

만일 실험을 마치려면, 모든 terminal은 CTL+C로 종단 후에 종료
Turtlebot은 sudo shutdown now로 종료

ROS Programming (SLAM)

(file #5 / ?) ver1.0 Noetic

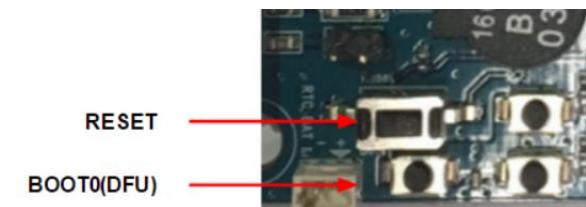
Battery 사전에 충전하기

(미충전 경우 Buzzer 소리 발생)

→ 방치 시 완전 방전됨



Yongseok Chi



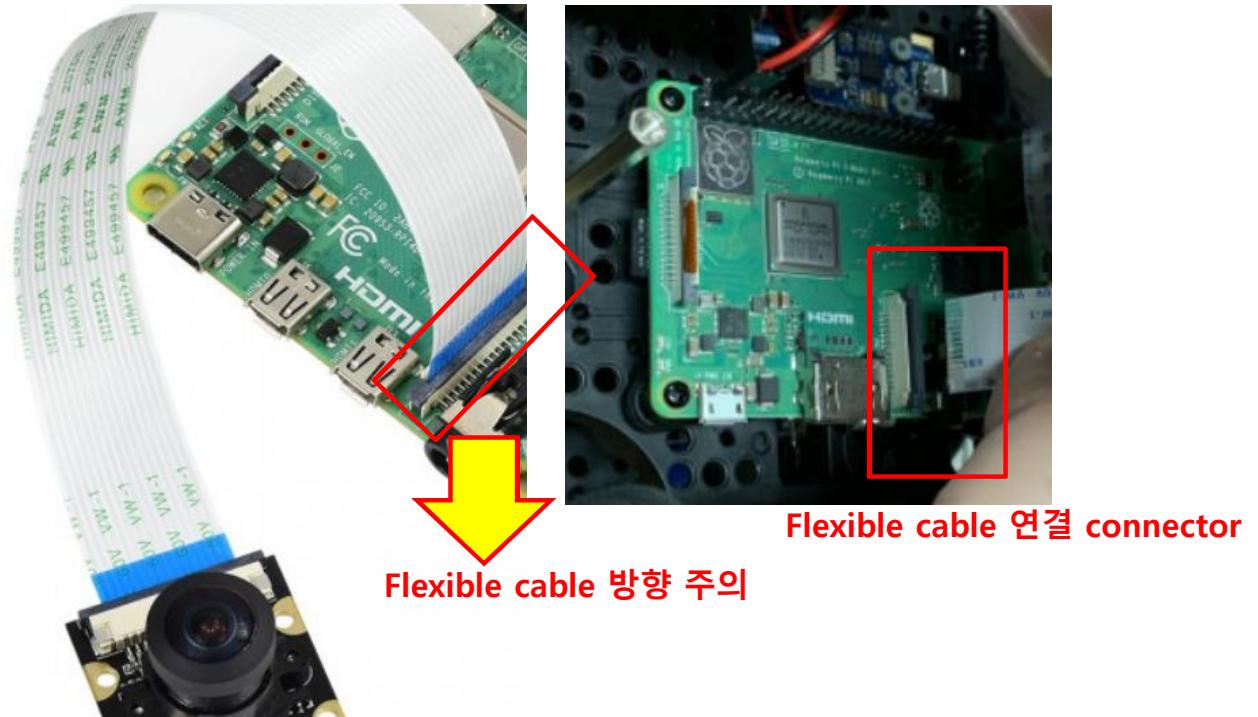
만일 회전 안하면, openCR의 RESET누르고, Ctl+c, 새 터미널
roslaunch turtlebot3_bringup turtlebot3_robot.launch

8. Autonomous Driving

8-1. RPi Camera(G) - Fisheye Lens 연결하기

(1) RPi Camera(G) - Fisheye Lens specification

- supports all revisions of the Pi
- Fisheye Lens, Angle of View : 160 degree (while other normal cameras are typically 72 degree)
- 5 megapixel OV5647 image sensor
- CCD size : 1/4inch
- Aperture (F) : 2.35
- Focal Length : 3.15mm
- Sensor best resolution : 1080p
- Provides 3.3V power output



8. Autonomous Driving

8-1. RPi Camera(G) - Fisheye Lens 연결하기

(2) RPi Camera(G) - Fisheye Lens를 Turtlebot3에 조립 및 Raspberry PI 3 B+ 연결

- Camera USB interface

: USB video device class (**UVC**) 라고 정의 함

→ USB video device class Ref. https://en.wikipedia.org/wiki/USB_video_device_class



https://www.usb.org/documents?search=&type%5B0%5D=55&items_per_page=50

- USB camera package

: libuvc-camera - UVC 표준 카메라 사용 위한 인터페이스 패키지

: uvc-camera - 상세 카메라 설정 변경 가능

- 카메라 두 대를 이용하여 스테레오 카메라 활용 가능 패키지

: usb-cam - Bosch에서 사용 카메라 드라이버

: freenect-camera, openni-camera, openni2-camera

- Kinect나 Xtion과 같은 심도 카메라

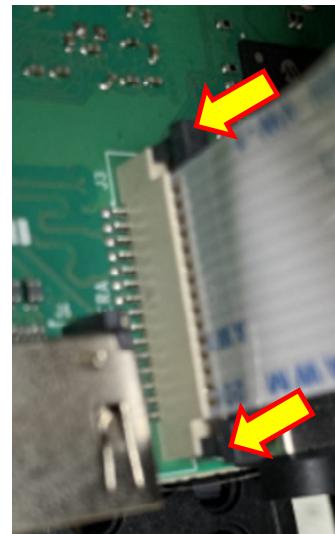
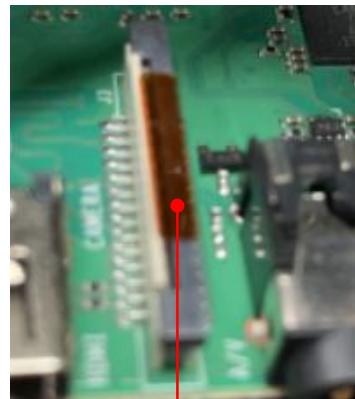
The screenshot shows the USB.org website's document library. The URL is https://www.usb.org/documents?search=&type%5B0%5D=55&items_per_page=50. The page title is "Document Library". It features a search bar, category filters, and a table of contents for the UVC Device Class Specification. The table includes columns for Title, Category, and Type. Three documents are listed: HID Usage Tables 1.22 (Device Class Specification), Billboard Device Class Spec Revision 1.2.2 and Adopters Agreement (Specification), and USB Class Definition for MIDI Devices v2.0 (Specification).

Title	Category	Type
HID Usage Tables 1.22	Device Class Specification	Device Class Specification
Billboard Device Class Spec Revision 1.2.2 and Adopters Agreement	Specification	Device Class Specification
USB Class Definition for MIDI Devices v2.0	Specification	Device Class Specification

8. Autonomous Driving

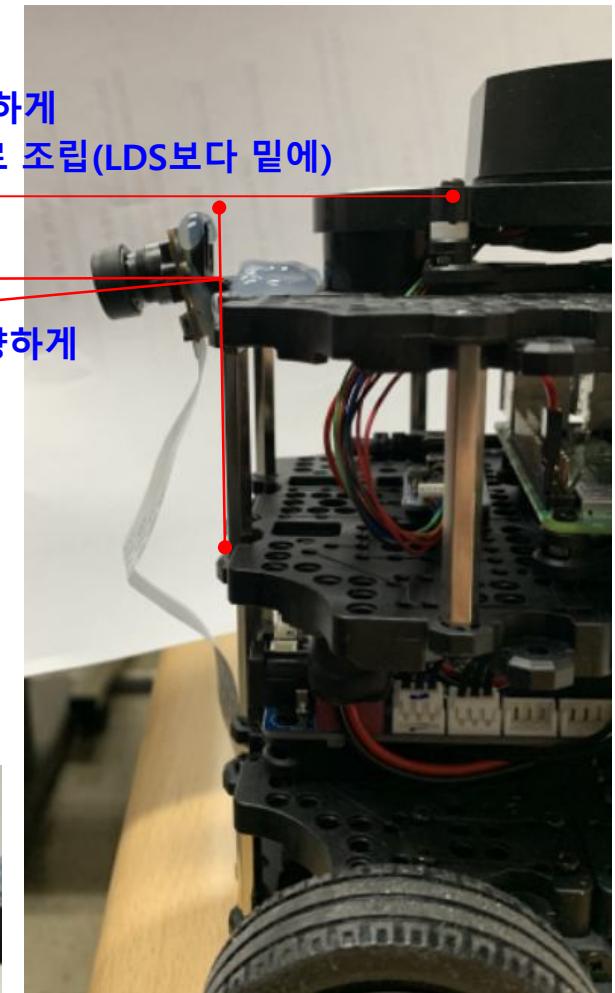
8-1. RPi Camera(G) - Fisheye Lens 연결하기

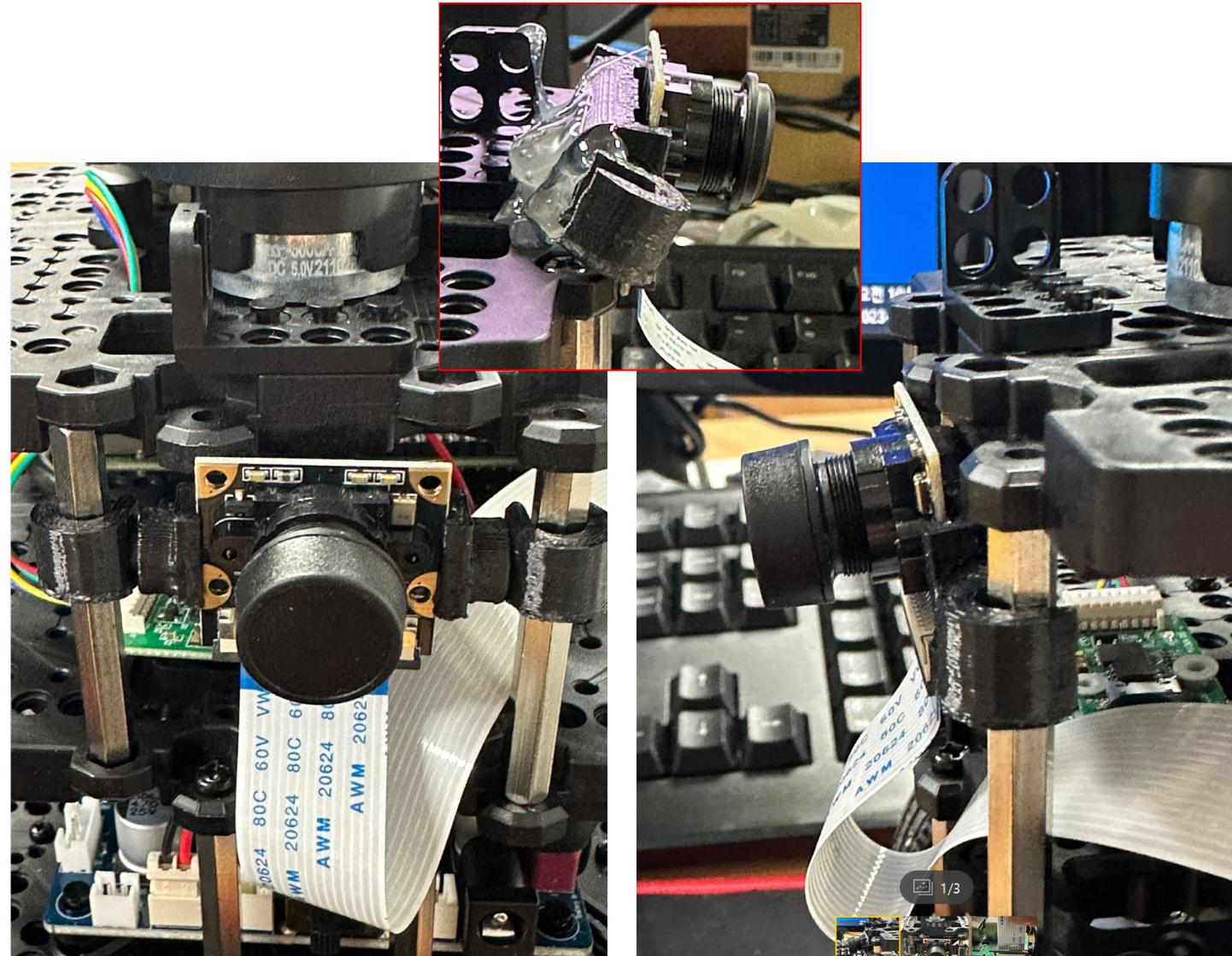
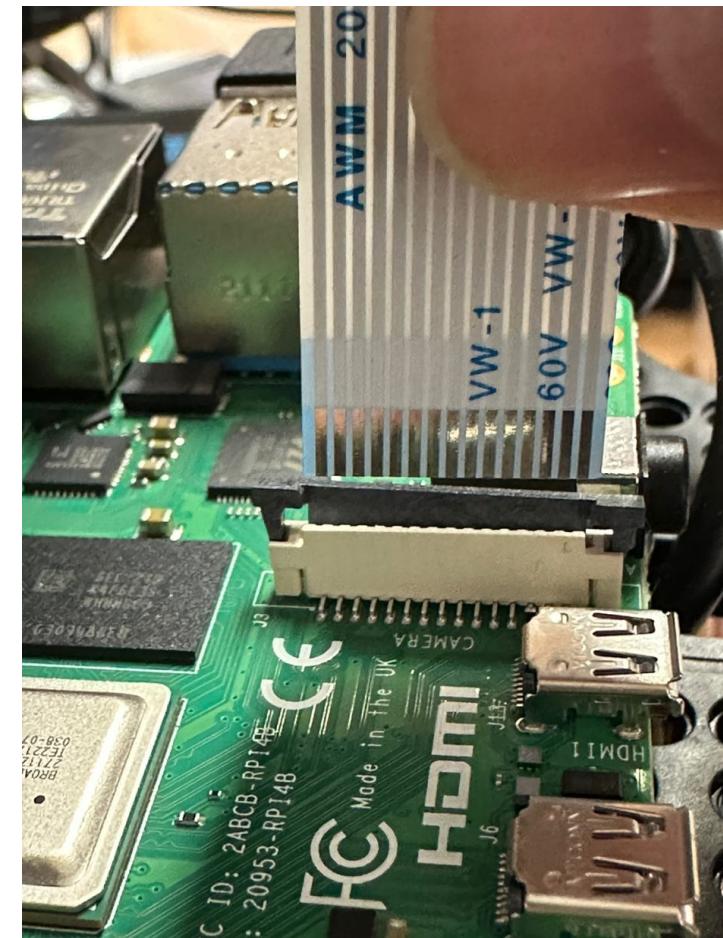
(2) RPi Camera(G) - Fisheye Lens를 Turtlebot3에 조립 및 Raspberry PI 4B 연결



: 정면을 향하게
: 글루건으로 조립(LDS보다 밑에)

5 degree 밑을 향하게





: <https://wiki.ros.org>

: Robotis e-manual 8장 https://emanual.robotis.com/docs/en/platform/turtlebot3/autonomous_driving/#getting-started

Enter Search Terms

Kinetic Melodic Noetic Dashing Foxy Humble Windows

TurtleBot3

- 1. Overview
- 2. Features
- 3. Quick Start Guide
- 4. SLAM
- 5. Navigation
- 6. Simulation
- 7. Manipulation
- 8. Autonomous Driving
 - 8.1. Getting Started
 - 8.2. Camera Calibration
 - Camera Imaging Calibration
 - Intrinsic Camera Calibration
 - Extrinsic Camera Calibration
 - Check Calibration Result
 - 8.3. Lane Detection
 - 8.4. Traffic Sign Detection
 - 8.5. Missions
 - Traffic Lights
 - Intersection
 - Construction
 - Parking
 - Level Crossing
 - Tunnel
 - 8.6. TurtleBot3 AutoRace 2019
- 9. Machine Learning
- 10. Examples

8. Autonomous Driving

8.1. Getting Started

NOTE

- Autorace package is mainly developed on [Ubuntu 20.04](#) with [ROS1 Noetic Nijjemys](#).
- Autorace package is mainly tested under the Gazebo simulation.
- To simulate given examples properly, complete [Simulation](#).

Tip: If you have actual TurtleBot3, you can perform up to [Lane Detection](#) from our Autonomous Driving package. For more details, click expansion note (ξ) at the end of content in each sub section.

The contents in e-Manual are subject to be updated without a prior notice. Therefore, some video may differ from the contents in e-Manual.

8.1.1. Prerequisites

[Remote PC](#)

- ROS 1 Noetic installed Laptop or desktop PC.
- This instruction is based on Gazebo simulation, but can be ported to the actual robot later.

ξ Click to expand : Prerequisites for use of actual TurtleBot3

8.1.2. Install Autorace Packages

1. Install the AutoRace 2020 meta package on [Remote PC](#).

```
$ cd ~/catkin_ws/src/
$ git clone -b noetic-devel https://github.com/ROBOTIS-GIT/turtlebot3_autorace_2020.git
$ cd ~/catkin_ws && catkin_make
```
2. Install additional dependent packages on [Remote PC](#).

```
$ sudo apt install ros-noetic-image-transport ros-noetic-cv-bridge ros-noetic-vision-opencv python3-opencv libopencv-dev ros-noetic-image-proc
```

ξ Click to expand : Autorace Package Installation for an actual TurtleBot3

8. Autonomous Driving

8.1.3 Install Autorace Packages

주의 : Turtlebot3 에 전원을 넣고 실행하기(Battery X ← 설치 시간 소요)

Remote PC에서 새터미널 열고

실행

8. 1. 3. Install Autorace Packages

in Gazebo simulation.

1. Install the AutoRace 2020 meta package on **Remote PC**.

```
$ cd ~/catkin_ws/src/  
$ git clone -b noetic-devel https://github.com/ROBOTIS-GIT/turtlebot3_autorace_2020.git  
$ cd ~/catkin_ws && catkin_make
```

2. Install additional dependent packages on **Remote PC**.

```
$ sudo apt install ros-noetic-image-transport ros-noetic-cv-bridge ros-noetic-vision-opencv python3-opencv libopencv-dev ros_noetic-image-proc
```



Click to expand : Autorace Package Installation for an actual TurtleBot3

The following instructions describes how to install packages and to calibrate camera.

1. Install AutoRace package on both **Remote PC** and **cnc**.

```
$ cd ~/catkin_ws/src/  
$ git clone -b feature-raspicam https://github.com/ROBOTIS-GIT/turtlebot3_autorace_2020.git  
$ cd ~/catkin_ws && catkin_make
```



8.1.3 Install Autorace Packages

fallocate failed

swap 파일을 생성하려고 하지만 같은 이름의 파일이 이미 존재하고 아직 사용 중임을 의미. 비활성화

실행

새 터미널 열고 \$ ssh ubuntu@192.168.0.21 (IP is Raspberry, password turtlebot)

 Ubuntu(SBC)에서 아래 실행 (1h 30m소요)

실행

☞ Click to expand : Autorace Package Installation for an actual TurtleBot3

The following instructions describes how to install packages and to calibrate camera.

1. Install AutoRace package on both [Remote PC] and [SBC].

```
$ cd ~/catkin_ws/src/  
$ git clone -b feature-raspicam https://github.com/ROBOTIS-GIT/turtlebot3_autorace_2020.git  
$ cd ~/catkin_ws && catkin_make
```

2. Install additional dependent packages on [SBC].

- Create a swap file to prevent lack of memory in building OpenCV.

```
$ sudo fallocate -l 4G /swapfile  
$ sudo chmod 600 /swapfile  
$ sudo mkswap /swapfile  
$ sudo swapon /swapfile
```

만일 fallocate fail 발생하면,
sudo swapoff -a
실행(비활성화)하고 다시 진행

```
ubuntu@ubuntu1:~/catkin_ws$ cd ..  
ubuntu@ubuntu1:~$ sudo fallocate -l 4G /swapfile  
fallocate: fallocate failed: Text file busy  
ubuntu@ubuntu1:~$ sudo swapoff -a  
ubuntu@ubuntu1:~$ sudo fallocate -l 4G /swapfile  
ubuntu@ubuntu1:~$ sudo chmod 600 /swapfile  
ubuntu@ubuntu1:~$ sudo mkswap /swapfile  
mkswap: /swapfile: warning: wiping old swap signature.  
Setting up swapspace version 1, size = 4 GiB (4294963200 bytes)  
no label, UUID=3546a200-c603-45f1-9197-3b5afbcef35  
ubuntu@ubuntu1:~$ sudo swapon /swapfile  
ubuntu@ubuntu1:~$
```

8.1.3 Install Autorace Packages

실행

- Install required dependencies.

```
$ sudo apt-get update  
$ sudo apt-get install build-essential cmake gcc g++ git unzip pkg-config  
$ sudo apt-get install libjpeg-dev libpng-dev libtiff-dev libavcodec-dev libavformat-dev libswscale-dev libgtk2.0-dev libcanberra-gtk* libxvidcore-dev libx264-dev python3-dev pythc
```



주의
끝까지 drag한 후에 실행

- Build with opencv & opencv_contrib

```
$ cd ~  
$ wget -O opencv.zip https://github.com/opencv/opencv/archive/4.5.0.zip  
$ wget -O opencv_contrib.zip https://github.com/opencv/opencv_contrib/archive/4.5.0.zip  
  
$ unzip opencv.zip  
$ unzip opencv_contrib.zip  
  
$ mv opencv-4.5.0 opencv  
$ mv opencv_contrib-4.5.0 opencv_contrib
```



 실행

주의

끝까지 drag한 후에 실행

○ Create cmake file.

```
$ 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 BUILD_TIFF=ON #  
    -D WITH_FFMPEG=ON #  
    -D WITH_GSTREAMER=ON #  
    -D WITH_TBB=ON #  
    -D BUILD_TBB=ON #  
    -D BUILD_TESTS=OFF #  
    -D WITH_EIGEN=OFF #  
    -D WITH_V4L=ON #  
    -D WITH_LIBV4L=ON #  
    -D WITH_VTK=OFF #  
    -D OPENCV_ENABLE_NONFREE=ON #  
    -D INSTALL_C_EXAMPLES=OFF #  
    -D INSTALL_PYTHON_EXAMPLES=OFF #  
    -D BUILD_NEW_PYTHON_SUPPORT=ON #  
    -D BUILD_opencv_python3=TRUE #  
    -D OPENCV_GENERATE_PKGCONFIG=ON #  
    -D BUILD_EXAMPLES=OFF ...
```

```
ubuntu@ubuntu1:~/opencv$ mkdir build  
mkdir: cannot create directory 'build': File exists  
ubuntu@ubuntu1:~/opencv$ ls  
3rdparty  CMakeLists.txt  doc      opencv-4.5.0  SECURITY.md  
apps      CONTRIBUTING.md  include   platforms  
build     COPYRIGHT        LICENSE  README.md  
cmake     data            modules  samples  
ubuntu@ubuntu1:~/opencv$ cd build  
ubuntu@ubuntu1:~/opencv/build$
```

실행 결과

```
-- Install to: /usr/local  
--  
-- Configuring done  
-- Generating done  
-- Build files have been written to: /home/ubuntu/opencv/build  
ubuntu@ubuntu1:~/opencv/build$
```





실행

- It will take an hour or two to build.

```
$ cd ~/opencv/build  
$ make -j4  
$ sudo make install  
$ sudo ldconfig  
$ make clean  
$ sudo apt-get update
```

make 완료 화면

```
[100%] Building CXX object modules/python3/CMakeFiles/opencv_python3.dir/_/_src2/cv2.cpp.o  
[100%] Linking CXX executable ../../bin/opencv_perf_ximgproc  
[100%] Built target opencv_perf_ximgproc  
[100%] Linking CXX executable ../../bin/opencv_perf_superres  
[100%] Built target opencv_perf_superres  
[100%] Linking CXX shared module ../../lib/python3/cv2.cpython-38-aarch64-linux-gnu.so  
[100%] Built target opencv_python3  
ubuntu@ubuntu1:~/opencv/build$
```

완료 화면

```
ubuntu@ubuntu1:~/opencv/build$ sudo ldconfig  
ubuntu@ubuntu1:~/opencv/build$ make clean  
ubuntu@ubuntu1:~/opencv/build$ sudo apt-get update  
Hit:1 http://packages.ros.org/ros/ubuntu focal InRelease  
Hit:2 http://ports.ubuntu.com/ubuntu-ports focal InRelease  
Get:3 http://ports.ubuntu.com/ubuntu-ports focal-updates InRelease [114  
Hit:4 http://ports.ubuntu.com/ubuntu-ports focal-backports InRelease  
Hit:5 http://ports.ubuntu.com/ubuntu-ports focal-security InRelease  
Hit:6 http://ports.ubuntu.com/ubuntu-ports focal-proposed InRelease  
Fetched 114 kB in 21s (5,529 B/s)  
Reading package lists... Done  
ubuntu@ubuntu1:~/opencv/build$
```

참고 : 우리는 실행 안함. Turtlebot3에 별도 모니터 연결하기 위해, 아래 실행하고 window에서 내용 수정하는 것임

- Turn off Raspberry Pi, take out the microSD card and edit the config.txt in system-boot section. add start_x=1 before the enable_uart=1 line.

여기부터
실행

```
$ sudo apt install ffmpeg  
$ ffmpeg -f video4linux2 -s 640x480 -i /dev/video0 -ss 0:0:2 -frames 1 capture_test.jpg
```

- Install additional dependent packages

```
$ sudo apt install ros-noetic-cv-camera
```

완료 화면

```
ubuntu@ubuntu1:~/opencv/build$ sudo apt install ros-noetic-cv-camera  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
ros-noetic-cv-camera is already the newest version (0.6.0-1focal.20230620.200022).  
The following packages were automatically installed and are no longer required:  
  libfwupdplugin1 libxml2-2.9.9 libxml2-dev libxml2-doc libxml2-utils  
  linux-headers-5.4.0-1043-raspi linux-image-5.4.0-1043-raspi  
  linux-modules-5.4.0-1043-raspi linux-raspi-headers-5.4.0-1043  
Use 'sudo apt autoremove' to remove them.  
0 upgraded, 0 newly installed, 0 to remove and 24 not upgraded.  
ubuntu@ubuntu1:~/opencv/build$
```

8.1.3 Install Autorace Packages



SBC 실행 이후 ubuntu@ubuntu1:~

Remote PC에서 새 터미널 열고 yongseok@yongseok:~\$
실행

3. Install additional dependent packages on [Remote PC].

```
$ sudo apt install ros-noetic-image-transport ros-noetic-image-transport-plugins ros-noetic-cv-bridge ros-noetic-vision-opencv python3-opencv libopencv-dev ros-noetic-image-proc ros-noe
```

주의
끝까지 drag한 후에 실행

완료 화면

```
yongseok@yongseok:~$ sudo apt install ros-noetic-image-transport ros-noetic-image-transport-plugins ros-noetic-cv-bridge ros-noetic-c-vision-opencv python3-opencv libopencv-dev ros-noetic-image-proc ros-noetic-cv-camera ros-noetic-camera-calibration
[sudo] password for yongseok:
Reading package lists... Done
Building dependency tree
Reading state information... Done
libopencv-dev is already the newest version (4.2.0+dfsg-5).
python3-opencv is already the newest version (4.2.0+dfsg-5).
ros-noetic-camera-calibration is already the newest version (1.17.0-1focal.20230620.192537).
ros-noetic-camera-calibration set to manually installed.
ros-noetic-cv-bridge is already the newest version (1.16.2-1focal.20230620.191830).
ros-noetic-image-proc is already the newest version (1.17.0-1focal.20230620.192452).
ros-noetic-image-transport is already the newest version (1.12.0-1focal.20230620.191833).
ros-noetic-image-transport-plugins is already the newest version (1.14.0-1focal.20230620.194116).
ros-noetic-image-transport-plugins set to manually installed.
ros-noetic-vision-opencv is already the newest version (1.16.2-1focal.20230620.193649).

The following NEW packages will be installed:
  ros-noetic-cv-camera
0 upgraded, 1 newly installed, 0 to remove and 5 not upgraded.
Need to get 61.7 kB of archives.
After this operation, 309 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://packages.ros.org/ubuntu focal/main amd64 ros-noetic-cv-camera amd64 0.6.0-1focal.20230620.193132 [61.7 kB]
Fetched 61.7 kB in 2s (28.8 kB/s)
Selecting previously unselected package ros-noetic-cv-camera.
(Reading database ... 299968 files and directories currently installed.)
Preparing to unpack .../ros-noetic-cv-camera_0.6.0-1focal.20230620.193132_amd64.deb ...
Unpacking ros-noetic-cv-camera (0.6.0-1focal.20230620.193132) ...
Setting up ros-noetic-cv-camera (0.6.0-1focal.20230620.193132) ...
yongseok@yongseok:~$
```

8. Autonomous Driving

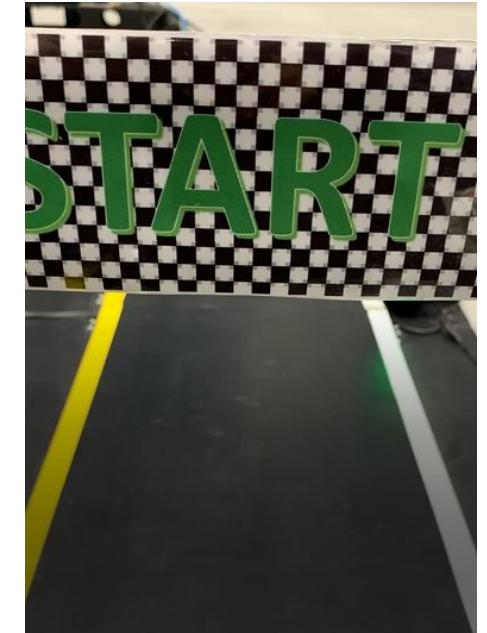
8-2. Camera calibration

(1) Install Autorace(2020) Packages

ⓐ Turtlebot3 Autorace 2020

- Mission에 따른 camera calibration

- : Mission1. Traffic Light → 신호등 green 확인 후 출발
- : Mission2. Intersection → 좌우 교차로 표지판 인식하여 교차로 주행
- : Mission3. Construction → 공사 표지판 인식하여 3개의 장애물 피함
- : Mission4. Parking → 주차 표지판 인식 후 주차 빈 자리에 주차
- : Mission5. Level Crossing → 정지 표지판 인식 후 차단 바를 본 후 정지하고 출발
- : Mission6. Tunnel → 터널 표지판 인식 후 SLAM을 실행하여 터널 통과



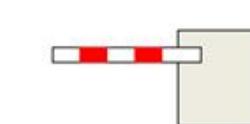
동영상 - Tracks image



신호등 미션



공사구간 미션



차단바 미션



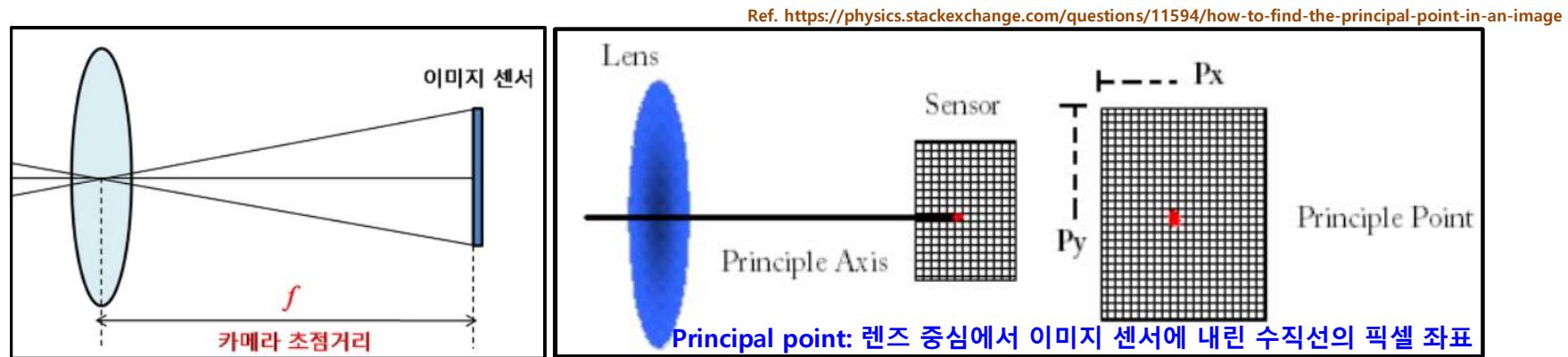
터널 미션

8-2. Camera calibration

(2) Camera calibration

- : 영상을 촬영할 때, 영상 품질은 카메라의 위치 및 방향 등 외부적(extrinsic) 요인에 의해 결정되나,
- : 그러나 실제 이미지는 사용된 렌즈, **렌즈와 이미지 센서와의 거리**, 렌즈와 이미지 센서가 이루는 각 등 카메라 내부(intrinsic) 기구 조립 요소에 의해 영향 받음.

→ **intrinsic calibration** - 초점거리(focal length), 주점(principal point) 등 카메라 렌즈의 광학적 요소를 보정(calibration)을 의미



→ **extrinsic calibration** - 카메라의 설치 높이, 방향(tilt) 등 카메라와 외부 공간과의 기하학적 관계에 의한 카메라 보정

8-2. Camera calibration

(2) Camera calibration

: camera calibration parameter

→ camera image calibration - imaging 보정

→ intrinsic calibration - 초점거리(focal length), 주점(principal point) 등 카메라 렌즈의
광학적 요소를 보정(calibration)을 의미

→ extrinsic calibration - 카메라의 설치 높이, 방향(tilt-기울기) 등 카메라와 외부 공간과의 기하학적 관계에 의한 카메라 보정

: intrinsic calibration parameter, extrinsic calibration parameter 를 camera matrix 라고 부름

: parameter 경로

→ intrinsic calibration

catkin_ws / src / turtlebot3_autorace_2020 / turtlebot3_autorace_camera
/ calibration / intrinsic_calibration

→ extrinsic calibration

catkin_ws / src / turtlebot3_autorace_2020 / turtlebot3_autorace_camera
/ calibration / extrinsic_calibration

8-2. Camera calibration

(3) Camera imaging calibration

: autonomous driving를 위해 camera calibration이 매우 중요

: Ref. 8.2.1장 https://emanual.robotis.com/docs/en/platform/turtlebot3/autonomous_driving/#autonomous-driving

ⓐ Remote PC에서 새 터미널 열기

```
$ roscore
```

ⓑ Remote PC에서 새 터미널 열기

```
$ ssh ubuntu@192.168.0.25
```

← Remote PC에서 TurtleBot3 Burger(Raspberry pi)을 원격접속



pw : turtlebot 입력

```
~$ roslaunch turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
```

```
Last login: Sat Dec 2 02:35:37 2023 from 192.168.1.141
ubuntu@ubuntu1:~$ roslaunch turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
... logging to /home/ubuntu/.ros/log/3af56884-9274-11ee-af9f-a
9587662cf9e/roslaunch-ubuntu1-1304.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.1.145:42779/
```

```
PARAMETERS
* /cv_camera/camera_info_url: package://turtleb...
* /cv_camera/frame_id: camera
* /cv_camera/image_height: 240
* /cv_camera/image_width: 320
* /cv_camera/rate: 30
* /rosdistro: noetic
* /rosversion: 1.16.0

NODES
/
  cv_camera (cv_camera/cv_camera_node)

ROS_MASTER_URI=http://192.168.1.141:11311

process[cv_camera-1]: started with pid [1318]
[ WARN:0] global ..../modules/videoio/src/cap_gstreamer.cpp (480
) isPipelinePlaying OpenCV | GStreamer warning: GStreamer: pip
eline have not been created
[ INFO] [1701673956.244524519]: camera calibration URL: packag
e://turtlebot3_autorace_camera/calibration/intrinsic_calibrati
on/camerav2_320x240_30fps.yaml
```

© Remote PC에서 새 터미널 열기

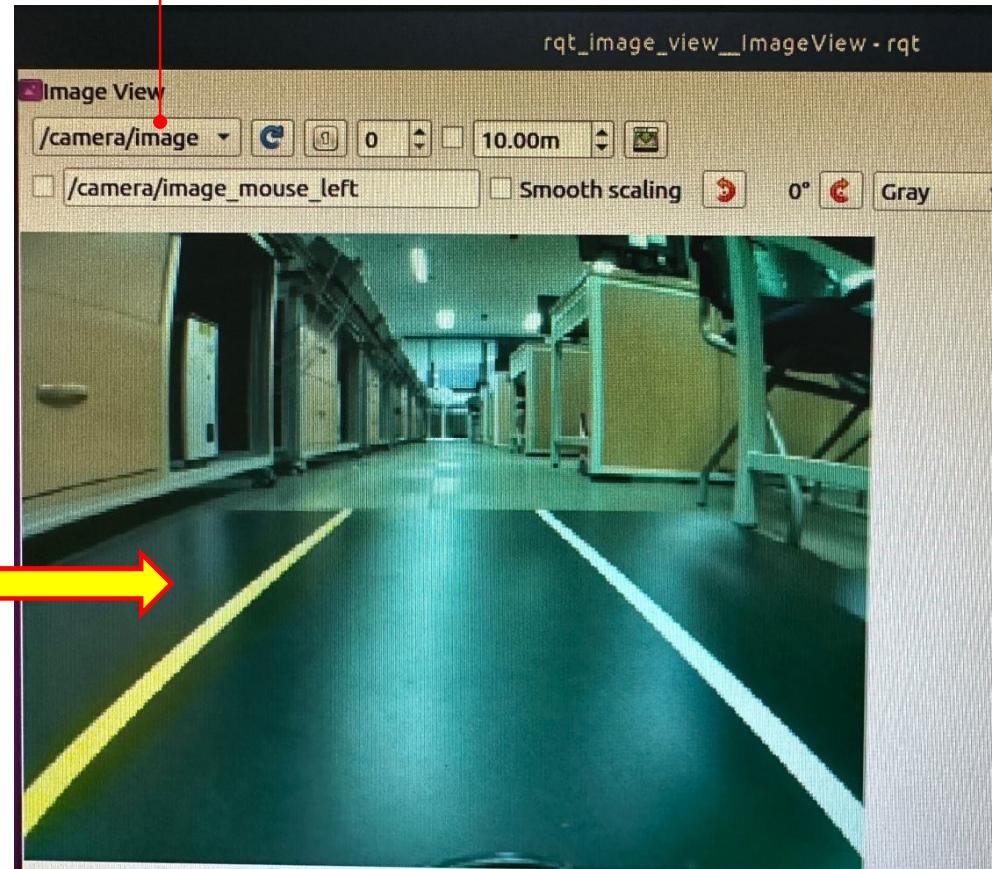
TurtleBot3 Burger를 경기장에 내려놓음

\$ rqt_image_view

```
yongseok@yongseok:~$ rqt_image_view
```



/camera/image 선택 → camera 화면 연결됨



실제 경기장

: Turtlebot의 왼쪽 Yellow, 오른쪽 white

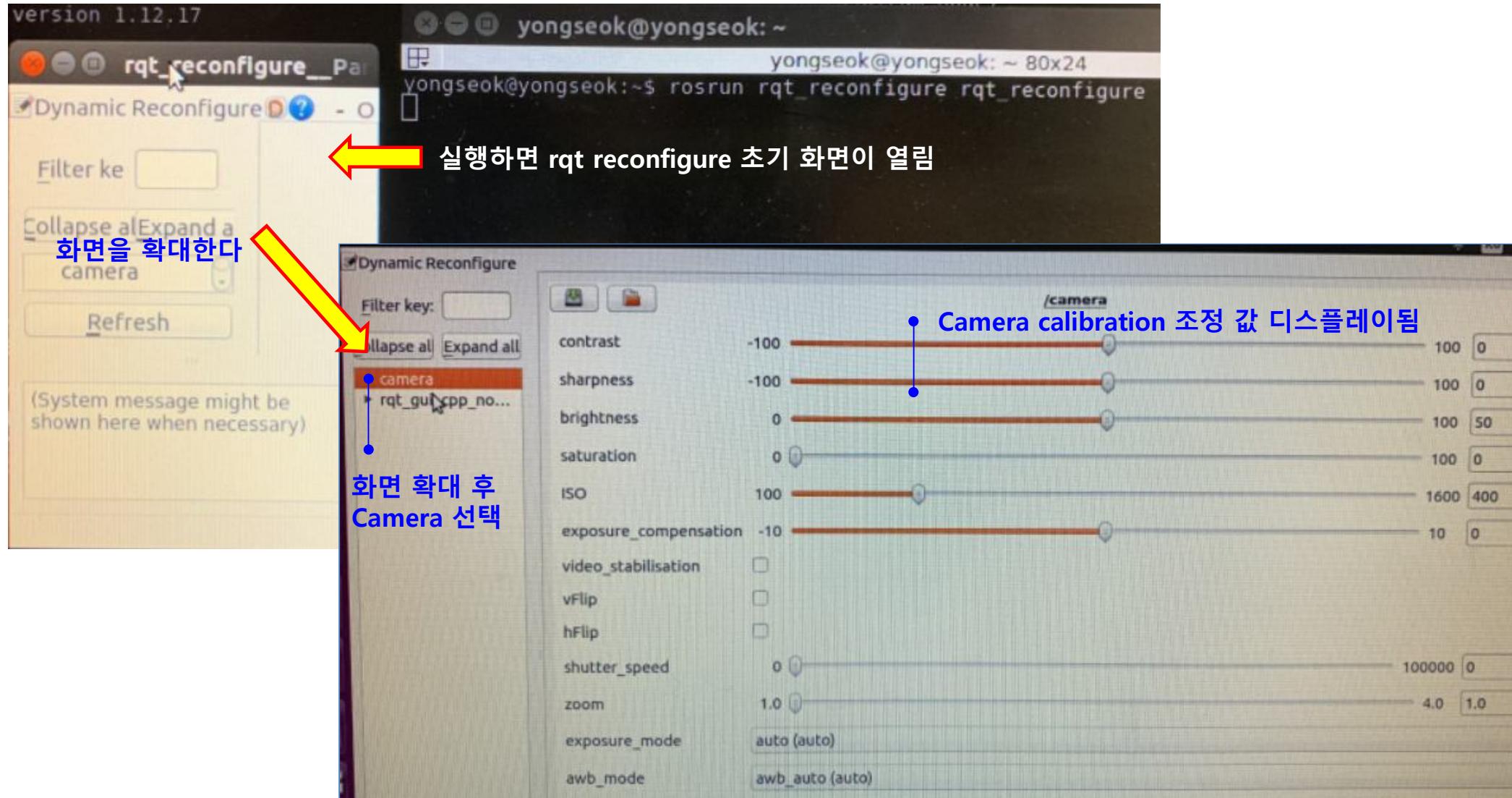
: calibration에서는 관계없음

만일 실험을 마치려면, terminal은 CTL+C로 중단 후에 종료

④ Remote PC에서 새 터미널 열기

Skip for Kinectic

\$ rosrun rqt_reconfigure rqt_reconfigure

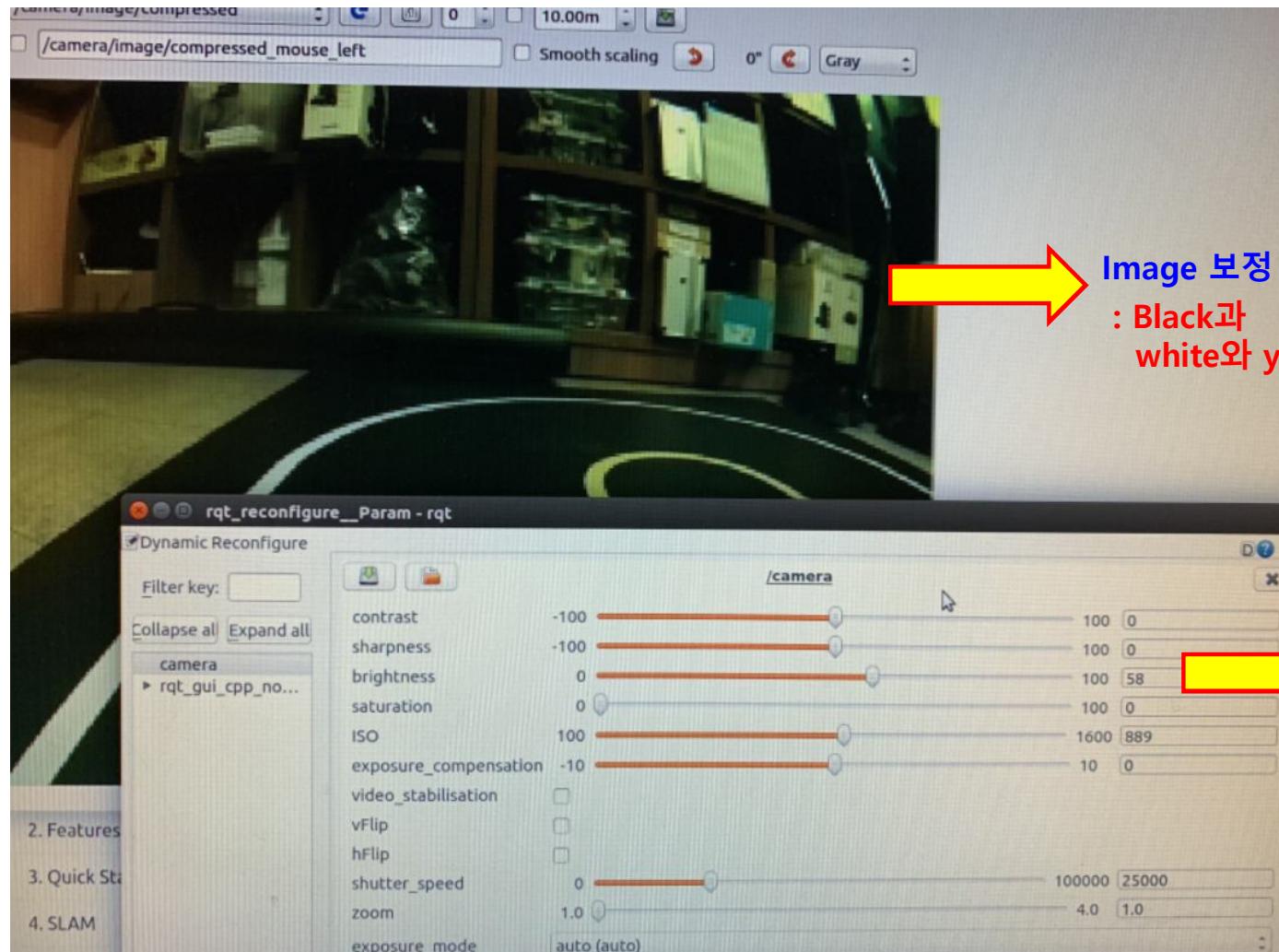


Skip for Kinectic

⑤ Remote PC에서

rqt_image_view 를 보면서, rqt_reconfigure의 camera image calibration 진행

← calibration에서 가장 중요한 것은 조도 값(외부 조명 조도)



: 야간, 주간 흰색과 노란색 구별 정도

Image 보정

: Black과

white와 yellow가 서로 잘 구분되도록 영상 보정



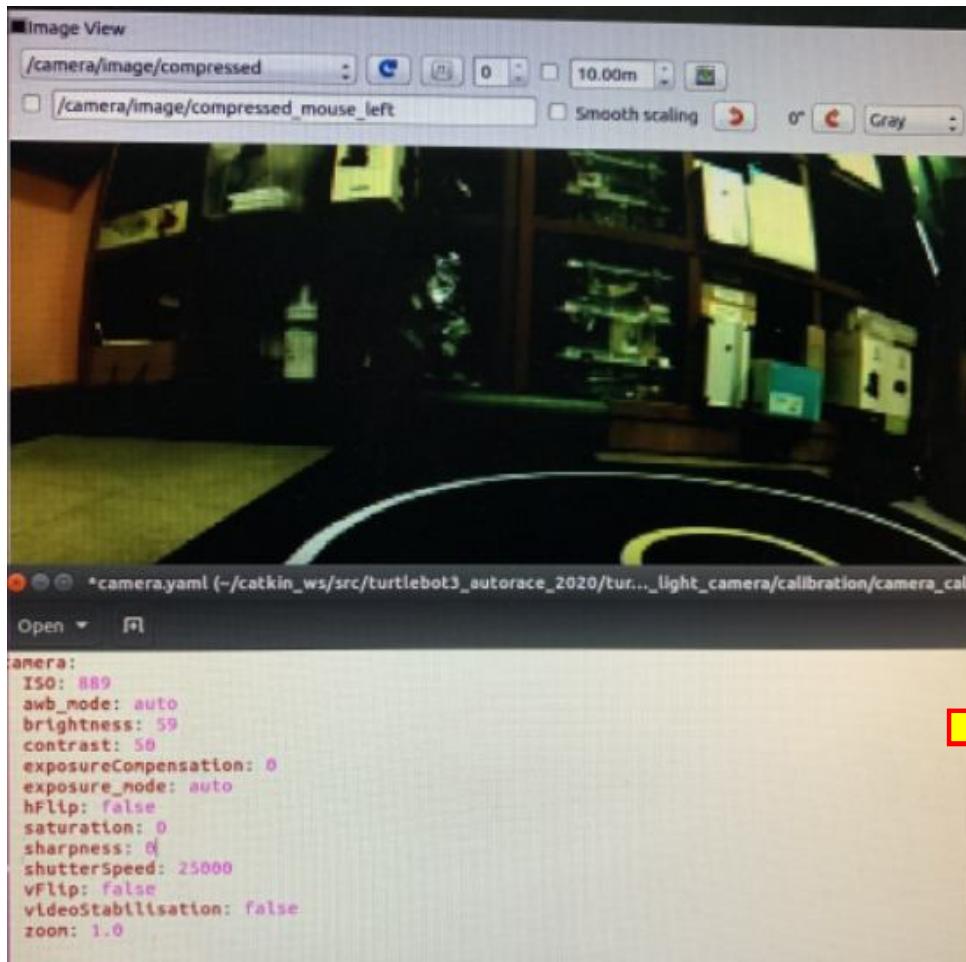
Camera calibration 조정

f) Remote PC에서

Skip for Kinetic

rqt_reconfigure의 camera image calibration 값을 camera.yaml에 저장

→ 경로 catkin_ws / src / turtlebot3_autorace_2020 / turtlebot3_autorace_traffic_light /
turtlebot3_autorace_traffic_light_camera / calibration / camera_calibration



The screenshot shows a terminal window with the title '*camera.yaml (~/catkin_ws/src/turtle...'. The file content is identical to the one in the rqt_reconfigure window:

```
camera:  
ISO: 889  
awb_mode: auto  
brightness: 59  
contrast: 50  
exposureCompensation: 0  
exposure_mode: auto  
hFlip: false  
saturation: 0  
sharpness: 0  
shutterSpeed: 25000  
vFlip: false  
videoStabilisation: false  
zoom: 1.0
```

A yellow arrow points from the left side of the terminal window towards the 'save' button on the right.

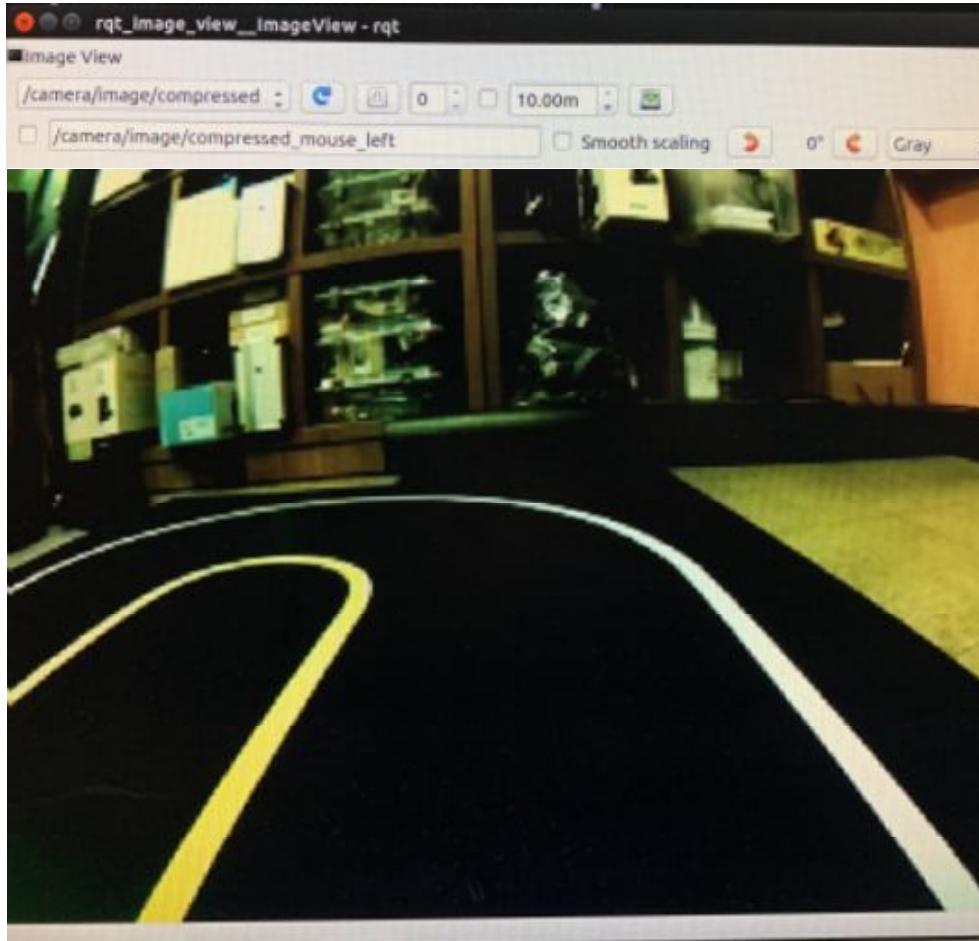
Skip for Kinetic

⑤ Remote PC에서

: rqt_image_view 터미널 CTL +C 로 종료 후 닫기. rqt_image_view 화면 닫기. rqt_reconfigure 화면 닫기. camera.yaml 닫기

\$ **rqt_image_view**

다시 열어서 camera.yaml에 의해 변경된 image 보정된 영상 확인하기



: yellow , white가 서로 잘 구분되도록 영상 보정 확인

8-2. RPi Camera(G) – Camera calibration

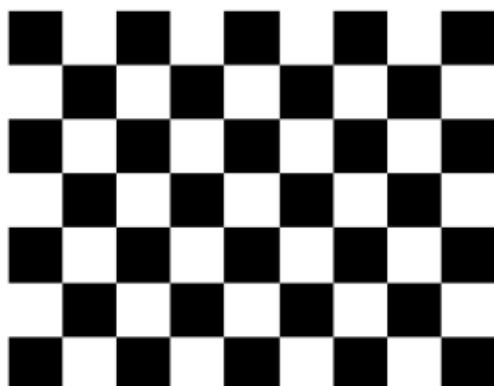
(4) Intrinsic Camera Calibration(터미널창 미리 6개 띄우기)

: Intrinsic Camera Calibration 를 위해 checkerboard 사용

: checkerboard

The checkerboard is stored at turtlebot3_autorace_camera/data/checkerboard_for_calibration.pdf

→ checkerboard



: A4 출력하여, Intrinsic Camera Calibration 사용



→ Calibration 진행 시, checkerboard는 흔들리지 않도록

(팔랑거리지 않도록 고정하는 것이 중요)

ⓐ Remote PC에서 새 터미널 열기

\$ roscore

ⓑ Remote PC에서 새 터미널 열기

\$ ssh ubuntu@192.168.0.25

← Remote PC에서 TurtleBot3 Burger(Raspberry pi)을 원격접속

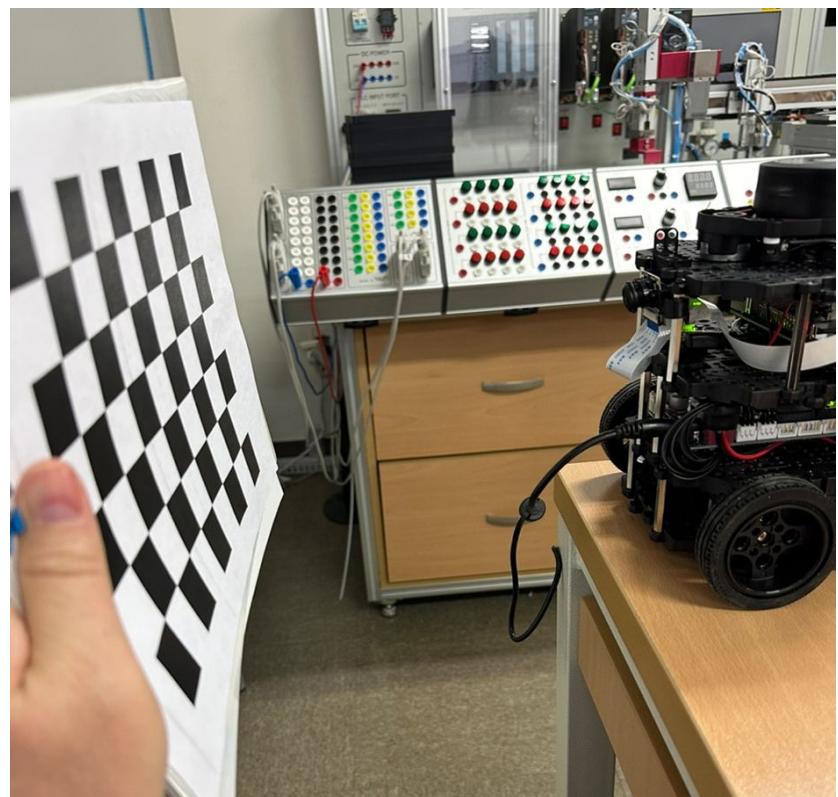
pw : turtlebot 입력

~\$ rosrun turtlebot3_autorace_camera raspberry_pi_camera_publish.launch

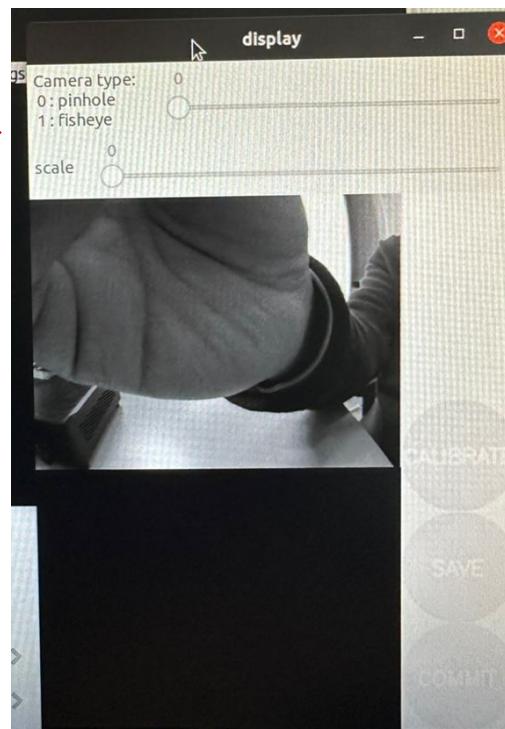
④ Remote PC에서 새 터미널 열기

TurtleBot3 Burger(책상 끝에) 앞에 checkerboard(가로방향) 내려놓음

```
$ roslaunch turtlebot3_autorace_camera intrinsic_camera_calibration.launch mode:=calibration
```



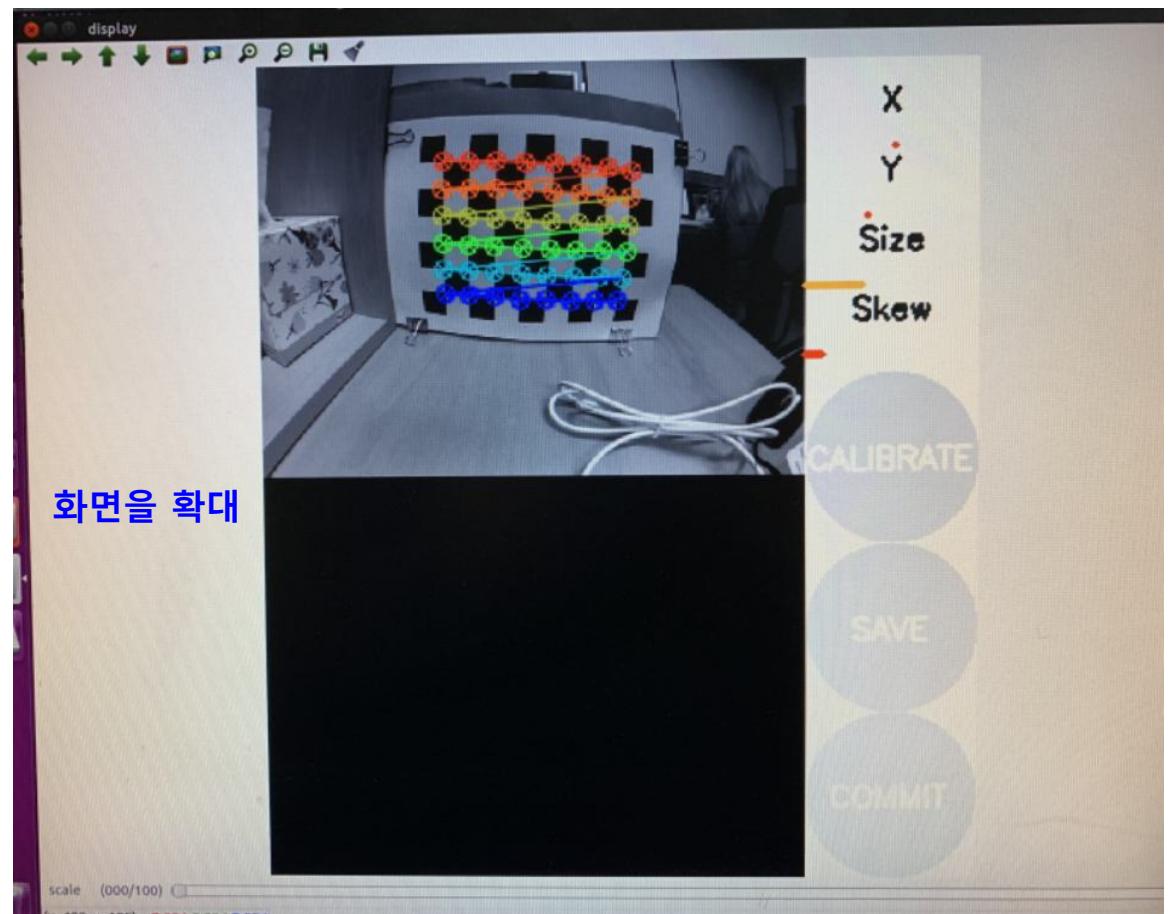
: A4 출력하여, Intrinsic Camera Calibration 사용
→ Calibration 진행 시, checkerboard는 흔들리지 않도록
(팔랑거리지 않도록 고정하는 것이 중요)



실행 후 화면

```
ROS_MASTER_URI=http://192.168.0.30:11311  
process[republish-1]: started with pid [5929]  
process[cameracalibrator-2]: started with pid [5930]  
('Waiting for service', '/camera/set_camera_info',  
'...')  
OK  
*** Added sample 1, p_x 0.511, p_y = 0.351, p_size = 0.321, skew = 0.097
```

The screenshot shows the ROS terminal output for the camera calibration process. It includes logs for the republish node (pid 5929) and the cameracalibrator node (pid 5930). The terminal shows the camera waiting for a service and then adding a sample point with specific coordinates and parameters. Below the terminal is the 'display' window of the cameracalibrator2 tool. This window shows a camera feed of a patterned board with colored dots. On the right side of the window, there are four buttons labeled 'CALIBRATE', 'SAVE', 'COMMIT', and 'DISCARD'. A vertical toolbar on the right contains buttons for 'X', 'Y', 'Size', and 'Skew'. Two yellow arrows point from the terminal log to the 'display' window: one arrow points to the 'Skew' button in the toolbar, and another arrow points to the main camera feed area.



⑤ Intrinsic Camera Calibration 진행

- : 카메라를 기준으로 왼쪽/오른쪽/위/아래로 checkerboard를 움직이기
- : 카메라를 기준으로 앞/뒤로 움직이기, checkerboard를 상하, 좌우 비스듬히 기울이기
- 조건들이 채워질수록 X, Y, Size, Skew의 BAR 점점 채워지며 녹색으로 바뀜



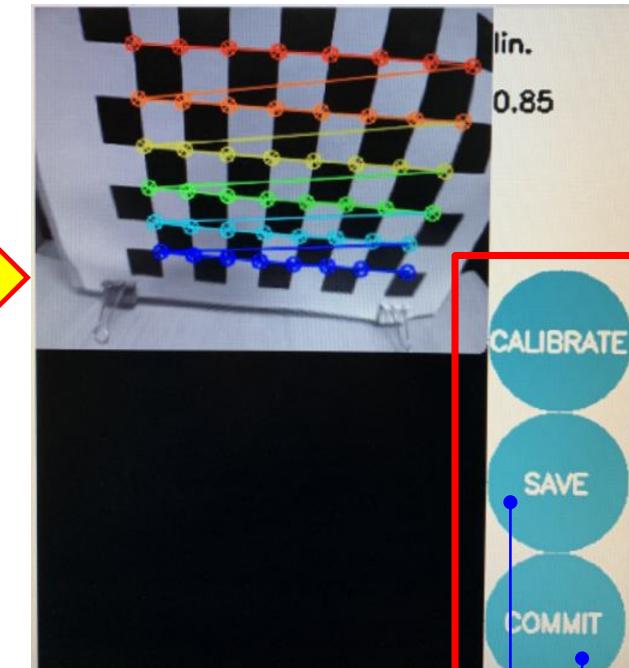
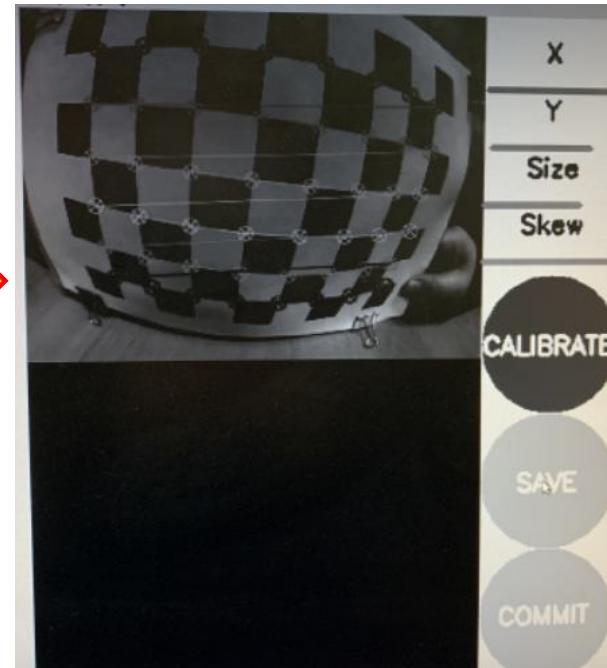
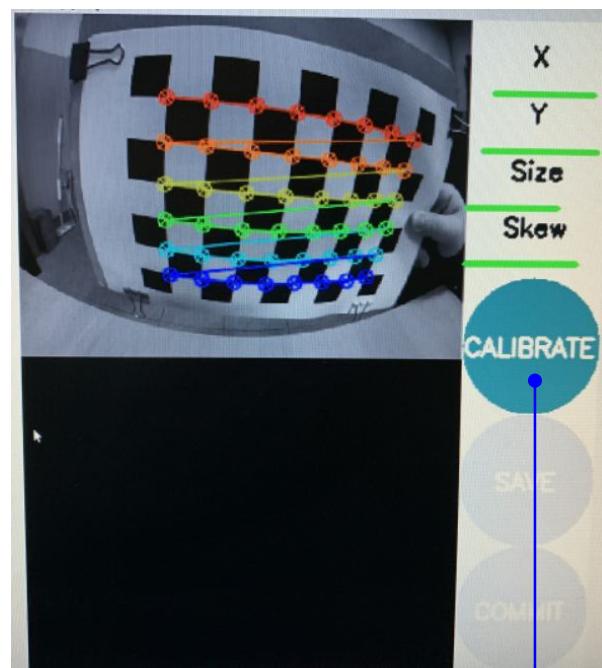
초기화면

checkerboard를 움직임에 따라 초록색 BAR

초록색 BAR 최대 생성되면
Calibration 활성화 됨

④ click Calibration

: Calibration 누르고 1분 기다리기 → SAVE 활성화 됨



활성화

만일 실험을 마치려면, terminal은 CTL+C로 중단 후에 종료

⑨ TMP 폴더에 calibrationdata.tar.gz 생성됨



Extract Here – click 하여 압축 풀기

④ ost.yaml 파일 열기(double click)

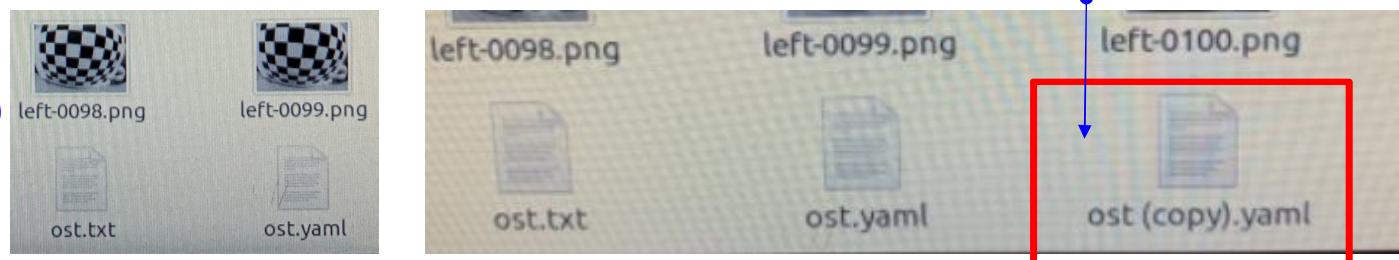
```
image_width: 320
image_height: 240
camera_name: narrow_stereo
camera_matrix:
  rows: 3
  cols: 3
  data: [154.023918, 0.000000, 152.573255, 0.000000, 154.518473, 99.828186, 0.000000, 0.000000, 1.000000]
distortion_model: plumb_bob
distortion_coefficients:
  rows: 1
  cols: 5
  data: [-0.260448, 0.044973, 0.000049, 0.000158, 0.000000]
rectification_matrix:
  rows: 3
  cols: 3
  data: [1.000000, 0.000000, 0.000000, 0.000000, 1.000000, 0.000000, 0.000000, 0.000000, 1.000000]
projection_matrix:
  rows: 3
  cols: 4
  data: [104.956978, 0.000000, 153.489556, 0.000000, 0.000000, 116.299614, 86.441693, 0.000000, 0.000000, 0.000000, 1.000000, 0.000000]
```

: 카메라 이름을 narrow_stereo에서 camera 변경하기 → 저장하기

: 이름 변경하기 camerav2_320x240_30fps.yaml

⑤ tmp 폴더의

ost.yaml을 CTL + C(복사), CTL+V(붙여 놓기)

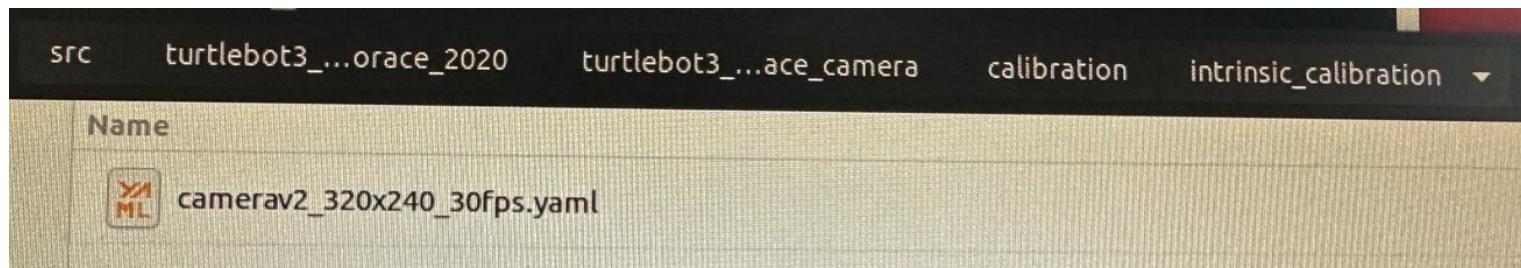


⑤ tmp 폴더에서 생성한 **camerav2_320x240_30fps.yaml** 을 복사(CTL + C)하여

아래 경로에 붙여 넣기(CTL + V) – Replace

→ 경로 **catkin_ws / src / turtlebot3_autorace_2020 / turtlebot3_autorace_camera / calibration / intrinsic_calibration**

→ 위 경로에는 이미 **camerav2_320x240_30fps.yaml** 있으며, 이를 새로 업데이트 한다는 의미(덮어 쓰기)



→ Intrinsic Calibration Data인 **camerav2_320x240_30fps.yaml** 를 덮어 씀(업데이트)

8-2. RPi Camera(G) – Camera calibration

(5) Extrinsic Camera Calibration

: Extrinsic Camera Calibration 를 위해 경기장(Lane) 사용

8. 2. 1. Extrinsic Camera Calibration

1. Open a new terminal on **Remote PC** and launch Gazebo.

```
$ roslaunch turtlebot3_gazebo turtlebot3_autorace_2020.launch
```

2. Open a new terminal and launch the intrinsic camera calibration node.

```
$ rosrun turtlebot3_autorace_camera intrinsic_camera_calibration.launch
```

3. Open a new terminal and launch the extrinsic camera calibration node.

```
$ rosrun turtlebot3_autorace_camera extrinsic_camera_calibration.launch mode
```

4. Execute rqt on **Remote PC**.

```
$ rqt
```

5. Select **plugins > visualization > Image view**. Create two image view windows.

6. Select **/camera/image extrinsic_calib/compressed** topic on one window and

in Gazebo simulation.

실행

turtlebot3_autorace_camera/calibration/extrinsic_calibration/ projection.yaml

Click to expand : Extrinsic Camera Calibration with an actual TurtleBot3

1. Launch roscore on **Remote PC**.

```
$ roscore
```

2. Trigger the camera on **SBC**.

```
$ roslaunch turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
```

3. Use the command on **Remote PC**.

```
$ rosrun turtlebot3_autorace_camera intrinsic_camera_calibration.launch mode:=action
```

4. Run the extrinsic camera calibration launch file on **Remote PC**.

```
$ roslaunch turtlebot3_autorace_camera extrinsic_camera_calibration.launch mode:=calibration
```

5. Execute rqt on **Remote PC**.

```
$ rqt
```

6. Click **plugins > visualization > Image view**: Multiple windows will be present.

8-2. RPi Camera(G) – Camera calibration

(5) Extrinsic Camera Calibration(터미널창 미리 8개 띄우기)

: Extrinsic Camera Calibration 를 위해 경기장(Lane) 사용

ⓐ Remote PC에서 새 터미널 열기

```
$ roscore
```

ⓑ Remote PC에서 새 터미널 열기

```
$ ssh ubuntu@192.168.0.25      ← Remote PC에서 TurtleBot3 Burger(Raspberry pi)을 원격접속  
pw : turtlebot 입력  
~$ roslaunch turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
```

ⓒ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_camera intrinsic_camera_calibration.launch mode:=action
```

Calibration에서 action 바뀜

실제 경기장

- : Turtlebot의 왼쪽 Yellow, 오른쪽 white
- : calibration에서는 관계없음

```
ROS_MASTER_URI=http://192.168.1.141:11311  
process[republish-1]: started with pid [8897]  
process[relay_camera_info-2]: started with pid [8898]  
process[camera/image_proc-3]: started with pid [8899]
```



④ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_camera extrinsic_camera_calibration.launch mode:=calibration
```

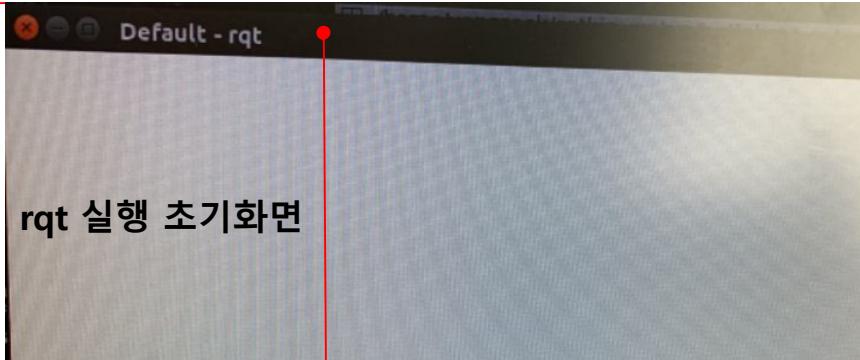


```
ROS_MASTER_URI=http://192.168.1.141:11311

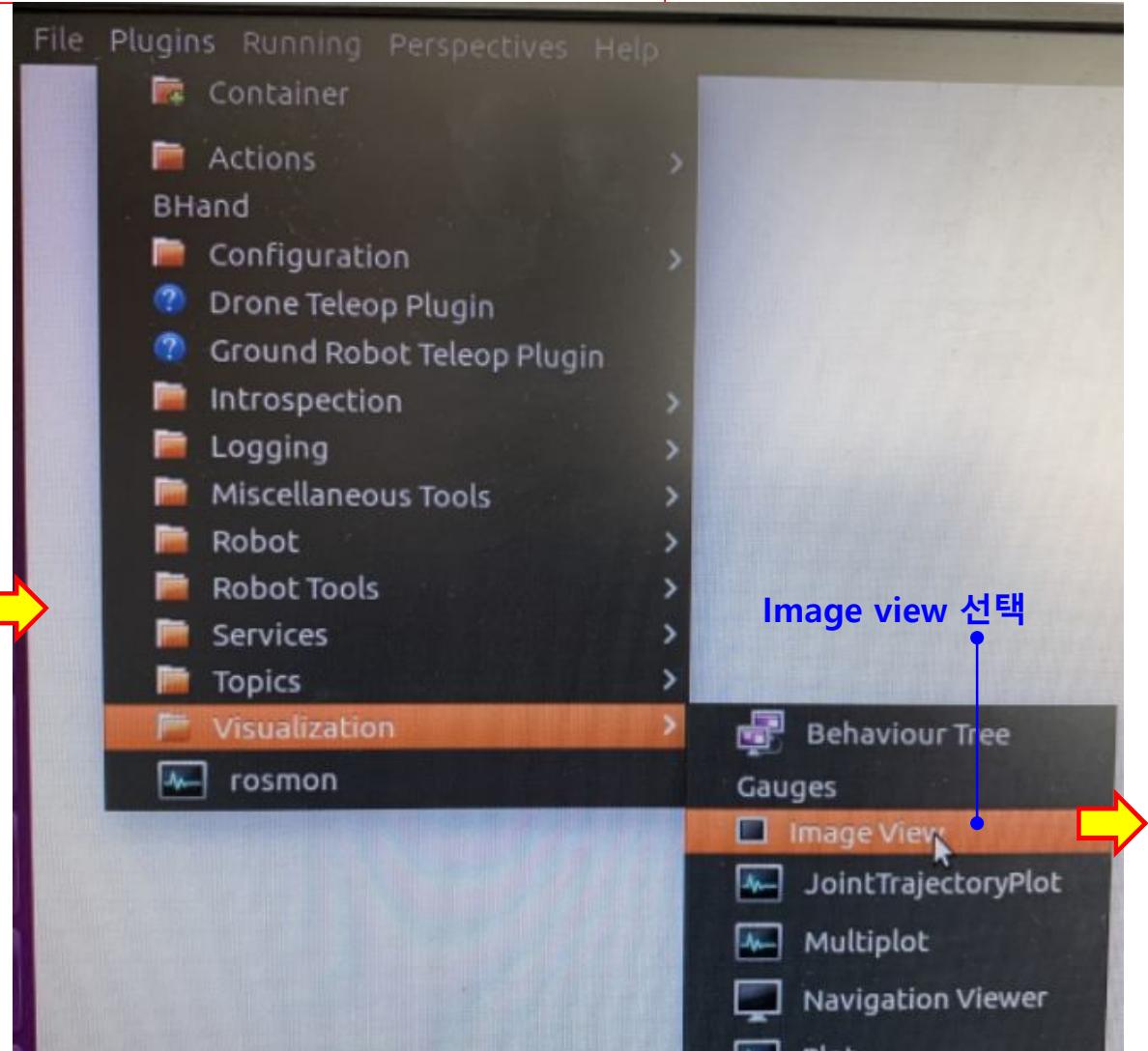
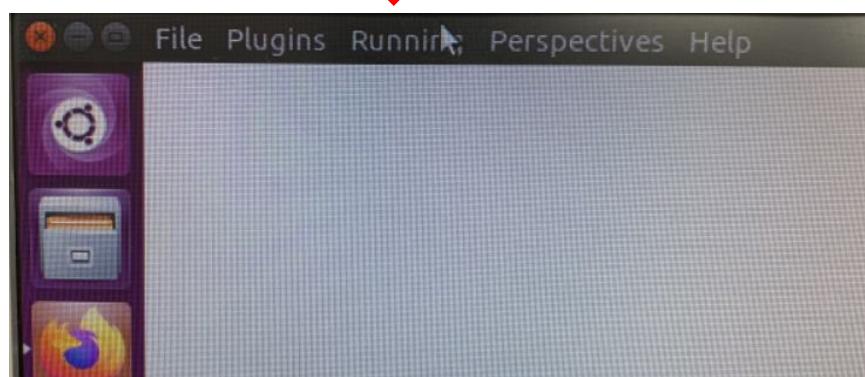
process[camera/image_compensation-1]: started with pid [8935]
process[camera/image_projection-2]: started with pid [8936]
process[camera/image_compensation_projection-3]: started with
pid [8937]
[INFO] [1701680387.627870]: [Image Compensation] Extrinsic Cam
era Calibration Parameter reconfigured to
[INFO] [1701680387.630220]: clip_hist_percent : 1.000000
[INFO] [1701680387.630720]: [Image Compensation] Extrinsic Cam
era Calibration Parameter reconfigured to
[INFO] [1701680387.633281]: clip_hist_percent : 1.000000
[INFO] [1701680387.644049]: [Image Projection] Extrinsic Camer
a Calibration Parameter reconfigured to
[INFO] [1701680387.647934]: top_x : 72, top_y : 4, bottom_x :
115, bottom_y : 120
```

① Remote PC에서 새 터미널 열기

\$ rqt

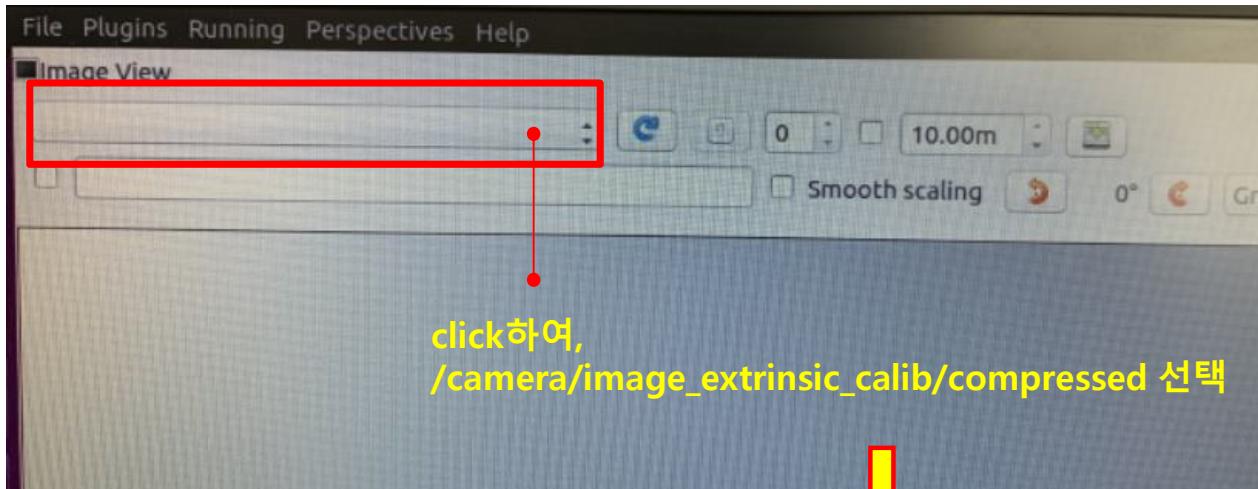


double click 하여 전체 화면으로 만들기



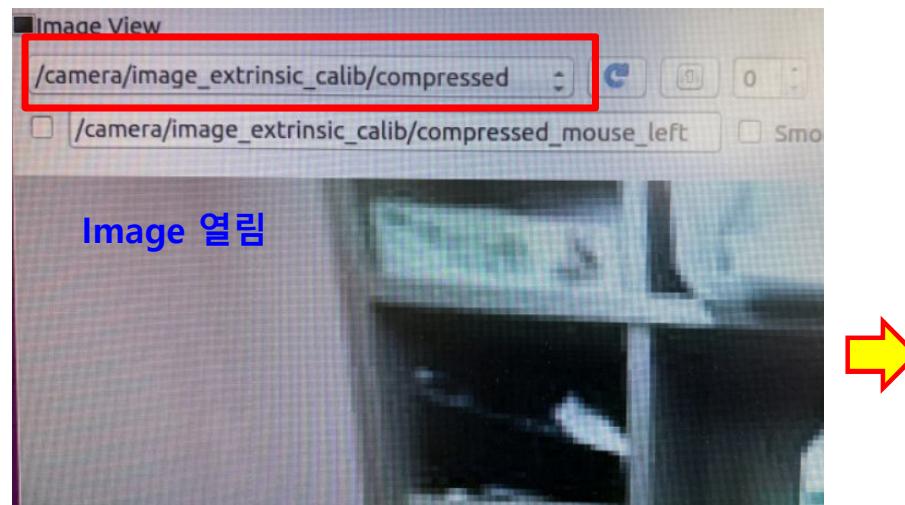
→ 또는 \$ rqt_image_view

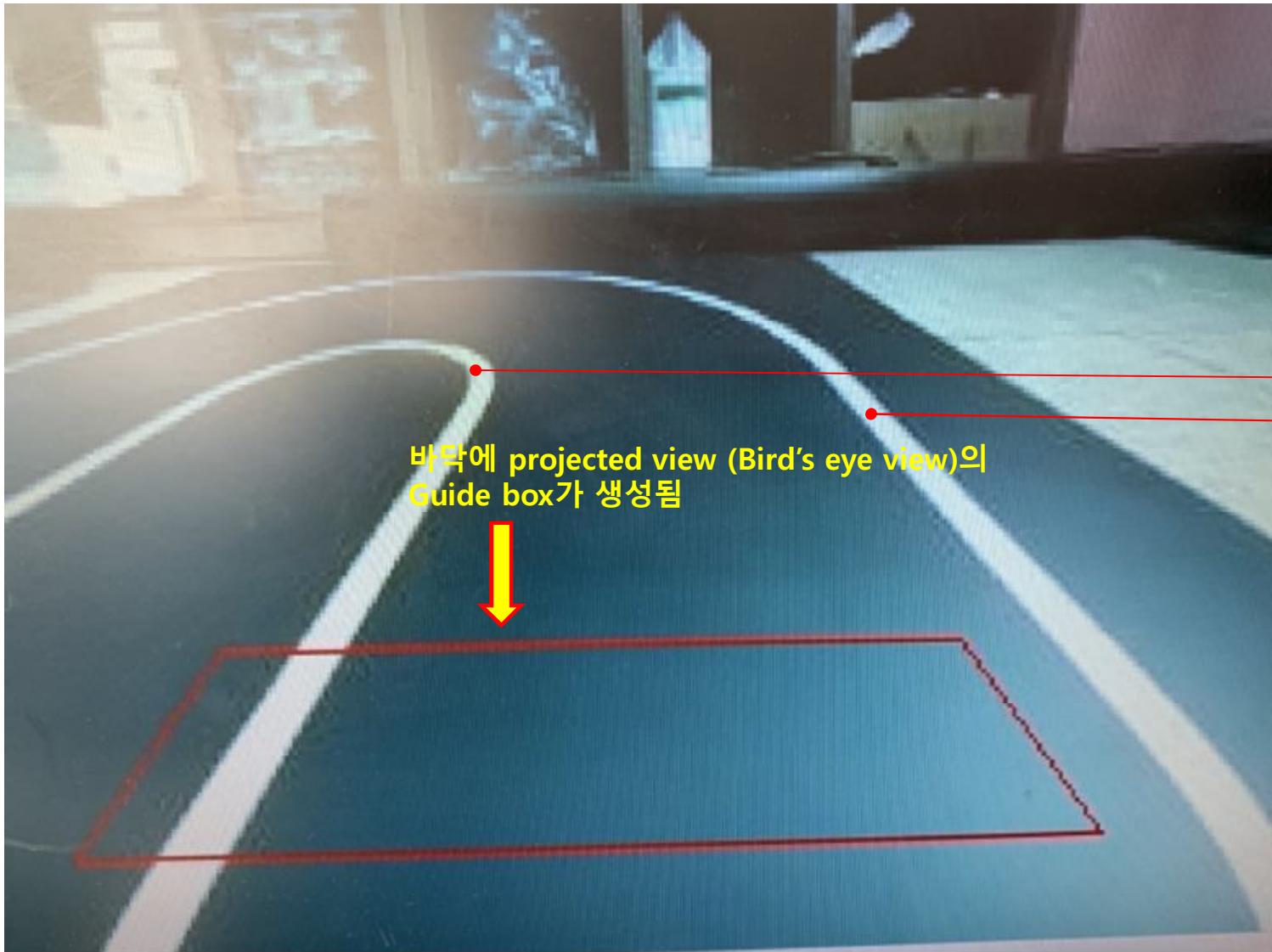
\$ rqt 실행 후, image viewer 선택 후 화면



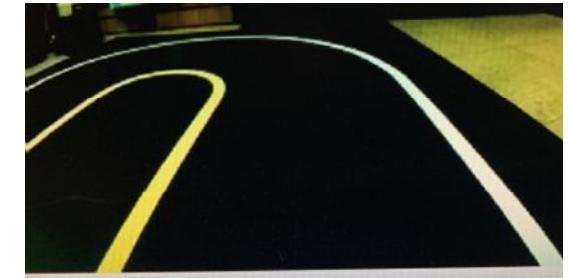
click하여,
/camera/image_extrinsic_calib/compressed 선택

rqt image viewer 초기 화면





주간(외부 조도가 강할 때 실험 영상)



야간(외부 조도가 약할 때 실험 영상)

Yellow 와

White color 구별되지 않음



실험 환경에 따라,



Camera imaging calibration

적용 결과가 다르게 나타남

[주의]

White, yellow lane 인식하는
자율주행 알고리즘

⑨ 외부 조도를 차단하고, 카메라 부착 부위를 조금씩 조정하여 projected view (Bird's eye view)가 중앙에 위치하도록 수정



외부 조도를 차단 전, 카메라 부착 부위 조정 전



외부 조도를 차단, 카메라 부착 부위 조정

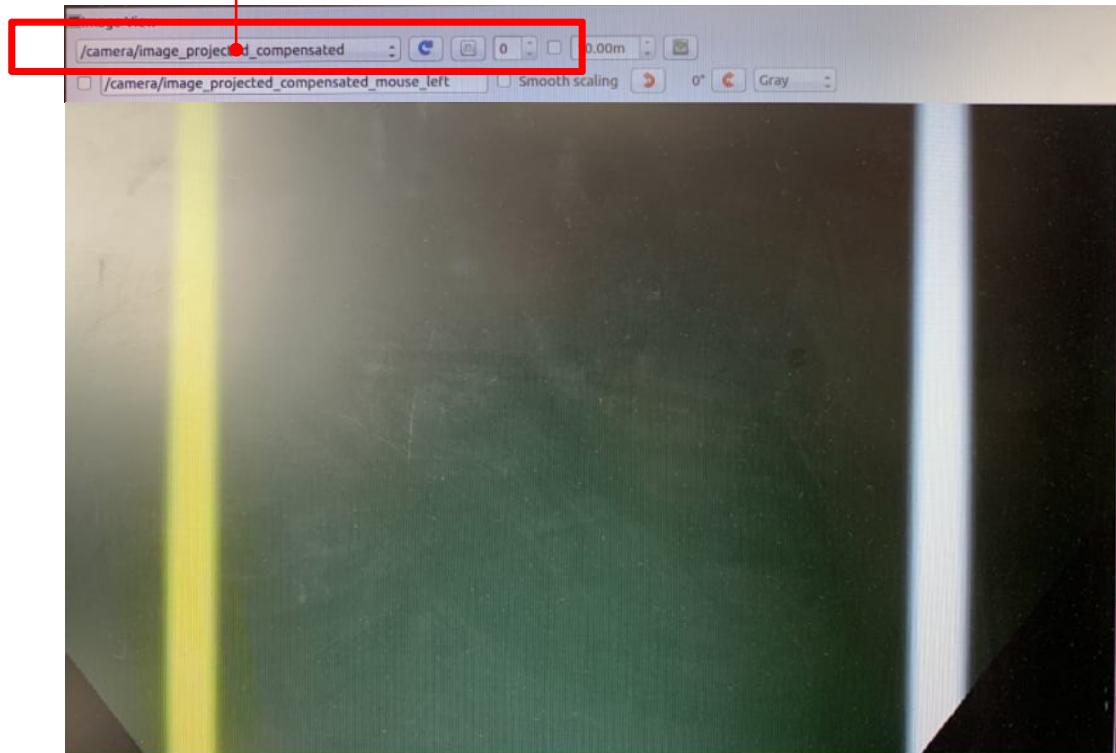
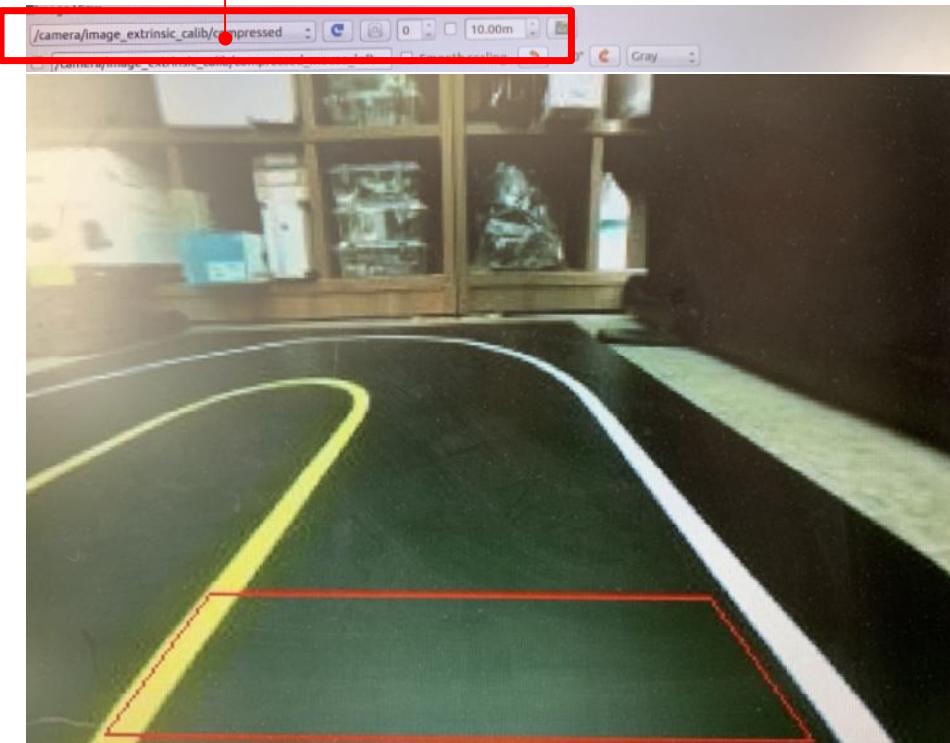
- White 와
- Yellow color 구별
- projected view Guide box
중앙에 위치함

④ rqt image viewer의 topic 변경

/camera/image_extrinsic_calib/compressed



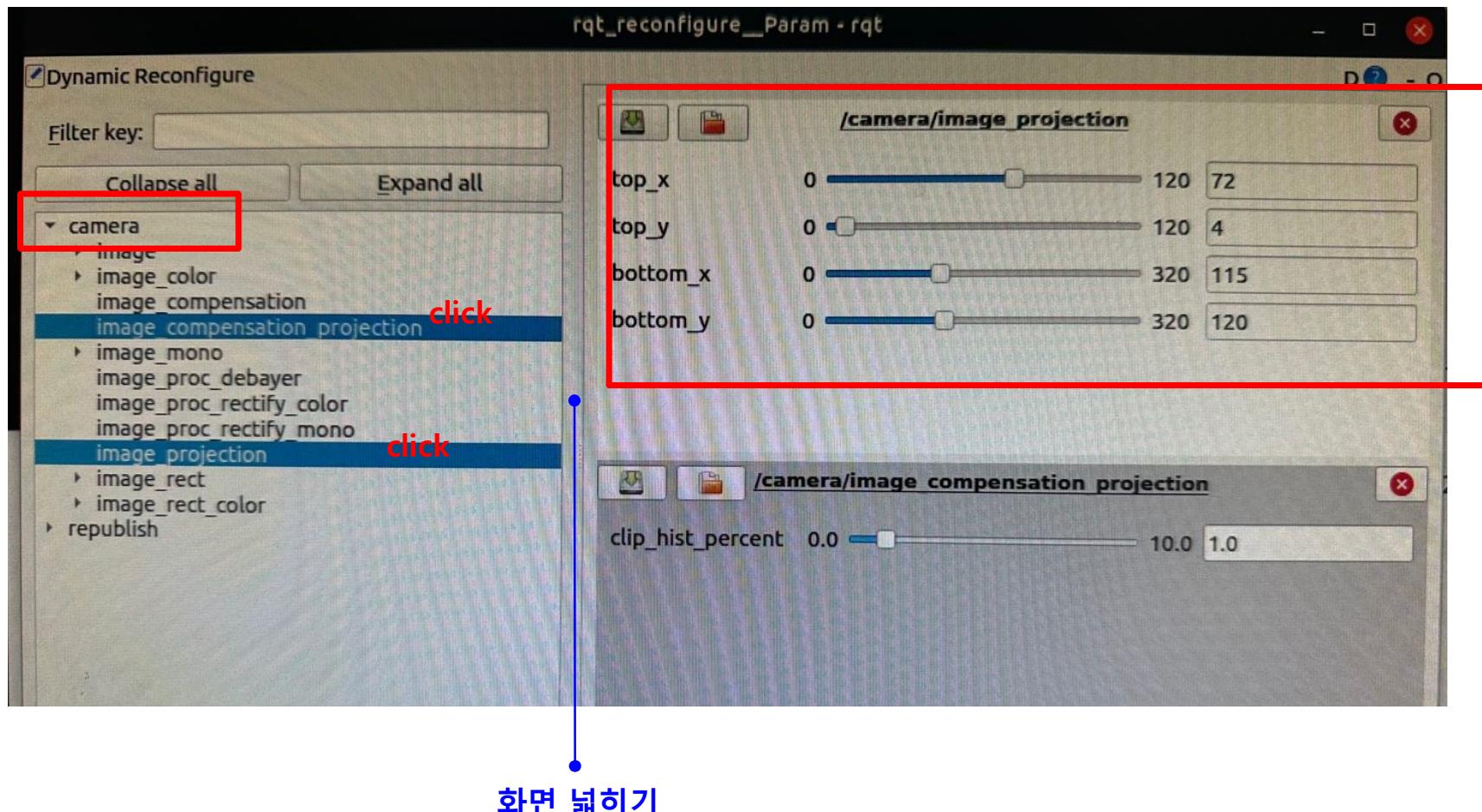
/camera/image_projected_compensated



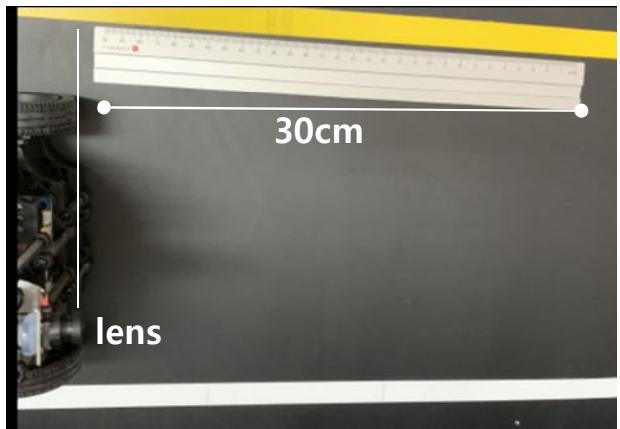
① Remote PC에서 새 터미널 열기

```
$ rosrun rqt_reconfigure rqt_reconfigure
```

: rqt_reconfigure 초기 화면



둘째 image_projection : extrinsic calibration(camera를 통한 lane 검출을 위한 area 설정)

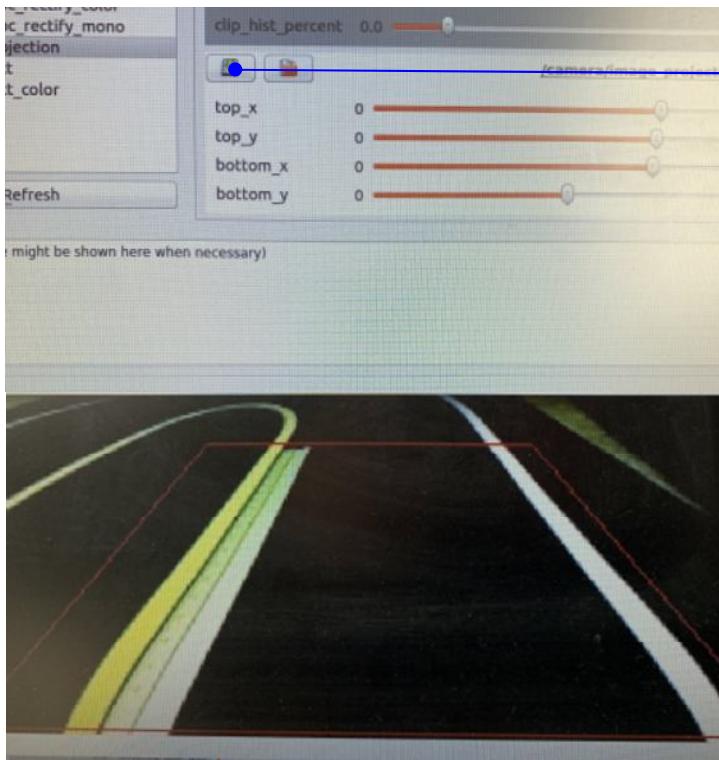


The screenshot shows the ROS rqt_reconfigure interface with several windows open. The main window displays the 'camera' configuration under 'Dynamic Reconfigure'. It includes parameters for 'image', 'image_color', 'image_compensation', and 'image_compensation_projection'. A red box highlights the 'image_compensation_projection' section, which contains four sliders for 'top_x', 'top_y', 'bottom_x', and 'bottom_y'. Below this is another window titled '/camera/image_compensation_projection' showing a slider for 'clip_hist_percent' with a value of 1.0. A red box highlights the numerical values 80, 15, 180, and 120, which correspond to the values set in the 'image_compensation_projection' section. A red arrow points from the text box below to the 'image_compensation_projection' section. Another red arrow points from the bottom right of the 'image_compensation_projection' window to the numerical values. A white box with red text provides instructions: 'projected view Guide box를 위 30cm, 좌우 lane 다 포함되도록 매개변수 변경하기' (Change parameters so that the projected view Guide box covers the top 30cm and both left and right lanes). The bottom right corner shows a terminal window displaying the configuration parameters.

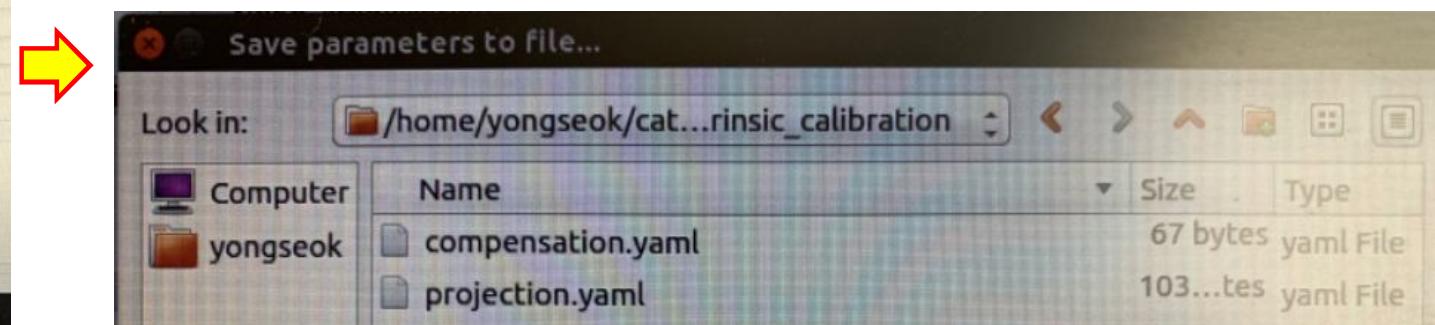
projected view Guide box를
위 30cm, 좌우 lane 다 포함되도록 매개변수 변경하기

```
camera:  
extrinsic_camera_calibration:  
top_x: 80  
top_y: 35  
bottom_x: 161  
bottom_y: 118
```

둘째 image_projection : extrinsic calibration(camera를 통한 lane 검출을 위한 area 설정)



매개변수 수정 후 저장 : click

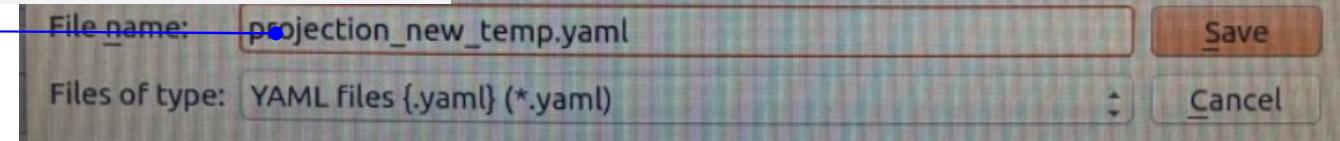


방금 수정한 값을 다른 이름으로 저장 후에, 보정한 값을 projection.yaml 반영해야

→ 경로 catkin_ws / src / turtlebot3_autorace_2020 / turtlebot3_autorace_camera / calibration / extrinsic_calibration



Projection_new_temp.yaml 저장



→ 매개변수 반영하기

경로 catkin_ws / src / turtlebot3_autorace_2020 / turtlebot3_autorace_camera / calibration / **extrinsic_calibration** 의 **projection.yaml**에 반영하기

Projection_new_temp.yaml

```
1 !!python/object/new:dynamic_reconfigure.encoding.Config
2 dictitems:
3     bottom_x: 180
4     bottom_y: 120
5     groups: !!python/object/new:dynamic_reconfigure.encoding.Config
6         dictitems:
7             bottom_x: 180
8             bottom_y: 120
9             groups: !!python/object/new:dynamic_reconfigure.encoding.Config
10                state: []
11                id: 0
12                name: Default
13                parameters: !!python/object/new:dynamic_reconfigure.encoding.Config
14                    state: []
15                parent: 0
16                state: true
17                top_x: 80
18                top_y: 15
19                type: ''
20                state: []
21                top_x: 80
22                top_y: 15
23                state: []
```

```
camera:
  extrinsic_camera_calibration:
    top_x: 80
    top_y: 35
    bottom_x: 161
    bottom_y: 118
```

projection.yaml 수정 후 저장

```
1 ---
2 camera:
3   extrinsic_camera_calibration:
4     top_x: 72
5     top_y: 4
6     bottom_x: 115
7     bottom_y: 120
```

매개변수 반영하기

80
15
180
120

만일 실험을 마치려면, terminal은 CTL+C로 중단 후에 종료

8-2. RPi Camera(G) – Camera calibration test(터미널창 미리 8개 띄우기)

(6) test

: Intrinsic Camera Calibration, Extrinsic Camera Calibration 반영 값 확인하기

ⓐ Remote PC에서 새 터미널 열기

```
$ roscore
```

ⓑ Remote PC에서 새 터미널 열기

```
$ ssh ubuntu@192.168.0.25
```

← Remote PC에서 TurtleBot3 Burger(Raspberry pi)을 원격접속

pw : turtlebot 입력

```
~$ roslaunch turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
```

ⓒ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_camera intrinsic_camera_calibration.launch mode:=action
```

Calibration에서 action 바뀜

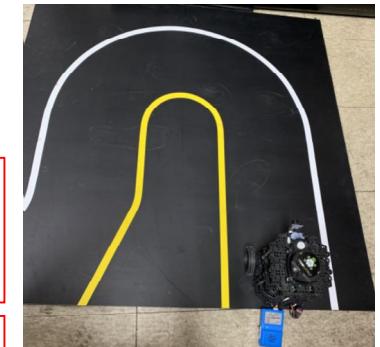
ⓓ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_camera extrinsic_camera_calibration.launch mode:=action
```

Calibration에서 action 바뀜

ⓕ Remote PC에서 새 터미널 열기

```
$ rqt
```



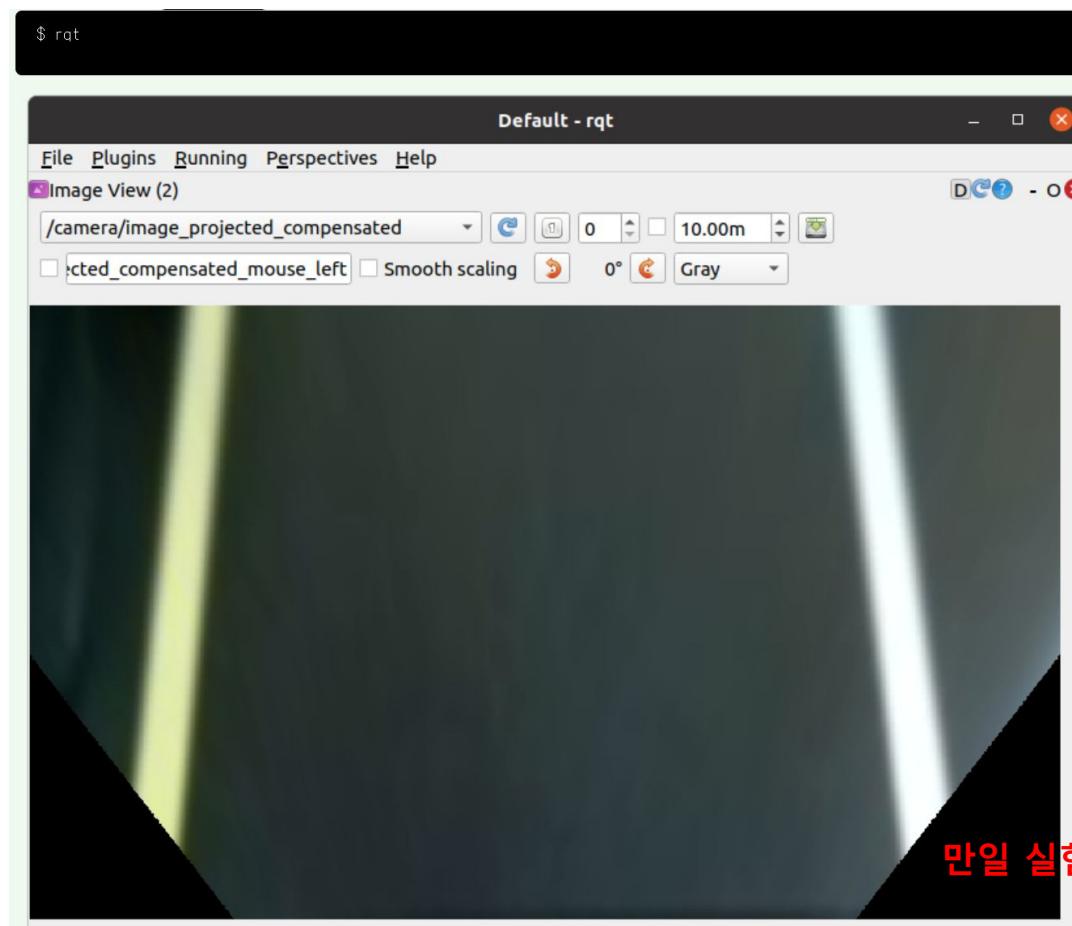
8-2. RPi Camera(G) – Camera calibration test

(6) test

: Intrinsic Camera Calibration, Extrinsic Camera Calibration 반영 값 확인하기

⑤ Remote PC에서 새 터미널 열기

\$ rqt

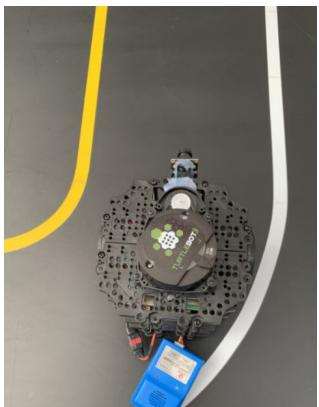


8. Autonomous Driving

8-3. Lane detection

(1) Setting : Place TurtleBot3 between yellow and white lanes

→ yellow(left) – Turtlebot3 burger – white(right)



8. 3. Lane Detection

Lane detection package that runs on the **Remote PC** receives camera images either from TurtleBot3 or Gazebo simulation to drive the Turtlebot3 along them.

The following instructions describe how to use and calibrate the lane detection feature via rqt.

1. Place the TurtleBot3 inbetween yellow and white lanes.

NOTE: The lane detection filters yellow on the left side while filters white on the right side. Be sure that the yellow lane is on the left side of the robot.

in Gazebo simulation.

2. Open a new terminal and launch Autorace Gazebo simulation. The **roscore** will be automatically launched with the command.

```
$ roslaunch turtlebot3_gazebo turtlebot3_autorace_2020.launch
```

3. Open a new terminal and launch the intrinsic calibration node.

```
$ roslaunch turtlebot3_autorace_camera intrinsic_camera_calibration.launch
```

4. Open a new terminal and launch the extrinsic calibration node.

```
$ roslaunch turtlebot3_autorace_camera extrinsic_camera_calibration.launch
```

5. Open a new terminal and launch the lane detection calibration node.

```
$ roslaunch turtlebot3_autorace_detect detect_lane.launch mode:=calibration
```

6. Open a new terminal and launch the rqt.

```
$ rqt
```

Battery 사전에 충전하기(미충전 경우 Buzzer 소리 발생)

→ 방지 시 완전 방전됨

Detect lane 실행 시 반드시 battery 충전되어 있어야 함



실행

Click to expand : How to Perform Lane Detection with Actual TurtleBot3?

Lane detection package allows Turtlebot3 to drive between two lanes without external influence.

The following instructions describe how to use the lane detection feature and to calibrate camera via rqt.

1. Place TurtleBot3 between yellow and white lanes.

NOTE: Be sure that yellow lane is placed left side of the robot and White lane is placed right side of the robot.

2. Launch roscore on **Remote PC**.

```
$ roscore
```

3. Trigger the camera on **SBC**.

```
$ roslaunch turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
```

4. Run a intrinsic camera calibration launch file on **Remote PC**.

8-3. Lane detection(터미널창 미리 8개 띄우기)

ⓐ Remote PC에서 새 터미널 열기

```
$ roscore
```

ⓑ Remote PC에서 새 터미널 열기

```
$ ssh ubuntu@192.168.0.25      ← Remote PC에서 TurtleBot3 Burger(Raspberry pi)을 원격접속  
pw : turtlebot 입력  
~$ roslaunch turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
```

ⓒ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_camera intrinsic_camera_calibration.launch mode:=action
```

ⓓ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_camera extrinsic_camera_calibration.launch mode:=action
```

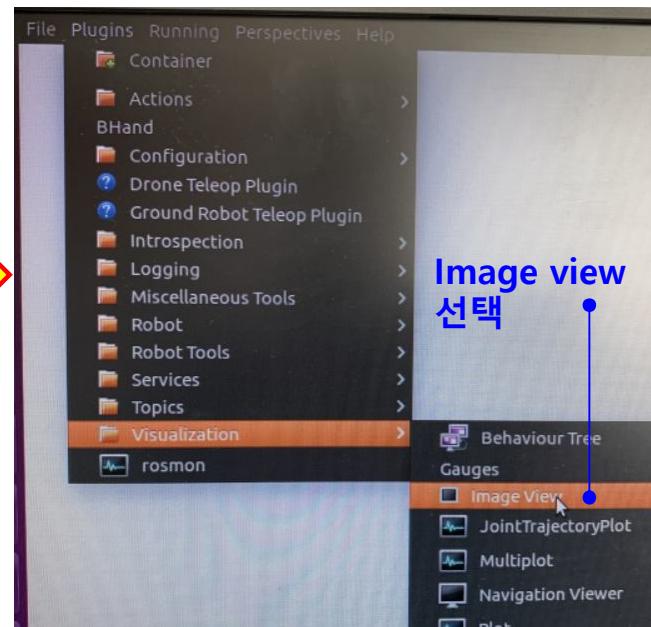
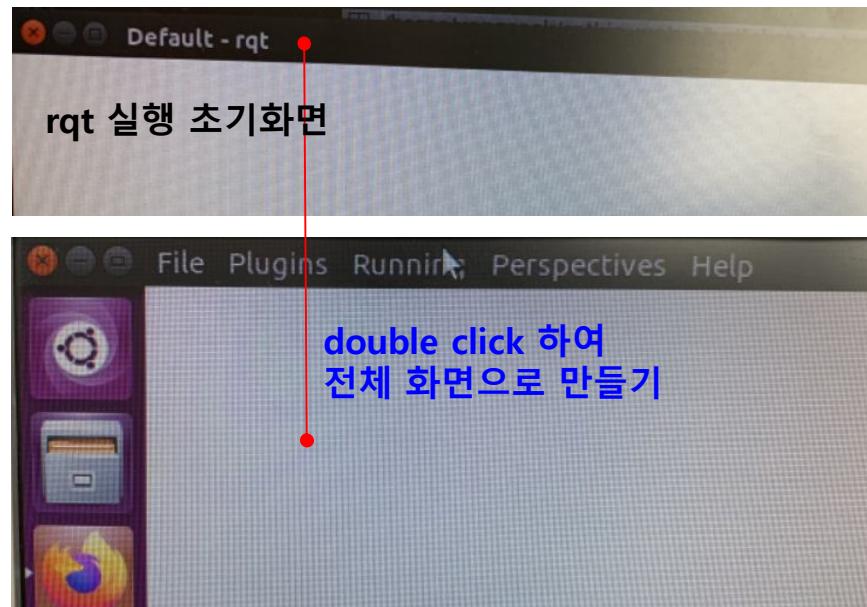
ⓕ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_detect detect_lane.launch mode:=calibration
```

Remote PC에서 새 터미널 열기

```
$ rqt
```

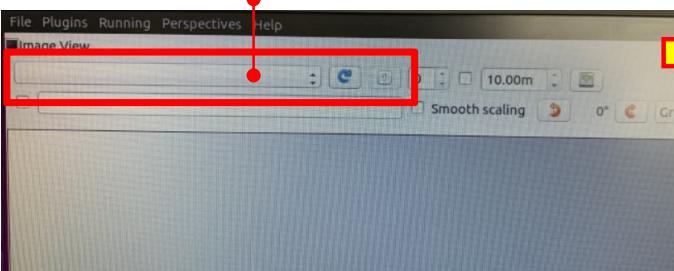
\$ rqt



plugins > visualization > Image view 3번 반복 ☆

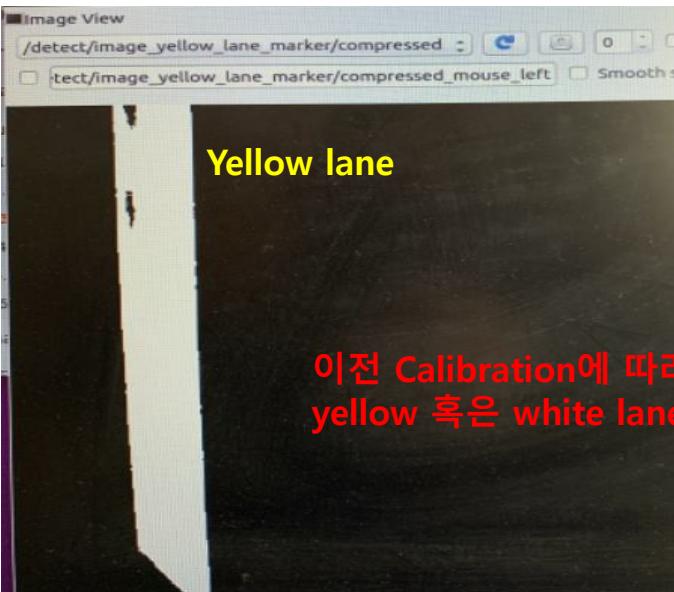
click하여, 아래 선택

/detect/image_yellow_lane_marker/compressed



Yellow lane

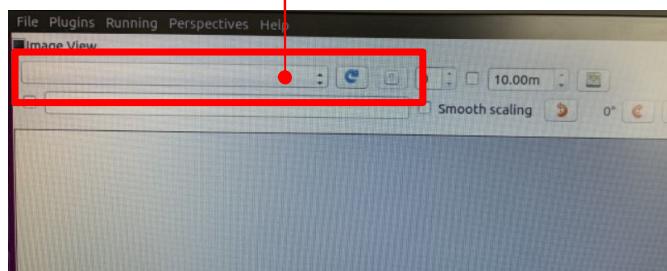
이전 Calibration에 따라, 외부 조도 환경에 따라
yellow 혹은 white lane이 안보일 수 있음



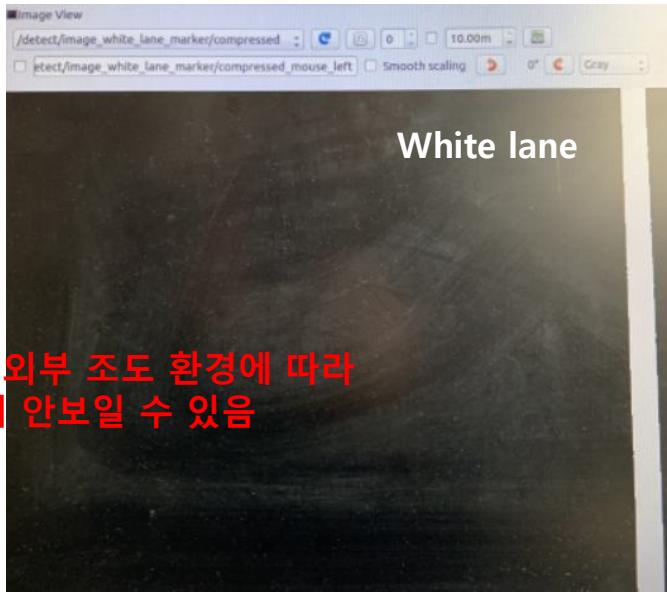
rqt는 multi window를 지원하기에 세개의 image를 실행하여 window에 적절히 배치하기

click하여, 아래 선택

/detect/image_white_lane_marker/compressed

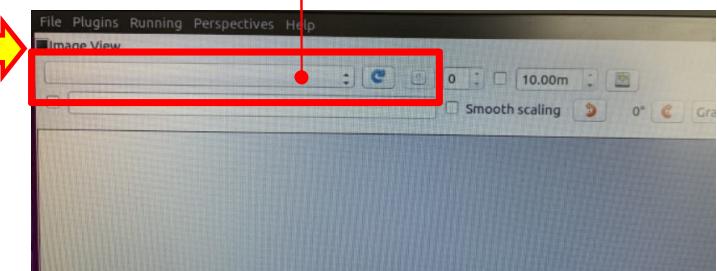


White lane



click하여, 아래 선택

/detect/image_lane/compressed



Red Yellow

Yellow

Cyan White

rqt 3개 image 실행 배치 화면

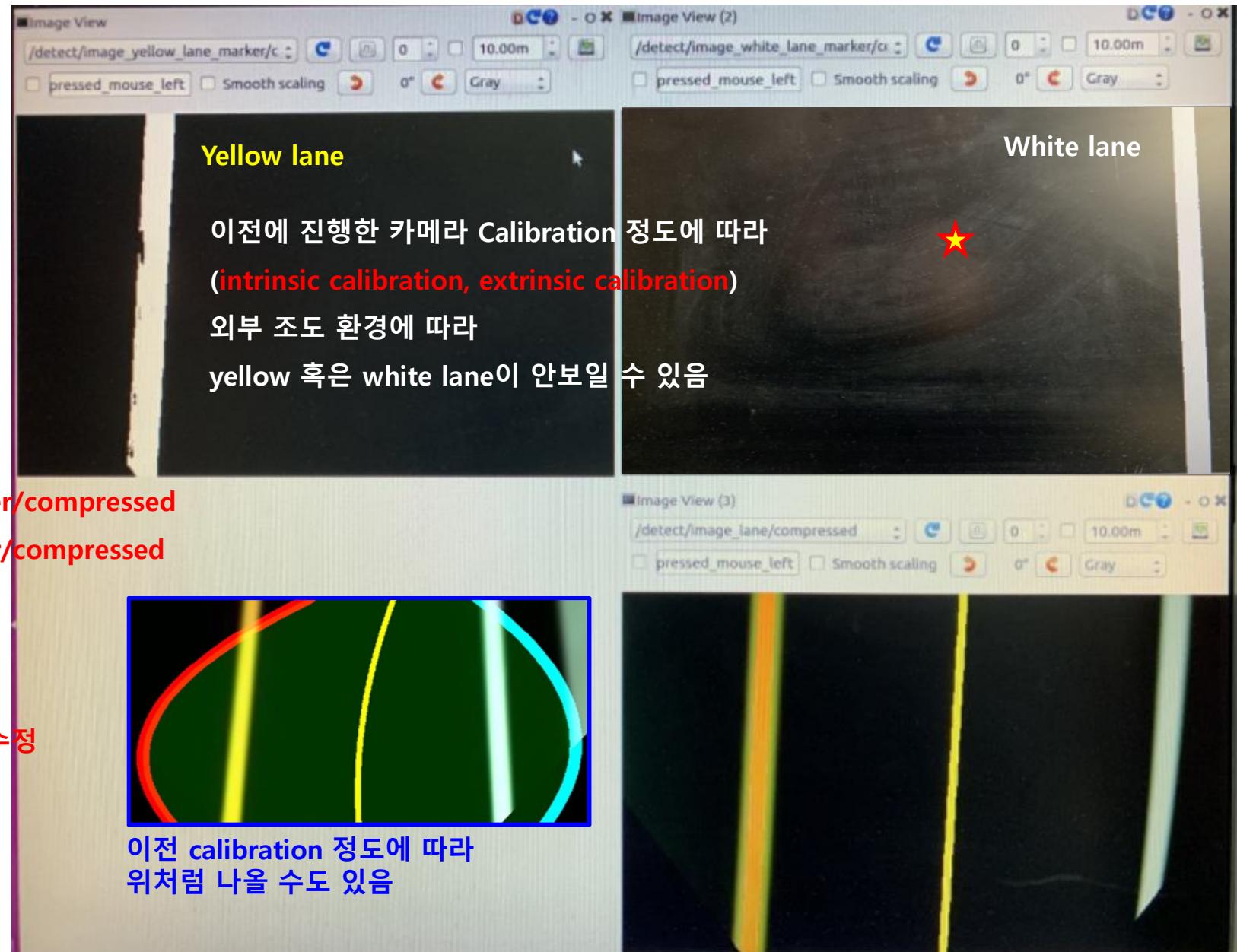
/detect/image_yellow_lane_marker/compressed

/detect/image_white_lane_marker/compressed

/detect/image_lane/compressed

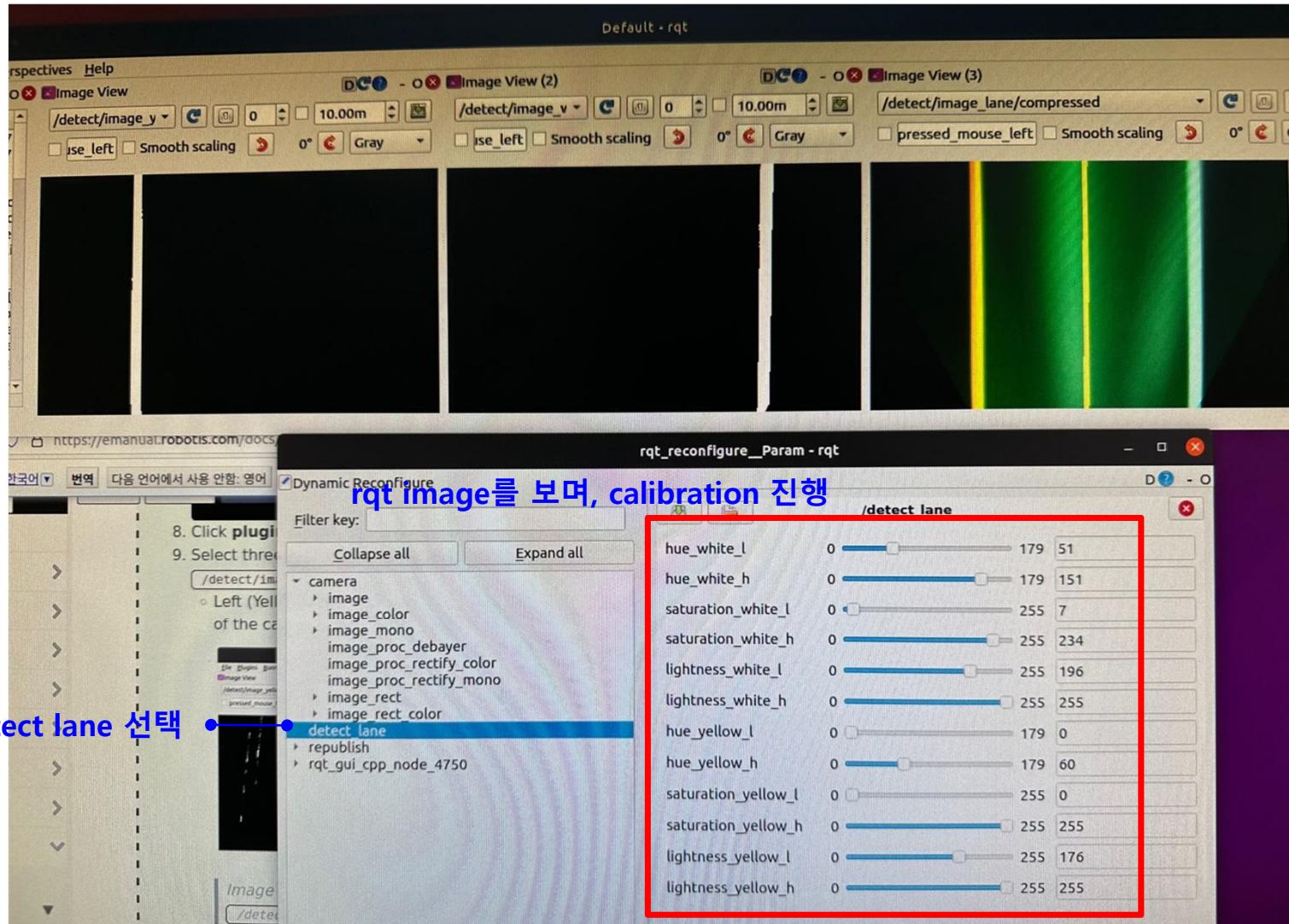
/detect/image_lane/compressed

화면이 안보이면, 다음 페이지 값 수정

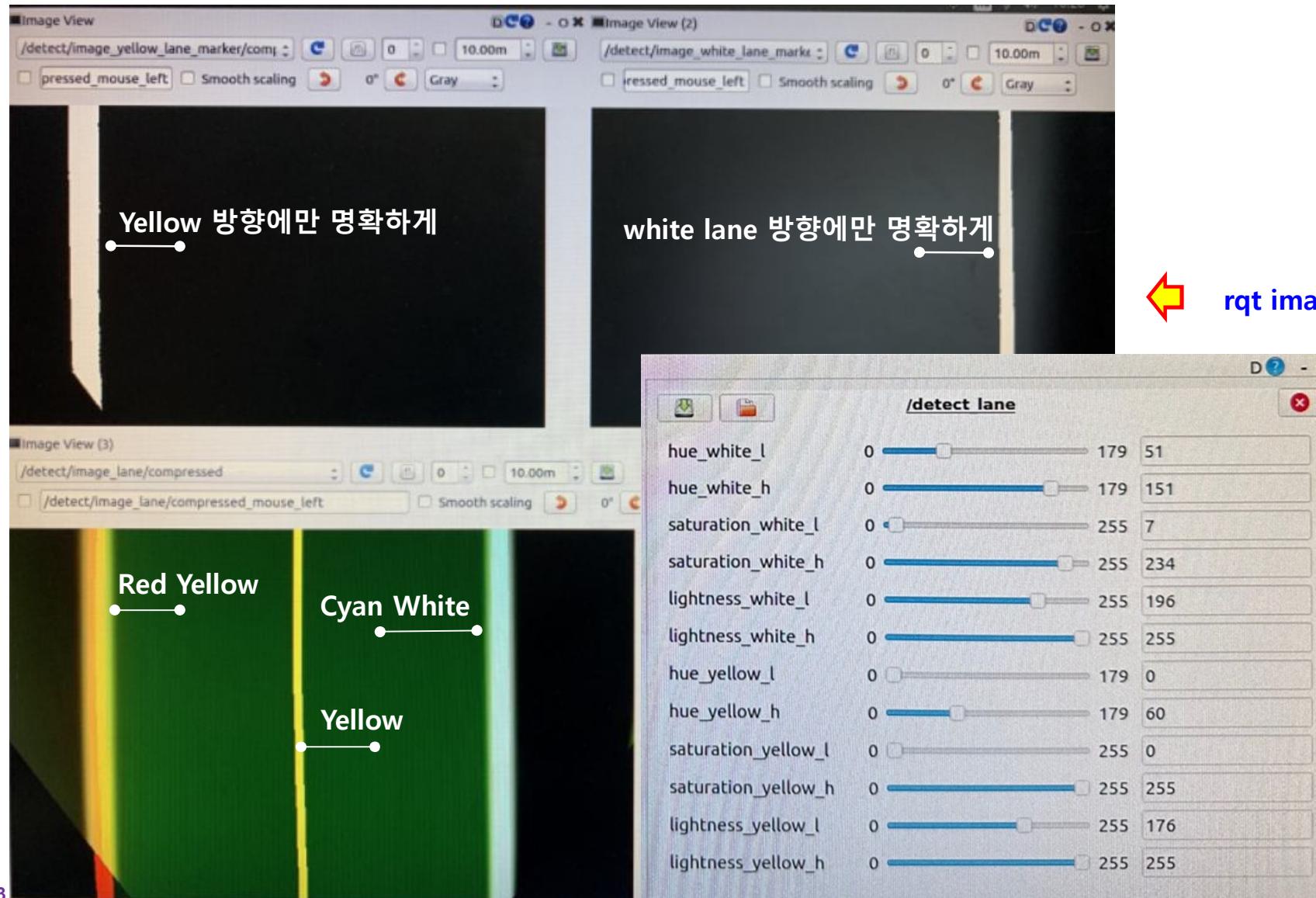


⑨ Remote PC에서 새 터미널 열기

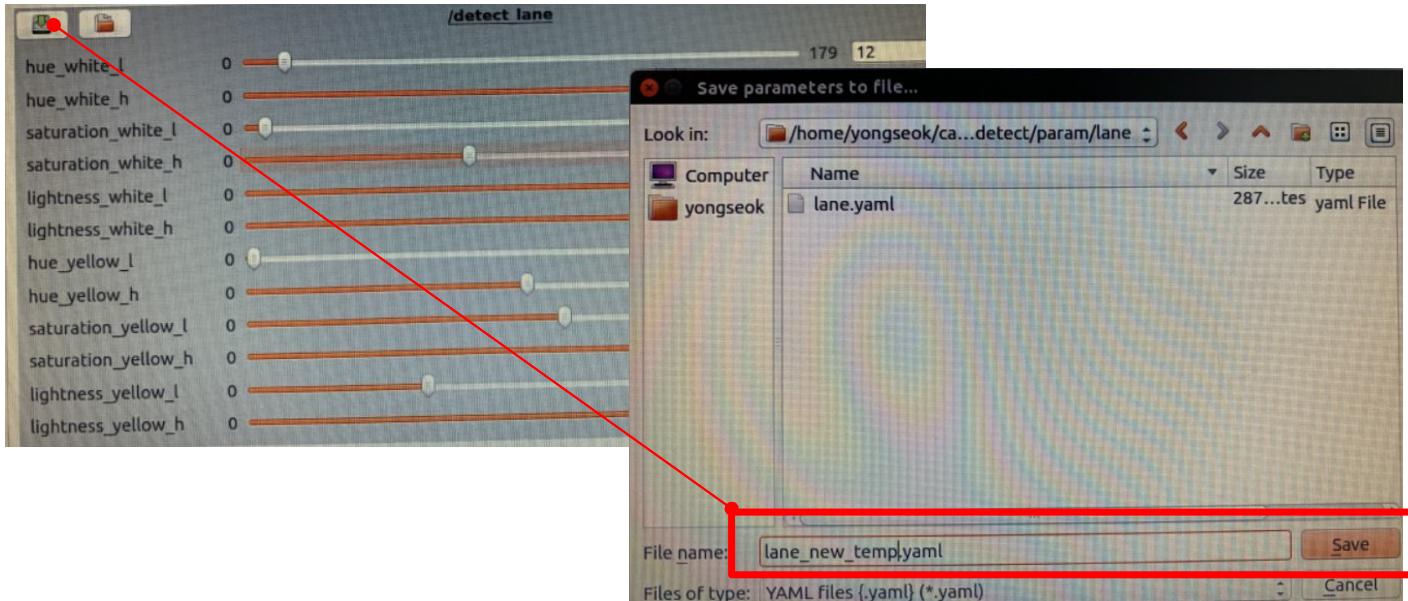
\$ rosrun rqt_reconfigure rqt_reconfigure



④ rqt image를 보며, calibration 진행



① 매개변수 반영(저장)하기



다른 이름(lane_new_temp)으로 저장 후
변경 값을 lane.yaml 반영위해 tmepl로 저장

→ 경로

[catkin_ws / src / turtlebot3_autorace_2020 / turtlebot3_autorace_detect / param / lane](#)

→ 매개변수 반영하기

→ 경로

catkin_ws / src / turtlebot3_autorace_2020 / turtlebot3_autorace_detect / param / lane

temp.yaml

```
1 !!python/object/new:dynamic_reconfigure.encoding.Config
2 dictitems:
3   groups: !!python/object/new:dynamic_reconfigure.encoding.Config
4     dictitems:
5       groups: !!python/object/new:dynamic_reconfigure.encoding.Config
6         state: []
7         hue_white_h: 151
8         hue_white_l: 51
9         hue_yellow_h: 60
10        hue_yellow_l: 0
11        id: 0
12        lightness_white_h: 255
13        lightness_white_l: 196
14        lightness_yellow_h: 255
15        lightness_yellow_l: 176
16        name: Default
17        parameters: !!python/object/new:dynamic_reconfigure.encoding.Config
18          state: []
19        parent: 0
20        saturation_white_h: 234
21        saturation_white_l: 7
22        saturation_yellow_h: 255
23        saturation_yellow_l: 0
24        state: true
25        type:
26        state: []
27        hue_white_h: 151
28        hue_white_l: 51
29        hue_yellow_h: 60
30        hue_yellow_l: 0
31        lightness_white_h: 255
32        lightness_white_l: 196
33        lightness_yellow_h: 255
34        lightness_yellow_l: 176
35        saturation_white_h: 234
36        saturation_white_l: 7
37        saturation_yellow_h: 255
38        saturation_yellow_l: 0
39 state:
```

```
1 ---
2 detect:
3   lane:
4     white:
5       hue_l: 0
6       hue_h: 179
7       saturation_l: 0
8       saturation_h: 70
9       lightness_l: 105
10      lightness_h: 255
11      yellow:
12        hue_l: 10
13        hue_h: 127
14        saturation_l: 70
15        saturation_h: 255
16        lightness_l: 95
17        lightness_h: 255
```

```
1 ---
2 detect:
3   lane:
4     white:
5       hue_l: 9
6       hue_h: 128
7       saturation_l: 7
8       saturation_h: 94
9       lightness_l: 126
10      lightness_h: 255
11      yellow:
12        hue_l: 0
13        hue_h: 62
14        saturation_l: 0
15        saturation_h: 196
16        lightness_l: 145
17        lightness_h: 255
```

lane.yaml

lane.yaml 수정 후 저장

```
1 ---
2 detect:
3   lane:
4     white:
5       hue_l: 51
6       hue_h: 151
7       saturation_l: 7
8       saturation_h: 234
9       lightness_l: 196
10      lightness_h: 255
11      yellow:
12        hue_l: 0
13        hue_h: 60
14        saturation_l: 0
15        saturation_h: 255
16        lightness_l: 176
17        lightness_h: 255
```

만일 실험을 마치려면, 모든 terminal은 CTL+C로 중단 후에 종료

8-3. Lane detection 정리

ⓐ Remote PC에서 새 터미널 열기

```
$ roscore
```

ⓑ Remote PC에서 새 터미널 열기

```
$ ssh ubuntu@192.168.0.25
```

pw : turtlebot 입력

```
~$ rosrun turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
```

```
ubuntu@ubuntu1:~$ rosrun turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
... logging to /home/ubuntu/.ros/log/d68743cc-9313-11ee-b3dc-496b540f778a/roslaunch-ubuntu1-1136.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.1.145:33503/
SUMMARY
=====
PARAMETERS
* /cv_camera/camera_info_url: package://turtleb...
* /cv_camera/frame_id: camera
* /cv_camera/image_height: 240
* /cv_camera/image_width: 320
* /cv_camera/rate: 30
* /rosdistro: noetic
* /rosversion: 1.16.0

NODES
/
  cv_camera (cv_camera/cv_camera_node)

ROS_MASTER_URI=http://192.168.1.141:11311

process[cv_camera-1]: started with pid [1152]
[ WARN:0] global ..../modules/videoio/src/cap_gstreamer.cpp (480)
isPipelinePlaying OpenCV | GStreamer warning: GStreamer: pipeline have not been created
[ INFO] [1701742495.704572783]: camera calibration URL: package://turtlebot3_autorace_camera/calibration/intrinsic_calibration/camerav2_320x240_30fps.yaml
```

8-3. Lane detection 정리

④ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_camera intrinsic_camera_calibration.launch mode:=action
```

```
[yongseok@yongseok: ~]$ roslaunch turtlebot3_autorace_2020/turtlebot3_autorace_camera/launch/intrinsic_camera_calibration.launch mode:=action
... logging to /home/yongseok/.ros/log/d68743cc-9313-11ee-b3dc-496b540f778a/roslaun[yongseok-7442.log]
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.1.141:36331/

SUMMARY
=====

PARAMETERS
* /camera/image_proc/queue_size: 20
* /rosdistro: noetic
* /rosversion: 1.16.0

NODES
/
  relay_camera_info (topic_tools/relay)
  republish (image_transport/republish)
/camera/
  image_proc (image_proc/image_proc)

ROS_MASTER_URI=http://192.168.1.141:11311

process[republish-1]: started with pid [7456]
process[relay_camera_info-2]: started with pid [7457]
process[camera/image_proc-3]: started with pid [7458]
```

8-3. Lane detection 정리

④ Remote PC에서 새 터미널 열기

```
$ rosrun turtlebot3_autorace_camera extrinsic_camera_calibration.launch mode:=action
```

```
yongseok@yongseyongseok@yongseyongseok@yongseok:~$ rosrun turtlebot3_autorace_camera extrinsic_camera_calibration.launch mode:=action
... logging to /home/yongseok/.ros/log/d68743cc-9313-11ee-b3dc-496b540f778a/roslaunch-yongseok-7482.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.1.141:45895/

SUMMARY
=====

PARAMETERS
  * /camera/image_compensation/camera/extrinsic_camera_calibration/clip_hist_percent: 1.0
  * /camera/image_compensation/is_extrinsic_camera_calibration_mode: False
  * /camera/image_compensation_projection/camera/extrinsic_camera_calibration/clip_hist_percent: 1.0
  * /camera/image_compensation_projection/is_extrinsic_camera_calibration_mode: False
  * /camera/image_projection/camera/extrinsic_camera_calibration/bottom_x: 180
  * /camera/image_projection/camera/extrinsic_camera_calibration/bottom_y: 120
  * /camera/image_projection/camera/extrinsic_camera_calibration/top_x: 80
  * /camera/image_projection/camera/extrinsic_camera_calibration/top_y: 15
  * /camera/image_projection/is_extrinsic_camera_calibration_mode: False
  * /rosdistro: noetic
  * /rosversion: 1.16.0

NODES
  /camera/
    image_compensation (turtlebot3_autorace_camera/image_compensation)
    image_compensation_projection (turtlebot3_autorace_camera/image_compensation)
    image_projection (turtlebot3_autorace_camera/image_projection)

ROS_MASTER_URI=http://192.168.1.141:11311

process[camera/image_compensation-1]: started with pid [7496]
process[camera/image_projection-2]: started with pid [7497]
process[camera/image_compensation_projection-3]: started with pid [7498]
INFO: cannot create a symlink to latest log directory: [Errno 17] File exists: '/home/yongseok/.ros/log/d68743cc-9313-11ee-b3dc-496b540f778a' -> '/home/yongseok/.ros/log/latest'
```

① Remote PC에서 새 터미널 열기

```
$ rosrun turtlebot3_autorace_detect detect_lane.launch mode:=calibration
```

```
[yongseok@yongseok: ~]$ rosrun turtlebot3_autorace_detect detect_lane.launch mode:=action
... logging to /home/yongseok/.ros/log/d68743cc-9313-11ee-b3dc-496b540f778a/rosrun-yongseok-7555.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started rosrun server http://192.168.1.141:44509/
SUMMARY
=====
PARAMETERS
* /detect_lane/detect/lane/white/hue_h: 151
* /detect_lane/detect/lane/white/hue_l: 51
* /detect_lane/detect/lane/white/lightness_h: 255
* /detect_lane/detect/lane/white/lightness_l: 196
* /detect_lane/detect/lane/white/saturation_h: 234
* /detect_lane/detect/lane/white/saturation_l: 7
* /detect_lane/detect/lane/yellow/hue_h: 60
* /detect_lane/detect/lane/yellow/hue_l: 0
* /detect_lane/detect/lane/yellow/lightness_h: 255
* /detect_lane/detect/lane/yellow/lightness_l: 176
* /detect_lane/detect/lane/yellow/saturation_h: 255
* /detect_lane/detect/lane/yellow/saturation_l: 0
* /detect_lane/is_detection_calibration_mode: False
* /rosdistro: noetic
* /rosversion: 1.16.0
NODES
/
    detect_lane (turtlebot3_autorace_detect/detect_lane)
ROS_MASTER_URI=http://192.168.1.141:11311
process[detect_lane-1]: started with pid [7569]
```

8-3. Lane detection 최종 실습 (터미널창 미리 8개 띄우기) – 방법1 (turtlebot3 느리게 이동)

ⓐ Remote PC에서 새 터미널 열기

```
$ roscore
```

ⓑ 새 터미널 열기

```
$ ssh ubuntu@192.168.0.25      ← Remote PC에서 TurtleBot3 Burger(Raspberry pi)을 원격접속  
pw : turtlebot 입력  
~$ roslaunch turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
```

ⓑ 새 터미널 열기

```
$ ssh ubuntu@192.168.0.25  
~$ roslaunch turtlebot3_bringup turtlebot3_robot.launch      ← TurtleBot3 Burger bringup 기다리기
```

ⓒ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_camera intrinsic_camera_calibration.launch mode:=action
```

ⓓ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_camera extrinsic_camera_calibration.launch mode:=action
```

```
roslaunch turtlebot3_autorace_driving turtlebot3_autorace_control_lane.launch
```

⑤ Remote PC에서 새 터미널 열기

```
$ rosrun turtlebot3_autorace_detect detect_lane.launch mode:=action
```

```
started roslaunch server http://192.168.1.141:44737/
SUMMARY
=====
PARAMETERS
* /detect_lane/detect/lane/white/hue_h: 151
* /detect_lane/detect/lane/white/hue_l: 51
* /detect_lane/detect/lane/white/lightness_h: 255
* /detect_lane/detect/lane/white/lightness_l: 196
* /detect_lane/detect/lane/white/saturation_h: 234
* /detect_lane/detect/lane/white/saturation_l: 7
* /detect_lane/detect/lane/yellow/hue_h: 60
* /detect_lane/detect/lane/yellow/hue_l: 0
* /detect_lane/detect/lane/yellow/lightness_h: 255
* /detect_lane/detect/lane/yellow/lightness_l: 176
* /detect_lane/detect/lane/yellow/saturation_h: 255
* /detect_lane/detect/lane/yellow/saturation_l: 0
* /detect_lane/is_detection_calibration_mode: False
* /rosdistro: noetic
* /rosversion: 1.16.0

NODES
/
  detect_lane (turtlebot3_autorace_detect/detect_lane)

ROS_MASTER_URI=http://192.168.1.141:11311
process[detect_lane-1]: started with pid [9615]
```

⑥ Remote PC에서 새 터미널 열기

```
$ rosrun turtlebot3_autorace_driving turtlebot3_autorace_control_lane.launch
```



Turtlebot은 lane을 detecting하며 자동으로 이동함

```

/home/yongseok/catkin_ws/src/turtlebot3_autorace_2020/turtlebot3_autorace_driving/launch/turtlebot3_autorace_control_lane.launch http://192.168.1.141:11311
yongseok@yongseok:~$ roscore
... logging to /home/yongseok/.ros/log/3720a280-9339-11ee-b3dc-496b540f778a/roslaun[yongseok-9994.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.1.141:43009/
ros_comm version 1.16.0

```

1. roscore

```

SUMMARY
=====
PARAMETERS
* /rosdistro: noetic
* /rosversion: 1.16.0
NODES

```

40 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at <https://ubuntu.com/esm>

2 \$ ssh ubuntu@192.168.0.25

```

~$ rosrun turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
Last login: Tue Dec 5 04:57:59 2023 from 192.168.1.141
ubuntu@ubuntu1:~$ rosrun turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
... logging to /home/ubuntu/.ros/log/3720a280-9339-11ee-b3dc-496b540f778a/roslaun[yongseok-1068.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.


```

3 \$ ssh ubuntu@192.168.0.25

```

~$ rosrun turtlebot3_bringup turtlebot3_robot.launch
SUMMARY
=====
PARAMETERS
* /rosdistro: noetic
* /rosversion: 1.16.0

```

4. Intrinsic camera

```

/home/yongseok/catkin_ws/src/turtlebot3_autorace_2020/turtlebot3_autorace_driving/launch/turtlebot3_autorace_control_lane.launch http://192.168.1.141:11311
yongseok@yongseok:~$ rosrun turtlebot3_autorace_camera intrinsic_camera_calibration.launch mode:=action
... logging to /home/yongseok/.ros/log/3720a280-9339-11ee-b3dc-496b540f778a/roslaun[yongseok-10056.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started rosrun turtlebot3_autorace_camera intrinsic_camera_calibration.launch mode:=action
SUMMARY
=====
```

5. Extrinsic camera

```

yongseok@yongseok:~$ rosrun turtlebot3_autorace_detect detect_lane.launch mode:=action
... logging to /home/yongseok/.ros/log/3720a280-9339-11ee-b3dc-496b540f778a/roslaun[yongseok-10094.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started rosrun turtlebot3_autorace_detect detect_lane.launch mode:=action
SUMMARY
=====
```

6. detect_lane

```

yongseok@yongseok:~$ rosrun turtlebot3_autorace_detect detect_lane.launch mode:=action
... logging to /home/yongseok/.ros/log/3720a280-9339-11ee-b3dc-496b540f778a/roslaun[yongseok-10160.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started rosrun turtlebot3_autorace_detect detect_lane.launch mode:=action
SUMMARY
=====
```

7. rosrun turtlebot3_autorace_driving turtlebot3_autorace_control_lane.launch

```

yongseok@yongseok:~$ rosrun turtlebot3_autorace_driving turtlebot3_autorace_control_lane.launch
... logging to /home/yongseok/.ros/log/3720a280-9339-11ee-b3dc-496b540f778a/roslaun[yongseok-10193.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started rosrun turtlebot3_autorace_driving turtlebot3_autorace_control_lane.launch
SUMMARY
=====
```

만일 실험을 마치려면, 모든 terminal은 CTL+C로 중단 후에 종료

Detect lane 결과 : 자율주행 진행 : 명령을 사용하면 잠시 후 터틀봇이 실행. 터틀봇을 멈추고 싶을 경우 Open CR 리셋 버튼



만일 실행이 되지 않으면, 원인 - turtlebot 위치가 바뀜
: Extrinsic camera calibration 수행 후, =action 으로

만일 실험을 마치려면, 모든 terminal은 CTL+C로 중단 후에 종료
Turtlebot은 sudo shutdown now로 종료

8-3. Lane detection 최종 실습 (터미널창 미리 8개 띄우기) – 방법2 (turtlebot3 정상 이동)

ⓐ Remote PC에서 새 터미널 열기

```
$ roscore
```

ⓑ 새 터미널 열기

```
$ ssh ubuntu@192.168.0.25      ← Remote PC에서 TurtleBot3 Burger(Raspberry pi)을 원격접속  
pw : turtlebot 입력  
~$ roslaunch turtlebot3_autorace_camera raspberry_pi_camera_publish.launch
```

ⓒ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_camera intrinsic_camera_calibration.launch mode:=action
```

ⓓ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_camera extrinsic_camera_calibration.launch mode:=action
```

ⓔ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_detect detect_lane.launch mode:=action
```

ⓕ Remote PC에서 새 터미널 열기

```
$ roslaunch turtlebot3_autorace_driving turtlebot3_autorace_control_lane.launch
```

ⓑ 새 터미널 열기

```
$ ssh ubuntu@192.168.0.25  
~$ roslaunch turtlebot3_bringup turtlebot3_robot.launch
```

감사합니다