6. 라이다

6-1. 라이다 토픽 읽어보기

1. /scan 토픽 확인

```
jungseong@jungseong-ubuntu:~$ ros2 topic list -t
/battery [std_msgs/msg/UInt16]
/beep [std_msgs/msg/UInt16]
/cmd_vel [geometry_msgs/msg/Twist]
/imu [sensor_msgs/msg/Imu]
/odom_raw [nav_msgs/msg/Odometry]
/parameter_events [rcl_interfaces/msg/ParameterEvent]
/rosout [rcl_interfaces/msg/Log]
/scan [sensor_msgs/msg/LaserScan]
/servo_s1 [std_msgs/msg/Int32]
/servo_s2 [std_msgs/msg/Int32]
```

2. sensor_msgs/msg/LaserScan 메세지 타입 확인

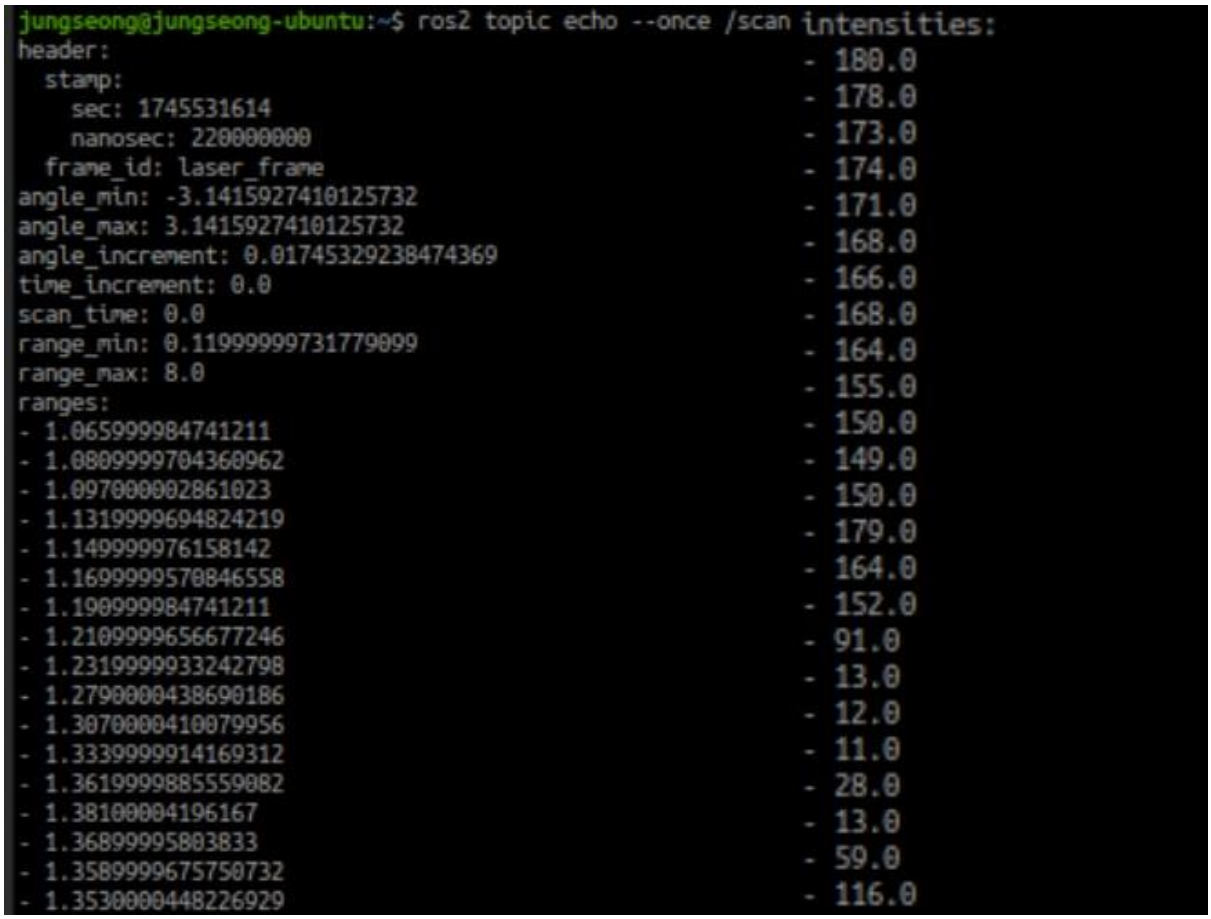
```
1. ros2 interface show sensor_msgs/msg/LaserScan
```

```
jungseong@jungseong-ubuntu:~$ ros2 interface show sensor_msgs/msg/LaserScan
# Single scan from a planar laser range-finder
#
# If you have another ranging device with different behavior (e.g. a sonar
# array), please find or create a different message, since applications
# will make fairly laser-specific assumptions about this data

std_msgs/Header header # timestamp in the header is the acquisition time of
        builtin_interfaces/Time stamp
                int32 sec
                uint32 nanosec
        string frame_id
                        # the first ray in the scan.
                        #
                        # in frame frame_id, angles are measured around
                        # the positive Z axis (counterclockwise, if Z is up)
                        # with zero angle being forward along the x axis

float32 angle_min           # start angle of the scan [rad]
float32 angle_max           # end angle of the scan [rad]
float32 angle_increment     # angular distance between measurements [rad]

float32 time_increment      # time between measurements [seconds] - if your scanner
                            # is moving, this will be used in interpolating position
                            # of 3d points
float32 scan_time           # time between scans [seconds]

float32 range_min           # minimum range value [m]
float32 range_max           # maximum range value [m]

float32[] ranges            # range data [m]
                            # (Note: values < range_min or > range_max should be discarded)
float32[] intensities       # intensity data [device-specific units].  If your
                            # device does not provide intensities, please leave
                            # the array empty.
```

라이다 전용으로 만든 메세지 타입으로 보인다

3. /scan 토픽 실행

```
1. ros2 topoic echo --once /scan
```



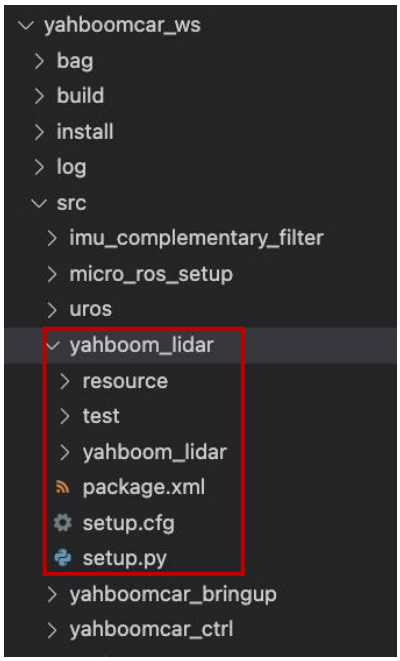
토픽이 잘 출력되는 것을 확인할 수 있다

6-2. 시각화를 위한 yahboomcar_lidar 패키지 만들기

1. yahboomcar_lidar 패키지 만들기

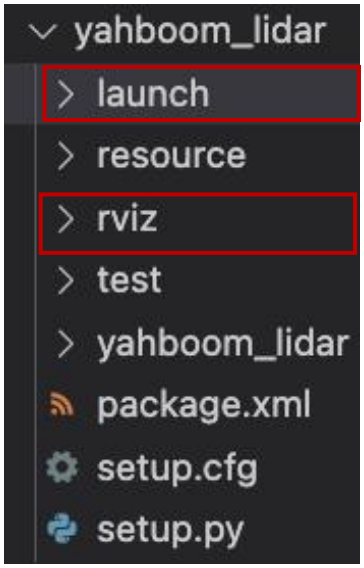
```
1. cd ~/yahboomcar_ws/src # 사용자 작업 폴더 들어가서
2. ros2 pkg create --build-type ament_python yahboomcar_lidar #
ament_python 빌드 시스템을 사용하는 yahboomcar_lidar 이라는 이름의 패키지
생성
```

yahboomcar_lidar 패키지가 잘 생성되는 것을 확인할 수 있다



2. launch, rviz 폴더 만들기

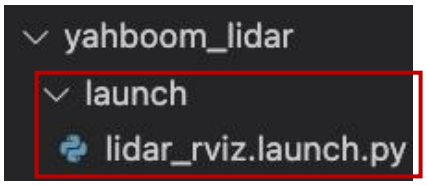
지난번 imu 시각화와 동일하게 **rviz2를 실행할 런치 파일이 필요하기 때문**에 yahboomcar_lidar 폴더 아래에 launch, rviz 폴더를 생성해준다



3. lidar_rviz.launch.py라는 이름의 런치 파일 만들기

시각화를 위해 TF가 필요하며, 복수의 노드를 실행하기 위해 launch 파일을 제작한다

lidar_rviz.launch.py라는 이름의 런치 파일을 제작했다.



rviz_config_dir : rpidar.rviz이라는 이름의 rviz 파일이 저장된 경로

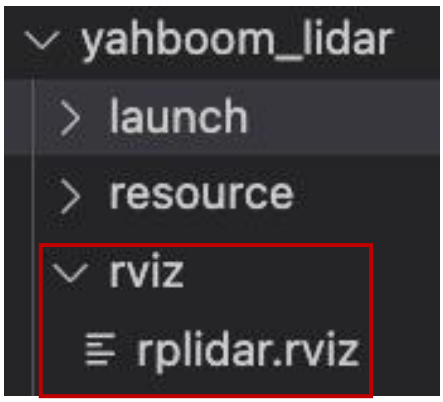
TF를 위해 base_link_to_laser_tf_node와 laser_link_to_laser_frame_tf_node를 만들어 주고, 런치 파일을 통해 rviz_node까지 세 노드를 동시해 실행한다

```
import os

from ament_index_python.packages import get_package_share_directory
from launch import LaunchDescription
from launch_ros.actions import Node

def generate_launch_description():
    rviz_config_dir = os.path.join(
    get_package_share_directory('yahboom_lidar'),
    'rviz',
    'rplidar.rviz')

    rviz_node = Node(
    package='rviz2',
    executable='rviz2',
    name='rviz2',
    arguments=['-d', rviz_config_dir],
    output='screen'
    )

    # base_link → laser_link 트랜스폼 (lidar 의 위치 설정)
    base_link_to_laser_tf_node = Node(
    package='tf2_ros',
    executable='static_transform_publisher',
    name='base_link_to_base_laser',
    arguments=['-0.0046412', '0' , '0.094079',
    '0','0','0','1',
```

```
    'base_link','laser_link']
    )

    # imu_link → imu_frame 트랜스폼 (lidar의 회전 처리)
    laser_link_to_laser_frame_tf_node = Node(
    package='tf2_ros',
    executable='static_transform_publisher',
    name='imu_link_to_imu_frame',
    arguments=['0', '0', '0',
    '0', '0', '0', '1',
    'laser_link', 'laser_frame']
    )

    return LaunchDescription([
    base_link_to_laser_tf_node,
    laser_link_to_laser_frame_tf_node,
    rviz_node
 ])
```

```
런치 프로그래밍 관련

 1. - 'package'는 실행할 패키지 이름을 기재하면 된다.
 2. - 'executable'은 실행 가능한 노드의 이름을 적어주면 된다.
 3. - 'name'에는 지정한 노드를 실행할 때 실제로 사용할 이름을 기재하면 된다. 보통은
executable에 기재한 본래의 노드 이름을 적어주면 되는데 필요에 의해 다른 이름으로 설정
가능하다.
 4. - 'argument'에는 해당 실행 파일에 전달되는 명령줄 인자를 의미한다
 5. - 'remapping'은 특정 이름을 변경해 줄 수 있다
 6. - 'parameters'는 특정 파라미터 값을 넣어줘도 되고 'DeclareLaunchArgument'에서
지정한 'param_dir'와 같은 선언 값을 사용해도 된다. 여기서는 'param_dir' 을 입력값으로
넣어주었기에 지정된 'arithmetic_config.yaml' 파라미터 파일을 사용하게 된다.
 7. - 'output'은 로깅 설정으로 기본이면 특정 파일 이름(~/.ros/log/xxx/launch.log)에
로깅 정보가 기록되고 터미널창에도 표시해주고 싶다면 'screen' 이라고 지정하면 된다.
 8.
 9. [출처] 040 런치 프로그래밍 (Python, C++) (오픈소스 소프트웨어 & 하드웨어: 로봇 기술
공유 카페 (오로카)) | 작성자 표윤석
10.
```

4. rplidar.rviz 추가하기



5. setup.py 내용 변경

중요!! ROS2 패키지 파일 설명

```
1. package.xml : 패키지의 정보를 기술하는 파일이다. 패키지 이름, 저작자, 라이선스, 의존성 패키지… 등등이 기술되어 있다. 의존성 패키지들이 모두 기술되기에 빌드, 패키지 설치, 사용에 있어서 매우 중요한 파일이다. 모든 ROS 패키지의 필수 파일로 각 패키지 당 무조건 1 개의 패키지 설정 파일을 포함한다.
2. CMakeLists.txt : C++에서 ROS 패키지를 빌드하는 방법을 정의하는 파일이다. 이 빌드 설정 파일에 실행 파일 생성, 의존성 패키지 우선 빌드... 등등을 설정할 수 있다.
3. setup.py : 파이썬 패키지 설정 파일
4. setup.cfg : 파이썬 패키지 환경 설정 파일, ROS2 Python 패키지에서만 사용하는 배포를 위한 구성 파일
5. plugin.xml : RQT 플러그인으로 패키지를 작성할 때의 필수 구성요소.
6. CHANGELOG.rst : 패키지의 업데이트 내역을 기술하는 파일
7. LICENSE : 라이선스를 기술하는 파일
8. README.md : 패키지의 부가 설명을 기술하는 파일
```

그 중 setup.py

```
 1. from setuptools import find_packages, setup
 2.
 3. package_name = 'yahboom_lidar'
 4.
 5. setup(
 6.     name=package_name,
 7.     version='0.0.0',
 8.     packages=find_packages(exclude=['test']),
 9.     data_files=[
10.         ('share/ament_index/resource_index/packages',
11.             ['resource/' + package_name]),
12.         ('share/' + package_name, ['package.xml']),
13.     ],
14.     install_requires=['setuptools'],
15.     zip_safe=True,
16.     maintainer='jungseong',
17.     maintainer_email='jungseonglian@sju.ac.kr',
18.     description='TODO: Package description',
19.     license='TODO: License declaration',
20.     tests_require=['pytest'],
21.     entry_points={
22.         'console_scripts': [
23.         ],
24.     },
25. )
```

data_files

- 패키지를 빌드 시 함께 사용되는 파일들, **추가되는 파일이 있으면 변경한다**

- `ROS`에서는 주로 `resource` 폴더 내에 있는 `ament_index`를 위한 패키지의 이름의 빈 파일이나 `package.xml`, `*.launch.py`, `*.yaml` 등을 기입한다.

우리는 기존의 yahboomcar_lidar 패키지에

rviz/rpidar.rviz

launch/lidar_rviz.launch.py

파일을 새롭게 넣었기 때문에, data_files에 해당 파일들도 찾을 수 있도록 경로를 추가해야 한다.

```
from setuptools import find_packages, setup
import os
from glob import glob

package_name = 'yahboom_lidar'

setup(
    name=package_name,
    version='0.0.0',
    packages=find_packages(exclude=['test']),
    data_files=[
        ('share/ament_index/resource_index/packages',
            ['resource/' + package_name]),
        ('share/' + package_name, ['package.xml']),
        (os.path.join('share',package_name,'rviz'), glob(os.path.join('rviz', '*.rviz'))),
        (os.path.join('share',package_name,'launch'), glob(os.path.join('launch', '*launch.py'))),
    ],
    install_requires=['setuptools'],
    zip_safe=True,
    maintainer='jungseong',
    maintainer_email='jungseonglian@sju.ac.kr',
    description='TODO: Package description',
    license='TODO: License declaration',
    tests_require=['pytest'],
    entry_points={
        'console_scripts': [
        ],
    },
)
```

glob.glob : 사용자가 제시한 조건에 맞는 파일명을 리스트 형식으로 반환, '*', '?' 같은 와일드 카드를 지원한다.

6. yahboomcar_lidar 패키지 빌드

정상적으로 빌드되는 것을 확인할 수 있다.

```
jungseong@jungseong-ubuntu:~/yahboomcar_ws$ colcon build --pac
kages-select yahboom_lidar
Starting >>> yahboom_lidar
Finished <<< yahboom_lidar [1.81s]

Summary: 1 package finished [2.28s]
```

7. lidar_rviz.launch.py 파일 실행

```
jungseong@jungseong-ubuntu:~/yahboomcar_ws$ shs
humble/setup.bash loaded
jungseong@jungseong-ubuntu:~/yahboomcar_ws$ lsb
install/local_setup.bash loaded
jungseong@jungseong-ubuntu:~/yahboomcar_ws$ ros2 launch yahboom_lidar lidar_rviz.launch.py
[INFO] [launch]: All log files can be found below /home/jungseong/.ros/log/2025-04-26-10-0
0-09-235737-jungseong-ubuntu-6442
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [static_transform_publisher-1]: process started with pid [6443]
[INFO] [static_transform_publisher-2]: process started with pid [6445]
[INFO] [rviz2-3]: process started with pid [6447]
[static_transform_publisher-1] [WARN] [1745629209.310841806] []: Old-style arguments are d
eprecated; see --help for new-style arguments
[static_transform_publisher-2] [WARN] [1745629209.310847673] []: Old-style arguments are d
eprecated; see --help for new-style arguments
[rviz2-3] Warning: Ignoring XDG_SESSION_TYPE=wayland on Gnome. Use QT_QPA_PLATFORM=wayland
 to run on Wayland anyway.
[static_transform_publisher-1] [INFO] [1745629209.467428475] [base_link_to_base_laser]: Sp
inning until stopped - publishing transform
[static_transform_publisher-1] translation: ('-0.004641', '0.000000', '0.094079')
[static_transform_publisher-1] rotation: ('0.000000', '0.000000', '0.000000', '1.000000')
[static_transform_publisher-1] from 'base_link' to 'laser_link'
[static_transform_publisher-2] [INFO] [1745629209.475133200] [imu_link_to_imu_frame]: Spin
ning until stopped - publishing transform
[static_transform_publisher-2] translation: ('0.000000', '0.000000', '0.000000')
[static_transform_publisher-2] rotation: ('0.000000', '0.000000', '0.000000', '1.000000')
[static_transform_publisher-2] from 'laser_link' to 'laser_frame'
[rviz2-3] [INFO] [1745629209.880481647] [rviz2]: Stereo is NOT SUPPORTED
[rviz2-3] [INFO] [1745629209.880809614] [rviz2]: OpenGl version: 4.5 (GLSL 4.5)
[rviz2-3] [INFO] [1745629209.968328645] [rviz2]: Stereo is NOT SUPPORTED
```

8. 결과

라이다 포인트 클라우드가 잘 찍힌 것을 확인할 수 있다.

2D 라이다기 때문에 z축 값 정보는 알 수 없다.