# 비전기반 물체검출 실습

Turtlebot HOST PC Raspberry PI를 이용한 Yolo 기반 물체검출 알고리즘 적용 실습

**한양대학교**

**최 준 원**

# Contents

- Raspberry PI OS Setup
- Raspberry PI initial Setting
- Check if OpenCV is installed & Remove existing Opencv
- Various library installation For OpenCV
- OpenCV install (Build & Compile, Install)
- Darknet install (Modify Makefile, Compile)
- Run YOLO_v3 (Single Image & PI Camera)

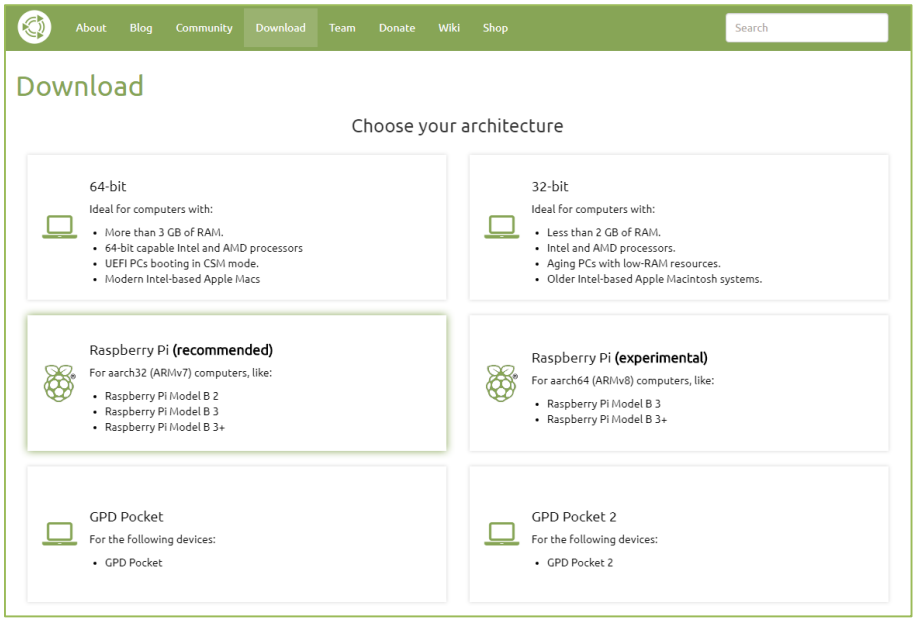
# Raspberry PI OS Setup

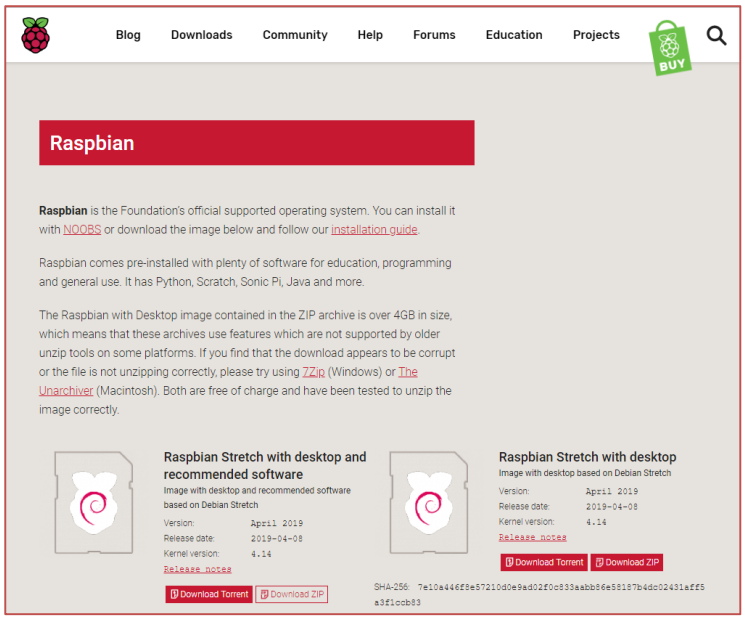
- **Ubuntu Mate 16.04 설치 시**
  - Ubuntu Mate 16.04.2 Raspberry Pi용 Image 파일 다운로드

    ※ 현재 서비스 중인 Ubuntu Mate는 18.04 Beta1으로 미지원 library가 다수 존재, OpenCV 설치 불가.

    - Download URL : https://ubuntu-mate.org/raspberry-pi/ubuntu-mate-16.04.2-desktop-armhf-raspberry-pi.img.xz
- **Rasbian 설치 시**
  - Raspberry Pi 공식 사이트에서 Image 다운로드

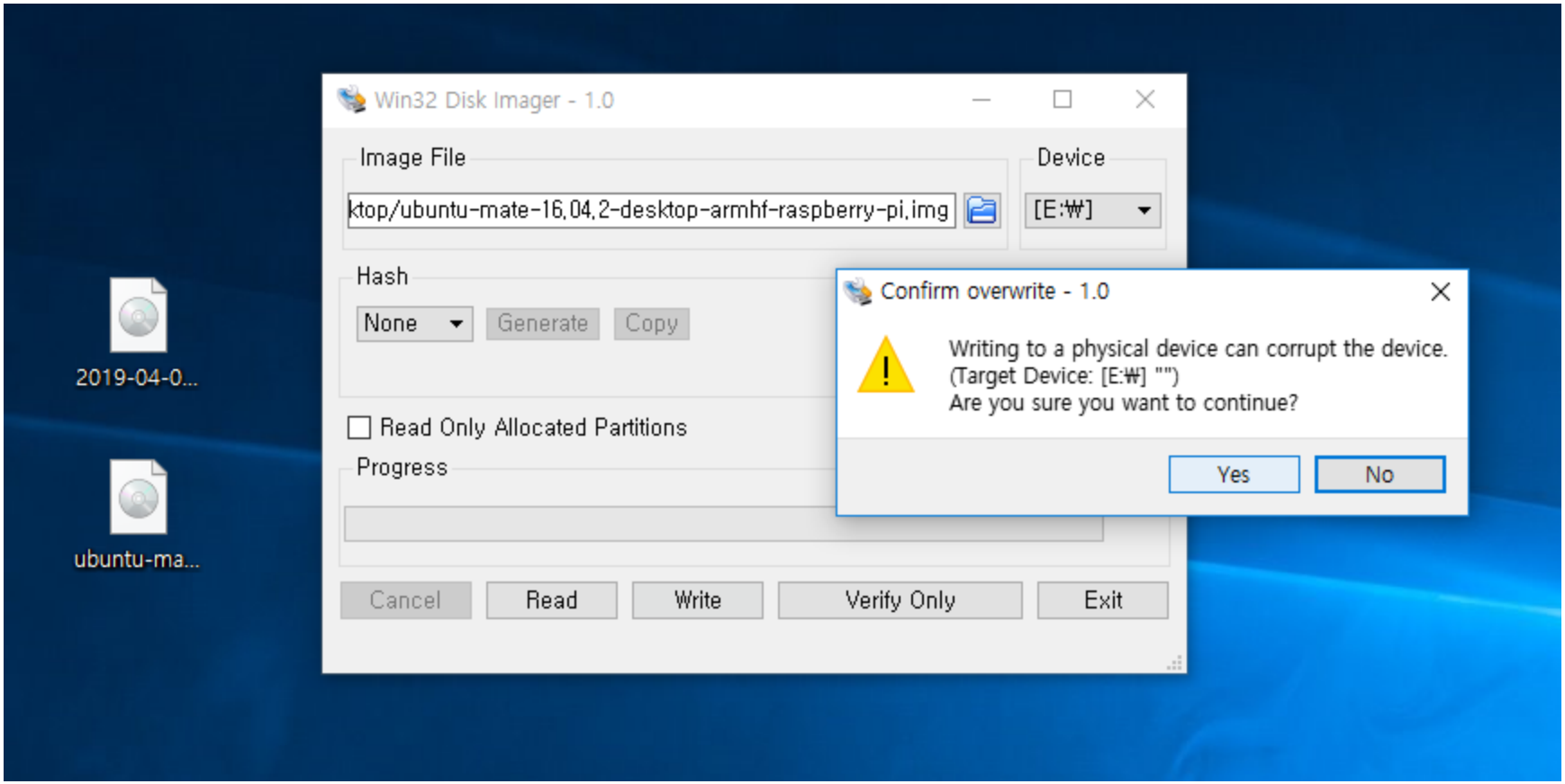
    - Download URL : https://www.raspberrypi.org/downloads/raspbian/



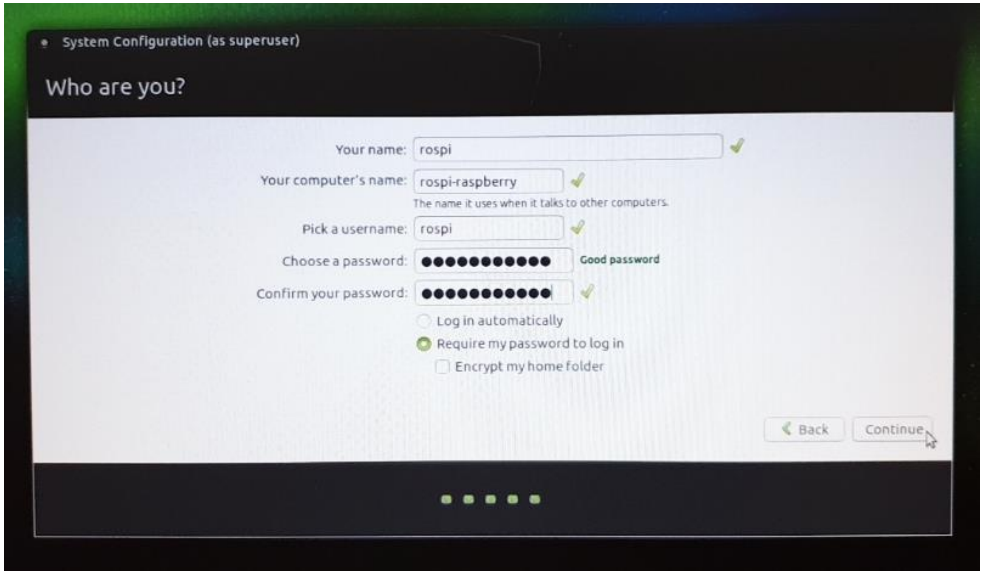
# Raspberry PI OS Setup

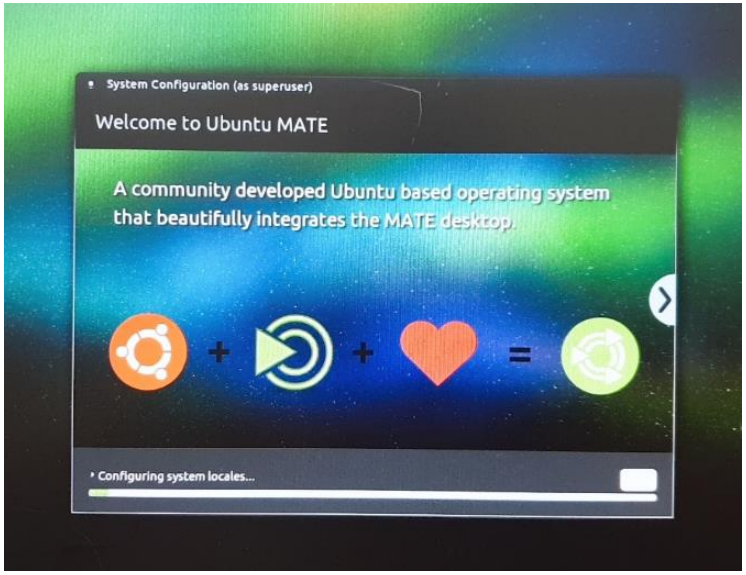
- **Win32DiskImager 등 프로그램 이용, SD Card로 Image Writing**



# Raspberry PI OS Setup

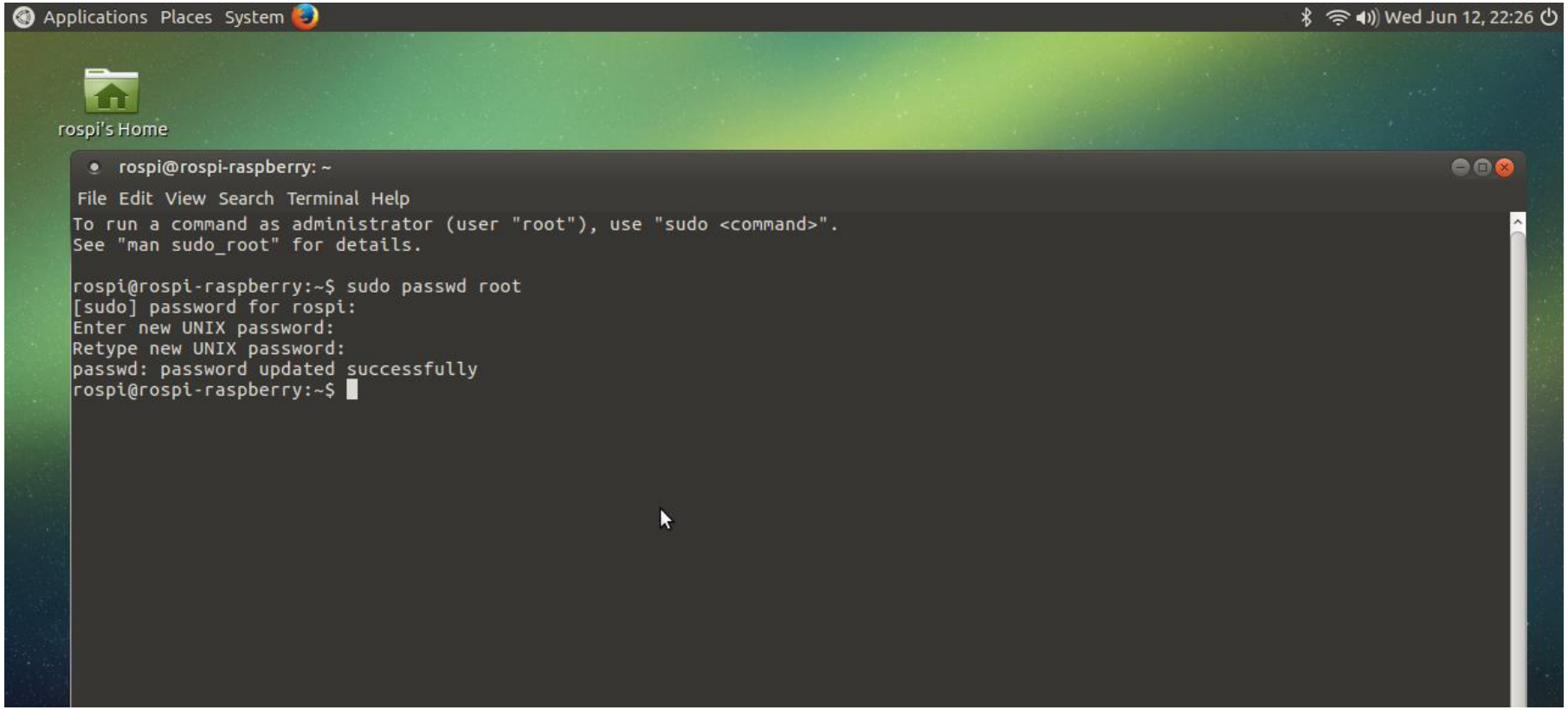
- **Raspbian 설치 시**
  - Image writing을 마친 SD카드를 Raspberry PI에 삽입하고 부팅하면 자동설치
  - 지역 및 언어는 United State 기준, 계정명 pi로 자동 셋팅 완료
  - 따라서 부팅 이후 언어, WIFI, 지역, 키보드 레이아웃 등 별도 설정 필요
- **Ubuntu Mate 16.04 설치 시**
  - Image writing을 마친 SD카드를 Raspberry PI에 삽입하고 부팅하면 설치시작
  - 언어, WIFI, 지역, 키보드 레이아웃, 계정 등 별도의 정보 입력과정 필요



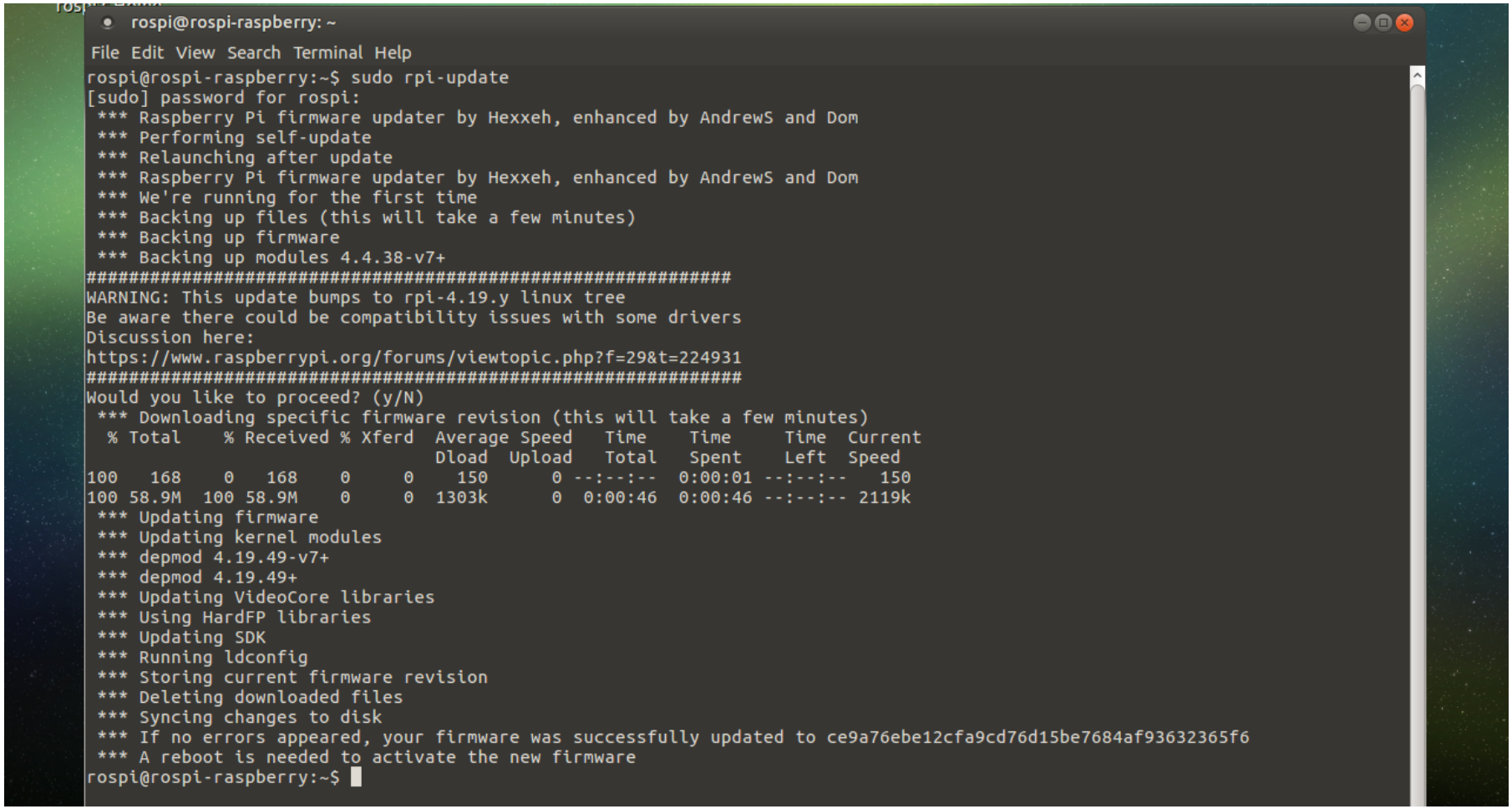
# Raspberry PI initial Setting

- **Root 계정 Password 설정**
  - 터미널 명령어 : sudo passwd root
  - Password 입력이 정상적으로 완료되면 Password updated Successfully 확인



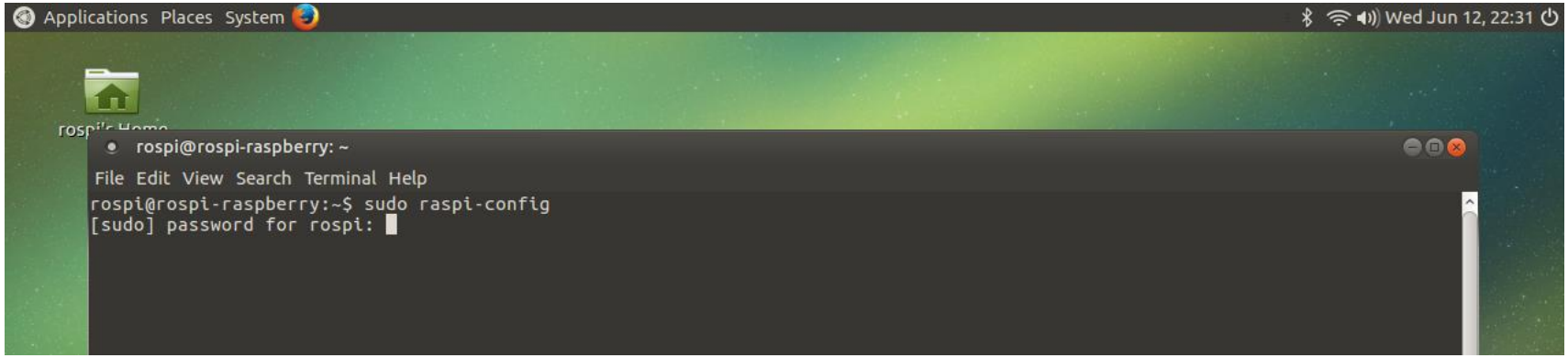
# Raspberry PI initial Setting

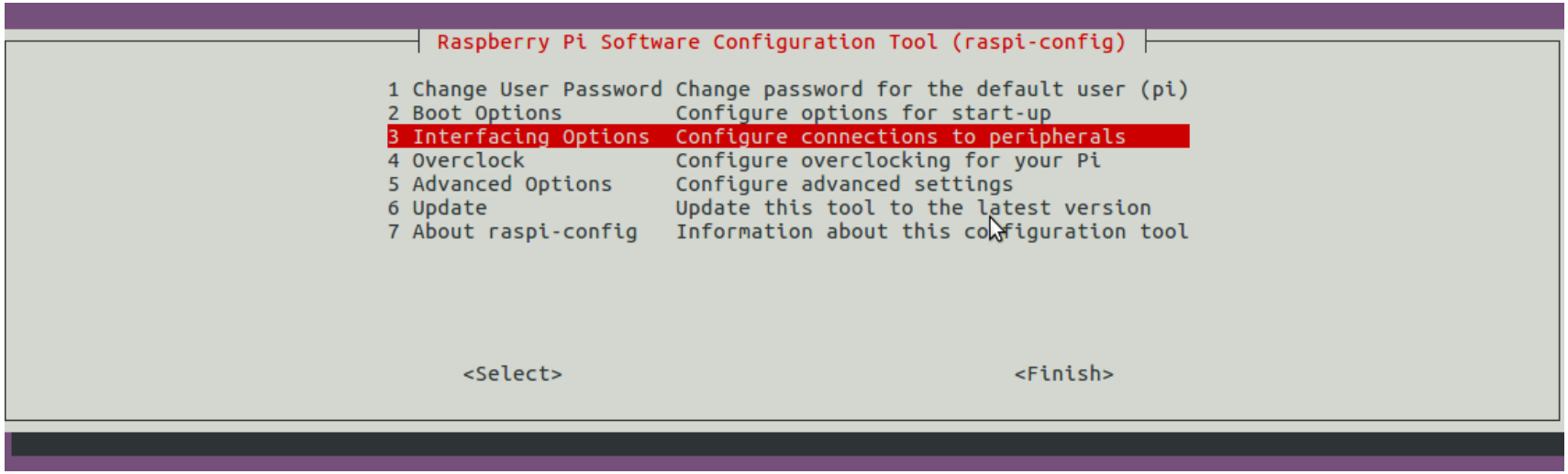
- **Raspberry PI 펌웨어 업데이트**
  - 명령어 : sudo rpi-update



# Raspberry PI initial Setting

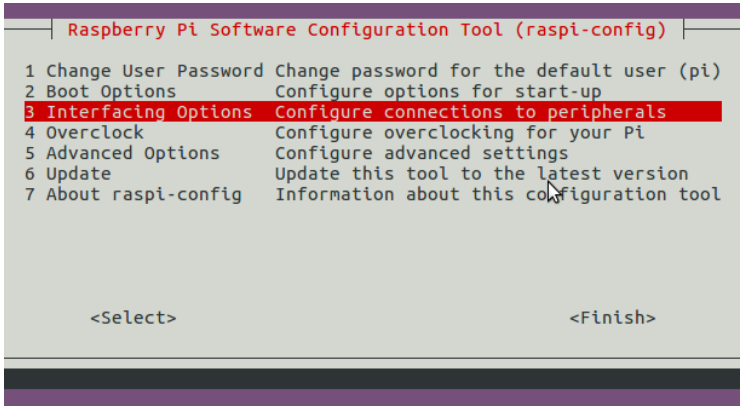
- **Raspberry PI Configuration Tool 설정**
  - Configuration Tool 진입 명령어 : sudo raspi-config
  - Boot 옵션, 입출력 기기사용 활성화, 파일시스템, Update 등 셋팅 가능

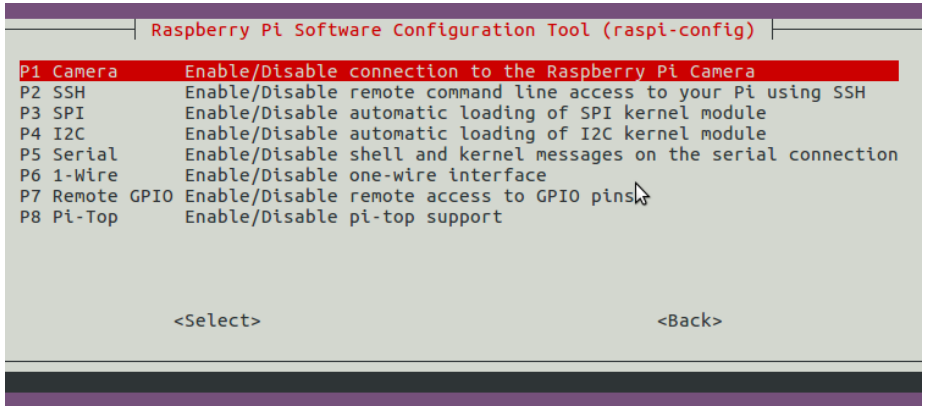


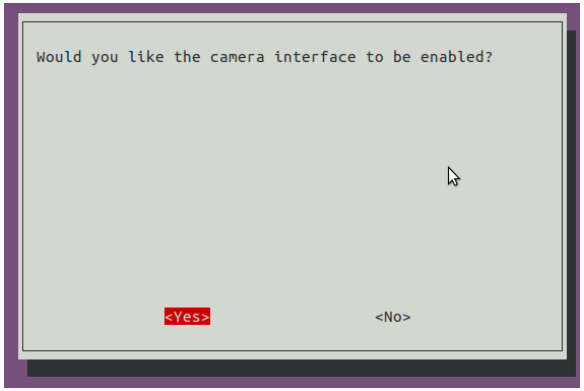



# Raspberry PI initial Setting

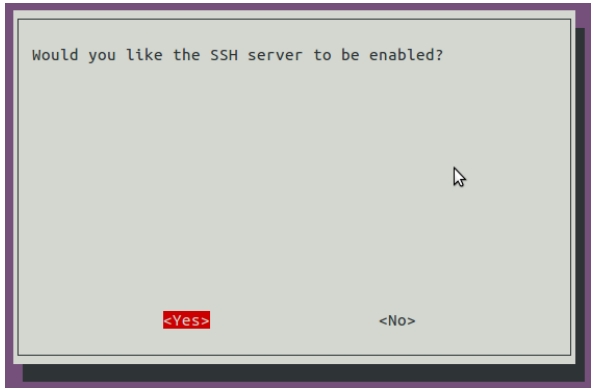
- **Raspberry PI Configuration Tool 설정(Pi Camera & SSH)**
  - Interfacing Options 에서 P1 Camera / P2 SSH 를 Enable 상태로 변경



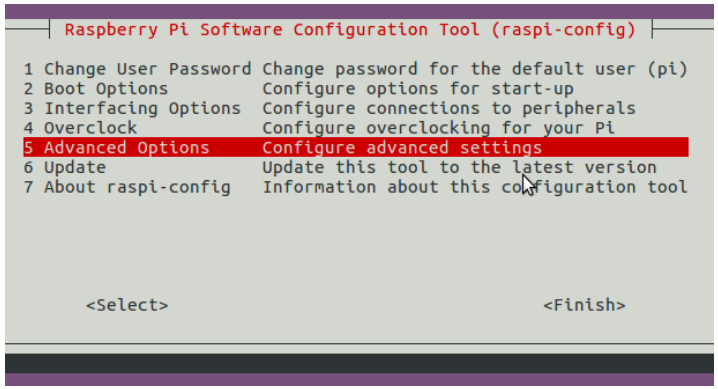



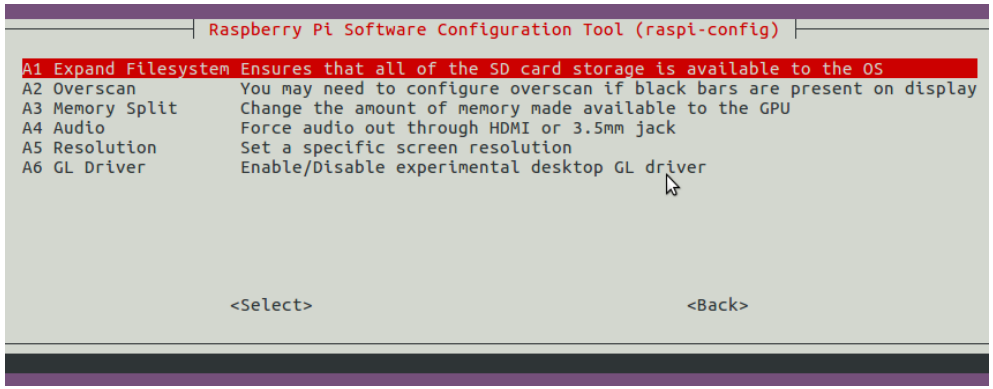

P1 Camera interface를 enable 상태로 변경 후, 위 과정 동일하게 반복. P2 SSH server 기능도 enable 상태로 변경.

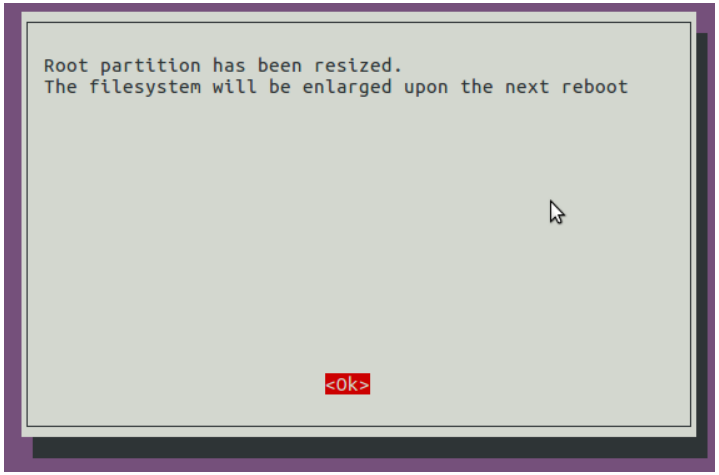


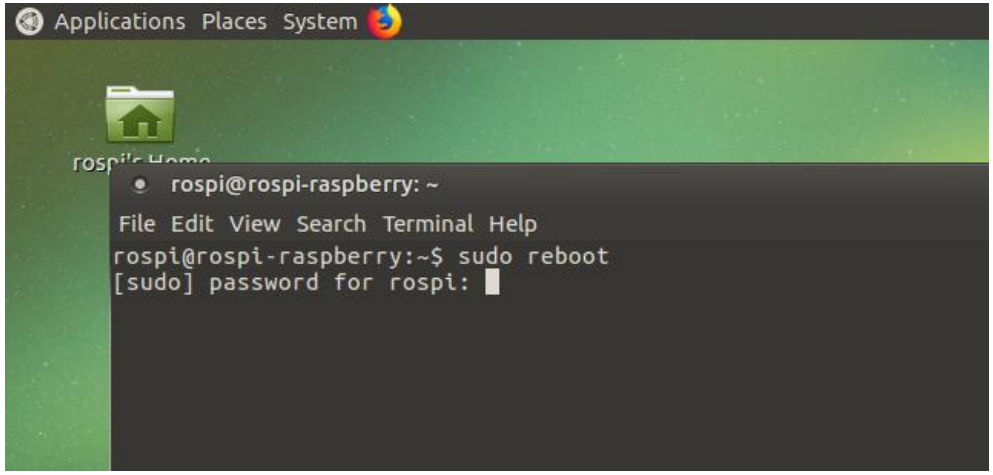
# Raspberry PI initial Setting

- **Raspberry PI Configuration Tool 설정 (Filesystem 확장)**
  - Advanced Option에서 A1 Expand Filesystem 선택
  - Root Partition resized 후 시스템 재시작 (Reboot 명령어 : sudo reboot)



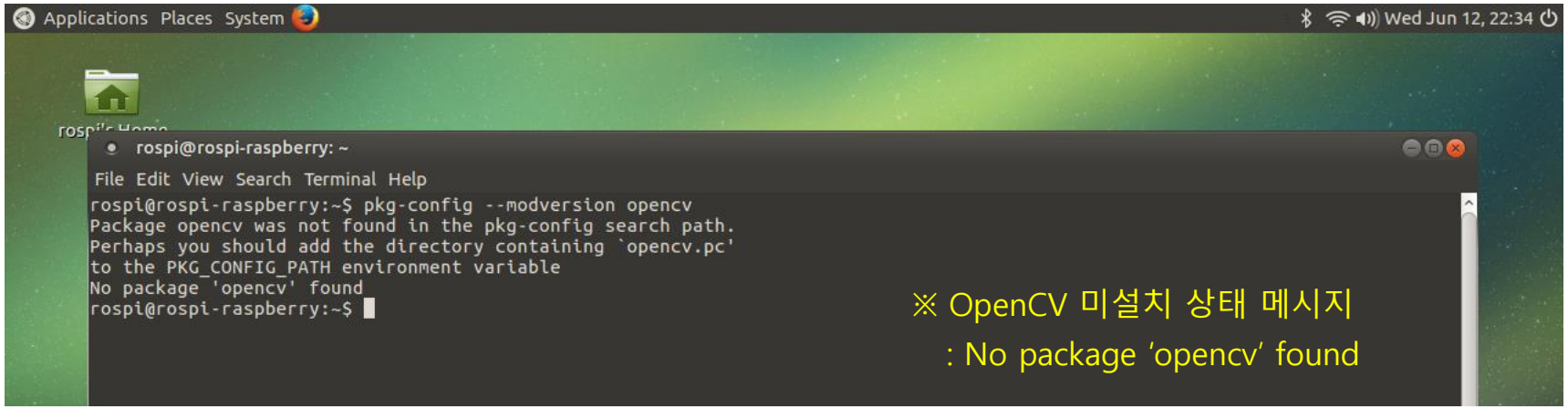



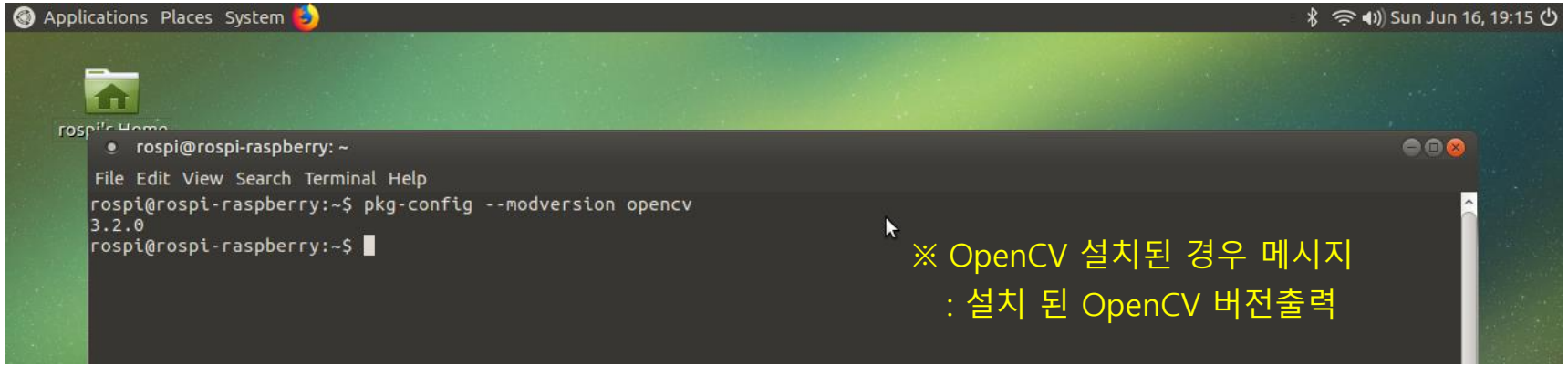



# Check if OpenCV is installed

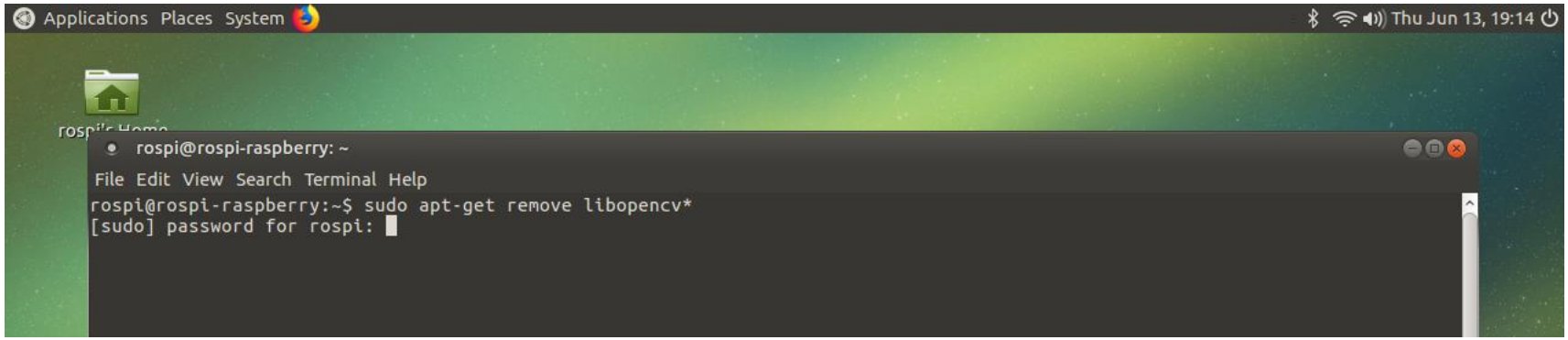
- **OpenCV 설치유무 확인**
  - 명령어 : pkg-config --modversion opencv



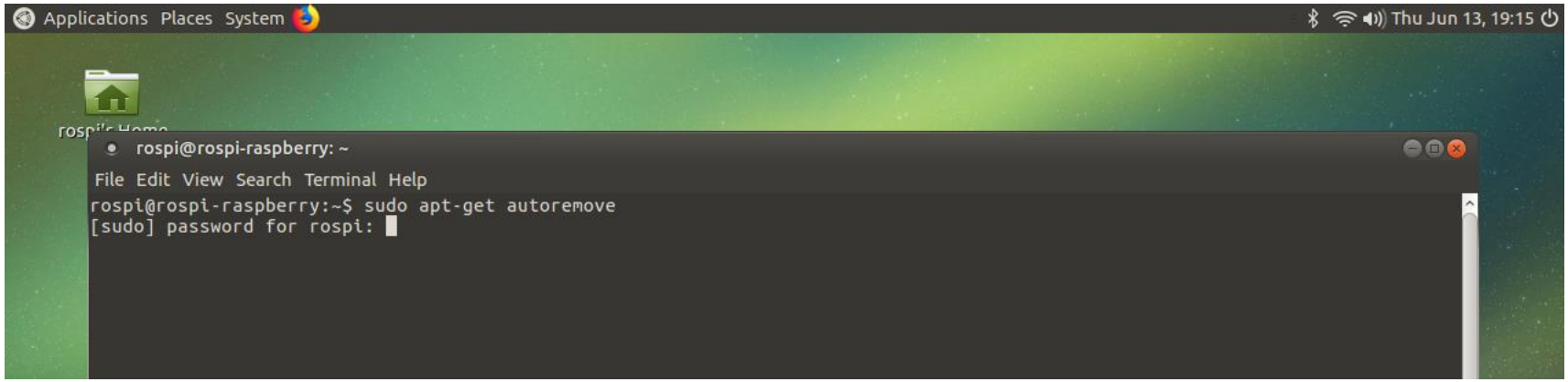



# Remove existing Opencv

- **타 버전 OpenCV & 관련 library 삭제**
  - OpenCV관련 패키지 삭제 - 명령어 : sudo apt-get remove libopencv*



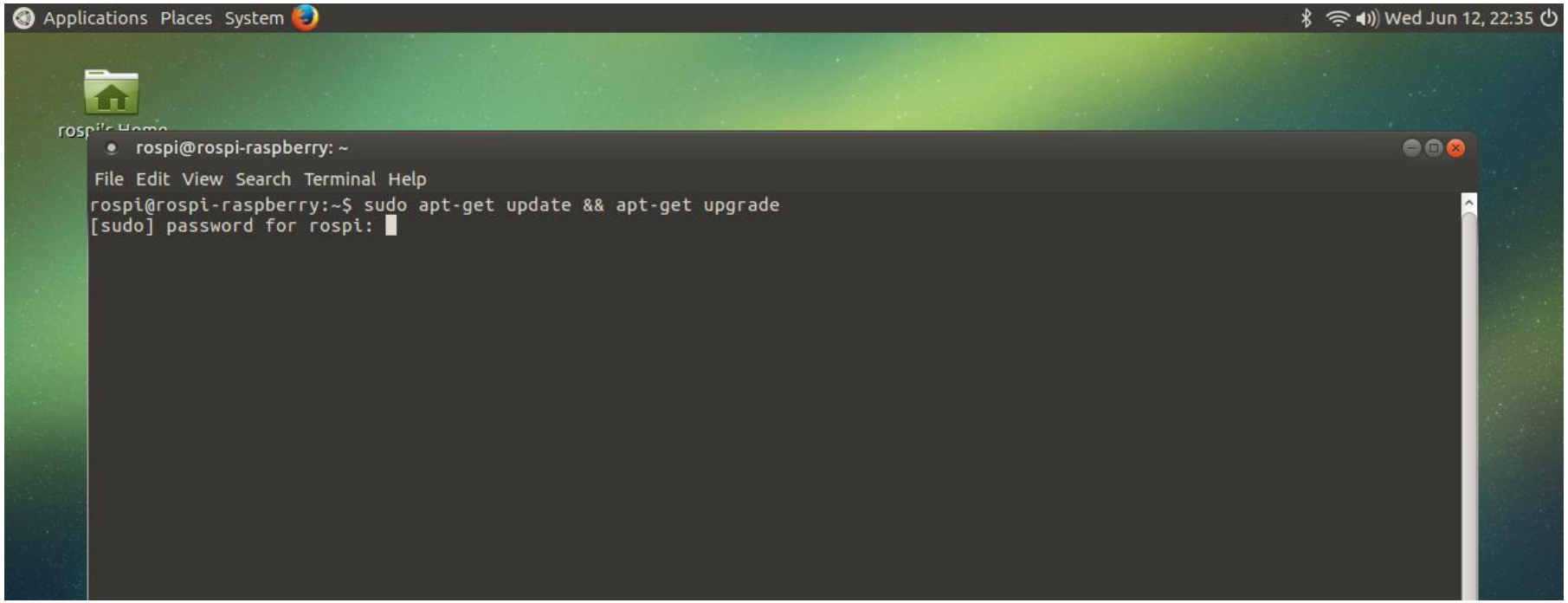

  - 더 이상 OS에서 사용하지 않는 패키지 삭제 - 명령어 : sudo apt-get auroremove



# Various Library installation For OpenCV

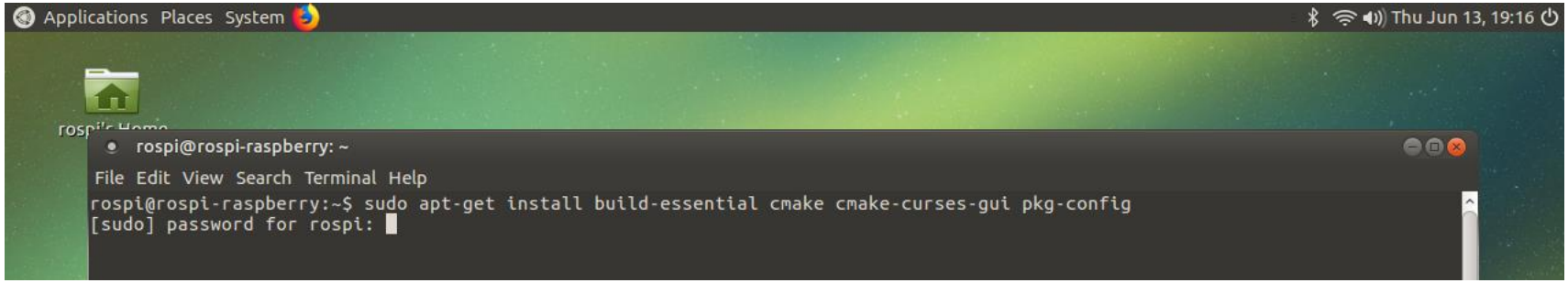
- **Library 설치 전 System update & upgrade**
  - Apt server Update 및 패키지 Upgrade

    -명령어 : sudo apt-get update && apt-get upgrade

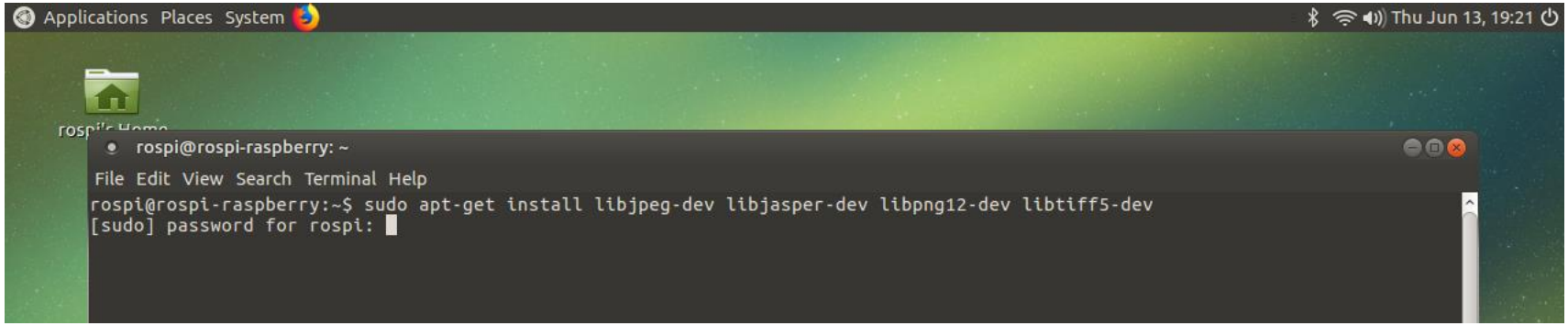


# Various Library installation For OpenCV

- **개발자 도구 설치**
  - 명령어 : sudo apt-get install build-essential cmake cmake-curses-gui pkg-config

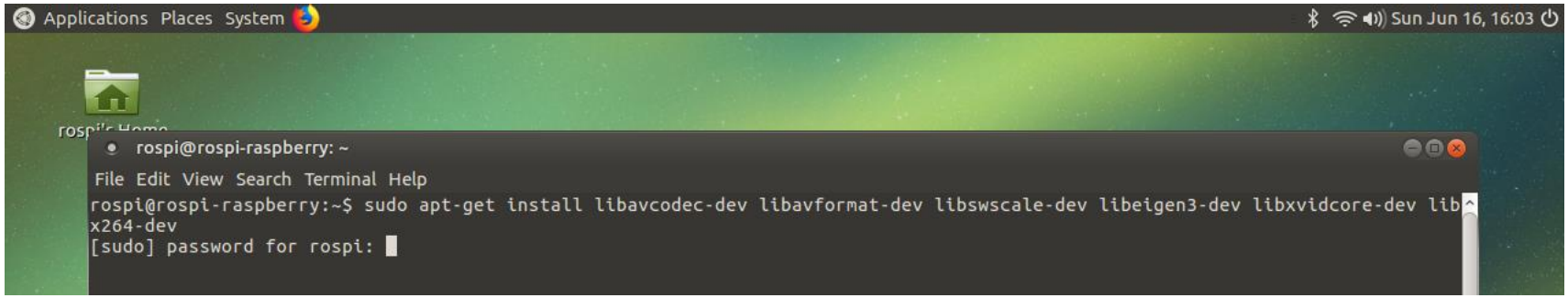



- **JPEG, PNG, TIFF 파일 형식 사용을 위한 이미지 처리 I/O 패키지 설치**
  - 명령어 : sudo apt-get install libjpeg-dev libjasper-dev libpng12-dev libtiff5-dev

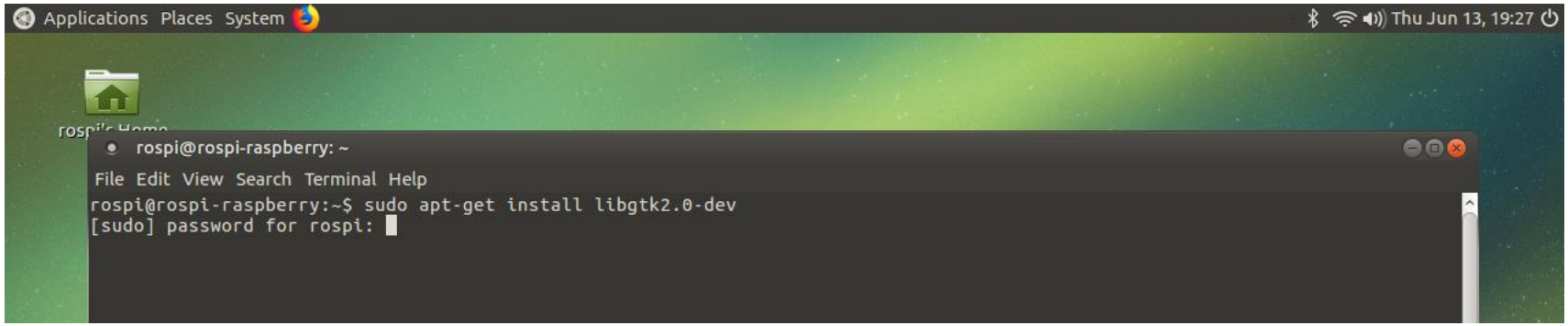


# Various Library installation For OpenCV

- **카메라 Stream 작업이 가능하도록 Video I/O 패키지 설치**
  - 명령어 : sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libeigen3-dev libxvidcore-dev libx264-dev



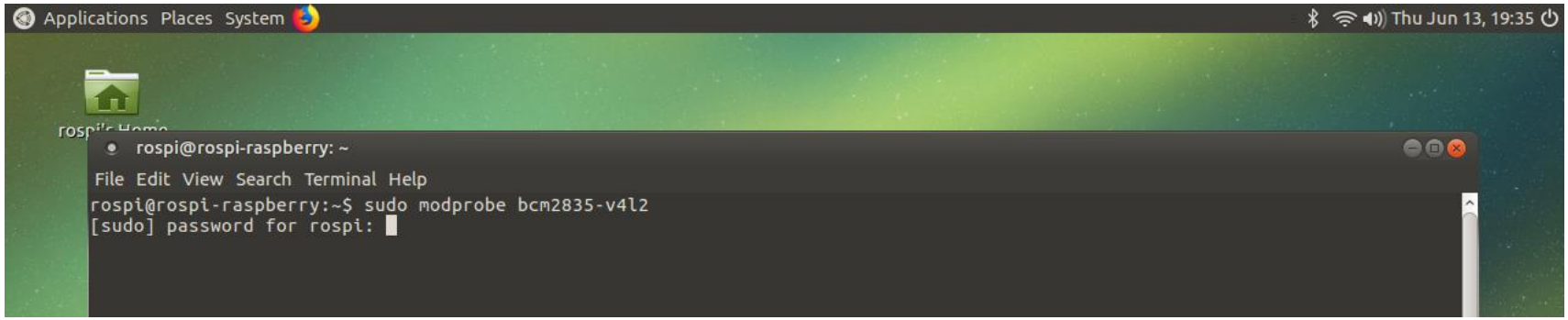

- **OpenCV GUI 프로세싱을 위한 GTK 라이브러리 패키지 설치**
  - 명령어 : sudo apt-get install libgtk2.0-dev



# Various Dependency installation For OpenCV

- **실시간 Video Capture 지원 V4L 라이브러리 및 Utility 패키지 설치**
  - 명령어 : sudo apt-get -y install libv4l-dev v4l-utils


- **Linux Kernel module로 V4L 패키지 enable**
  - 명령어 : sudo modprobe bcm2835-v4l2



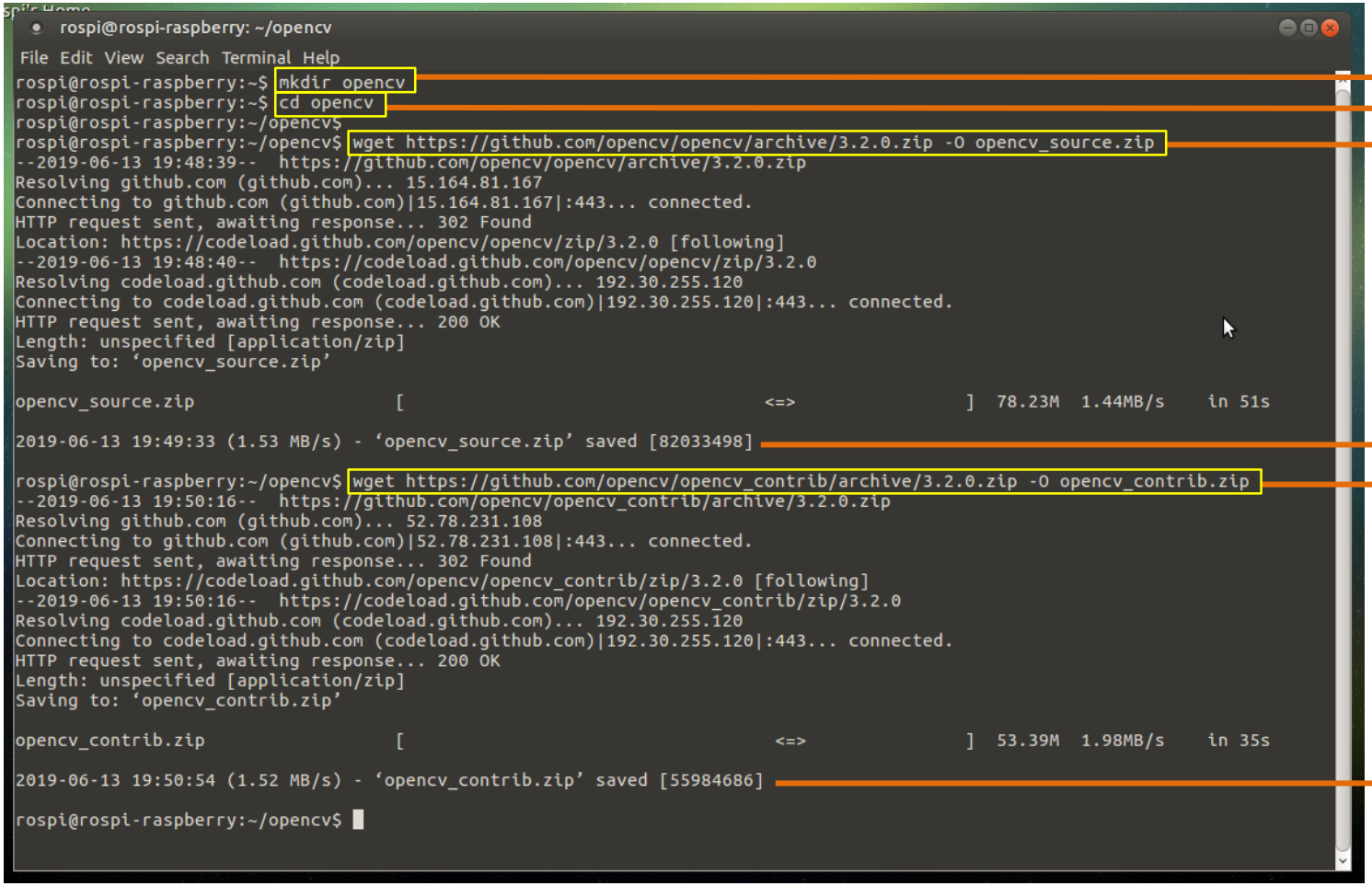
# Various Dependency installation For OpenCV

- **OpenCV 기능 최적화 라이브러리 패키지 설치**
  - 명령어 : sudo apt-get install libatlas-base-dev gfortran


- **Python 개발자 도구와 Numpy 라이브러리 설치**
  - 명령어 : sudo apt-get install python2.7-dev python-numpy python3-dev python3-numpy

# OpenCV install

- **OpenCV & OpenCV contrib 다운로드**
  - Home/user/ 디렉토리 내 opencv 폴더 생성 및 설치파일 다운로드




① opencv 폴더 생성
② opencv 폴더 진입
③ OpenCV source 파일 다운로드
다운로드 완료확인
④ OpenCV contrib 파일 다운로드
다운로드 완료확인

# OpenCV install

- **OpenCV & OpenCV contrib 다운로드 & 압축풀기**
  - 현재 경로 확인

    - pwd 명령어 사용하여 "/home/계정명" 경로 확인 후, 해당 경로에 폴더 생성 할 것.
  - 폴더 생성 명령어

    ~$ mkdir opencv
  - 폴더 진입 명령어

    ~$ cd opencv
  - OpenCV Source 파일 다운로드 명령어

    ~/opencv$ wget  https://github.com/opencv/opencv/archive/3.2.0.zip -O opencv_source.zip
  - OpenCV Contrib 파일 다운로드 명령어

    ~/opencv$ wget https://github.com/opencv/opencv_contrib/archive/3.2.0.zip -O opencv_contrib.zip
  - 다운로드 받은 파일 압축풀기 명령어

    ~/opencv$ unzip opencv_source.zip

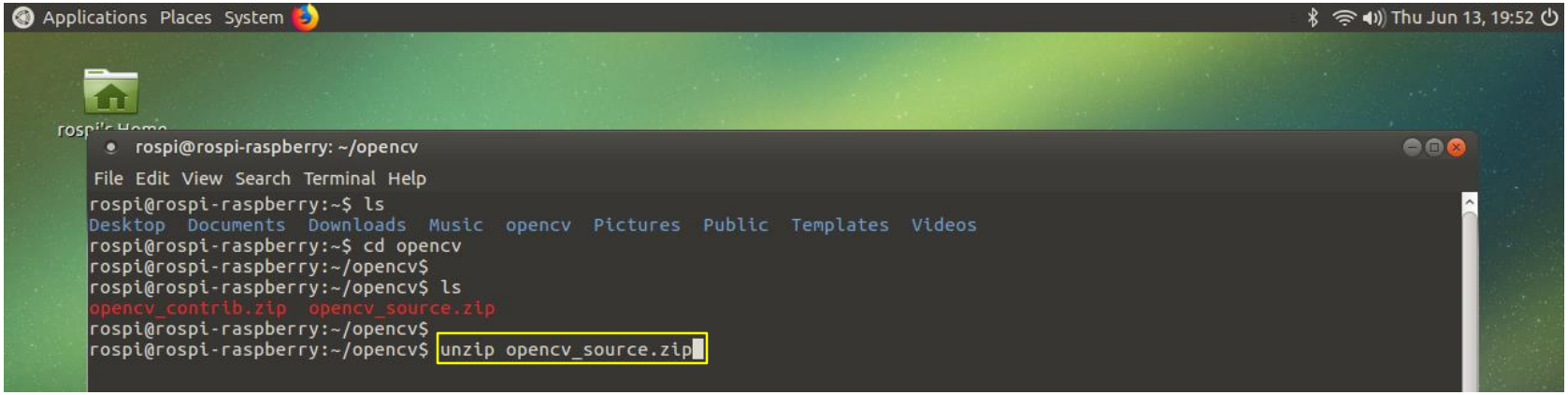
    ~/opencv$ unzip opencv_contrib.zip

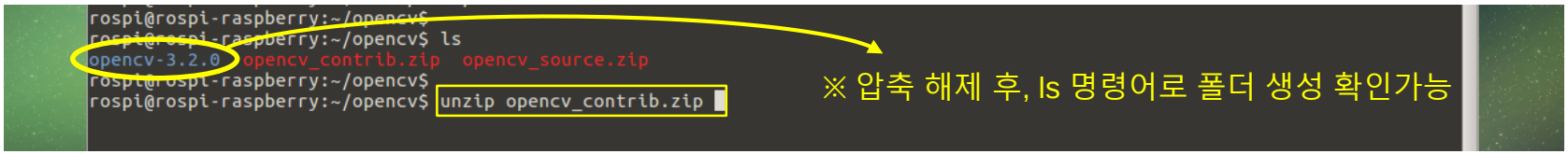
# OpenCV install

- **OpenCV & OpenCV contrib 압축풀기**
  - 다운로드 받은 파일 압축풀기 명령어

~/opencv$ unzip opencv_source.zip

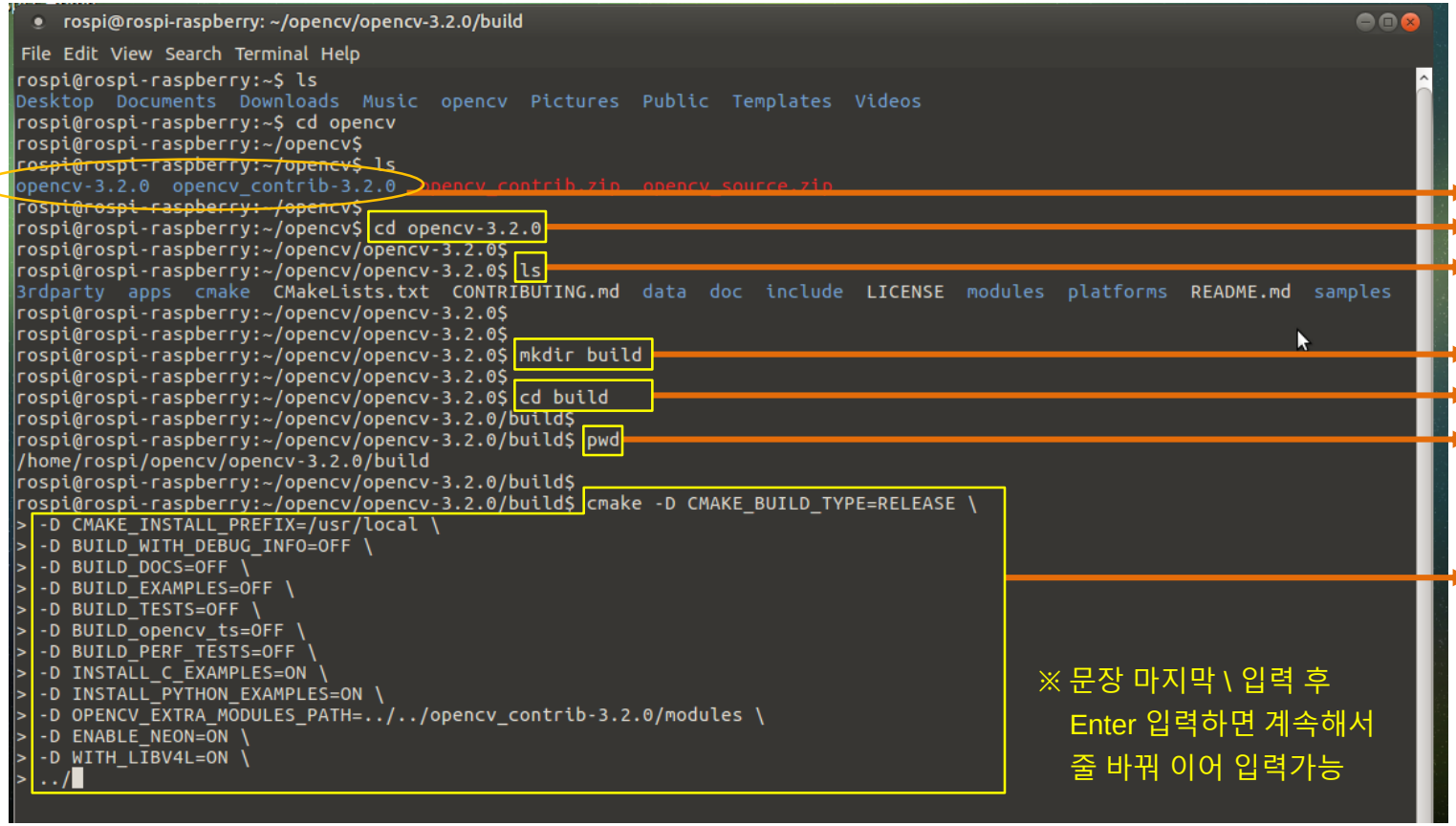
~/opencv$ unzip opencv_contrib.zip



# OpenCV install

- **OpenCV Build 준비**
  - Build 폴더 생성 및 Build 구성 입력



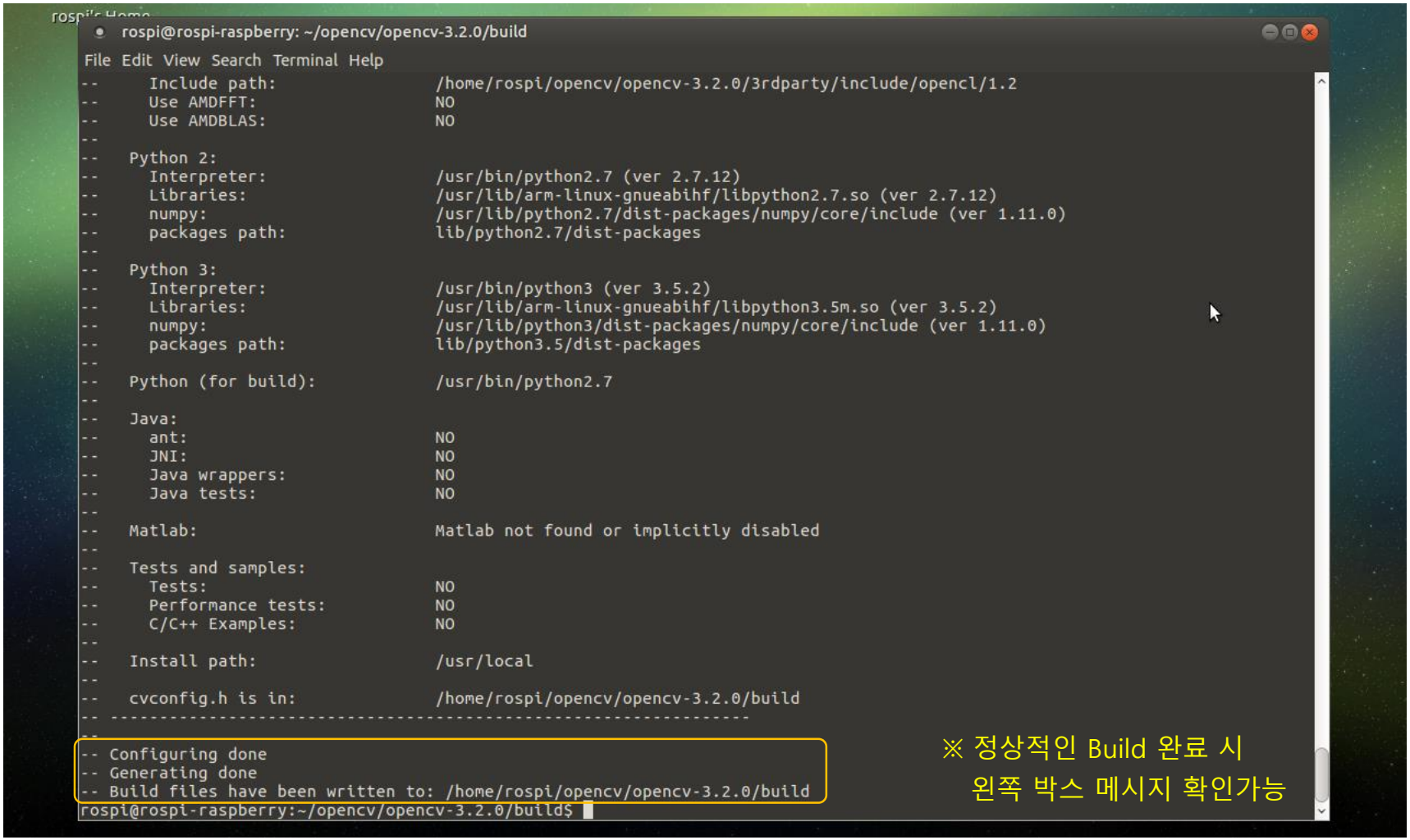
# OpenCV install

- **OpenCV Build 준비**
  - Build 폴더 생성 명령어 : ~/opencv$ mkdir build
  - 폴더 진입 명령어 : ~/opencv$ cd build
  - 현재 경로확인 명령어 : ~/opencv$ pwd
  - Build 구성 입력 명령어 :

```
~/opencv$ cmake -D CMAKE_BUILD_TYPE=RELEASE \
-D CMAKE_INSTALL_PREFIX=/usr/local \
-D BUILD_WITH_DEBUG_INFO=OFF \
-D BUILD_DOCS=OFF \
-D BUILD_EXAMPLES=OFF \
-D BUILD_TESTS=OFF \
-D BUILD_opencv_ts=OFF \
-D BUILD_PERF_TESTS=OFF \
-D INSTALL_C_EXAMPLES=ON \
-D INSTALL_PYTHON_EXAMPLES=ON \
-D OPENCV_EXTRA_MODULES_PATH=../../opencv_contrib-3.2.0/modules \
-D ENABLE_NEON=ON \
-D WITH_LIBV4L=ON \
../
```

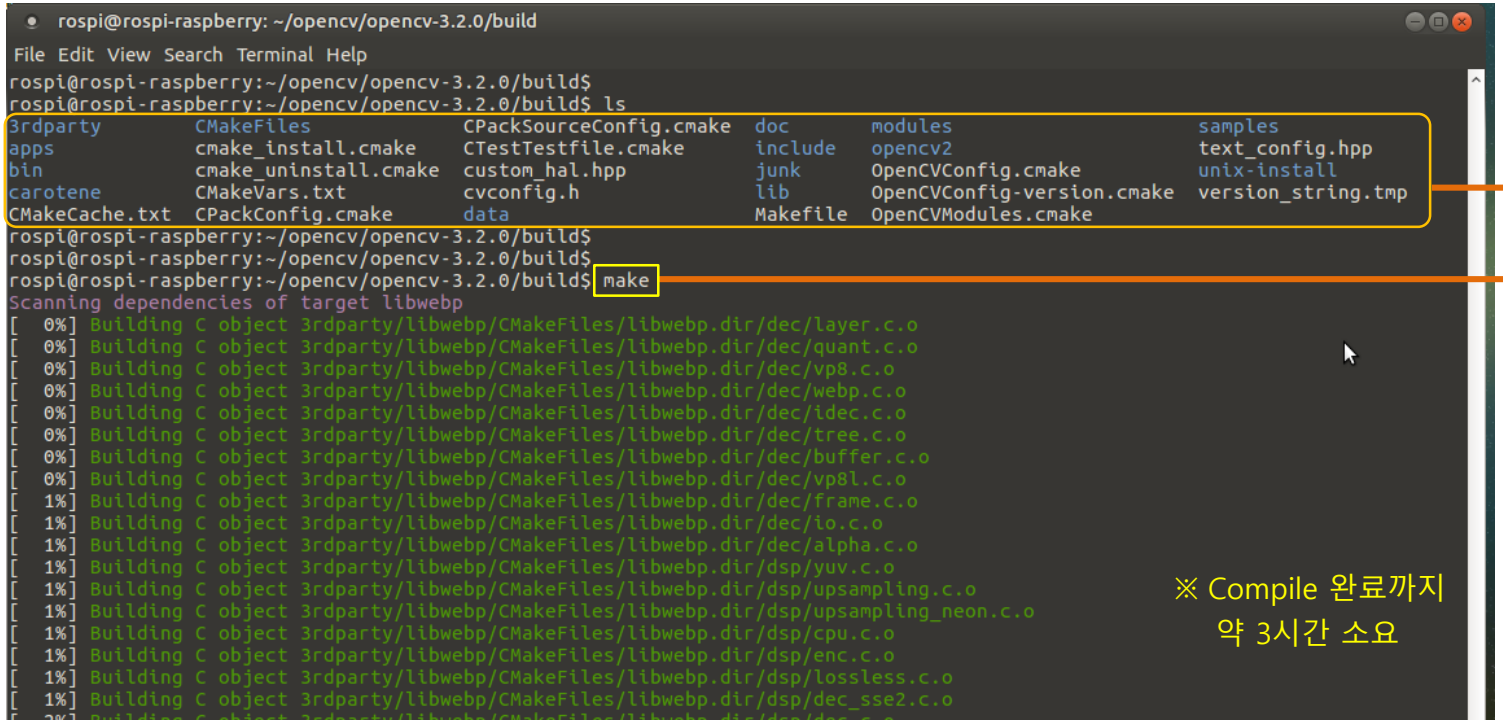
# OpenCV install

- **OpenCV Build 완료**



# OpenCV install

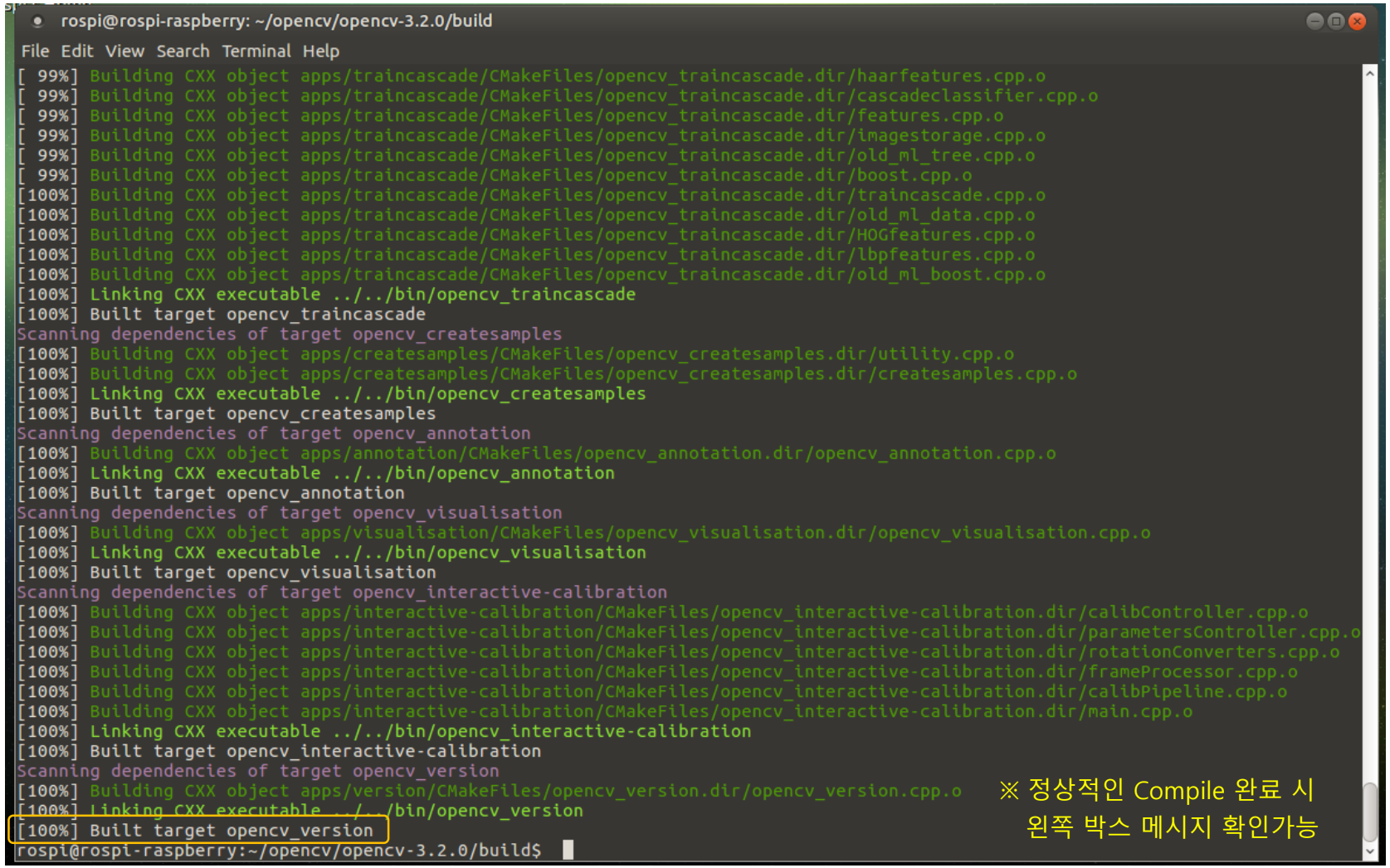
- **OpenCV Compile 시작**
  - Compile 명령어 : make
  - /home/계정명/opencv/opencv-3.2.0/build 디렉토리 내에서 실행




※ 1GB로 Swap Memory 공간 확장 후, make -j4 명령어 사용 시에는 Compile 시간 단축 가능
그러나, Ubuntu mate에서는 Allocate memory Error 발생으로 make 명령어 사용을 추천함.

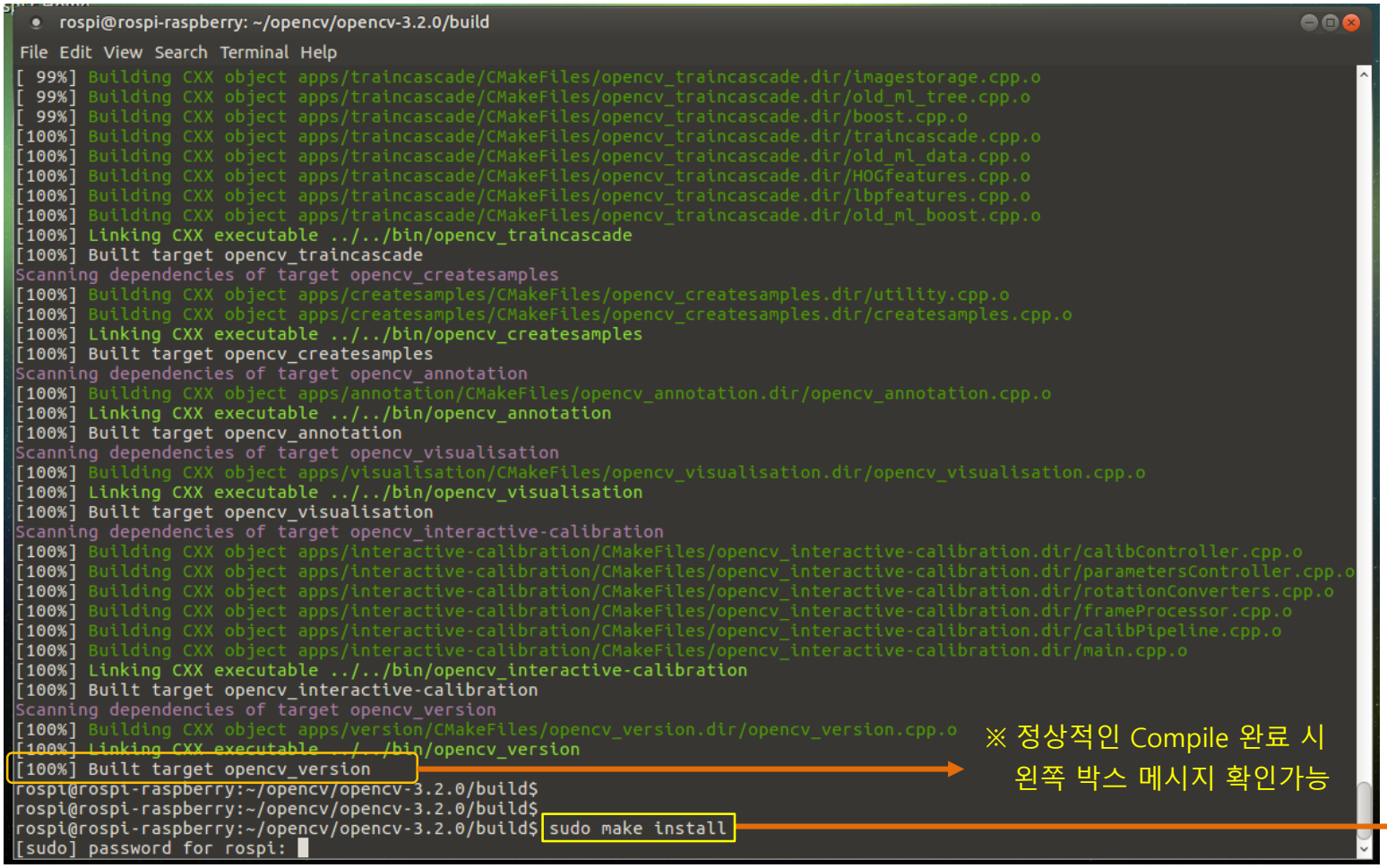
# OpenCV install

- **OpenCV Compile 완료**



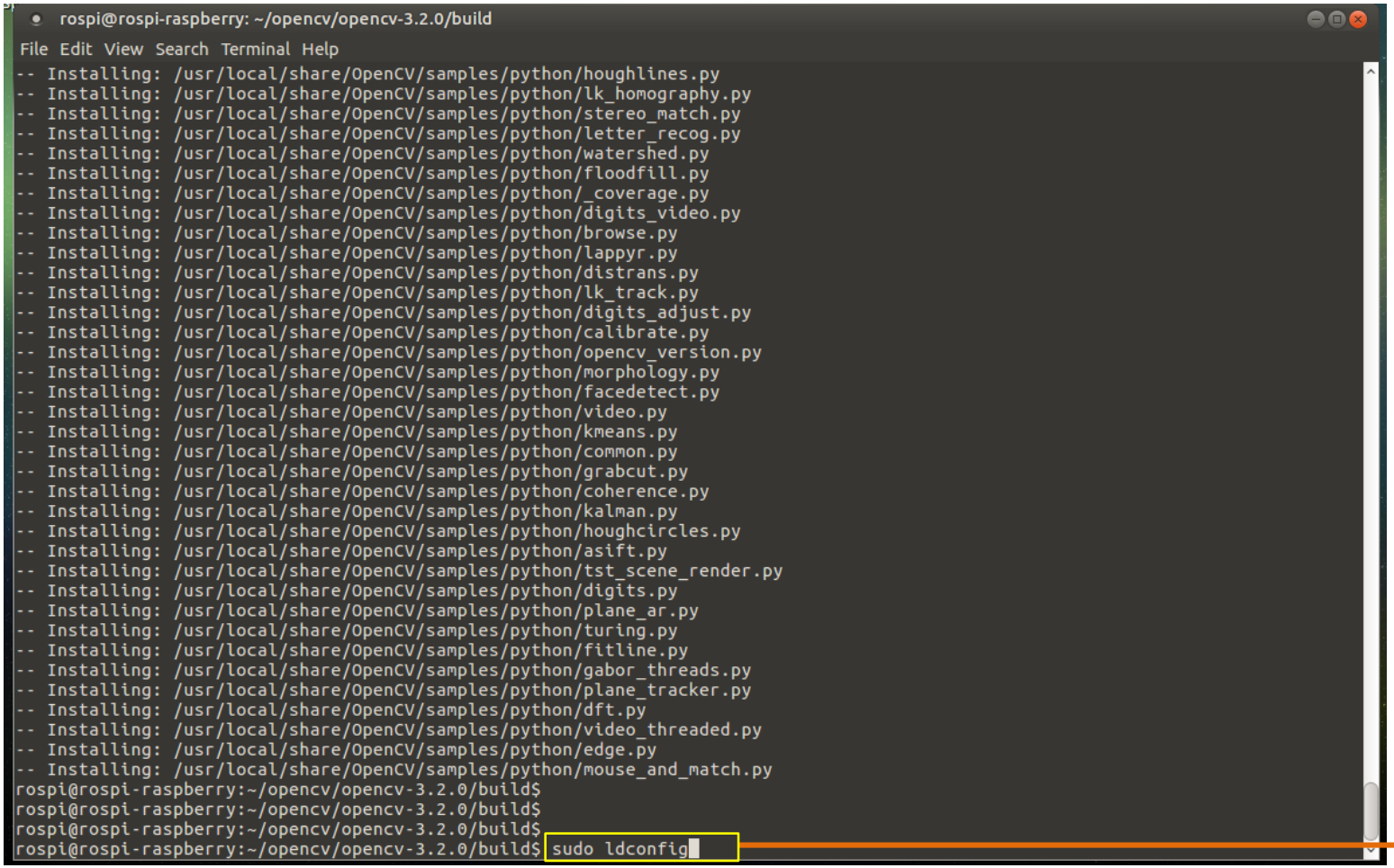
# OpenCV install

- **OpenCV Compile 완료 확인 및 Make install**
  - 명령어 : sudo make install



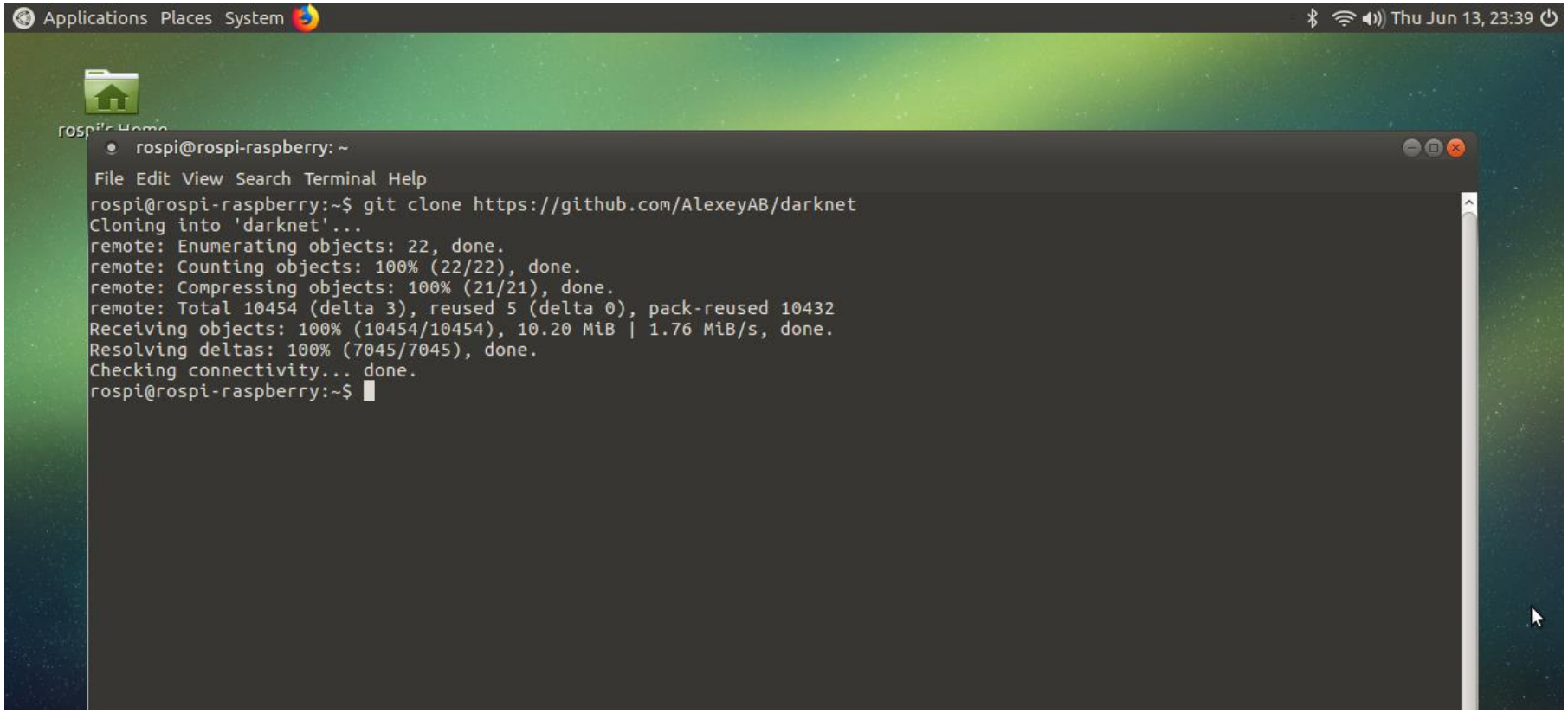
# OpenCV install

- **OpenCV install 완료 확인 및 cache 갱신**
  - 명령어 : sudo ldconfig



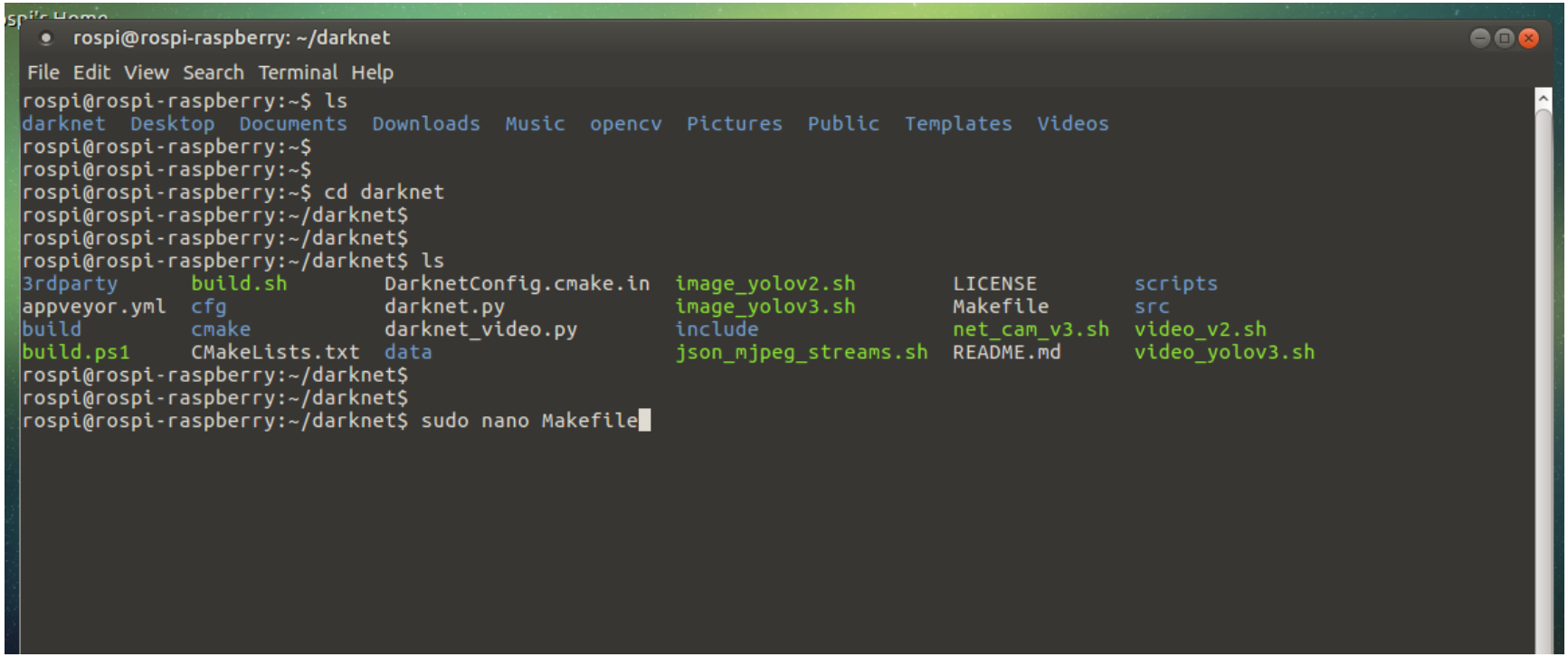
# Darknet install

- **Darknet download**
  - 명령어 : git clone https://github.com/AlexeyAB/darknet
  - Git 패키지가 설치되어 있지 않다면, sudo apt install git 명령어 사용하여 설치



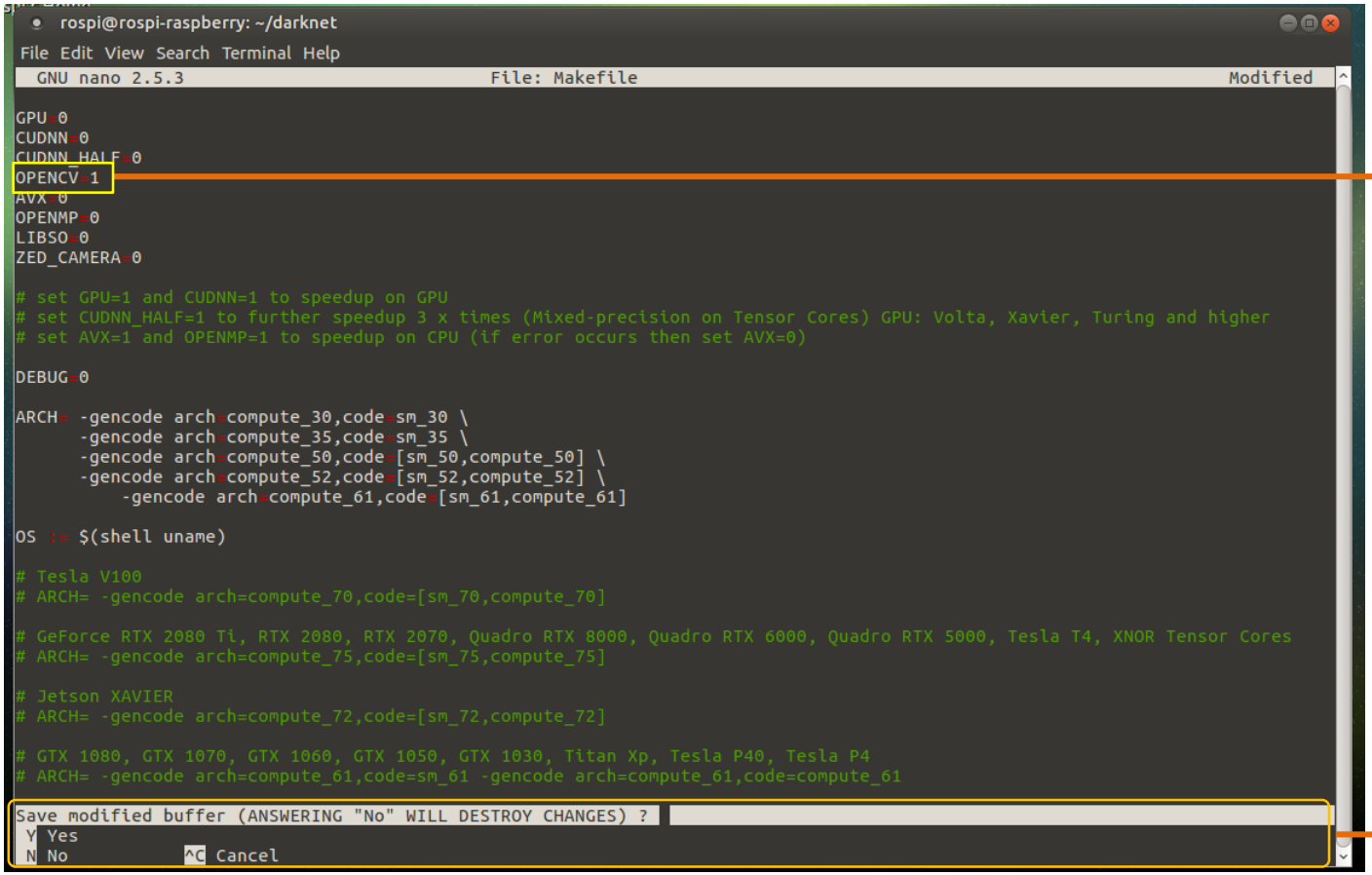
# Darknet install

- **Makefile 수정**
  - Darknet 폴더 진입 후, ls 명령어로 파일 다운로드 정상여부 확인
  - Nano 편집기 사용 Makefile 열기 명령어 : sudo nano Makefile
  - Nano 패키지가 설치되어 있지 않다면, sudo apt install nano 명령어 사용 설치



# Darknet install

- **Makefile 수정**
  - Makefile 내 5-Line의 OPENCV=0을 OPENCV=1로 화면과 같이 수정
  - 수정 후, Ctrl + x 단축키 사용 y를 눌러 저장하고 터미널로 돌아간다.

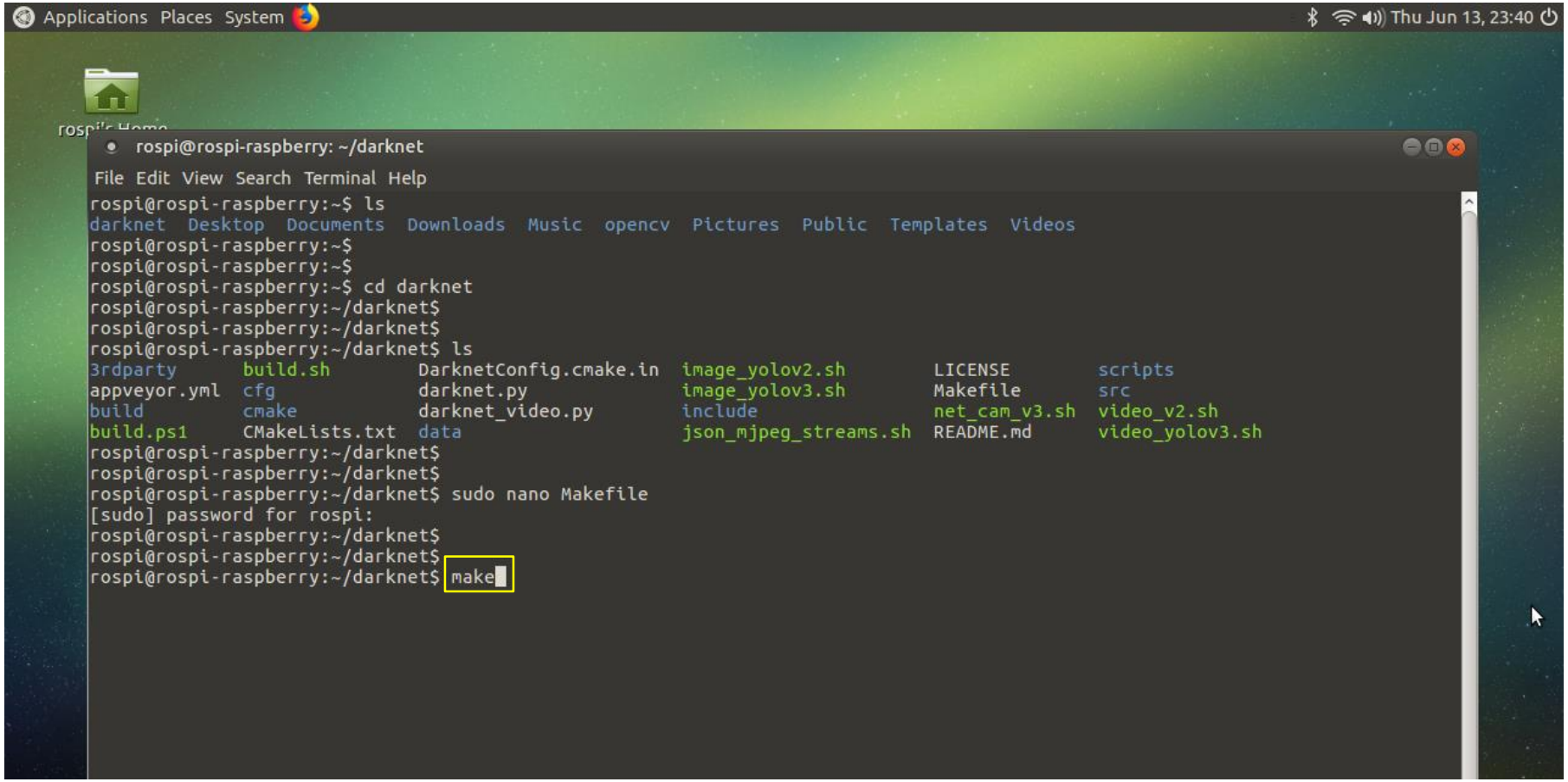



① OPENCV = 0 을 OPENCV = 1 로 수정

② 파일 저장 후 종료

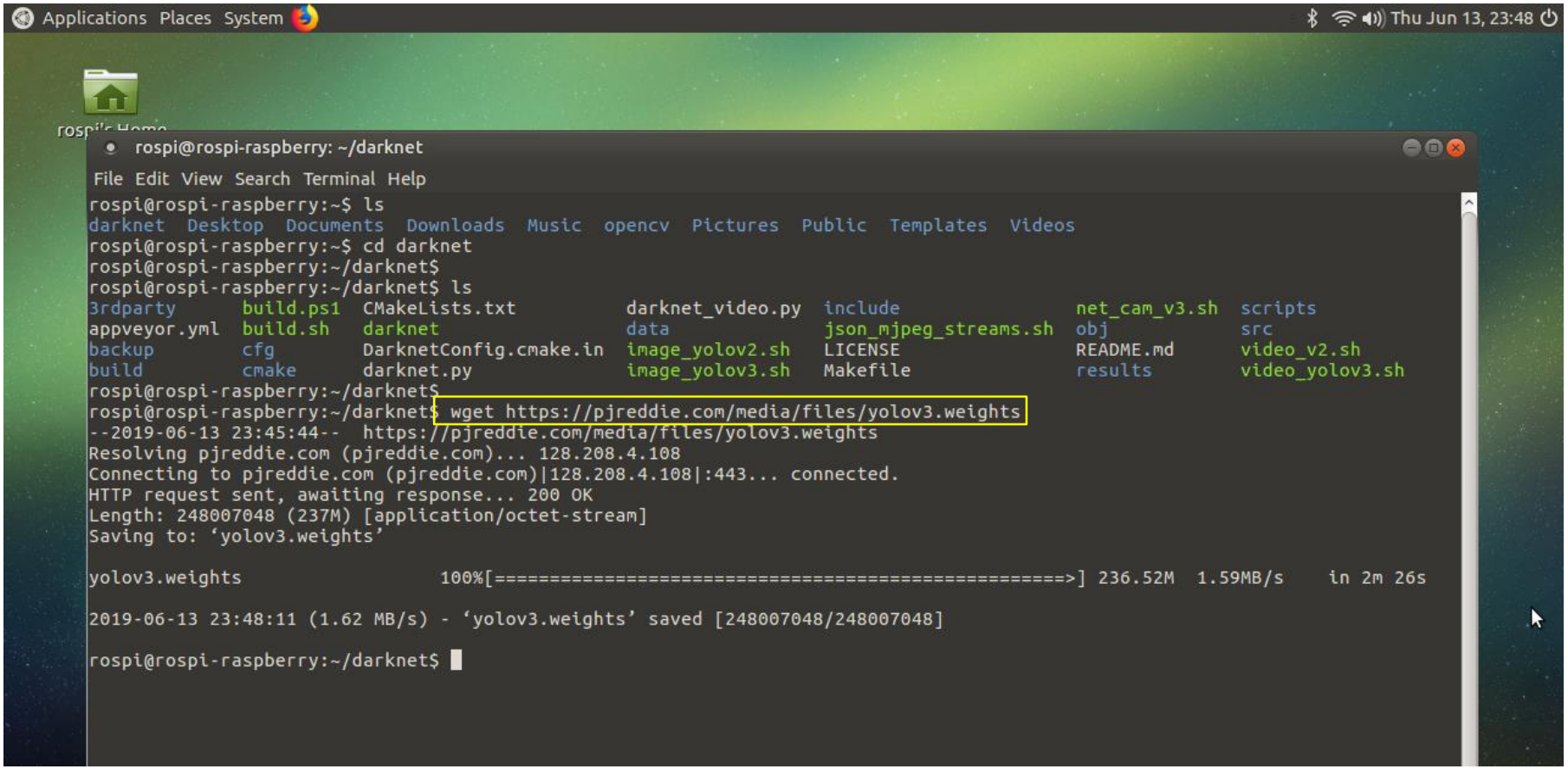
# Darknet install

- **Darknet Make install**
  - Darknet 디렉토리 내에서 명령어 : make



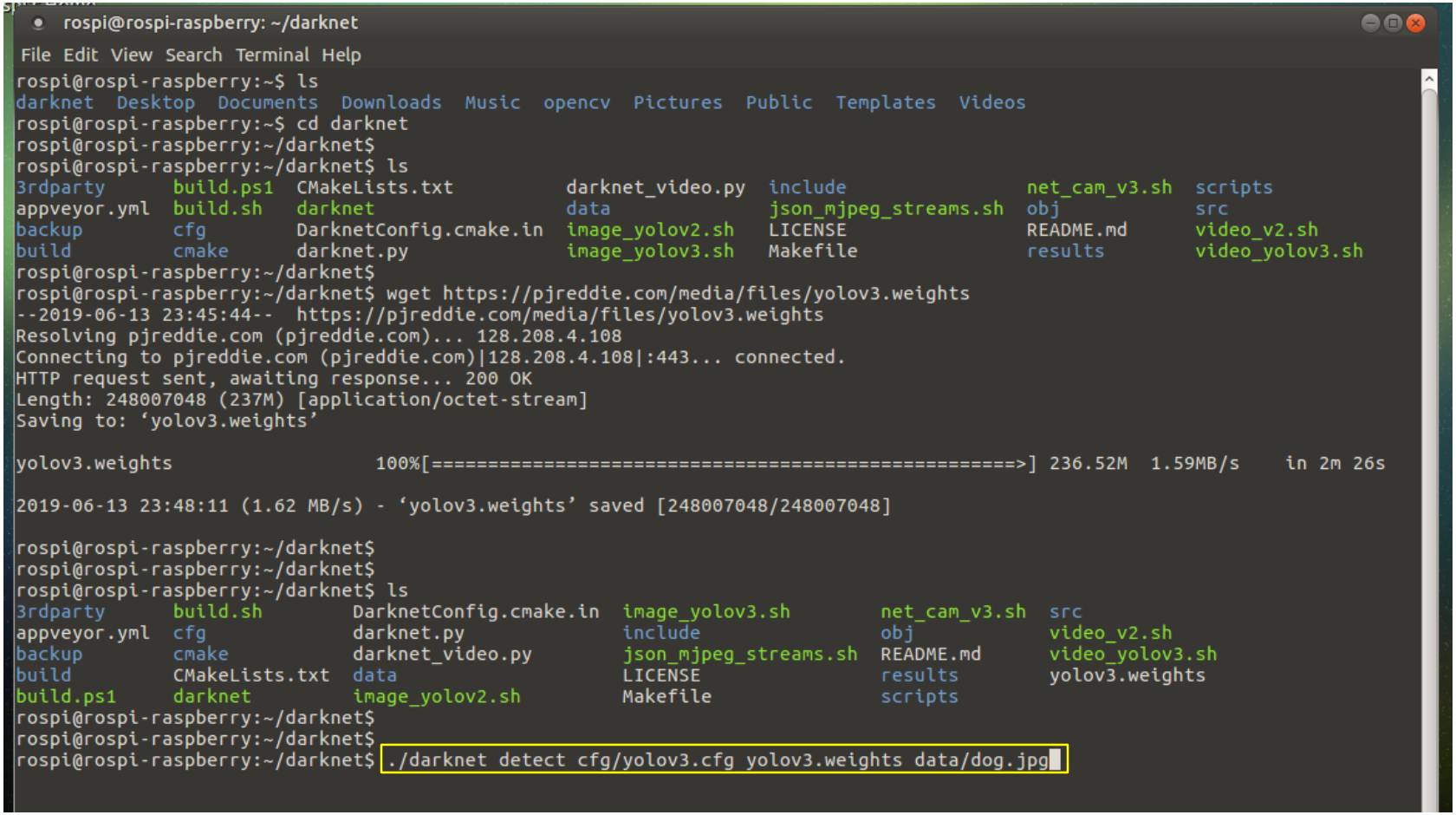
# Run YOLO v3

- **YOLO v3 Weight file Download**
  - Darknet 디렉토리 내에서 명령어 : wget https://pjreddie.com/media/files/yolov3.weights



# Run YOLO v3

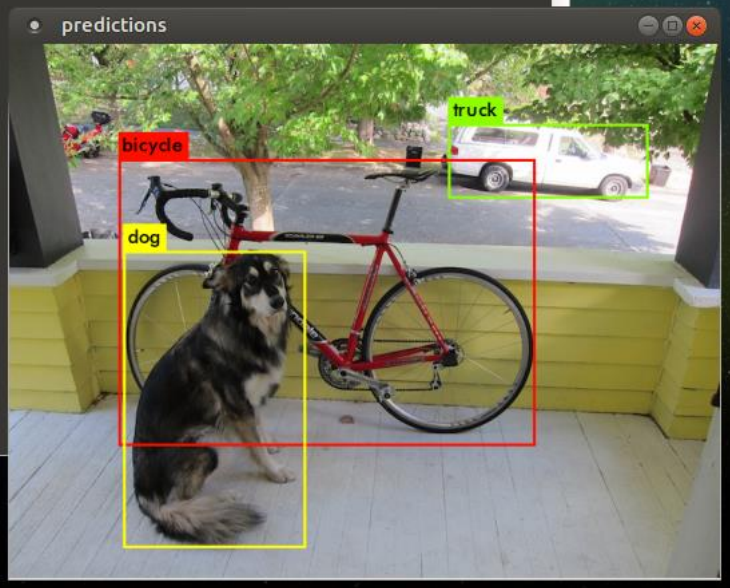
- **YOLO v3 단일 이미지 Object-Detection 실행**
  - 명령어 : ./darknet detect cfg/yolov3.cfg yolov3.weights data/dog.jpg



# Run YOLO v3

- **YOLO v3 단일 이미지 Object-Detection 실행**
  - Processing 완료 및 Object Detection 여부 확인

```
  89 conv    256  1 x 1 / 1(1)  26 x  26 x 512   ->    26 x  26 x 256 0.177 BF
  90 conv    512  3 x 3 / 1(1)  26 x  26 x 256   ->    26 x  26 x 512 1.595 BF
  91 conv    256  1 x 1 / 1(1)  26 x  26 x 512   ->    26 x  26 x 256 0.177 BF
  92 conv    512  3 x 3 / 1(1)  26 x  26 x 256   ->    26 x  26 x 512 1.595 BF
  93 conv    255  1 x 1 / 1(1)  26 x  26 x 512   ->    26 x  26 x 255 0.177 BF
  94 yolo
[yolo] params: iou loss: mse, iou_norm: 0.75, cls_norm: 1.00, scale_x_y: 1.00
  95 route  91
  96 conv    128  1 x 1 / 1(1)  26 x  26 x 256   ->    26 x  26 x 128 0.044 BF
  97 upsample            2x    26 x  26 x 128   ->    52 x  52 x 128
  98 route  97 36
  99 conv    128  1 x 1 / 1(1)  52 x  52 x 384   ->    52 x  52 x 128 0.266 BF
 100 conv    256  3 x 3 / 1(1)  52 x  52 x 128   ->    52 x  52 x 256 1.595 BF
 101 conv    128  1 x 1 / 1(1)  52 x  52 x 256   ->    52 x  52 x 128 0.177 BF
 102 conv    256  3 x 3 / 1(1)  52 x  52 x 128   ->    52 x  52 x 256 1.595 BF
 103 conv    128  1 x 1 / 1(1)  52 x  52 x 256   ->    52 x  52 x 128 0.177 BF
 104 conv    256  3 x 3 / 1(1)  52 x  52 x 128   ->    52 x  52 x 256 1.595 BF
 105 conv    255  1 x 1 / 1(1)  52 x  52 x 256   ->    52 x  52 x 255 0.353 BF
 106 yolo
[yolo] params: iou loss: mse, iou_norm: 0.75, cls_norm: 1.00, scale_x_y: 1.00
Total BFLOPS 65.864
Loading weights from yolov3.weights...
 seen 64
Done!
data/dog.jpg: Predicted in 698240.288000 milli-seconds.
bicycle: 99%
dog: 100%
truck: 93%
```




- **YOLO_v3 PI Camera를 이용한 Object Detection 실행**
  - 명령어 : ./darknet detector demo cfg/coco.data cfg/yolov3.cfg yolov3.weights -c 0