화성 탐사 로봇 시뮬레이터

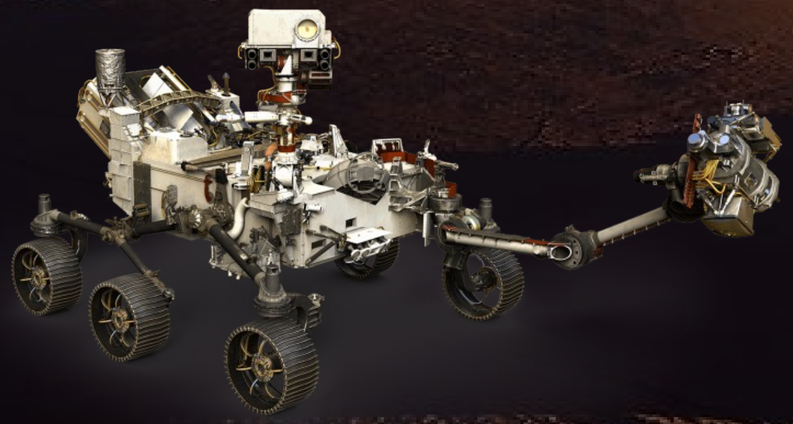
C-1 김교희 김유찬 박병주 박영건 오동규

Instructions

1. 시나리오
2. Rover , Drone 소개
3. 문제점
4. 해결 방법
5. 추가 개선 사항

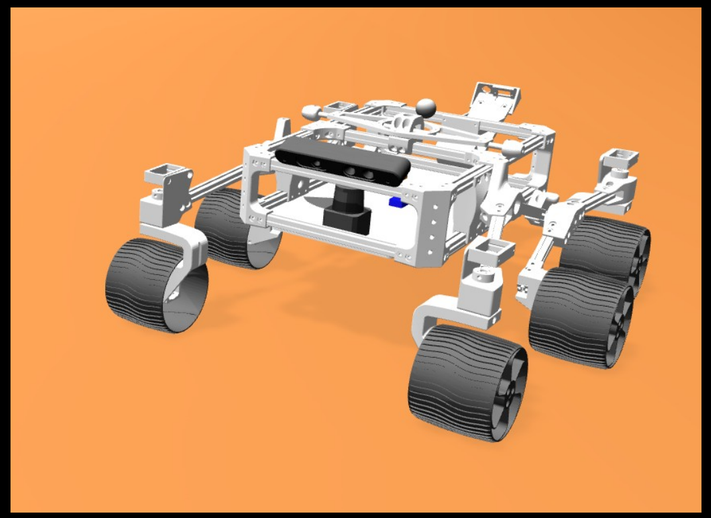
시나리오

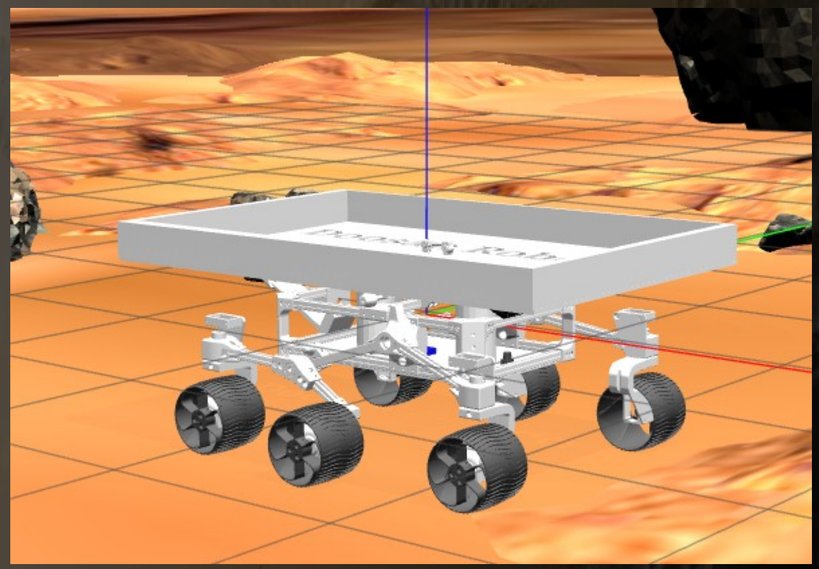
탐사 시나리오



ROVER

Git 에 공유된 큐리오시티의 Source 를 사용
-> URDF 수정
-> 드론 이착륙 공간 확보
-> Rover 크기 확대





ROVER

● mesh stl 파일 수정 한계

=> urdf.xacro scale 값 수정

=> joint x,y,z 좌표 수정 필요

=> visual, collision 수정 필요

=> spawn 위치 수정 필요

```
<robot name="body" xmlns:xacro="http://www.ros.org/wiki/xacro">
    <xacro:property name="body_box_length" value="1.29" />
    <xacro:property name="body_box_width" value="0.87" />
    <xacro:property name="body_box_depth" value="0.30" />
    <xacro:property name="body_box_mass" value="1.0" />
    <xacro:property name="scale_factor" value="3.0"/>
```

```
    <origin rpy="0 0 0" xyz="0 0 0" />
    <geometry>
        <mesh filename="package://rover_description/meshes/bases/body_box.stl"
        scale="${scale_factor} ${scale_factor} ${scale_factor}"/>
        <!-- <box size="${body_box_length} ${body_box_width} ${body_box_depth}"/> -->
    </geometry>
    <material name="white" />
```

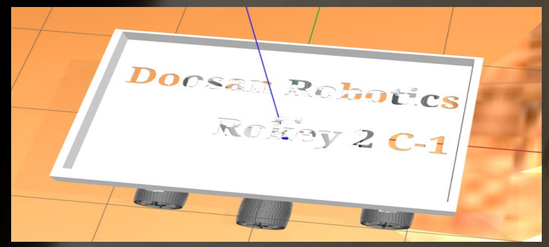
```
<joint name="roof_joint" type="fixed">
    <origin rpy="0 0 0" xyz="0 0 -0.15"/>
    <parent link="${parent}"/>
    <child link="roof"/>
</joint>
```

ROVER

● 드론 착지 , 보관

1. Rover .stl 크기 파악
2. stl 제작

3. urdf.xacro 파일 생성
4. 크기 고려 joint, parent 설정
5. 상호작용을 위한 물성치 부여




```
<joint name="roof_joint" type="fixed">
    <origin rpy="0 0 0" xyz="0 0 -0.15"/>
    <parent link="${parent}"/>
    <child link="roof"/>
</joint>
```

```
    </visual>
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0"/>
        <geometry>
            <mesh filename="package://rover_description/meshes/bases/roof.stl" scale="1 1 1"/>
        </geometry>
    </collision>
    <xacro:solid_cuboid_inertial
        rpy="0 0 0" xyz="0 0 0"
        mass="${roof_mass}"
        x="${roof_length}" y="${roof_width}" z="${roof_depth}" />
</link>
```

Drone

- Take off 시 위치 저장
- 비행 좌표 전송 -> 좌표에 도달할 때까지 비행
- 좌표 도착 후 Hovering
- Return to Home 전송 -> Take off 한 장소로 이동
- Landing

Drone

```python
class DronePositionControl(DroneObject):
    def __init__(self):
        super().__init__('drone_position_control')

        self.callback_group = ReentrantCallbackGroup()

        self.dt = 0.28
        self.kp = 1.0
        self.kd = 5.0
        self.ki = 1.0

        self.error_prev_x = 0.0
        self.error_prev_y = 0.0
        self.error_prev_z = 0.0

        self.state = 'ready'
        self.key = ''
        self.home = [0.0, 0.0, 0.0]
        self.goal = [0.0, 0.0, 0.0]


        self.count = 0
        self.time = []
        self.error = []
        # super().takeOff()
        # self.get_logger().info('Drone takeoff')

        self.pid_pub = self.create_publisher(Float32MultiArray,'/pid',10)

        # # Set the m_posCtrl flag to True
        self.posCtrl(True)
        # self.get_logger().info('Position control mode set to True')
        print('start')
        self.sub = self.create_subscription(Odometry, '/simple_drone/odom', self.callback, 10, callback_group=self.callback_group)
        # self.timer = self.create_timer(1,self.timer_callback)
```

Drone

```python
def callback(self,data):
    global user
    if len(user) == 1:
        if user == 't':
            self.state = 'takeoff'
        elif user == 'h':
            self.state = 'home'
    elif len(user) > 1:
        pos = user.split(' ')
        self.goal = [float(pos[0]), float(pos[1]), float(pos[2])]
        self.state = 'search'


    user = ''

    # 현재 위치 파악
    current_x = data.pose.pose.position.x
    current_y = data.pose.pose.position.y
    current_z = data.pose.pose.position.z

    # takeoff
    if self.state == 'takeoff':
        self.home = [current_x, current_y, current_z]
        self.takeoff()
        # self.state = 'search'

    # search
    elif self.state == 'search':
        x,y,z = self.home
        gx, gy, gz = self.goal

        x += gx
        y += gy
        z += gz

        error_x = x-current_x
        error_y = y-current_y
        error_z = z-current_z

        vel_x, vel_y, vel_z = self.pid(error_x, error_y, error_z)

        self.count += self.dt
        self.time.append(self.count)
        self.error.append(current_x)
```

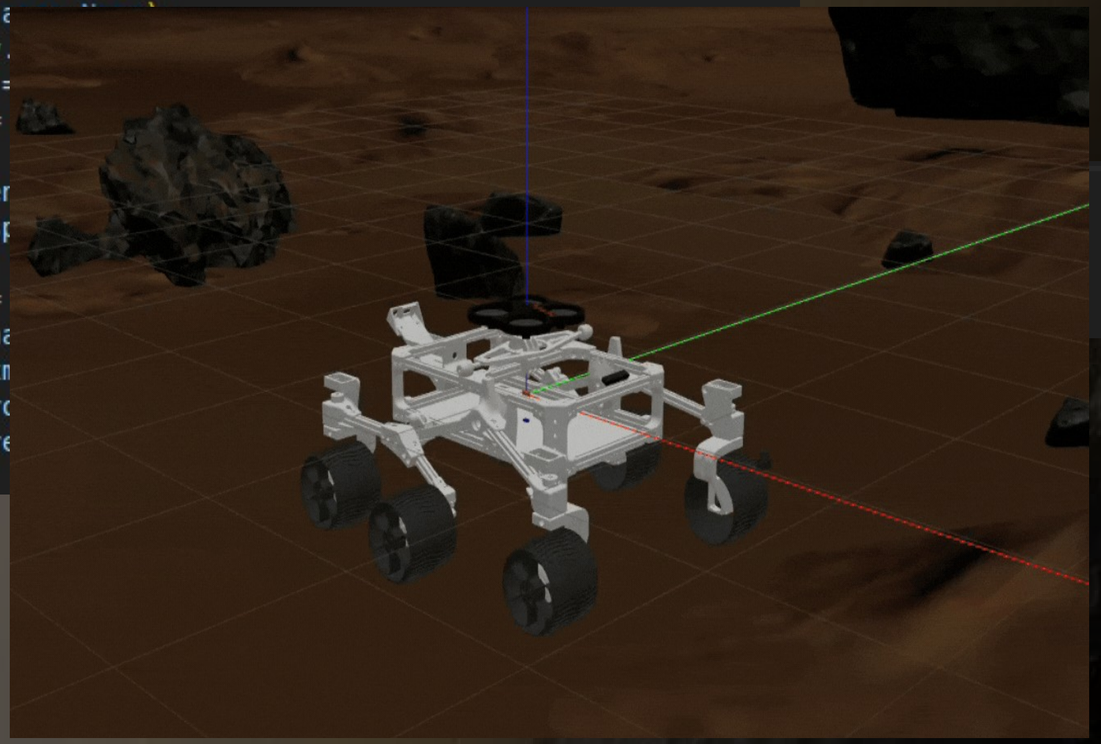
```python
            if abs(error_x) < 0.1 and abs(error_y) < 0.1 and abs(error_z) < 0.1:
                print('pid')

                self.Hover()
                # self.state = 'home'
            else:
                print('-----------moving--------')
                self.move_drone_to_pose(vel_x, vel_y, vel_z)

        # back home
        elif self.state == 'home':
            x, y, z = self.home
            z += 0.5

            error_x = x-current_x
            error_y = y-current_y
            error_z = z-current_z

            vel_x, vel_y, vel_z = self.pid(error_x, error_y, error_z)

            if abs(error_x) < 0.1 and abs(error_y) < 0.1 and abs(error_z) <0.2:
                self.Hover()
                self.state = 'land'
            else:
                print('-----------home--------')
                self.move_drone_to_pose(vel_x, vel_y, vel_z)

        # land
        elif self.state == 'land':
            super().land()
            self.state = 'ready'

    def takeoff(self):
        super().takeOff()

    def Hover(self):
        super().hover()

    def move_drone_to_pose(self, x, y, z):
        # Override the move_drone_to_pose method if specific behavior is needed
        super().moveTo(x, y, z)
        self.get_logger().info(f'Moving drone to pose: x={x}, y={y}, z={z}')
```

문제

Object	문제	원인 분석
Rover	Rover 크기 수정 실패	stl 파일 수정의 어려움
	Rover 에 Drone 스테이션 설치 실패	meshes 파일 병합
	Gazebo 상에서 rover 가 튕김	spawn 시 map 과 겹침
Rover + Drone	Rover odom 오류	추정 실패
	Drone, Rover 의 TF 분리	서로 TF 연결이 다름
	Drone 과 Rover 동시 spawn	코드 구현 방법 차이
Drone	Drone 명령 오류	코드 오류

해결

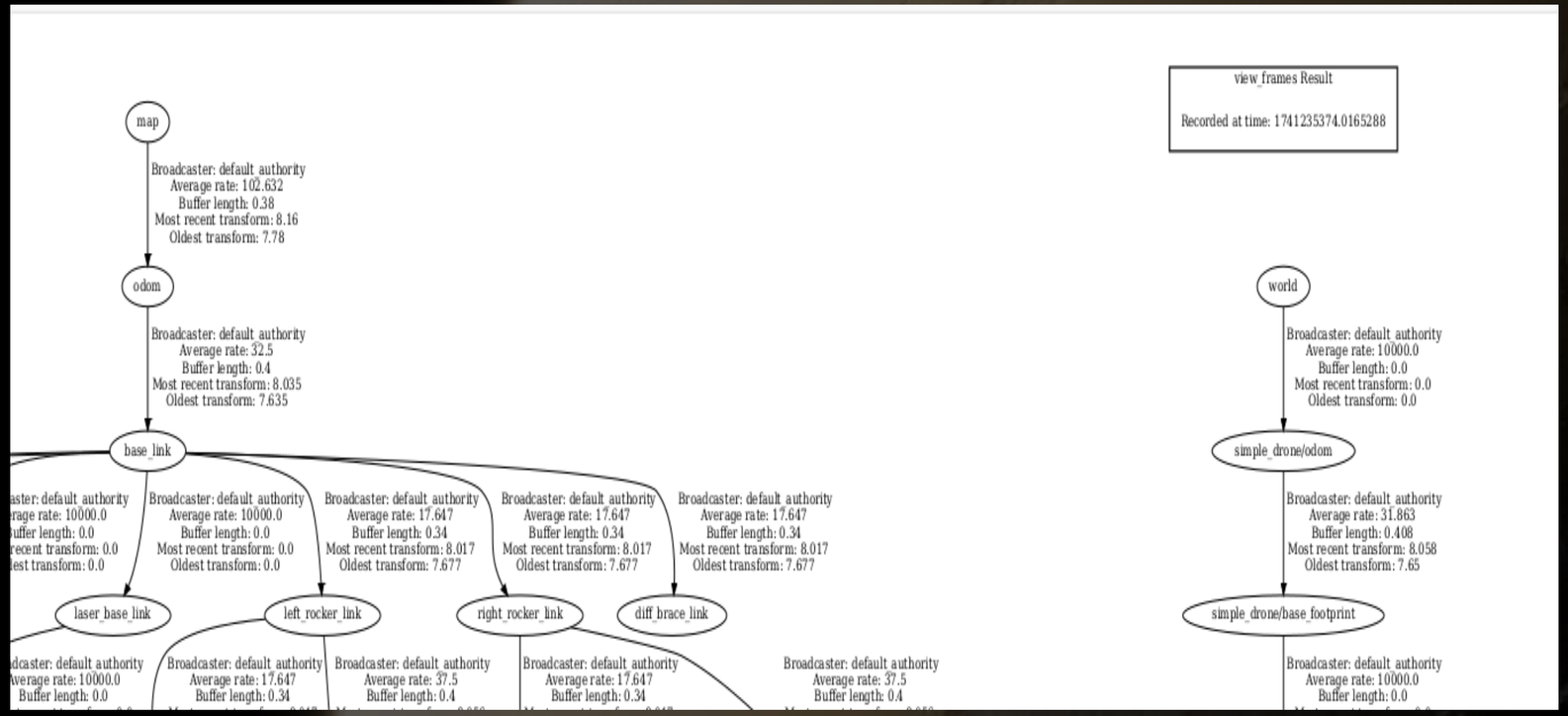
sjtu_drone_bringup/sjtu_drone_bringup/
spawn_drone.py



Spa
Error

해결

TF 문제


view_frames Result
Recorded at time: 1741235374.0165288
map
Broadcaster: default_authority
Average rate: 102.632
Buffer length: 0.38
Most recent transform: 8.16
Oldest transform: 7.78
odom
Broadcaster: default_authority
Average rate: 32.5
Buffer length: 0.4
Most recent transform: 8.035
Oldest transform: 7.635
base_link
Broadcaster: default_authority
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
Broadcaster: default_authority
Average rate: 17.647
Buffer length: 0.34
Most recent transform: 8.017
Oldest transform: 7.677
Broadcaster: default_authority
Average rate: 17.647
Buffer length: 0.34
Most recent transform: 8.017
Oldest transform: 7.677
Broadcaster: default_authority
Average rate: 17.647
Buffer length: 0.34
Most recent transform: 8.017
Oldest transform: 7.677
laser_base_link
left_rocker_link
right_rocker_link
diff_brace_link
Broadcaster: default_authority
Average rate: 17.647
Buffer length: 0.34
Broadcaster: default_authority
Average rate: 37.5
Buffer length: 0.4
Broadcaster: default_authority
Average rate: 37.5
Buffer length: 0.4
world
Broadcaster: default_authority
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
simple_drone/odom
Broadcaster: default_authority
Average rate: 31.863
Buffer length: 0.408
Most recent transform: 8.058
Oldest transform: 7.65
simple_drone/base_footprint
Broadcaster: default_authority
Average rate: 10000.0
Buffer length: 0.0

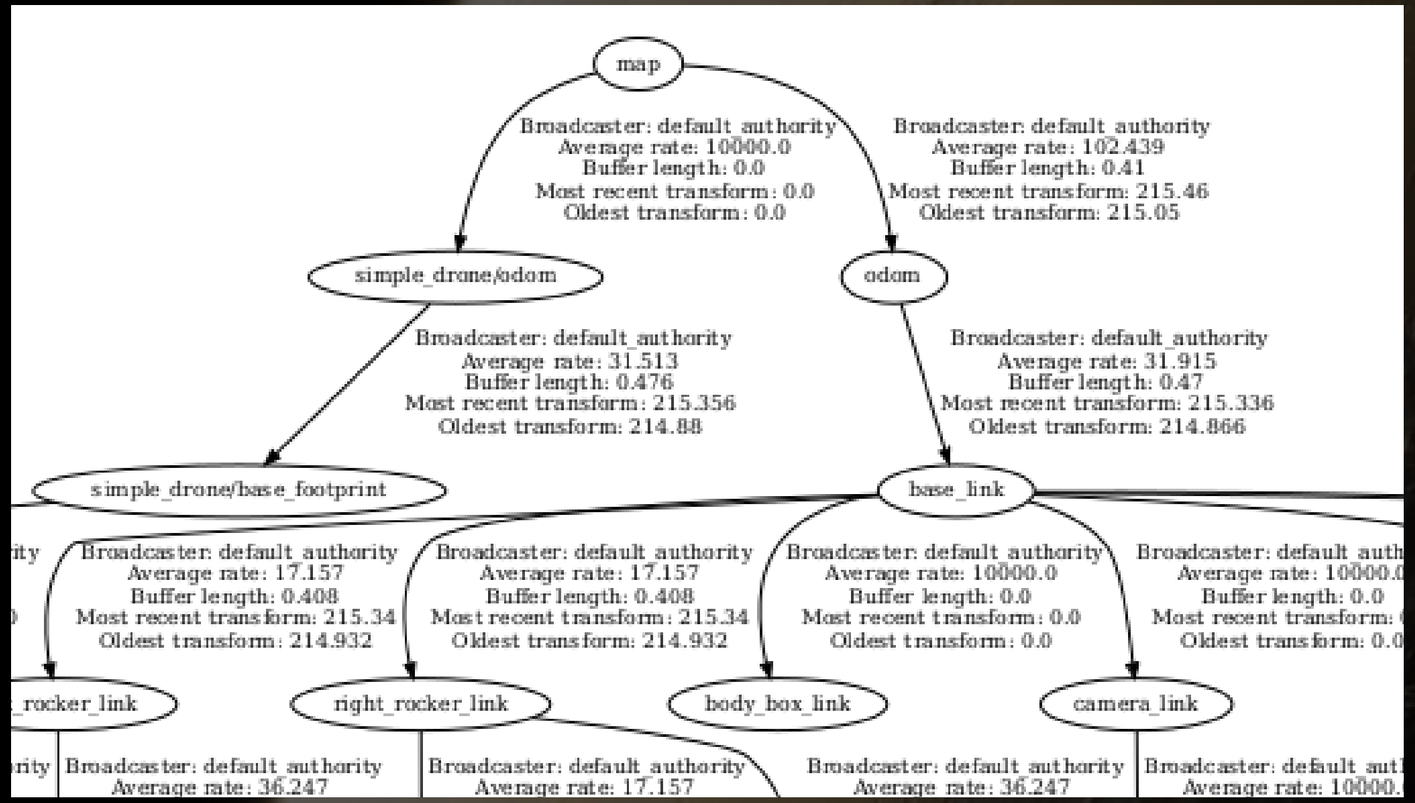
해결

TF 문제

```
Node(
    package="tf2_ros",
    executable="static_transform_publisher",
    arguments=["0", "0", "0", "0", "0", "0", "world", f"{model
    output="screen"
),
```

```
Node(
    package="tf2_ros",
    executable="static_transform_publisher",
    arguments=["0", "0", "0", "0", "0", "0", "map", f"
    output="screen"
),
```

sjtu_drone_gazebo.launch.py



해결

spawn 구조 문제

```
### NODES ###
spawn_entity_cmd = Node(
    package="gazebo_ros",
    executable="spawn_entity.py",
    arguments=[
        "-entity",
        "rover",
        "-topic",
        "robot_description",
        "-timeout",
        "120",
        "-x",
        initial_pose_x,
        "-y",
        initial_pose_y,
        "-z",
        initial_pose_z,
        "-Y",
        initial_pose_yaw,
    ],
    output="screen",
    parameters=[{"use_sim_time": True}],
)
```

rover - gazebo.launch -> spawn.launch
내부에서
gazebo_ros pkg 의 spawn_entity.py 사용

해결

spawn 구조 문제

drone - drone_gazebo.launch
-> spawn_drone.py 를 실행

```python
# -*- coding: utf-8 -*-
import sys
import rclpy
from gazebo_msgs.srv import SpawnEntity


def main(args=None):
    rclpy.init(args=args)
    node = rclpy.create_node('spawn_drone')
    cli = node.create_client(SpawnEntity, '/spawn_entity')

    content = sys.argv[1]
    namespace = sys.argv[2]

    req = SpawnEntity.Request()
    req.name = namespace
    req.xml = content
    req.robot_namespace = namespace
    req.reference_frame = "world"

    while not cli.wait_for_service(timeout_sec=1.0):
        node.get_logger().info('service not available, waiting again...')

    future = cli.call_async(req)
    rclpy.spin_until_future_complete(node, future)

    if future.result() is not None:
        node.get_logger().info(
            'Result ' + str(future.result().success) + " " + future.result().status_message)
    else:
        node.get_logger().info('Service call failed %r' % (future.exception(),))

    node.destroy_node()
    rclpy.shutdown()


if __name__ == '__main__':
    main()
```

해결

spawn 구조 문제

rover gazebo.launch 내부에
sjtu_drone_gazebo.launch
병합

```
########
# drone
########
ld.add_action(
    DeclareLaunchArgument(
        "controller",
        default_value="keyboard",
        description="Type of controller: keyboard (default) or joystick",
    ))

ld.add_action(Node(
        package="rviz2",
        executable="rviz2",
        name="rviz2",
        arguments=[
            "-d", rviz_path
        ],
        output="screen",
    ))

ld.add_action(IncludeLaunchDescription(
        PythonLaunchDescriptionSource(
            os.path.join(sjtu_drone_bringup_path, 'launch', 'sjtu_drone_gazebo.launch.py')
        )
    ))

ld.add_action(Node(
        package='joy',
        executable='joy_node',
        name='joy',
        namespace=model_ns,
        output='screen',
    ))

ld.add_action(OpaqueFunction(
        function=get_teleop_controller,
        kwargs={'model_ns': model_ns},
    ))



return ld
```

해결

drone 명령 문제

```
lass DronePositionControl(DroneObject):
    def __init__(self):
        super().__init__('drone_position_control')

        self.takeOff()
        self.get_logger().info('Drone takeoff')

        # Set the m_posCtrl flag to True
```

```
def takeoff(self):
    super().takeOff()
```

추가사항

PID

- 오버슈팅 크기 감소
- 세들링타임 감소 -> 튜닝 필요

추가 개선 사항

Drone Take off 후

Rover 가 움직일 경우 ,

Drone 이 Rover 의 위치를 인식하도록 개선

3 차원 맵핑을 통한 탐색

Drone 과 Rover 의 3 차원맵 병합

Drone 비행경로 개선

여분의 Drone 추가

THANK YOU
FOR LISTENING!

화성 갈끄니끄아
~