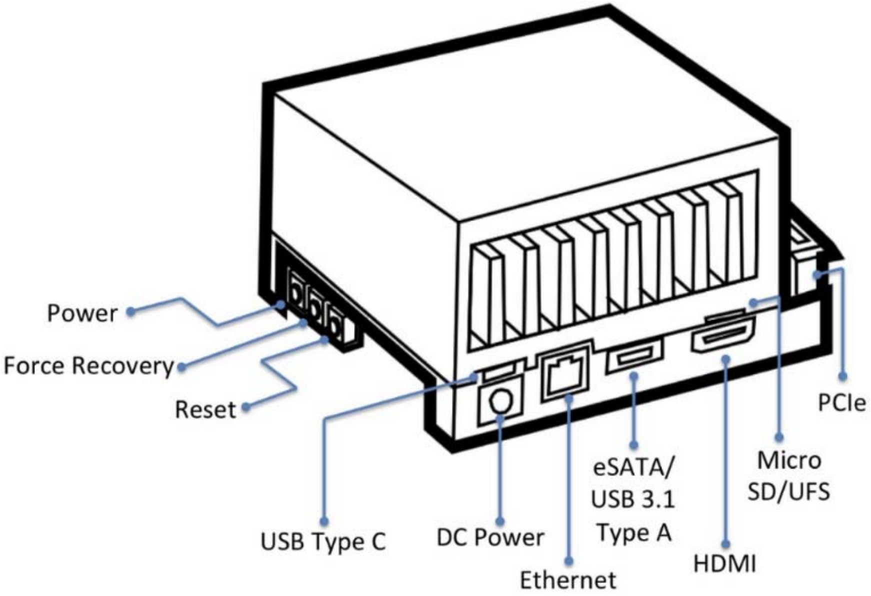
딥러닝(Deep Learning) 실습

DV (주)딥비전

1. USB Camera

Physical Setting

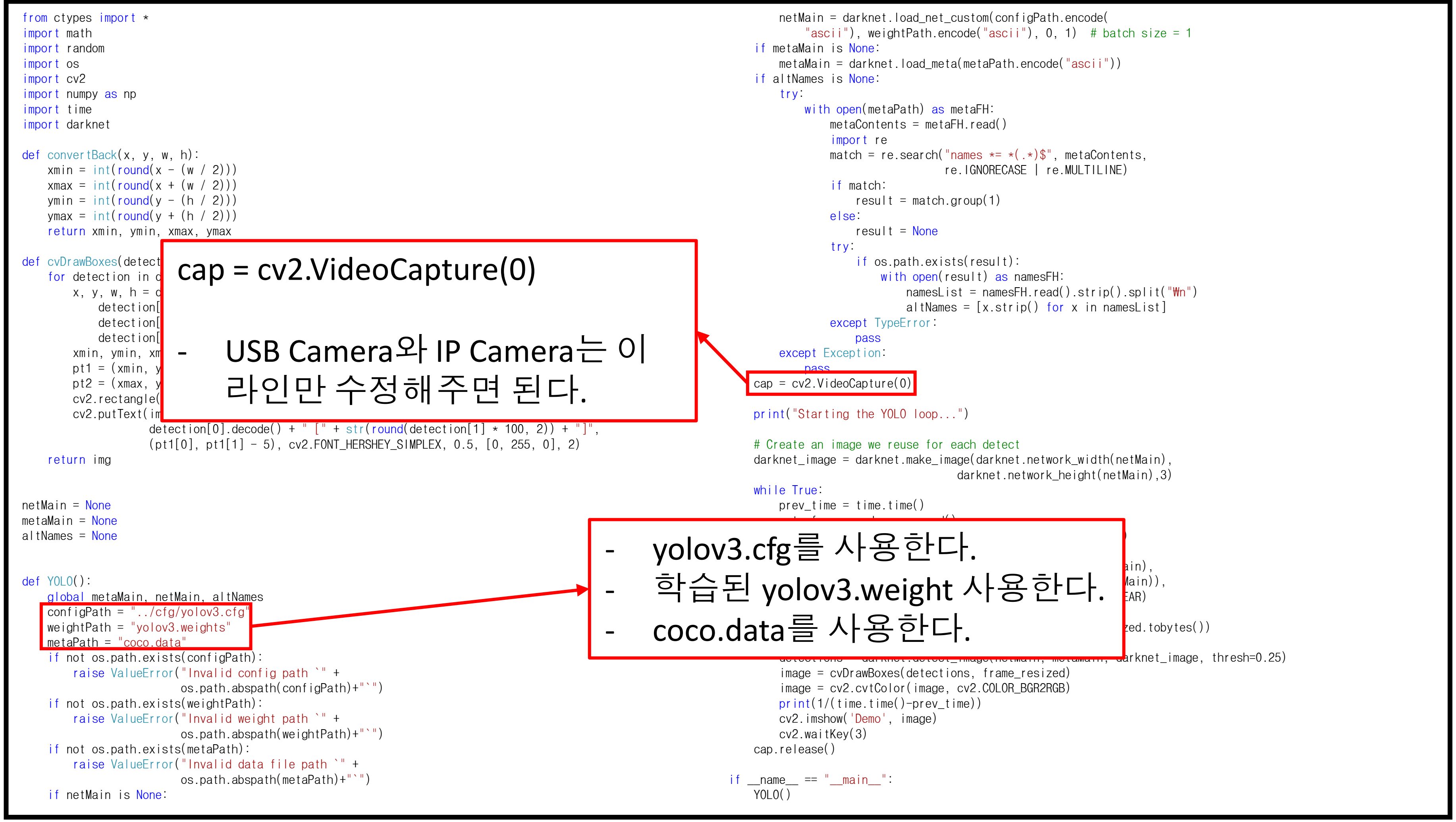
- 뒤쪽 USB Type-C 포트에 제공된 Type C to Type A 어댑터를 통해 연결한다.
  (허브 사용이 가능하다.)



Rear & Right

1. USB Camera

Code



```
from ctypes import *
import math
import random
import os
import cv2
import numpy as np
import time
import darknet

def convertBack(x, y, w, h):
    xmin = int(round(x - (w / 2)))
    xmax = int(round(x + (w / 2)))
    ymin = int(round(y - (h / 2)))
    ymax = int(round(y + (h / 2)))
    return xmin, ymin, xmax, ymax

def cvDrawBoxes(detect
    for detection in d
        x, y, w, h = d
            detection[
            detection[
            detection[
        xmin, ymin, xm
        pt1 = (xmin, y
        pt2 = (xmax, y
        cv2.rectangle(
        cv2.putText(im
                    detection[0].decode() + " [" + str(round(detection[1] * 100, 2)) + "]",
                    (pt1[0], pt1[1] - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.5, [0, 255, 0], 2)
    return img


netMain = None
metaMain = None
altNames = None


def YOLO():
    global metaMain, netMain, altNames
    configPath = "../cfg/yolov3.cfg"
    weightPath = "yolov3.weights"
    metaPath = "coco.data"
    if not os.path.exists(configPath):
        raise ValueError("Invalid config path `" +
                         os.path.abspath(configPath)+"`")
    if not os.path.exists(weightPath):
        raise ValueError("Invalid weight path `" +
                         os.path.abspath(weightPath)+"`")
    if not os.path.exists(metaPath):
        raise ValueError("Invalid data file path `" +
                         os.path.abspath(metaPath)+"`")
    if netMain is None:
        netMain = darknet.load_net_custom(configPath.encode(
            "ascii"), weightPath.encode("ascii"), 0, 1)  # batch size = 1
    if metaMain is None:
        metaMain = darknet.load_meta(metaPath.encode("ascii"))
    if altNames is None:
        try:
            with open(metaPath) as metaFH:
                metaContents = metaFH.read()
                import re
                match = re.search("names *= *(.*)$", metaContents,
                                  re.IGNORECASE | re.MULTILINE)
                if match:
                    result = match.group(1)
                else:
                    result = None
                try:
                    if os.path.exists(result):
                        with open(result) as namesFH:
                            namesList = namesFH.read().strip().split("₩n")
                            altNames = [x.strip() for x in namesList]
                except TypeError:
                    pass
        except Exception:
            pass
    cap = cv2.VideoCapture(0)

    print("Starting the YOLO loop...")

    # Create an image we reuse for each detect
    darknet_image = darknet.make_image(darknet.network_width(netMain),
                                    darknet.network_height(netMain),3)
    while True:
        prev_time = time.time()
                                                           ain),
                                                           Main)),
                                                           EAR)
                                                           zed.tobytes())
                                                           arknet_image, thresh=0.25)
        image = cvDrawBoxes(detections, frame_resized)
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        print(1/(time.time()-prev_time))
        cv2.imshow('Demo', image)
        cv2.waitKey(3)
    cap.release()

if __name__ == "__main__":
    YOLO()
```

cap = cv2.VideoCapture(0)

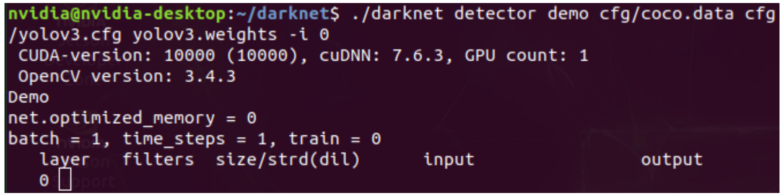
- USB Camera와 IP Camera는 이 라인만 수정해주면 된다.

- yolov3.cfg를 사용한다.
- 학습된 yolov3.weight 사용한다.
- coco.data를 사용한다.

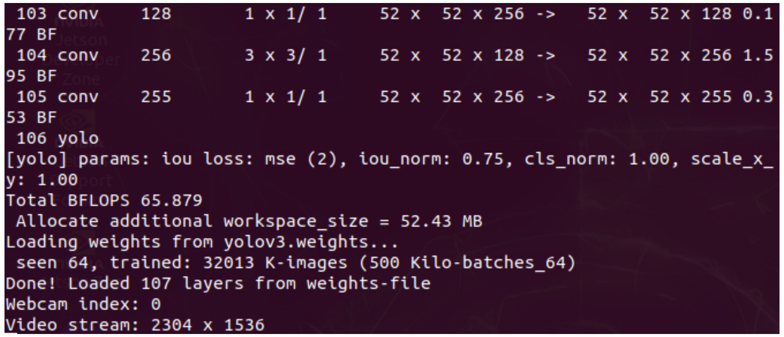
1. USB Camera

실행

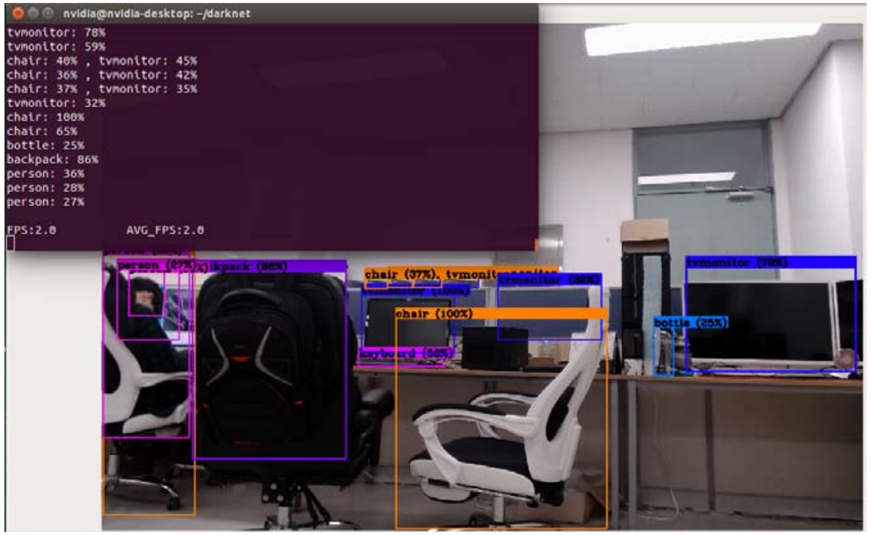
```
$ ./darknet detector demo cfg/coco.data cfg/yolov3.cfg yolov3.weights -i 0
```



```
nvidia@nvidia-desktop:~/darknet$ ./darknet detector demo cfg/coco.data cfg
/yolov3.cfg yolov3.weights -i 0
 CUDA-version: 10000 (10000), cuDNN: 7.6.3, GPU count: 1
 OpenCV version: 3.4.3
Demo
net.optimized_memory = 0
batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 
```



```
 103 conv    128       1 x 1/ 1     52 x  52 x 256 ->   52 x  52 x 128 0.1
77 BF
 104 conv    256       3 x 3/ 1     52 x  52 x 128 ->   52 x  52 x 256 1.5
95 BF
 105 conv    255       1 x 1/ 1     52 x  52 x 256 ->   52 x  52 x 255 0.3
53 BF
 106 yolo
[yolo] params: iou loss: mse (2), iou_norm: 0.75, cls_norm: 1.00, scale_x_
y: 1.00
Total BFLOPS 65.879
 Allocate additional workspace_size = 52.43 MB
Loading weights from yolov3.weights...
 seen 64, trained: 32013 K-images (500 Kilo-batches_64)
Done! Loaded 107 layers from weights-file
Webcam index: 0
Video stream: 2304 x 1536
```



```
tvmonitor: 78%
tvmonitor: 59%
chair: 40% , tvmonitor: 45%
chair: 36% , tvmonitor: 42%
chair: 37% , tvmonitor: 35%
tvmonitor: 32%
chair: 100%
chair: 65%
bottle: 25%
backpack: 86%
person: 36%
person: 28%
person: 27%

FPS:2.0        AVG_FPS:2.0
```

종료방법 : 터미널 창에서 Ctrl + c

1. USB Camera

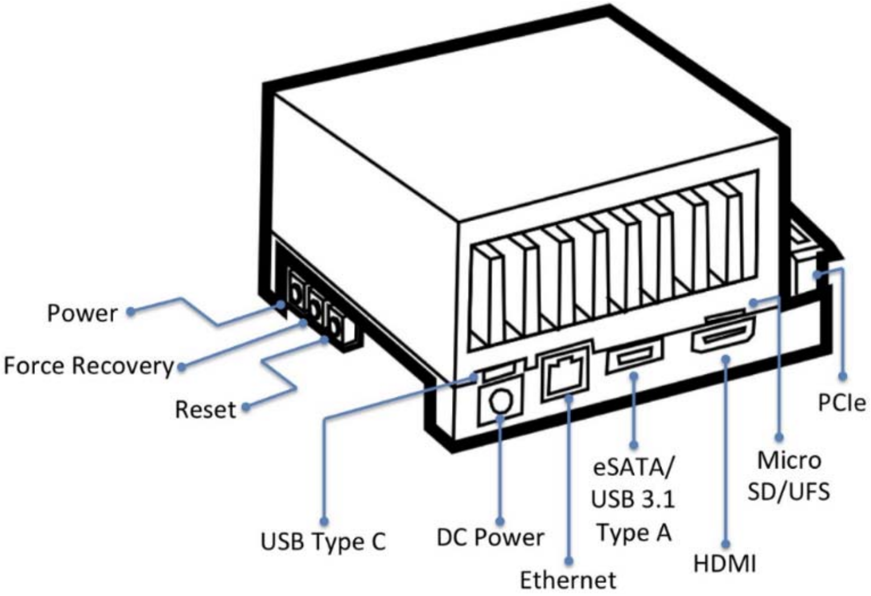
전체 Code

```python
from ctypes import *
import math
import random
import os
import cv2
import numpy as np
import time
import darknet

def convertBack(x, y, w, h):
    xmin = int(round(x - (w / 2)))
    xmax = int(round(x + (w / 2)))
    ymin = int(round(y - (h / 2)))
    ymax = int(round(y + (h / 2)))
    return xmin, ymin, xmax, ymax

def cvDrawBoxes(detections, img):
    for detection in detections:
        x, y, w, h = detection[2][0],₩
            detection[2][1],₩
            detection[2][2],₩
            detection[2][3]
        xmin, ymin, xmax, ymax = convertBack(float(x), float(y), float(w), float(h))
        pt1 = (xmin, ymin)
        pt2 = (xmax, ymax)
        cv2.rectangle(img, pt1, pt2, (0, 255, 0), 1)
        cv2.putText(img,
                    detection[0].decode() + " [" + str(round(detection[1] * 100, 2)) + "]",
                    (pt1[0], pt1[1] - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.5, [0, 255, 0], 2)
    return img


netMain = None
metaMain = None
altNames = None


def YOLO():
    global metaMain, netMain, altNames
    configPath = "../cfg/yolov3.cfg"
    weightPath = "yolov3.weights"
    metaPath = "coco.data"
    if not os.path.exists(configPath):
        raise ValueError("Invalid config path `" +
                         os.path.abspath(configPath)+"`")
    if not os.path.exists(weightPath):
        raise ValueError("Invalid weight path `" +
                         os.path.abspath(weightPath)+"`")
    if not os.path.exists(metaPath):
        raise ValueError("Invalid data file path `" +
                         os.path.abspath(metaPath)+"`")
    if netMain is None:
        netMain = darknet.load_net_custom(configPath.encode(
            "ascii"), weightPath.encode("ascii"), 0, 1)  # batch size = 1
    if metaMain is None:
        metaMain = darknet.load_meta(metaPath.encode("ascii"))
    if altNames is None:
        try:
            with open(metaPath) as metaFH:
                metaContents = metaFH.read()
                import re
                match = re.search("names *= *(.*)$", metaContents,
                                  re.IGNORECASE | re.MULTILINE)
                if match:
                    result = match.group(1)
                else:
                    result = None
                try:
                    if os.path.exists(result):
                        with open(result) as namesFH:
                            namesList = namesFH.read().strip().split("₩n")
                            altNames = [x.strip() for x in namesList]
                except TypeError:
                    pass
        except Exception:
            pass
    cap = cv2.VideoCapture(0)

    print("Starting the YOLO loop...")

    # Create an image we reuse for each detect
    darknet_image = darknet.make_image(darknet.network_width(netMain),
                                       darknet.network_height(netMain),3)
    while True:
        prev_time = time.time()
        ret, frame_read = cap.read()
        frame_rgb = cv2.cvtColor(frame_read, cv2.COLOR_BGR2RGB)
        frame_resized = cv2.resize(frame_rgb,
                                   (darknet.network_width(netMain),
                                    darknet.network_height(netMain)),
                                   interpolation=cv2.INTER_LINEAR)

        darknet.copy_image_from_bytes(darknet_image,frame_resized.tobytes())

        detections = darknet.detect_image(netMain, metaMain, darknet_image, thresh=0.25)
        image = cvDrawBoxes(detections, frame_resized)
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        print(1/(time.time()-prev_time))
        cv2.imshow('Demo', image)
        cv2.waitKey(3)
    cap.release()

if __name__ == "__main__":
    YOLO()
```

2. IP Camera

Physical Setting

- 뒤쪽 Ethernet 포트에 Ethernet Cable을 통해 연결한다.

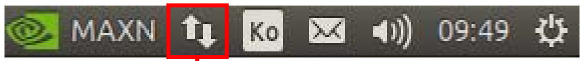


Rear & Right

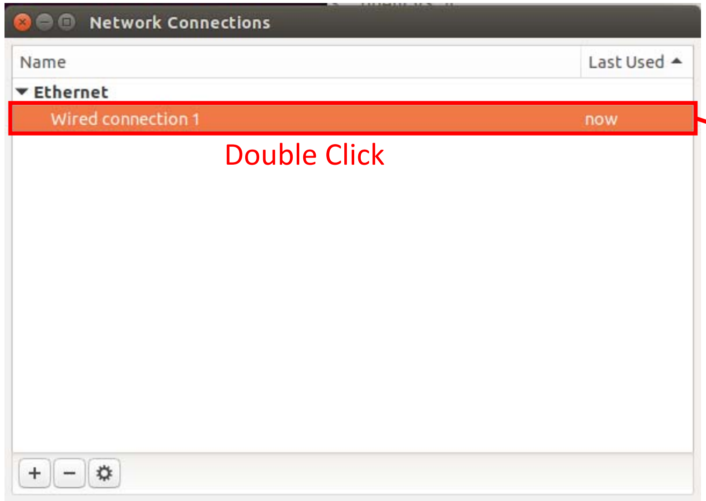
2. IP Camera

IP Setting

- 우분투가 설치된 Xavier에서 설정한다.



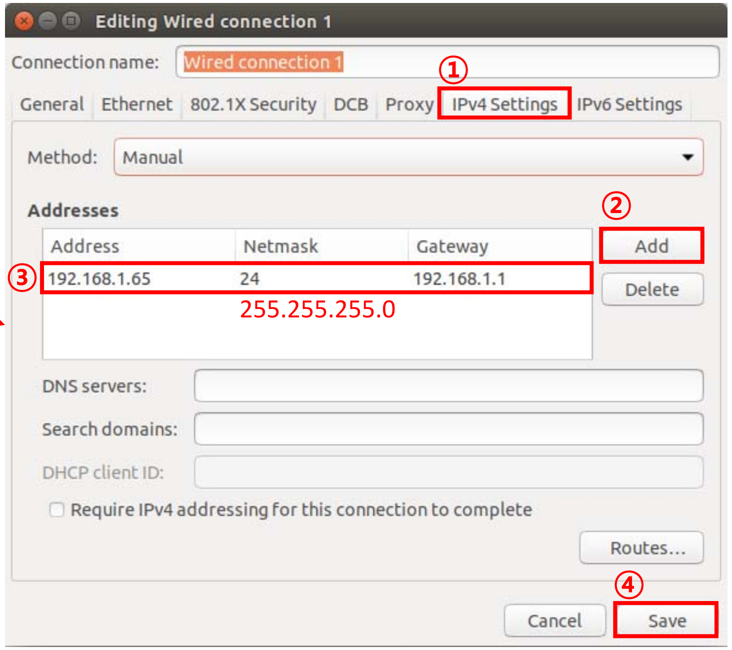
Edit Connections...



Name	Last Used
Ethernet	
Wired connection 1	now

Double Click

Put 255.255.255.0 in Netmask space



Connection name: Wired connection 1

①

Method: Manual

Addresses

②

Address	Netmask	Gateway
192.168.1.65	24	192.168.1.1

③

255.255.255.0

DNS servers:

Search domains:

DHCP client ID:

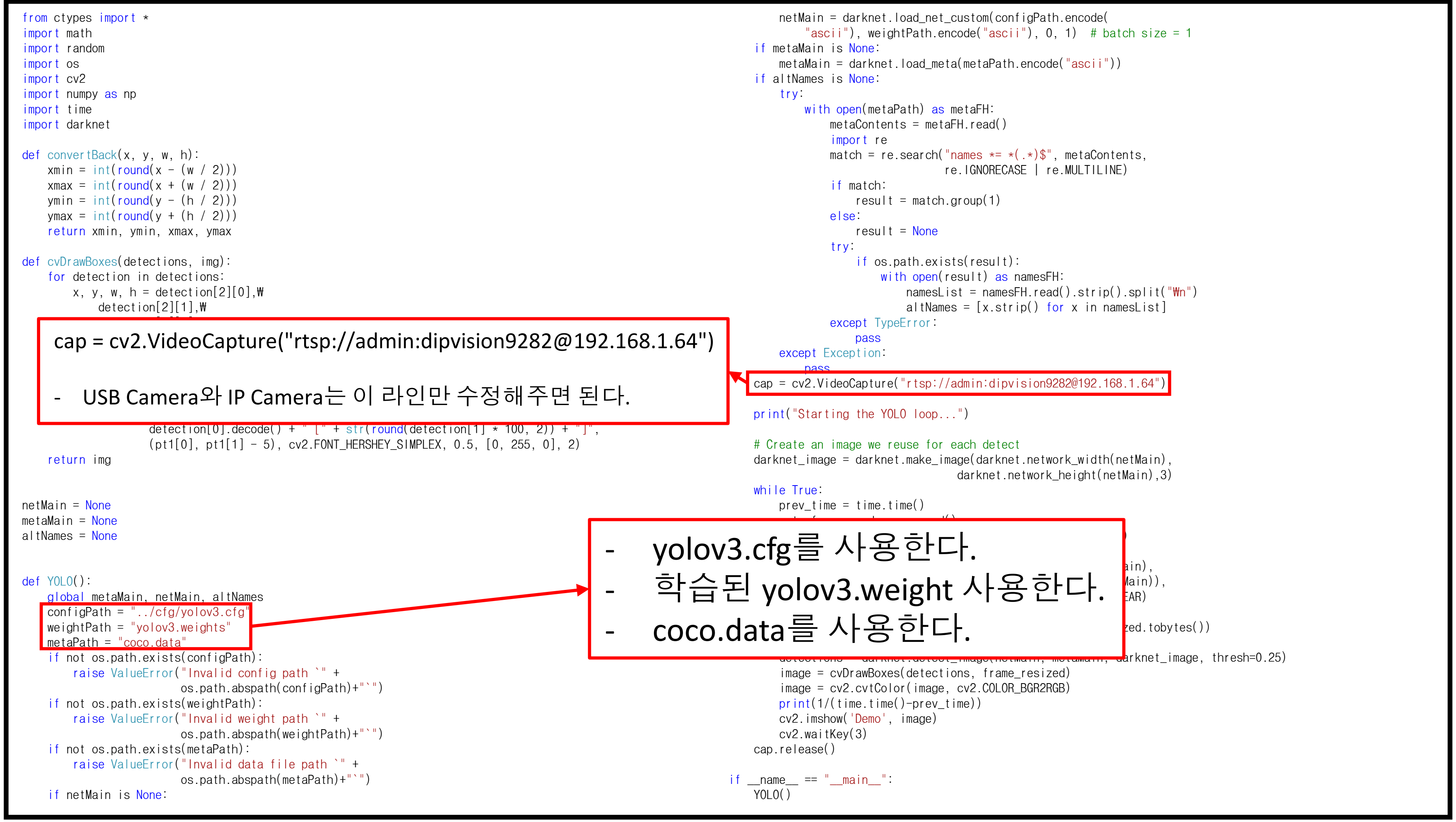
□ Require IPv4 addressing for this connection to complete

④

설정 저장 후 네트워크 아이콘을 눌러 네트워크를 끊고 다시 연결한다.

2. IP Camera

Code



```python
from ctypes import *
import math
import random
import os
import cv2
import numpy as np
import time
import darknet

def convertBack(x, y, w, h):
    xmin = int(round(x - (w / 2)))
    xmax = int(round(x + (w / 2)))
    ymin = int(round(y - (h / 2)))
    ymax = int(round(y + (h / 2)))
    return xmin, ymin, xmax, ymax

def cvDrawBoxes(detections, img):
    for detection in detections:
        x, y, w, h = detection[2][0],₩
            detection[2][1],₩
                        detection[0].decode() + " [" + str(round(detection[1] * 100, 2)) + "]",
                        (pt1[0], pt1[1] - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.5, [0, 255, 0], 2)
    return img


netMain = None
metaMain = None
altNames = None


def YOLO():
    global metaMain, netMain, altNames
    configPath = "../cfg/yolov3.cfg"
    weightPath = "yolov3.weights"
    metaPath = "coco.data"
    if not os.path.exists(configPath):
        raise ValueError("Invalid config path `" +
                         os.path.abspath(configPath)+"`")
    if not os.path.exists(weightPath):
        raise ValueError("Invalid weight path `" +
                         os.path.abspath(weightPath)+"`")
    if not os.path.exists(metaPath):
        raise ValueError("Invalid data file path `" +
                         os.path.abspath(metaPath)+"`")
    if netMain is None:
        netMain = darknet.load_net_custom(configPath.encode(
            "ascii"), weightPath.encode("ascii"), 0, 1)  # batch size = 1
    if metaMain is None:
        metaMain = darknet.load_meta(metaPath.encode("ascii"))
    if altNames is None:
        try:
            with open(metaPath) as metaFH:
                metaContents = metaFH.read()
                import re
                match = re.search("names *= *(.*)$", metaContents,
                                  re.IGNORECASE | re.MULTILINE)
                if match:
                    result = match.group(1)
                else:
                    result = None
                try:
                    if os.path.exists(result):
                        with open(result) as namesFH:
                            namesList = namesFH.read().strip().split("₩n")
                            altNames = [x.strip() for x in namesList]
                except TypeError:
                    pass
        except Exception:
            pass
    cap = cv2.VideoCapture("rtsp://admin:dipvision9282@192.168.1.64")

    print("Starting the YOLO loop...")

    # Create an image we reuse for each detect
    darknet_image = darknet.make_image(darknet.network_width(netMain),
                                    darknet.network_height(netMain),3)
    while True:
        prev_time = time.time()
                                                          ain),
                                                          Main)),
                                                          EAR)

                                                          zed.tobytes())

                                                          arknet_image, thresh=0.25)
        image = cvDrawBoxes(detections, frame_resized)
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        print(1/(time.time()-prev_time))
        cv2.imshow('Demo', image)
        cv2.waitKey(3)
    cap.release()

if __name__ == "__main__":
    YOLO()
```

cap = cv2.VideoCapture("rtsp://admin:dipvision9282@192.168.1.64")

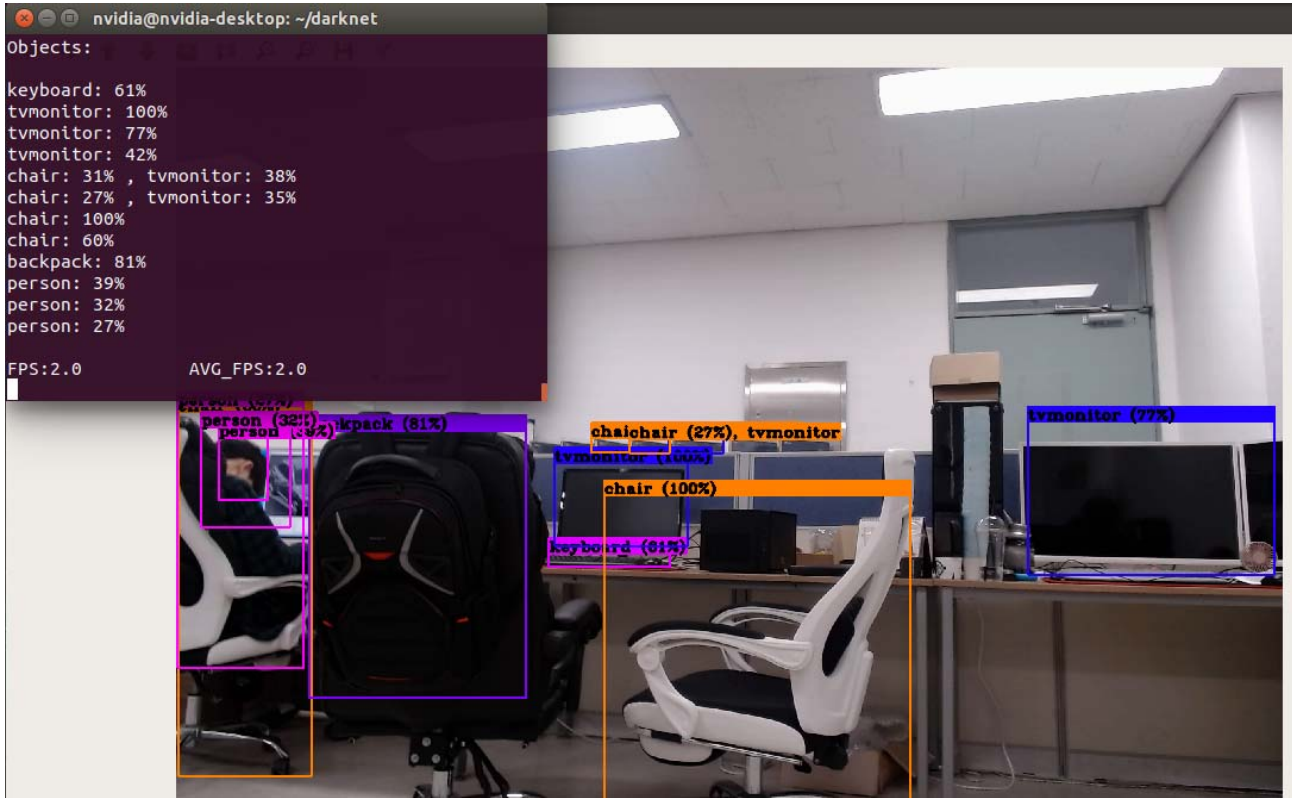
- USB Camera와 IP Camera는 이 라인만 수정해주면 된다.

- yolov3.cfg를 사용한다.
- 학습된 yolov3.weight 사용한다.
- coco.data를 사용한다.

2. IP Camera

실행

```
$ ./darknet detector demo cfg/coco.data cfg/yolov3.cfg yolov3.weights
rtsp://admin:dipvision9282@192.168.1.64 -i 0
```



```
Objects:

keyboard: 61%
tvmonitor: 100%
tvmonitor: 77%
tvmonitor: 42%
chair: 31% , tvmonitor: 38%
chair: 27% , tvmonitor: 35%
chair: 100%
chair: 60%
backpack: 81%
person: 39%
person: 32%
person: 27%

FPS:2.0          AVG_FPS:2.0
```

2. IP Camera

전체 Code

```
from ctypes import *
import math
import random
import os
import cv2
import numpy as np
import time
import darknet

def convertBack(x, y, w, h):
    xmin = int(round(x - (w / 2)))
    xmax = int(round(x + (w / 2)))
    ymin = int(round(y - (h / 2)))
    ymax = int(round(y + (h / 2)))
    return xmin, ymin, xmax, ymax

def cvDrawBoxes(detections, img):
    for detection in detections:
        x, y, w, h = detection[2][0],₩
            detection[2][1],₩
            detection[2][2],₩
            detection[2][3]
        xmin, ymin, xmax, ymax = convertBack(float(x), float(y), float(w), float(h))
        pt1 = (xmin, ymin)
        pt2 = (xmax, ymax)
        cv2.rectangle(img, pt1, pt2, (0, 255, 0), 1)
        cv2.putText(img,
                    detection[0].decode() + " [" + str(round(detection[1] * 100, 2)) + "]",
                    (pt1[0], pt1[1] - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.5, [0, 255, 0], 2)
    return img


netMain = None
metaMain = None
altNames = None


def YOLO():
    global metaMain, netMain, altNames
    configPath = "../cfg/yolov3.cfg"
    weightPath = "yolov3.weights"
    metaPath = "coco.data"
    if not os.path.exists(configPath):
        raise ValueError("Invalid config path `" +
                         os.path.abspath(configPath)+"`")
    if not os.path.exists(weightPath):
        raise ValueError("Invalid weight path `" +
                         os.path.abspath(weightPath)+"`")
    if not os.path.exists(metaPath):
        raise ValueError("Invalid data file path `" +
                         os.path.abspath(metaPath)+"`")
    if netMain is None:
        netMain = darknet.load_net_custom(configPath.encode(
            "ascii"), weightPath.encode("ascii"), 0, 1)  # batch size = 1
    if metaMain is None:
        metaMain = darknet.load_meta(metaPath.encode("ascii"))
    if altNames is None:
        try:
            with open(metaPath) as metaFH:
                metaContents = metaFH.read()
                import re
                match = re.search("names *= *(.*)$", metaContents,
                                  re.IGNORECASE | re.MULTILINE)
                if match:
                    result = match.group(1)
                else:
                    result = None
                try:
                    if os.path.exists(result):
                        with open(result) as namesFH:
                            namesList = namesFH.read().strip().split("₩n")
                            altNames = [x.strip() for x in namesList]
                except TypeError:
                    pass
        except Exception:
            pass
    cap = cv2.VideoCapture("rtsp://admin:dipvision9282@192.168.1.64")

    print("Starting the YOLO loop...")

    # Create an image we reuse for each detect
    darknet_image = darknet.make_image(darknet.network_width(netMain),
                                    darknet.network_height(netMain),3)
    while True:
        prev_time = time.time()
        ret, frame_read = cap.read()
        frame_rgb = cv2.cvtColor(frame_read, cv2.COLOR_BGR2RGB)
        frame_resized = cv2.resize(frame_rgb,
                                   (darknet.network_width(netMain),
                                    darknet.network_height(netMain)),
                                   interpolation=cv2.INTER_LINEAR)

        darknet.copy_image_from_bytes(darknet_image,frame_resized.tobytes())

        detections = darknet.detect_image(netMain, metaMain, darknet_image, thresh=0.25)
        image = cvDrawBoxes(detections, frame_resized)
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        print(1/(time.time()-prev_time))
        cv2.imshow('Demo', image)
        cv2.waitKey(3)
    cap.release()

if __name__ == "__main__":
    YOLO()
```

감사합니다.

Thank you.