

CUDA AND IMPLEMENTATION ON ROS

OUR SCHEDULE

DAY 1

Session 1 : Revision Yolo and OpenCV

- Quick Flashback on Phase 2

Session 2: Installation OF CUDA

- Grouping
- Check Device
- Download and install CUDA

Revision on YOLO and OpenCV

SESSION 1

A solid orange horizontal bar spanning the width of the slide, located at the bottom.

YOLO



Test Image

Testing Model

Tiny YOLOv3

To use this model, first download the weights:

```
wget https://pjreddie.com/media/files/yolov3-tiny.weights
```

Then run the detector with the tiny config file and weights:

```
./darknet detect cfg/yolov3-tiny.cfg yolov3-tiny.weights data/dog.jpg
```

`./darknet detect <config-file-path> <weight-file-path> <Image-file-path>`

TEST ON WEBCAM

```
./darknet detector demo cfg/coco.data cfg/yolov3.cfg yolov3-tiny.weights
```

```
./darknet detect demo <config-file-path> <weight-file-path>
```

OpenCV

Blurring

```
cv2.imshow('Original', img)
cv2.imshow('cv2 Blur', cv2_blur)
cv2.imshow('GaussianBlur', GaussianBlur)
cv2.imshow('medianBlur', medianBlur)
cv2.imshow('bilateralFilter', bilateralFilter)
cv2.waitKey(0)
```

```
import cv2
import numpy as np
img = cv2.imread('water.png')

cv2_blur = cv2.blur(img, (3,3))

GaussianBlur = cv2.GaussianBlur(img,(5,5),0)

medianBlur = cv2.medianBlur(img,5)

bilateralFilter = cv2.bilateralFilter(img,9,75,75)

cv2.imshow('Original', img)
cv2.imshow('cv2 Blur', cv2_blur)
cv2.imshow('GaussianBlur', GaussianBlur)
cv2.imshow('medianBlur', medianBlur)
cv2.imshow('bilateralFilter', bilateralFilter)
cv2.waitKey(0)
```

Colourspace



```
import cv2
import numpy as np
cap = cv2.VideoCapture(0)

while cv2.waitKey(1) < 0 or False:
    hasframe, img = cap.read()
    img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    img_hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
    img_yuv = cv2.cvtColor(img, cv2.COLOR_BGR2YUV)

    cv2.imshow('Original', img)
    cv2.imshow('img_rgb', img_rgb)
    cv2.imshow('img_gray', img_gray)
    cv2.imshow('img_hsv', img_hsv)
    cv2.imshow('img_yuv', img_yuv)
```

Edge

Task

Show me the four different Edge on image and webcam

Drawing

Task

Write “Revision” on the image / webcam

Contour

```
import numpy as np
import cv2
img = cv2.imread('input_boomerang_shapes.png')

img = cv2.bilateralFilter(img,9,75,75)
imgray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

#canny = cv2.Canny(img, 50, 240)
ret, thresh = cv2.threshold(imgray, 127, 255, 0)
im2, contours, hierarchy = cv2.findContours(thresh, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)

cnt = contours[2]
cv2.drawContours(img, cnt, -1, (0,255,255), 10)

print "number of contour is " + str(len(contours))
print (hierarchy)

cv2.imshow("img",img)
cv2.imshow("canny",thresh)
cv2.waitKey(0)
```

INSTALLATION ON CUDA



Why not AMD DEVICE / Intel Integrated Device ?

AMD has ROCm which is similar however, not many framework support it.

<https://github.com/RadeonOpenCompute/ROCm>

*Not recommended

For Intel, OpenVINO is provided but similar situation as well



Conclusion

Use Nvidia CUDA is easier since got many support and examples online.



Nvidia Docker (Only For Online)

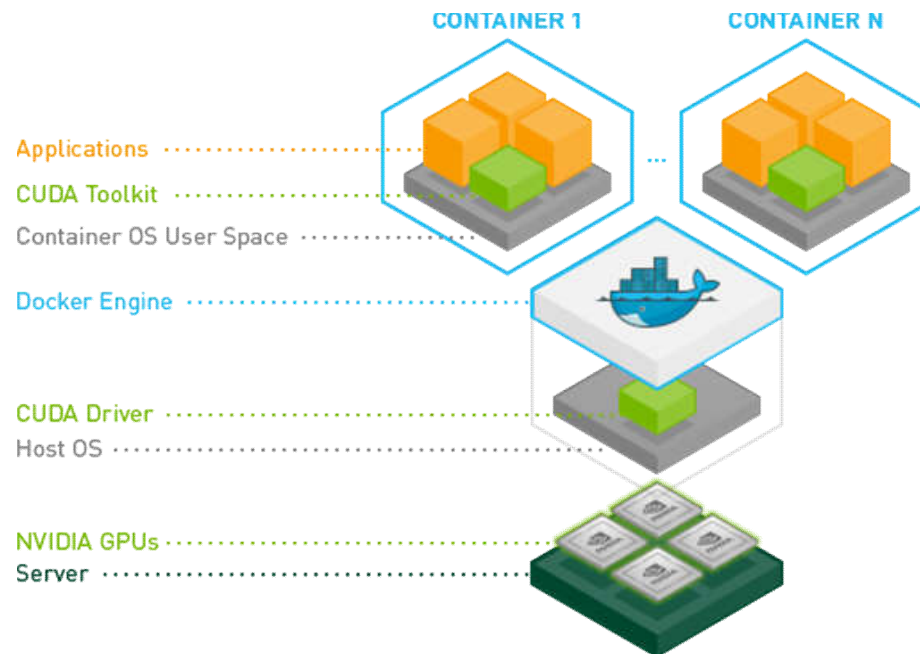
The NVIDIA Container Toolkit allows users to build and run GPU accelerated Docker containers. The toolkit includes a container runtime [library](#) and utilities to automatically configure containers to leverage NVIDIA GPUs.

Advantages:

- Easy to install
- Use it on multiple devices

Disadvantages:

- Requires internet
- Hard to debug



Nvidia Devices

<https://developer.nvidia.com/cuda-gpus>

First, check whether your device have Nvidia GPU

```
$ lspci | grep -i nvidia
```


BACKUP FIRST

For those who don't have Nvidia GPU, may group with others

One group 5-8 person

Requirement

Table 1. CUDA Toolkit and Compatible Driver Versions



CUDA Toolkit	Linux x86_64 Driver Version
CUDA 10.1 (10.1.105)	≥ 418.39
CUDA 10.0 (10.0.130)	≥ 410.48
CUDA 9.2 (9.2.88)	≥ 396.26
CUDA 9.1 (9.1.85)	≥ 390.46
CUDA 9.0 (9.0.76)	≥ 384.81
CUDA 8.0 (8.0.61 GA2)	≥ 375.26
CUDA 8.0 (8.0.44)	≥ 367.48
CUDA 7.5 (7.5.16)	≥ 352.31
CUDA 7.0 (7.0.28)	≥ 346.46

Installation Step

In a new terminal

```
sudo apt-get install linux-headers-$(uname -r)
```

```
sudo apt-get install openjdk-8-jdk git python-dev python3-dev python-numpy python3-numpy build-essential python-pip python3-pip python-virtualenv swig python-wheel libcurl3-dev curl
```

```
curl -O http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86\_64/cuda-repo-ubuntu1604\_10.0.130-1\_amd64.deb
```

```
sudo apt-key adv --fetch-keys
```

```
http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86\_64/7fa2af80.pub
```

```
sudo dpkg -i ./cuda-repo-ubuntu1604_10.0.130-1_amd64.deb
```


Installation Step

`sudo apt-get update`

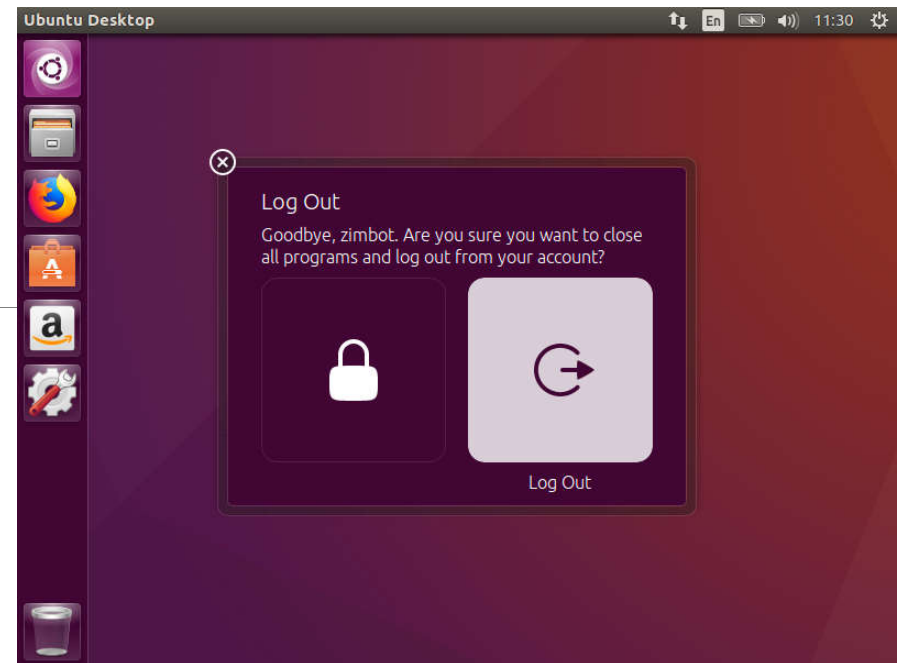
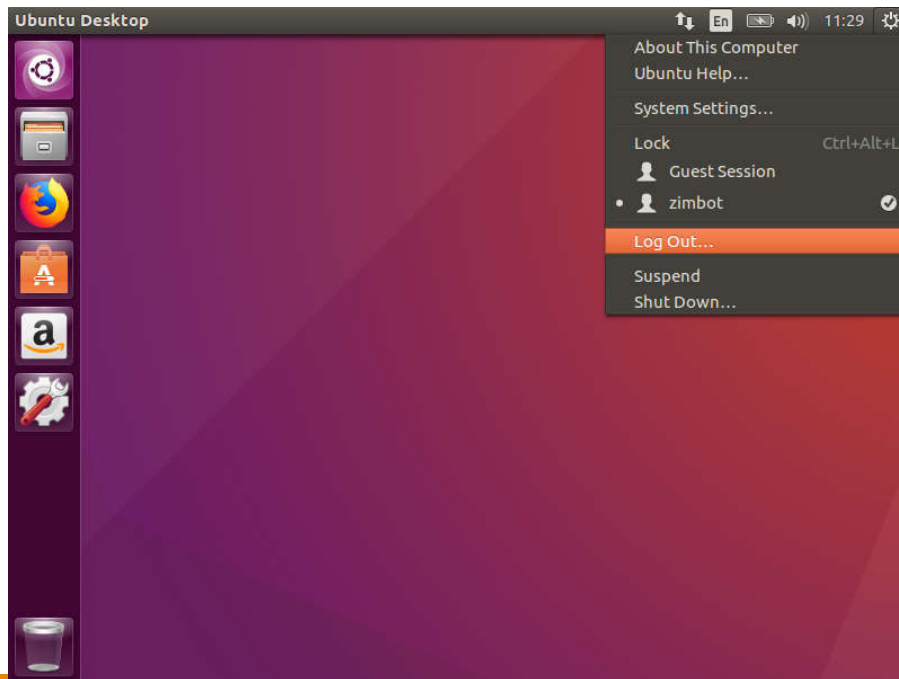
`sudo apt-get install cuda-10-0`

`sudo reboot`

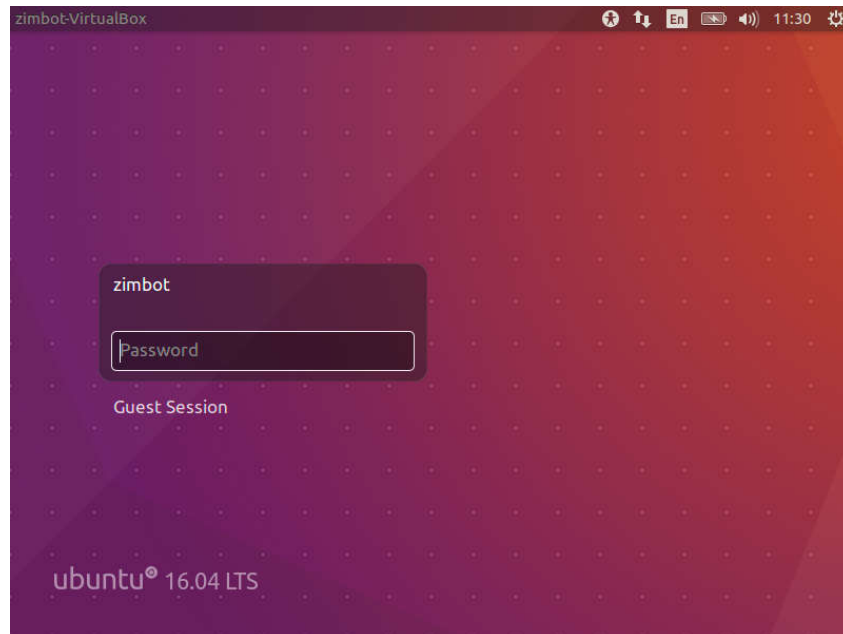
`nvidia-smi`

*If failed, follow these steps

Log out



Alt + Ctrl + F1



Login using username and password

```
Ubuntu 16.04.6 LTS zimbot-VirtualBox tty1
zimbot-VirtualBox login: zimbot
Password:
Last login: Tue Oct 15 23:30:41 +08 2019 on tty1
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-45-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

268 packages can be updated.
193 updates are security updates.

zimbot@zimbot-VirtualBox:~$ _
```

```
268 packages can be updated.  
193 updates are security updates.
```

```
zimbot@zimbot-VirtualBox:~$ sudo apt purge nvidia-*
```

```
sudo apt purge nvidia-*
```
