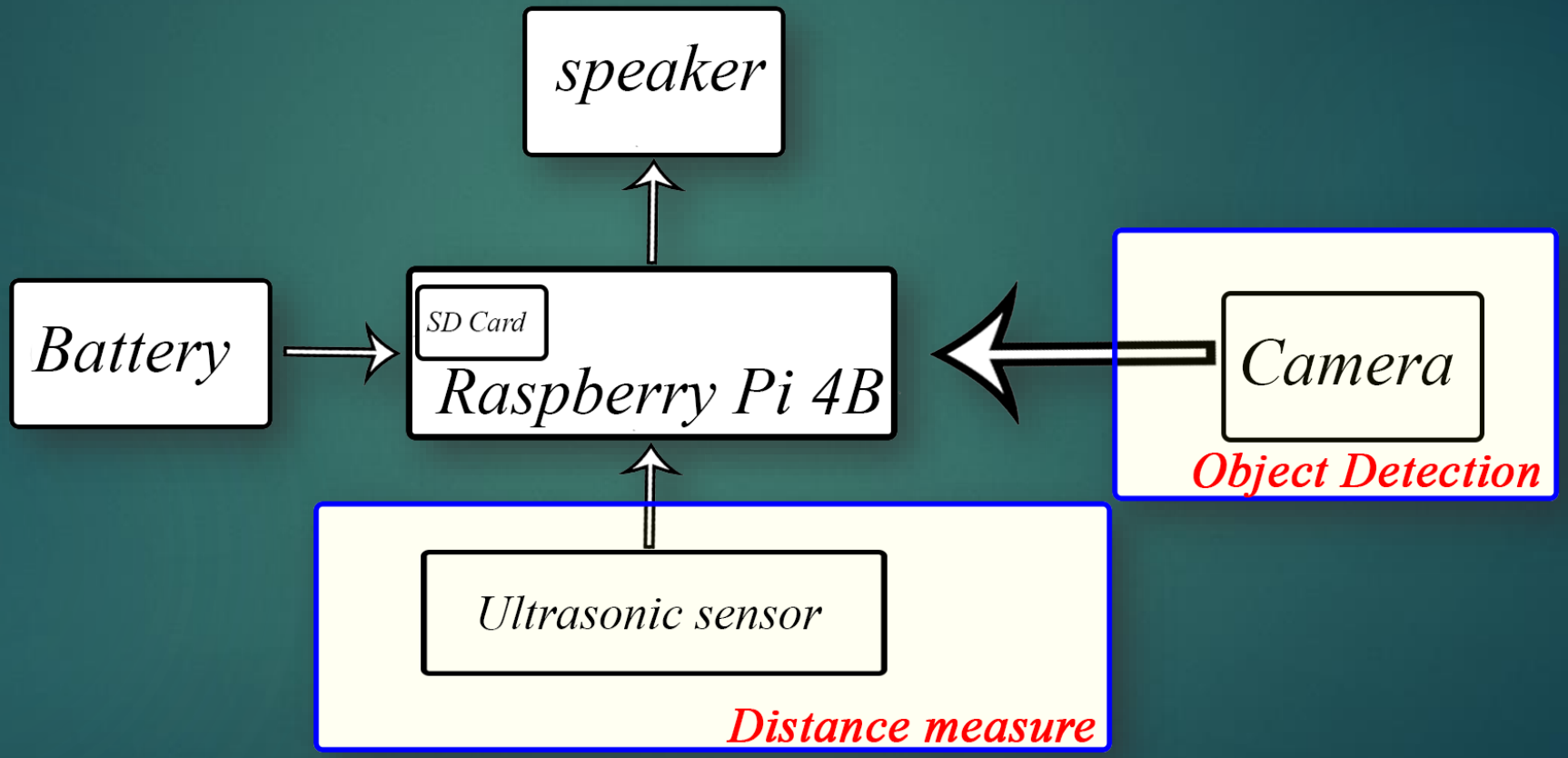
OneStep Reader
Instant access to print
EGP 1,699.99

Seeing AI 4+
Talking Camera for the Blind
Microsoft Corporation
4.4 • 484 Ratings
Free

Proposed System Prototype



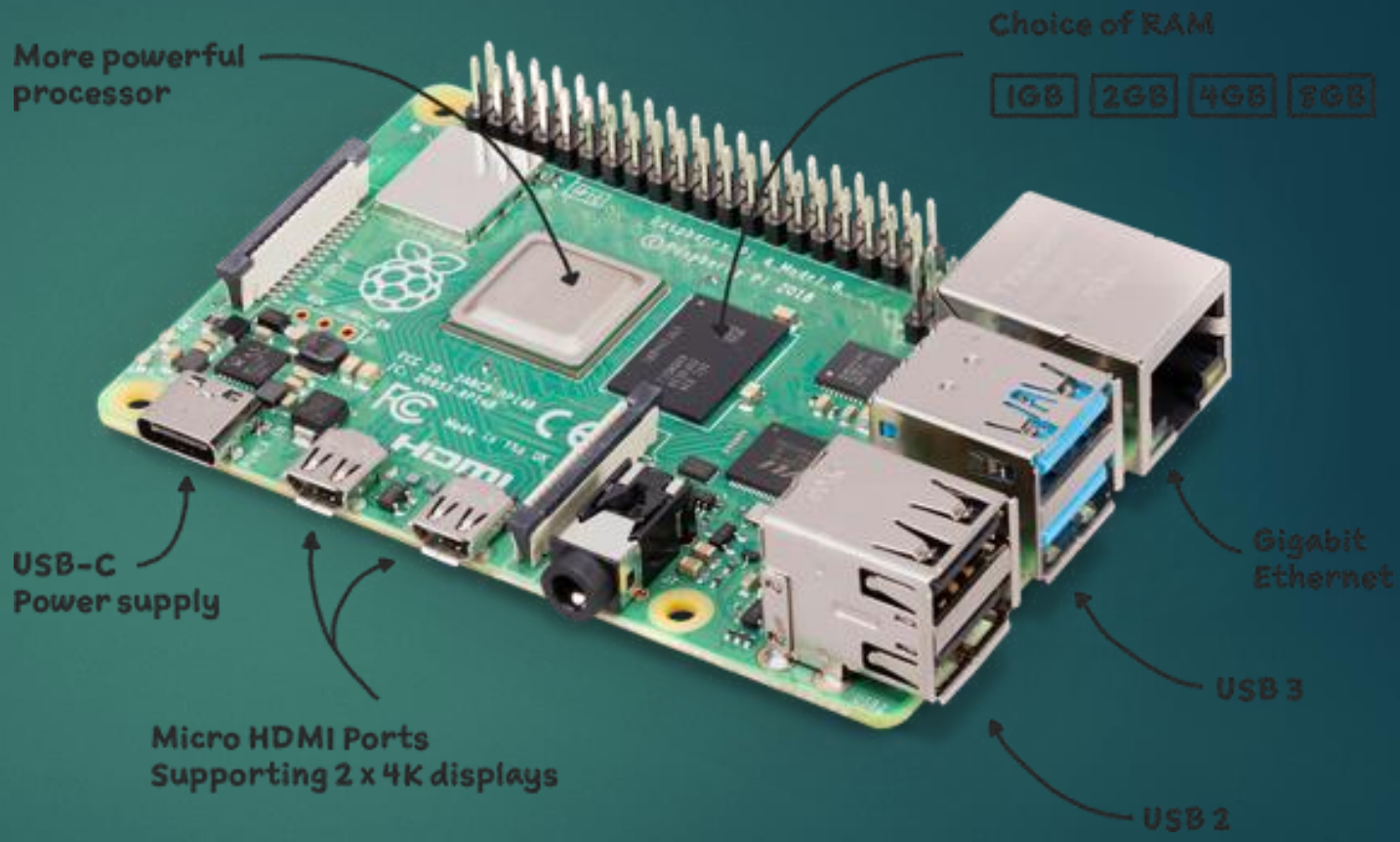
Proposed System Prototype



System Block Diagram

Hardware Development

1. Raspberry Pi 4 Model B+



Raspberry pi model A or B

Raspberry pi Model A

- Single USB port
- 40 Pin GPIO header
- No Ethernet
- 256MB SDRAM
- Smaller form factor

Raspberry pi Model B

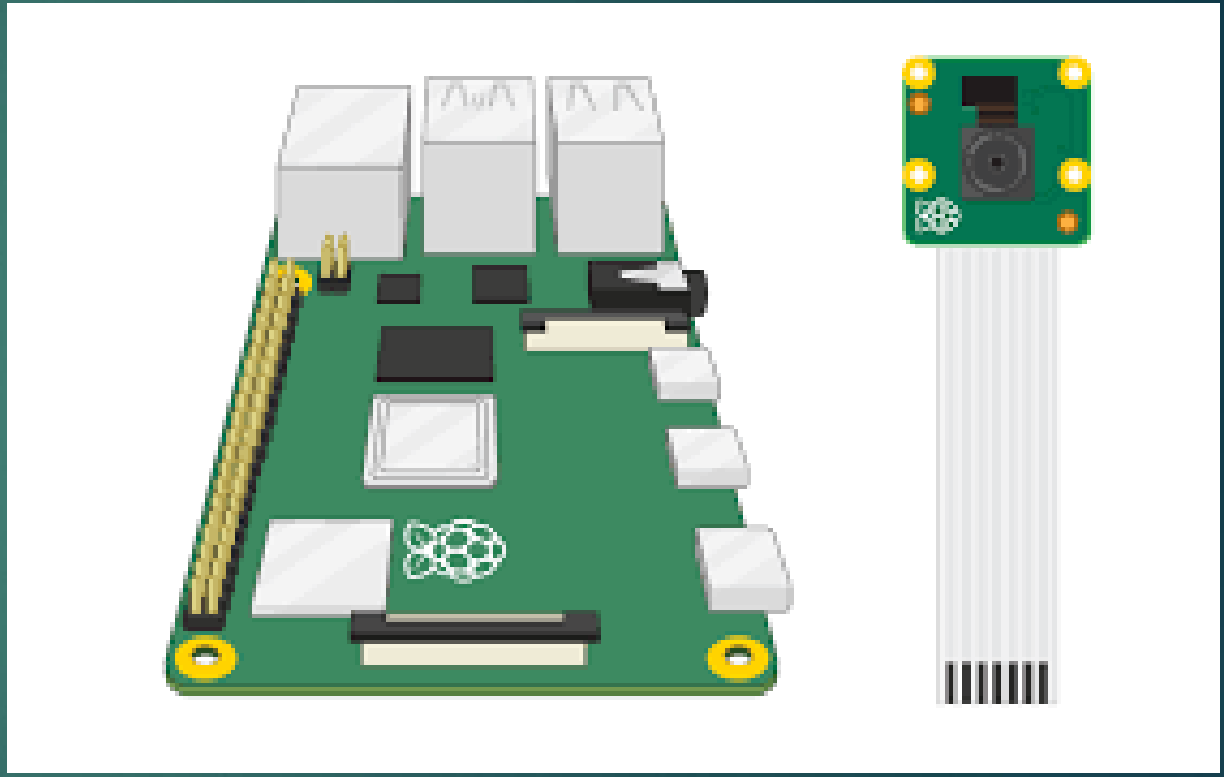
- 4 USB ports
- 40 Pin GPIO header
- 3.5mm audio/video jack
- Ethernet

Raspberry Pi VS Arduino VS Teensy

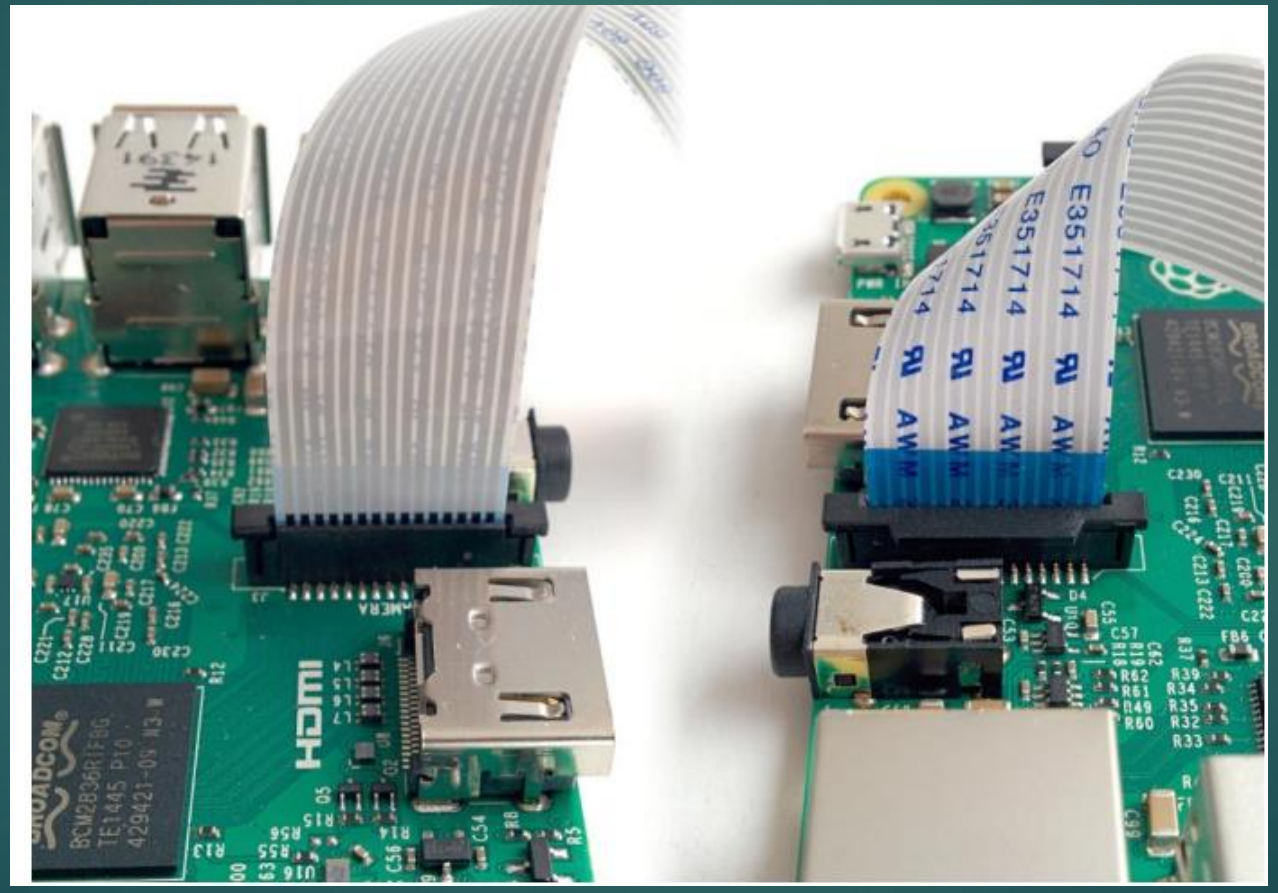
	Raspberry Pi	Teensy	Arduino
CPU Clock	1.5GHz	150MHz	16 MHZ
CPU Cores	4	1	20
CPU Architecture	ARMv8-A, 64-bit	ARMv7E-M 32-bit	ARM Cortex-M3 32-bit
RAM Size	1, 2, or 4GB	256KB	2KB
Storage	MicroSD(any size)	1024KB	32KB
GPU	Broadcom VideoCore VI	N/A	N/A
GPIO Pins	40	58	20
USB Ports	2x USB 2.0, 2x USB 3.0	2x USB 2.0	1x USB 2.0
Ethernet	1Gbps	100Mbps	N/A
WIFI	2.4/5 GHz, 802.11 b/g/n/ac	N/A	N/A
Camera Input	15-pin Serial Interface	N/A	N/A
Bluetooth	5.0 with BLE	N/A	N/A

2. Raspberry Pi Camera

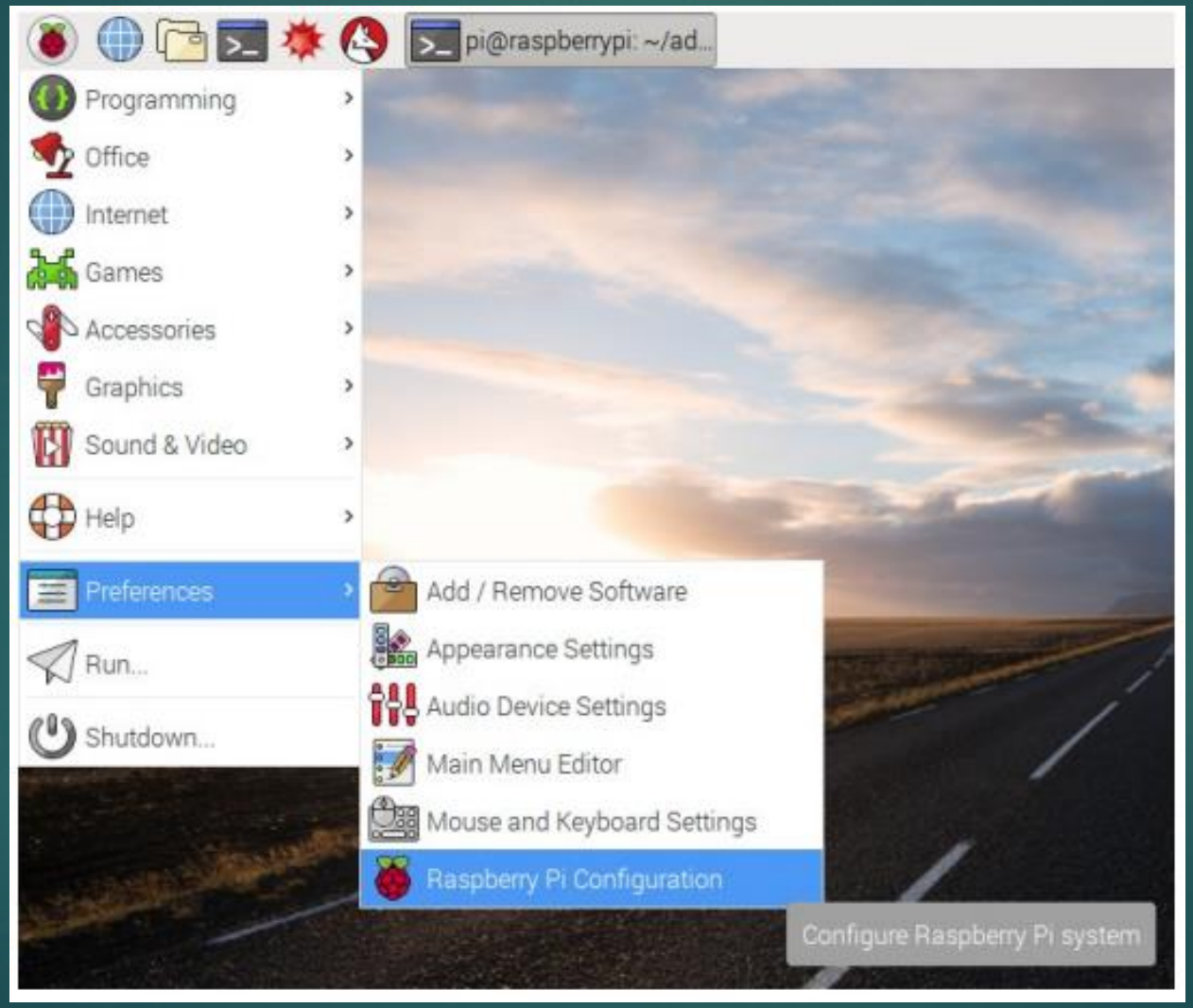
- The Camera Module can be used to take high definition video, as well as stills photographs. It's easy to use for beginners
- The camera module is very popular in home security applications, and in wildlife camera traps.



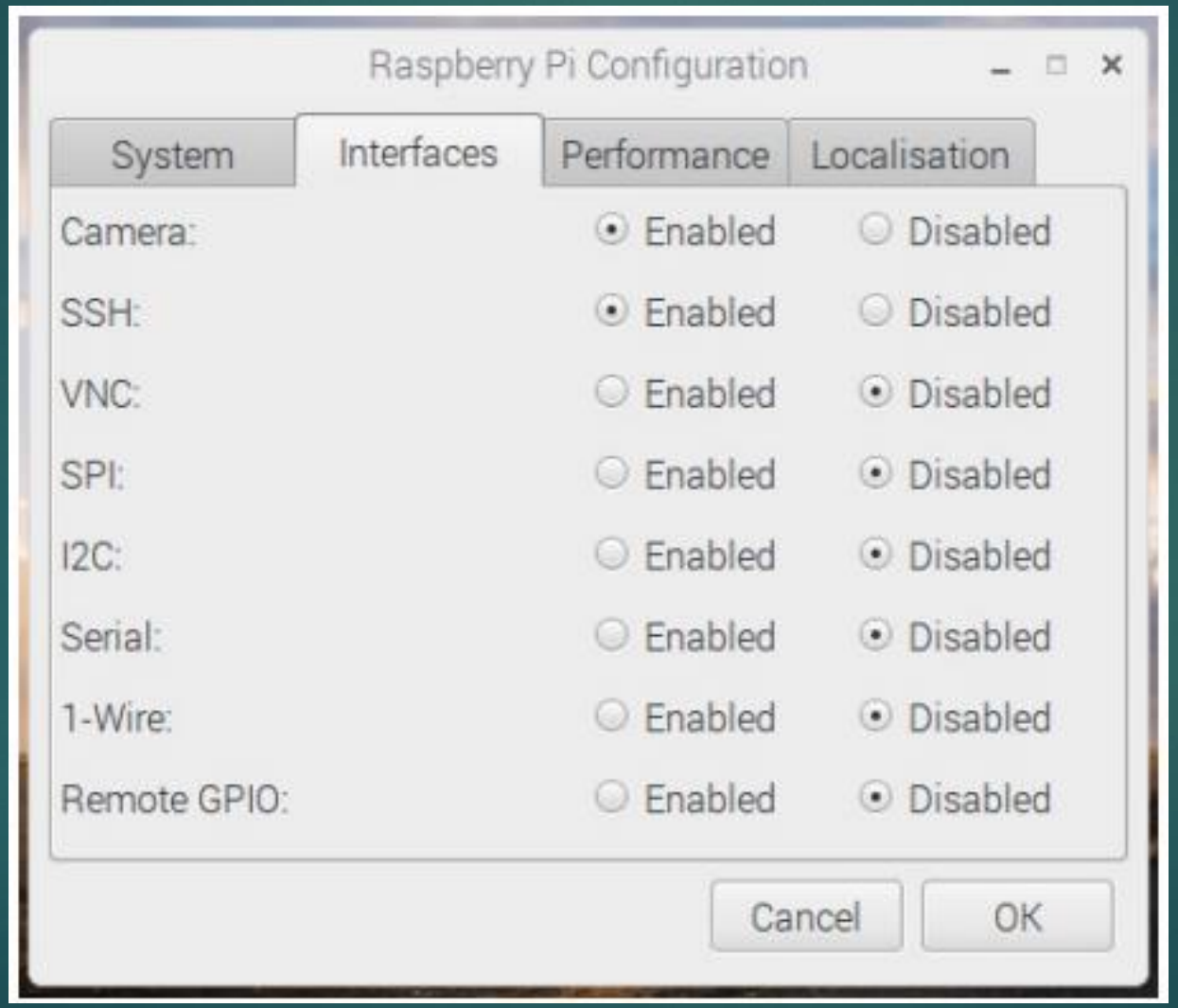
Connecting Raspberry Pi Camera



Connecting Raspberry Pi Camera



Connecting Raspberry Pi Camera



3.Battery

Model	Battery	Hours
Model Zero	1000 mAH	7.4
Model Zero W/WH	1000 mAH	6.2
Model A	1000 mAH	5.3
Model A+	1000 mAH	7.4
Model B	1000 mAH	2.1
Model B+	1000 mAH	3.7
Model 2 B	1000 mAH	3.2
Model 3 B	1000 mAH	3.2
Model 3 A+	1000 mAH	3.1
Model 3 B+	1000 mAH	1.9
Model 4 B	1000 mAH	1.3

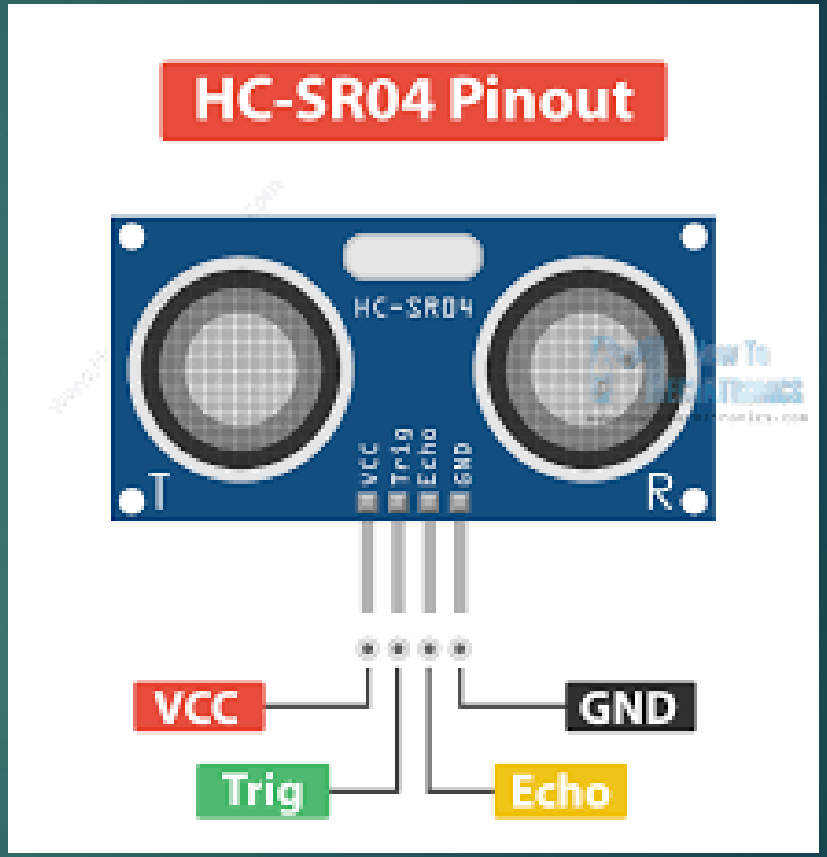
4. SD card

- SD card is the storage of raspberry pi used to keeps the operating system and files.
- The Raspberry Pi Imager Application is used to install Raspberry pi operating system (Raspbian OS) on SD Card .
- The minimum SD card size you should get is 8GB
- There are some SD Cards which are suitable for Raspberry Pi : SanDisk Extreme Pro

4. SD card Test

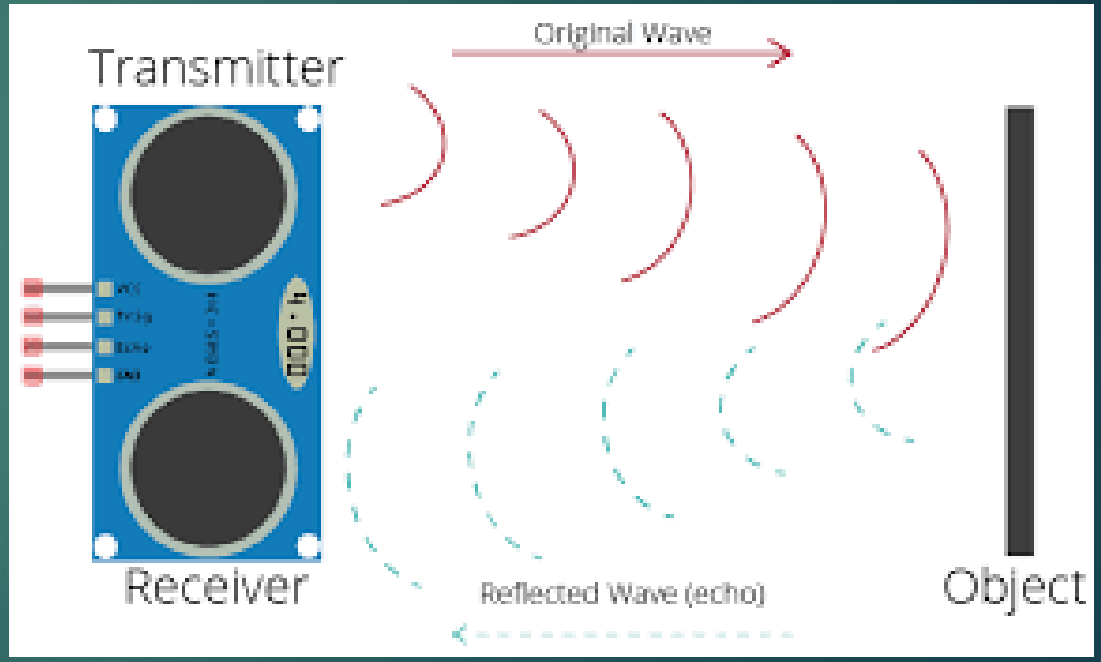
5. Ultrasonic Sensor

Ultrasonic sensors are used primarily as **proximity sensors**. They can be found in automobile self-parking technology and anti-collision safety systems. Ultrasonic sensors are also used in robotic obstacle detection systems, as well as manufacturing technology.



5. Ultrasonic Sensor operation

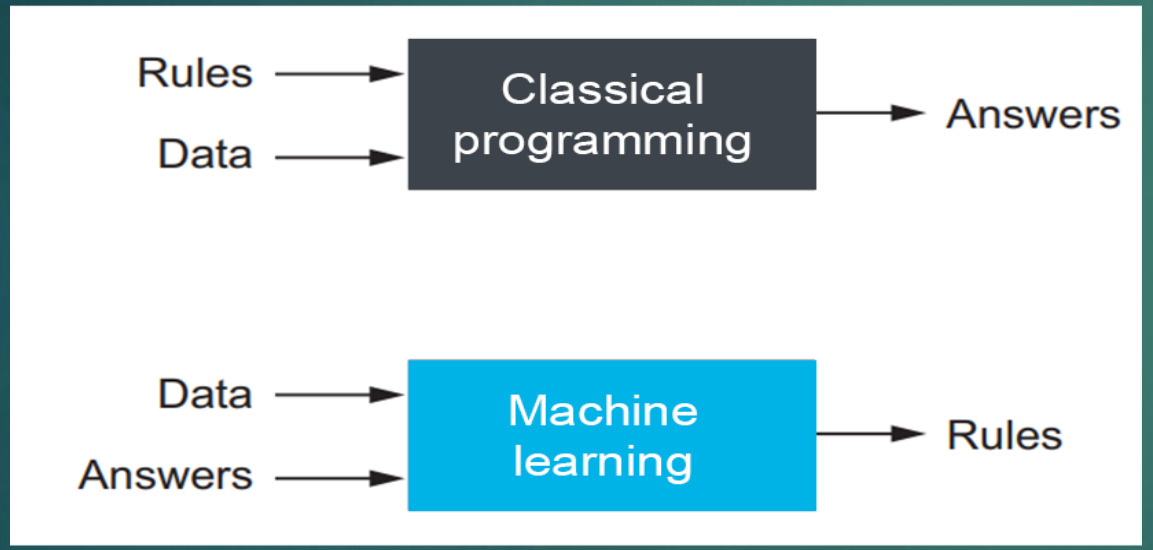
Ultrasonic sensors are used primarily as **proximity sensors**. They can be found in automobile self-parking technology and anti-collision safety systems. Ultrasonic sensors are also used in robotic obstacle detection systems, as well as manufacturing technology.

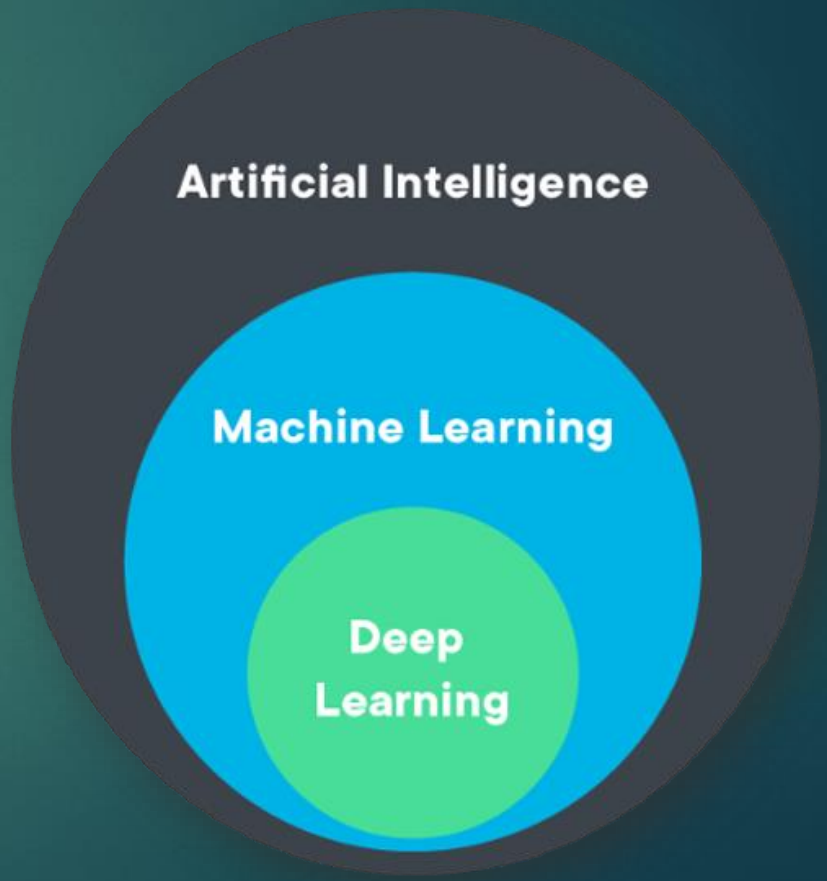


6. Head phone

The headphones will be used to help the user listen to the text that is been converted to audio after it is been captured by the camera or to listen to the translation of the text.

Deep Learning



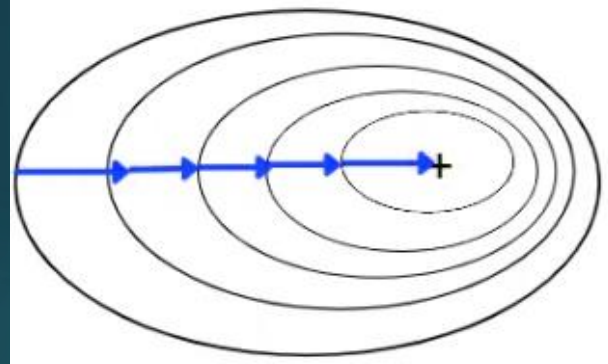


Types of ML

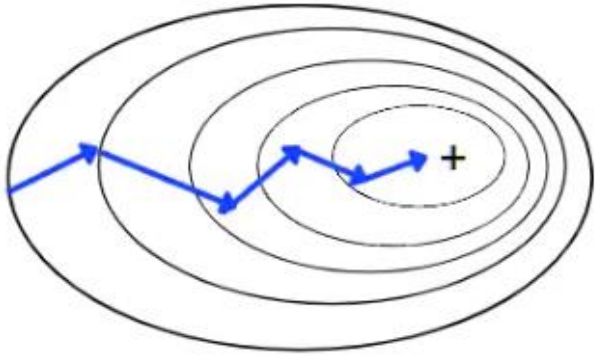

MACHINE LEARNING
UNSUPERVISED LEARNING
SUPERVISED LEARNING
REINFORCEMNET LEARNING
DIMENSIONALLY REDUCTION
CLUSTERING
CLASSIFICATION
REGRESSION
Structure Discovery
Feature Elicitation
Meaningful compression
Big data Visualisation
Image Classification
Customer Retention
Fraud Detection
Diagnostics
Recommended Systems
Targetted Marketing
Customer Segmentation
Forecasting
Predictions
Process Optimization
New Insights
Real-Time Decisions
Robot Navigation
Game AI
Skill Aquisition
Learning Tasks

Optimizers

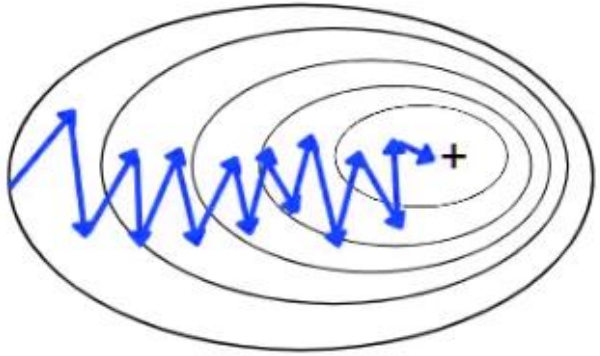
Batch Gradient Descent



Mini-Batch Gradient Descent



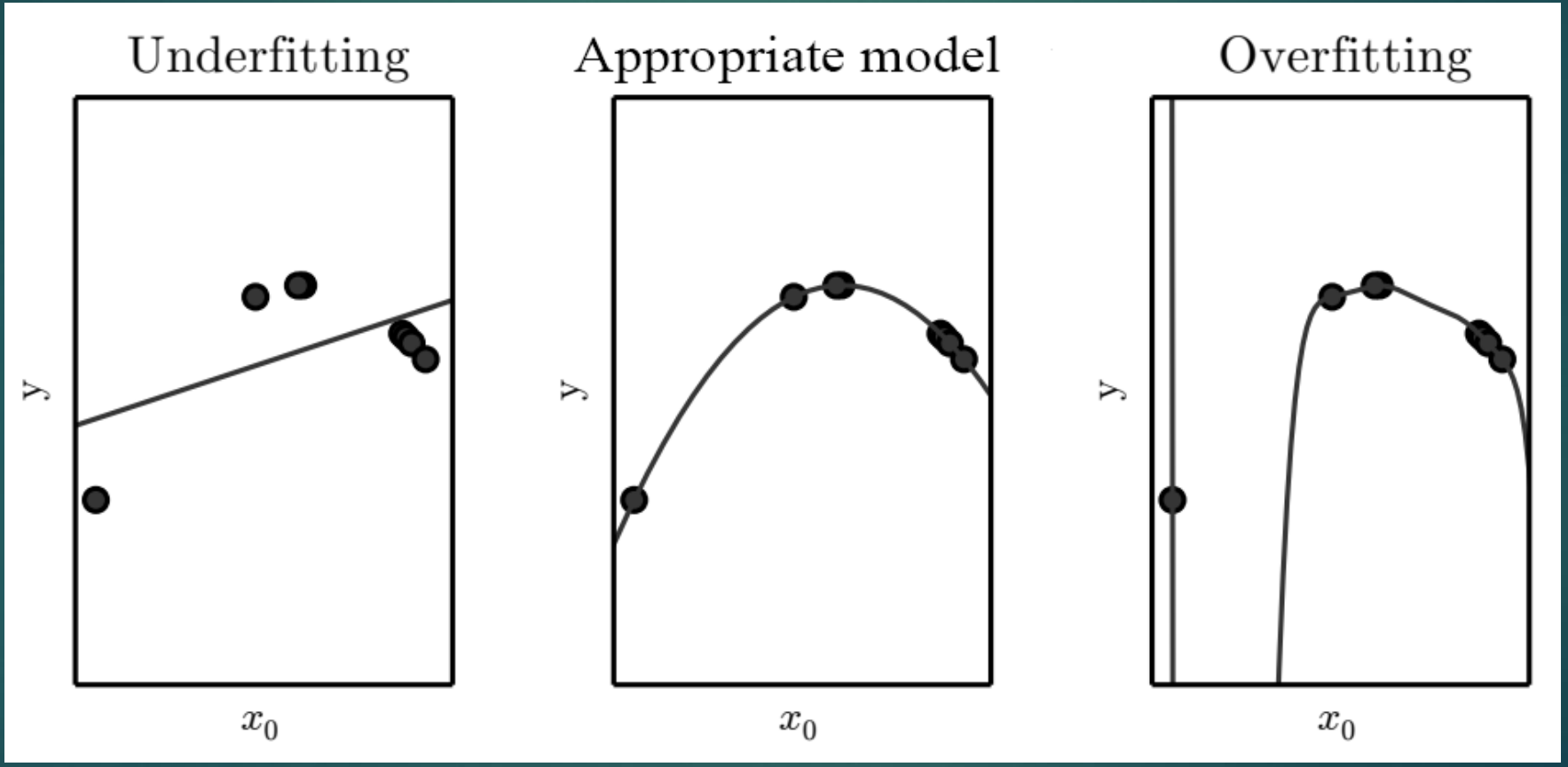
Stochastic Gradient Descent



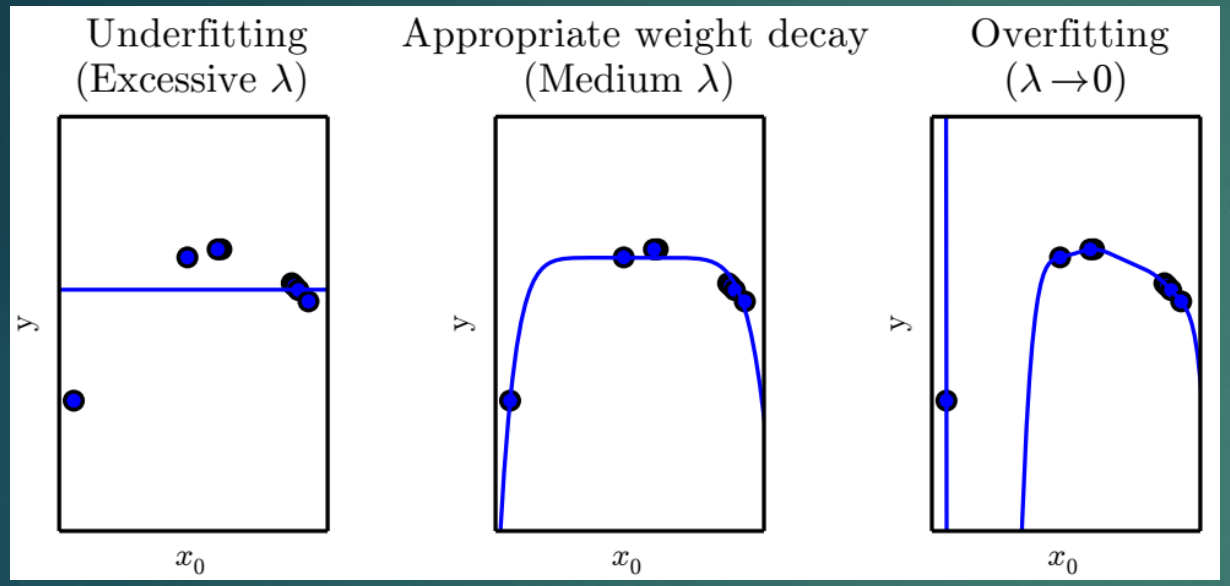
Training examples

Student	X1 Science grade [%]	X2 Chemistry grade [%]	X3 # hours studied [hrs]	Y_{act} Final math grade[%]
1	60	80	5	82
2	70	75	7	94
3	50	55	10	45
4	40	56	7	43

Main ML challenge



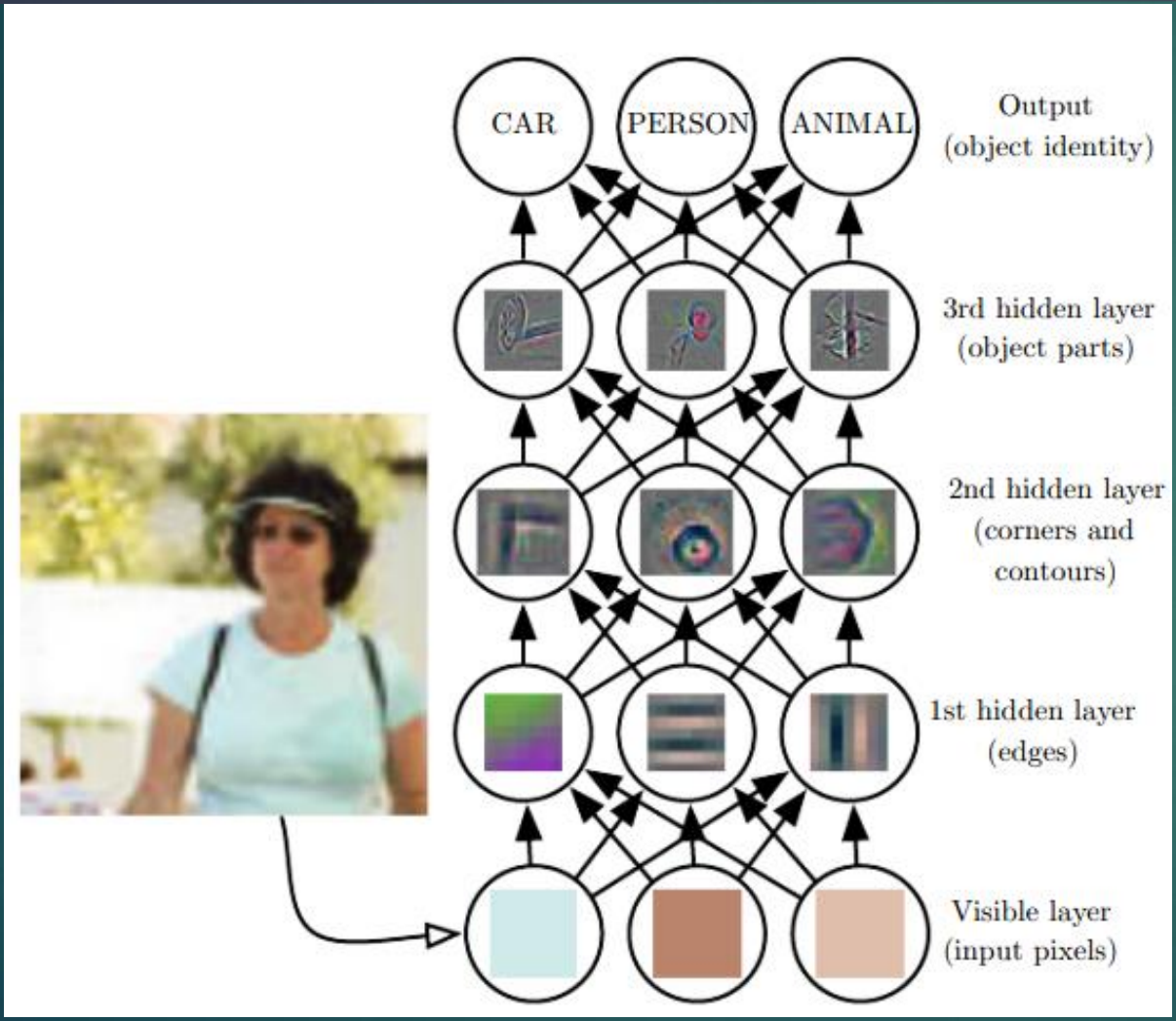
solutions

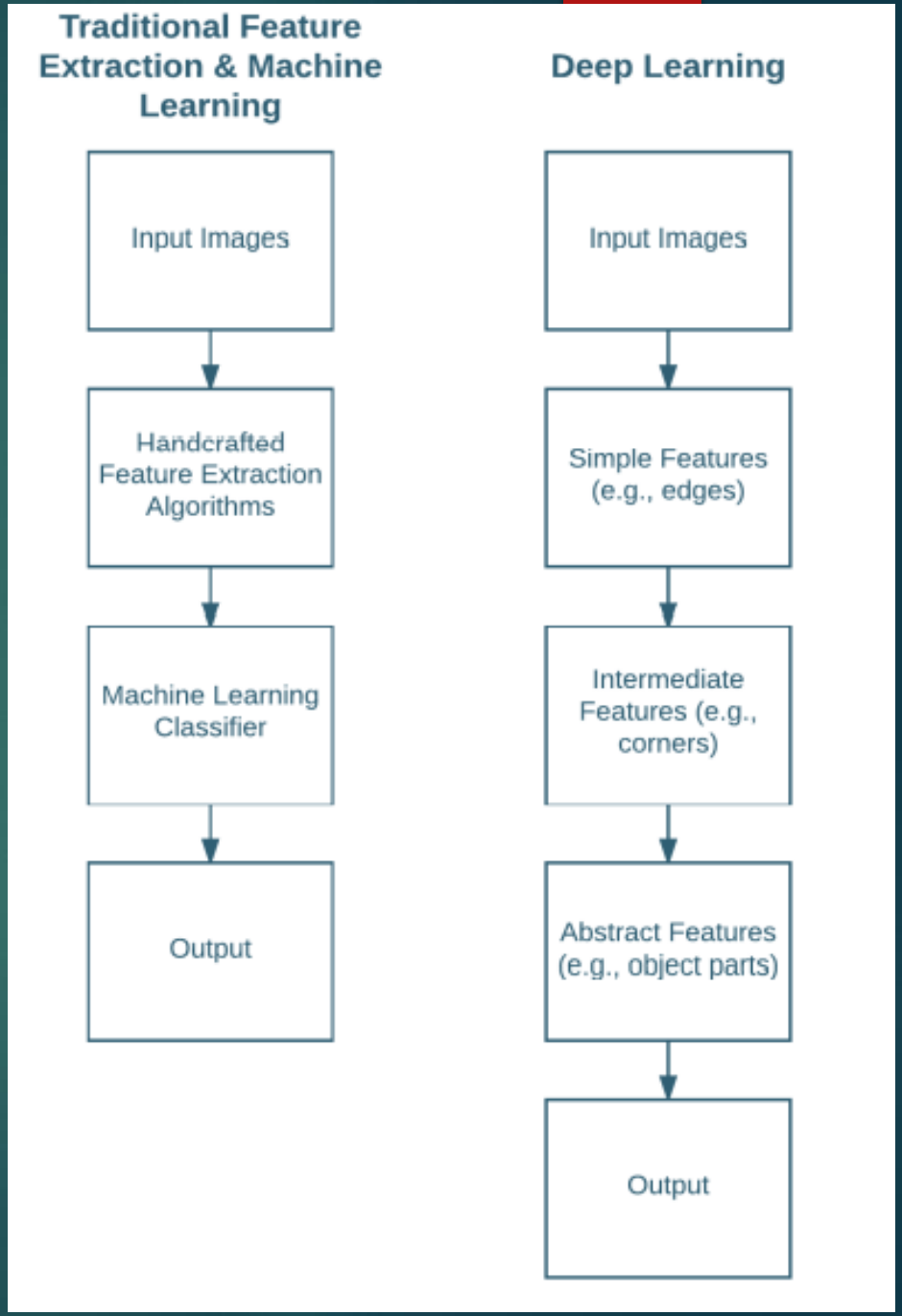


Regularization

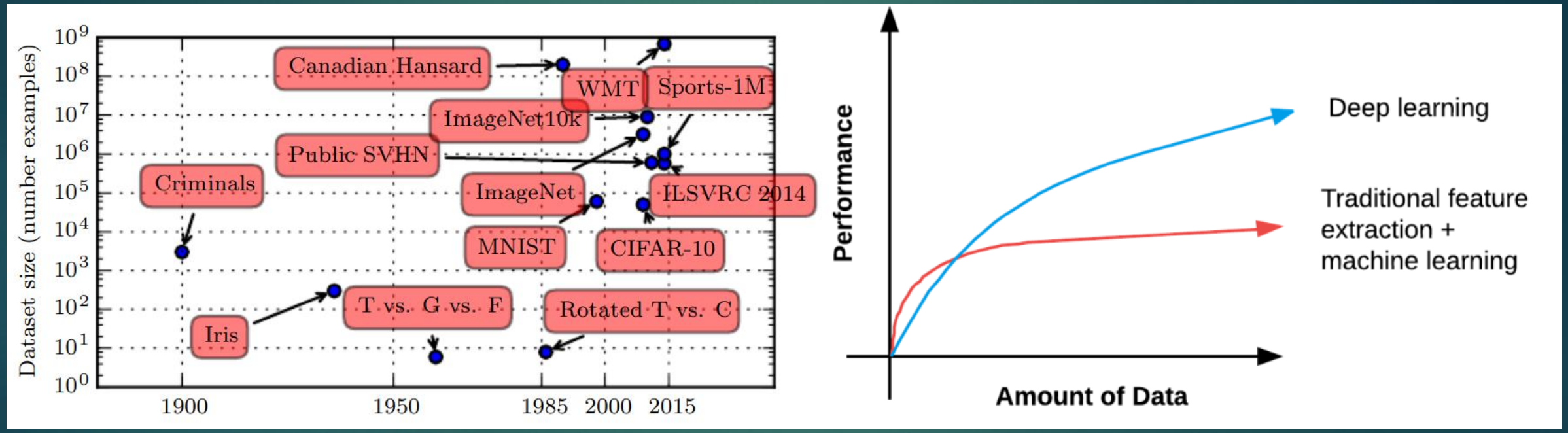
Augmentation

From ML to DL

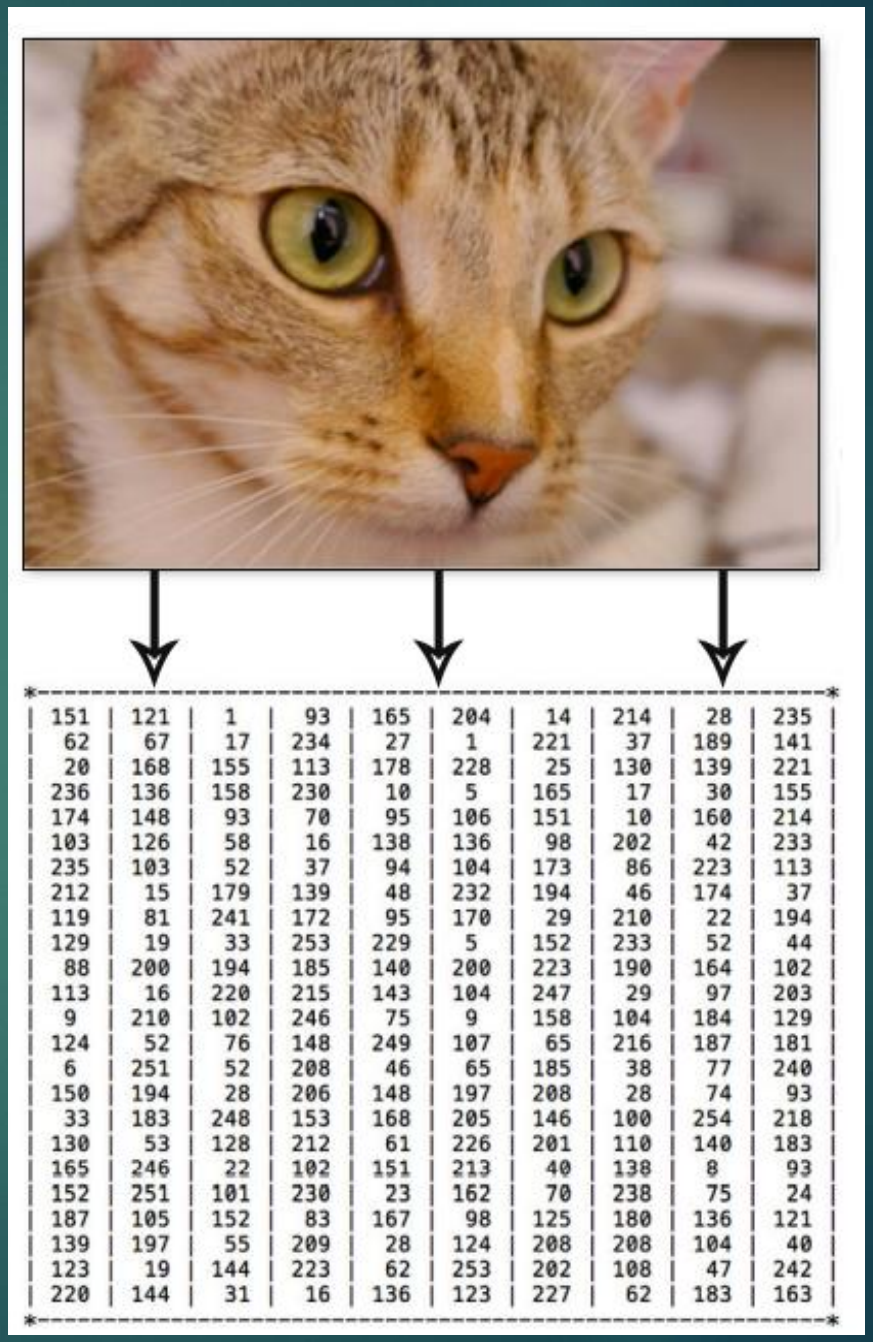

CAR
PERSON
ANIMAL
Output
(object identity)
3rd hidden layer
(object parts)
2nd hidden layer
(corners and
contours)
1st hidden layer
(edges)
Visible layer
(input pixels)


Traditional Feature Extraction & Machine Learning
Input Images
Handcrafted Feature Extraction Algorithms
Machine Learning Classifier
Output
Deep Learning
Input Images
Simple Features (e.g., edges)
Intermediate Features (e.g., corners)
Abstract Features (e.g., object parts)
Output

Why DL ?

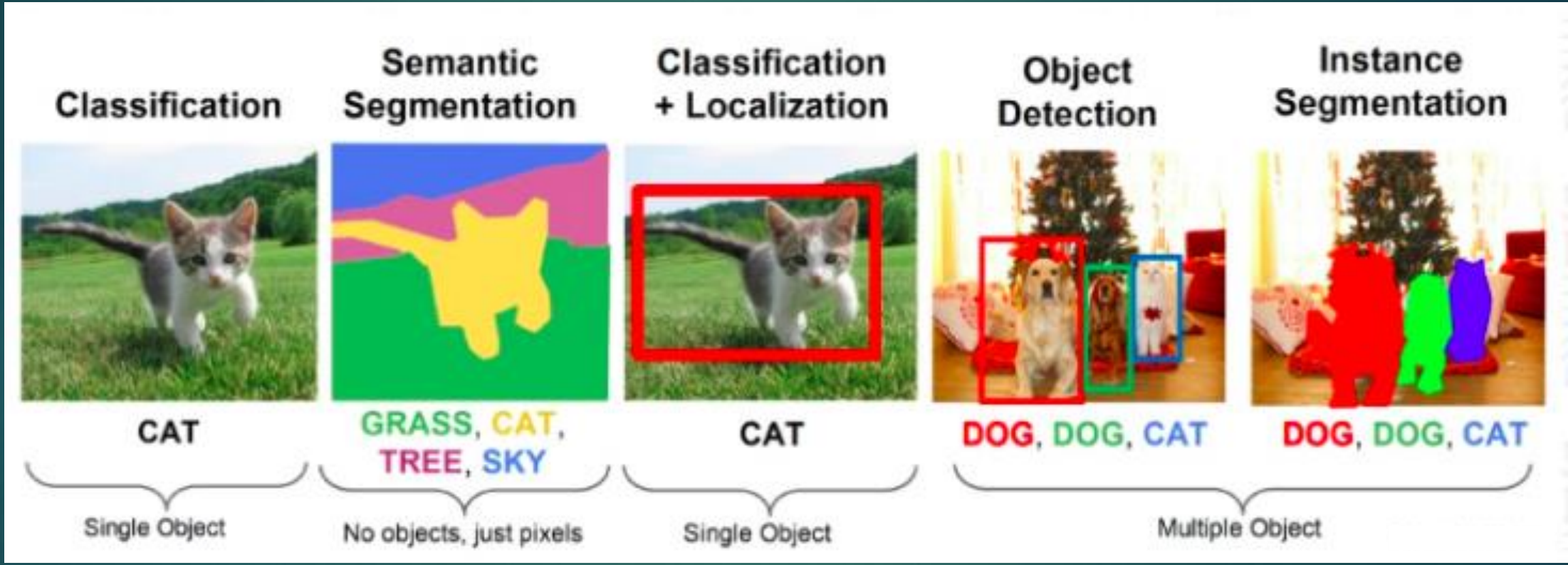

Dataset size (number examples)
10^9
10^8
10^7
10^6
10^5
10^4
10^3
10^2
10^1
10^0
1900
1950
1985
2000
2015
Canadian Hansard
WMT
Sports-1M
ImageNet10k
Public SVHN
Criminals
ImageNet
ILSVRC 2014
MNIST
CIFAR-10
T vs. G vs. F
Rotated T vs. C
Iris
Performance
Amount of Data
Deep learning
Traditional feature extraction + machine learning

Computer Vision

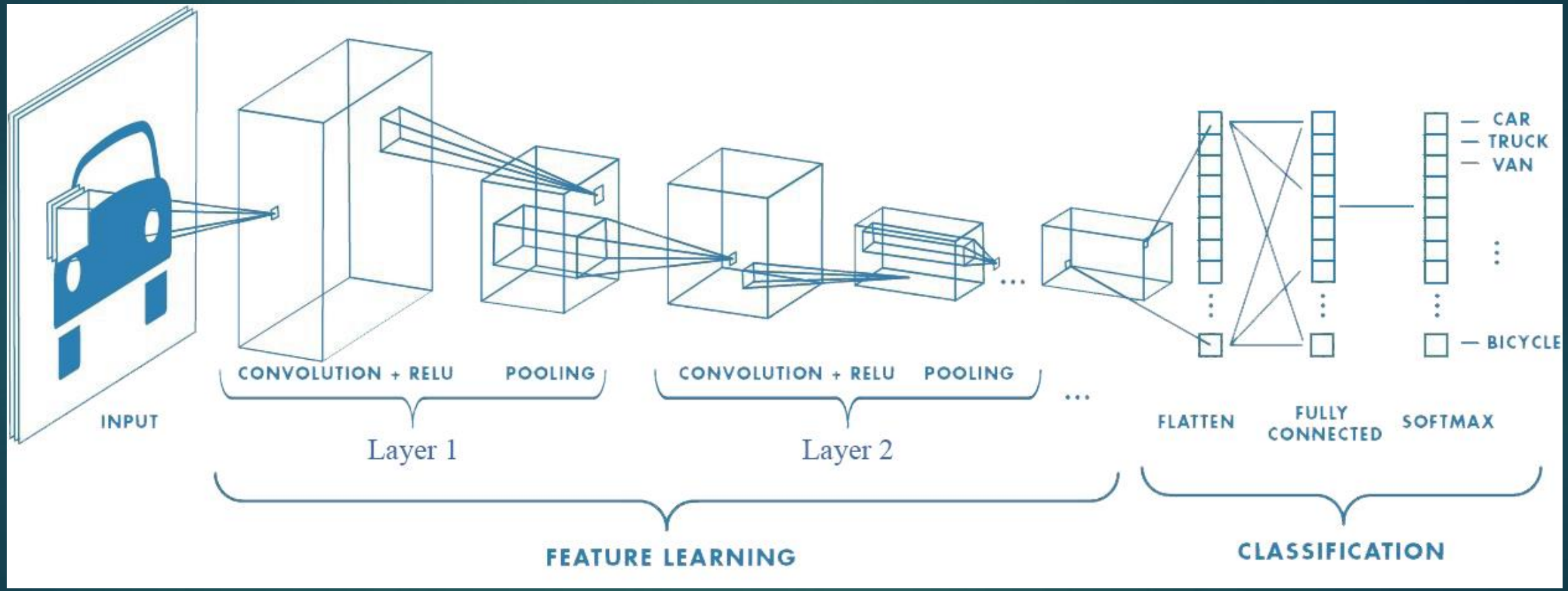


151	121	1	93	165	204	14	214	28	235
62	67	17	234	27	1	221	37	189	141
20	168	155	113	178	228	25	130	139	221
236	136	158	230	10	5	165	17	30	155
174	148	93	70	95	106	151	10	160	214
103	126	58	16	138	136	98	202	42	233
235	103	52	37	94	104	173	86	223	113
212	15	179	139	48	232	194	46	174	37
119	81	241	172	95	170	29	210	22	194
129	19	33	253	229	5	152	233	52	44
88	200	194	185	140	200	223	190	164	102
113	16	220	215	143	104	247	29	97	203
9	210	102	246	75	9	158	104	184	129
124	52	76	148	249	107	65	216	187	181
6	251	52	208	46	65	185	38	77	240
150	194	28	206	148	197	208	28	74	93
33	183	248	153	168	205	146	100	254	218
130	53	128	212	61	226	201	110	140	183
165	246	22	102	151	213	40	138	8	93
152	251	101	230	23	162	70	238	75	24
187	105	152	83	167	98	125	180	136	121
139	197	55	209	28	124	208	208	104	40
123	19	144	223	62	253	202	108	47	242
220	144	31	16	136	123	227	62	183	163

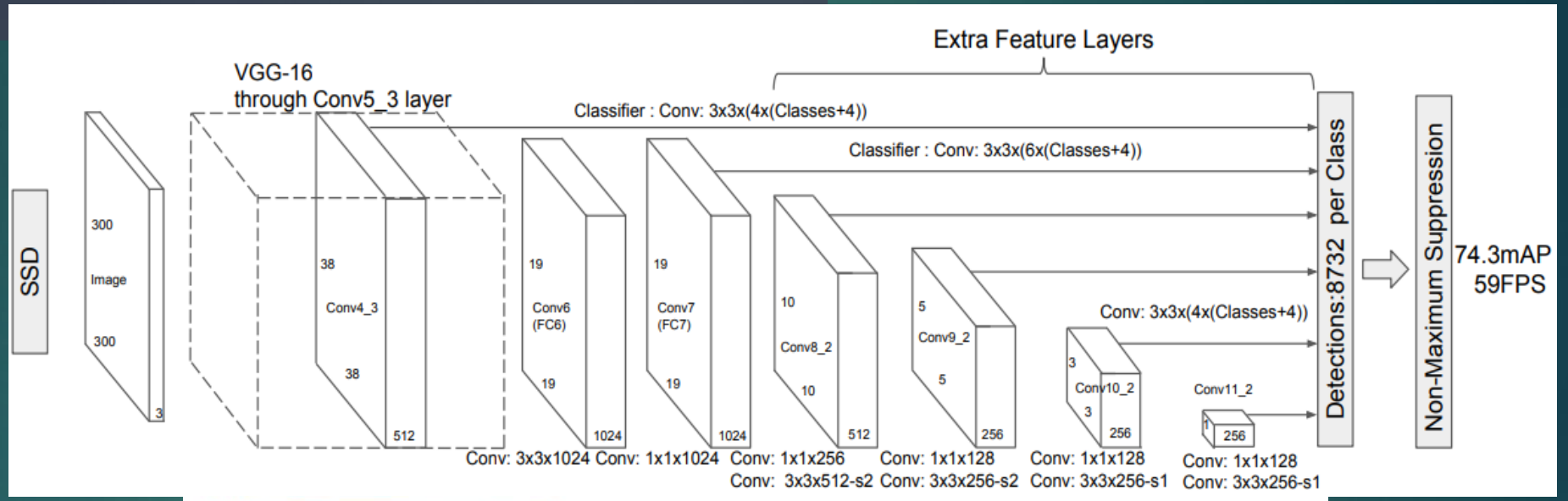
CV tasks

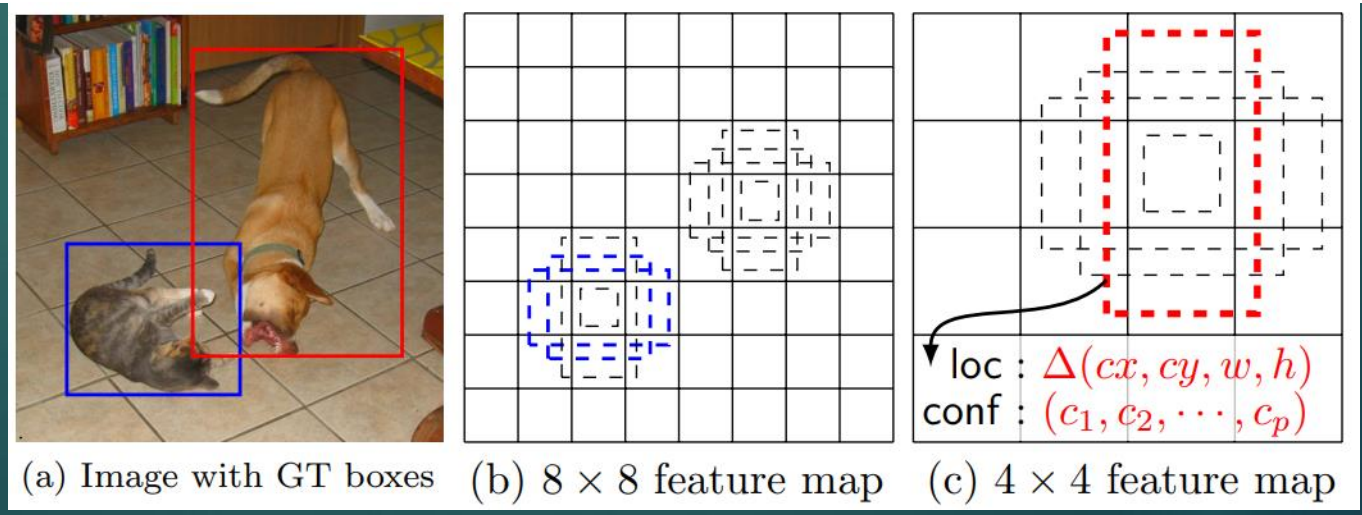

Classification
Semantic Segmentation
Classification + Localization
Object Detection
Instance Segmentation
CAT
GRASS, CAT, TREE, SKY
CAT
DOG, DOG, CAT
DOG, DOG, CAT
Single Object
No objects, just pixels
Single Object
Multiple Object

CNN


INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
Layer 1
Layer 2
FLATTEN
FULLY CONNECTED
SOFTMAX
CAR
TRUCK
VAN
BICYCLE
FEATURE LEARNING
CLASSIFICATION

SSD algorithm



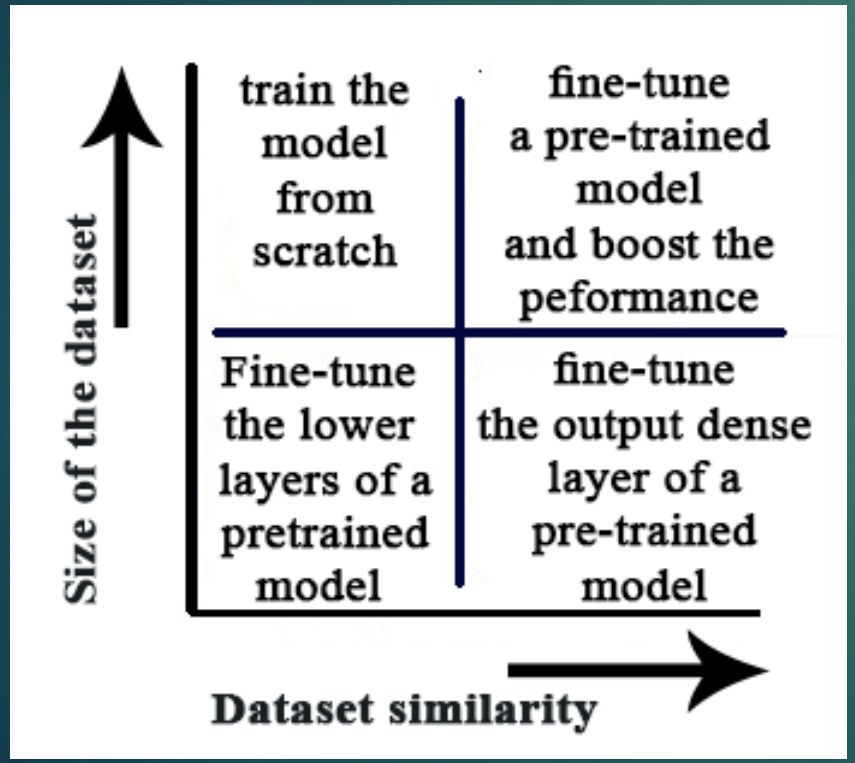


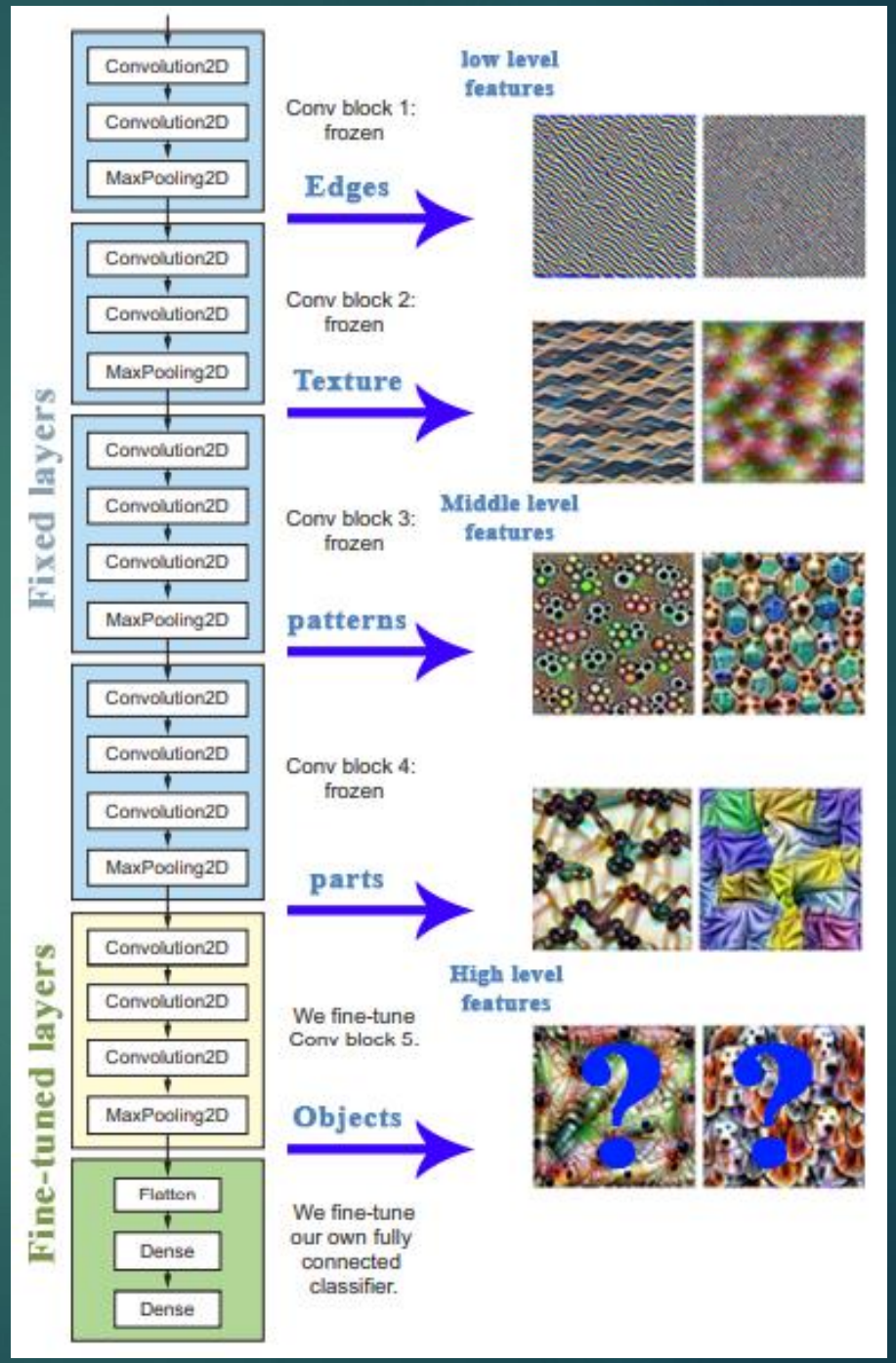
(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

Why SSD?

Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	~ 1000 × 600
Fast YOLO	52.7	155	1	98	448 × 448
YOLO (VGG16)	66.4	21	1	98	448 × 448
SSD300	74.3	46	1	8732	300 × 300
SSD512	76.8	19	1	24564	512 × 512
SSD300	74.3	59	8	8732	300 × 300
SSD512	76.8	22	8	24564	512 × 512

Transfer Learning

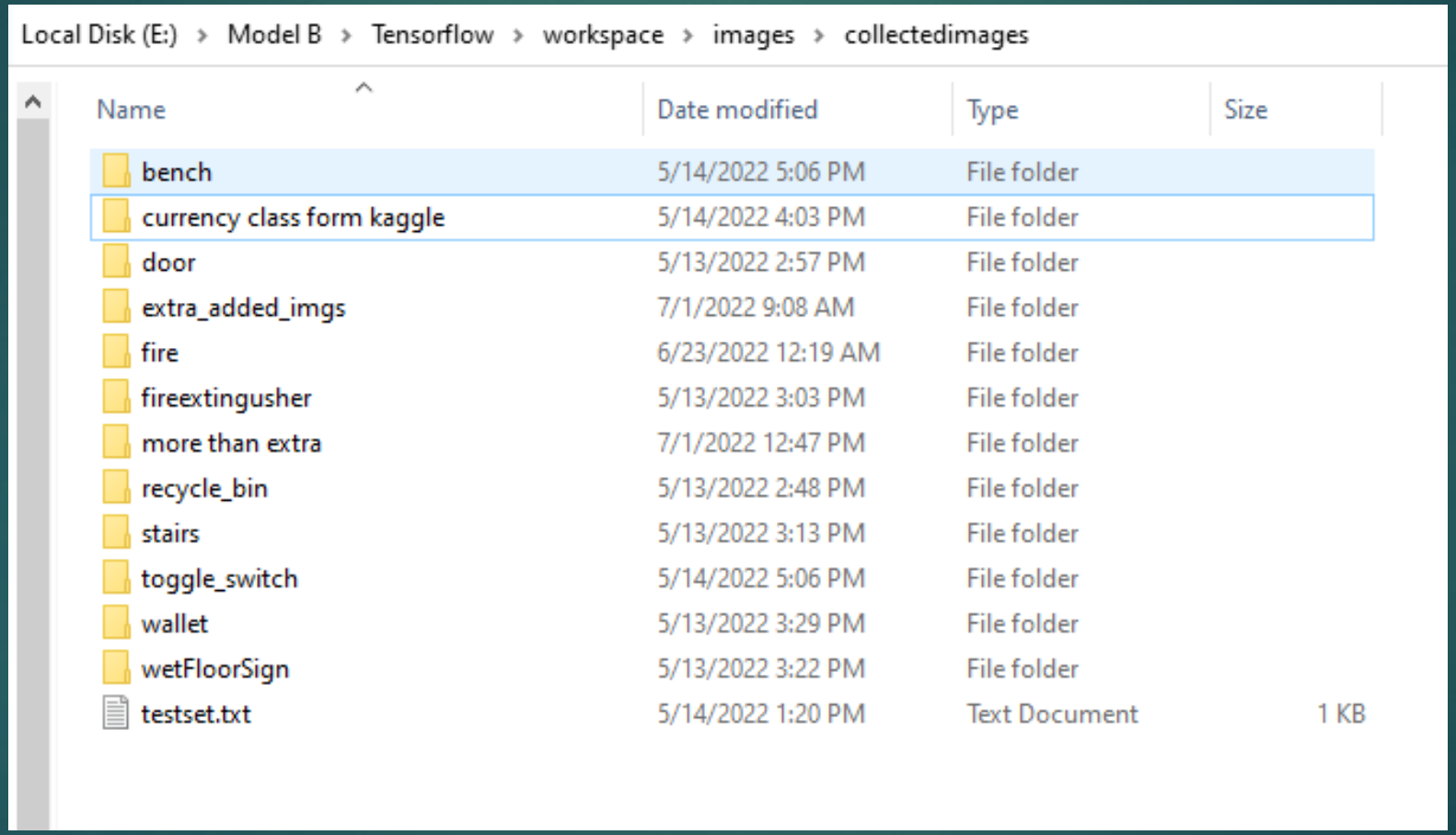




OD Mode (mode 0)

Model A	Model B
Pre-trained model On COCO dataset	Custom OD model On ~ 800 images

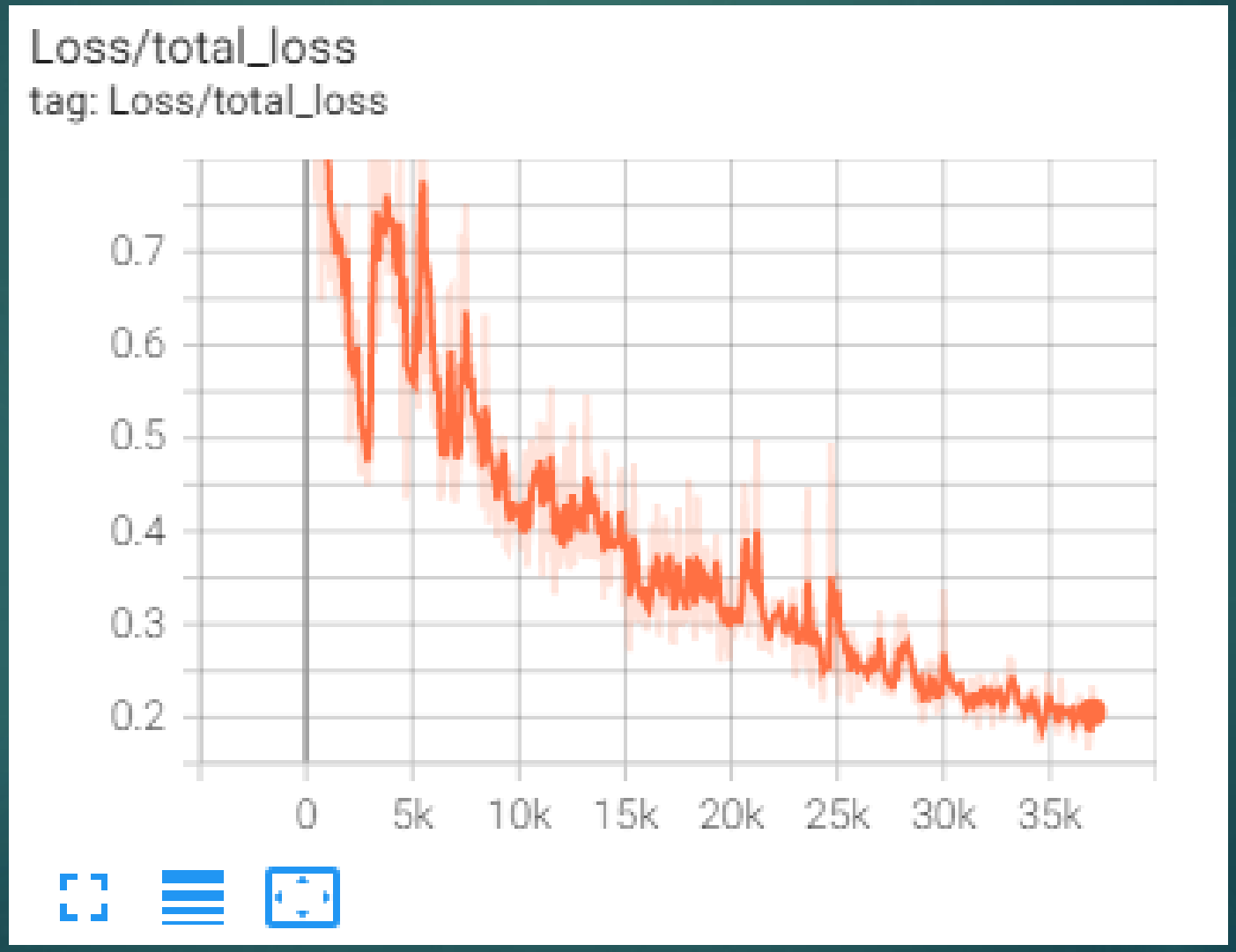
OD Mode (mode 0)



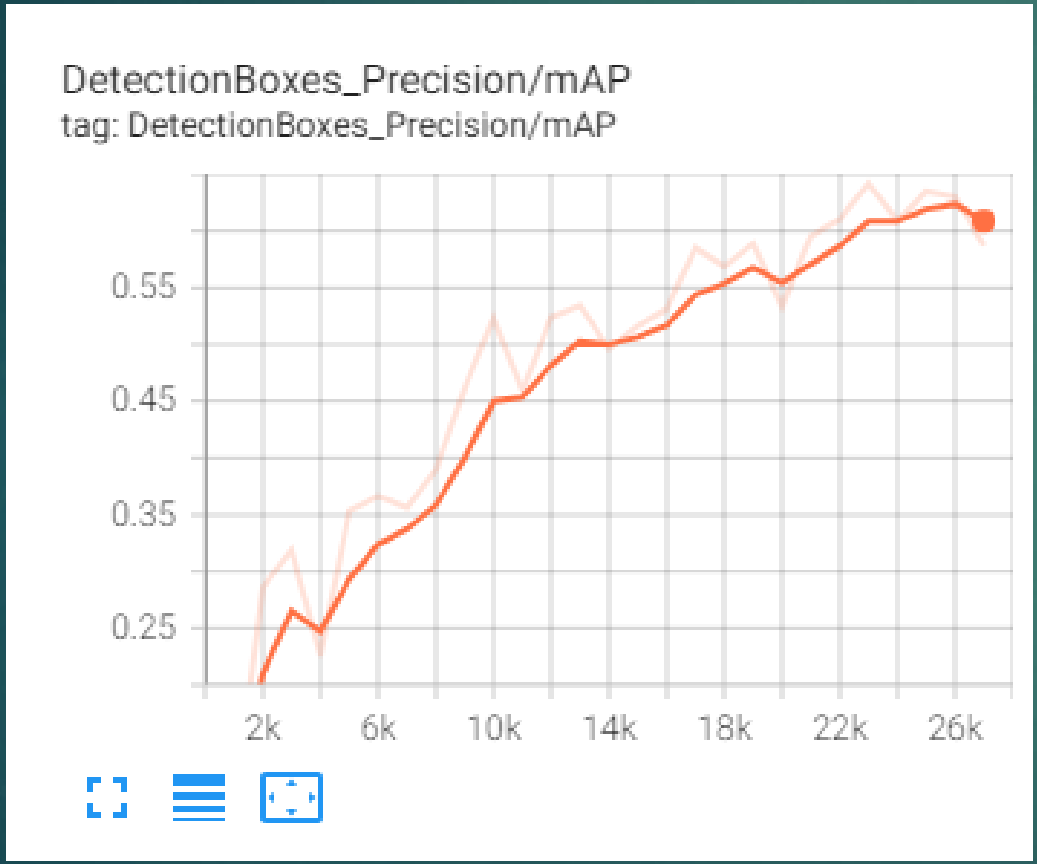
Model B Image Set

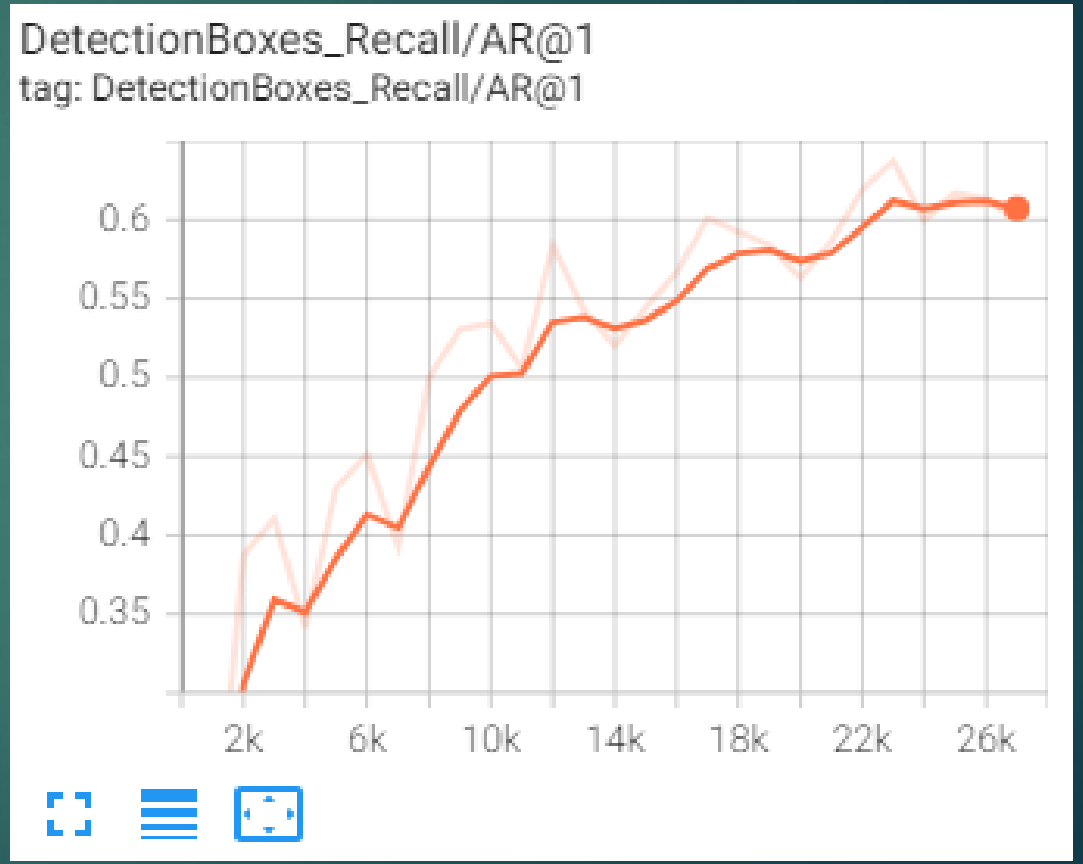
Preparing TFrecord files

Training Metrics



Evaluation Metrics





Model B Results

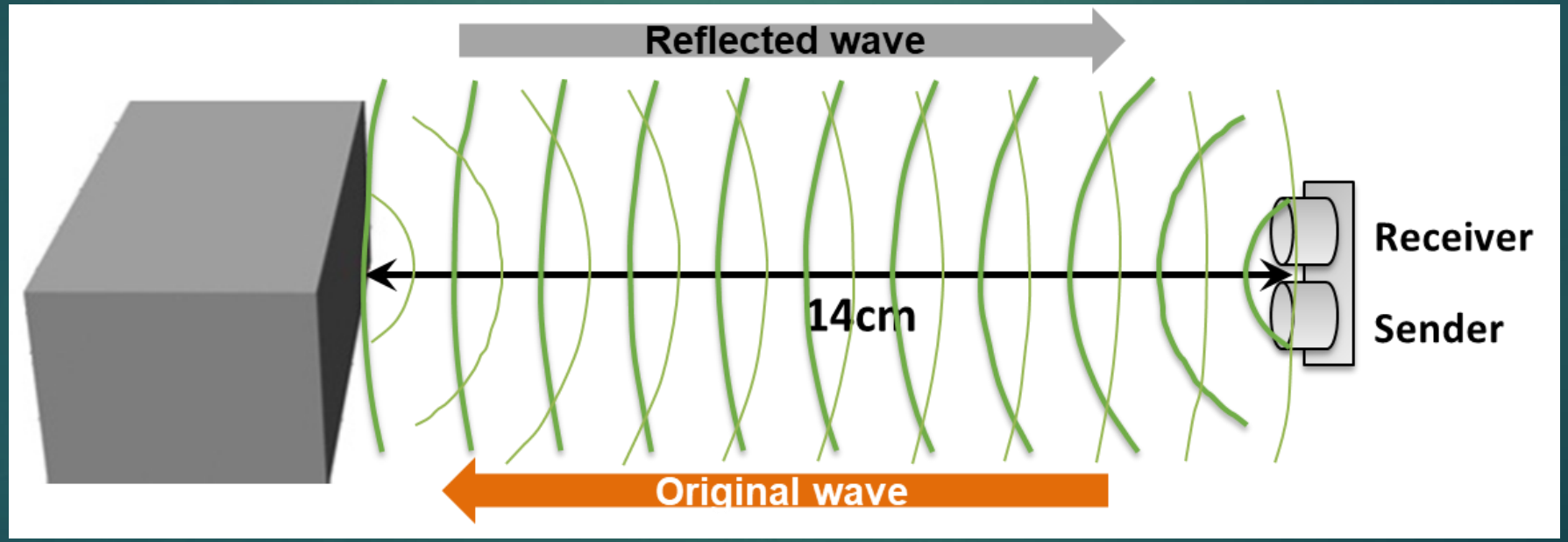
Ultrasonic Sensor

How Ultrasonic Sensors Work

- Ultrasonic sensors use sound wave to determine the distance between the sensor and the closest object in its path

- they operate at a frequency above human hearing

The sensor sends out a sound wave at a specific frequency. It then listens for that specific sound wave to bounce off of an object and come back



HCSR04 Specifications

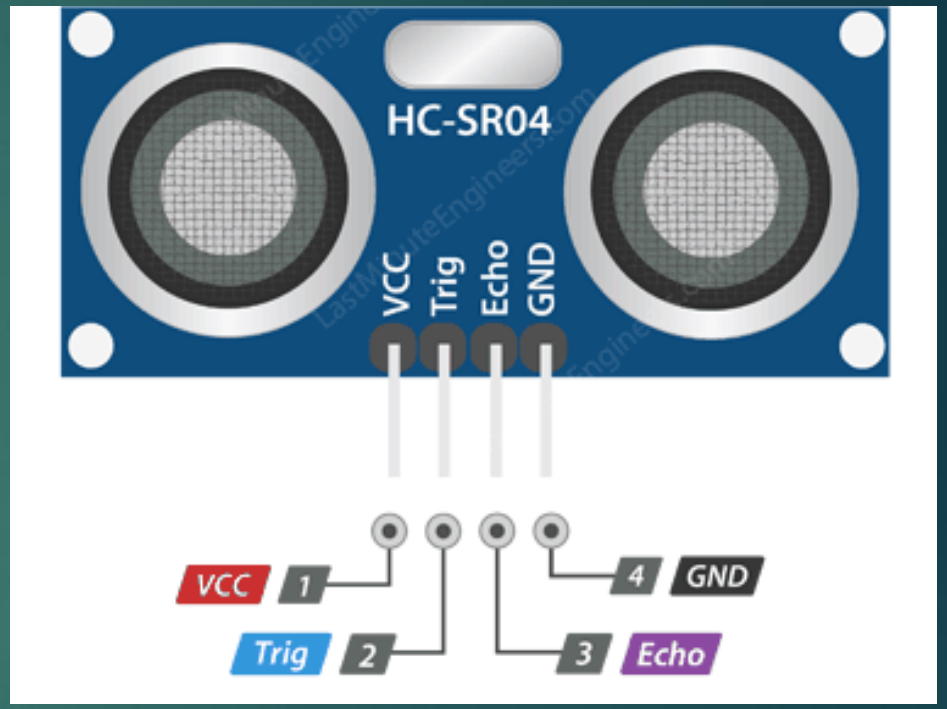
- Power Supply: +5V DC
- Working current: 15mA
- Ranging Distance: 2 - 400 cm
- Resolution: 0.3 cm
- Measuring Angle: 30°
- Trigger Input Pulse width: 10uS
- Dimension: 45mm x 20mm x 15mm
- Weight: approx. 10 g

The HCSR04's best selling point is its price; it can be purchased at around $2 per unit.

HCSR04's pins

The HCSR04 has four pins

1. **VCC** need to be connected to a +5 volt source
2. **TRIG** send the ultrasonic Wave
3. **ECHO** receive the ultrasonic Wave
4. **GND** need to be connected to ground



Wiring HCSR04 with Raspberry pi

Hardware

- HC-SR04 Module
- Resistors:R1= 330Ω and R2 = 470Ω
- Jumper wire

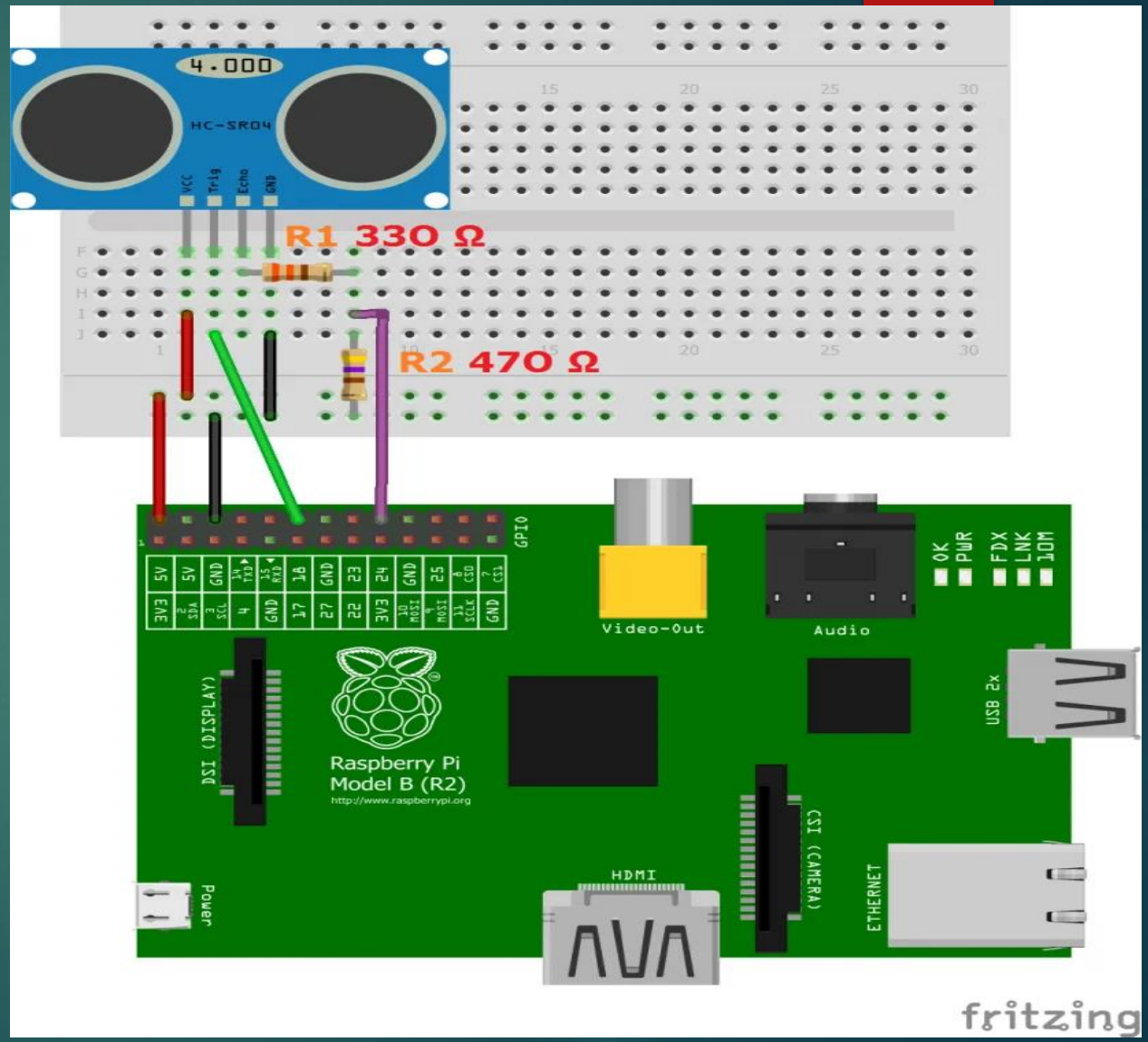
Wiring

There are four pins on the HCSR04 that are connected to the Raspberry:

- VCC to Pin 2 (VCC) +5 volt
- GND to Pin 6 (GND)
- TRIG to Pin 12 (GPIO18)

connect the 330Ω resistor to ECHO. On its end you connect it to Pin 18 (GPIO24) and through a 470Ω resistor you connect it also to Pin6 (GND).

We do this because the GPIO pins only tolerate maximal 3.3V.



Script for controlling

```
import RPi.GPIO as GPIO # Python module to control the GPIO on a Raspberry Pi
import time   #to use delay
import os


GPIO.setmode(GPIO.BOARD)


TRIG = 16    #trigger pin
ECHO = 18    #echo pin


GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)
```

```
while (True):
        GPIO.output(TRIG, True)   #to make trigger transmit power
        time.sleep(0.00001)       #for this time
        GPIO.output(TRIG, False)  # off trigger

        while GPIO.input(ECHO)==0: #when echo don't receive take time
            startTime = time.time()

        while GPIO.input(ECHO)==1: #when echo  receive take time
            endTime = time.time()
        timeElapsed= endTime - startTime

          # d=(s*t/2)
          # speed=s=343 m/s=34300 cm/s
        distance = (timeElapsed * 34300) / 2
```

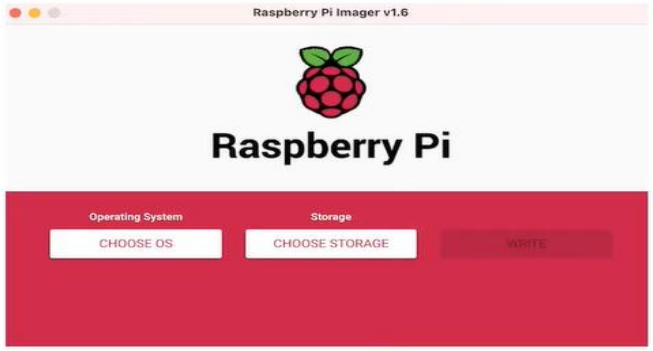
```
    # d=(s*t/2)
    # speed=s=343 m/s=34300 cm/s
distance = (timeElapsed * 34300) / 2



if distance < 30:
    os.system("mpg321 Sounds/stop.mp3")
```

Installation of Raspberry OS

Installation of Raspberry OS

Software required:

Link download in description



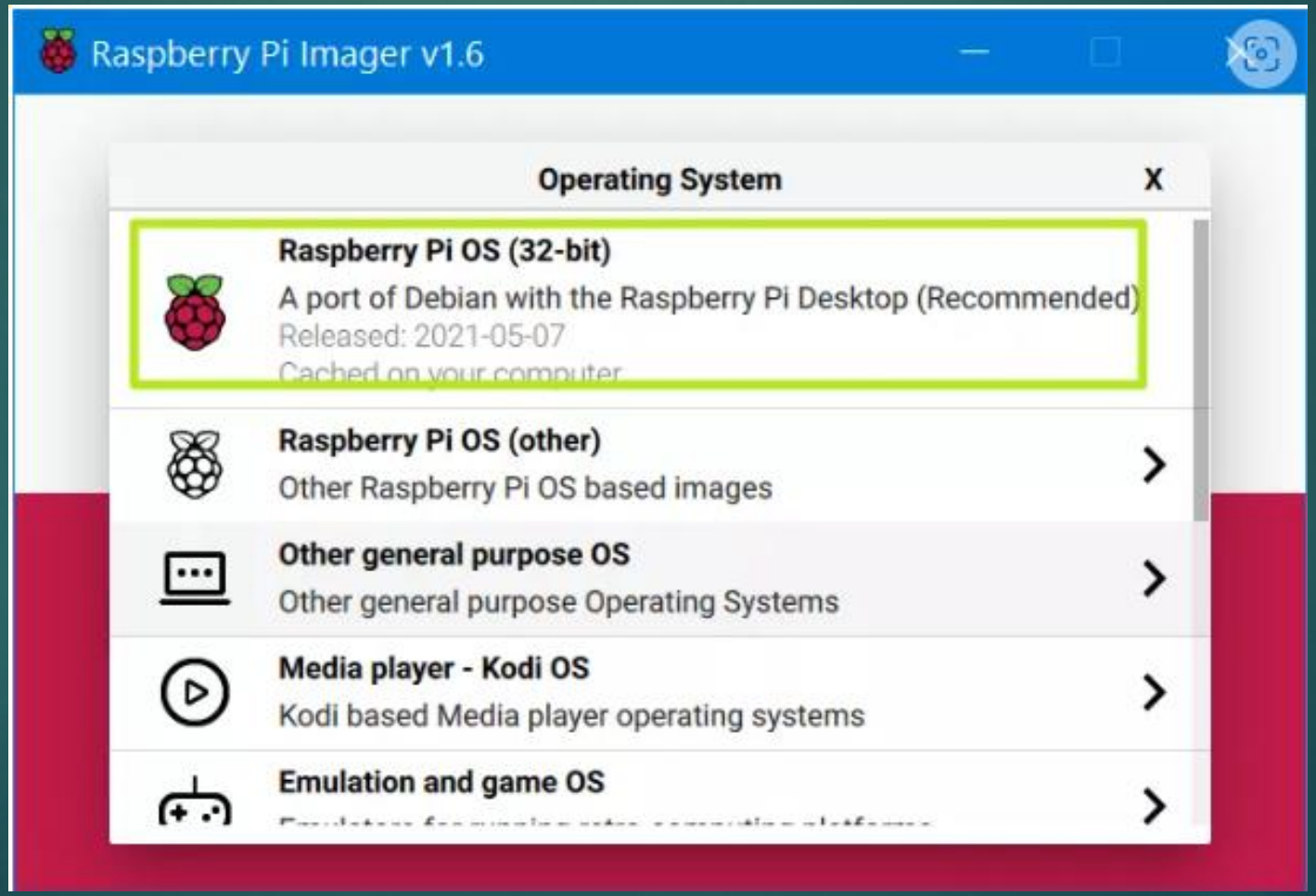
Raspberry Pi OS

VNC Viewer

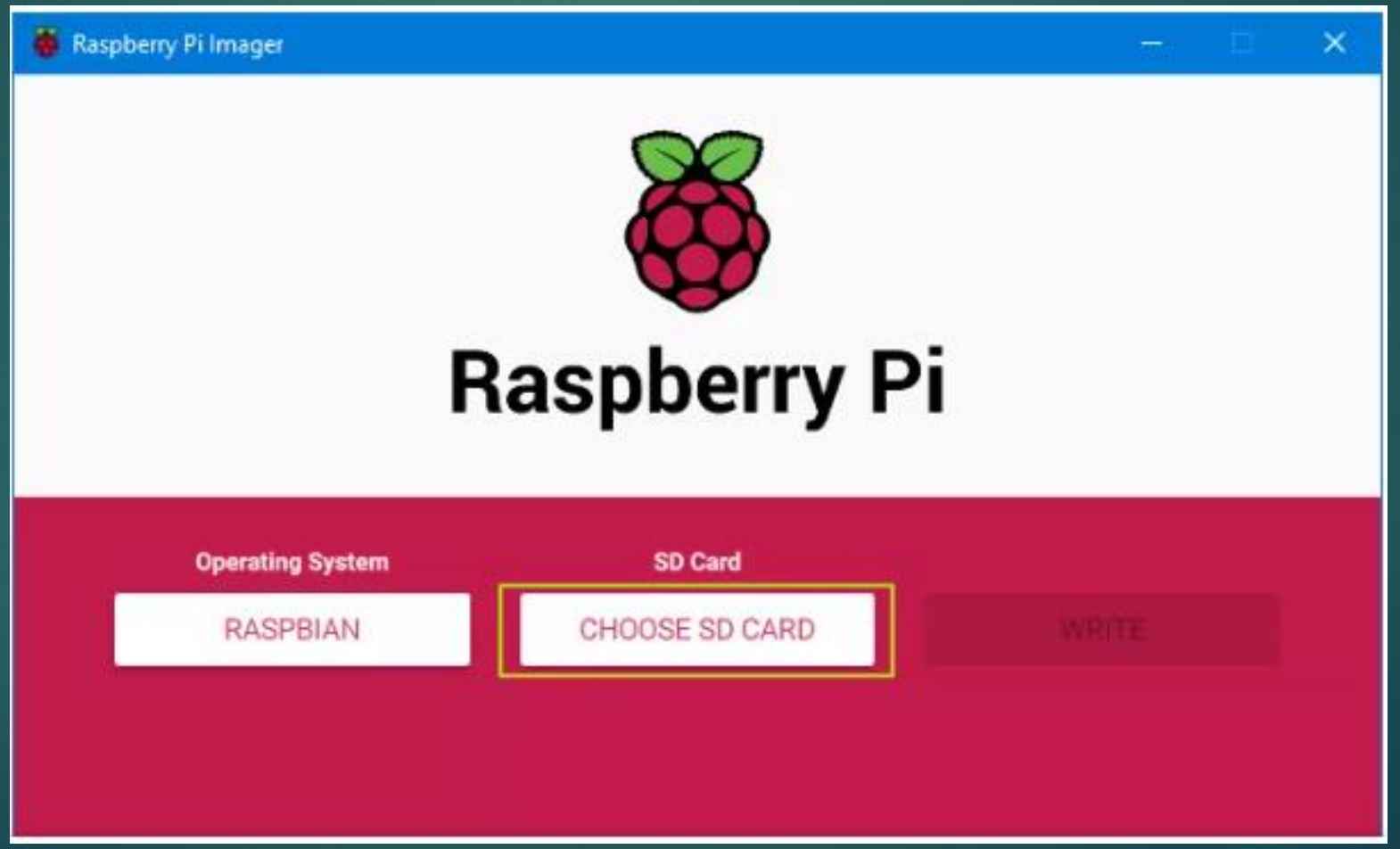
PuTTY

Installation of Raspberry OS

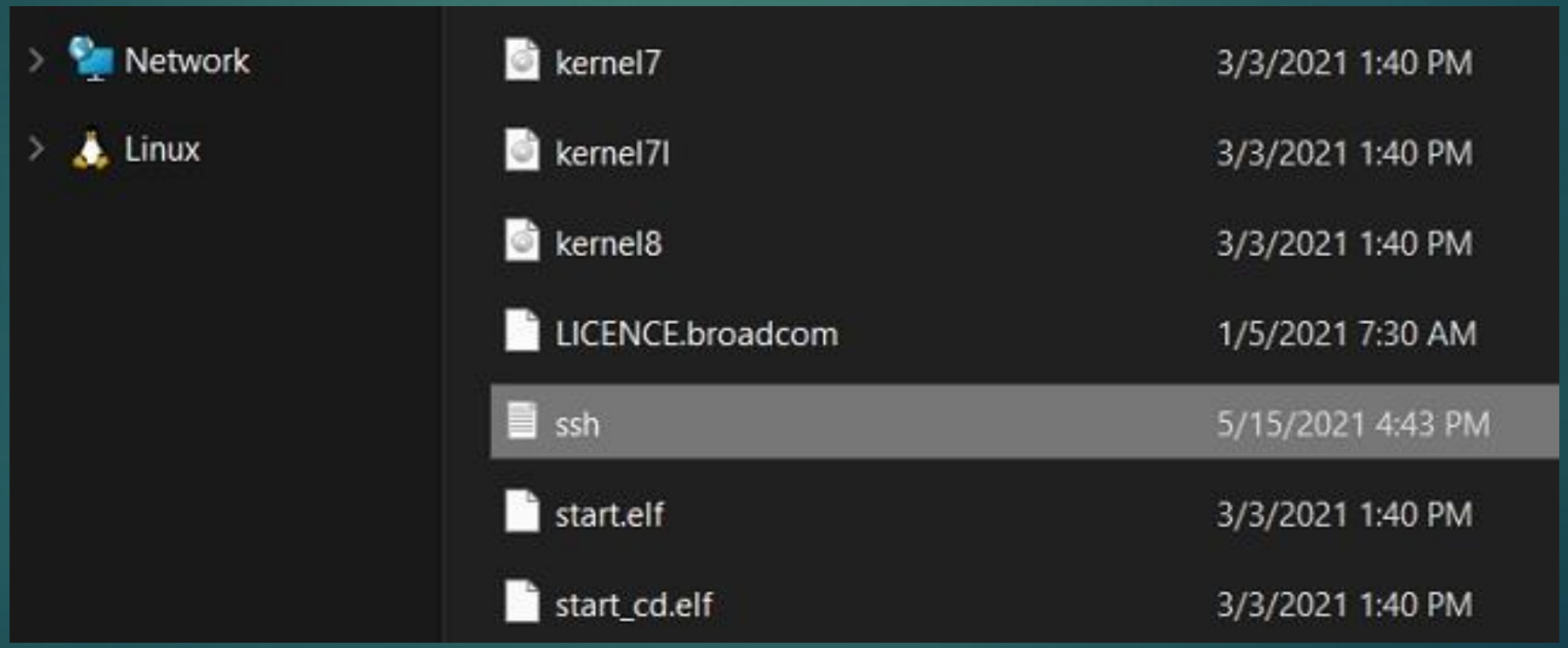
Installation of Raspberry OS



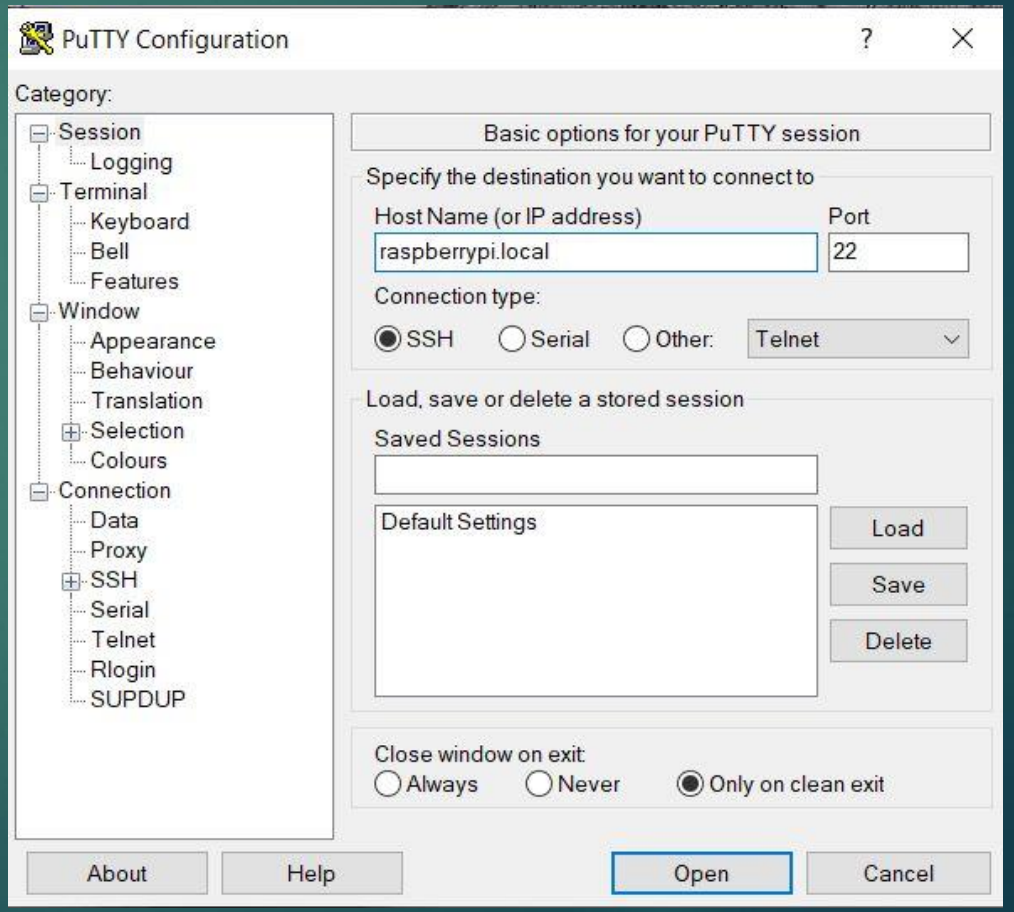
Installation of Raspberry OS



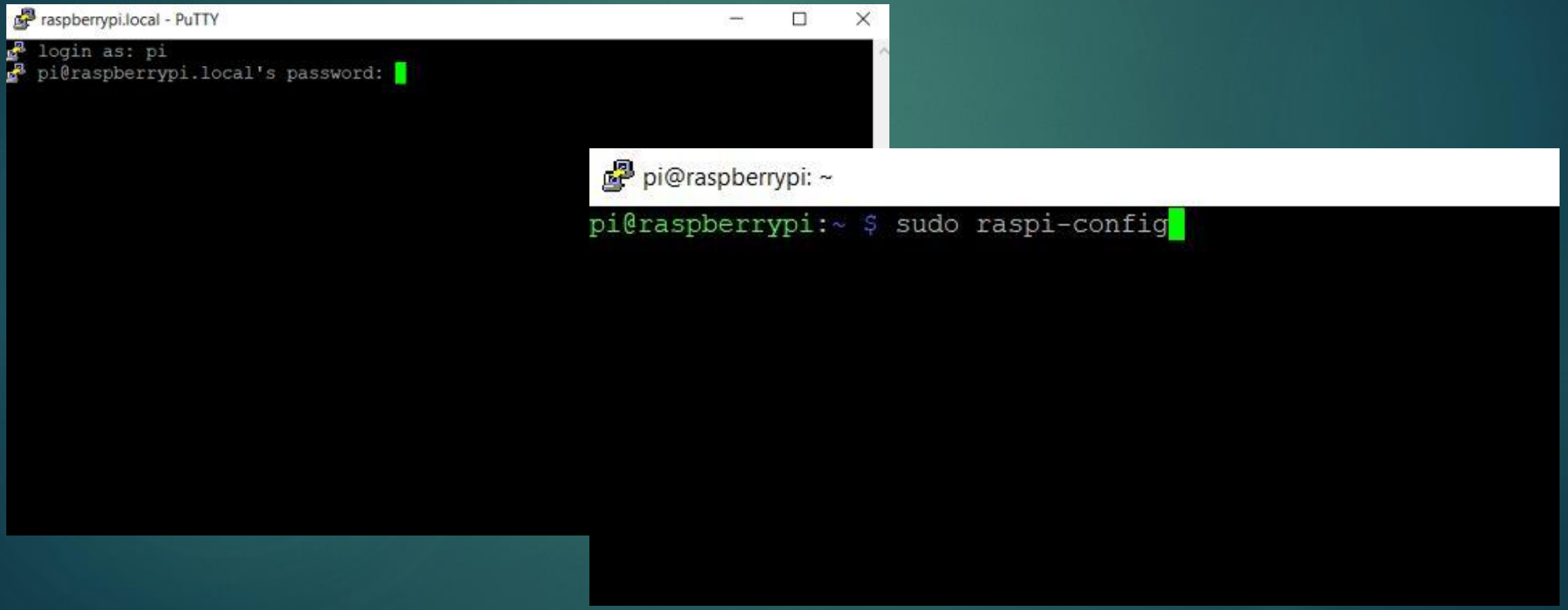
Connecting raspberry pi



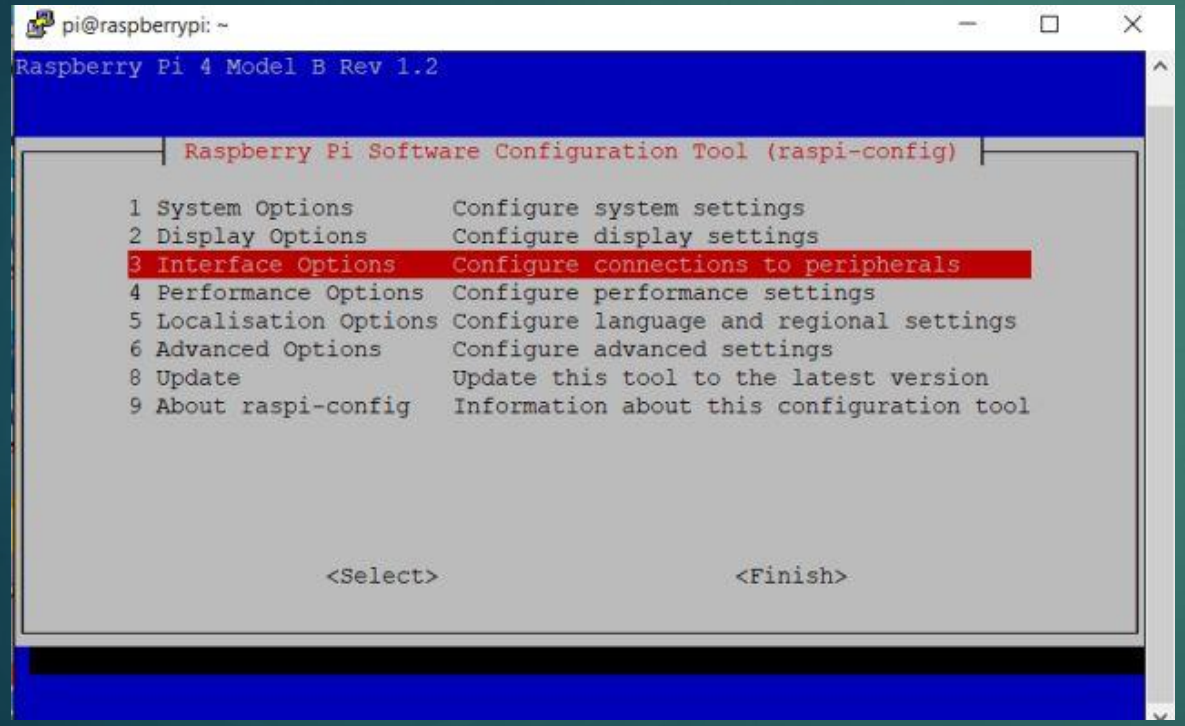
Connecting raspberry pi

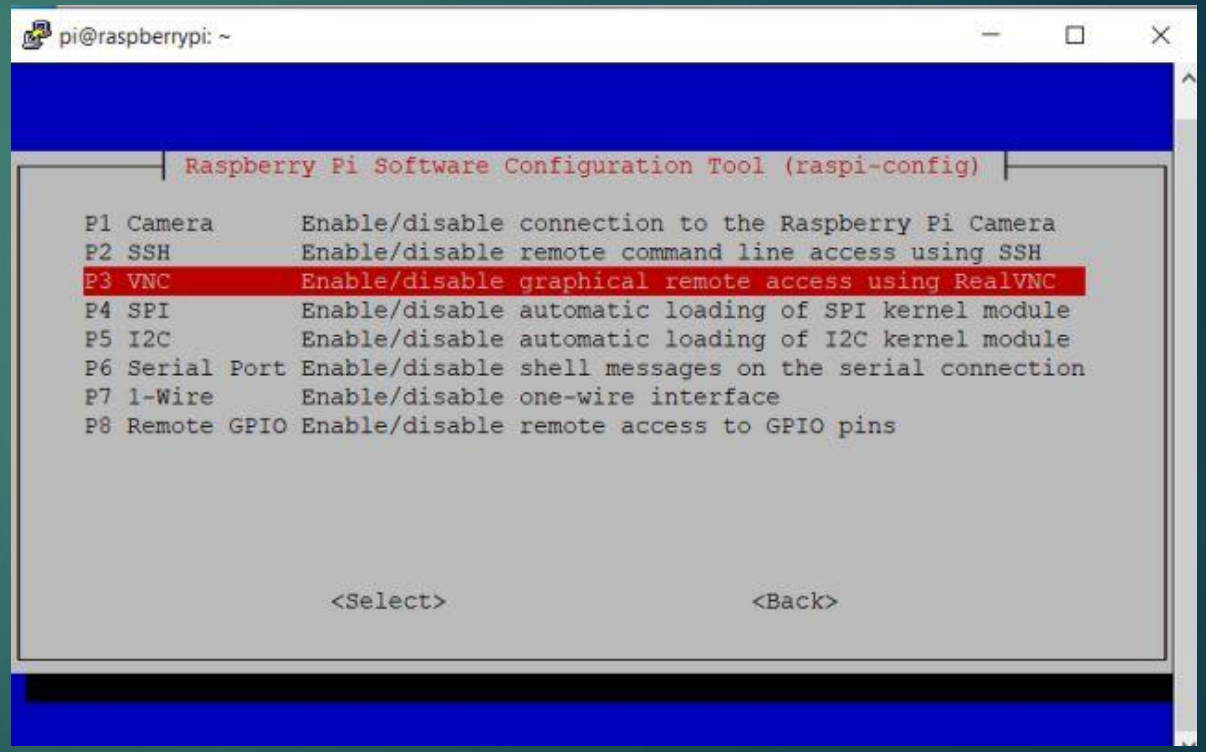


Connecting raspberry pi

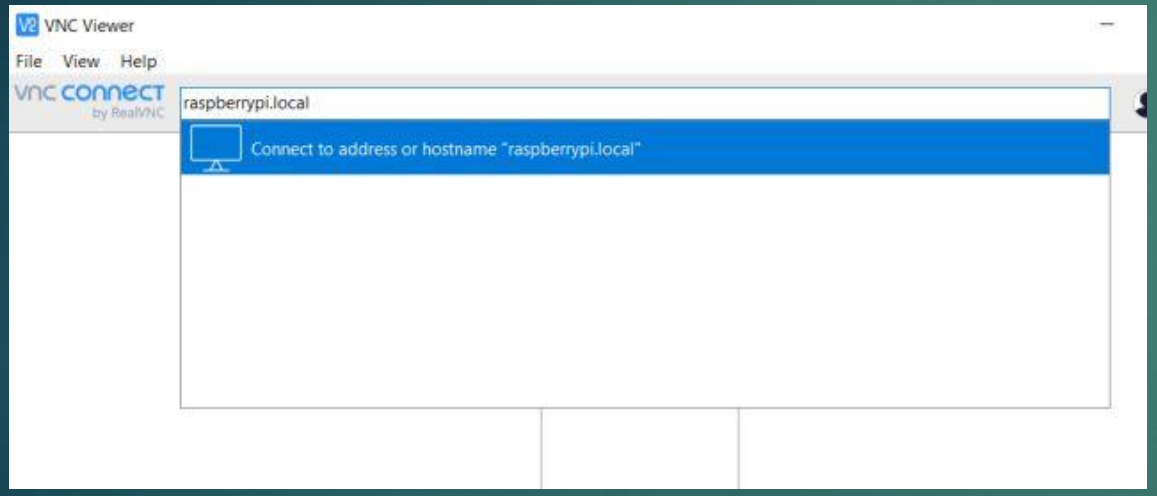


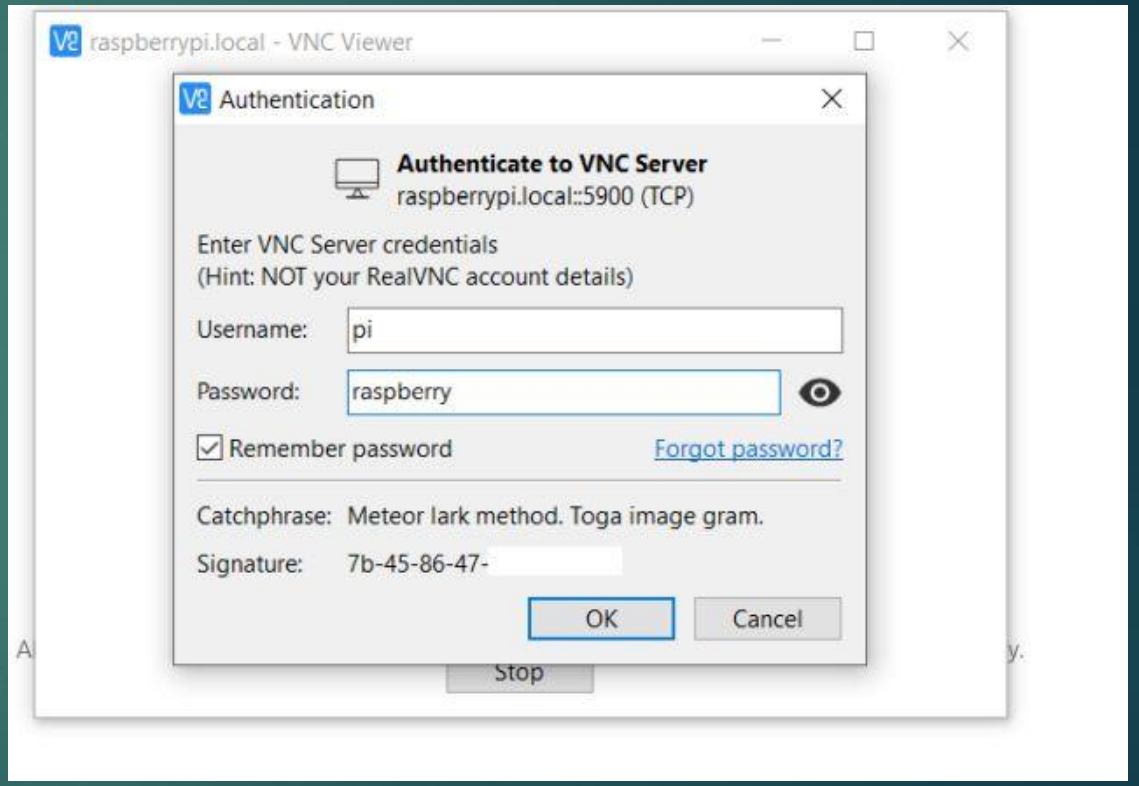
Connecting raspberry pi



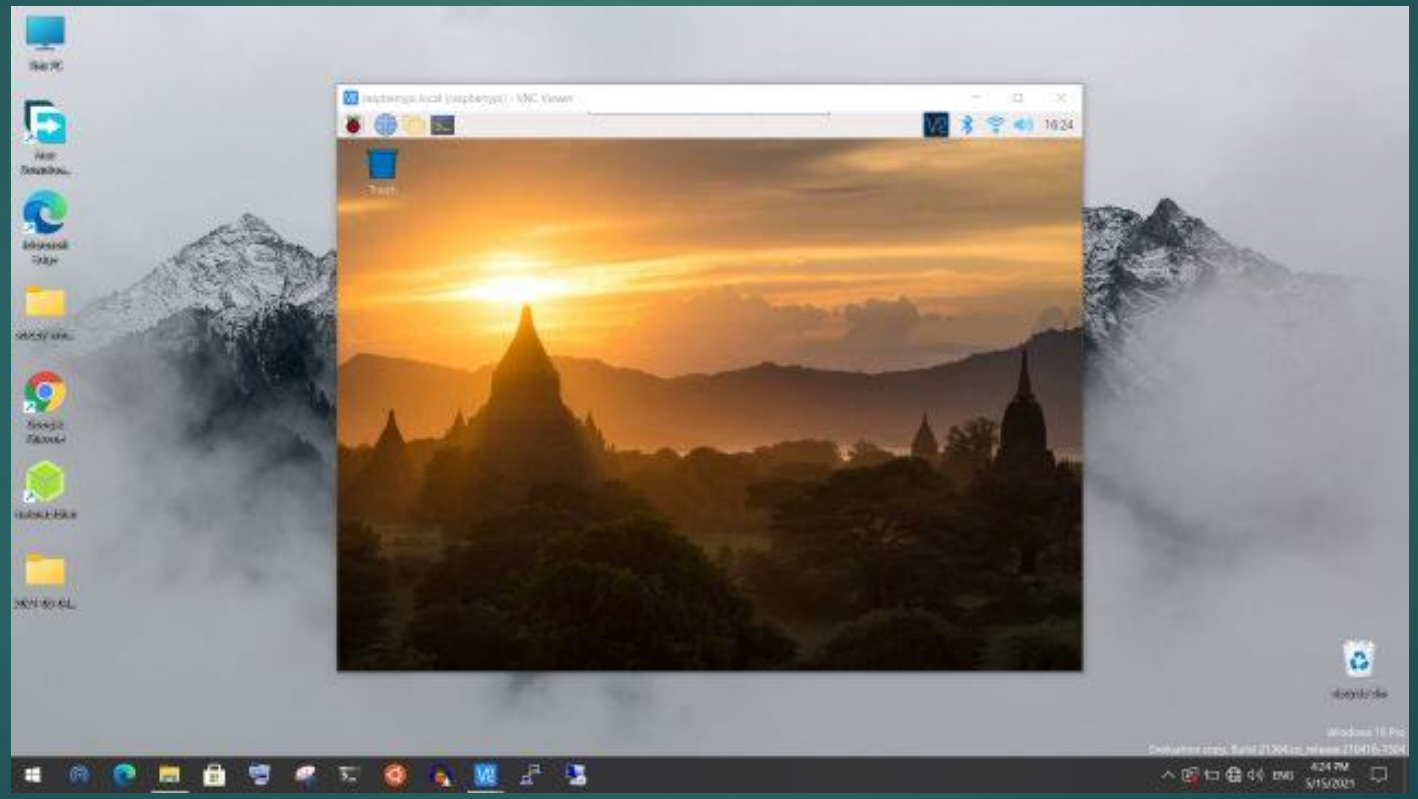


Connecting raspberry pi

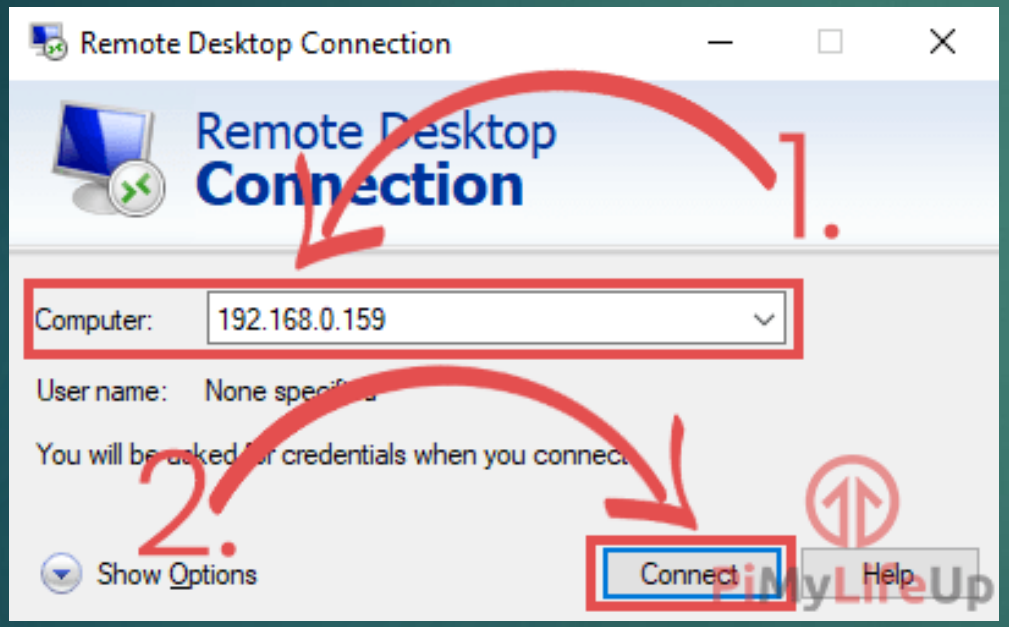


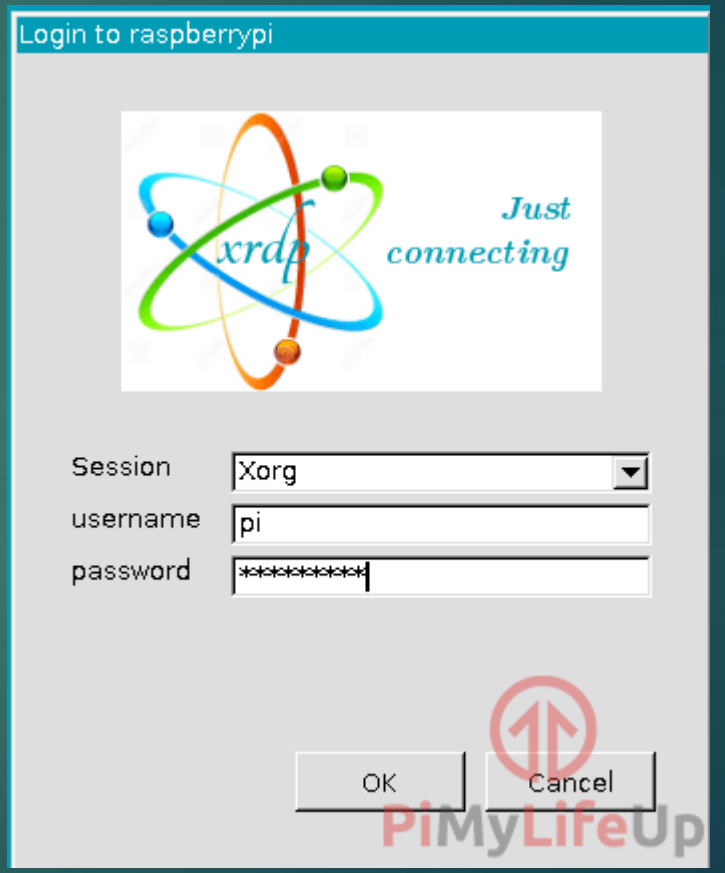


Connecting raspberry pi



Method 2 : using RDC





Open CV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.

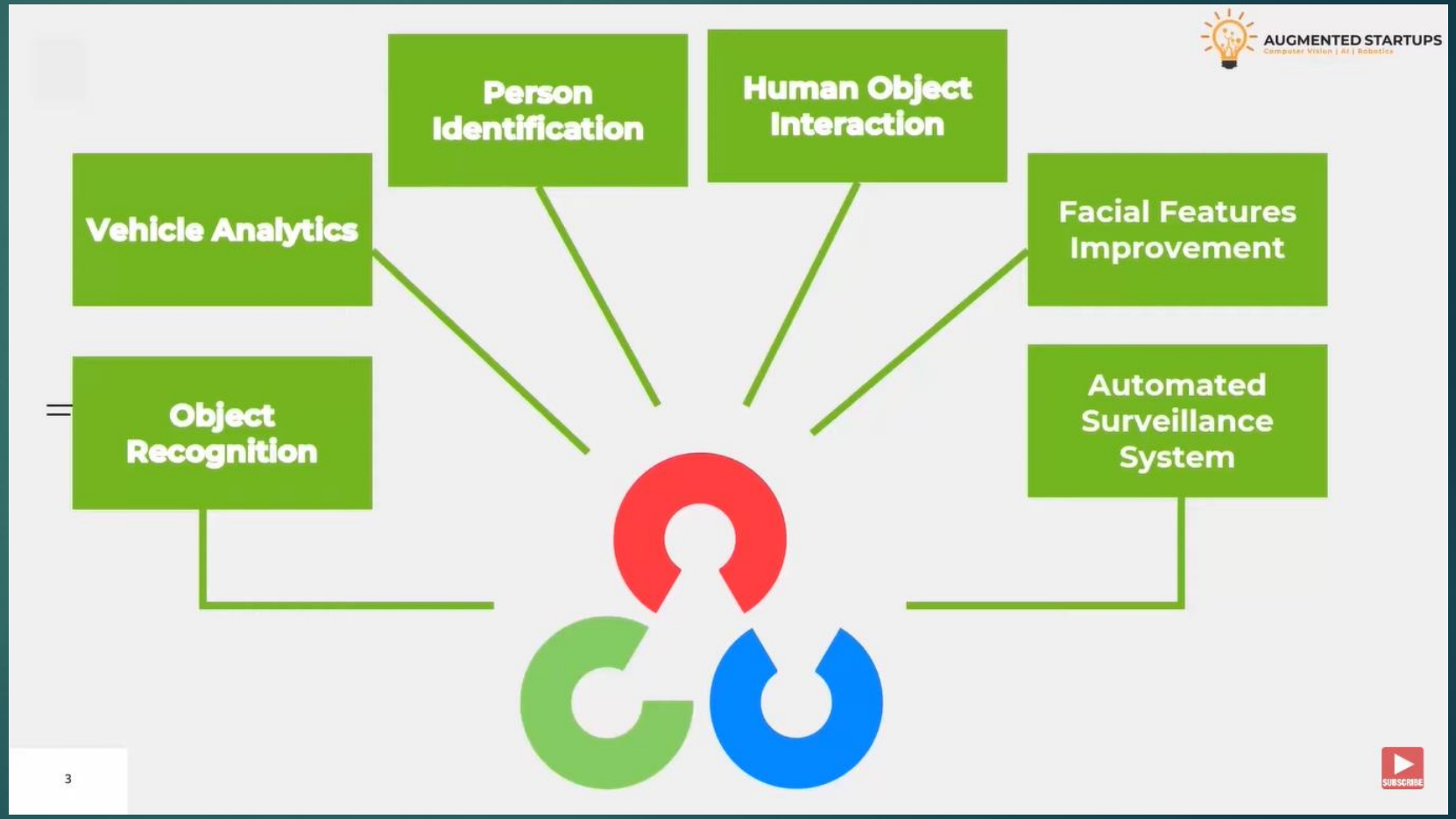
Why OpenCV?

- It contains more than 2500 optimized algorithms
- excellent accuracy in performance and speed.
- excellent accuracy in performance and speed.
- extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene
- find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery.
- OpenCV has great people and community involved as users, developers and researchers

Why OpenCV?

- the number is more than 47 thousand and estimated number of downloads exceeding 7 million.
- Many well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library.
- It has C++, C, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS.
- OpenCV leans mostly towards real-time vision applications
- . A full-featured CUDA and OpenCL interfaces are being actively developed right now.

OpenCV Applications

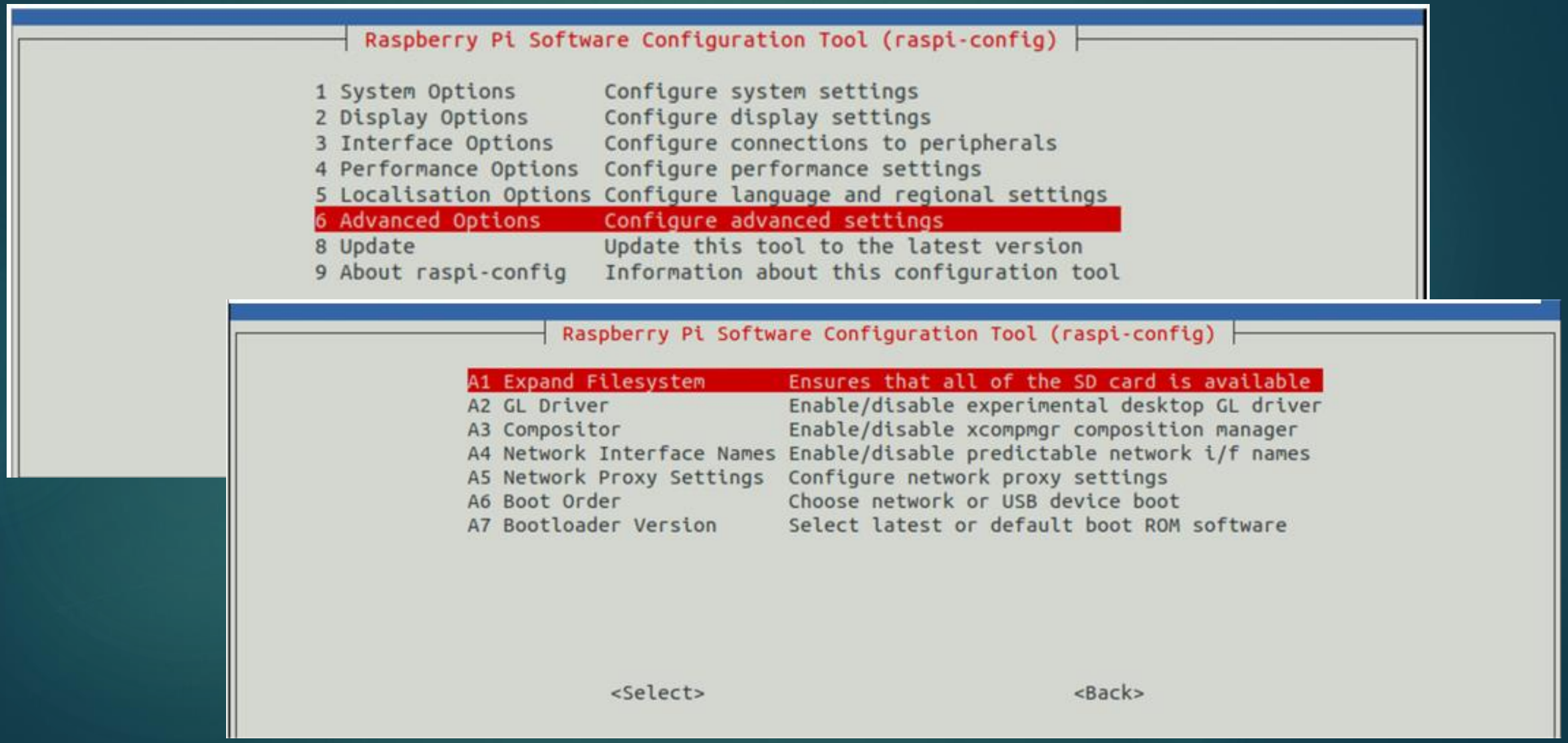


Installing OpenCV on Rasspberry pi 4

Requirements

- A Raspberry Pi 4 board
- Raspberry Pi official operating system
- An active internet connection
- A reliable power supply
- At least 8GB SD card

Expand the File system



Install OpenCV dependencies

```
sudo apt update

sudo apt upgrade
```

```
sudo apt install build-essential cmake pkg-config
```

```
sudo apt install libjpeg-dev libtiff5-dev libjasper-dev libpng-dev
```

```
sudo apt install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev

sudo apt install libxvidcore-dev libx264-dev
```

Install OpenCV dependencies cont.

```
sudo apt install libfontconfig1-dev libcairo2-dev

sudo apt install libgdk-pixbuf2.0-dev libpango1.0-dev

sudo apt install libgtk2.0-dev libgtk-3-dev
```

```
sudo apt install libatlas-base-dev gfortran
```

```
sudo apt install libhdf5-dev libhdf5-serial-dev libhdf5-103

sudo apt install libqt5gui5 libqt5webkit5 libqt5test5 python3-pyqt5
```

```
sudo apt install python3-dev          //Check our python version here
```

```
pi@raspberrypi:~ $ sudo apt-get install python3-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3-dev is already the newest version (3.7.3-1).
python3-dev set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 210 not upgraded.
pi@raspberrypi:~ $
```

Install Numpy and create a Python Virtual environment

```
sudo apt-get install python3-pip
```

```
sudo pip3 install virtualenv virtualenvwrapper
```

```
nano ~/.bashrc
```

```
# virtualenv and virtualenvwrapper

export WORKON_HOME=$HOME/.virtualenvs

export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3
```

Install Numpy and create a Python Virtual environment cont.

```
mkvirtualenv sbb_cv -p python3
```

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

Install OpenCV
1. Install OpenCV with pip on Raspberry

workon sbb_cv

```
pi@raspberrypi:~ $
pi@raspberrypi:~ $
pi@raspberrypi:~ $ workon sbb_cv
(sbb_cv) pi@raspberrypi:~ $
```

pip3 install opencv-python

```
(sbb_cv) pi@raspberrypi:~ $ pip3 install opencv-python
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Collecting opencv-python
  Using cached https://www.piwheels.org/simple/opencv-python/opencv_python-4.
5.1.48-cp37-cp37m-linux_armv7l.whl (10.5 MB)
Requirement already satisfied: numpy>=1.14.5 in ./.virtualenvs/sbb_cv/lib/pyt
hon3.7/site-packages (from opencv-python) (1.20.2)
Installing collected packages: opencv-python
Successfully installed opencv-python-4.5.1.48
(sbb_cv) pi@raspberrypi:~ $
```

Install OpenCV
2.from the source (recommended)

```
workon sbb_cv
```

1. Activate your virtual environment

2. Download the source code for both OpenCV and Opencv_contrib from Github.

```
wget -O opencv_contrib.zip https://github.com/opencv/opencv_contrib/archive/4.5.2.zip

wget -O opencv.zip https://github.com/opencv/opencv/archive/4.5.2.zip
```

3. unzip the contents of the two files we downloaded.

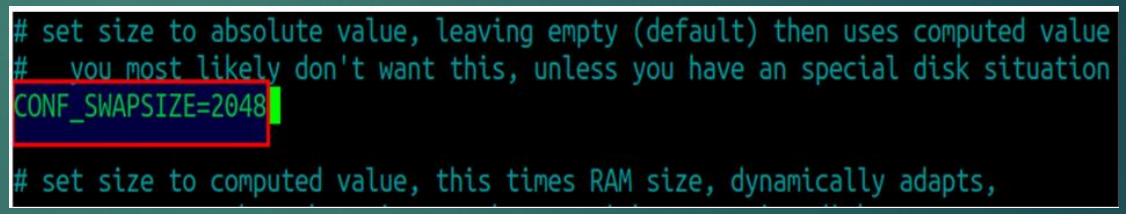
```
unzip opencv.zip

unzip opencv_contrib.zip
```

Install OpenCV from the source (cont.)

- rename these two to something memorable
- increase the SWAP space and utilize all four cores of the Pi in the compiling process.
- To apply the changes, restart the SWAP service

```
mv opencv-4.5.2 opencv

mv opencv_contrib-4.5.1 opencv_contrib
```

```
sudo nano /etc/dphys-swapfile
```




```
sudo /etc/init.d/dphys-swapfile stop

sudo /etc/init.d/dphys-swapfile start
```

Install OpenCV from the source (cont.)

- Install Numpy with the pip command
- we can now start configuring OpenCV

```
cd opencv

mkdir build

cd build

cmake -D CMAKE_BUILD_TYPE=RELEASE \
    -D CMAKE_INSTALL_PREFIX=/usr/local \
    -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules \
    -D ENABLE_NEON=ON \
    -D ENABLE_VFPV3=ON \
    -D BUILD_TESTS=OFF \
    -D INSTALL_PYTHON_EXAMPLES=OFF \
    -D OPENCV_ENABLE_NONFREE=ON \
    -D CMAKE_SHARED_LINKER_FLAGS=-latomic \
    -D BUILD_EXAMPLES=OFF ..
```

Install OpenCV from the source (cont.)

- compiling with all the Four cores of the Pi
- install OpenCV
- Restart the swap service
- create symbolic links of cv2 to our virtual environment

```
make -j4
```

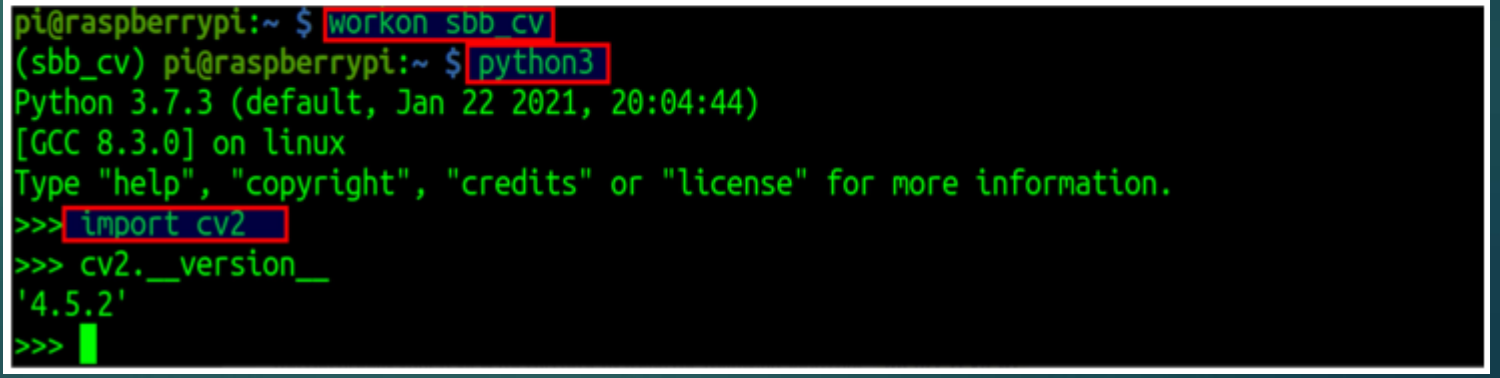
```
sudo make install

sudo ldconfig
```

```
sudo /etc/init.d/dphys-swapfile stop

sudo /etc/init.d/dphys-swapfile start
```

```
cd /usr/local/lib/python3.7/site-packages/cv2/python-3.7/

sudo mv cv2.cpython-37m-arm-linux-gnueabihf.so cv2.so

cd ~/.virtualenvs/sbb_cv/lib/python3.7/site-packages/

ln -s /usr/local/lib/python3.7/site-packages/cv2/python-3.7/cv2.so cv2.so
```

Testing OpenCV

```
$ workon sbb_cv

$ python3

>>> import cv2

>>> cv2.__version__

'4.5.2'

>>>
```



Face Recognition

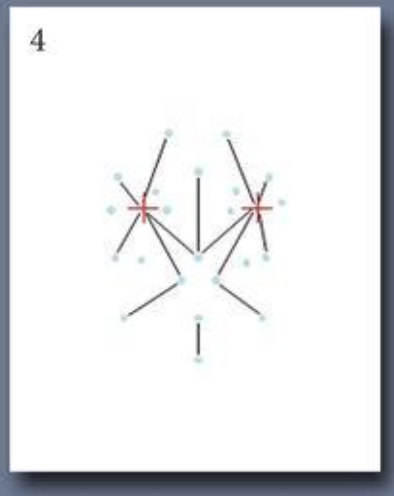
Face Recognition

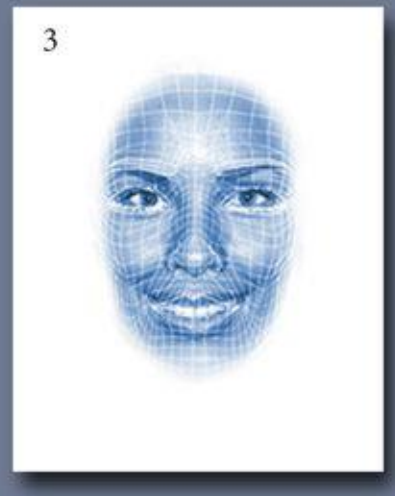
- Facial recognition is a way of recognizing a human face through technology. A facial recognition system uses biometrics to map facial features from a photograph or video. It compares the information with a database of known faces to find a match. Facial recognition can help verify a person's identity.

How Face Recognition works

How facial identification works

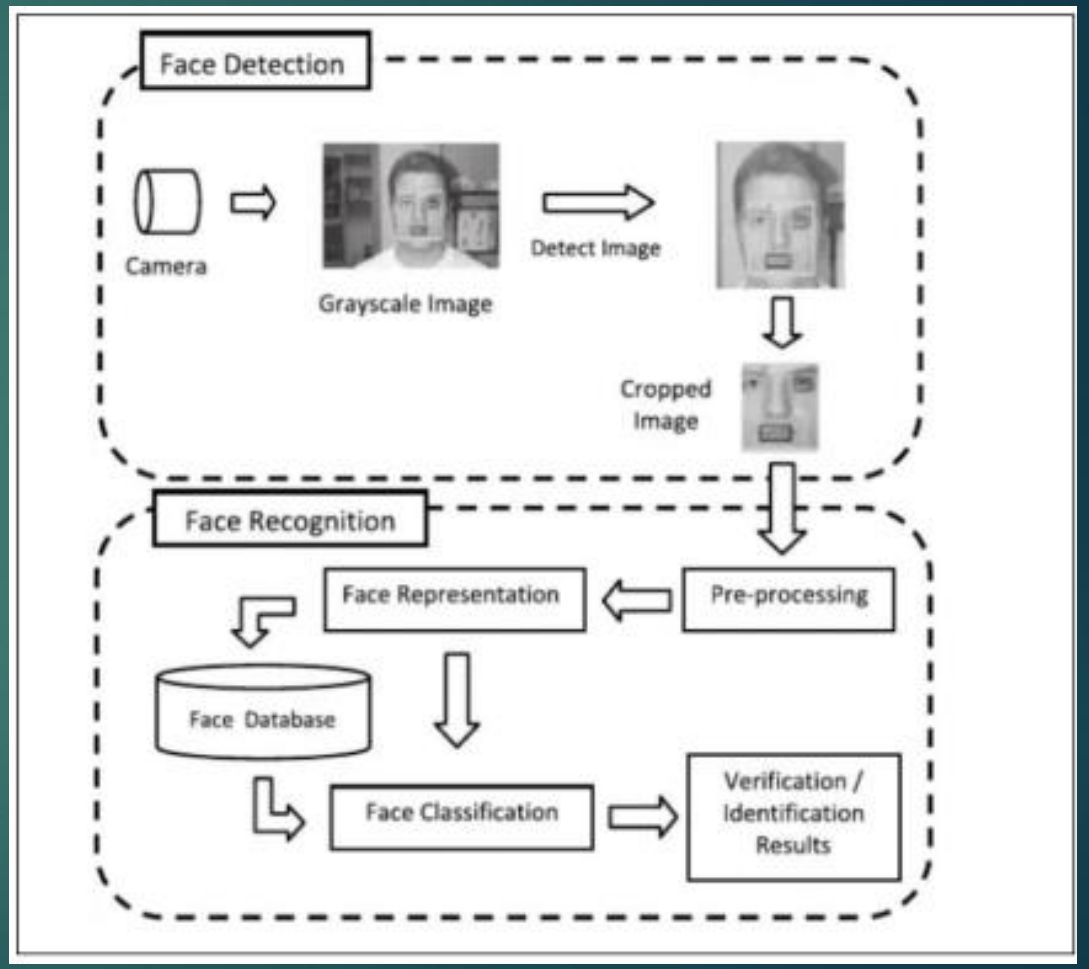
1. Image is captured
2. Eye locations are determined
3. Image is converted to grayscale and cropped
4. Image is converted to a template used by the search engine for facial comparison results
5. Image is searched and matched using a sophisticated algorithm to compare the template to other templates on file
6. Duplicate licenses are investigated for fraud





How Face Recognition works

- Pre-processing :To reduce the variability in the faces, the images are processed before they are fed into the network.
- All positive examples that is the face images are obtained by cropping images with frontal faces to include only the front view. All the cropped images are then corrected for lighting through standard algorithms.
- Classification : Neural networks are implemented to classify the images as faces or non faces by training on these examples and if there is face in image compares It with database of known faces to find a match .



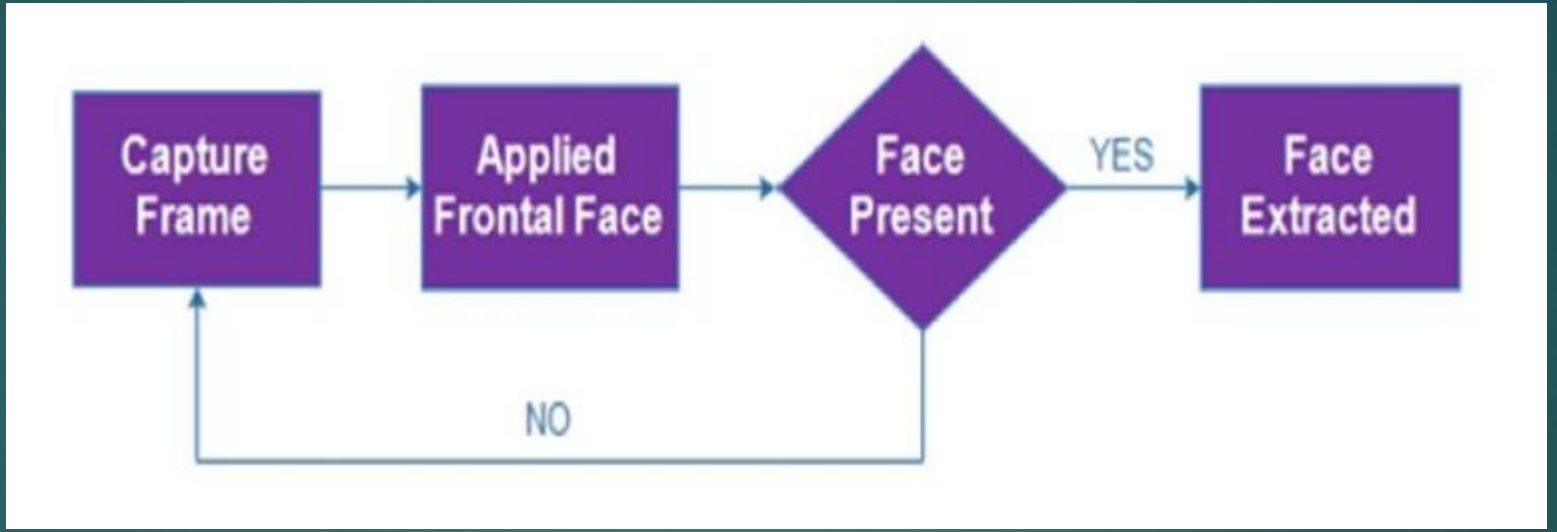
Face Recognition Applications

- **Social media like**
  - Facebook & Snapchat.
- **ID verification**
  - Used in banks, Many companies and other e-business.
- **Face ID**
  - Used by apple which use infra red for face recognition

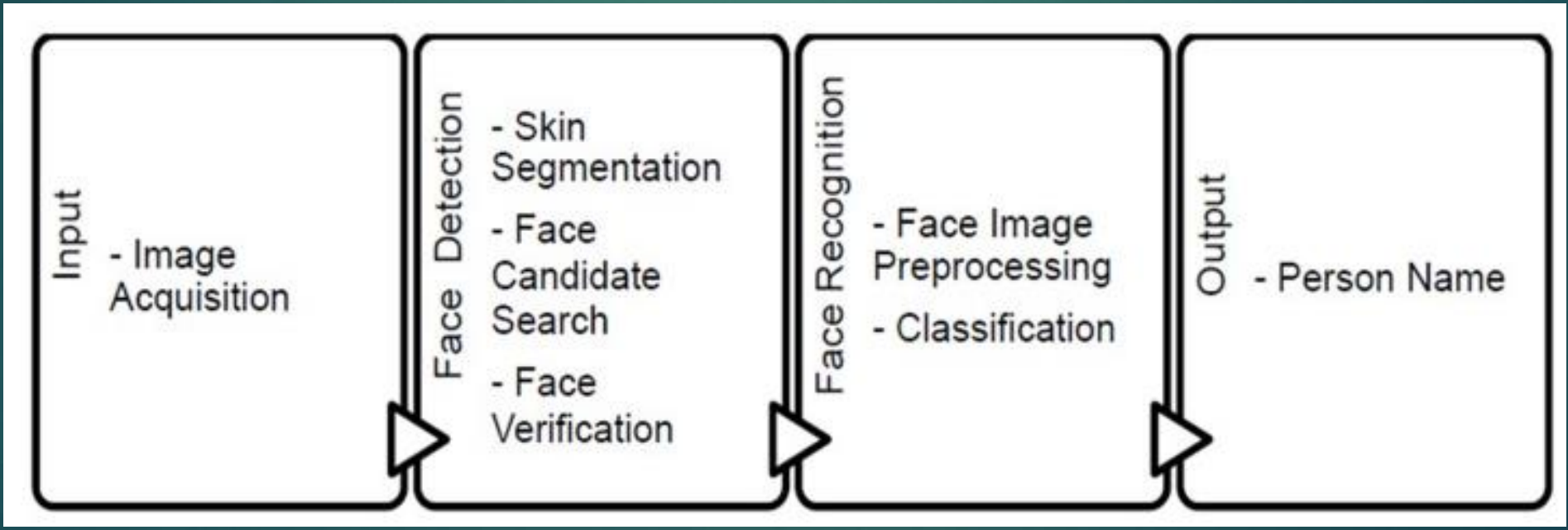
Frontal Face Haar Cascade

It is pre-trained cascade classifiers to detect the frontal facial features of a person as shown in the next Figure. The idea behind this cascade is to feed the classifier with a large number of images having faces i.e. positive images and images that don't contain any face known as negative images

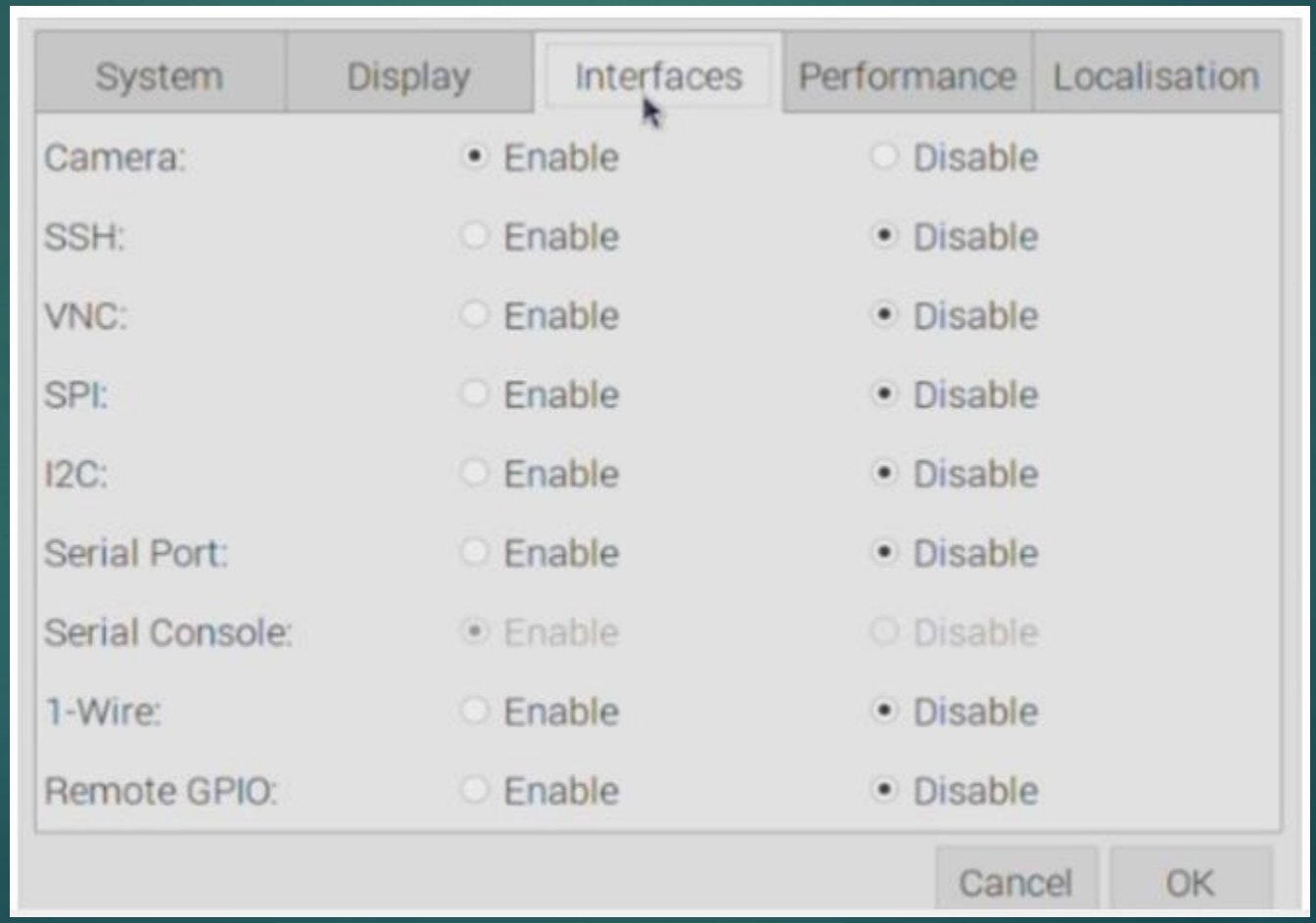
Feature extraction



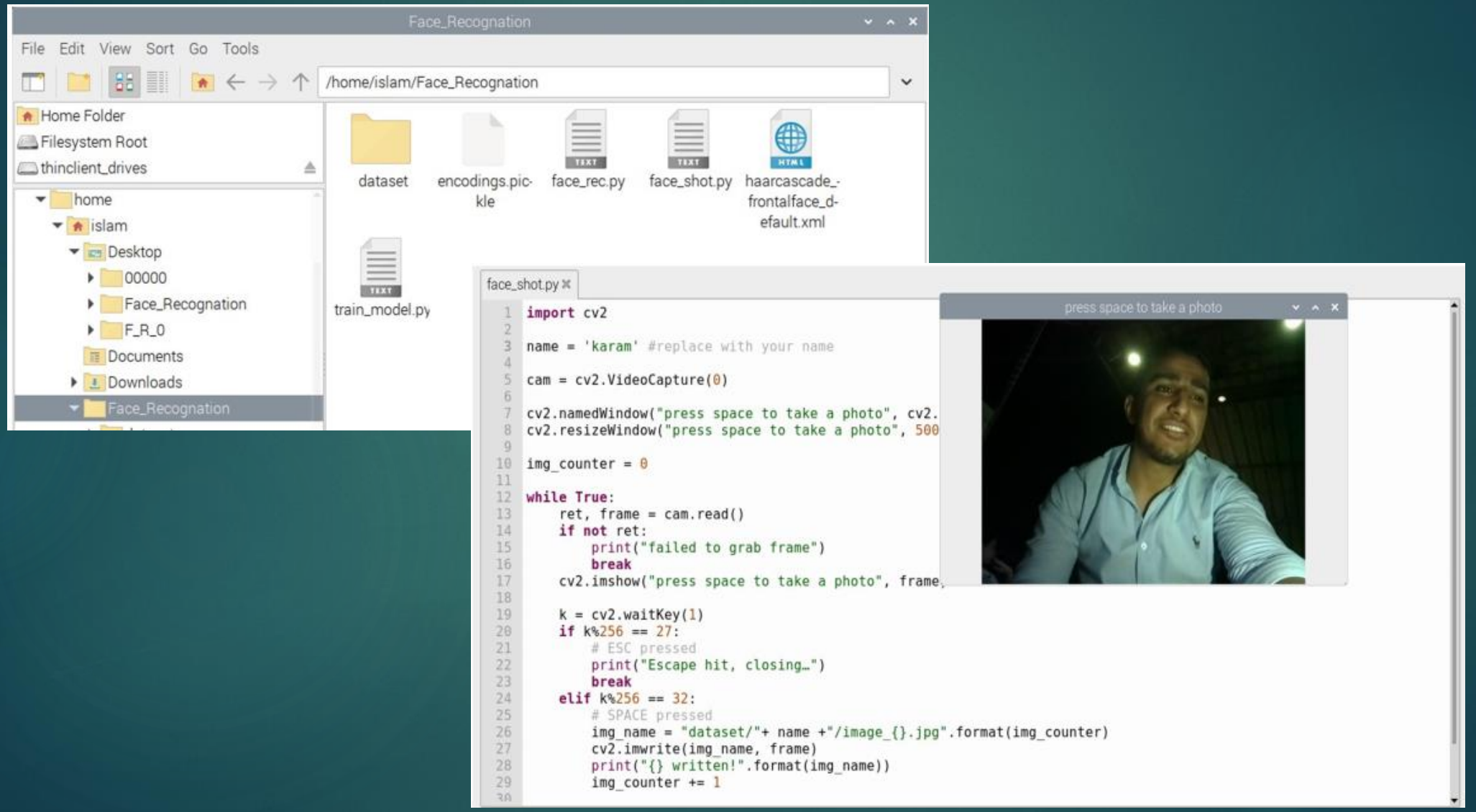
Design of a face recognition system



Training the System



Training the System



Training the System

face_shot.py | train_model.py

```
1 # import the necessary packages
2 from imutils import paths
3 import face_recognition
```

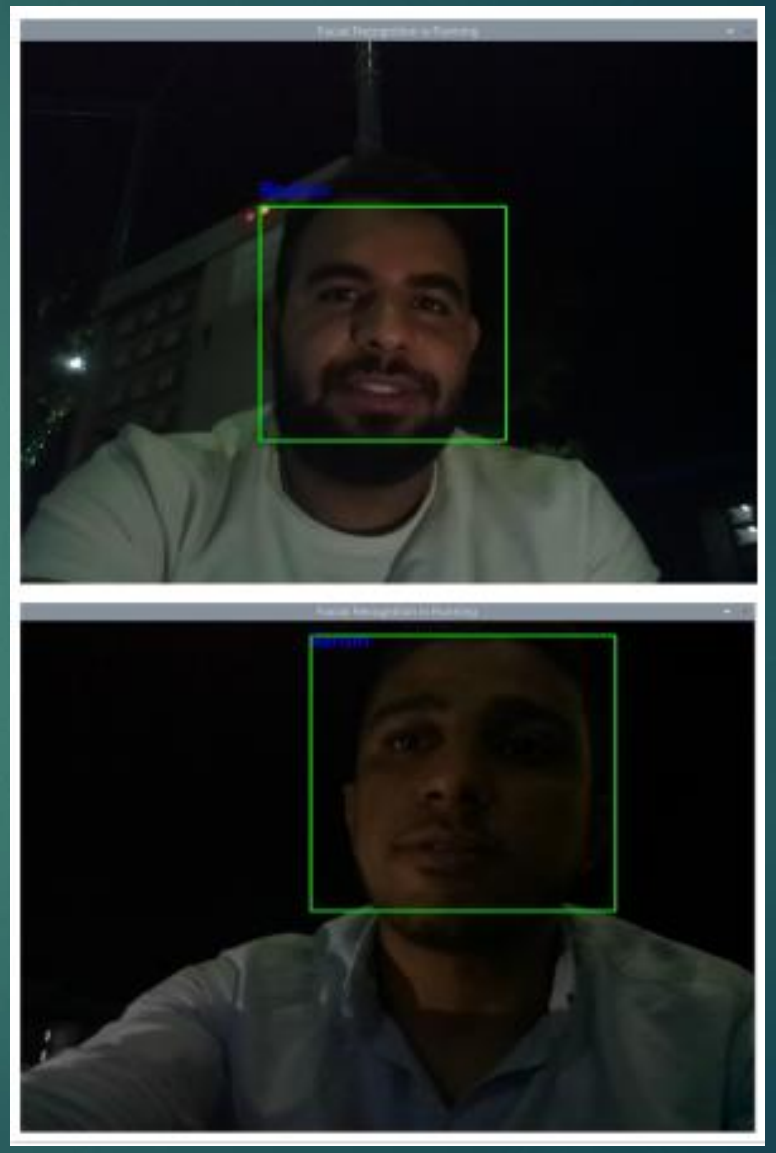
Shell

```
[INFO] processing image 1/30
[INFO] processing image 2/30
[INFO] processing image 3/30
[INFO] processing image 4/30
[INFO] processing image 5/30
[INFO] processing image 6/30
[INFO] processing image 7/30
[INFO] processing image 8/30
[INFO] processing image 9/30
[INFO] processing image 10/30
[INFO] processing image 11/30
[INFO] processing image 12/30
[INFO] processing image 13/30
[INFO] processing image 14/30
[INFO] processing image 15/30
[INFO] processing image 16/30
[INFO] processing image 17/30
[INFO] processing image 18/30
[INFO] processing image 19/30
[INFO] processing image 20/30
[INFO] processing image 21/30
[INFO] processing image 22/30
[INFO] processing image 23/30
[INFO] processing image 24/30
[INFO] processing image 25/30
[INFO] processing image 26/30
[INFO] processing image 27/30
[INFO] processing image 28/30
[INFO] processing image 29/30
[INFO] processing image 30/30
[INFO] serializing encodings...
```

Results

```
Shell
Python 3.9.2 (/usr/bin/python3)
>>> %cd /home/islam/Desktop/Face_Recognation
>>> %Run face_rec.py

  [INFO] loading encodings + face detector…
  [INFO] starting video stream…
```

We used pyttsx3 to transform Person name to speech.



3D printing Design

What is 3D Printing?

3D printing or additive manufacturing is a process of making three dimensional solid objects from a digital file.

The creation of a 3D printed object is achieved using additive processes. In an additive process an object is created by laying down successive layers of material until the object is created. Each of these layers can be seen as a thinly sliced cross-section of the object.

3D printing is the opposite of subtractive manufacturing which is cutting out / hollowing out a piece of metal or plastic with for instance a milling machine.

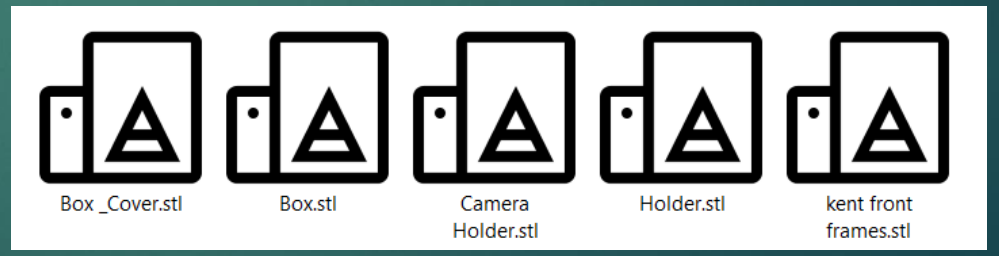
3D printing enables you to produce complex shapes using less material than traditional manufacturing methods.

What is the best CAD software to design 3D prints?

We Use Auto desk Fusion360 : It is great for designing and creating efficient mechanical parts

STL (file format)

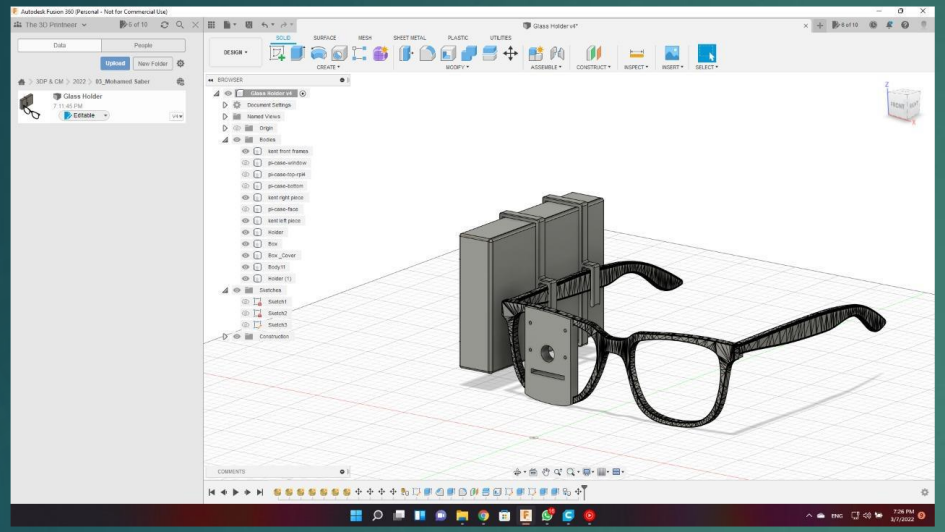
STL is a file format native to the stereolithography CAD software created by 3D Systems. STL means "Standard Triangle Language" . This file format is supported by many other software packages; it is widely used for rapid prototyping, 3D printing and computer-aided manufacturing.

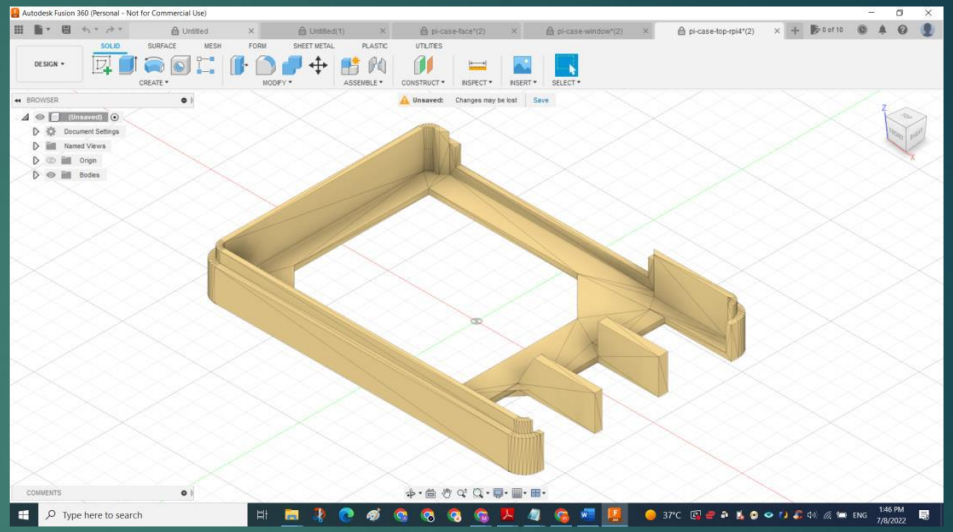


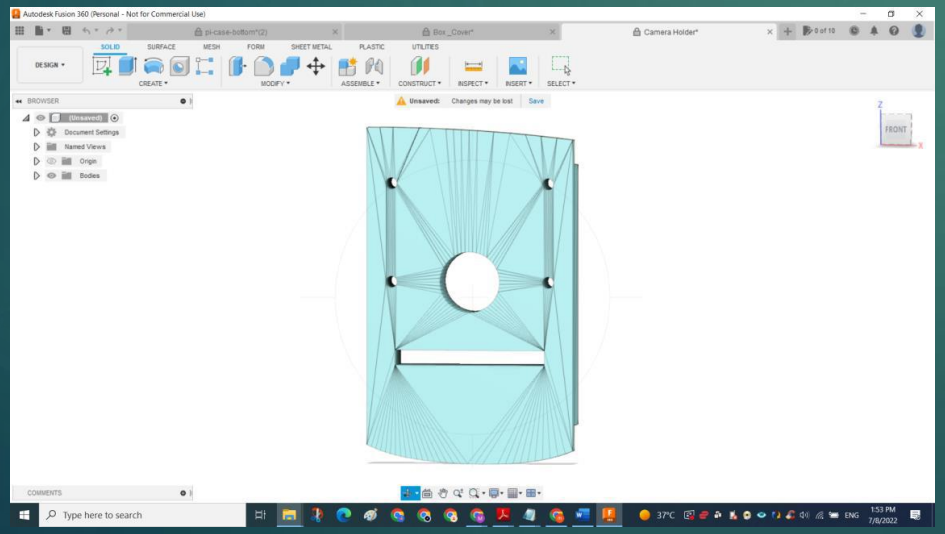
The material used in 3D printing (PLA+)

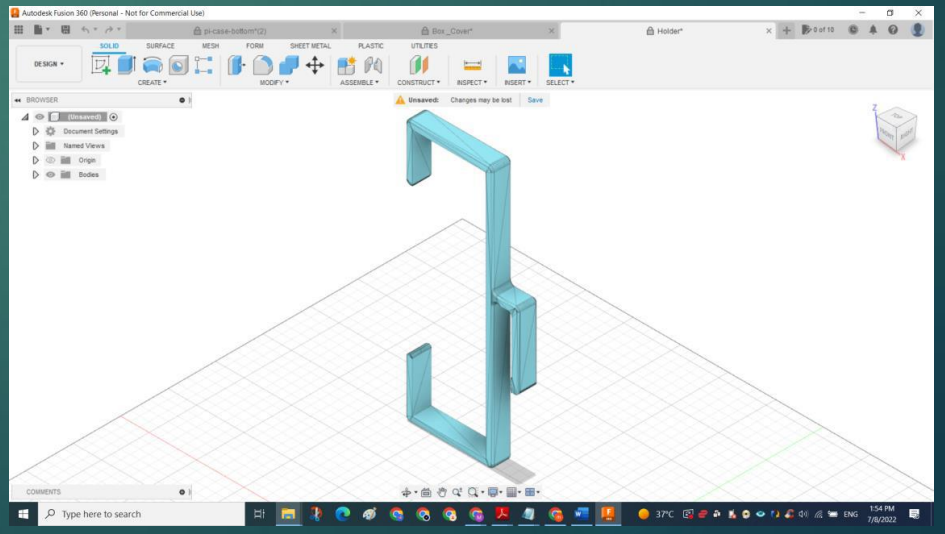
PLA filament, the most popular 3D printing material out there. Polylactic acid, commonly referred to as PLA, is a bioplastic and thermoplastic made from natural materials like corn starch. It's most known for its use in the 3D printing industry, sold in spools of a stranded material known as filament, but PLA plastic is also used in other manufacturing contexts due to its unique properties

Smart Glasses Model

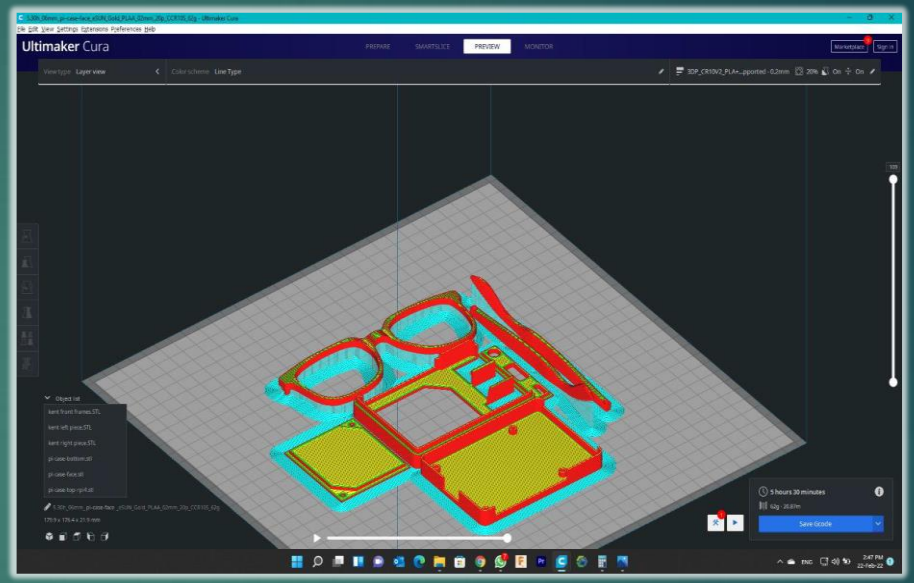


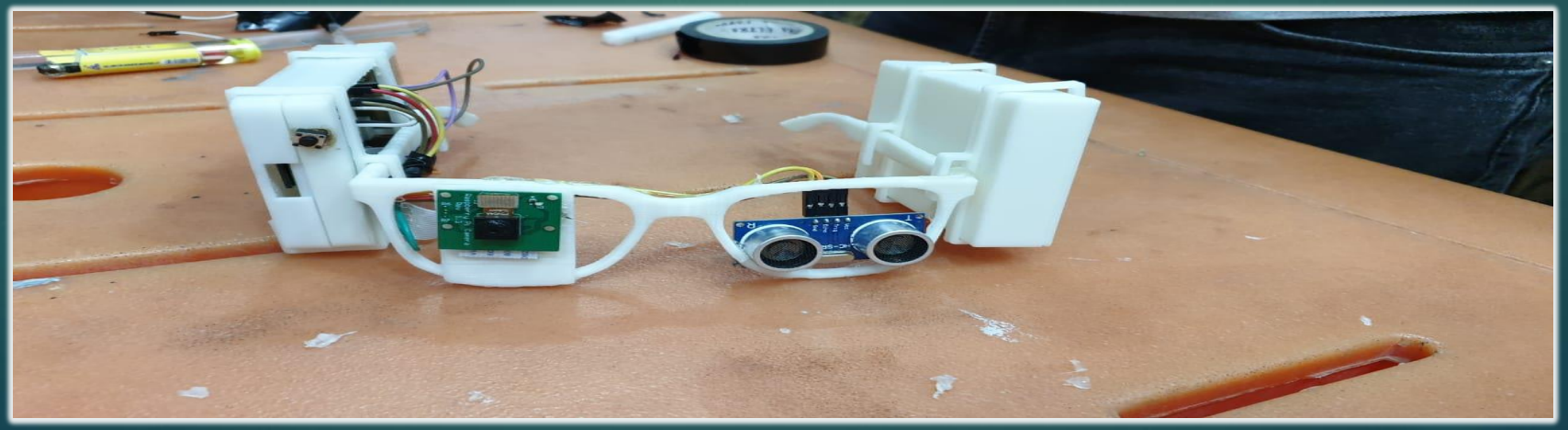






Smart Glasses





Optical character recognition (OCR)

What is OCR (Optical Character Recognition)?

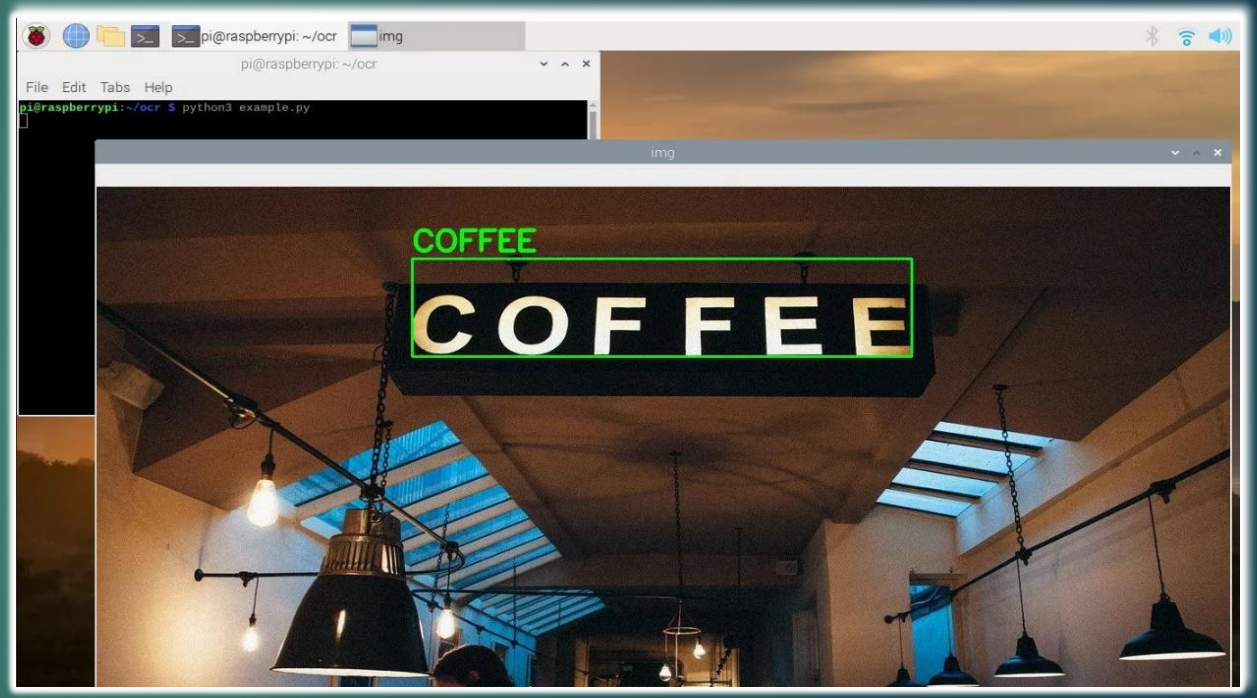
Optical Character Recognition (OCR) is the process that converts an image of text into a machine-readable text format.

What is OCR used for ?

- Banking
- Healthcare
- Logistics

types of OCR

- Simple optical character recognition software
- Intelligent character recognition software
- Intelligent word recognition.
- Optical mark recognition

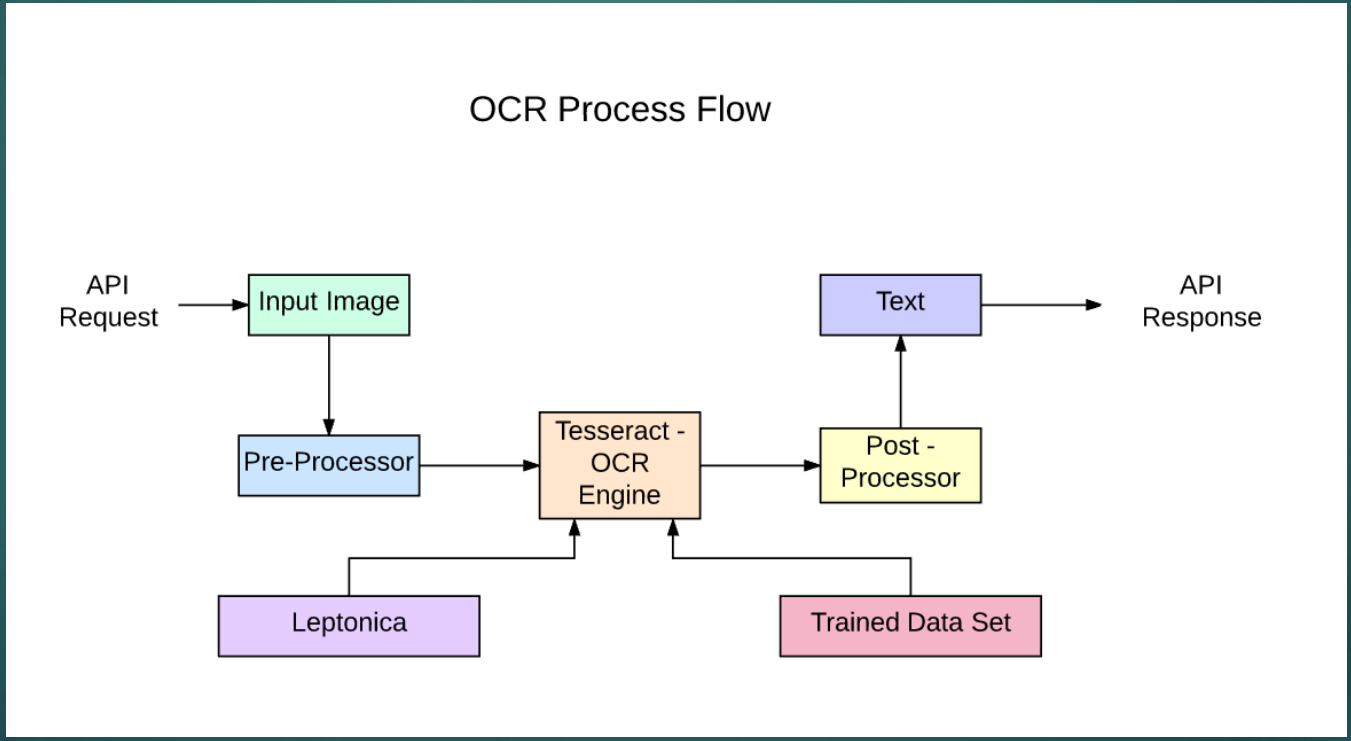


Open-Source OCR Tools

- Tesseract
- SwiftOCR
- OCRopus

Tesseract OCR

Tesseract is an open source text recognition (OCR) Engine, It can be used directly, or (for programmers) using an API to extract printed text from images. It supports a wide variety of languages. Tesseract is compatible with many programming languages and frameworks. It can be used with the existing layout analysis to recognize text within a large document, or it can be used in conjunction with an external text detector to recognize text from an image of a single text line.



OCR with Pytesseract and OpenCV

Pytesseract or Python-tesseract is an OCR tool for python that also serves as a wrapper for the Tesseract-OCR Engine. It can read and recognize text in images and is commonly used in python ocr image to text use cases

```
1   import cv2 import pytesseract
2   img = cv2.imread('image.jpg')
3   # Adding custom options custom_config = r'--oem 3 --psm 6'
4   #  pytesseract.image_to_string(img, config=custom_config)
```

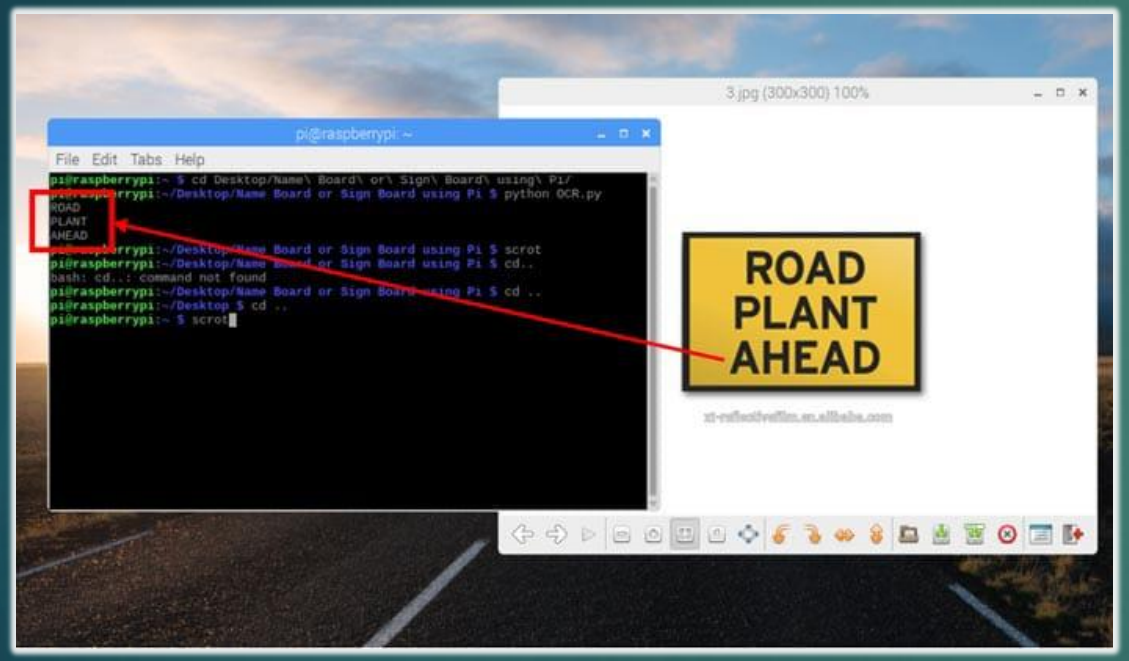
Text to speech

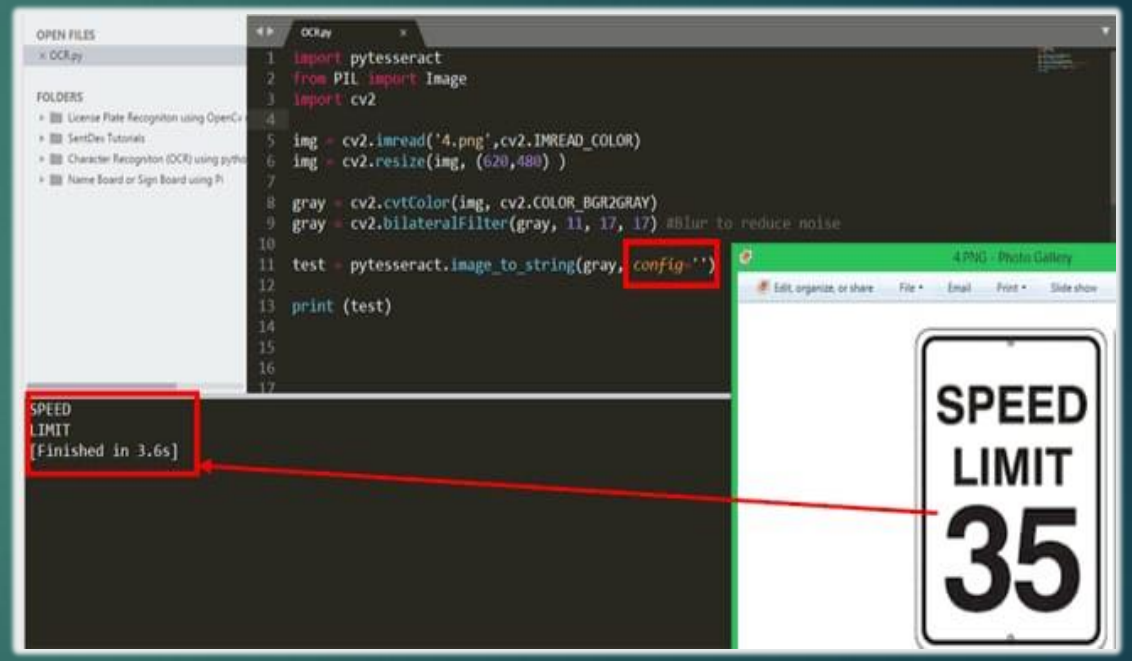
libbrary	gtts	pyttsx
How to work?	Google translate API	Independent engine
On/off line	online	offline

Usage:

```
import pyttsx3
engine = pyttsx3.init()
engine.say("I will speak this text")
engine.runAndWait()
```

Result


3.jpg (300x300) 100%
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~ $ cd Desktop/Name\ Board\ or\ Sign\ Board\ using\ Pi/
pi@raspberrypi:~/Desktop/Name Board or Sign Board using Pi $ python OCR.py
ROAD
PLANT
AHEAD
pi@raspberrypi:~/Desktop/Name Board or Sign Board using Pi $ scrot
pi@raspberrypi:~/Desktop/Name Board or Sign Board using Pi $ cd..
bash: cd..: command not found
pi@raspberrypi:~/Desktop/Name Board or Sign Board using Pi $ cd ..
pi@raspberrypi:~/Desktop $ cd ..
pi@raspberrypi:~ $ scrot
ROAD
PLANT
AHEAD


OPEN FILES
OCR.py
FOLDERS
OCR.py
1 import pytesseract
2 from PIL import Image
3 import cv2
4
5 img = cv2.imread('4.png',cv2.IMREAD_COLOR)
6 img = cv2.resize(img, (620,480))
7
8 gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
9 gray = cv2.bilateralFilter(gray, 11, 17, 17) #Blur to reduce noise
10
11 test = pytesseract.image_to_string(gray, config='')
12
13 print (test)
SPEED
LIMIT
[Finished in 3.6s]
SPEED
LIMIT
35