Speak, OpenCV and ROS

ROS INTRODUCTION COURSE

OUTLINE

Speak

- Understanding the mapping and navigation concept
- Setup Software Environment

OpenCV + dnn

- SLAM
- Creating a map with slam gmapping package
- Save the map and Spawn the robot

With ROS

- AMCL Localization
- Install and testing Kinect (3D sensor)
- Autonomous driving with a simple program

KASETSART
UNIVERSITY

KU

KASETSART
UNIVERSITY

OpenCV

Image processing in Python?

Python provides lots of libraries for image processing, including —

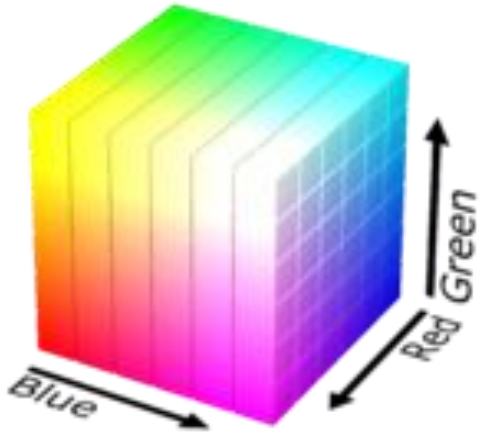
OpenCV — Image processing library mainly focused on real-time computer vision with application in wide-range of areas like 2D and 3D feature toolkits, facial & gesture recognition, Human-computer interaction, Mobile robotics, Object identification and others.

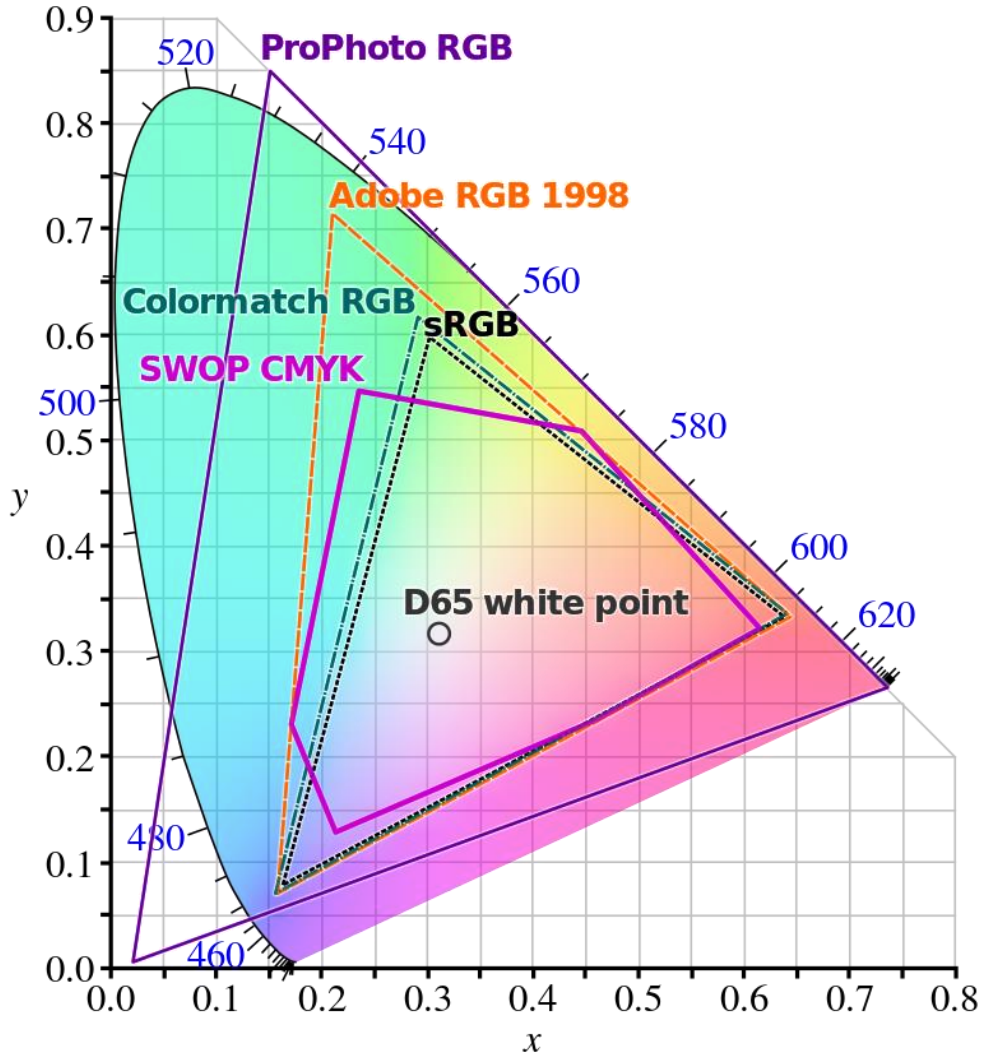
- **Numpy and Scipy libraries** — For image manipuation and processing.
- **Sckikit** — Provides lots of alogrithms for image processing.
- **Python Imaging Library (PIL)** — To perform basic operations on images like create thumnails, resize, rotation, convert between different file formats etc.

```
sudo apt-get install python-pip python3-pip python-numpy
pip install pillow    or    pip3 install pillow
sudo apt-get install python-opencv or pip3 install opencv-python
```

Image Color System

A **color space** is a specific organization of colors. In combination with physical device profiling, it allows for reproducible representations of color, in both analog and digital representations.





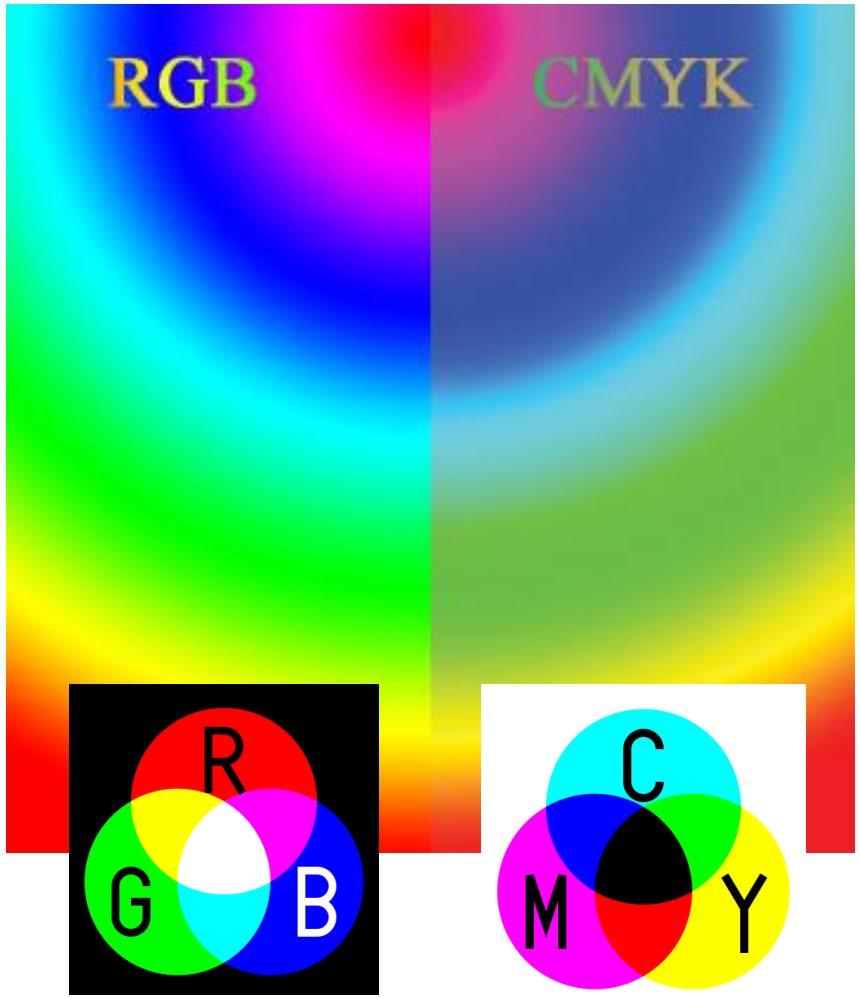
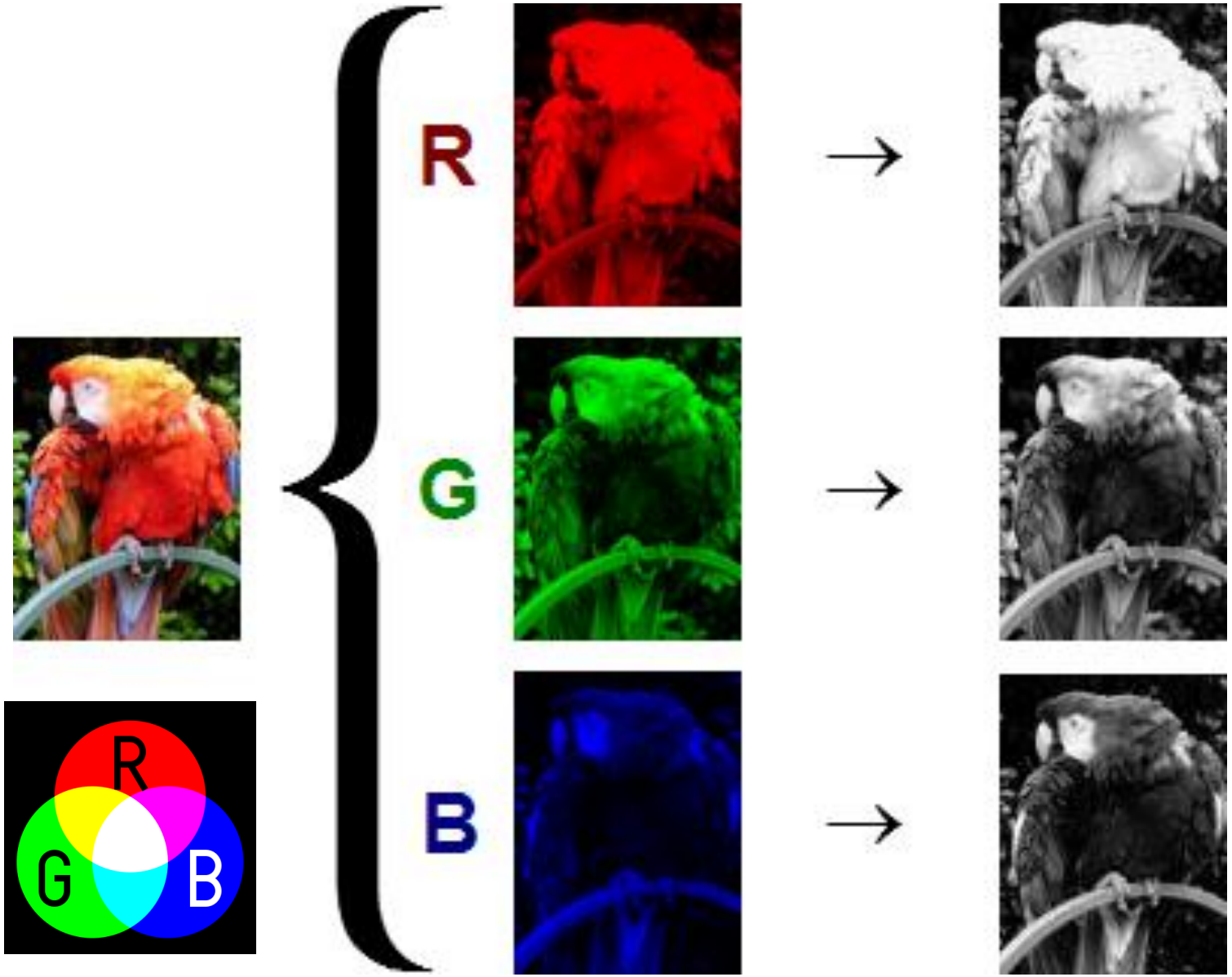


Image Color System



RGB color format & calculation

RGB code has 24 bits format (bits 0..23):

RED[7:0]								GREEN[7:0]								BLUE[7:0]							
23							16	15							8	7							0

```
RGB = (R*65536)+(G*256)+B , (when R is RED, G is GREEN and B is BLUE)
```

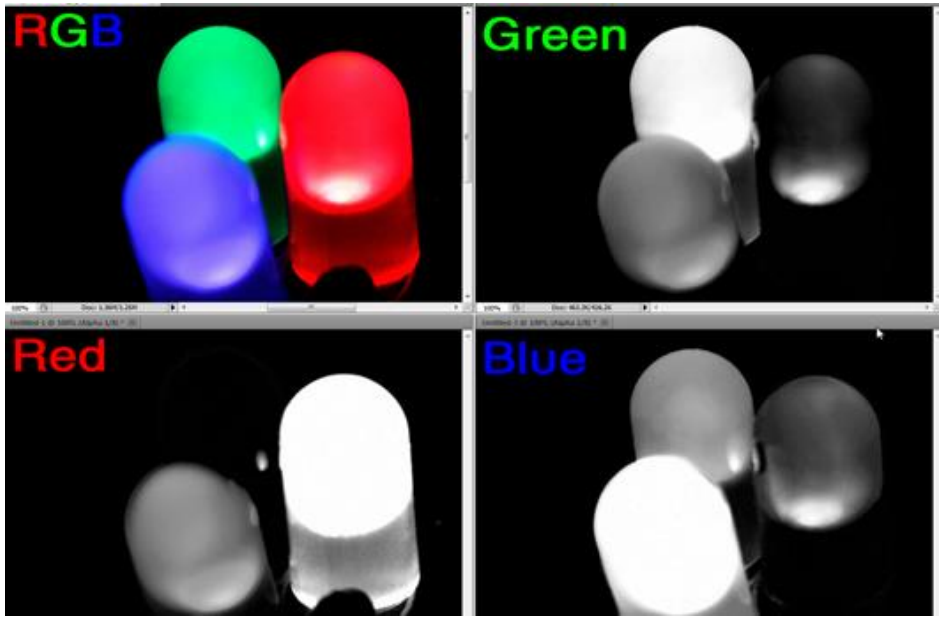
Calculation examples

White RGB Color

```
White RGB code = 255*65536+255*256+255 = #FFFFFF
```

Blue RGB Color

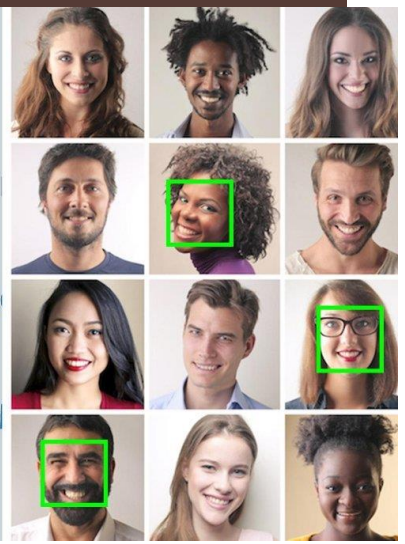
```
Blue RGB code = 0*65536+0*256+255 = #0000FF
```



What is OpenCV ?

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. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

```
img = cv2.imread("faces.j
gray = cv2.cvtColor(img,
path = "haarcascade_eye.x

eye_cascade = cv2.Cascade

eyes = eye_cascade.detect
print(len(eyes))
```



OpenCV Basic:

NumPy

NumPy.org

NumPy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

```
>>> A = np.array( [[1,1],
...               [0,1]] )
>>> B = np.array( [[2,0],
...               [3,4]] )
>>> A * B                          # elementwise product
array([[2, 0],
       [0, 4]])
>>> A @ B                          # matrix product
array([[5, 4],
       [3, 4]])
>>> A.dot(B)                       # another matrix product
array([[5, 4],
       [3, 4]])
```

OpenCV Basic:

opencv-python 4.1.1.26

Latest version

```
pip install opencv-python
```

Last released: Sep 2, 2019

Wrapper package for OpenCV python bindings.

OpenCV use BRG rather than RGB

Navigation

- Project description
- Release history
- Download files

Project description

downloads 47M

OpenCV on Wheels

Unofficial pre-built OpenCV packages for Python.

opencv-contrib-python==4.1.2.30

opencv-python==4.1.1.26

```
student@student-VirtualBox:~$ python
Python 2.7.12 (default, Aug 22 2019, 16:36:40)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'2.4.9.1'
>>> exit()
student@student-VirtualBox:~$ python3
Python 3.5.2 (default, Jul 10 2019, 11:58:48)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'4.1.1'
```

Cannot use pip to install opencv-python for python 2.7 due to error of numpy package

If need newer version of opencv in python 2.7 need to build from source process

OpenCV Basic:

cv2.VideoCapture(0): Means first camera or webcam.

cv2.VideoCapture(1): Means second camera or webcam.

cv2.VideoCapture("file name.mp4"): Means video file

cap = cv2.VideoCapture(0) --> ret, img =cap.read()

cv2.imread(image path)

cv2.imwrite(filename, img)

cv2.imshow("window name", image)

cv2.waitkey(0)

cv2.destroyAllWindows()

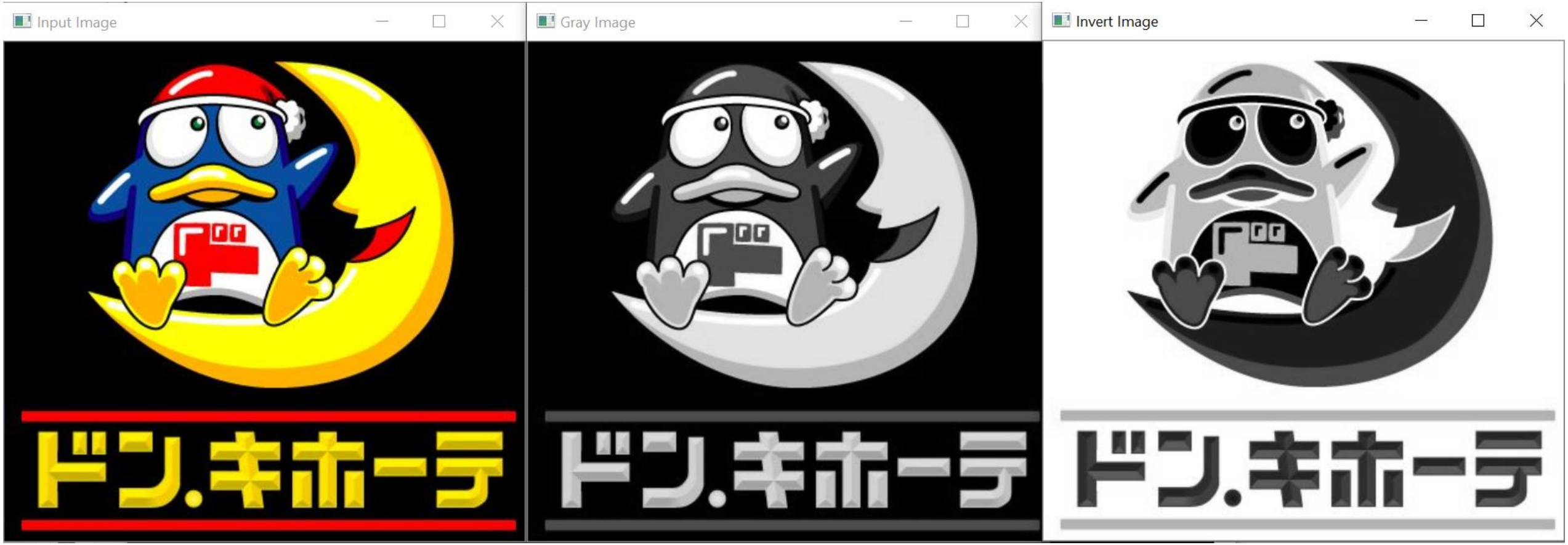
https://www.geeksforgeeks.org/opencv-python-tutorial/

OpenCV Basic:

```python
import cv2
import argparse
#Usage python main_img.py --file Scenic009smaller.bmp #
img_location = 'DonKi.jpg'
parser = argparse.ArgumentParser(description='input image name')
parser.add_argument('--file', help="location of the image file")
args = parser.parse_args()
if args.file:
    img_location = args.file
print('<Image File Location>')
print(img_location)
img = cv2.imread(img_location)
print('File loaded')
h,w,c = img.shape
print('----------------------------------------------------')
print("Image Information")
print("Dimension w x h = %s x %s" % (w,h))
print("Channels = %s" %(c))
print("Total Pixels of input image = %s pixels which %s channels" % (w*h,c))
print('----------------------------------------------------')
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
Invert_img = 255-gray_img
cv2.imshow('Input Image',img)
cv2.imshow('Gray Image',gray_img)
cv2.imshow('Invert Image',Invert_img)
cv2.imwrite('Invert_img.bmp',Invert_img)
print('File Saved')
cv2.waitKey(0)
```

```
<Image File Location>
DonKi.jpg
File loaded
----------------------------------------------------
Image Information
Dimension w x h = 421 x 404
Channels = 3
Total Pixels of input image = 170084 pixels which 3 channels
----------------------------------------------------
File Saved
```

OpenCV Basic:



OpenCV Basic:

```
import cv2
import argparse
#Usage python main_img.py --file Scenic009smaller.bmp #
img_location = 'DonKi.jpg'
parser = argparse.ArgumentParser(description='input image name')
parser.add_argument('--file', help="location of the image file")
args = parser.parse_args()
if args.file:
    img_location = args.file
print('<Image File Location>')
print(img_location)
img = cv2.imread(img_location)
print('File loaded')
h,w,c = img.shape
print('-----------------------------------------------')
print("Image Information")
print("Dimension w x h = %s x %s" % (w,h))
print("Channels = %s" %(c))
print("Total Pixels of input image = %s pixels which %s channels" % (w*h,c))
print('-----------------------------------------------')
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
Invert_img = 255-gray_img
cv2.imshow('Input Image',img)
cv2.imshow('Gray Image',gray_img)
cv2.imshow('Invert Image',Invert_img)
cv2.imwrite('Invert_img.bmp',Invert_img)
print('File Saved')
cv2.waitKey(0)
```

Pass file name

Import image file (BGR) to image (numpy) array

Get image information (height, Width, Channel)

Change BGR image to Gray Scale image

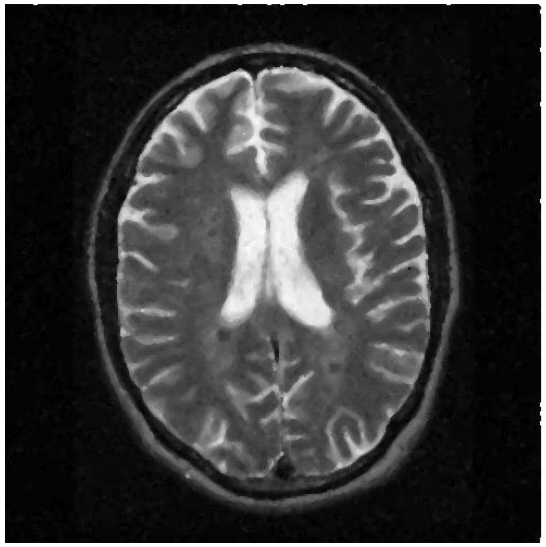
Invert Gray Scale image

Display image

Save image array to file

Wait for user press any key

Image Filter (2D convolution)



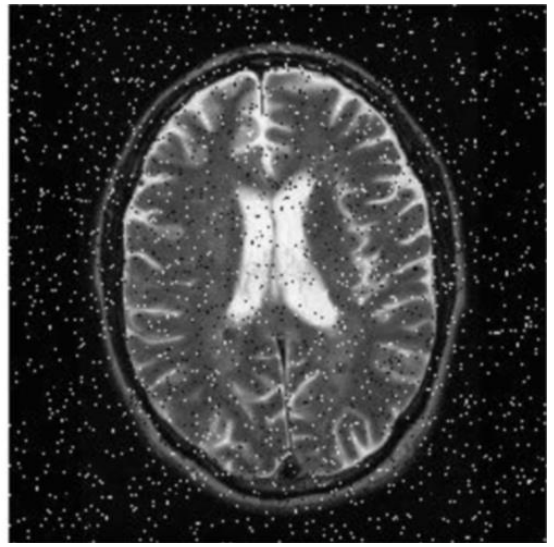
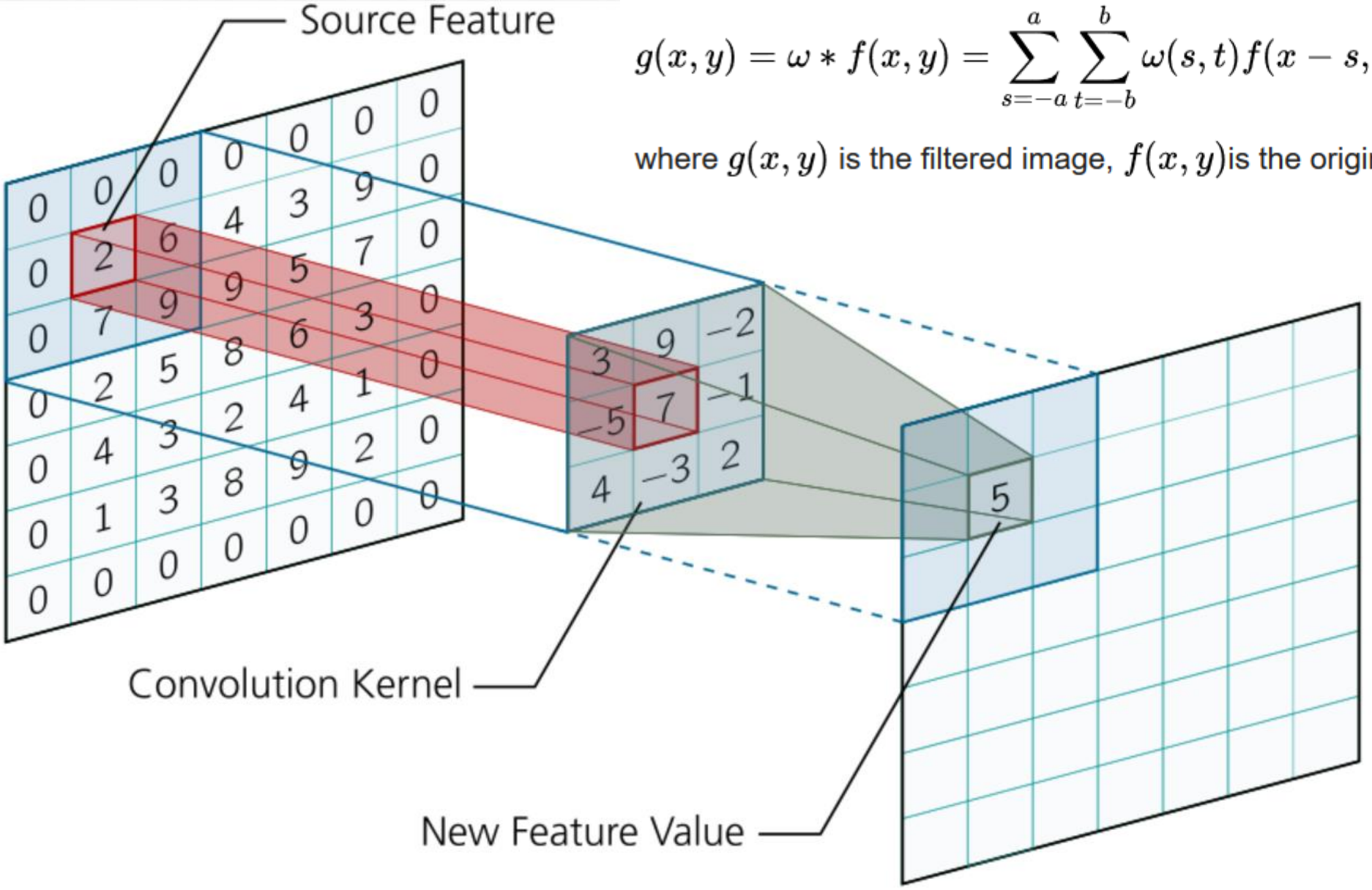


Image Filter (2D convolution)



$$g(x,y) = \omega * f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s,t) f(x-s, y-t),$$

where $g(x,y)$ is the filtered image, $f(x,y)$is the original image, ω is the filter kernel.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

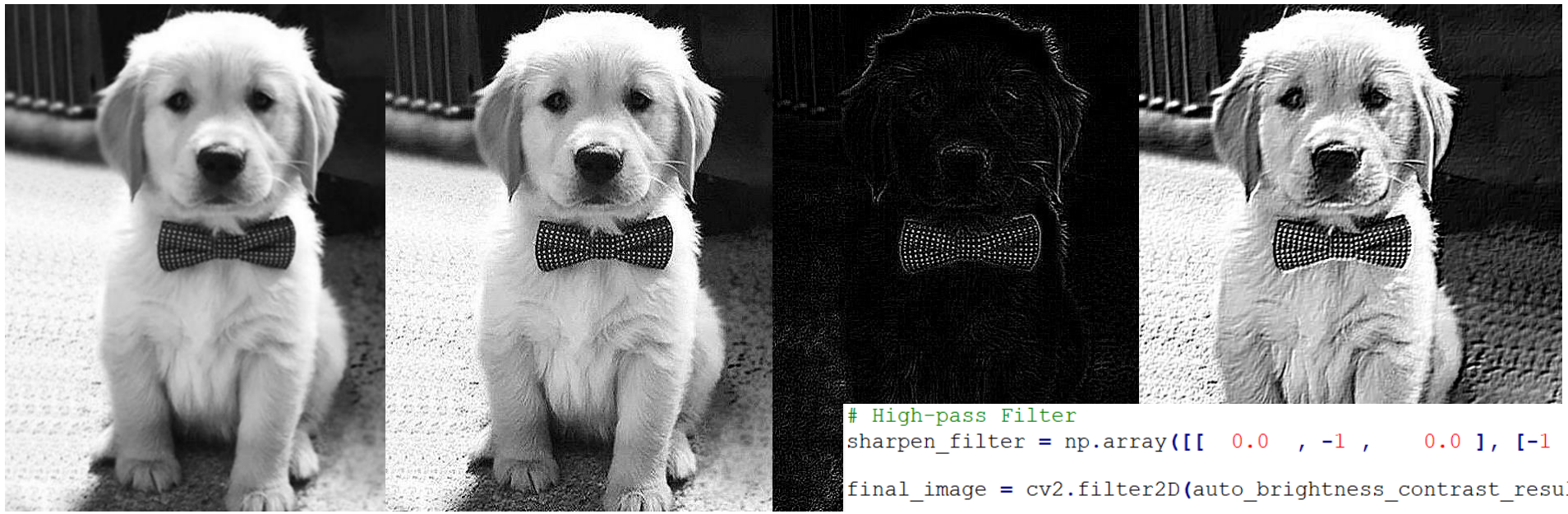
$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Image Filter (2D convolution)




cv2.filter2D( )

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \quad \begin{bmatrix}0 & -0.5 & 0\\ -0.5 & 3 & -0.5\\ 0 & -0.5 & 0\end{bmatrix} \quad \begin{bmatrix}-1 & -1 & -1\\ -1 & 8 & -1\\ -1 & -1 & -1\end{bmatrix} \quad \begin{bmatrix}-2 & -1 & 0\\ -1 & 0 & 1\\ 0 & 1 & 2\end{bmatrix}$$

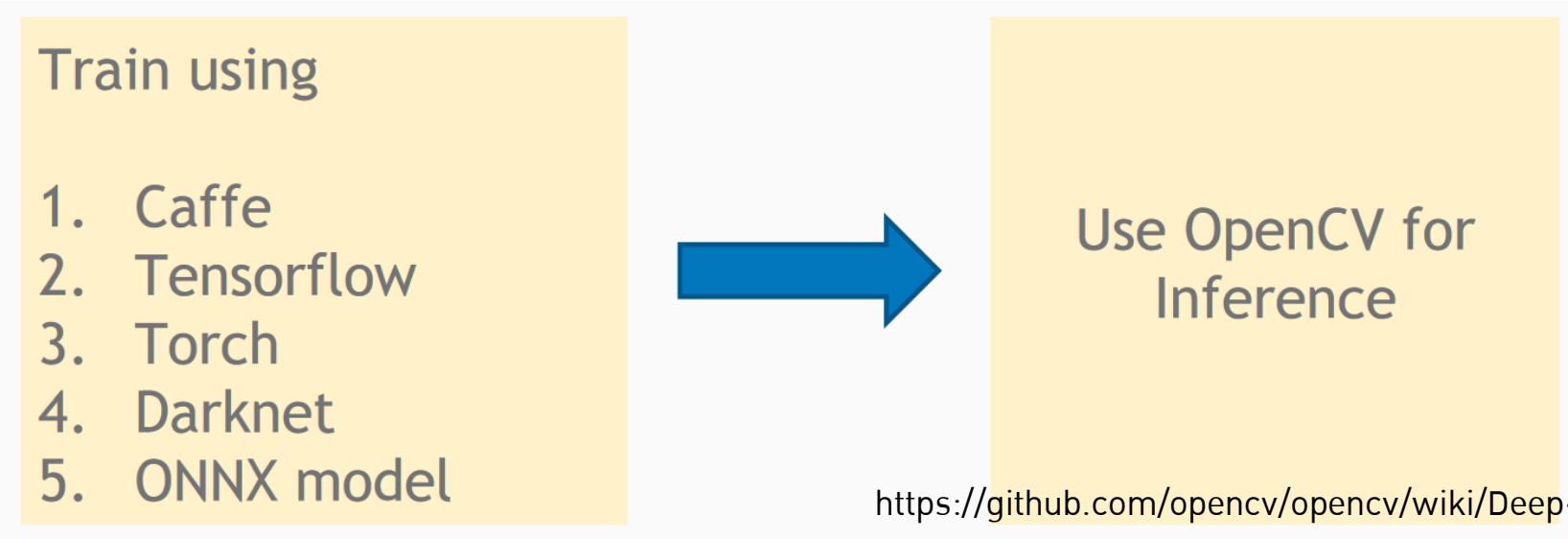


```
# High-pass Filter
sharpen_filter = np.array([[  0.0   , -1 ,    0.0 ], [-1 ,   5   , -1 ],[  0.0   , -1 ,    0.0 ]])

final_image = cv2.filter2D(auto_brightness_contrast_result, -1, sharpen_filter)
```

Cv2.dnn (Deep Neural Networks (dnn module))

Deep Neural Networks Module in OpenCV

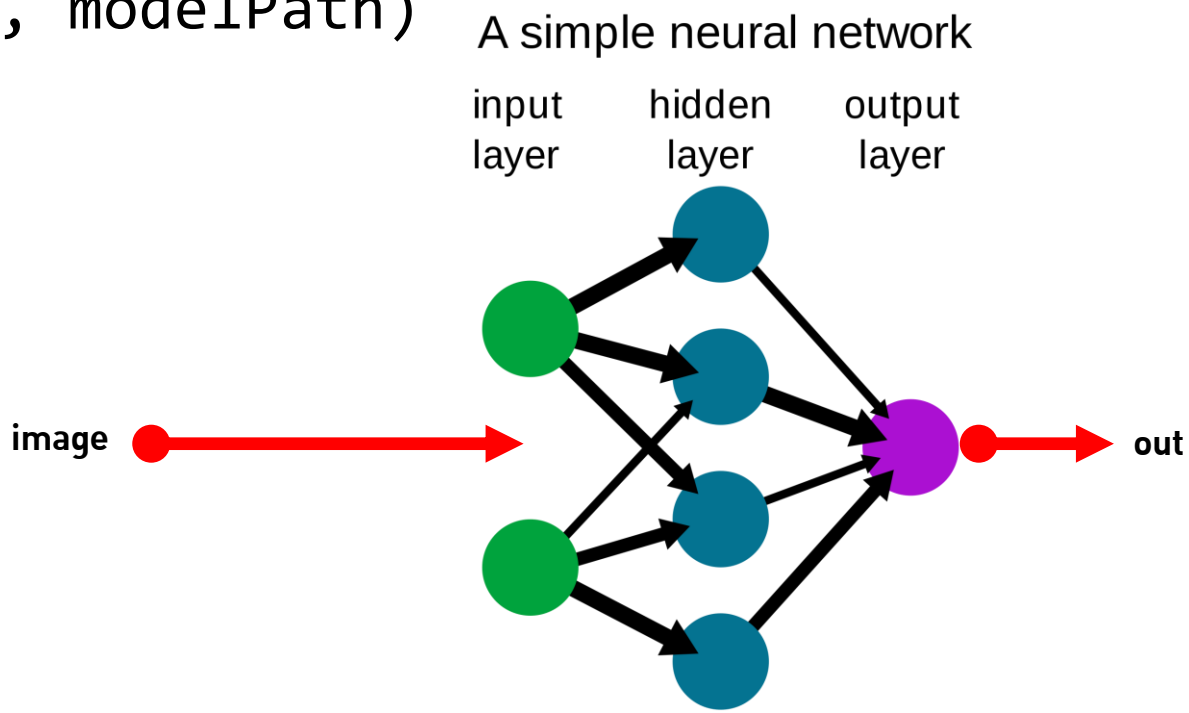
Deep Learning is the most popular and the fastest growing area in Computer Vision nowadays. Since OpenCV 3.1 there is DNN module in the library that implements forward pass (inferencing) with deep networks, pre-trained using some popular deep learning frameworks, such as Caffe. In OpenCV 3.3 the module has been promoted from opencv contrib repository to the main repository (https://github.com/opencv/opencv/tree/master/modules/dnn) and has been accelerated significantly.



https://github.com/opencv/opencv/wiki/Deep-Learning-in-OpenCV

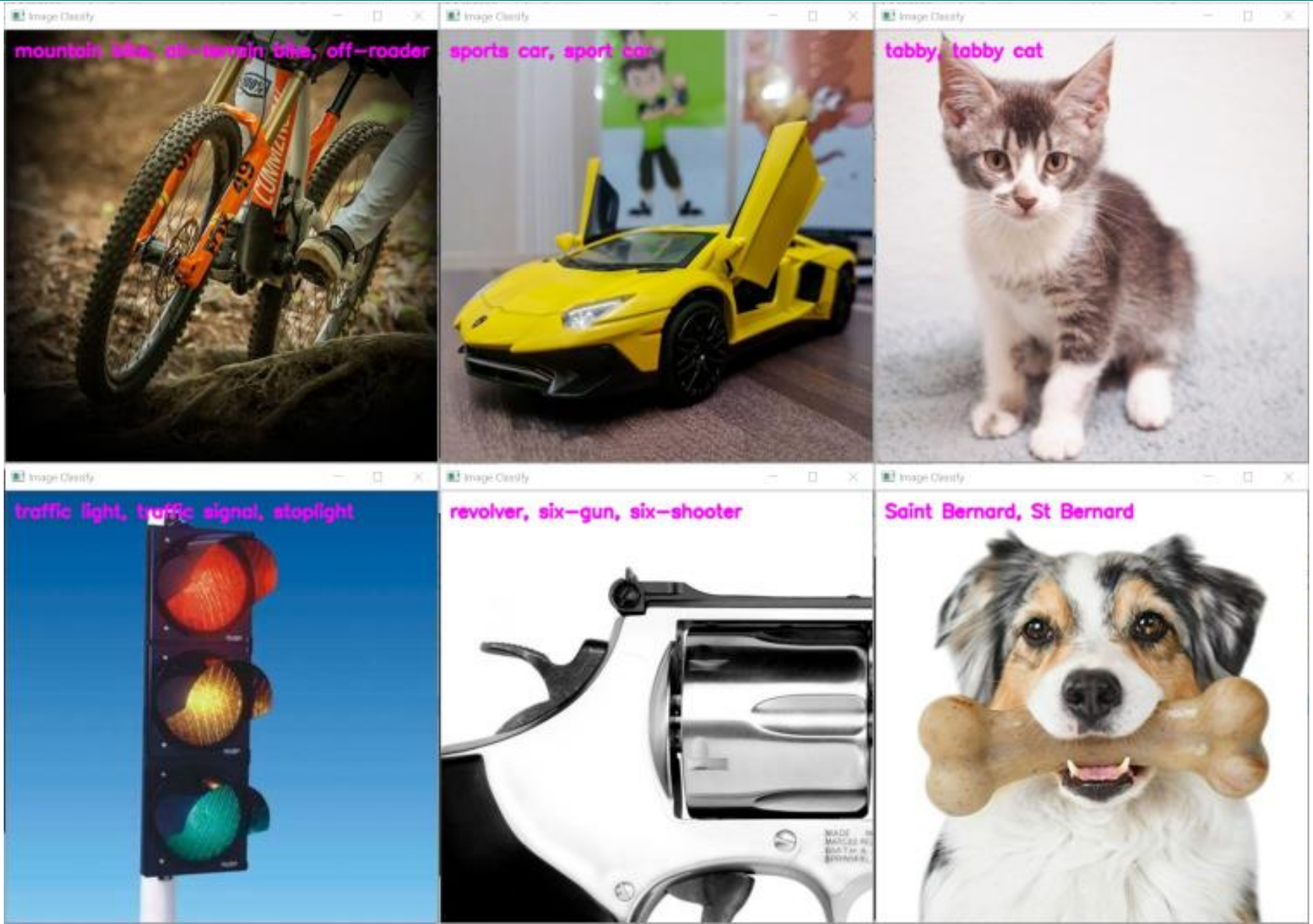
Cv2.dnn for object detection

```
protoPath = "./face_detection_model/deploy.prototxt"
modelPath =
"./face_detection_model/res10_300x300_ssd_iter_140000.caffemodel"

detector = cv2.dnn.readNetFromCaffe(protoPath, modelPath)
detector = cv2.dnn.readNet(protoPath, modelPath)

imageBlob = cv2.dnn.blobFromImage( )
detector.setInput(imageBlob)
detections = detector.forward()
```

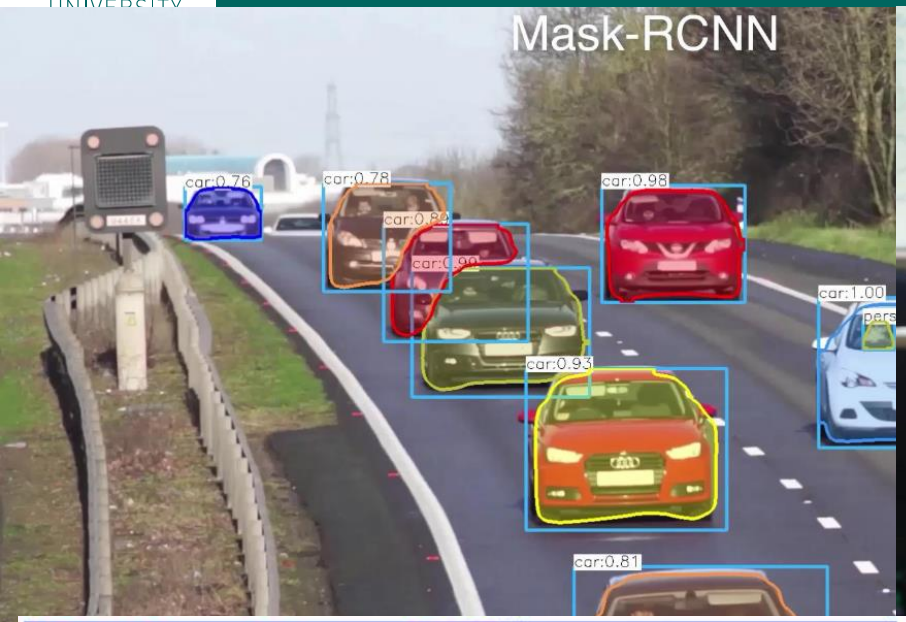


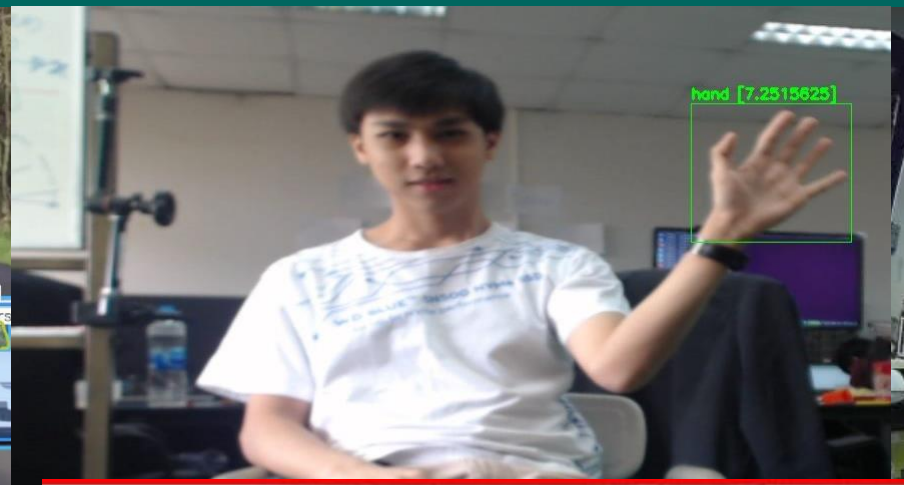
https://github.com/opencv/opencv/wiki/Deep-Learning-in-OpenCV

Cv2.dnn

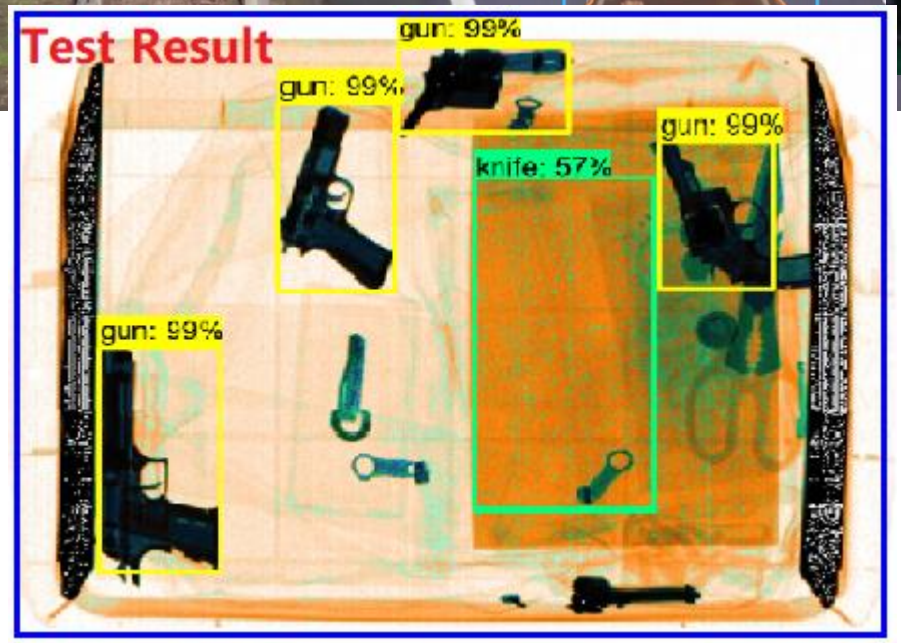

mountain bike, all-terrain bike, off-roader
sports car, sport car
tabby, tabby cat
traffic light, traffic signal, stoplight
revolver, six-gun, six-shooter
Saint Bernard, St Bernard

Cv2.dnn


Mask-RCNN


hand [7.2515625]


person 0.90533316
person 0.91750956
drinking
reading


Test Result
gun: 99%
gun: 99%
gun: 99%
knife: 57%
gun: 99%


Male, (17–25), Happy
bottle


fire

KASETSART
UNIVERSITY

OpenCV and ROS

OpenCV and ROS

```
ros@ros-VirtualBox: ~
ros@ros-VirtualBox:~$ python
Python 2.7.12 (default, Apr 15 2020, 17:07:12)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'3.3.1-dev'
>>>
```

Problem with ROS **When use dnn**

```
OpenCV Error: Assertion failed (input.dims == 4 && (input.type() == 5 || input.type() == 6)) in finalize, file /tmp/binary
deb/ros-kinetic-opencv3-3.3.1/modules/dnn/src/layers/convolution_layer.cpp, line 78
[ERROR] [1590997247.972806]: bad callback: <function callback_facedetection at 0x7fa15f011848>
Traceback (most recent call last):
  File "/opt/ros/kinetic/lib/python2.7/dist-packages/rospy/topics.py", line 750, in _invoke_callback
    cb(msg)
  File "dectect_face_ros.py", line 33, in callback_facedetection
    detections = detector.forward()
error: /tmp/binarydeb/ros-kinetic-opencv3-3.3.1/modules/dnn/src/layers/convolution_layer.cpp:78: error: (-215) input.dims
== 4 && (input.type() == 5 || input.type() == 6) in function finalize
```

Problem cause by CV2.SO (binary kernel object)

Cv2.dnn

ROS-kinetic CV2.so location

OpenCV3.3.1- dev(path : /opt/ros/kinetic/lib/python2.7/dist-packages/cv2.so)

$cd

$cd /opt/ros/kinetic/lib/python2.7/dist-packages

$ls cv*

$sudo apt-get install python-pip

```
ros@ros-VirtualBox:/opt/ros/kinetic/lib/python2.7/dist-packages$ ls cv*
cv2.so  cv_bridge-1.12.8.egg-info

cv_bridge:
boost  core.py  core.pyc  __init__.py  __init__.pyc
```

$sudo mv cv2.so cv4.so

```
ros@ros-VirtualBox:/opt/ros/kinetic/lib/python2.7/dist-packages$ sudo mv cv2.so cv4.so
ros@ros-VirtualBox:/opt/ros/kinetic/lib/python2.7/dist-packages$ ls cv2*
ls: cannot access 'cv2*': No such file or directory
```

Cv2.dnn

```
ros@ros-VirtualBox:/opt/ros/kinetic/lib/python2.7/dist-packages$ python
Python 2.7.12 (default, Apr 15 2020, 17:07:12)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named cv2
>>> exit()
ros@ros-VirtualBox:/opt/ros/kinetic/lib/python2.7/dist-packages$
```

$sudo apt-get install python-pip

$pip install opencv-contrib-python==3.4.2.16 opencv-python==3.4.2.16

Cv2.dnn

$pip install opencv-contrib-python==3.4.2.16 opencv-python==3.4.2.16

```
ros@ros-VirtualBox:/opt/ros/kinetic/lib/python2.7/dist-packages$ pip install opencv-contrib-python==3.4.2.16
opencv-python==3.4.2.16
Collecting opencv-contrib-python==3.4.2.16
  Downloading https://files.pythonhosted.org/packages/43/1d/e5e7c01fba5ae64abbf76cb3d38ffb3958c38b46ec6292166
e549dde75a1/opencv_contrib_python-3.4.2.16-cp27-cp27mu-manylinux1_x86_64.whl (30.6MB)
    100% |                                | 30.6MB 40kB/s
Collecting opencv-python==3.4.2.16
  Downloading https://files.pythonhosted.org/packages/b0/37/5baf002774374f0694330cc4d11cbd8af38890e928246ee58
877111a70e3/opencv_python-3.4.2.16-cp27-cp27mu-manylinux1_x86_64.whl (25.0MB)
    100% |                                | 25.0MB 42kB/s
Collecting numpy>=1.11.1 (from opencv-contrib-python==3.4.2.16)
  Downloading https://files.pythonhosted.org/packages/2d/f3/795e50e3ea2dc7bc9d1a2eeea9997d5dce63b801e08dfc37c
2efce341977/numpy-1.18.4.zip (5.4MB)
    100% |                                | 5.4MB 178kB/s
    Complete output from command python setup.py egg_info:
    Traceback (most recent call last):
      File "<string>", line 1, in <module>
      File "/tmp/pip-build-TrjkuE/numpy/setup.py", line 32, in <module>
        raise RuntimeError("Python version >= 3.5 required.")
    RuntimeError: Python version >= 3.5 required.

    ----------------------------------------
Command "python setup.py egg_info" failed with error code 1 in /tmp/pip-build-TrjkuE/numpy/
You are using pip version 8.1.1, however version 20.1.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
ros@ros-VirtualBox:/opt/ros/kinetic/lib/python2.7/dist-packages$
```

What ! Python 2.7 need 3.5

Cv2.dnn

$pip install --upgrade pip

$pip install --upgrade buildtools

```
ros@ros-VirtualBox:/opt/ros/kinetic/lib/python2.7/dist-packages$ pip install opencv-contrib-python==3.4.2.16
opencv-python==3.4.2.16
WARNING: pip is being invoked by an old script wrapper. This will fail in a future version of pip.
Please see https://github.com/pypa/pip/issues/5599 for advice on fixing the underlying issue.
To avoid this problem you can invoke Python with '-m pip' instead of running pip directly.
/home/ros/.local/lib/python2.7/site-packages/pip/_vendor/requests/__init__.py:83: RequestsDependencyWarning:
Old version of cryptography ([1, 2, 3]) may cause slowdown.
  warnings.warn(warning, RequestsDependencyWarning)
DEPRECATION: Python 2.7 reached the end of its life on January 1st, 2020. Please upgrade your Python as Pytho
n 2.7 is no longer maintained. pip 21.0 will drop support for Python 2.7 in January 2021. More details about
Python 2 support in pip, can be found at https://pip.pypa.io/en/latest/development/release-process/#python-2-
support
Defaulting to user installation because normal site-packages is not writeable
Collecting opencv-contrib-python==3.4.2.16
  Using cached opencv_contrib_python-3.4.2.16-cp27-cp27mu-manylinux1_x86_64.whl (30.6 MB)
Collecting opencv-python==3.4.2.16
  Using cached opencv_python-3.4.2.16-cp27-cp27mu-manylinux1_x86_64.whl (25.0 MB)
Collecting numpy>=1.11.1
  Downloading numpy-1.16.6-cp27-cp27mu-manylinux1_x86_64.whl (17.0 MB)
     |                                | 17.0 MB 10.7 MB/s
Installing collected packages: numpy, opencv-contrib-python, opencv-python
Successfully installed numpy-1.16.6 opencv-contrib-python-3.4.2.16 opencv-python-3.4.2.16
ros@ros-VirtualBox:/opt/ros/kinetic/lib/python2.7/dist-packages$
```

Cv2.dnn

```
ros@ros-VirtualBox:/opt/ros/kinetic/lib/python2.7/dist-packages$ python
Python 2.7.12 (default, Apr 15 2020, 17:07:12)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'3.4.2'
>>>
```

READY for cv2.dnn

```
$tar –xzvf ROS_OpenCV.tar
```

```
ros@ros-VirtualBox:~/ROS_OpenCV$ ls
dectect_face_ros.py   object_detection_ros.py          vokoscreen-2020-05-19_14-24-16.mkv
face_detection_model  publish_video.py
imutils               vokoscreen-2020-05-19_12-57-55.mkv
```

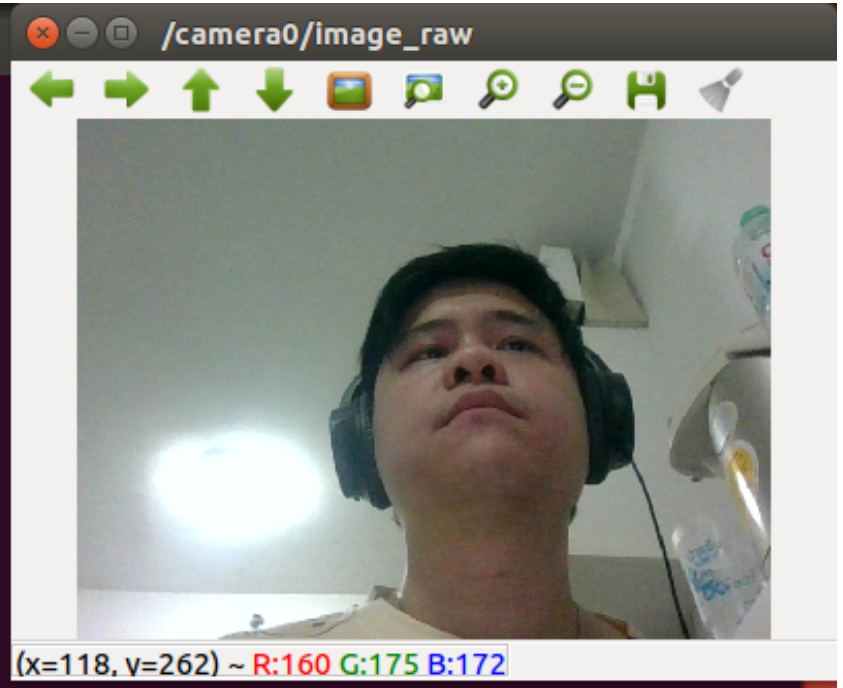
```
$roscore
```

```
$python publish_video.py
```

```
$rosrun image_view image_view image:=/camera0/image_raw
```

Cv2.dnn

```
ros@ros-VirtualBox: ~/ROS_OpenCV
[INFO] [1591107167.665112]: Capture image at 1591107167.66
[INFO] [1591107167.865674]: Capture image at 1591107167.87
[INFO] [1591107168.065611]: Capture image at 1591107168.07
[INFO] [1591107168.265022]: Capture image at 1591107168.26
[INFO] [1591107168.466501]: Capture image at 1591107168.47
[INFO] [1591107168.666270]: Capture image at 1591107168.67
[INFO] [1591107168.865820]: Capture image at 1591107168.87
[INFO] [1591107169.066785]: Capture image at 1591107169.07
[INFO] [1591107169.265991]: Capture image at 1591107169.27
[INFO] [1591107169.466013]: Capture image at 1591107169.47
[INFO] [1591107169.665425]: Capture image at 1591107169.67
[INFO] [1591107169.865194]: Capture image at 1591107169.87
[INFO] [1591107170.065215]: Capture image at 1591107170.07
[INFO] [1591107170.265516]: Capture image at 1591107170.27
[INFO] [1591107170.465677]: Capture image at 1591107170.47
[INFO] [1591107170.665032]: Capture image at 1591107170.66
[INFO] [1591107170.865927]: Capture
[INFO] [1591107171.070089]: Capture
[INFO] [1591107171.265679]: Capture
[INFO] [1591107171.465585]: Capture
[INFO] [1591107171.665326]: Capture
[INFO] [1591107171.865361]: Capture
```




```
roscore http://ros-VirtualBox:11311/
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://ros-VirtualBox:46661/
ros_comm version 1.12.14
```

```
ros@ros-VirtualBox: ~
ros@ros-VirtualBox:~$ rosrun image_view image_view image:=/camera0/image_raw
[ INFO] [1591107147.375128180]: Using transport "raw"
```

Image Captured by opencv

Publish to ROS by cvBridge

Display by package image view

Cv2.dnn

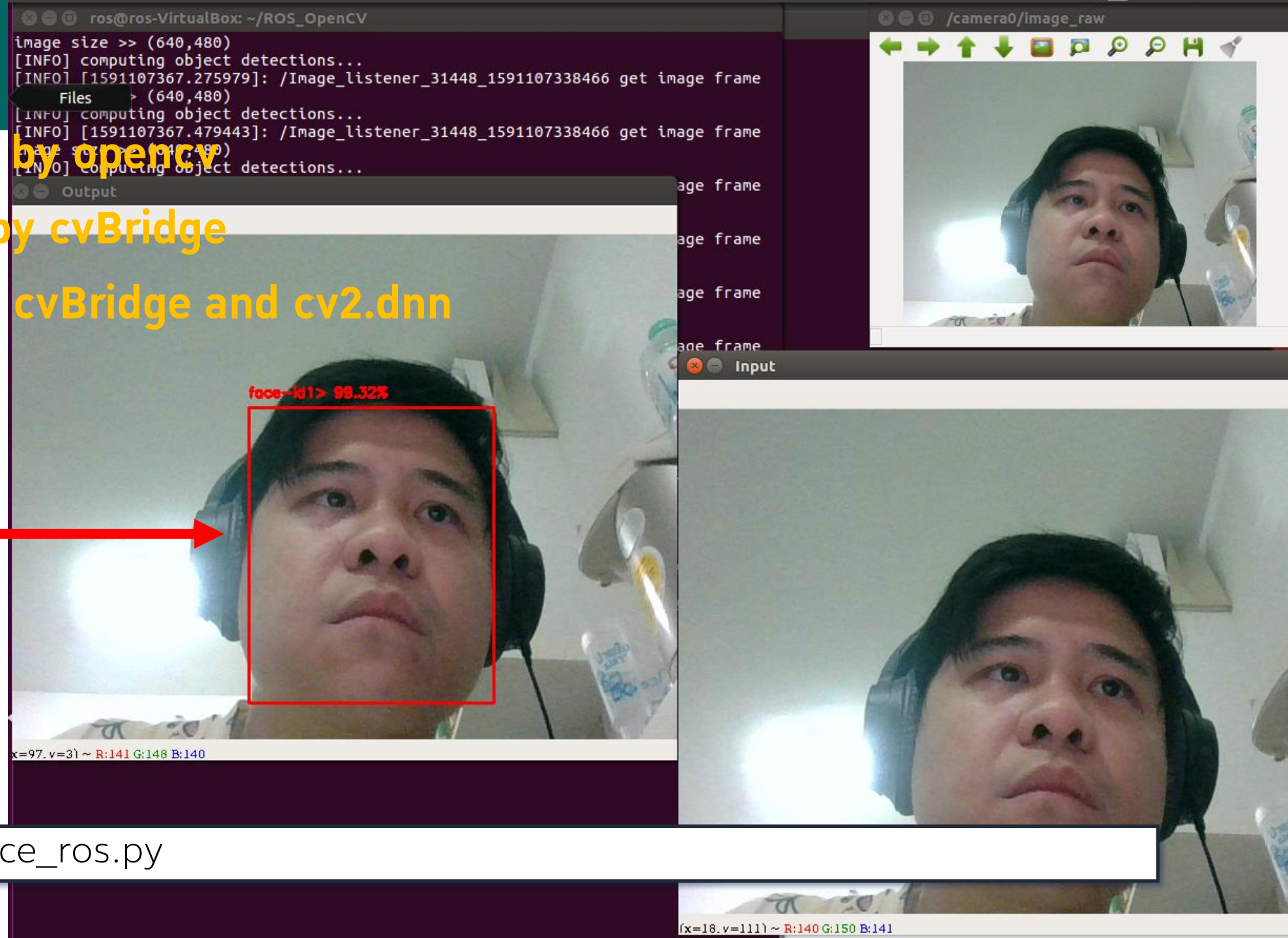
Image Captured by opencv

Publish to ROS by cvBridge

Convert back by cvBridge and cv2.dnn

Face detection !

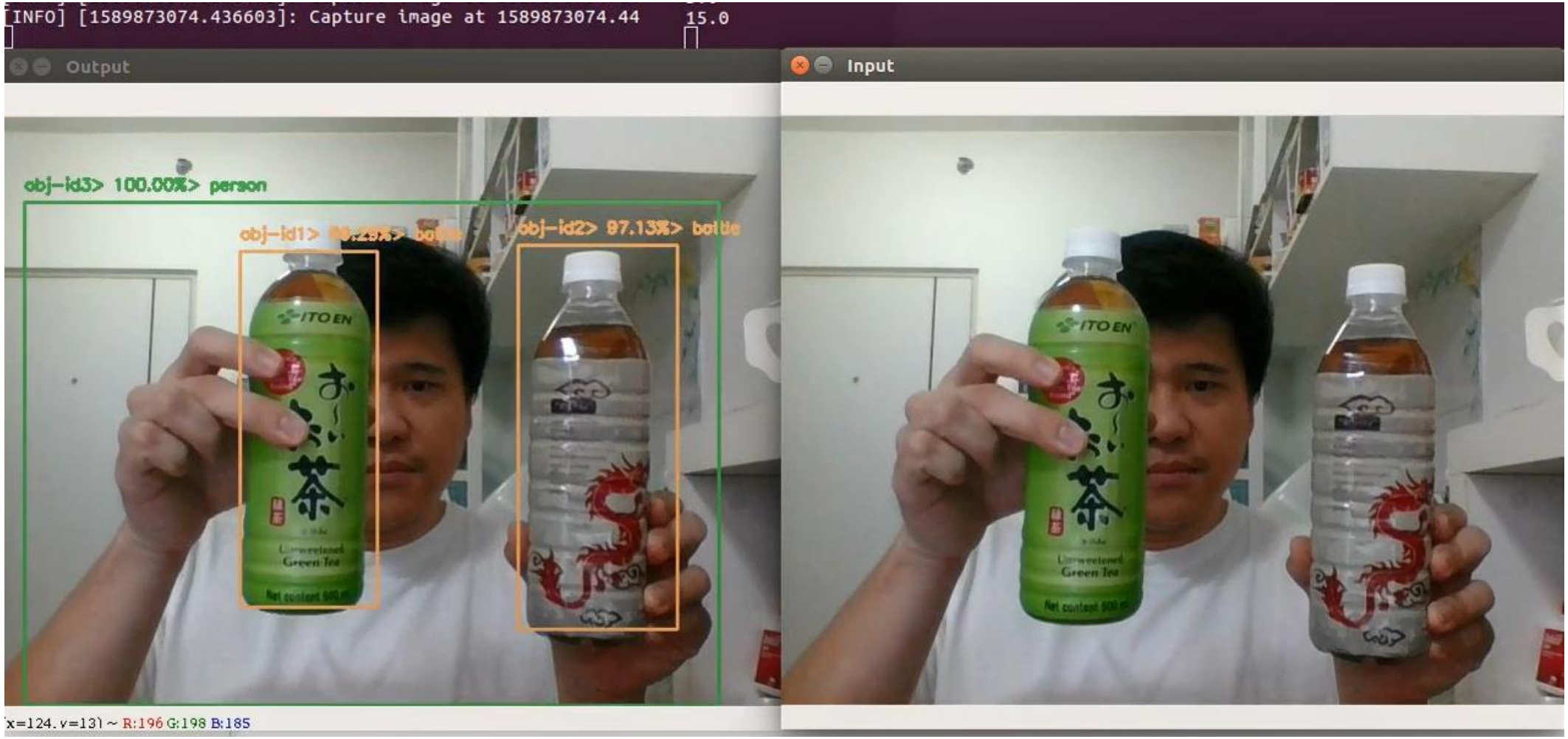
By dnn



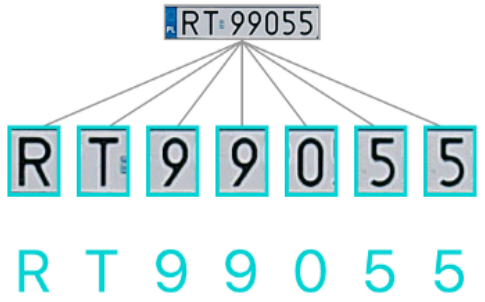

```
$python dectect_face_ros.py
```

Cv2.dnn

$python object_detection_ros.py



STOP



Car detection → License plate detection → Symbol detection

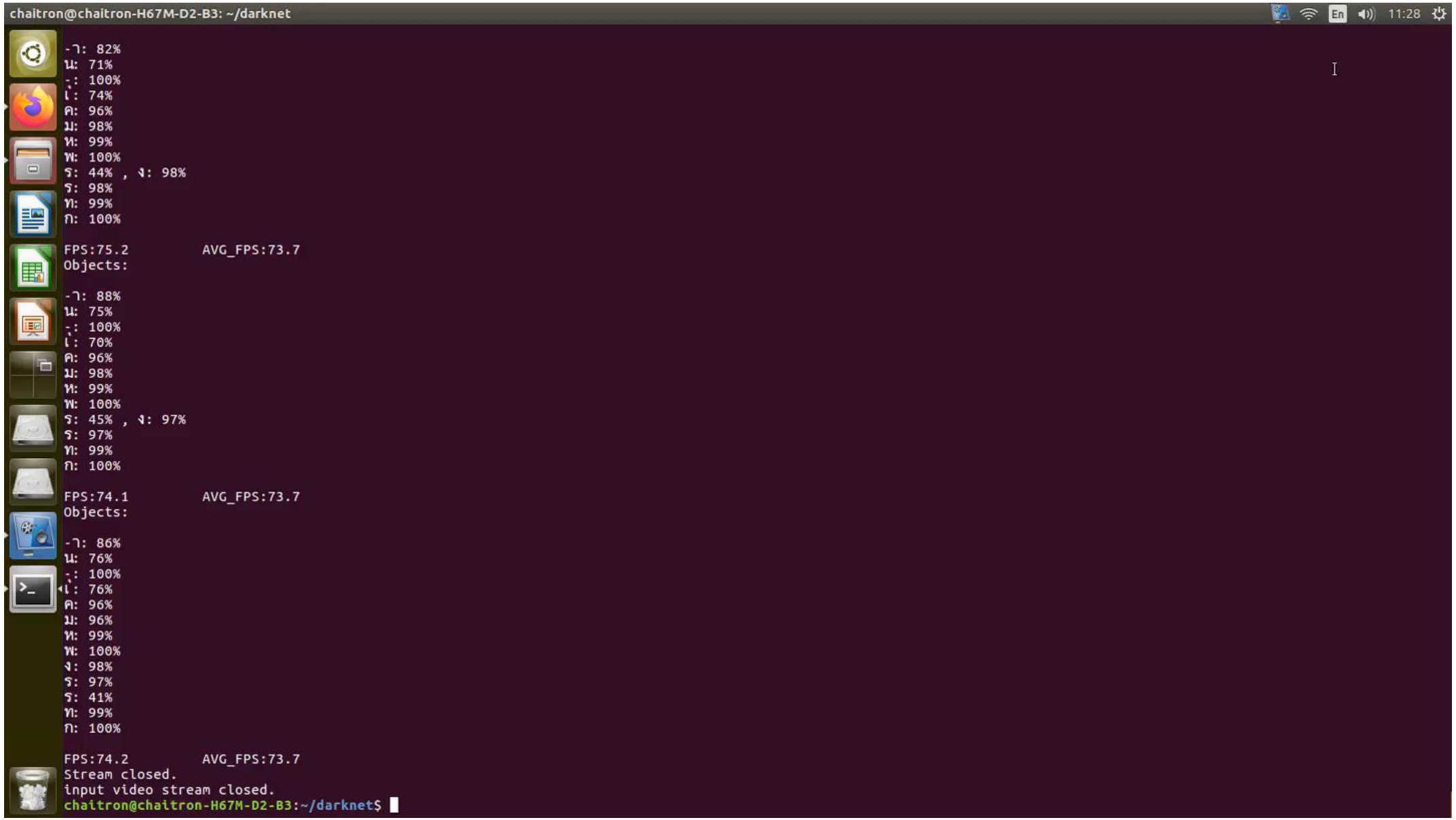
Cv2.dnn

EAST (cv2.dnn) + Tesseract thai

↓

Character detect + OCR thai

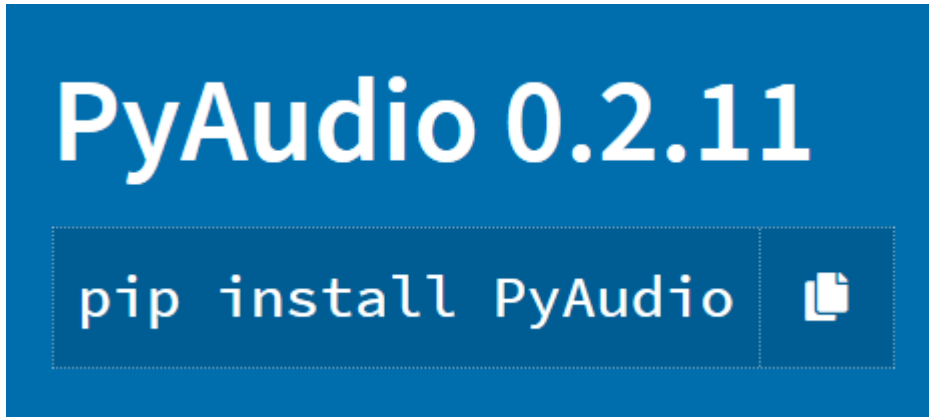
If deep learning is trained, Cv2.dnn can implemented



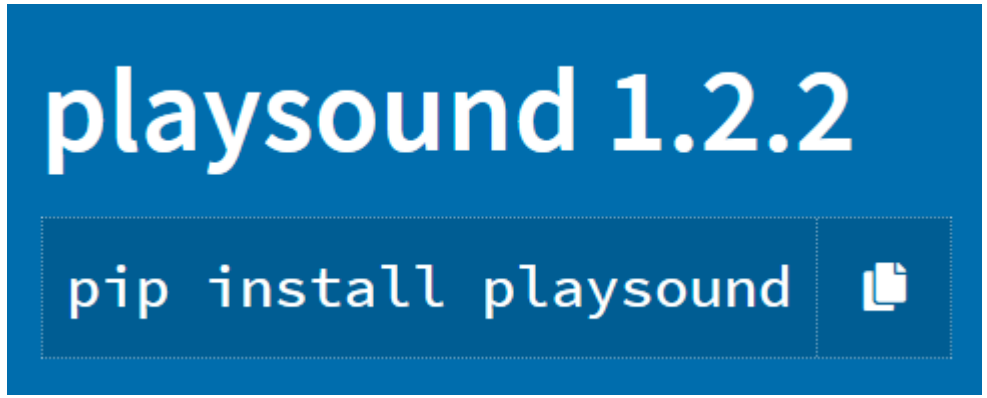
KASETSART
UNIVERSITY

We need to Speak

Speak



PyAudio provides Python bindings for PortAudio, the cross-platform audio I/O library. With PyAudio, you can easily use Python to play and record audio on a variety of platforms, such as GNU/Linux, Microsoft Windows, and Apple Mac OS X / macOS.



The playsound module contains only one thing - the function (also named) playsound.
It requires one argument - the path to the file with the sound youd like to play. This may be a local file, or a URL.

Speak

Library	Platform	Playback	Record	Convert	Dependencies
`playsound`	Cross-platform	WAV, MP3	-	-	None
`simpleaudio`	Cross-	WAV, array,	-	-	None

$pip install playsound

`winsound`	Windows	WAV	-	-	None
`sounddevice`	Cross-platform	NumPy array	NumPy array	-	`numpy`, `soundfile`

$sudo apt-get install ffmpeg libav-tools
$sudo apt-get install libasound-dev portaudio19-dev libportaudio2 libportaudiocpp0
$pip install pyAudio

`pyaudio`	Cross-platform	`bytes`	`bytes`	-	`wave`
`wavio`	Cross-platform	-	-	WAV, NumPy array	`numpy`, `wave`

Speak

Ispeak.py

```
from playsound import playsound
import pyaudio
import wave
print(' Speaking...')
playsound("./unknown.mp3")
playsound("./covid.mp3")
playsound('./nictomeet.mp3')
print(' Start to ask for speak')
playsound("./speak_after.mp3")
playsound("./signal.mp3")
```

Speak

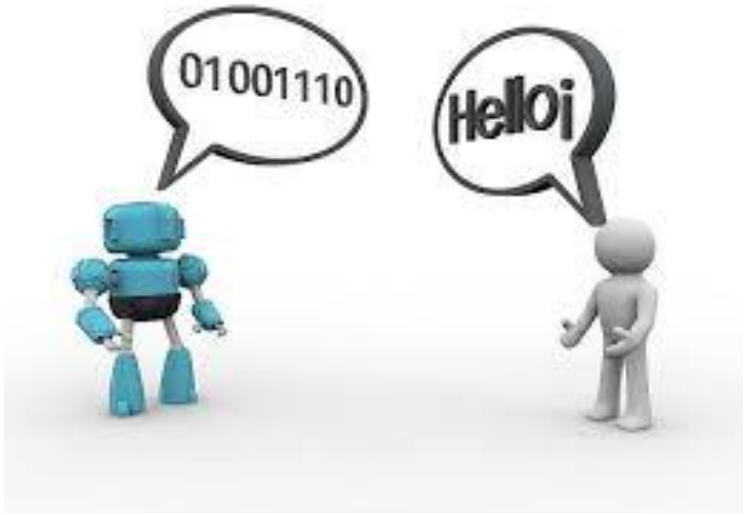
```
chunk = 1024        # Record in chunks of 1024 samples
sample_format = pyaudio.paInt16  # 16 bits per sample
channels = 2
fs = 44100          # Record at 44100 samples per second
seconds = 3
filename = "sound_out.wav“
print('Recording ……’)

p = pyaudio.PyAudio()  # Create an interface to PortAudio
stream = p.open(format=sample_format, channels=channels, rate=fs,
frames_per_buffer=chunk, input=True)

frames = []  # Initialize array to store frames
```

Speak

```
for i in range(0, int(fs / chunk * seconds)):    # Store data in chunks for 3 secs
        data = stream.read(chunk)
        frames.append(data)
# Stop and close the stream
stream.stop_stream()
stream.close()
# Terminate the PortAudio interface
p.terminate()
print('Finished recording ....')       # Save the recorded data as a WAV
filewf = wave.open(filename,'wb')
wf.setnchannels(channels)
wf.setsampwidth(p.get_sample_size(sample_format))
wf.setframerate(fs)
wf.writeframes(b''.join(frames))
wf.close()
```

Speak

```
playsound("./signal.mp3")
playsound('./ihear.mp3’)
playsound('./sound_out.wav’)
playsound('./UV_Room.mp3')
playsound('./thanks.mp3')
```



Real Python

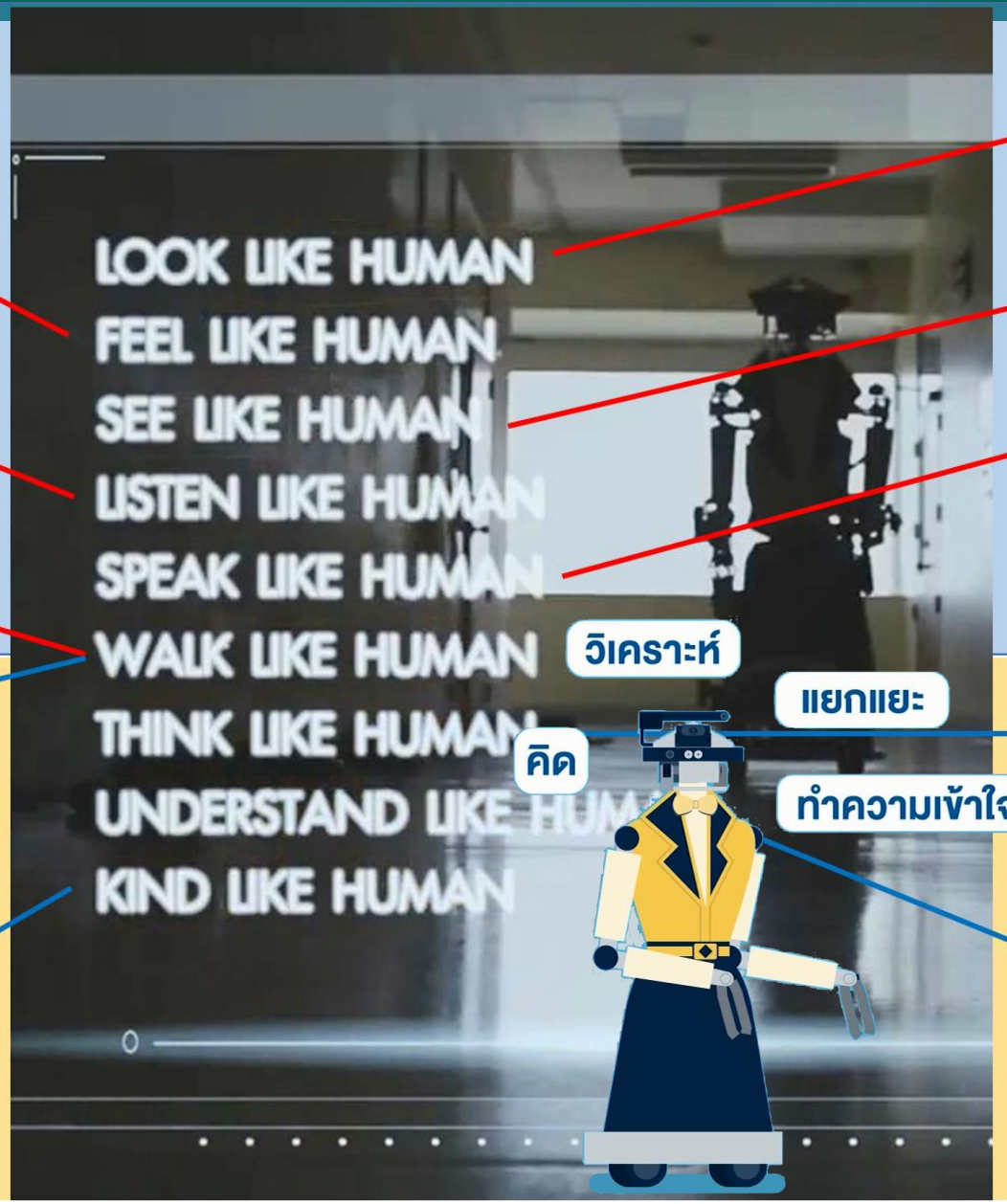
$python Ispeak.py

@Home Robot

Force sensor

Microphone (array)

Wheels or Legs



Hardware/Arms

2D & 3D cameras

Speech to Text
Text to Voice

Sensor/Hardware

Navigation / Path Planning
Object avoidance
Sound direction

Human-Robot Interface
Emotion display
Chat Bot !

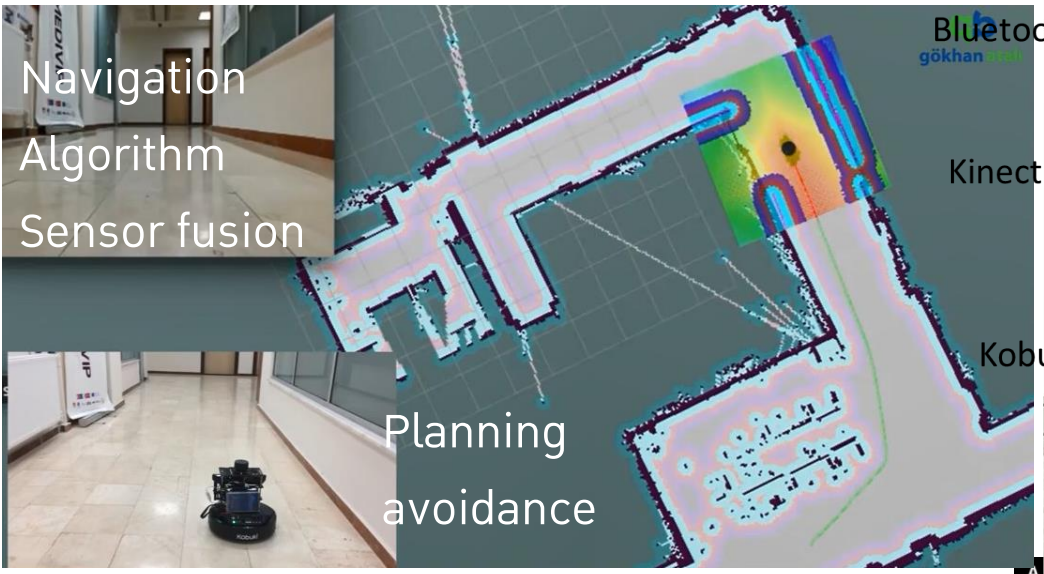
AI / NLP / Finite SM
Algorithms

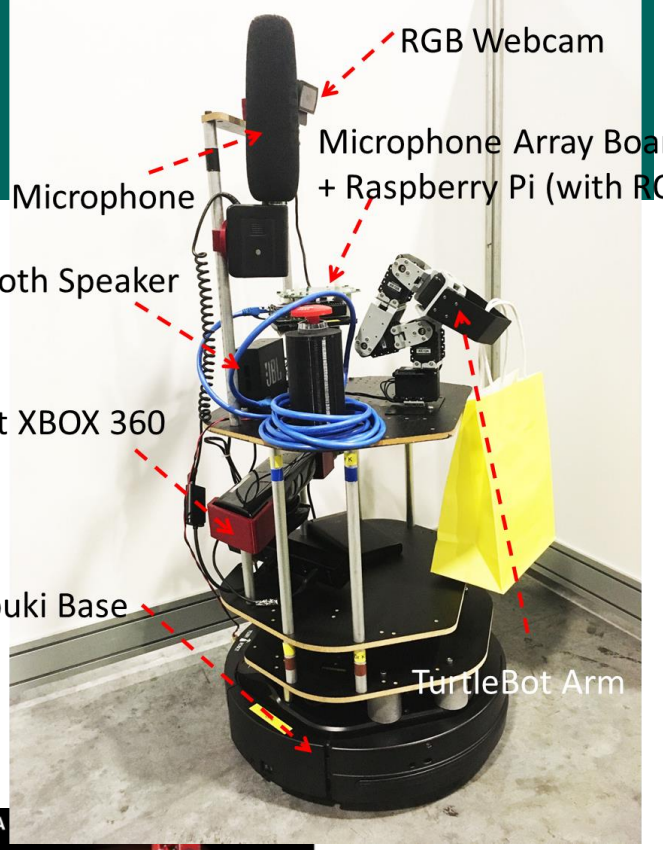
Recognition system
Machine Learning
Machine Vision

Software

@Home Robot

KASETSART UNIVERSITY





WiFi adapter
Standards: IEEE 802.11b/g/n

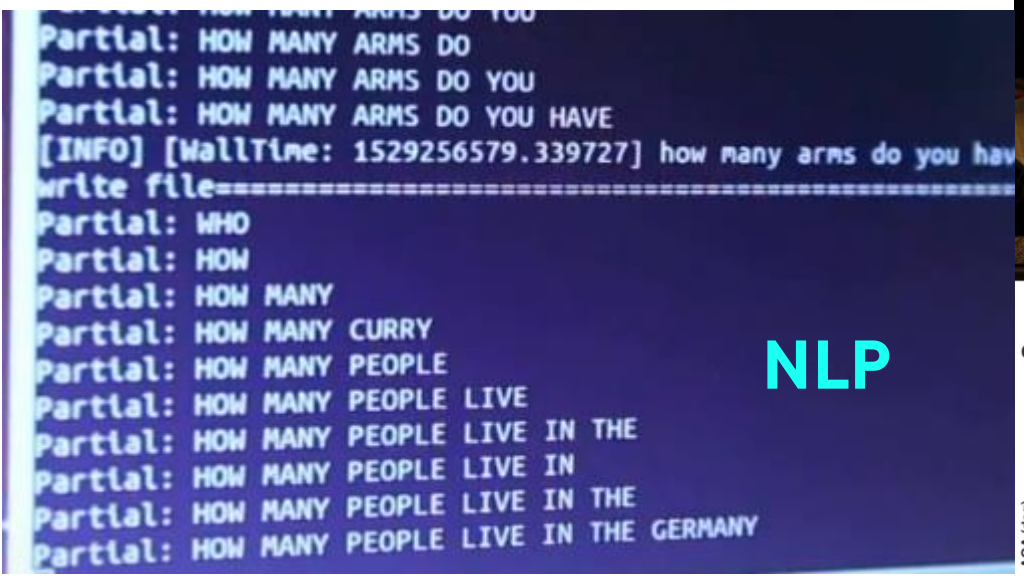
Wide-angle camera
Logitech C905

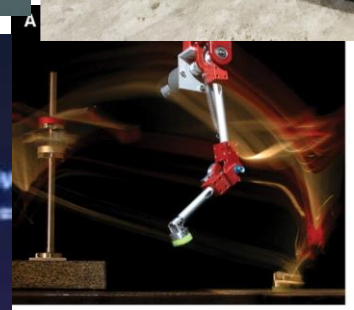
Speaker

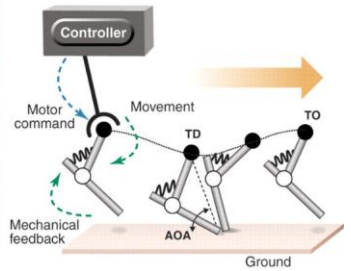
ZBOX nano XS
AMD E-450 processor 1.65 GHz
2 GB RAM, 64 GB SSD
USB 3.0, HDMI
Gigabit Ethernet
Memory card slot

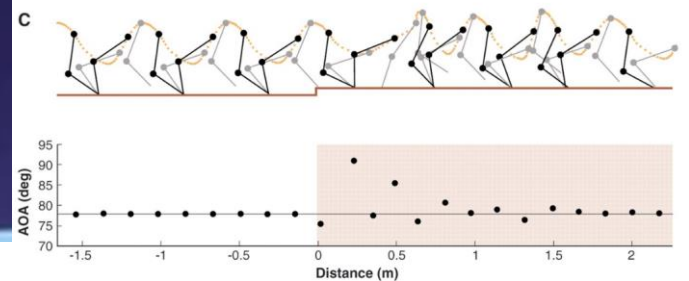
CM730 control board
Servo communication
3-axis accelerometer
3-axis gyroscope

LiPo battery
14.8 V, 3.6 Ah









20 actuators
Networked Dynamixel
MX actuators
6 per leg (MX-106)
3 per arm (MX-64)
2 in the neck (MX-64)

Lightweight materials
Carbon composite
Aluminum
ABS+

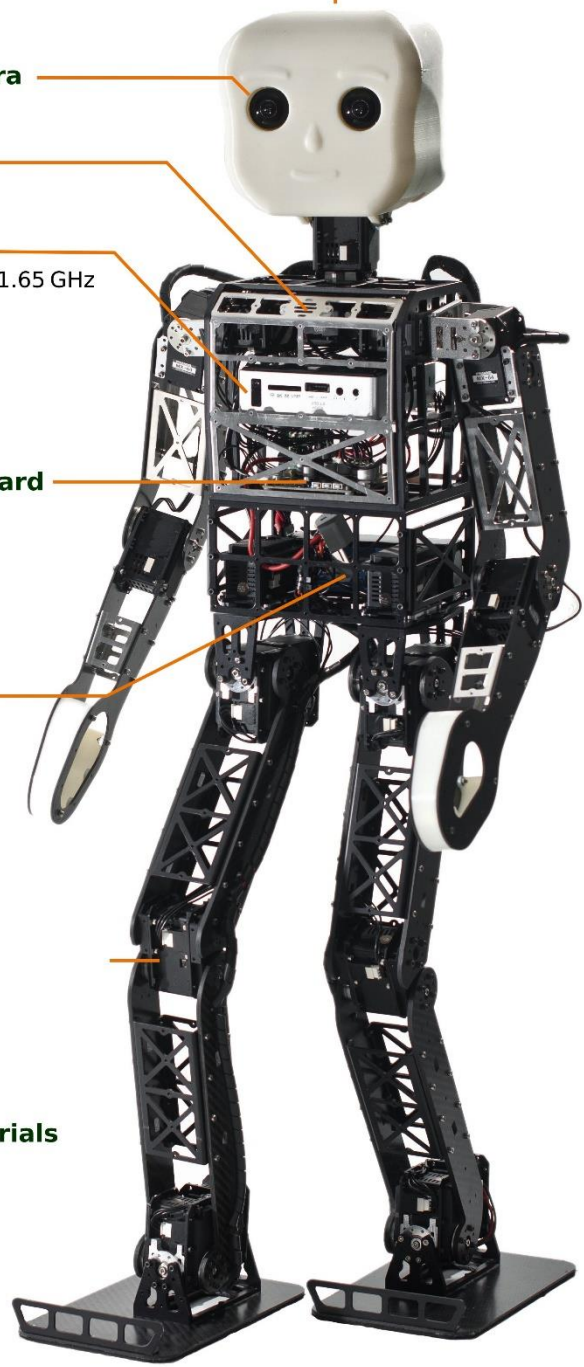


Photo: Felix Oprean

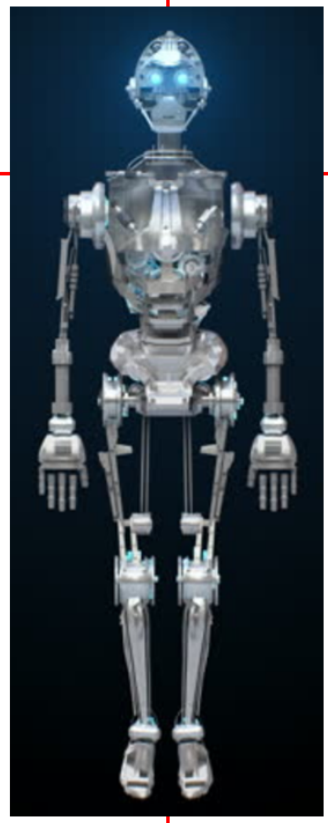
@Home Robot

Hardware

1. Mechanic Design
2. Embedded & Electronic Design
3. Mathematic Modelling
4. Low-Level Firmware
5. Low-level Control algorithm
6. Edge Processing Enhancement

System Integration (SI)

1. Combine Existing Technology
2. Create Application
3. Robustness & Optimization
4. Training
5. Education
6. Commercialization



software

1. Navigation & Avoidance & Path Planning
2. Robot Arm Motion for Specific Trajectory
3. High-Level Control algorithm
4. Machine Vision for 2D and 3D
5. Natural Language Processing
6. Recognition System
7. Decision System
8. Human-Robot Interfaces

THANK YOU